Computation of three-dimensional standing water waves

Chris H. Rycroft[a,b,c,d], Jon Wilkening[b,c]

[a]*Department of Physics, University of California, Berkeley, CA 94720, United States*
[b]*Department of Mathematics, University of California, Berkeley, CA 94720, United States*
[c]*Department of Mathematics, Lawrence Berkeley Laboratory, Berkeley, CA 94720, United States*
[d]*School of Engineering and Applied Sciences, Harvard University, MA 02138, United States*

Abstract

We develop a method for computing three-dimensional gravity-driven water waves, which we use to search for time-periodic standing wave solutions. We simulate an inviscid, irrotational, incompressible fluid bounded below by a flat wall, and above by an evolving free surface. The computations make use of spectral derivatives on the surface, but also require computing a velocity potential in the bulk, which we carry out using a finite element method with fourth-order elements that are curved to match the free surface. This computationally expensive step is solved using a parallel multigrid algorithm, which is discussed in detail. Time-periodic solutions are searched for using a previously developed overdetermined shooting method. Several families of large-amplitude three-dimensional standing waves are found in both shallow and deep regimes, and their physical characteristics are examined and compared to previously known two-dimensional solutions.

Keywords: optimization, multigrid methods, water waves

1. Introduction

Gravity-driven water waves have been studied for well over a century and have a rich mathematical structure arising from their nonlinearity. Effects such as resonances [1, 2] can have important consequences in ocean engineering, and may be exploited in the design of maritime structures [3]. One method of investigating the properties of water waves has been to search for special solutions of the free-surface Euler equations for an inviscid, incompressible fluid such as traveling and standing waves. One of the earliest examples of this is due to Stokes, who in 1880 postulated that the traveling wave of maximum height has a crest with an internal angle of 120°. This was later investigated numerically [4, 5], and proved analytically [6]. The self-similar asymptotics of the almost-highest traveling wave has also been investigated [7, 8, 9].

A similar proposition was given for standing waves by Penney and Price in 1952 [10]. By considering several terms in a perturbation expansion, they proposed that the largest amplitude standing wave would form a sharp crest with an internal angle of 90°. This prediction was in reasonable agreement with experiments carried out by Taylor [11]. However, subsequent analytical and numerical studies have reached a variety of different conclusions concerning the precise form of

Email addresses: chr@seas.harvard.edu (Chris H. Rycroft), wilken@math.berkeley.edu (Jon Wilkening)

the geometric singularity the limiting extreme wave should possess [12, 13, 14, 15, 16, 17, 18, 19]. Recently, Wilkening [20] has shown that at higher resolutions, the self-similar sharpening of the crest eventually breaks down, and several families of time-periodic solutions can be found featuring small-scale oscillations near the crest. This casts doubt on the assumption that a limiting wave profile exists at all, much less one with 90° crests.

All of the aforementioned studies consider two-dimensional (2D) fluids with one-dimensional surfaces, and it is natural to ask how these results may generalize to three dimensions. However, the investigation of three-dimensional standing waves has been comparatively limited. Verma and Keller [21] and Bridges [22] carried out calculations of small-amplitude waves using perturbation expansions. They were able to examine bifurcations in the families of solutions, and to determine how the periods of standing waves vary as a function of amplitude. More recently Bryant and Stiassnie [23] and Zhu et al. [24] have investigated questions of three-dimensional wave stability and evolution while Engsig-Karup et al. [25] have developed a large-scale parallel code for solving the nonlinear evolution of free-surface waves using a finite difference framework. None of these works attempt a computation of large-amplitude three-dimensional standing waves.

Searching for three-dimensional standing water waves offers a number of technical challenges. Some methods used to search for two-dimensional standing waves, such as conformal mapping methods [26, 27, 28, 29, 30], do not have a generalization to three dimensions. Furthermore, calculating three-dimensional standing waves requires significantly more computational resources. Simulating the wave itself requires one more dimension, and the number of degrees of freedom parameterizing the configuration space over which to search is also larger.

In this paper, we take advantage of improvements in computational power and new algorithms to calculate time-periodic, three-dimensional gravity-driven waves in an incompressible, inviscid fluid. To search for time-periodic solutions, we make use of a recently developed methodology where the problem is framed as an overdetermined nonlinear system and a minimization technique is employed to search a configuration space for solutions that are progressively closer to being time-periodic. This approach has been used to find time-periodic solutions of the Benjamin–Ono equation [31, 32], the vortex sheet with surface tension [33], and two-dimensional standing water waves [34]. Different minimization methods have been employed, such as an adjoint-based BFGS approach [35], but here we make use of a variant of the trust-region based Levenberg–Marquardt minimization. This technique requires computing an entire Jacobian: at a given wave configuration, it is necessary to determine how the time-periodicity will change in each direction of the configuration space. While this is expensive to calculate, the minimization requires far fewer iterations than the BFGS approach, and is more amenable to parallelization. Newton–Krylov methods [36, 37] would be an interesting alternative to explore; however, these methods generally work better in externally driven, dissipative systems [38, 39, 40] in which viscosity damps high-frequency oscillations as the solution evolves.

Computation of irrotational water waves requires time-integrating the height of the free surface and a velocity potential on the surface. However, at each step, it is also necessary to solve the Laplace equation in the three-dimensional bulk of the fluid. We have developed a fourth-order finite element discretization of the fluid domain for this purpose, where the order is measured with respect to the H^1 Sobolev norm. The use of finite elements is not typical, and for two-dimensional studies, boundary integral methods are more common. However in three dimensions, an argument can be

made that a finite element discretization is more suitable, since it requires solving an $O(N^3)$ sparse linear system, as opposed to an $O(N^2)$ dense linear system for a boundary integral approach (where N is a typical number of grid points in one dimension). This difference in complexity is more favorable in two dimensions, where the comparison would be between an $O(N^2)$ sparse system and an $O(N)$ dense system. Naturally, fast algorithms can be used to reduce the computational cost of the boundary integral approach, but the prefactors are currently very large in three-dimensional implementations of these algorithms [41].

Solving the finite element problem is the most computationally intensive part of our fluid solver. To carry this out, we have developed a parallel geometric multigrid algorithm, which is presented in detail below. The multigrid algorithm can also compute solutions for several right-hand sides concurrently, and due to memory bandwidth considerations, this can be carried out in a fraction of the time required to compute each solution sequentially. This feature is exploited in the computation of the Jacobian needed in the Levenberg–Marquardt minimization.

In this paper, we present several families of time-periodic solutions that we have found using this methodology. We examine waves in two depth regimes, one relatively shallow (with fluid depth equal to $1/12$ the wavelength) and one relatively deep (with fluid depth equal to $1/2$ the wavelength). The distinction boils down to whether $\tanh kh$ is close to 1, where k is the wave number and h is the depth. Given the difficulties of computation, whereby calculating a single time-periodic solution can take several days using sixteen threads, the numerical results we present are of relatively low resolution when compared to two-dimensional studies, and we exploit a large amount of symmetry in order to reduce the dimension of the configuration space that must be searched. Since little is known about three-dimensional standing waves, our main aim in this paper is to examine their physical characteristics and compare them to two-dimensional solutions, paying particular attention to ways in which there may be significant differences. In three dimensions, we are also able to ask fundamentally new questions about wave morphology that would not be applicable in two dimensions. Our results cover only a very small part of the possible range of three-dimensional time-periodic solutions that may exist, but serve to highlight some interesting questions for further study.

2. Methods

2.1. Governing equations

We make use of an (x,y,z) coordinate system that is periodic in the horizontal x and y directions, and where gravity g points in the negative z direction. We employ non-dimensionalized units, where $g = 1$ and the horizontal coordinates cover the range $[0, 2\pi)$. The fluid is bounded below by a flat base at $z = 0$, and has a free surface given by $z = \eta(x,y,t)$. The fluid is inviscid, irrotational, and incompressible, and its velocity can therefore be given as $\mathbf{v} = \nabla\phi$ for a potential function $\phi(x,y,z,t)$. The height of the free surface satisfies the partial differential equation

$$\eta_t = \partial_n\phi\sqrt{1+\eta_x^2+\eta_y^2} = \phi_z - \eta_x\phi_x - \eta_y\phi_y \tag{1}$$

where the subscripts refer to derivatives, and ∂_n refers to a derivative normal to the surface. We also make use of the function

$$\varphi(x,y,t) = \phi(x,y,\eta(x,y,t),t), \tag{2}$$

which is the restriction of the velocity potential to the free surface. Based on the unsteady Bernoulli equation, and the equation $\varphi_t = \phi_t + \phi_z \eta_t$, the velocity potential can be shown to satisfy

$$\varphi_t = P\left[\phi_z \eta_t - \frac{|\nabla\phi|^2}{2} - g\eta\right] \tag{3}$$

where P is the projection operator to zero mean and $g = 1$, as mentioned above. While including P is not standard in the water waves literature, it is a natural means of pinning down the arbitrary additive constant in the velocity potential, which must be done when comparing the initial and final states of the system to measure deviation from time-periodicity.

Since the fluid is incompressible and $\nabla \cdot \mathbf{v} = 0$, the bulk velocity potential must satisfy the three-dimensional Laplace equation $\nabla^2\phi = 0$. By solving the Laplace equation in the bulk, with Dirichlet conditions given by Eq. 2 and the Neumann condition

$$\phi_z(x,y,0,t) = 0, \tag{4}$$

the bulk velocity potential can be uniquely determined in terms of φ. To track the time-evolution of the free surface, it is therefore sufficient to track $\eta(x,y,t)$ and $\varphi(x,y,t)$, and integrate them according to Eqs. 1 and 3. However, at each stage, it is necessary to determine the bulk velocity potential in order to be able to evaluate the partial derivatives of ϕ that feature in these equations.

2.2. Spatial discretization

The fields η and φ are discretized on a square $M \times M$ grid with spacing $h = \frac{2\pi}{M}$. To evaluate the terms in these equations, the spectral derivatives η_x, η_y, φ_x and φ_y are calculated with the fast Fourier transform (FFT) using the FFTW3 library [42]. To evaluate the normal derivative of the bulk velocity potential, we make use of a fourth-order finite element discretization [43, 44], employing a rectangular grid of $M \times M \times (N+1)$ nodes, which is scaled in the vertical direction to conform with the free surface. Specifically, the (i,j,k) node is located at $(hi, hj, \eta(hi,hj)k/N)$, for i and j in the range $0,\ldots,M-1$ and k in the range $0,\ldots,N$.

The grid is then divided into $4 \times 4 \times 4$ patches as shown in Fig. 1. At each node, a finite element basis function can be defined in terms of Lagrange interpolants on each of the patches that includes that node; details of this are discussed below. This leads to a set of basis functions $\psi_1,\ldots,\psi_n$ at all of the interior nodes where $k < N$, and a further set of basis functions $\tilde{\psi}_1,\ldots,\tilde{\psi}_m$ on all surface nodes where $k = N$. The solution can be represented as

$$\phi = \sum_{i=1}^{n} \phi_i \psi_i + \sum_{i=1}^{m} \tilde{\phi}_i \tilde{\psi}_i \tag{5}$$

where $\tilde{\phi}_i$ are coefficients that are known from the current state of φ, and ϕ_i are unknown coefficients. The finite element formulation of the Laplace equation for the potential can then be expressed as

$$\sum_{i=1}^{n} \phi_i \iiint_V \nabla\psi_i \cdot \nabla\psi_j \, d^3x = -\sum_{i=1}^{m} \tilde{\phi}_i \iiint_V \nabla\tilde{\psi}_i \cdot \nabla\psi_j \, d^3x \tag{6}$$

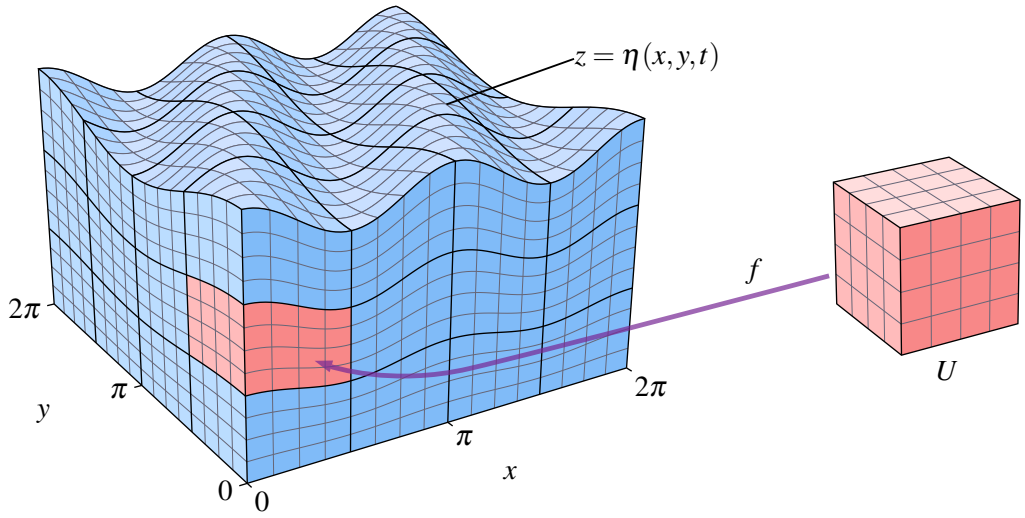

Figure 1: Illustration of the finite element formulation, in which the bulk of the fluid is divided into $4\times4\times4$ patches that are scaled according to the height of the fluid $z=\eta(x,y,t)$. To compute the finite element matrix, a reference space $U=[0,4]^3$ is introduced, and a map f from U to one of these patches is considered.

where the integration domain V refers to the entire fluid volume. To calculate the integrals in Eq. 6, a reference space $U=[0,4]^3$ is introduced, with coordinates (α,β,γ). In this space, the Lagrange interpolants L_0,L_1,L_2,L_3,L_4 can be defined as quartic polynomials that satisfy $L_i(i)=1$, $L_j(i)=0$ for $i=0,\ldots,4$, and $j=0,\ldots,4$ with $j\neq i$. A map f from the reference space into the $4\times4\times4$ patch in the real space $f(U)$ can then be defined as

$$f(\alpha,\beta,\gamma)=(x,y,z)=\left(h(n_x+\alpha),h(n_y+\beta),\tfrac{\eta(\alpha,\beta)}{N}(n_z+\gamma)\right).$$

Here, $\eta(\alpha,\beta)$ is the height of the surface, given by

$$\eta(\alpha,\beta)=\sum_{i=0}^{4}\sum_{j=0}^{4}\eta_{ij}L_i(\alpha)L_j(\beta)$$

where η_{ij} are coefficients that are given from the position of the top surface. The integers n_x, n_y, and n_z are multiples of four and determine which patch is being considered. To construct the stiffness matrix, we consider two basis functions on the reference space

$$\begin{aligned}T_1(\alpha,\beta,\gamma) &= L_a(\alpha)L_b(\beta)L_c(\gamma),\\ T_2(\alpha,\beta,\gamma) &= L_d(\alpha)L_e(\beta)L_f(\gamma),\end{aligned}$$

along with $\psi_1=T_1\circ f^{-1}$ and $\psi_2=T_2\circ f^{-1}$. The corresponding contribution to the finite element matrix is then

$$\begin{aligned}A_{12} &= \int_{f(U)}\nabla_{\mathbf{x}}\psi_1(\mathbf{x})\cdot\nabla_{\mathbf{x}}\psi_2(\mathbf{x})\,d^3x\\ &= \int_U(D^{-1}\nabla_\alpha T_1(\alpha))\cdot(D^{-1}\nabla_\alpha T_2(\alpha))(\det D)\,d^3\alpha,\end{aligned}\tag{7}$$

where the gradients are treated as column vectors and D is the (transpose of the) Jacobian matrix, given by

$$D = \begin{pmatrix} \frac{\partial x}{\partial \alpha} & \frac{\partial y}{\partial \alpha} & \frac{\partial z}{\partial \alpha} \\ \frac{\partial x}{\partial \beta} & \frac{\partial y}{\partial \beta} & \frac{\partial z}{\partial \beta} \\ \frac{\partial x}{\partial \gamma} & \frac{\partial y}{\partial \gamma} & \frac{\partial z}{\partial \gamma} \end{pmatrix} = \begin{pmatrix} h & 0 & \frac{n_z+\gamma}{N} \sum_{ij} \eta_{ij} L_i'(\alpha) L_j(\beta) \\ 0 & h & \frac{n_z+\gamma}{N} \sum_{ij} \eta_{ij} L_i(\alpha) L_j'(\beta) \\ 0 & 0 & \eta(\alpha,\beta)/N \end{pmatrix}.$$

The finite element matrix depends on the the height of the surface and has to be recomputed each time the surface moves. As described in more detail in Appendix B, many of the terms that feature in Eq. 7 can be simplified by storing pre-computed tables of integrals of the L_i basis functions and their derivatives. However, one term requires the use of two-dimensional integration by Gaussian quadrature.

For nodes on patch boundaries, the corresponding finite element basis function has a part in each adjoining patch. The number of neighbors to which a node is connected therefore varies from $5 \times 5 \times 5$ (if it is in a patch interior) to $9 \times 9 \times 9$ (if it is at a patch corner). The finite element problem is solved using a multigrid method described in more detail in the following section. However, once this is accomplished, a further step is required to extract a fourth-order accurate value for $\partial_n \phi$, with errors on derivatives measured in L^2. For an arbitrary test function v, assumed periodic in x and y, applying Green's first identity gives

$$\iiint_V \nabla\phi \cdot \nabla v \, d^3x = \iint_S v \frac{\partial \phi}{\partial n} \, dS = \int_{x=0}^{2\pi} \int_{y=0}^{2\pi} v \frac{\partial \phi}{\partial n} \sqrt{1+\eta_x^2+\eta_y^2} \, dx\, dy. \tag{8}$$

Here we used periodicity in x and y and the Neumann boundary condition of Eq. 4 to eliminate the other boundary terms. Rather than use this formula to compute $\partial_n \phi$, it is more convenient to compute the quantity

$$\mathscr{G}(\eta)[\varphi] = \sqrt{1+\eta_x^2+\eta_y^2}\, \partial_n \phi, \qquad (\varphi \text{ as in Eq. 2}) \tag{9}$$

since the additional factor is already present in Eq. 1. $\mathscr{G}$ is known as the Dirichlet–Neumann operator, and is readily shown to be self-adjoint [45]. If the solution in the bulk is represented as in Eq. 5 and the required quantity on the surface is

$$\mathscr{G}[\varphi] = \sum_{i=1}^{m} \zeta_i \tilde{\psi}_i|_S,$$

Eq. 8 can be written as the linear system

$$\sum_{i=1}^{n} \phi_i A_{ij} + \sum_{i=1}^{m} \tilde{\phi}_i \tilde{A}_{ij} = \sum_{i=1}^{m} \zeta_i \tilde{B}_{ij} \tag{10}$$

where

$$A_{ij} = \iiint_V \nabla\psi_i \cdot \nabla\tilde{\psi}_j \, d^3x, \qquad \tilde{A}_{ij} = \iiint_V \nabla\tilde{\psi}_i \cdot \nabla\tilde{\psi}_j \, d^3x$$

and

$$\tilde{B}_{ij} = \int_{x=0}^{2\pi} \int_{y=0}^{2\pi} \left(\tilde{\psi}_i|_S\right)\left(\tilde{\psi}_j|_S\right) dx\, dy.$$

Eq. 10 is a two-dimensional linear system for ζ_i, which can also be solved using the same multigrid algorithm described in the following section, by setting the vertical grid size to be one. Compared to the bulk finite element problem, the computation time required to solve this problem is small.

2.3. Multigrid algorithm

2.3.1. Mathematical formulation

In the calculations, solving the finite element problem in the bulk is the most computationally expensive step. Therefore, we have written a parallel geometric multigrid algorithm in C++ using the OpenMPI library that can take advantage of the problem's structure. The finite element problems introduced in the previous section are equivalent to solving a linear system $A_0 x_0 = b_0$, where A_0 is a sparse matrix, b_0 is a source term, and x_0 is the unknown quantity.

In the algorithm, the computational grid is labeled as the zeroth grid, and a hierarchy of progressively coarser grids labeled from 1 to G are introduced. The hierarchy follows standard procedures [46], where at each coarser level, the odd-numbered grid points are removed, so that grid g has dimensions $2^{-g}M \times 2^{-g}M \times (2^{-g}N + 1)$. The interpolation operator T_g from grid g to grid $g-1$ is implemented using trilinear interpolation; the interpolation is with respect to the logical grid structure, and does not take into account the curved, physical placement of the nodes. The corresponding restriction operator can then be defined as the matrix transpose, so that $R_g = T_{g+1}^T$.

If the linear operator describing the finite element problem on the computational grid is denoted as A_0, then the coarsened versions A_g can be recursively defined on the grid hierarchy by making use of

$$A_{g+1} = R_g A_g T_{g+1}. \tag{11}$$

Once these matrices are determined, they can be used to define smoothing operators S_g on each level that improve the solution via a Gauss–Seidel sweep. The restriction, interpolation, and smoothing operators can then be employed to carry out a standard V-cycle [46], discussed in more detail in Sec. 2.3.4.

2.3.2. Parallel decomposition of the top level problem

The multigrid algorithm is carried out in parallel by dividing the computational domain of grid points into a rectangular grid of 2^P vertical columns, so that a given thread will be responsible for the nodes satisfying $i_{\mathrm{lo}} \le i < i_{\mathrm{hi}}$, $j_{\mathrm{lo}} \le j < j_{\mathrm{hi}}$. Each point is uniquely assigned to one thread and there are no overlaps at the boundaries, and thus i_{hi} for one thread is equal to i_{lo} for the neighboring thread in the positive x direction. Typically, in the grids considered here, the number of grid points in the horizontal directions are larger than in the vertical direction, and thus it was found unnecessary to also decompose the domain in the vertical direction. During the multigrid algorithm, each thread is responsible for storing the parts of the solution within its vertical column of grid points, as well as the corresponding rows of the sparse matrix A_0. Each thread is assigned a unique index (discussed in more detail in the following section), and has a table of the eight threads that are either orthogonally and diagonally adjacent.

For the finite element problem considered, in the row of A_0 corresponding to a given grid point (i, j, k) there will be non-zero entries in columns corresponding to a rectangular box of nearby grid points (i', j', k') that satisfy the constraints $i_- \le i' \le i_+$, $j_- \le j' \le j_+$, and $k_- \le k' \le k_+$.

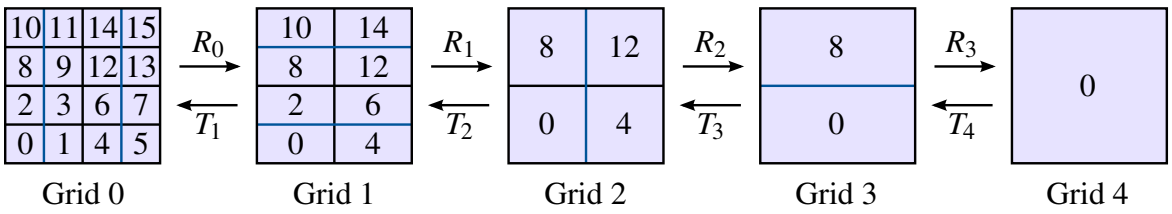

Figure 2: Typical multigrid hierarchy when using sixteen threads. On the finest grid 0, the domain is divided into a 4×4 grid, which is assigned to the threads via a binary numbering scheme as described in the text. At each coarser level, the number of threads involved is halved, and the domains are amalgamated between the remaining threads.

For each grid point, the code therefore stores the indices $i_\pm$, $j_\pm$, and $k_\pm$, plus an array of length $(i_+ - i_- + 1)(j_+ - j_- + 1)(k_+ - k_- + 1)$ of the corresponding matrix entries. Each thread is responsible for computing the matrix entries using the tables and quadrature calculations that are described above, which requires knowledge of η in the local patch of grid points corresponding to that thread's column.

Grid points near the boundary of a particular column will have matrix entries corresponding to grid points in neighboring columns, and the solution values x_0 at these points will be required in order to carry out a smoothing operation. Because of this, memory is allocated for ghost regions of appropriate dimensions in which to store the values of x_0 from neighboring threads. A smoothing operation can then be carried out as a two step process:

1. Communicate with the eight orthogonally and diagonally adjacent threads to populate the ghost regions with the current solution x_0.
2. Carry out a Gauss–Seidel sweep at all grid points in the column.

This smoothing operation is a hybrid between the Gauss–Seidel and Jacobi methods. While the values of x_0 within a thread's column are updated *in situ*, the values of x_0 at the ghost grid points that a thread uses remain fixed throughout step 2. For comparison, a variation on this method was considered in which a Jacobi iteration was employed by each thread in step 2, in which case the overall update is equivalent to a Jacobi iteration. However, the hybrid approach gave faster convergence, although with the largest residuals typically occurring at column boundaries.

2.3.3. Implementation of the multigrid hierarchy

In certain parallel architectures (*e.g.* a cluster of multi-core processors), it is faster to communicate between threads with similar indices, and thus the assignment of the different columns in the multigrid algorithm is done in a way to maximize the proportion of communication between threads in appropriate consecutive ranges. To do this, a thread's index is viewed as a binary number, and the odd-numbered bits are used to create the x position of the column, while the even-numbered bits are used to create the y position of the column. Thus, if a thread has a binary representation given by $fedcba$, its x position is eca and its y position is fdb. An example thread decomposition on sixteen threads into a 4×4 grid is shown in Fig. 2.

Within the multigrid algorithm, as the grid is coarsened, the number of grid points that are owned by a single thread will rapidly decrease, making the column decomposition less efficient, as proportionally more time will be spent communicating solution values with neighboring threads.

Because of this issue, the multigrid algorithm employs a strategy of halving the number of threads used to compute each coarser level, by amalgamating pairs of neighboring domains together. Specifically, for the computations on grid g, the threads with indices that are multiples of 2^g are employed. To restrict from grid g to grid $g+1$, a thread with index of the form $2^g(2k+1)$ passes its domain to $2^g(2k)$. Because of the binary thread numbering system used, the thread amalgamation will happen alternately in the x and y directions. A typical thread decomposition for the multigrid hierarchy with sixteen threads is shown in Fig. 2.

On each of the coarser levels, the threads involved employ exactly the same procedures for representing the linear system and carrying out the smoothing operation as discussed in the previous section. While the majority of communication in the algorithm comes from the amalgamation of grids from neighboring threads when moving between levels, it is also necessary to carry out small communications within a level to complete some operations. To carry out the interpolation operation, threads require an edge strip from their neighbors in order to be able compute the interpolated values at their local domain boundaries.

To initially set up the linear system, the following recursive procedure is carried out to compute the system at level $g+1$ using the information at level g:

1. Threads at level g communicate the bound information $(i_\pm, j_\pm, k_\pm)$ in edge strips to their neighbors. Using this, and their own bound information, they calculate the bound information at level $g+1$.
2. Threads at level g with indices of the form $2^g(2k+1)$ communicate their calculated bound information to those with the form $2^g(2k)$.
3. Threads at level $g+1$ use the bound information to calculate the precise amount of memory required for their representation of the linear system at level $g+1$, and allocate it.
4. Threads at level g communicate edge strips of the linear system to their neighbors. Using this, and their own information they calculate the linear system at level $g+1$.
5. Threads at level g with indices of the form $2^g(2k+1)$ communicate the calculated linear system to those with the form $2^g(2k)$.

For the current problem, where the finite element coefficients change as the surface moves, but the bound information remains constant, a quicker set-up method can be carried out by only employing steps 4 and 5.

2.3.4. Testing and configuration of the multigrid algorithm

The multigrid code was tested on the fourth-order curved finite element problem described previously, as well as a simpler fourth-order finite element problem using rectangular patches. Within the V-cycle, n_{down} pre-smoothing Gauss–Seidel operations are applied on the way down the grid hierarchy, and n_{up} post-smoothing Gauss–Seidel operations are applied on the way up. On the coarsest grid level, the problem was smoothed by making use of twenty Gauss–Seidel operations; changing this number of smoothing operations did not affect the computed residuals, suggesting that this was adequate in solving the coarse problem, without the need to carry out a direct solve using dense linear algebra.

To find optimal choices of n_{down} and n_{up}, a test problem for $\Omega(x,y,z)$ was considered where $0 \leq z \leq 2\pi$, and x and y are 2π-periodic. The function satisfied $\nabla^2\Omega = 0$ subject to the boundary

	$n_{\text{up}}=1$	$n_{\text{up}}=2$	$n_{\text{up}}=3$	$n_{\text{up}}=4$	$n_{\text{up}}=5$	$n_{\text{up}}=6$
$n_{\text{down}}=0$	19.58	24.73	26.43	28.77	28.63	25.66
$n_{\text{down}}=1$	24.66	27.00	28.61	28.38	25.45	23.04
$n_{\text{down}}=2$	27.17	28.71	28.36	25.40	23.04	21.25
$n_{\text{down}}=3$	28.82	28.38	25.33	23.03	21.22	19.79
$n_{\text{down}}=4$	28.38	25.39	23.05	21.26	19.80	18.65
$n_{\text{down}}=5$	25.31	23.03	21.24	19.84	18.65	17.60

Table 1: Convergence rates for a test problem using a $64\times64\times33$ grid using eight threads on a dual Intel Xeon E5-2650L system. For each choice of n_{up} and n_{down} in the V-cycle, the convergence rate measured in terms of digits of accuracy obtained per wall-clock second is reported. Each measurement is based on the average of sixteen trials.

	$n_{\text{up}}=1$	$n_{\text{up}}=2$	$n_{\text{up}}=3$	$n_{\text{up}}=4$	$n_{\text{up}}=5$	$n_{\text{up}}=6$
$n_{\text{down}}=0$	0.802	1.005	1.101	1.159	1.150	1.016
$n_{\text{down}}=1$	1.002	1.097	1.158	1.147	1.026	0.916
$n_{\text{down}}=2$	1.097	1.159	1.145	1.026	0.928	0.843
$n_{\text{down}}=3$	1.163	1.146	1.027	0.929	0.855	0.786
$n_{\text{down}}=4$	1.147	1.025	0.929	0.857	0.798	0.739
$n_{\text{down}}=5$	1.022	0.928	0.856	0.798	0.748	0.692

Table 2: Convergence rates for the multigrid test problem using a $256\times256\times65$ grid using 32 threads on a dual Intel Xeon E5-2650L system. For each choice of n_{up} and n_{down} in the V-cycle, the convergence rate measured in terms of digits of accuracy obtained per wall-clock second is reported. Each measurement is based on the average of sixteen trials.

conditions $\partial_z\Omega(x,y,0)=0$ and

$$\Omega(x,y,2\pi)=\begin{cases}1 & \text{for } 0\le x<\pi \text{ and } 0\le y<\pi,\\ 0 & \text{otherwise.}\end{cases} \tag{12}$$

This problem was solved with a variety of grid sizes and choices of n_{down} and n_{up}. Here, and for all subsequent timing results, a Scientific Linux system with 64 GB of memory and dual 1.8 GHz Intel Xeon E5-2650L processors was used—this system has sixteen physical cores, and thirty-two virtual cores using hyperthreading. Version 4.7.1 of the GNU Compiler and version 1.6 of the OpenMPI library were used. For each parameter choice, the wall-clock time t_w for fifteen V-cycles was measured, and the residuals r_{10} and r_{15} were computed after ten and fifteen V-cycles. From this, a convergence rate can be calculated as $(\log_{10}r_{10}-\log_{10}r_{15})/(3t_w)$, as a measure of digits of convergence per wall-clock time.

Table 1 shows the convergence rates on a $64\times64\times33$ grid using eight threads for various choices of n_{down} and n_{up}. Table 2 shows corresponding data for $256\times256\times65$ grid using 32 threads. In both cases, it can be seen that the convergence rates are largely governed by the sum $n_{\text{down}}+n_{\text{up}}$, and are optimal for $n_{\text{down}}+n_{\text{up}}=4$ and $n_{\text{down}}+n_{\text{up}}=5$. Values of $n_{\text{down}}=n_{\text{up}}=2$ were therefore chosen. Other multi-processor systems, such as a 64-core AMD Opteron server or Intel Mac Pro, had the same optimal values of n_{down} and n_{up}. However, the optimal values are likely to be hardware dependent and on systems with a slower method of data transfer (*e.g.* a Beowulf cluster) they may be larger. A full multigrid cycle [46] was also investigated although for the problems considered here the convergence rate was similar to the V-cycle.

Carrying out a multigrid solve therefore proceeds as follows. Initially, for diagnostic purposes, the mean square residual is calculated. V-cycles are then performed, and after every five the mean square residual per grid point is evaluated to see if a threshold of 10^{-25} has been reached. Once the threshold has been reached, an additional five V-cycles are carried out to approach the limit of machine precision. In this multigrid solution procedure the residual is only checked after every five V-cycles since it is relatively expensive to evaluate.

2.4. Time-integration of the forward problem

By making use of the Fourier transform to compute η_x, η_y, φ_x, φ_y and the finite element problem to compute $\partial_n\phi$, all information needed to time-integrate Eqs. 1 and 3 can be obtained. By considering the matrix problem

$$\begin{pmatrix} \varphi_x \\ \varphi_y \\ \partial_n\phi\sqrt{1+\eta_x^2+\eta_y^2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \eta_x \\ 0 & 1 & \eta_y \\ -\eta_x & -\eta_y & 1 \end{pmatrix} \begin{pmatrix} \phi_x \\ \phi_y \\ \phi_z \end{pmatrix}$$

at each grid point, the surface derivatives and normal derivative of the velocity potential can be used to calculate the orthogonal derivatives that feature in these equations. The equations are time-integrated by making use of the eighth-order Dormand–Prince DOP853 scheme [47]. This is a thirteen-step scheme with the "first same as last" property, requiring twelve evaluations of the right hand sides of Eqs. 1 and 3 per time step. We have also found the six-stage fifth-order explicit scheme DOPRI5 [47] to be very effective for this problem.

The code was tested against a previously developed two-dimensional boundary integral code by using initial conditions that were a function of the x coordinate only, and it produced identical results to a high degree of accuracy. In these tests it was discovered that filtering was required to keep the calculations stable over long time intervals. The largest source of error was due to the computation of $\partial_n\phi$, and an oscillation with a wavelength of four grid points (corresponding to the width of a finite element patch) became visible in the height and velocity potential fields. Because of this, we filter out all Fourier modes with frequencies down to and including this four grid point oscillation. Filtering is applied at each intermediate step of the Dormand–Prince time-integration.

Figure 3 shows several snapshots of a test simulation with initial conditions

$$\eta(x,y,0) = \frac{\pi + e^{-3((x-\pi)^2+(y-\pi)^2)}}{2}, \qquad \varphi(x,y,0) = 0,$$

extended to be 2π-periodic in x and y. The simulation was carried out on a $256\times256\times65$ computational grid using 32 threads on the test system with dual Intel Xeon E5-2650L processors. For this example, the sparse matrix representing the finite element problem has 896,598,016 non-zero entries; total memory allocation for the simulation was 8.3 GB. The simulation was integrated from $t=0$ to $t=20$ using 200 time steps of fixed size 0.1.

For this test calculation, Table 3 shows a breakdown of the average wall-clock times needed to evaluate the derivatives (η_t,φ_t) for one step of the time integration. The times given here are based on when the zeroth thread starts and ends each task, and no explicit synchronization is employed;

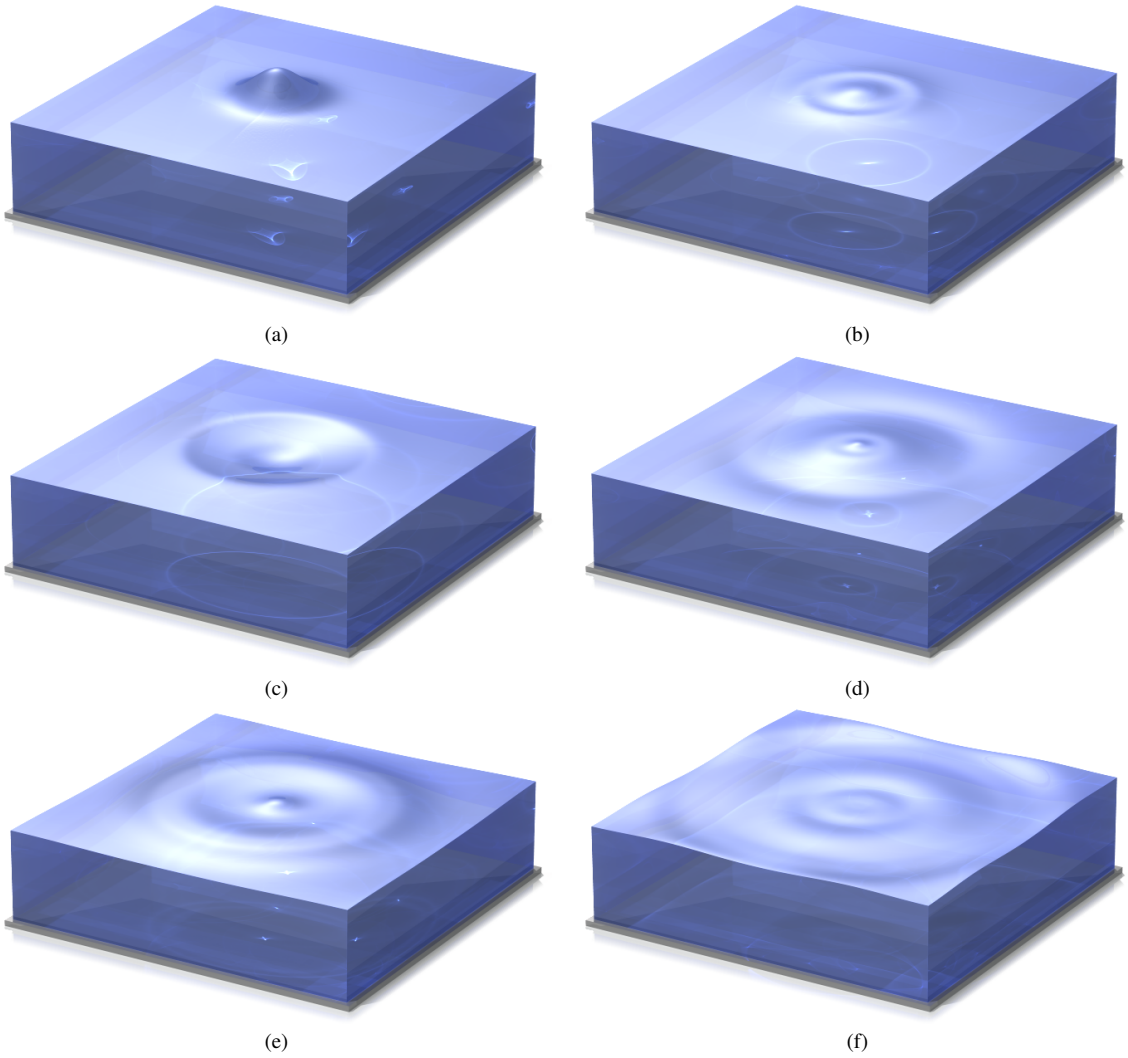

Figure 3: Snapshots of an example computation, in which the fluid is initially at rest, and the surface is flat apart from a small Gaussian peak in the center of the simulation domain. Snapshots of the system at $t = 0, 1, 2.5, 4.5, 6.5, 9.5$ are given in (a) to (f) respectively, showing how the peak collapses and a ripple moves radially outwards.

Task	Wall clock time	Percentage
x and y derivatives using FFT	25.85 ms	0.094%
Quadrature pre-calculation	0.1714 s	0.624%
Bulk system initialization	2.731 s	9.943%
Bulk multigrid {24.99}	23.38 s	85.118%
Bulk error evaluations {4.999}	1.018 s	3.705%
Surface setup	62.63 ms	0.228%
Surface multigrid {14.98}	63.80 ms	0.232%
Surface error evaluations {2.997}	4.851 ms	0.018%
FFT filtering of normal derivative	8.593 ms	0.031%
Evaluation of η_t and φ_t	1.357 ms	0.005%

Table 3: Summary of the time required for a single evaluation of the time derivatives η_t and φ_t for the test problem discussed in the text, using 32 threads on a $256 \times 256 \times 65$ grid. Times are recorded by the zeroth thread and no explicit synchronization is used. Results are averaged over 2,400 evaluations (200 time steps with 12 evaluations each). For the multigrid calculations and error evaluations, the values in braces show the average number of V-cycle and error evaluations respectively that are employed. The tasks are listed in the order of execution, other than the V-cycle and error evaluations, which are interspersed.

Level	Grid size	Threads	ASE	Pre-smoothing	Restriction	Interpolation	Post-smoothing
0	$256 \times 256 \times 65$	32	$210\,{}^{31}/_{65}$	37.179%	19.556%		37.180%
1	$128 \times 128 \times 33$	16	$62\,{}^{18}/_{33}$	1.929%	1.123%	0.272%	1.810%
2	$64 \times 64 \times 17$	8	$25\,{}^{16}/_{17}$	0.239%	0.146%	0.040%	0.209%
3	$32 \times 32 \times 9$	4	25	0.073%	0.043%	0.015%	0.059%
4	$16 \times 16 \times 5$	2	$23\,{}^{2}/_{5}$	0.027%	0.017%	0.006%	0.017%
5	$8 \times 8 \times 3$	1	21	0.054%		0.007%	

Table 4: Percentage of times spent in each stage of the bulk multigrid computation for the test wave problem, using explicit processor synchronization. ASE refers to the average number of stencil entries in a row of the linear system. Pre- and post-smoothing involves two Gauss–Seidel sweeps for levels 0 to 4, and twenty Gauss–Seidel sweeps on level 5. The average wall-clock time for a single V-cycle is 30.28 s, 30% slower than when run asynchronously as in Table 3. Results are averaged over the 59,985 V-cycles that are required to carry out the test problem.

Level	Grid size	Threads	ASE	Pre-smoothing	Restriction	Interpolation	Post-smoothing
0	256×256	32	36	25.640%	12.355%		19.435%
1	128×128	16	16	7.587%	4.704%	1.832%	5.878%
2	64×64	8	9	3.142%	1.812%	0.847%	2.429%
3	32×32	4	9	2.153%	1.288%	0.678%	1.463%
4	16×16	2	9	1.183%	1.136%	0.395%	1.202%
5	8×8	1	9	4.302%		0.541%	

Table 5: Percentage of times spent in each stage of the surface multigrid computation for the test wave problem, using explicit processor synchronization. ASE refers to the average number of stencil entries in a row of the linear system. Pre- and post-smoothing involves two Gauss–Seidel sweeps for levels 0 to 4, and twenty Gauss–Seidel sweeps on level 5. The average wall-clock time for a single V-cycle is 78.14 ms, 22% slower than when run asynchronously as in Table 3. Results are averaged over the 35,960 V-cycles that are required to carry out the test problem.

while this gives the best reflection of the normal operation, it does not take into account that some tasks may start and end on the other processors at slightly different times. For solving the velocity potential in the bulk, it can be seen that the quadrature calculation takes a relatively small amount of time, since it only needs to be carried out over a two-dimensional grid of points. The time to initialize the finite element matrix is appreciable, but the time to carry out the bulk multigrid solve dominates. The time spent on the surface problem is much smaller than for the bulk problem, and fewer V-cycles are needed on average. Compared to the multigrid problems, the times for the two-dimensional calculations, such as computing the x and y derivatives and filtering $\partial_n \phi$, are relatively minor.

For this test problem the different operations within the bulk multigrid algorithm were also timed and are shown in Table 4. Here, to ensure representative timings of each operation, an `MPI_Barrier` call was inserted between each component of the algorithm, so that all processors are synchronized. With the addition of the barriers, the average time for the multigrid algorithm is 30.28 s, which is a 30% speed reduction compared to the time in Table 3. It can be seen that the wall-clock times of the Gauss–Seidel sweeps are the largest, since they require multiple evaluations of all of the matrix components. Restrictions are more expensive than interpolations, since they require an evaluation of the residual, which involves accessing all of the matrix components.

The wall-clock times reduce rapidly at each level down the hierarchy. The number of grid points is reduced by a factor of eight, and in addition the stencil size is also decreased; taken together, these two factors significantly outweigh that the number of processors is halved. As a consequence, the majority of time, 74.36%, is spent on Gauss–Seidel smoothing on the top level. Table 5 shows the corresponding data for the surface multigrid calculation. Since this is a two-dimensional problem, the reduction in work at each level down the hierarchy is not as significant, but the largest fraction of wall-clock time is still spent on top-level smoothing.

2.5. Optimization

To search for periodic solutions, a situation is considered whereby a wave is at rest so that $\varphi(x,y,0) = 0$, and the initial wave surface is given by

$$\eta(x,y,0) = \sum_{i=1}^{Q} \alpha_i f_i(x,y), \tag{13}$$

where the α_i are real parameters and the f_i are a basis of functions for the surface; typically, the f_i are given in terms of a two-dimensional Fourier decomposition of the free surface, and are described in detail in the specific problems that are presented in the following sections.

We are interested in solutions that pass through a rest configuration twice per cycle. A time-reversal argument shows that if $\varphi \equiv 0$ at $t = 0$ and $t = \frac{T}{2}$, the solution will be time-periodic with period T. Thus, to numerically search for a time-periodic solution, the functional

$$S = \frac{1}{2} \int_0^{2\pi} \int_0^{2\pi} \left| \varphi\left(x, y, \frac{T}{2}\right) \right|^2 dx\, dy \tag{14}$$

is minimized over the configuration space given by

$$c = (T, \alpha_1, \ldots, \alpha_Q). \tag{15}$$

In a discretized form, this functional can be written as

$$S(c) = \frac{1}{2} r(c)^T r(c)$$

where r is a residual vector of length N^2 representing the discretized field φ at $\frac{T}{2}$. Since the surface is represented spectrally, we use a two-dimensional trapezoidal rule to approximate S in Eq. 14 rather than integrating patch by patch using the finite element basis. Either approach can be done by weighting the components of r appropriately.

The minimization procedure is carried out in the overdetermined shooting framework of Wilkening and Yu [34], which has been found to be more robust than a standard shooting method in which the number of nonlinear equations is equal to the number of unknowns. The problem is overdetermined because the initial conditions are zero-padded in Fourier space while the residual vector r contains φ at all grid points. Sufficient zero-padding ensures that all the unknown modes in Eq. 13 are well-resolved by the finite element mesh. The underlying solver is a variant of the Levenberg–Marquardt method [48] in which re-computation of the Jacobian

$$J = \nabla_c r \tag{16}$$

is delayed for several iterations until the previous Jacobian ceases to be helpful. In this approach, S is approximated near a particular configuration c via

$$S_{\text{approx}}(c+p) = S(c) + r^T J p + \frac{p^T (J^T J) p}{2}$$

and minimized over a trust region $\|p\| \le \Delta$. Details of how the trust region radius Δ should be varied when the Jacobian updates are delayed over several residual computations are given in Ref. [34].

To carry out the minimization procedure, it is therefore necessary to be able to efficiently evaluate the Jacobian in Eq. 16 for a given configuration c. Computing the first column of the Jacobian, corresponding to $\partial_T r$, can be done by evaluating $\varphi_t(x, y, \frac{T}{2})$. This is straightforward since φ_t is already required to time-evolve the original problem. To evaluate the remaining columns of J, it is necessary to consider a variational problem for a solution (η_*, φ_*) where $\eta_* = \eta + \varepsilon\dot{\eta}$ and $\varphi_* = \varphi + \varepsilon\dot{\varphi}$. Here (η, φ) is the base solution, and $(\dot{\eta}, \dot{\phi})$ corresponds to a small perturbation. To compute the column of J corresponding to α_i in Eq. 15, the initial conditions $\dot{\eta}(x, y, 0) = f_i(x, y)$ and $\dot{\varphi}(x, y, 0) = 0$ are employed, with f_i as in Eq. 13.

As described in Appendix A, a partial differential equation system for $\dot{\eta}$ and $\dot{\varphi}$ can be derived by substituting the expressions for η_* and φ_* into Eqs. 1, 2, and 3 and using that η and φ are a solution of these equations. In a similar manner to the original problem, a velocity potential variation $\dot{\phi}(x, y, z, t)$ in the bulk must be considered. It satisfies

$$\nabla^2 \dot{\phi} = 0 \tag{17}$$

in the region $0 < z < \eta(x, y, t)$, and $\partial_z \dot{\phi} = 0$ at $z = 0$. However, by considering Eq. 2, it can be seen that on the free surface at $z = \eta(x, y, t)$,

$$\dot{\varphi} = \dot{\phi} + \phi_z \dot{\eta}, \tag{18}$$

with the additional term arising because a variation in the surface $\dot{\eta}$ affects the point at which $\dot{\varphi}$ is evaluated. By considering Eq. 1 and 3 the evolution equations

$$\dot{\eta}_t = -(\dot{\eta}\phi_x)_x - (\dot{\eta}\phi_y)_y + \dot{\phi}_n\sqrt{1+\eta_x^2+\eta_y^2} \tag{19}$$

and

$$\dot{\phi}_t = P\left[-(\dot{\eta}\phi_x\phi_z)_x - (\dot{\eta}\phi_y\phi_z)_y - K - \dot{\eta}\right] \tag{20}$$

can be derived, where

$$K = \begin{pmatrix} \phi_x & \phi_y & -\phi_z \end{pmatrix} \begin{pmatrix} 1 & 0 & \eta_x \\ 0 & 1 & \eta_y \\ -\eta_x & -\eta_y & 1 \end{pmatrix} \begin{pmatrix} \dot{\phi}_x \\ \dot{\phi}_y \\ \dot{\phi}_z \end{pmatrix}.$$

The corresponding column of J can then be computed using $\dot{\varphi}(x,y,\frac{T}{2})$. The partial differential equation system given by Eqs. 19 and 20 is solved using the same numerical procedures as for the original problem. In order to solve these equations, it also necessary to concurrently integrate the base solution η and ϕ, since these terms feature in the equations.

A significant numerical speed-up can be achieved by solving for multiple columns of J simultaneously. The velocity potential variation in the bulk is solved on the same geometry for each different variation, and thus solving Eq. 17 for different variations is equivalent to solving the same linear system $Ax = b$ for different inputs b. The matrix A is very large, and expensive to read from memory, and thus it is more efficient to make use of the values multiple times once they have been read. The parallel multigrid algorithm described in the previous section was therefore extended to allow for solving a linear system for a vector of different inputs. The code to carry this out is automatically built from the source code of the base algorithm by adding loops over the different inputs b and corresponding solutions, and by increasing the size of all inter-thread communications.

For a test problem on a $160 \times 160 \times 65$ grid, the wall-clock time for a V-cycle on a vector input of length n on the dual Intel Xeon E5-2650L system using sixteen threads can be well-modeled by a linear relationship $(0.616 + 0.155n)$ s, meaning that each additional input adds only a fifth as much additional computation time. Since memory allocation increases as n is increased, the columns of J are computed in batches of up to 64 columns at a time. For the problems considered here, this typically results in no more than three batches of columns being considered. Since the columns in a batch will represent disparate wavelengths and oscillation frequencies, each with a different stepsize requirement, a uniform timestep is used and adaptive timestepping procedures are not considered.

3. Results

Compared to previous two-dimensional studies, the computational cost of searching for three-dimensional time-periodic solutions is very high. For typical cases that are considered, where the free surface is parameterized with up to 150 degrees of freedom, a search takes three to seven days using sixteen threads on a multiple-processor machine. For each of the cases considered, the search

for time-periodic solutions begins by considering small perturbations to a flat surface of the form

$$\eta(x,y,t) = H + \varepsilon \cos kx \cos ly \cos \frac{2\pi t}{T}, \tag{21}$$

$$\varphi(x,y,t) = -\frac{\varepsilon T}{2\pi} \cos kx \cos ly \sin \frac{2\pi t}{T}, \tag{22}$$

where ε is a small parameter. By substituting these perturbations into Eqs. 19 and 20, this can be verified to be a solution provided that

$$T = \frac{2\pi}{\sqrt{\kappa \tanh \kappa H}}, \tag{23}$$

where $\kappa^2 = k^2 + l^2$. Searching for a time-periodic solution can then be carried out by fixing the size of the perturbed mode, and then minimizing over a configuration space $c = (T, \alpha_1, \ldots, \alpha_Q)$ where the α_i are other modes of a Fourier decomposition of the surface. As an initial guess to supply to the minimization, T is taken from Eq. 23, and the α_i are all assumed to be zero. For a small perturbation, the corrections needed from the α_i are small, and decay rapidly with wave number. Typically the minimization procedure only requires a single Jacobian computation to converge.

This procedure can be used to find several configurations $c_1, c_2, \ldots, c_b$ corresponding to small values $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_b$ of the fixed mode. To move to a slightly higher value of the mode, ε_*, an initial guess for the configuration vector can be found using Lagrangian interpolation, as

$$c'_* = \sum_{i=1}^{b} c_i \left(\prod_{\substack{j=1 \\ j \neq i}}^{b} \frac{\varepsilon_* - \varepsilon_j}{\varepsilon_i - \varepsilon_j} \right). \tag{24}$$

This formula can then be used trace out a path of time-periodic solutions by moving to progressively larger amplitudes of the fixed mode. In previous two-dimensional studies [20], a similar numerical continuation approach has been used, although using only linear extrapolation from two previous solutions. Here, given the high cost of each minimization, and the fact that a good initial guess may reduce the number of Jacobian computations, high-order extrapolation was deemed advantageous. Throughout the study, each extrapolation was carried out manually, usually using three or four previous solutions to achieve quadratic or cubic accuracy.

3.1. Waves of intermediate depth

The first case that is considered is for water of depth $H = \pi$, which is comparable to the wavelengths $\lambda_x = \lambda_y = 2\pi$. The free surface height is assumed to be even in the x and y directions, and also symmetric about the line $x = y$. With these constraints, the free surface can be described in terms of Fourier modes of the form

$$f_{r,s}(x,y) = \cos rx \cos sy + \cos sx \cos ry, \tag{25}$$

which is referred to as the (r,s) mode. In this section, we consider time-periodic solutions by varying the $(1,1)$ mode. By reference to Eq. 23, the period for a small amplitude mode of this form

is $T = 5.28424$. At this depth the hyperbolic tangent featuring in the equation is close to one, and the period of small-amplitude waves is close to the infinite depth limit, $T = 2^{3/4}\pi$.

The waves are calculated on a $160 \times 160 \times 65$ grid, making use of 25 time steps to simulate up to $\frac{T}{2}$. The $(1,1)$ mode has an additional symmetry under a translation by (π,π), and in order to preserve this symmetry it is therefore sufficient to search for a time-periodic solution using modes of the form in Eq. 25 with $r+s$ being even. The configuration space vector $c = (T, \alpha_1, \ldots, \alpha_Q)$ was therefore assembled as all of the modes (r,s) with $r+s$ even, omitting $(1,1)$, up to a cutoff.

To begin the search for time-periodic solutions, all modes with $r < 12$ and $s < 12$ were considered, which corresponds to a total of 41 degrees of freedom once all symmetries have been accounted for. The $(1,1)$ amplitude was fixed to be 0.02, and the linearized solution was used as the initial guess for the period; the initial guess for each of the other modes was set to zero. The minimization procedure converged rapidly, with the residual in Eq. 14 being $S = 5.93 \times 10^{-21}$. Figure 4(a) shows a plot of the initial wave surface $\eta(x,y,0)$, which appears very close to the linearized solution given by $0.02 f_{1,1}(x,y)$. The largest amplitude found during the minimization is 2.829×10^{-4} in the $(2,2)$ mode, and the period of the wave is 5.28551, which differs from the linearized period by 0.024%.

In a similar manner, a wave with a $(1,1)$ amplitude of 0.04 can be found, and using these two solutions Eq. 24 can be employed to search for waves of increasingly higher amplitude. A sequence of cross-sections of the waves on the line $y = 0$ is shown in Fig. 4(b). By the time that the $(1,1)$ amplitude has been increased to 0.28, the wave heights clearly deviate from the linearized form, with the wave crests becoming sharper and wave troughs becoming wider, in a similar manner to the trends seen in two-dimensional computations [14, 49].

Figure 4(c) shows a plot of the initial wave surface for the case when the $(1,1)$ amplitude is 0.24. The wave crests are noticeably sharper than in Fig. 4(a), and appear pyramidal, being connected by four diagonal ridges to the neighboring crests. Here, and in subsequent analysis, it is useful to examine diagonal cross-sections of the waves, by introducing a new coordinate ξ such that $(x,y) = (\xi/\sqrt{2}, \xi/\sqrt{2})$. Figure 4(d) shows cross-sections along this coordinate, demonstrating that the ridges between crests grow in height as the $(1,1)$ amplitude is increased.

The wave shapes can be compared to previously-studied two-dimensional waves. To do this, we applied the minimization procedure to a degenerate system with translational symmetry, using a basis of the form $f'_r(x,y) = \cos(r(x+y))$. Figure 5(a) shows a plot of a standing wave η_d for the case when the amplitude of f'_1 is 0.24. The other modes f'_r for $r = 2, 3, \ldots, 19$ were found by minimization. Finding solutions of this form was a useful test of the minimization procedure, since it verified that translational symmetry of the solutions can be preserved even when the symmetry is not aligned with a coordinate direction. The solid lines in Fig. 5(b) show cross-sections of the wave height along the x-axis for four amplitudes of f'_1. For comparison, the thin dashed black overlays are calculated using the boundary integral method of Ref. [34] for deep water, which has very similar dynamics to the current regime. Since the functions f'_r have a wavelength of $2\pi/\sqrt{2}$, it is necessary to rescale the boundary integral results by $1/\sqrt{2}$ to make the comparison. Each boundary integral curve is calculated by fixing the amplitude of $\cos x$ to be equivalent to that of f'_1. The two methods agree to 5–6 digits of accuracy, with the discrepancies being due to both numerical errors and the difference in fluid depth.

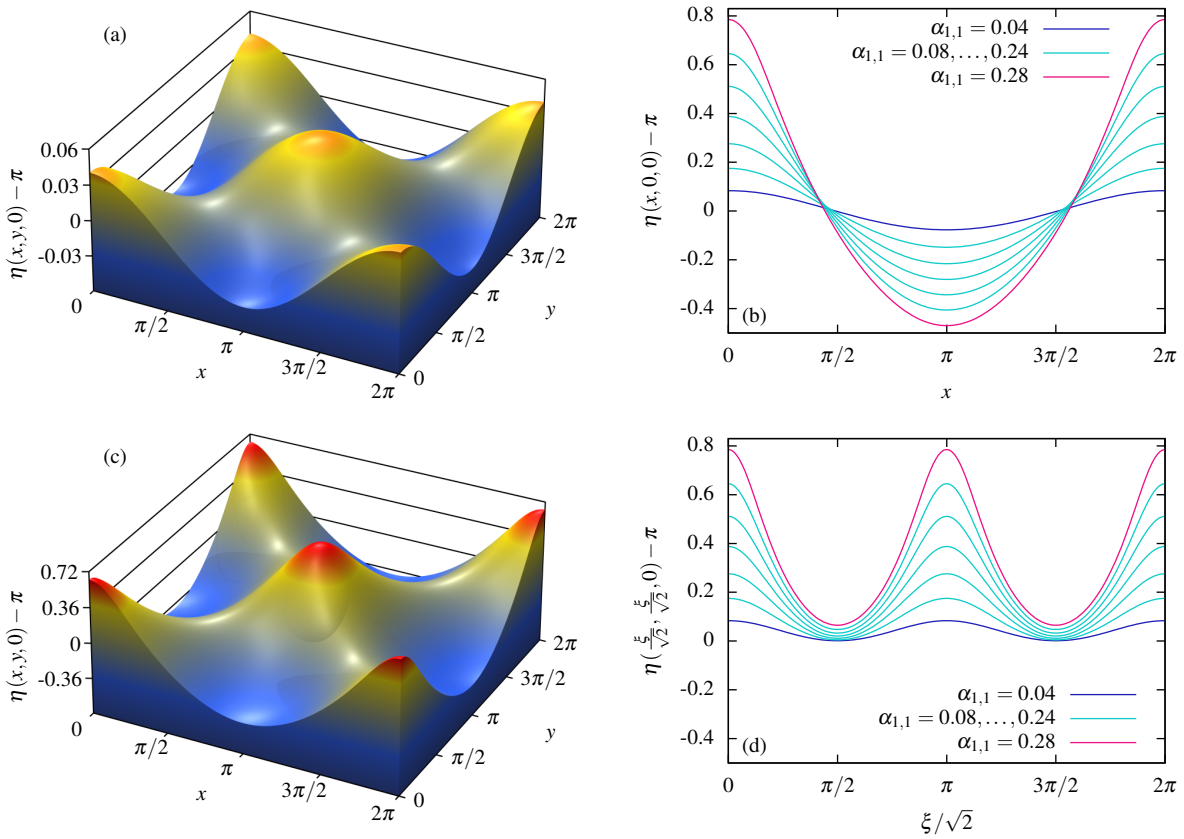

Figure 4: (a) The initial surface of a time-periodic wave close to the linearized regime, in which the amplitude of the $(1,1)$ mode is fixed to be 0.02 and the amplitudes of the other modes are determined by minimizing S; (b) cross-sections of the initial wave heights for a sequence of $(1,1)$ mode amplitudes, defined as $\alpha_{1,1}$; (c) the initial surface of a wave where the $(1,1)$ mode is increased to 0.24, showing noticeably sharper wave crests and wider wave troughs; (d) diagonal cross-sections of initial wave heights for a sequence of $(1,1)$ mode amplitudes.

To quantitatively compare the shapes of the degenerate two-dimensional waves to the three-dimensional counterparts, note that

$$f_{1,1}(x,y) = 2\cos x\cos y = \cos(x+y)+\cos(x-y) = f_1'(x,y)+f_1'(x,-y) \tag{26}$$

and thus $f_{1,1}$ is the linear superposition of two translationally invariant waves. It is therefore possible to examine how much the three-dimensional waves shown in Fig. 4 differ from a linear superposition of the degenerate wave shown in Fig. 5(a). Figure 5(c) shows a plot of the superposition, defined as

$$\eta_s(x,y,0) = \eta_d(x,y,0)+\eta_d(x,-y,0)-\pi. \tag{27}$$

The height of the superposition is similar to Fig. 4(c) but there are some small variations. In Fig. 5(d) the difference between the three-dimensional wave and the linear superposition is plotted, showing that the three-dimensional wave has both higher crests and troughs, while having lower ridges between crests.

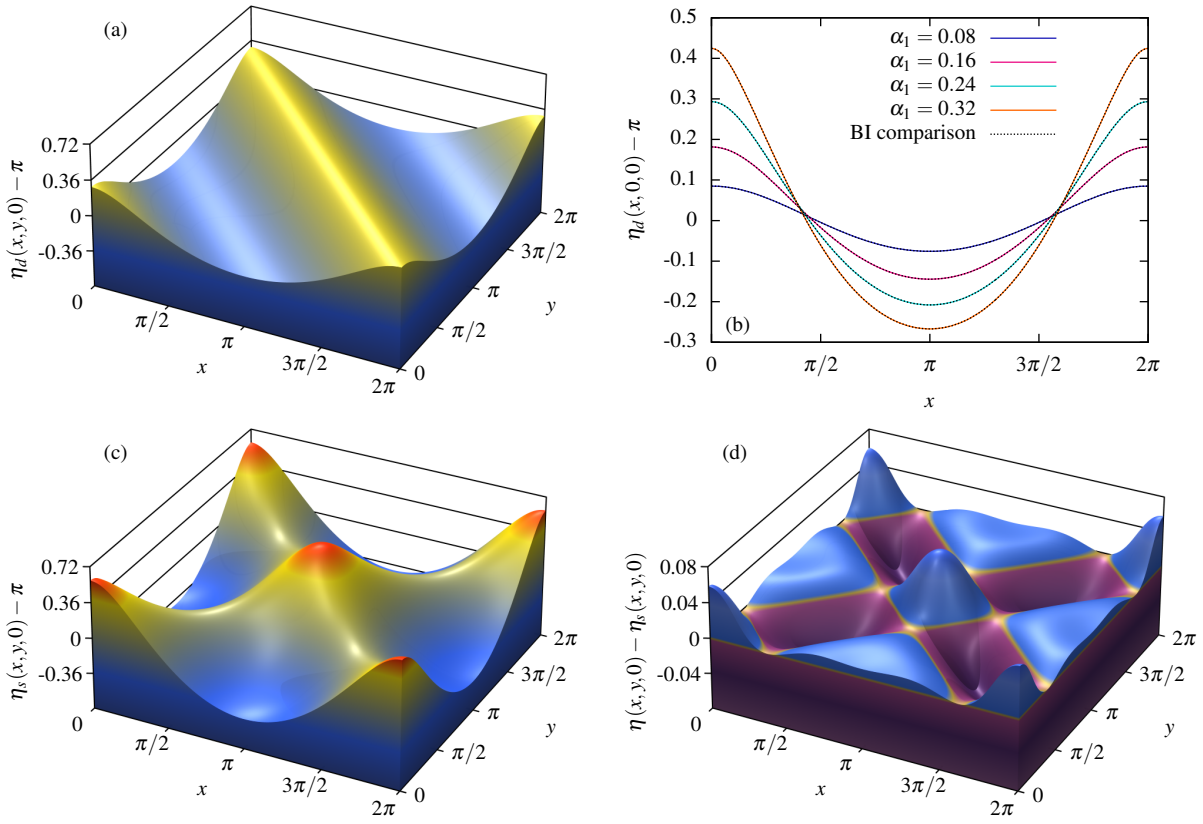

Figure 5: (a) A degenerate standing wave η_d with translational symmetry, found by fixing the mode $0.24\cos(x+y)$ and then searching over modes of the form $\cos(r(x+y))$ for $r>1$; (b) cross-sections of waves with $\alpha_1\cos(x+y)$ held fixed, and a comparison to rescaled, pure 2D standing wave solutions in deep water, computed using the boundary integral (BI) method from Ref. [34]; (c) a linear superposition of two waves η_s, defined as $\eta_s(x,y,0)=\eta_d(x,y,0)+\eta_d(x,-y,0)-\pi$; (d) the difference in height between the superposition η_s and the three-dimensional standing wave shown in Fig. 4(d).

The dynamics of three-dimensional standing waves over a period will now be examined. In Fig. 6(a) to Fig. 6(e) a sequence of plots of the wave height are shown for the case when the $(1,1)$ amplitude is fixed to 0.28. The plots cover the time interval from $t=0$ up to $t=\frac{T}{2}$, when the wave comes to rest again. To confirm this, a plot of the velocity potential at $t=\frac{T}{2}$ is given in Fig. 6(f). This plot has a vertical scale on the order of 10^{-6}, showing that the wave is almost stationary other than for numerical errors, and limitations of the Fourier modes to capture the initial wave shape. For this solution the residual is $S=5.82\times10^{-13}$.

The plot of the wave at $t=\frac{T}{4}$ shown in Fig. 6(c) is not flat as would be the case for the linearized solution, and has small peaks at intervals of $\pi/2$. To investigate this, and the general shape of the wave surface, contour plots of the wave height were plotted. The contours at $t=0$ are shown in Fig. 7(a), with the thick magenta lines corresponding to the contour $\eta(x,y,t)=\pi$, and a spacing of 0.1 between subsequent contours. The positive contours are shown in blue, and exhibit square shapes, corresponding to the pyramidal forms seen in the previous figures. The negative contours shown in cyan are more rounded and further apart, indicating wide and shallow wave troughs. The contours for $t=\frac{T}{4}$ are shown in Fig. 7(b), using a smaller contour spacing of 0.02. Similar patterns

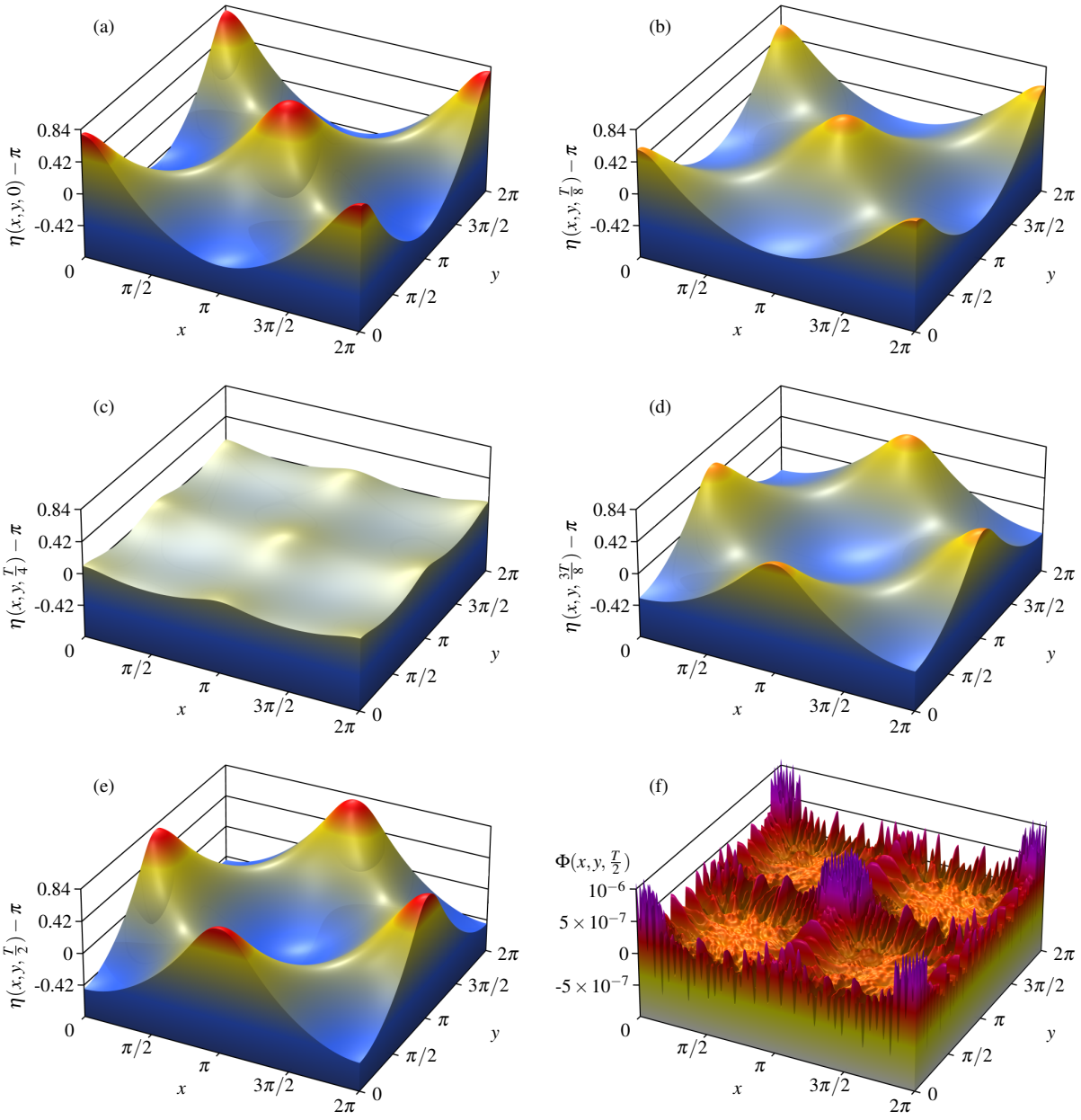

Figure 6: (a–e) Plots of the wave height $\eta(x,y,t)$ over half a period, at $t=0,\frac{T}{8},\frac{T}{4},\frac{3T}{8},\frac{T}{2}$, for the time-periodic wave where the $(1,1)$ mode amplitude is fixed to 0.28; (f) a plot of the velocity potential at $t=\frac{T}{2}$, showing that it is on the order of 10^{-6}, corresponding to the wave being almost stationary.

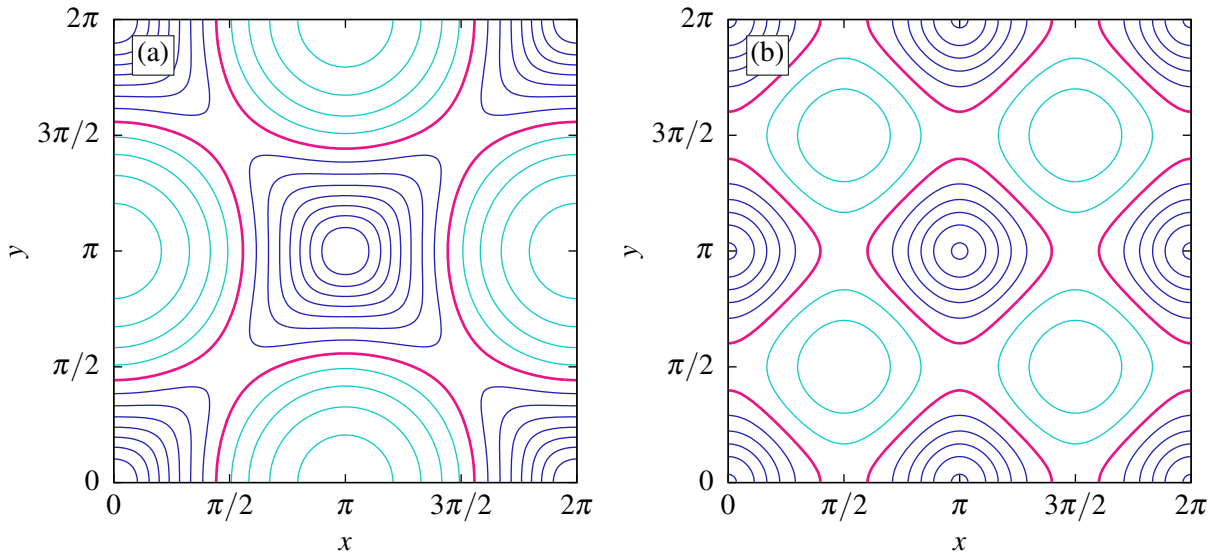

Figure 7: Contour plots of the time-periodic wave where the $(1,1)$ mode is fixed to 0.28 for (a) $t = 0$ using a contour spacing of 0.1, and (b) $t = \frac{T}{4}$ using a contour spacing of 0.02. The contour $\eta(x,y,t) = \pi$ is shown by the thick magenta lines, and positive and negative contours are shown in blue and cyan respectively.

can be seen, whereby the small peaks are pyramidal in shape, although they are rotated by 45°.

Figure 8(a) shows a sequence of cross-sections of the wave height of the same solution along the x axis, at times from $t = 0$ to $t = \frac{T}{2}$. On this axis, the cross-sections look similar to two-dimensional time-periodic waves. In Fig. 8(b), cross-sections on the diagonal ξ axis are shown. At $t = 0$ the cross-section passes through the wave crests and along ridges between them, whereas at $t = \frac{T}{2}$, the cross-section passes through the wave troughs, and over the ridges.

To examine the growth of the Fourier modes that make up the solution, a large sequence of runs was carried out for different values of the $(1,1)$ mode. Figure 9 shows a log–log plot of all of these other modes that are determined using the optimization procedure, in terms of the fixed $(1,1)$ mode. As expected, the other modes grow in size as the $(1,1)$ mode increases, so that at higher amplitudes more modes are needed to accurately represent the solution. Four computational regimes were considered. Below a $(1,1)$ amplitude of 0.1, 41 modes with $r,s < 12$ were used, leading to wall-clock times of two days for a 16-thread computation. Above an amplitude of 0.08, 63 modes with $r,s < 15$ were used, requiring three days of computation time. Above an amplitude of 0.2, 109 modes with $r,s < 20$ were used, requiring five days of computation time. Above an amplitude of 0.26, the computation was switched to a $256 \times 256 \times 65$ grid, which showed near-identical results for modes larger than 10^{-6}, and improved the accuracy for smaller modes.

An interesting feature of Fig. 9 is that while all of the modes initially start out as positive, some begin to switch to negative at higher amplitudes. To study this in more detail, a bar chart of the sizes of the modes is shown in Fig. 10 when the $(1,1)$ mode amplitude is fixed to 0.1. At this amplitude, the modes decay with a regular pattern as the wave number is increased. Physically, if all the mode amplitudes are positive, then each mode sharpens the initial wave crest at the origin

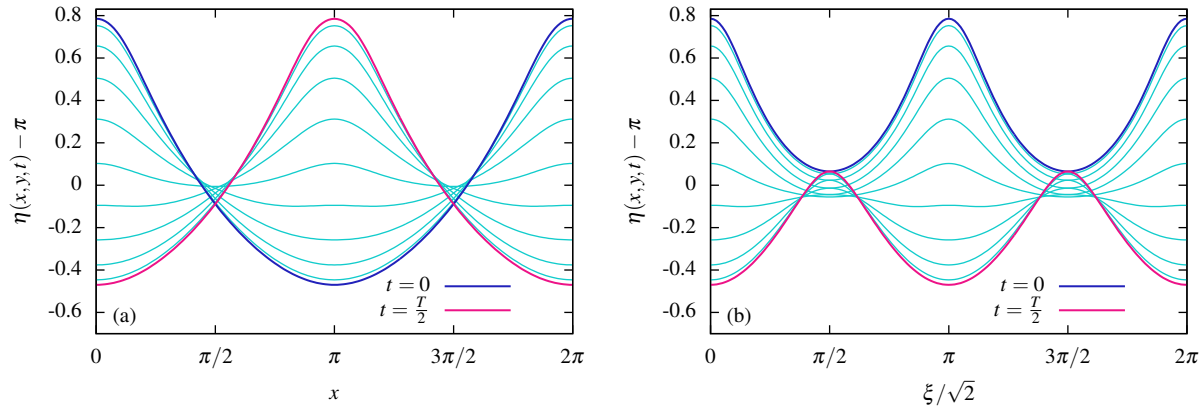

Figure 8: Cross sections of the height of the time-periodic wave in the x-direction (left) and diagonal ξ direction (right). The $(1,1)$ mode is fixed at 0.28. The cyan curves are at intervals of $\frac{T}{20}$.

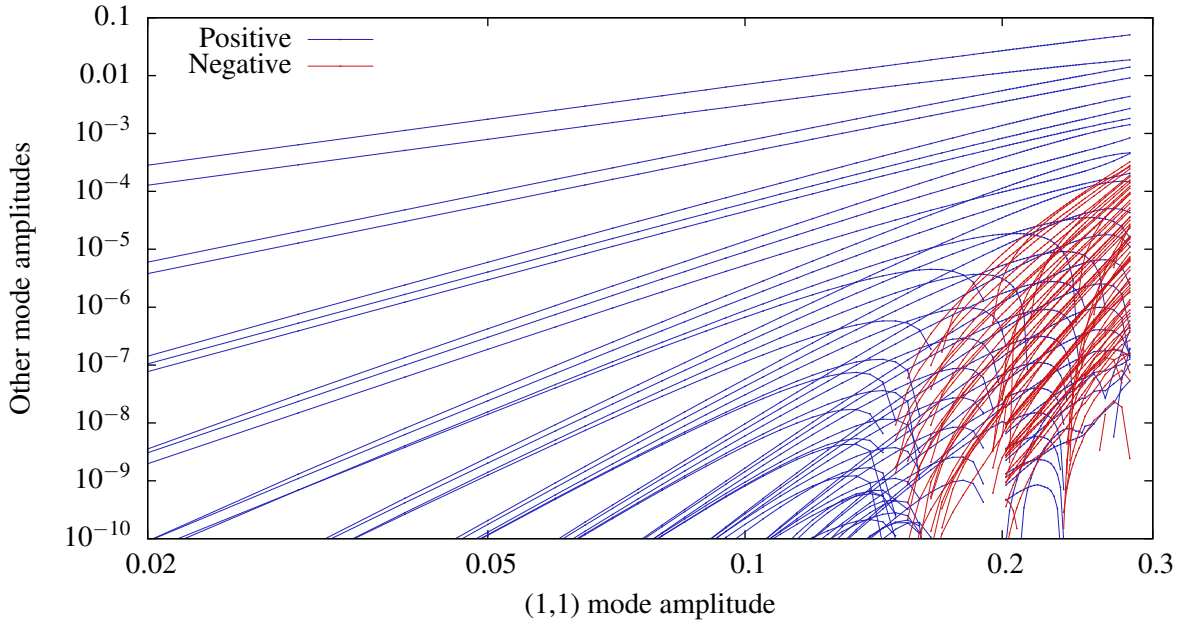

Figure 9: Log–log plot showing the growth of the other modes that are determined using the optimization procedure, as a function of the fixed (1,1) mode. Modes that become negative are shown in red.

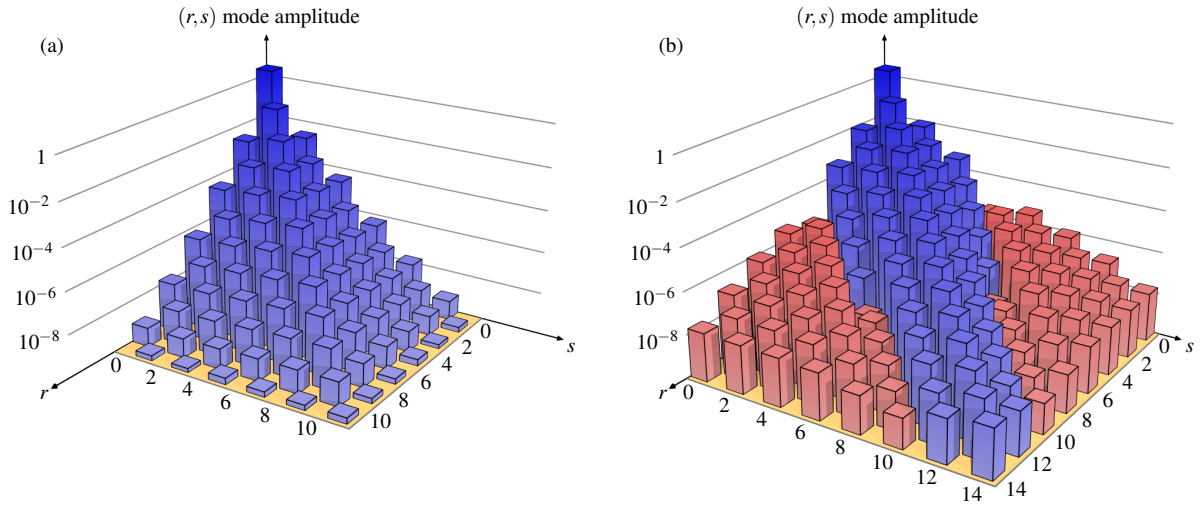

Figure 10: Bar charts of the mode sizes calculated using the optimization procedure for the cases when the (1,1) mode amplitude is fixed to (a) 0.1 and (b) 0.2. Positive modes are shown in blue, and negative modes are shown in red. The (r,s) mode amplitudes for $r \le s$ are determined by the optimization procedure and exactly match vertical cross-sections of the data in Fig. 9. By symmetry, each (r,s) mode amplitude for $r > s$ is plotted as being equal to the value at (s,r).

since all the terms have negative mean (and positive Gaussian) curvature there. However, when the $(1,1)$ mode amplitude is increased to 0.2, the modes with large values of $|r-s|$ change sign and have a slight flattening effect against the remaining (larger) positive terms that sharpen the crest. We do not know the geometric significance of this sign pattern beyond observing that in Fig. 7 above, the contour lines near the crests flatten into square-like shapes while the troughs remain closer to circular. Figures 9 and 10 show that for the entire sequence of runs, the Fourier modes appear to grow smoothly in magnitude as the amplitude of the wave increases, and decay exponentially as r and s increase for fixed amplitude. Resonant effects lead to mildly erratic behavior in the smaller-amplitude Fourier modes of Fig. 10(b). This is expected by analogy with theoretical results on two-dimensional standing waves [29, 30], which have only been proved to exist for values of an amplitude parameter in a totally disconnected Cantor set.

Figure 11(a) shows a plot of the wave period T versus the $(1,1)$ mode amplitude, indicating that as the $(1,1)$ amplitude is increased, the period increases slightly. This can be compared to the second-order asymptotic expansion for the period that was calculated by Verma and Keller [21]. For the case of a wave in a square domain, the angular frequency is given by

$$\omega = \omega_0 + 2\alpha_{1,1}^2 \omega_2 + O(\alpha_{1,1}^3), \tag{28}$$

where $\alpha_{1,1}$ is the (1,1) mode amplitude, $\omega_0^2 = \sqrt{2}\tanh(\sqrt{2}H)$ in agreement with Eq. 23, and

$$\omega_2 = -\frac{(3\omega_0^4 - 2)^2}{16\omega_0(\tanh 2H - 2\omega_0^2)} + \frac{72\omega_0^{-7} - 48\omega_0^{-3} + 10\omega_0 - 46\omega_0^5}{128}. \tag{29}$$

For a depth of $H = \pi$ these formulae give $\omega_0 = 1.189043$ and $\omega_2 = -0.357535$. As can be seen in Fig. 11(a), this is in close agreement with the computed results although some deviations become visible at higher amplitudes.

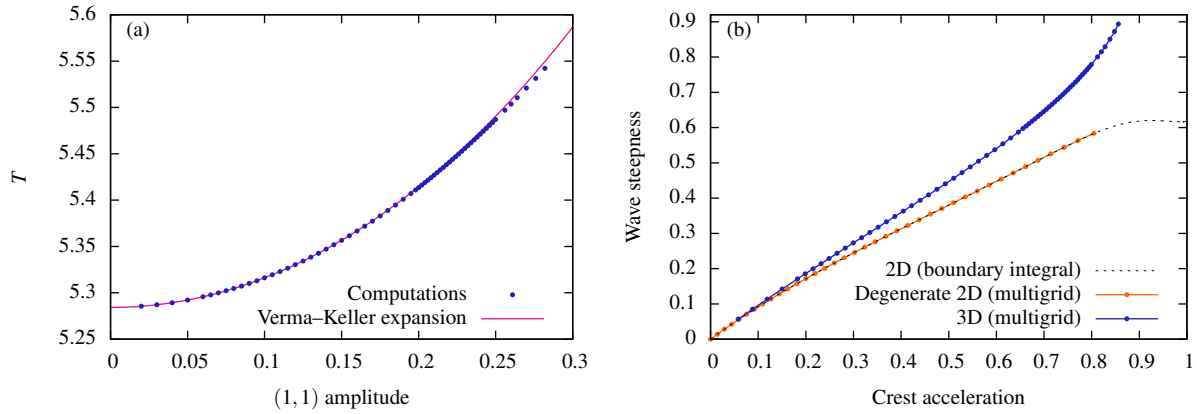

Figure 11: Plots of (a) period versus size of the fixed $(1,1)$ mode amplitude, compared against the asymptotic expansion of Verma and Keller [21], and (b) crest acceleration versus wave steepness.

In studies of two-dimensional standing waves [14, 20], it has been useful to examine the relationship between the initial downward acceleration of the crest, and the wave steepness, defined as half of the crest-to-trough height difference. The dashed line in Fig. 11(b) is based on two-dimensional data from Ref. [20], and shows that wave steepness initially grows linearly with crest acceleration before flattening out and passing through a turning point at a crest acceleration of 0.93. As discussed in Ref. [20], for crest accelerations beyond 0.99, the deep water bifurcation curve fragments into a number of disjoint branches corresponding to different resonant mode patterns that emerge near the crest tip due to complex dynamics. However, these features are too small to be seen in Fig. 11(b), and occur in a regime that we cannot currently reach with our three-dimensional code. The corresponding data for the family of degenerate 2D standing waves that were considered is also shown, with the wave steepnesses multiplied by a factor of $1/\sqrt{2}$ due to the aforementioned difference in wavelength. The two methods agree to approximately 4–5 digits of accuracy.

The data for the three-dimensional standing waves is also shown. For compatibility with the 2D data, and since the three-dimensional waves have a wavelength of $2\pi/\sqrt{2}$ in the ξ direction, the three-dimensional wave steepness was also multiplied by $1/\sqrt{2}$. Initially the wave steepness grows approximately linearly, but at high crest accelerations the opposite trend to the 2D data is seen and the slope of the curve increases with crest acceleration. This suggests that the limiting behavior of three-dimensional waves may be fundamentally different than in two dimensions. Physically, when the fluid converges to form a wave crest in three dimensions, it generates a stronger fluid jet pushing the surface upward. This is consistent with Fig. 5(d), which shows that three-dimensional wave crests are higher than would be expected from a superposition of two-dimensional standing waves oriented orthogonally to each other.

Using the same depth of fluid, a second case was considered with less symmetry, using an alternative Fourier basis of

$$f^*_{r,s}(x,y) = \cos rx \cos 2sy. \tag{30}$$

An example solution in which the $f^*_{1,2}$ mode amplitude is fixed to 0.2 is shown in Fig. 12(a). In

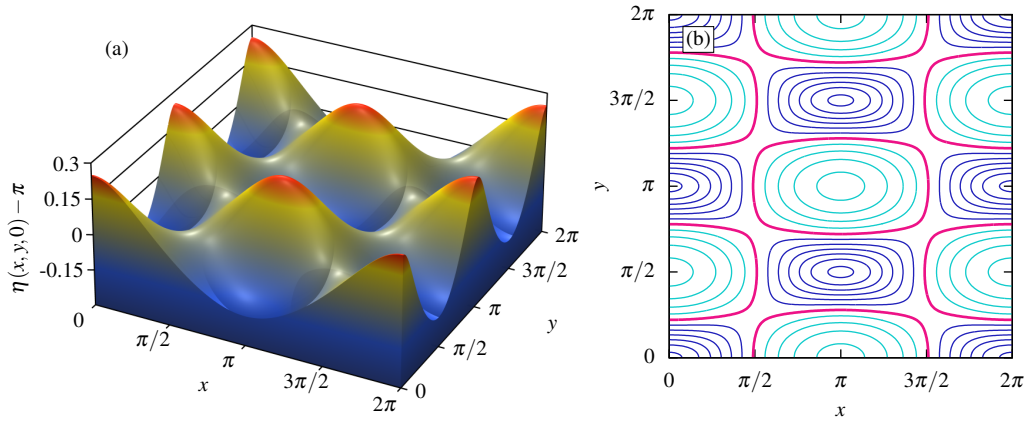

Figure 12: (a) A time-periodic wave for the case when the Fourier mode $\cos x \cos 2y$ is fixed to amplitude 0.2; (b) a contour plot of the wave height, where the magenta curves correspond to $\eta(x,y,0) = \pi$, the blue curves are positive contours and cyan curves are negative contours. The contour spacing is 0.04.

a similar manner as the previous case, the wave crests are noticeably sharper than the linearized solution. It also appears that the crests begin to take on a wedge-like shape. A contour plot shown in Fig. 12(b) also highlights this behavior.

3.2. Waves in shallow water

In this section, waves with a much shallower fluid depth of $\pi/6$ are considered. The symmetric Fourier basis given by Eq. 25 is used, although here, the $(0,1)$ mode corresponding to $\cos x + \cos y$ is fixed. From Eq. 23, the linearized period is 9.06453, with the hyperbolic tangent term having a significant effect. Unlike the previous case, the mode that is varied is not invariant under a translation of (π,π). In searching for periodic solutions, it is therefore necessary to consider all modes (r,s) with $r \geq s$ without imposing the constraint that $r+s$ is even. In a similar manner to the previous case, initial solutions close to the linearized regime are first determined, making use of a $256 \times 256 \times 17$ grid and considering 152 degrees of freedom up to $r,s < 17$. After this, Eq. 24 is used to move to successively larger amplitudes of the fixed mode.

Figure 13 shows a sequence of plots of the wave height for the case when the $(0,1)$ amplitude is fixed to 0.14. The plots share a number of features with those in Fig. 6, with sharp pyramidal wave crests and ridges between adjacent crests. However, the troughs are wider and flatter, in a similar manner to those seen in two-dimensional studies. Fig. 13(f) shows the velocity potential at $t = \frac{T}{2}$, using a vertical scale on the order of 10^{-5}; the computed residual is $S = 3.95 \times 10^{-10}$. Cross sections of the wave height are shown in Fig. 14, which highlight the flat wave troughs between successive peaks.

In a similar manner to the intermediate depth fluid, the wave shapes can be compared with their two-dimensional counterparts. We present this analysis at $t = \frac{T}{2}$, since at that time the wave crest is in the middle of the simulation domain and easier to visualize. We denote the degenerate basis functions by $f_r^\dagger(x,y) = \cos(rx)$. Figure 15(a) shows a plot of a standing wave η_d at $t = \frac{T}{2}$ for the

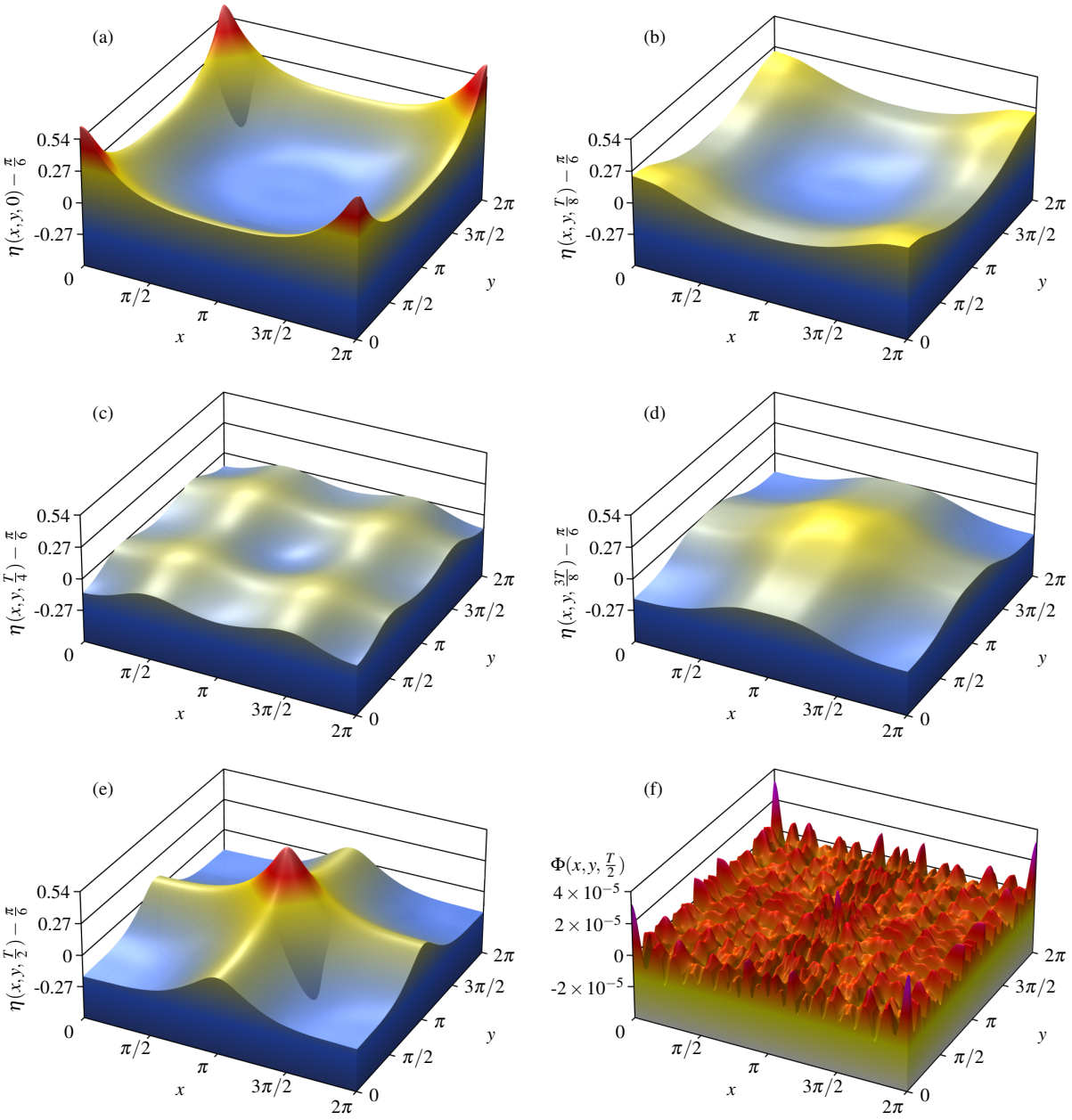

Figure 13: (a–e) Plots of the wave height of a standing wave in a shallow fluid depth of $\pi/6$ at $t = 0, \frac{T}{8}, \frac{T}{4}, \frac{3T}{8}, \frac{T}{2}$ for the case when the $(0,1)$ mode amplitude is fixed to 0.14; (f) a plot of the velocity potential at $t = \frac{T}{2}$, showing that it is on the order of 10^{-5}, corresponding to the wave being almost stationary at that instant.

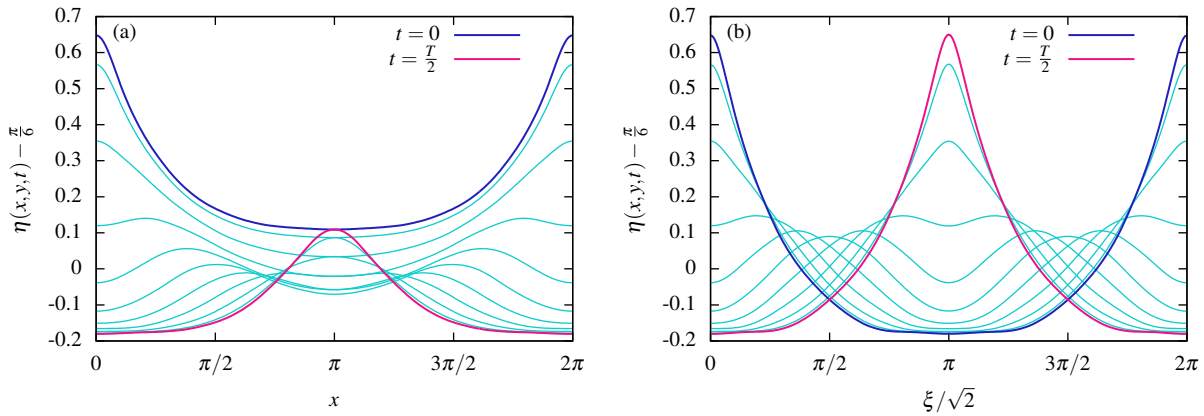

Figure 14: Cross sections of the height of the time-periodic wave where the $(0,1)$ mode is fixed to 0.14 in the (a) x direction, and (b) diagonal ξ direction. The cyan curves are at intervals of $\frac{T}{20}$.

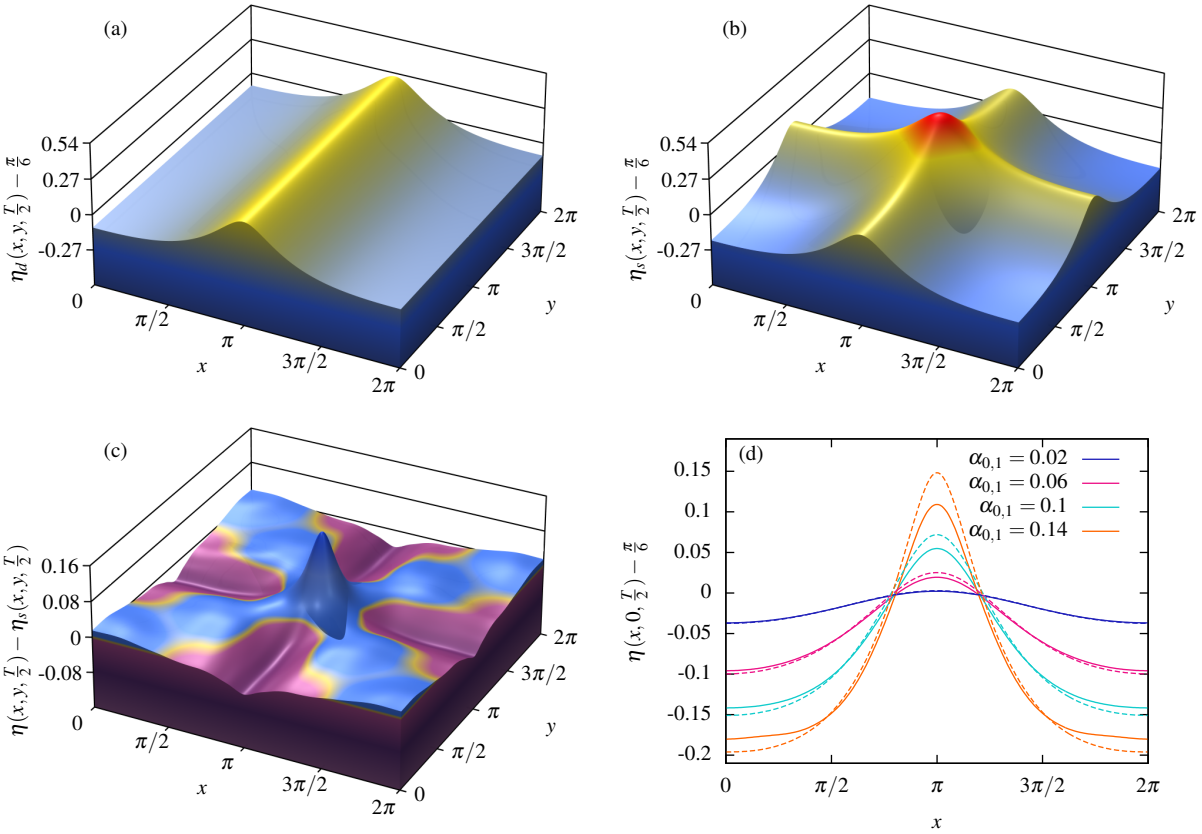

Figure 15: (a) A degenerate standing wave η_d in a shallow fluid of depth $\pi/6$, found by fixing the $0.14\cos x$ mode and minimizing over modes of the form $\cos rx$ for $r>1$, plotted at $t=\frac{T}{2}$; (b) a linear superposition of two degenerate waves, defined as $\eta_s(x,y,\frac{T}{2})=\eta_d(x,y,\frac{T}{2})+\eta_d(y,x,\frac{T}{2})-\frac{\pi}{6}$; (c) difference between height of the three-dimensional wave shown in Fig. 13 and the linear superposition η_s; (d) cross-sections of the ridges for three-dimensional waves for various values of the $(0,1)$ mode $\alpha_{0,1}$, shown in solid lines, compared to the equivalent data for a superposition of degenerate two-dimensional waves η_s in dashed lines.

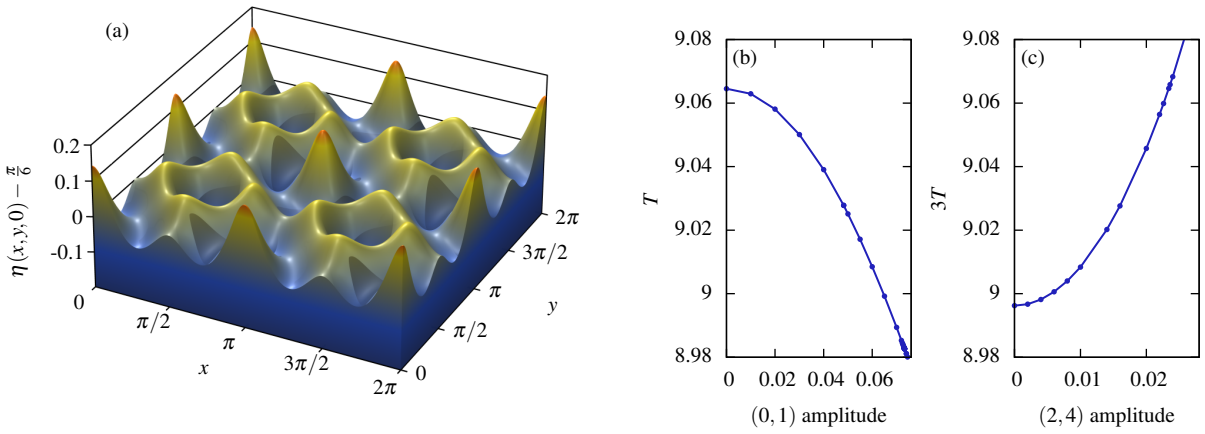

Figure 16: (a) Plot of the wave height for a time-periodic wave in a shallow fluid of depth $\pi/6$, for the case when the $(2,4)$ mode amplitude is fixed at 0.028; (b) plot of the period for the case when the $(0,1)$ mode amplitude is varied, corresponding to solutions of the type shown in Fig. 13; (c) plot of three times the period of solutions on the $(2,4)$ branch, such as the wave shown at left. Plots (b) and (c) share the same vertical axis, indicating the possibility of finding a combined solution incorporating significant $(0,1)$ and $(2,4)$ components.

case when the amplitude of $f_1^\dagger$ is fixed to 0.14, and a solution is searched for over modes $f_r^\dagger$ for $r = 2,3,\ldots,19$. A superposition can then be constructed as

$$\eta_s\left(x,y,\tfrac{T}{2}\right) = \eta_d\left(x,y,\tfrac{T}{2}\right) + \eta_d\left(y,x,\tfrac{T}{2}\right) - \tfrac{\pi}{6}, \tag{31}$$

and is plotted in Fig. 15(b). While similar to the three-dimensional counterpart shown in Fig. 14(e), the wave crest of η_d is noticeably less pronounced. This can be confirmed by plotting the difference between the two, as shown in Fig. 15(c), which has similar features to Fig. 5(d) with the crests being higher and the ridges being lower. Figure 15(d) shows a quantitative comparison between the ridge shape in three dimensions and the corresponding linear superpositions of two-dimensional standing waves.

In the study of two-dimensional waves, it has been shown that there are many more complex periodic solutions, which often occur due to resonances between different modes. A good candidate for resonant solutions in three dimensions would combine the $(2,4)$ and $(0,1)$ modes. Small-amplitude $(2,4)$ standing waves have a period of 2.9987, which is approximately a third of the period of small-amplitude $(0,1)$ waves. Figure 16 shows a typical solution on the $(2,4)$ branch, along with the periods T and $3T$ versus amplitude on the $(0,1)$ and $(2,4)$ branches, respectively. Since the two curves cross, there is a good chance that a combined solution exists with large $(0,1)$ and $(2,4)$ components.

Using this data, we constructed a guess for a combined solution by superposing the solution with the $(2,4)$ amplitude fixed to 0.016, and the solution with the $(0,1)$ amplitude fixed to 0.04822, which are both time-periodic with a period of approximately 9.027. From this initial guess, the minimization procedure was successful in finding a combined solution. The initial surface of this solution is shown in Fig. 17(b). From this solution, it was possible to use the extrapolation procedure to trace out a new family of solutions. In one direction, the $(2,4)$ amplitude increases, and the $(0,1)$

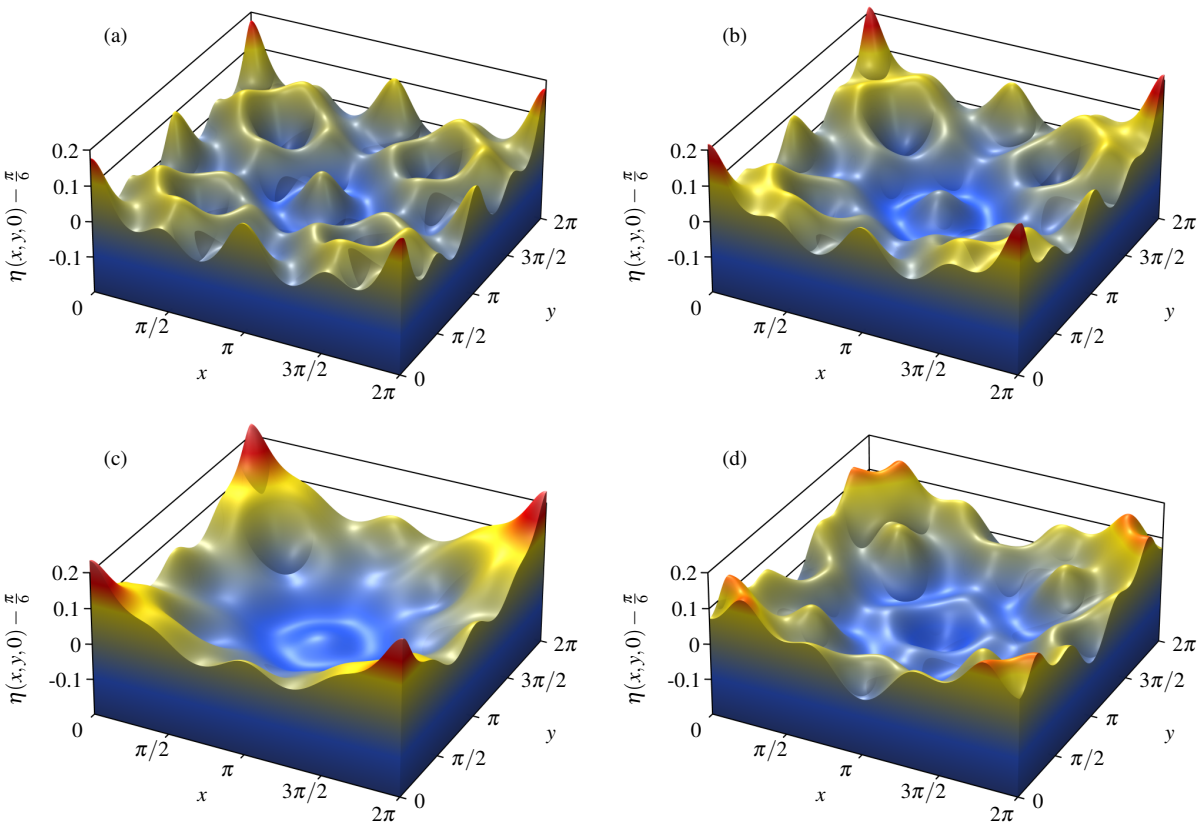

Figure 17: Plots of the initial wave height for several time-periodic solutions featuring both a $(0,1)$ component and a $(2,4)$ component. The $(0,1)$ amplitude is (a) 0.0280, (b) 0.0482, (c) 0.0680, (d) 0.0540. The $(2,4)$ amplitude is (a) 0.0207, (b) 0.0159, (c) 0.0072, (d) -0.0140.

amplitude decreases; a typical solution on this path is shown in Fig. 17(a). In the opposite direction, the $(2,4)$ amplitude decreases and the $(0,1)$ amplitude increases, and a typical solution is shown in Fig. 17(c). Eventually, as the $(2,4)$ amplitude decreases further, this branch of solutions meets the branch of pure $(0,1)$ solutions. A further branch can be found where the $(2,4)$ amplitude becomes negative, as shown in Fig. 17(d).

Figure 18 shows a plot of the different families of solutions and their connections in terms of the $(0,1)$ and $(2,4)$ mode amplitudes. The basic $(0,1)$ branch of solutions is shown in blue; at large amplitudes, the line becomes slightly curved, indicating a non-negligible correction term from the $(2,4)$ mode. The $(2,4)$ branch forms a perfect vertical line, since a $(0,1)$ contribution would break symmetry. The solutions forming the combined mode, shown in yellow, form an arc connecting the pure mode branches together. The solutions plotted in Figs. 17(a–c) are on the branch in the top right quadrant, while the solution in Fig. 17(d) is on the branch in the bottom right quadrant.

The gray lines in the plot can be deduced using symmetry. Given any solution, another solution must exist when the height field is translated by (π,π), which will flip the sign of the $(0,1)$ mode, and leave the $(2,4)$ mode invariant. This can be used to deduce all of the solutions in the left half of

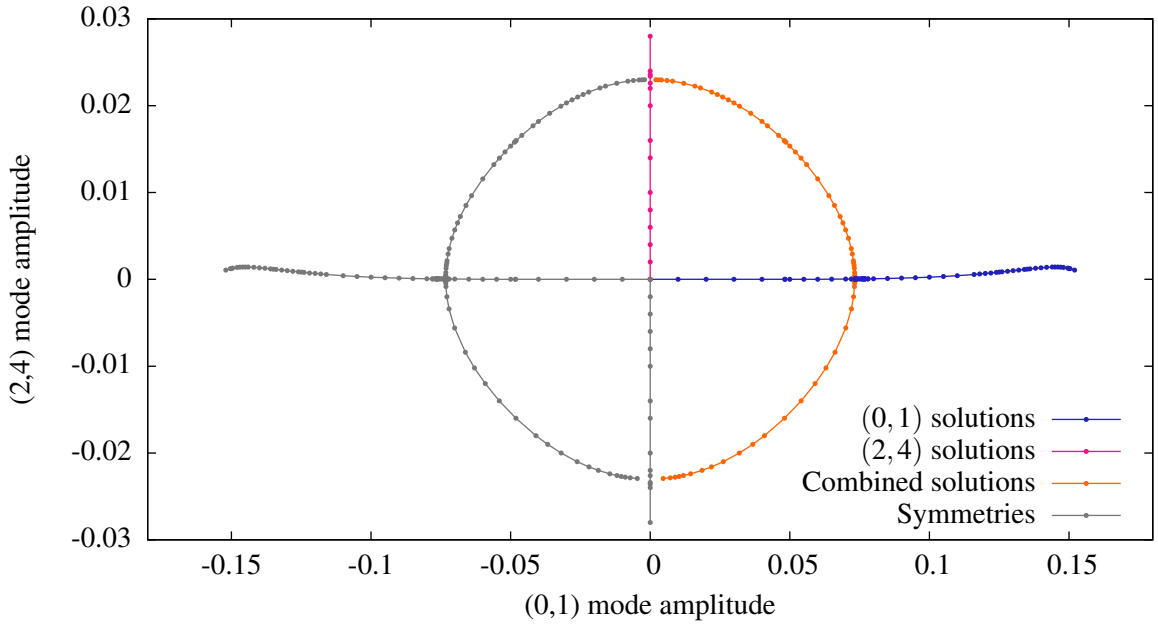

Figure 18: Plots of several families of admissible periodic solutions for a fluid depth of $\pi/6$, in terms of the $(0,1)$ and $(2,4)$ mode amplitudes. Dots represent individual solutions, each requiring approximately a week of computation time using sixteen threads. Colored lines represent solutions that were calculated, while gray lines can be deduced from symmetries.

the graph. For a pure $(2,4)$ mode, another valid solution is obtained from translation by $(\pi/2,\pi/2)$, which can be used to deduce the branch of solutions pointing downward from the origin. With these symmetries accounted for, a complete ring of combined modes becomes apparent, intersecting the pure mode branches in four locations.

We believe that the intersections of the mixed mode branches with the $(2,4)$ branches are perfect bifurcations while those with the $(0,1)$ branches are imperfect bifurcations. At the critical depth $H \approx 1.020325\pi/6$, the period of small-amplitude $(0,1)$ mode solutions is exactly three times that of small-amplitude $(2,4)$ mode solutions:

$$T = \frac{2\pi}{\sqrt{\tanh H}} = \frac{6\pi}{\sqrt{\kappa \tanh \kappa H}}, \qquad \kappa = \sqrt{20}. \tag{32}$$

In our simulations, if the depth were varied from $\pi/6$ to this critical depth, we believe the four secondary bifurcations from pure to mixed-mode solutions would coalesce at the origin, leading to a degenerate bifurcation with eight rays leaving the origin. The four mixed mode rays would correspond to a Wilton's ripple phenomenon [50, 51, 52, 53, 54] in which multiple wavelengths are present in the leading order asymptotics. Although this is speculation, it is consistent with recent finding in the two-dimensional case [34], where it was computationally feasible to resolve the degenerate bifurcation and study how it splits under perturbation of the fluid depth. It is also consistent with Bridges' analysis [22] of degenerate bifurcations for three-dimensional water waves

in the weakly nonlinear regime.

4. Conclusion

In this paper, we have demonstrated the existence of large-amplitude, three-dimensional, time-periodic water waves. The computations have made use of several high-performance algorithms, plus advances in computational power, to make them tractable within a reasonable time frame. The three-dimensional waves that we have computed bear many similarities with their two-dimensional counterparts, and become more sharply crested as the amplitude is increased. However, there appear to be significant differences, most notably in the relation between crest acceleration and wave height. While our computations currently lack the resolution to examine the crest formation in more detail, this result suggests that the behavior as the crest continues to sharpen may be significantly different from the behavior in the two-dimensional case [20].

Our results have also been able to examine the shapes of the wave surfaces. For the $(1,1)$ mode, pyramidal-shaped crests connected by ridges are observed, whereas for the $f^*_{1,2}$ mode defined in Eq. 30, the crests appear to become wedge-like. Both of these results suggest that interesting behavior may be seen if higher amplitude waves can be calculated. In shallow water, we have demonstrated the existence of several families of solutions that are connected by bifurcations.

The results that we have presented represent only a very small subset of the time-periodic waves that are likely to exist. We have mainly been limited by computation time: since a single solution takes five to seven days to compute, finding families of solutions requires many weeks of computation. However, in the future, we aim to use the same methodology to search for other types of time-periodic solutions, such as standing waves with less symmetry or with larger amplitude. Searching for three-dimensional solitary waves and breathers, as well as pattern formation in Faraday waves [38, 39, 40], are other avenues to consider.

Acknowledgments

This work was supported by the Director, Office of Science, Computational and Technology Research, U.S. Department of Energy under contract number DE-AC02-05CH11231, and by the the National Science Foundation through grant DMS-0955078.

Appendix A. Derivation of the variational problem

To derive the variational partial differential equation system given by Eqs. 19 and 20, a solution (η_*, φ_*) for the height and velocity potential can be considered where $\eta_* = \eta + \varepsilon\dot{\eta}$ and $\varphi_* = \varphi + \varepsilon\dot{\varphi}$. Here, ε is a small parameter, (η, φ) is a solution of the original partial differential equation system, and $(\dot{\eta}, \dot{\varphi})$ is a small variation. In a similar manner the bulk velocity potential can be written as $\phi_* = \phi + \varepsilon\dot{\phi}$. Since $\nabla^2\phi_* = \nabla^2\phi = 0$, it follows that $\nabla^2\dot{\phi} = 0$ also. The surface velocity potential satisfies $\varphi_*(x,y,t) = \phi_*(x,y,\eta_*(x,y,t),t)$ and expanding both sides of this equation gives

$$\begin{aligned}
\varphi(x,y,t) + \varepsilon\dot{\varphi}(x,y,t) &= \phi(x,y,\eta(x,y,t) + \varepsilon\dot{\eta}(x,y,t),t) \\
&= \phi(x,y,\eta(x,y,t),t) + \varepsilon\phi_z(x,y,\eta(x,y,t),t)\dot{\eta}(x,y,t) + O(\varepsilon^2) \\
&= \varphi(x,y,t) + \varepsilon\dot{\varphi}(x,y,t) + \varepsilon\phi_z(x,y,\eta(x,y,t),t)\dot{\eta}(x,y,t) + O(\varepsilon^2).
\end{aligned}$$

By examining the terms of order ε it can be seen that $\dot{\varphi} = \dot{\phi} + \phi_z \dot{\eta}$. Substituting the expressions for η and φ into Eq. 1 and keeping terms of order ε yields

$$\begin{aligned}
\dot{\eta}_t &= \dot{\phi}_z + \phi_{zz}\dot{\eta} - \dot{\eta}_x\phi_x - \eta_x\dot{\phi}_x - \eta_x\phi_{xz}\dot{\eta} - \dot{\eta}_y\phi_y - \eta_y\dot{\phi}_y - \eta_y\phi_{yz}\dot{\eta} \\
&= \dot{\phi}_z - \eta_x\dot{\phi}_x - \eta_y\dot{\phi}_y - (\dot{\eta}\phi_{xx} + \dot{\eta}_x\phi_x + \eta_x\phi_{xz}\dot{\eta}) - (\dot{\eta}\phi_{yy} + \dot{\eta}_y\phi_y + \eta_y\phi_{yz}\dot{\eta}) \\
&= \dot{\phi}_n\sqrt{1+\eta_x^2+\eta_y^2} - (\dot{\eta}\phi_x)_x - (\dot{\eta}\phi_y)_y.
\end{aligned}$$

Here, $\dot{\phi}_{zz} = -\dot{\phi}_{xx} - \dot{\phi}_{yy}$ is used, and the subscripts on the bracketed terms denote partial derivatives of the terms when they are viewed as a function restricted to the surface.

To derive the variational equation for the surface velocity potential, Eq. 1 is first substituted into Eq. 3 to remove the η_t term, giving

$$\varphi_t = P\left[-\eta_x\phi_x\phi_z - \eta_y\phi_y\phi_z + \frac{\phi_z^2 - \phi_x^2 - \phi_y^2}{2} - g\eta\right].$$

Substituting the expressions for η_* and φ_* yields

$$\begin{aligned}
\dot{\varphi}_t &= P\Big[-\dot{\eta}_x\phi_x\phi_z - \eta_x\dot{\phi}_x\phi_z - \eta_x\phi_x\dot{\phi}_z - \eta_x\phi_{xz}\dot{\eta}\phi_z - \eta_x\phi_x\phi_{zz}\dot{\eta} \\
&\quad -\dot{\eta}_y\phi_y\phi_z - \eta_y\dot{\phi}_y\phi_z - \eta_y\phi_y\dot{\phi}_z - \eta_y\phi_{yz}\dot{\eta}\phi_z - \eta_y\phi_y\phi_{zz}\dot{\eta} \\
&\quad +\phi_z(\dot{\phi}_z + \phi_{zz}\dot{\eta}) - \phi_x(\dot{\phi}_x + \phi_{xz}\dot{\eta}) - \phi_y(\dot{\phi}_y + \phi_{yz}\dot{\eta}) - g\dot{\eta}\Big].
\end{aligned}$$

This expression features seventeen terms. Ten of the terms can be combined by making use of

$$\begin{aligned}
-(\dot{\eta}\phi_x\phi_z)_x &= -\dot{\eta}_x\phi_x\phi_z - \dot{\eta}\phi_{xx}\phi_z - \dot{\eta}\phi_x\phi_{xz} - \dot{\eta}\phi_{xz}\phi_z\eta_x - \dot{\eta}\phi_x\phi_{zz}\eta_x, \\
-(\dot{\eta}\phi_y\phi_z)_y &= -\dot{\eta}_y\phi_y\phi_z - \dot{\eta}\phi_{yy}\phi_z - \dot{\eta}\phi_y\phi_{yz} - \dot{\eta}\phi_{yz}\phi_z\eta_y - \dot{\eta}\phi_y\phi_{zz}\eta_y.
\end{aligned}$$

Six of the terms feature one derivative of ϕ and one of $\dot{\phi}$, which can be expressed as a matrix product. Hence the equation can be written as

$$\dot{\varphi}_t = P[-(\dot{\eta}\phi_x\phi_z)_x - (\dot{\eta}\phi_y\phi_z)_y - K - g\dot{\eta}]$$

where

$$K = \begin{pmatrix} \phi_x & \phi_y & -\phi_z \end{pmatrix} \begin{pmatrix} 1 & 0 & \eta_x \\ 0 & 1 & \eta_y \\ -\eta_x & -\eta_y & 1 \end{pmatrix} \begin{pmatrix} \dot{\phi}_x \\ \dot{\phi}_y \\ \dot{\phi}_z \end{pmatrix}.$$

Appendix B. Finite element computation

Here, we consider the computation of the integral in Eq. 7. The determinant of D is $h^2\eta/N$ and the inverse is

$$D^{-1} = \begin{pmatrix} h^{-1} & 0 & -\frac{\sum_{ij} L_i'(\alpha)L_j(\beta)(n_z+\gamma)}{h\eta} \\ 0 & h^{-1} & -\frac{\sum_{ij} L_i(\alpha)L_j'(\beta)(n_z+\gamma)}{h\eta} \\ 0 & 0 & N/\eta \end{pmatrix}.$$

Substituting this expression into Eq. 7 yields

$$\begin{aligned} A_{12} &= \int_U \begin{pmatrix} \partial_\alpha T_1 & \partial_\beta T_1 & \partial_\gamma T_1 \end{pmatrix} \left(D^{-1}\right)^T (\det D) D^{-1} \begin{pmatrix} \partial_\alpha T_2 \\ \partial_\beta T_2 \\ \partial_\gamma T_2 \end{pmatrix} d^3\alpha \\ &= \int_U \begin{pmatrix} \partial_\alpha T_1 & \partial_\beta T_1 & \partial_\gamma T_1 \end{pmatrix} \begin{pmatrix} I & 0 & J \\ 0 & I & K \\ J & K & M \end{pmatrix} \begin{pmatrix} \partial_\alpha T_2 \\ \partial_\beta T_2 \\ \partial_\gamma T_2 \end{pmatrix} d^3\alpha \end{aligned} \tag{B.1}$$

where

$$I(\alpha,\beta,\gamma) = \frac{\eta}{N}, \tag{B.2}$$

$$J(\alpha,\beta,\gamma) = -\frac{n_z+\gamma}{N}\sum_{ij}\eta_{ij}L_i'(\alpha)L_j(\beta), \tag{B.3}$$

$$K(\alpha,\beta,\gamma) = -\frac{n_z+\gamma}{N}\sum_{ij}\eta_{ij}L_i(\alpha)L_j'(\beta), \tag{B.4}$$

$$M(\alpha,\beta,\gamma) = \frac{(n_z+\gamma)^2}{\eta N}\left(\left(\sum_{ij}\eta_{ij}L_i'(\alpha)L_j(\beta)\right)^2 + \left(\sum_{ij}\eta_{ij}L_i(\alpha)L_j'(\beta)\right)^2\right) + \frac{Nh^2}{\eta}. \tag{B.5}$$

It could be possible that the integral in Eq. B.1 would have to be evaluated using quadrature in three dimensions, carrying out a separate calculation for each different value of n_z. However, the terms in the integral are structured in a manner that allows the calculation to be significantly simplified. The integral can be decomposed into $A_{12} = A_{12}^I + A_{12}^J + A_{12}^K + A_{12}^M$, where each of these four terms corresponds the part involving the corresponding function in Eqs. B.2–B.5. To begin,

$$\begin{aligned} A_{12}^I &= \int_U \frac{\eta(\alpha,\beta)}{N}(\partial_\alpha T_1\,\partial_\alpha T_2 + \partial_\beta T_1\,\partial_\beta T_2)\,d^3\alpha \\ &= \frac{1}{N}\sum_{ij}\eta_{ij}\int_U L_i(\alpha)L_j(\beta)\Big(L_a'(\alpha)L_b(\beta)L_c(\gamma)L_d'(\alpha)L_e(\beta)L_f(\gamma) \\ &\quad + L_a(\alpha)L_b'(\beta)L_c(\gamma)L_d(\alpha)L_e'(\beta)L_f(\gamma)\Big)\,d^3\alpha. \end{aligned}$$

If the quantities

$$\begin{aligned} R_{ij} &= \int_0^4 L_i(\omega)L_j(\omega)d\omega, \\ S_{ijk} &= \int_0^4 L_i(\omega)L_j(\omega)L_k(\omega)d\omega, \\ T_{ijk} &= \int_0^4 L_i(\omega)L_j'(\omega)L_k'(\omega)d\omega \end{aligned}$$

are introduced, then A_{12}^I becomes

$$A_{12}^I = \frac{1}{N}\sum_{ij}\eta_{ij}R_{cf}(T_{iad}S_{jbe} + S_{iad}T_{jbe}).$$

Since R, S and T are integrals of polynomials, they can be computed exactly beforehand. The next term to consider is

$$\begin{aligned} A_{12}^J &= -\frac{1}{N}\sum_{ij}\eta_{ij}\int_U (n_z+\gamma)L_i'(\alpha)L_j(\beta)\Big(L_a'(\alpha)L_b(\beta)L_c(\gamma)L_d(\alpha)L_e(\beta)L_f'(\gamma) \\ &\quad + L_a(\alpha)L_b(\beta)L_c'(\gamma)L_d'(\alpha)L_e(\beta)L_f(\gamma)\Big)\,d^3\alpha. \end{aligned}$$

With the additional pre-computed quantities

$$\begin{aligned} U_{ij} &= \int_0^4 L_i(\omega)L_j'(\omega)d\omega \\ V_{ij} &= \int_0^4 \omega L_i(\omega)L_j'(\omega)d\omega \end{aligned}$$

the integral becomes

$$A_{12}^J = -\frac{1}{N}\sum_{ij}\eta_{ij}S_{bej}\Big(T_{dai}(n_zU_{cf}+V_{cf})+T_{adi}(n_zU_{fc}+V_{fc})\Big).$$

The next term to consider is very similar:

$$\begin{aligned} A_{12}^K &= -\frac{1}{N}\sum_{ij}\eta_{ij}\int_U (n_z+\gamma)L_i(\alpha)L_j'(\beta)\Big(L_a(\alpha)L_b'(\beta)L_c(\gamma)L_d(\alpha)L_e(\beta)L_f'(\gamma) \\ &\quad + L_a(\alpha)L_b(\beta)L_c'(\gamma)L_d(\alpha)L_e'(\beta)L_f(\gamma)\Big)\,d^3\alpha. \\ &= -\frac{1}{N}\sum_{ij}\eta_{ij}S_{adi}\Big(T_{ebj}(n_zU_{cf}+V_{cf})+T_{bej}(n_zU_{fc}+V_{fc})\Big). \end{aligned}$$

The final term is the most complicated, and cannot rely completely on pre-computed tables due to the factors of $\eta(\alpha,\beta)$ in the denominator. However, it is possible to carry out the integral in the γ direction by making use of tables

$$W_{ij}^k = \int_0^4 \omega^k L_i'(\omega)L_j'(\omega)d\omega$$

for $k = 0,1,2$. This allows us to write

$$\begin{aligned} A_{12}^M &= \frac{1}{N}\int_{\alpha=0}^4\int_{\beta=0}^4 \frac{(n_z^2W_{cf}^0+2n_zW_{cf}^1+W_{cf}^2)\left((\partial_\alpha\eta)^2+(\partial_\beta\eta)^2\right)+N^2h^2W_{cf}^0}{\eta(\alpha,\beta)} \\ &\quad \times L_a(\alpha)L_b(\beta)L_d(\alpha)L_e(\beta)\,d\alpha\,d\beta, \end{aligned}$$

which can be evaluated using two-dimensional Gaussian quadrature with a 10×10 grid of nodes.

References

[1] A. G. Davies, A. D. Heathershaw, Surface-wave propagation over sinusoidally varying topography, Journal of Fluid Mechanics 144 (1984) 419–443. doi:10.1017/S0022112084001671.

[2] M. Belzons, V. Rey, E. Guazzelli, Subharmonic Bragg resonance for surface water waves, Europhysics Letters 16 (2) (1991) 189. doi:10.1209/0295-5075/16/2/012.

[3] M.-R. Alam, Broadband cloaking in stratified seas, Phys. Rev. Lett. 108 (2012) 084502. doi:10.1103/PhysRevLett.108.084502.

[4] M. Tanaka, The stability of steep gravity waves, Journal of the Physical Society of Japan 52 (9) (1983) 3047–3055. doi:10.1143/JPSJ.52.3047.

[5] I. S. Gandzha, V. P. Lukomsky, On water waves with a corner at the crest, Proc. R. Soc. A 463 (2007) 1597–1614. doi:10.1098/rspa.2007.1840.

[6] C. Amick, L. Fraenkel, J. Toland, On the Stokes conjecture for the wave of extreme form, Acta Mathematica 148 (1982) 193–214. doi:10.1007/BF02392728.

[7] M. S. Longuet-Higgins, M. J. H. Fox, Theory of the almost-highest wave: the inner solution, J. Fluid Mech. 80 (4) (1977) 721–741. doi:10.1017/S0022112077002444.

[8] M. S. Longuet-Higgins, M. J. H. Fox, Theory of the almost-highest wave. Part 2. Matching and analytic extension, J. Fluid Mech. 85 (4) (1978) 769–786. doi:10.1017/S0022112078000920.

[9] D. V. Maklakov, Almost-highest gravity waves on water of finite depth, European Journal of Applied Mathematics 13 (01) (2002) 67–93. doi:10.1017/S0956792501004739.

[10] W. G. Penney, A. T. Price, Part II. Finite periodic stationary gravity waves in a perfect liquid, Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences 244 (882) (1952) 254–284. doi:10.1098/rsta.1952.0004.

[11] G. I. Taylor, An experimental study of standing waves, Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences 218 (1132) (1953) 44–59. doi:10.1098/rspa.1953.0086.

[12] M. A. Grant, Standing Stokes waves of maximum height, J. Fluid Mech. 60 (1973) 593–604. doi:10.1017/S0022112073000364.

[13] L. W. Schwartz, A. K. Whitney, A semi-analytic solution for nonlinear standing waves in deep water, J. Fluid Mech. 107 (1981) 147–171. doi:10.1017/S0022112081001717.

[14] G. N. Mercer, A. J. Roberts, Standing waves in deep water: Their stability and extreme form, Physics of Fluids A: Fluid Dynamics 4 (2) (1992) 259–269. doi:10.1063/1.858354.

[15] C.-P. Tsai, D.-S. Jeng, Numerical Fourier solutions of standing waves in finite water depth, Applied Ocean Research 16 (3) (1994) 185–193. doi:10.1016/0141-1187(94)90028-0.

[16] M. Okamura, On the enclosed crest angle of the limiting profile of standing waves, Wave Motion 28 (1998) 79–87. doi:10.1016/S0165-2125(97)00061-9.

[17] W. W. Schultz, J. M. Vanden-Broeck, L. Jiang, M. Perlin, Highly nonlinear standing water waves with small capillary effect, J. Fluid Mech. 369 (1998) 253–272.

[18] M. Okamura, Standing gravity waves of large amplitude on deep water, Wave Motion 37 (2003) 173–182. doi:10.1016/S0165-2125(02)00055-0.

[19] M. Okamura, Almost limiting short-crested gravity waves in deep water, Journal of Fluid Mechanics 646 (2010) 481–503. doi:10.1017/S0022112009992795.

[20] J. Wilkening, Breakdown of self-similarity at the crests of large-amplitude standing water waves, Phys. Rev. Lett. 107 (2011) 184501. doi:10.1103/PhysRevLett.107.184501.

[21] G. R. Verma, J. B. Keller, Three-dimensional standing surface waves of finite amplitude, Physics of Fluids 5 (1) (1962) 52–56. doi:10.1063/1.1706491.

[22] T. J. Bridges, Secondary bifurcation and change of type for three-dimensional standing waves in finite depth, Journal of Fluid Mechanics 179 (1987) 137–153. doi:10.1017/S0022112087001460.

[23] P. J. Bryant, M. Stiassnie, Water waves in a deep square basin, Journal of Fluid Mechanics 302 (1995) 65–90. doi:10.1017/S0022112095004010.

[24] Q. Zhu, Y. Liu, D. K. P. Yue, Three-dimensional instability of standing waves, Journal of Fluid Mechanics 496 (2003) 213–242. doi:10.1017/S0022112003006347.

[25] A. P. Engsig-Karup, M. G. Madsen, S. L. Glimberg, A massively parallel gpu-accelerated model for analysis of fully nonlinear free surface waves, International Journal for Numerical Methods in Fluids 70 (1) (2012) 20–36. doi:10.1002/fld.2675.

[26] L. W. Schwartz, A. K. Whitney, A semi-analytic solution for nonlinear standing waves in deep water, J. Fluid Mech. 107 (1981) 147–171. doi:10.1017/S0022112081001717.

[27] C. J. Amick, J. F. Toland, The semi-analytic theory of standing waves, Proc. Roy. Soc. Lond. A 411 (1987) 123–138. doi:10.1098/rspa.1987.0057.

[28] A. I. Dyachenko, V. E. Zakharov, E. A. Kuznetsov, Nonlinear dynamics of the free surface of an ideal fluid, Plasma Phys. Rep. 22 (1996) 829–840.

[29] P. Plotnikov, J. Toland, Nash–Moser theory for standing water waves, Arch. Rat. Mech. Anal. 159 (2001) 1–83. doi:10.1007/PL00004246.

[30] G. Iooss, P. I. Plotnikov, J. F. Toland, Standing waves on an infinitely deep perfect fluid under gravity, Arch. Rat. Mech. Anal. 177 (2005) 367–478. doi:10.1007/s00205-005-0381-6.

[31] D. M. Ambrose, J. Wilkening, Computation of time-periodic solutions of the Benjamin–Ono equation, J. Nonlinear Sci. 20 (3) (2010) 277–308. doi:10.1007/s00332-009-9058-x.

[32] D. M. Ambrose, J. Wilkening, Global paths of time-periodic solutions of the Benjamin–Ono equation connecting pairs of traveling waves, Comm. App. Math. and Comp. Sci. 4 (1) (2009) 177–215. doi:10.2140/camcos.2009.4.177.

[33] D. M. Ambrose, J. Wilkening, Computation of symmetric, time-periodic solutions of the vortex sheet with surface tension, Proceedings of the National Academy of Sciences 107 (8) (2010) 3361–3366. doi:10.1073/pnas.0910830107.

[34] J. Wilkening, J. Yu, Overdetermined shooting methods for computing standing water waves with spectral accuracy, Comput. Sci. Disc. 5 (1) (2012) 014017. doi:10.1088/1749-4699/5/1/014017.

[35] C. G. Broyden, The convergence of a class of double-rank minimization algorithms, IMA Journal of Applied Mathematics 6 (3) (1970) 222–231. doi:10.1093/imamat/6.3.222.

[36] D. A. Knoll, D. E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2004) 357–397. doi:10.1016/j.jcp.2003.08.010.

[37] D. Viswanath, Recurrent motions within plane Couette turbulence, J. Fluid Mech. 580 (2007) 339–358. doi:10.1017/S0022112007005459.

[38] L. Jiang, C. Ting, M. Perlin, W. W. Schultz, Moderate and steep Faraday waves: instabilities, modulation and temporal asymmetries, J. Fluid Mech. 329 (1996) 275–307. doi:10.1017/S0022112096008920.

[39] N. Périnet, D. Juric, L. S. Tuckerman, Numerical simulation of Faraday waves, J. Fluid Mech. 635 (2009) 1–26. doi:10.1017/S0022112009007551.

[40] G. Haller, Chaos Near Resonance, Applied Mathematical Sciences; 138, Springer-Verlag, NY, 1999.

[41] D. Ambrose, M. Siegel, S. Tlupova, A small-scale decomposition for 3D boundary integral computations with surface tension, J. Comput. Phys. 247 (2013) 168–191. doi:10.1016/j.jcp.2013.03.045.

[42] M. Frigo, S. G. Johnson, The design and implementation of FFTW3, Proceedings of the IEEE 93 (2) (2005) 216–231, special issue on "Program Generation, Optimization, and Platform Adaptation". doi:10.1109/JPROC.2004.840301.

[43] O. C. Zienkiewicz, R. L. Taylor, The finite element method: Its basis and fundamentals, 6th Edition, Elsevier, 2005.

[44] D. Braess, Finite Elements – Theory, fast solvers, and applications in solid mechanics, 3rd Edition, Cambridge University Press, Cambridge, 2007.

[45] W. Craig, C. Sulem, Numerical simulation of gravity waves, J. Comput. Phys. 108 (1993) 73–83. doi:10.1006/jcph.1993.1164.

[46] J. W. Demmel, Applied Numerical Linear Algebra, SIAM, 1997.

[47] E. Hairer, S. P. Norsett, G. Wanner, Solving Ordinary Differential Equations I: Nonstiff Problems, Springer, Berlin, 2000.

[48] J. Nocedal, S. J. Wright, Numerical Optimization, Springer, New York, 1999.

[49] G. N. Mercer, A. J. Roberts, The form of standing waves on finite depth water, Wave Motion 19 (3) (1994) 233–244. doi:10.1016/0165-2125(94)90056-6.

[50] J. R. Wilton, On ripples, Philosophical Magazine Series 6 29 (173) (1915) 688–700. doi:10.1080/14786440508635350.

[51] B. Chen, P. G. Saffman, Steady gravity-capillary waves on deep water–I. weakly nonlinear waves, Studies in Appl. Math. 60 (1979) 183–210.

[52] J.-M. Vanden-Broeck, Nonlinear gravity–capillary standing waves in water of arbitrary uniform depth, J. Fluid Mech. 139 (1984) 97–104. doi:10.1017/S0022112084000276.

[53] J.-M. Vanden-Broeck, Wilton ripples generated by a moving pressure distribution, Journal of Fluid Mechanics 451 (2002) 193–201. doi:10.1017/S0022112001006929.

[54] B. F. Akers, W. Gao, Wilton ripples in weakly nonlinear model equations, Comm. Math. Sci. 10 (3) (2012) 1015–1024.