A Steady-State Solver and Stability Calculator for Nonlinear Internal Wave Flows

Kevin C. Viner\textsuperscript{a}, Craig C. Epifanio\textsuperscript{b,*}, James D. Doyle\textsuperscript{a}

\textsuperscript{a}Naval Research Laboratory, Marine Meteorology Division, Monterey, CA 93943
\textsuperscript{b}Department of Atmospheric Sciences, Texas A&M University, College Station, TX 77843

Abstract

A steady solver and stability calculator is presented for the problem of nonlinear internal gravity waves forced by topography. Steady-state solutions are obtained using Newton’s method, as applied to a finite-difference discretization in terrain-following coordinates. The iteration is initialized with a boundary-inflation scheme, in which the nonlinearity of the flow is gradually increased over the first few Newton steps. The resulting method is shown to be robust over the full range of nonhydrostatic and rotating parameter space. Examples are given for both nonhydrostatic and rotating flows, as well as flows with realistic upstream shear and static stability profiles. With a modest extension, the solver also allows for a linear stability analysis of the steady-state wave fields. Unstable modes are computed using a shifted-inverse method, combined with a parameter-space search over a set of realistic target values. An example is given showing resonant instability in a nonhydrostatic mountain wave.

Keywords: internal waves, gravity waves, Newton’s method, resonant triad, mountain waves, stability analysis

1. Introduction

Internal gravity waves are ubiquitous in geophysical and astrophysical fluid systems, including the atmosphere, the oceans, and the sun [see reviews in 1, 2, 3, 4]. The forcing for these waves stems from a variety of thermal (or more generally, buoyant) and/or mechanical processes, including convective motions, flow over mountains and hills, fronts, shear instabilities, and processes related to geostrophic adjustment.

A key feature of internal gravity waves is that when the amplitude of the wave is sufficiently large, the wave will overturn and break. The dynamics of this process is perhaps best illustrated for the case of flow past an obstacle on the lower boundary, known in the atmospheric context as mountain waves. As first shown by Long [5, 6], as the amplitude of the wave (set here by the obstacle height) is made larger, the pattern of isentropes above the lee slope of the obstacle becomes more steeply inclined. This deepening leads to a decrease in the local static stability of the flow, while at the same time, the vertical wind shear at the base of the deepening region rapidly intensifies. For some critical amplitude, the steepened wave pattern becomes susceptible to three-dimensional Kelvin-Helmholtz (KH) instability, at which point the flow rapidly breaks down into localized turbulence [e.g., 7, 8, 9, 10].

The steepening and breaking process for mountain waves has been extensively studied and is at this point widely recognized in the general fluid dynamics community. On the other hand, it has become apparent only recently that mountain waves are also susceptible to resonant triad instabilities [11], in which a pair of growing disturbance modes mutually reinforce each other by way of interactions with the nonlinear mountain-wave state. The growth rates of this resonant instability are much smaller than those of the KH instability described above, typically by an order of magnitude or more. Nonetheless, as shown by Lee et al., the triad instability occurs for mountain heights much smaller than those needed for KH breakdown. Given sufficient time, the instability causes a global disruption of the entire mountain-wave pattern, with intermittent breaking and turbulence in various parts of the wave field.

To isolate the resonant instability modes, Lee et al. made use of a nonlinear, steady-state mountain wave solution, which was then discretized to produce a large-scale eigenvalue problem for the growing modes. The steady state was obtained using Long’s method [5, 6, 12], which reduces the full, nonlinear mountain-wave system to a single, linear Helmholtz equation. Unfortunately, while Long’s method is both simple and elegant, it is also limited to a relatively narrow range of the mountain-wave parameter space: specifically, nonrotating, Boussinesq flows in two dimensions (2D), with constant upstream wind and static stability. The method has also proved difficult for strongly nonhydrostatic flows (although a new approach based on boundary integrals was presented recently by [12]). Extending the analysis of Lee et al. to more realistic flow conditions thus depends on an alternative, more general method for obtaining steady-state flow fields.

\textsuperscript{*}Corresponding author at: 3150 TAMU, College Station, TX 77843. Tel: +1 9798459224, Fax: +1 9798624466
Email addresses: kevin.viner@nrlmry.navy.mil (Kevin C. Viner), cepi@geos.tamu.edu (Craig C. Epifanio), jones.doyle@nrlmry.navy.mil (James D. Doyle)
The present study describes a method for computing nonlinear, steady-state internal gravity-wave solutions numerically, with a particular emphasis on mountain-wave solutions. The method is based on a Newton iteration, as applied to a finite-difference discretization in terrain-following coordinates. For simplicity, the method is described for 2D flow (although 3D extensions are possible) but is otherwise quite general, allowing rotating and nonhydrostatic effects, as well as arbitrary wind shears and static stability profiles. A key feature of the method is a scheme for gradually increasing the nonlinearity over the first few Newton steps, which leads to a robust methodology over a wide range of parameter space. The method can also be leveraged into an associated eigenvalue problem, allowing a stability analysis for the resulting steady-state flows.

The following section describes the basic solver formulation and gives a general overview of the Newton method. The Newton update equations are presented, as well as a boundary-inflation scheme in which the obstacle height is gradually increased. Section 3 describes the finite-differencing, with an emphasis on boundary conditions. Verifications for both nonhydrostatic and rotating flows are presented in section 4. Section 5 describes applications to realistic upstream flow structures, with a layered stability case and a Gaussian jet presented as examples. The extension to an associated eigenvalue problem and stability analysis is provided in section 6. The final section gives a brief summary and discussion.

2. Basic methods

A Newton solver is developed for the problem of steady, 2D flow past finite-amplitude topography on an $f$-plane. For simplicity the flow is assumed Boussinesq, although an extension to the compressible case is possible.

2.1. Equations of motion

As currently implemented, the main limitation of the solver is the required system memory (i.e., RAM). To help reduce memory demands, the method will be formulated in the vorticity-streamfunction framework, with the disturbance velocity components given by

$$u^* = \frac{\partial \psi^*}{\partial z^*} \quad \text{and} \quad w^* = -\frac{\partial \psi^*}{\partial x^*}$$

where $\psi^*$ is the disturbance streamfunction and asterisks indicate dimensional variables. This will ultimately reduce the system to three (and in some cases two—see sec. 2.6) variables, thus minimizing the memory footprint.

To streamline the numerics, a reference state is defined with constant wind $U$ and uniform static stability $N$. The total streamfunction variable is then given by

$$\Psi^* = Uz^* + \psi^*$$  \hspace{1cm} (1)

while the total Boussinesq potential temperature variable is

$$\Theta^* = N^2z^* + b^*$$  \hspace{1cm} (2)

where $b^*$ is the Boussinesq buoyancy. For the moment, the reference state is assumed to be the background (or upstream) state for the flow as well. The extension to more general background states (including wind shears and static stability variations) is relatively straightforward and is given in section 5.

With the above assumptions and simplifications, the steady nonhydrostatic equations of motion can be written in vorticity-streamfunction form as

$$\left( U + \frac{\partial \psi^*}{\partial z^*} \right) \frac{\partial \eta^*}{\partial x^*} - \frac{\partial \psi^*}{\partial x^*} \frac{\partial \eta^*}{\partial z^*} = \frac{\partial b^*}{\partial x^*} - f \frac{\partial \psi^*}{\partial z^*}$$  \hspace{1cm} (3)

$$\left( U + \frac{\partial \psi^*}{\partial z^*} \right) \frac{\partial b^*}{\partial x^*} - \frac{\partial \psi^*}{\partial x^*} \frac{\partial b^*}{\partial z^*} - N^2 \frac{\partial \psi^*}{\partial x^*} = 0$$  \hspace{1cm} (4)

$$\left( U + \frac{\partial \psi^*}{\partial z^*} \right) \frac{\partial v^*}{\partial x^*} - \frac{\partial \psi^*}{\partial x^*} \frac{\partial v^*}{\partial z^*} = -f \frac{\partial \psi^*}{\partial z^*}$$  \hspace{1cm} (5)

where $\eta^*$ is the $\gamma$-component of the velocity, $f$ is the constant Coriolis parameter, and the vorticity $\eta^*$ is determined by the streamfunction variable as

$$\eta^* = \nabla^2 \psi^* .$$  \hspace{1cm} (6)

Since the flow is both steady and inviscid, the lower boundary is both a streamline for the flow and an isentropic surface. The conditions for $\psi^*$ and $b^*$ along the topography are then

$$\psi^* = -U h^* \quad \text{and} \quad b^* = -N^2 h^*$$  \hspace{1cm} at $z^* = h^*$  \hspace{1cm} (7)

where $h^*(x^*)$ is the terrain profile. The topography is assumed localized so that $b^*$, $v^*$ and $\nabla \psi^*$ all vanish as $|x^*| \to \infty$. The domain is in principle taken to be unbounded aloft.

Note that neither $\eta^*$ nor $v^*$ has an explicit lower boundary condition, implying that the appropriate conditions in this case are simply (3) and (5) applied along the boundary.
2.2. Scale analysis and computational coordinates

For scaling purposes, the topography profile \( h^* (x^*) \) is assumed to have a characteristic length scale \( L \) and a maximum height \( h_0 \). A standard mountain-wave scaling then suggests the following scaling factors:

\[
x^* = L x; \quad z^* = \frac{U}{N} z; \quad h^* = h_0 h; \quad \psi^* = \frac{U^2}{N} \psi; \quad \eta^* = N \eta; \quad b^* = N U \phi; \quad v^* = f L v;
\]

where asterisks again indicate dimensional variables. Substituting these scaling factors into (3)-(7) leads to three nondimensional control parameters: the nondimensional mountain height \( \varepsilon \) where \( \eta^* \) becomes \( \delta \eta \), and assume that the goal of the method is to find a displacement vector \( \delta s = \{ \delta \eta, \delta \phi, \delta v, \delta \psi \} \) so that \( s^n + \delta s \) is an exact solution; that is,

\[
\mathbf{F} (s^n + \delta s) = 0.
\]

The solver is formulated using the terrain-following nondimensional coordinates

\[
X = x \quad \text{and} \quad q = q(x, z)
\]

where

\[
q(x, z) = \frac{z - \varepsilon h}{z_T - \varepsilon h} z_T
\]

is the standard terrain-following coordinate of Gal-Chen and Somerville [13], with \( z_T \) being the nondimensional depth of the solver domain. The resulting equations can be put in terms of \( \psi, \eta, \phi, \) and \( v \) as

\[
\begin{align*}
(1 + \psi q_x) \eta_X + (q_x - \psi X q_z) \eta_q + \phi_X + \phi q_q q_x &- R^2 \nu_q q_z = 0 \quad (9) \\
(1 + \psi q_x) \phi_X + (q_x - \psi X q_z) \phi_q - \psi_X - \psi q_q q_x & = 0 \quad (10) \\
(1 + \psi_q q_z) \nu_X + (q_x - \psi x q_z) \nu_q + \psi_q q_z & = 0 \quad (11)
\end{align*}
\]

where subscripts indicate partial derivatives. The relation between \( \psi \) and \( \eta \) becomes

\[
(13)
\]

while the lower boundary conditions (7) are

\[
\psi = -\varepsilon h \quad \text{and} \quad \phi = -\varepsilon h \quad \text{at} \ q = 0.
\]

As before, the conditions for \( \eta \) and \( v \) at the lower boundary remain (9) and (11).

Conditions at the upper and lateral edges of the solver domain mimic the unbounded conditions for the original physical system. Sponge layers are used to damp disturbances at the lateral boundaries, with periodicity assumed at the domain edges. The upper boundary uses a linear radiation condition at the top of the domain coupled with a sponge layer below. The solver also uses a weak horizontal and vertical fourth-order filter to help control numerical noise.

2.3. Newton linearization

To simplify notation, suppose the disturbance variables are written in function space as

\[
s = \{ \eta, \phi, v, \psi \}
\]

and let the equations of motion be expressed in functional form as

\[
F_1 (s) = 0, \quad F_2 (s) = 0, \quad F_3 (s) = 0, \quad F_4 (s) = 0
\]

where \( F_1 \) through \( F_4 \) are the lefthand sides of (9)–(12), respectively. Letting

\[
\mathbf{F} (s) = \{ F_1 (s), F_2 (s), F_3 (s), F_4 (s) \}
\]

the goal of the steady solver is to find functions \( s \) such that \( \mathbf{F} (s) = 0 \).

The system (14) cannot be solved directly, and solutions are instead sought iteratively using Newton’s method. Let the approximate solution for any given iteration be defined by \( s^n \) (where the superscript is the iteration number), and assume that the goal of the method is to find a displacement vector \( \delta s \) so that \( s^n + \delta s \) is an exact solution; that is,

\[
\mathbf{F} (s^n + \delta s) = 0.
\]

(15)
In Newton’s method, the functional \( F(s^n + \delta s) \) is linearized about the approximation \( s^n \) by assuming small \( \delta s \); that is,

\[
F(s^n + \delta s) \approx F(s^n) + J_F(s^n)[\delta s]
\]

where \( J_F(s) \) is a linear Jacobian-type operator\(^1\), defined so that \( J_F(s)[\delta s] \) is the linearized version of \( F(s + \delta s) - F(s) \) (with product terms in \( \delta s \) neglected). Substituting (16) into (15) then gives the linearized system

\[
J_F(s^n)[\delta s] = -F(s^n)
\]

which is solved for the approximate displacement functions \( \delta s \). The solution is updated as

\[
s^{n+1} = s^n + \gamma \delta s
\]

and the process is repeated until convergence [that is, until \( F(s^{n+1}) \) is below tolerance]. If the starting point is sufficiently close to a root of multiplicity 1, then iterating (17) and (18) approaches the true solution quadratically. Close proximity to a root of multiplicity greater than 1 will give linear convergence.

Note that while \( F_1 \), \( F_2 \) and \( F_3 \) in this case are all nonlinear, the functional \( F_4 \) for (12) is actually linear. The approximation (16) applied to \( F_4 \) is thus exact, and any given Newton step (17) identically brings \( F_4 \) to zero. To keep matters simple, \( F_4 \) will be assumed zero in the initial state for the solver, after which it remains zero at all subsequent steps as well, according to (17).

Letting the approximate solution for the current Newton iteration be given by \( s = \{\eta, \phi, v, \psi\} \) (i.e., \( s = s^n \)), the method (17) as applied to (9)-(12) (with \( F_4 = 0 \)) gives

\[
\begin{align*}
(1 + \psi_q q_v) \delta \eta_X + (q_x - \psi_X q_v) \delta \eta_q + \eta_X q_v \delta q_X - \eta_q q_v \delta \psi_X + \delta \phi_X + q_v \delta \psi_q &= -F_1(s) \\
(1 + \psi_q q_v) \delta \phi_X + (q_x - \psi_X q_v) \delta \phi_q + \phi_X q_v \delta q_X - \phi_q q_v \delta \psi_X - \delta \psi_X + q_v \delta \psi_q &= -F_2(s) \\
(1 + \psi_q q_v) \delta \psi_X + (q_x - \psi_X q_v) \delta \psi_q + \psi_X q_v \delta q_X - \psi_q q_v \delta \psi_X + q_v \delta \psi_q &= -F_3(s) \\
\delta \eta - (q^2 + \delta^2(q^2)) \delta \psi_{q_i} - \delta^2 \left[ \delta \psi_{XX} + 2q_X \delta \psi_{q_X} + \left((q_X)_X + (q_X)_q \right) \right] \delta \psi_q &= 0
\end{align*}
\]

which is a set of linear PDE’s for the displacement variables \( \delta \eta, \delta \phi, \delta v \) and \( \delta \psi \) (given \( s \)). This system is ultimately solved through discretization, as described in section 3.

The lateral and upper boundary conditions for (19)–(22) are the same as those for the full nonlinear system (as described in section 2.2). Conditions at the lower boundary are adapted from (13), as described below.

2.4. The Armijo line search

Ideally, after each linear solve the solution would be updated by adding the full vector \( \delta s \), i.e., \( s^{n+1} = s^n + \delta s \). However, when the current iteration is far from the root of the system, taking the full Newton step may actually overshoot the root and cause the magnitude of the residual [i.e., \( ||F(s^n)|| \)] to increase. To remedy this problem, the Newton linearization is used with a line-search algorithm, which reduces the length of the displacement vector according to

\[
s^{n+1} = s^n + \gamma \delta s \tag{23}
\]

where \( 0 \leq \gamma \leq 1 \) is the line-search coefficient. The coefficient is reduced iteratively from \( \gamma = 1 \) until the Armijo sufficient-decrease condition [14] is met, which states that the decrease in the full nonlinear functionals must be at least a fraction \( \mu \) of the predicted linear decrease; specifically

\[
||F(s^{n+1})|| \leq ||F(s^n) + \mu \gamma J_F(s^n)[\delta s]|| = (1 - \mu \gamma)||F(s^n)||
\]

where \( 0 \leq \mu \leq 1 \) and \( || \) denotes the \( l_2 \) vector norm.

For the present study, the fractional decrease threshold is set to \( \mu = 0.1 \). If a full Newton step (\( \gamma = 1 \)) fails to meet the Armijo condition, then the line search algorithm iterates by bisection according to \( \gamma = 2^{-j} \), where \( j \) is the iteration number for the line search.

---

\(^1\)Technically speaking, the operator \( J_F(s) \) is a functional derivative, defined so that

\[
J_F(s)[\delta s] = \lim_{h \to 0} \frac{F(s + h \delta s) - F(s)}{h} = \frac{d}{dh} F(s + h \delta s) \bigg|_{h=0},
\]

but in practice this amounts to neglecting quadratic terms in \( F(s + \delta s) - F(s) \).
2.5. Initialization, boundary inflation, and convergence

The degree of nonlinearity in the flow is determined mainly by the nondimensional mountain height \( \epsilon \), with the linear case corresponding to \( \epsilon \ll 1 \). A useful starting point when \( \epsilon \) is small is thus the linear solution about the undisturbed background state. However, for larger \( \epsilon \) [that is, for \( \epsilon \sim O(1) \)], the linear solution is a much less adequate guess, and the resulting Newton iterations show slow convergence or failure to converge at all.

To improve robustness, the solver is initialized using a boundary inflation method, in which the effective mountain height is increased gradually over the first few iterations. The method involves replacing the mountain height \( \epsilon \) in (8) and (13) with the scaled mountain height \( \epsilon^s \), where \( n \) is the Newton iteration number and \( \epsilon^s \leq 1 \) is a scaling factor that increases with \( n \). As the iteration proceeds, the functionals in (14) are treated as explicit functions of \( \sigma \) (as a parameter dependence). The Newton linearization (16) is then replaced by

\[
F(s^n + \delta s; \sigma^n + \delta \sigma) \approx F(s^n; \sigma^n) + J_F(s^n; \sigma^n)(\delta s) + \left. \frac{\partial F}{\partial \sigma} \right|_{s^n, \sigma^n} \delta \sigma
\]  

(24)

where \( \delta \sigma = \sigma^{n+1} - \sigma^n \) is given and where \( J_F \) is the Jacobian-type operator described in section 2.3.

In practice, the \( \partial F/\partial \sigma \) term in (24) is algebraically awkward to implement, mostly due to the diagnostic inversion (12). A much simpler algorithm follows from

\[
\left. \frac{\partial F}{\partial \sigma} \right|_{s^n, \sigma^n} \delta \sigma \approx F(s^n; \sigma^n + \delta \sigma) - F(s^n; \sigma^n)
\]  

(25)

which is essentially a Jacobian-free approximation [e.g., 15]. Substituting (25) into (24) and setting the result to zero then gives the modified Newton update system

\[
J_F(s^n; \sigma^n)(\delta s) = -F(s^n; \sigma^n + \delta \sigma)
\]  

(26)

which has the modified lower boundary conditions

\[
\delta \psi(q = 0) = -\delta \sigma \epsilon h \quad \text{and} \quad \delta \phi(q = 0) = -\delta \sigma \epsilon h.
\]  

(27)

The inflation steps conclude once \( \sigma = 1 \), such that \( \delta \sigma = 0 \). The modified problem (26) and (27) then smoothly transitions to (17) and (13), with \( \delta \psi(q = 0) = \delta \phi(q = 0) = 0 \).

The method is initialized by setting \( \eta^0 = \phi^0 = \psi^0 = 0 \) (where the superscript is the iteration number), which corresponds to the flat-boundary case \( \sigma^0 = 0 \). The inflation factor is then increased according to

\[
\sigma^n = \begin{cases} 
0.5 \left( 1 - \cos \left( \pi \frac{n}{n_s} \right) \right), & \text{for } n \leq n_s, \\
1, & \text{for } n > n_s,
\end{cases}
\]

where \( n_s \) is the total number of inflation steps. The value of \( n_s \) must be chosen \textit{a priori}, with larger values of \( \epsilon \), \( R \), or \( \delta \) all requiring larger \( n_s \). Setting \( n_s = 1 \) leads to the linear state described previously, which is adequate for most small or moderate \( \epsilon \) cases.

During the inflation process the Armijo line search is not applied, since setting \( \gamma < 1 \) in (23) violates (27) for \( n < n_s \). Once \( n \) exceeds \( n_s \), the Newton-Armijo steps are repeated until convergence. Convergence for this study requires all nondimensional functionals to be less than \( 10^{-6} \) in magnitude at all points on the grid.

2.6. Reduced variable and hydrostatic formulations

As described below, the variables \( \eta \) and \( \delta q \) are replaced by \( \psi \) and \( \delta \psi \) as part of the discretization step, using (12) and (22). This reduces the problem to three variables, thereby decreasing the required solver memory. This reduction is also needed for the linear stability analysis described in section 6.

Other simplifications apply for special cases. For constant background wind, the total streamfunction \( \Psi = z + \psi \) and total potential temperature variable \( \Theta = z + \phi \) are both constant along streamlines [cf. (1), (2)]. Assuming \( \psi = \phi = 0 \) upstream then implies \( \phi = \psi \) everywhere, allowing the system to be reduced to two variables (\( \psi \) and \( \nu \)). The result is a smaller solver that is both faster and has a smaller memory footprint. For the nonrotating case (\( R = 0 \)), the variable \( \nu \) can be removed from the system as well.

The hydrostatic case follows from setting \( \delta = 0 \) in (12) and (22), which greatly simplifies the discretization.
3. Finite differencing

The system (19)–(22) is discretized on a collocated \( nx \times nq \) grid, with uniform grid spacings \( \Delta x \) and \( \Delta q \). Second-order centered differencing is used for all interior grid points, with second-order one-sided differencing at the lower boundary. Potential odd-even grid decouplings and other numerical noise sources are controlled with a weak fourth-order filter.

As part of the discretization step, the vorticity variables \( \eta \) and \( \delta \eta \) in (19) are eliminated in favor of the discretized stencils for (12) and (22), which reduces the solver to three variables (\( \psi, \phi \) and \( v \)). However, this reduction step leads to problems at both the upper and lower boundaries, where (12) and (22) cannot be discretized at second order. To resolve this problem, the \( q \) and \( \delta q \) variables are retained by the solver at \( q = 0 \) and \( q = z_T \) [where \( z_T \) is the nondimensional domain depth defined in (8)]. The vorticity at \( q = 0 \) is obtained directly from (19), while at \( q = z_T \) the vorticity is obtained through the radiation condition, as described in Appendix B.

Other terms requiring special attention include the \( v_q \) and \( \delta v_q \) terms in (9) and (19) as evaluated at \( q = 0 \). Differentiating these terms with straightforward one-sided differences was found to produce corrupted near-surface wave fields downstream of the mountain. Further details of this problem can be found in Appendix A. As described in the appendix, the solution to the problem requires adding a separate boundary equation for the \( x \)-component of vorticity

\[
\xi = -v_q q_z \quad \text{and} \quad \delta \xi = -\delta v_q q_z
\]  

which then allows \( v_q \) and \( \delta v_q \) to be evaluated in terms of \( \xi \) and \( \delta \xi \) at \( q = 0 \) (see the appendix). The full set of variables involved in the discretization is then \( s = \{\psi, \phi, v, \eta, \xi\} \), where the latter two variables are stored only at the boundary points.

Once discretized, the Newton update system (19)–(22) and (A.3) becomes a large linear algebra problem for \( \delta s = (\delta \psi, \delta \phi, \delta v, \delta \eta, \delta \xi) \) as evaluated on the grid. Symbolically the system is written as

\[
A \delta s = b
\]  

where \( A \) is an \( N \times N \) sparse matrix of size \( N = 3(nx)(nq) + 3nx \), and \( b \) includes both the righthand sides for the Newton update equations as well as the lower boundary conditions described in section 2.5. The system is inverted for \( \delta s \) using a direct sparse inversion method, as implemented in matlab.

Finally, one further complication is the use of a radiation condition at the upper boundary. The radiation condition is applied spectrally, meaning that the relevant forward and inverse Fourier transforms must be embedded directly into the matrix \( A \) in (29). A brief outline of the method is given in Appendix B.

4. Verification examples

Figure 1 shows a series of example calculations for flows past a Gaussian ridge

\[
h(x) = e^{-x^2}
\]

where the height and width of the ridge are set by the governing control parameters (as described in section 2.2). The examples span a wide range of the relevant parameter space, including a hydrostatic, nonrotating case with \( \delta = R = 0 \) (Fig 1a), a nonrotating, nonhydrostatic case with \( \delta = 0.5 \) and \( R = 0 \) (Fig 1c), and a hydrostatic, rotating case with \( \delta = 0 \) and \( R = 1.0 \) (Fig 1e). The calculation domains consist of \( 275 \times 625 \), \( 400 \times 500 \), and \( 375 \times 375 \) grid points, respectively, with \( \Delta x = \Delta q = \pi/25 \) in each case. For the hydrostatic, nonrotating case, the peak (at \( x = 0 \)) is centered in the domain, while for the remaining two cases the upstream boundary is at \( x = -6\pi \). A damping layer is imposed on the outer \( 2\pi \) of the domain in the horizontal (in both directions), with periodicity assumed at the grid edges. The radiation condition is applied at the upper boundary, with a damping layer above \( z = 8\pi \).

Figure 1a shows the isentropes and vertical velocity field for the hydrostatic, nonrotating (\( \delta = R = 0 \)) problem with \( \epsilon = 0.8 \), as computed using the Newton solver. The isentropes in the figure are strongly steepened, suggesting a high degree of nonlinearity. (Critical overturning occurs at roughly \( \epsilon = 0.85 \) in this case.) The corresponding solution from Long’s theory is shown in Fig. 1b, as computed using the non-iterative, Fourier method outlined by [12]. Comparison of the two panels shows that the result from the Newton method is essentially identical to Long’s solution. A similar comparison is shown in Figs. 1c and 1d for the nonhydrostatic, nonrotating (\( \delta = 0.5 \), \( R = 0 \)) problem with \( \epsilon = 0.8 \). Apart from some weak upstream artifacts in the Long’s calculation\(^2\), the Netwon solver again reproduces Long’s theory very closely.

\(^2\) As noted by [12], computing Long’s theory with Fourier methods leads to poor numerical conditioning when both \( \epsilon \) and \( \delta \) are large. The small artifacts in Fig. 1d are thus likely the result of conditioning problems.
Figure 1: Comparison of (a), (c), (e) solutions from the steady-state Newton solver to calculations using (b), (d) Long’s theory and (f) a time-dependent model. All panels show vertical velocity $\psi$ (colors; blue colors negative, white and red colors positive) and potential temperature $z + \phi$ (black lines), with contour intervals chosen for convenience. (a), (b) A hydrostatic, nonrotating case with $\delta = R = 0$ and $\epsilon = 0.8$, as computed using (a) the Newton solver and (b) a Fourier-based Long’s theory calculation. (c), (d) A nonhydrostatic, nonrotating case with $\delta = 0.5$, $R = 0$ and $\epsilon = 0.8$, as computed with (c) the Newton solver and (d) the Fourier-based Long’s theory. (e), (f) A rotating, hydrostatic case with $\delta = 0$, $R = 1.0$ and $\epsilon = 0.8$, computed using (e) the Newton solver and (f) a time dependent code integrated to steady state.
The Newton solution for the hydrostatic, rotating ($\delta = 0$, $\mathcal{R} = 1$) example with $\epsilon = 0.8$ is shown in Fig. 1e. Long’s theory does not apply in this case (since $\mathcal{R} \neq 0$), and the solver is instead compared to a time integration to steady state, as computed using the compressible-Boussinesq code described in [16] and [10] (run in hydrostatic mode). Figure 1f shows the model calculation at nondimensional time $U^* r^*/L = 200$, at which point the solution is effectively steady. Apart from some weak extra damping in the time-dependent code (necessary for stability), the two calculations agree quite closely.

Finally, recall that the steady solution can be reached by time integration only if the solution happens to be stable. For the rotating case shown in Figs. 1e and 1f, the value of $\epsilon$ chosen is close to the largest value allowing a stable solution (for the given $\mathcal{R}$ and $\delta$). For remaining two cases (Figs. 1a-d), the steady solutions are actually unstable. Further discussion of the stability problem can be found in section 6.

5. Vertically-varying background states

5.1. Solver modifications

To simplify the notation, the discussion in sections 2 and 3 assumed constant values for the upstream wind and static stability. Nonetheless, the same methods can be applied to flows with vertically varying upstream states as well.

As in previous sections, the total streamfunction and potential temperature fields are divided into a uniform reference state (with constant values for wind and static stability) plus a disturbance, with the solver scaling factors as well.

To simplify the notation, the discussion in sections 2 and 3 assumed constant values for the upstream wind and static stability. However, for the Newton update system the condition reverts to

$$ s = s_0(z) + \delta s(x,z) $$

where $s_0(z)$ is the vertically varying part of the upstream flow, and $\delta s(x,z)$ is the wave disturbance. The total horizontal wind field for the solver is then (for instance)

$$ u_{tot} = 1 + \psi_z = 1 + (\psi_0)_z + \psi_0 $$

with a similar expression for total static stability (and with $v_0$ assumed zero), and with the boundary condition on the upstream flow being $s \to 0$ as $x \to -\infty$.

Substituting (30) with $\delta s = \delta \delta s$ into (19)–(22) yields a modified set of Newton update equations, with the updates expressed in terms of the wave disturbance field $\delta s$. As before, the boundary conditions upstream and downstream are imposed through damping layers on $s$ at the periodic boundary. A radiation condition is imposed for $\delta s$ at the upper boundary, with a sponge layer below. The appropriate lower boundary condition is expressed in complete form as

$$ \psi = -\sigma \delta s + \psi_0(z = 0) \quad \text{and} \quad \phi = -\sigma \delta s + \phi_0(z = 0) \quad \text{at} \quad q = 0, $$

but for the Newton update system the condition reverts to

$$ \delta \psi = -\sigma \delta s \quad \text{and} \quad \delta \phi = -\sigma \delta s \quad \text{at} \quad q = 0, $$

as in (27). The method is initialized by setting $\delta s = 0$, with $\sigma = 0$. To avoid potential errors caused by differencing $s_0$ along the sloping coordinates, all horizontal derivatives of $s_0$ are explicitly set to zero.3

In the rotating case, a flow with background wind shear also requires changes to the basic equations. A rotating background flow is constrained by thermal wind balance, which in nondimensional form is expressed as

$$ (\phi_0)_y = -\mathcal{R}(\psi_0)_{zz}, $$

implying that the background potential temperature field is $y$-dependent. Adding the associated advection term into (10) then results in

$$ (1 + \psi_q q_x) \phi_x + (q_x - \psi_x q_z) \phi_q - \psi_x q_x - \mathcal{R}^2(\psi_0)_{zz} v = 0, $$

with a corresponding term added to (20). Similarly, the additional background wind field implies an additional geostrophic pressure gradient, which is expressed in (11) as

$$ (1 + \psi_q q_x) v_x + (q_x - \psi_x q_z) v_q + (\psi_q - (\psi_0)_q) q_z = 0, $$

with a corresponding term for $\xi$ added to (A.2). Note that despite the $y$-dependence of $\phi_0$, the disturbance flow remains 2D, since the additional terms in (31) and (32) are $y$-independent.

From a conceptual standpoint, the changes described above are relatively straightforward, and require no new methods beyond those described in sections 2 and 3.

---

3Testing suggests that computing horizontal derivatives of $s_0$ in the terrain-following coordinates can lead to significant and systematic truncation errors over the obstacle slopes, which in many cases corrupts the solution. Elimination of these terms is the main motivation for separating $s$ into background and wave disturbance parts.
5.2. Examples

Figure 2 shows a pair of example calculations for flows with nonconstant upstream states. As before, the calculations assume periodicity in the horizontal, with sponge layers on the outer 2σ of the domain in both directions. The upstream boundary is located at \( x = -6\sigma \), with the peak at \( x = 0 \). The radiation condition is imposed at the upper boundary, with a sponge layer applied above \( z = 8\sigma \). The computation domain consists of \( 400 \times 500 \) grid points for the case shown in Figs. 2a and b, and \( 492 \times 408 \) points for the case in Figs. 2c and d, with \( \Delta x = \Delta q = \pi /25 \) in both cases.

Figure 2a shows an example featuring a two-layer static stability profile, in which the reference stability (which defines the variable scales) applies below \( z = \pi \), while above \( z = \pi \) the stability is reduced by half. The associated control parameters for the flow are \( \epsilon = 0.5, \delta = 0.5 \) and \( \mathcal{R} = 0 \). As is apparent in the figure, the rapid stability difference at \( z = \pi \) leads to wave reflections and ducting, and the end result is a classical trapped lee wave flow. Figure 2b shows an analogous calculation using the time-dependent code of [16] and [10], as integrated to steady state. Clearly the model steady-state solution is closely reproduced by the Newton solver.

An example featuring a variable upstream wind profile is illustrated in Fig. 2c. The total upstream wind speed (reference plus background) follows a Gaussian jet, as defined by

\[
\hat{u}_\text{tot} = 1 + \psi_z = 1 + (u_{\text{max}} - 1) \exp \left[ -\left( \frac{z - z_{\text{max}}}{W} \right)^2 \right]
\]

where \( u_{\text{max}} \) is the nondimensional wind speed maximum, \( z_{\text{max}} \) is the height of the jet core, and \( W \) is the jet half-width. The example shown has \( u_{\text{max}} = 3, z_{\text{max}} = 10 \) and \( W = 5 \). Given typical dimensional parameters of \( U = 10 \text{ m/s} \) and \( N = 0.01 \text{ s}^{-1} \), this translates to a 30 m/s wind speed maximum at a height of 10 km, with a jet half-width of 5 km, which are reasonable values for the atmosphere. The terrain parameters are given by \( \epsilon = 0.5, \delta = 0.5 \) and \( \mathcal{R} = 0 \).

As in the previous example, the rapid increase in wind speed with height leads to strong wave reflections and ducting. An analogous solution from a time integration to steady state is shown in Fig. 2d. Again, the Newton solver closely reproduces the model fields.

6. Linear stability analysis

Leveraging the Jacobian matrix in (29), an extension of the Newton solver leads to a stability analysis of the steady-state flow fields. In the following, it is assumed that \( \eta \) has been eliminated in favor of \( \psi \), as described in section 3. The inversion relation (12) is then no longer necessary, and the equivalent time-dependent system is expressed entirely in terms of prognostic equations.

Suppose the steady solution produced by the Newton solver is \( s \), and let \( s' \) be a time-dependent disturbance to this solution. For clarity, let the solver matrix \( A \), defined in section 3, be written symbolically in block form as

\[
A = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}
\]

where each row of (33) corresponds to one of (19)–(21), and let the operator converting \( \psi \) to \( \eta \) be denoted by \( \mathcal{H} \).

The linear dynamics of \( s' \) is then given in discretized form by

\[
\frac{\partial s'}{\partial t} + A s' = 0
\]

where

\[
B = \begin{pmatrix} \mathcal{H} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\]

with \( I \) being the identity operator.

To compute stability, first define the modified operators

\[
\tilde{B} = \begin{pmatrix} -\mathcal{H} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \tilde{A} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ -A_{21} & -A_{22} & -A_{23} \\ -A_{31} & -A_{32} & -A_{33} \end{pmatrix}
\]

where the signs are chosen so that \( \tilde{B} \) is positive definite. Assuming a normal mode solution \( s' = \hat{s}' \exp(\lambda t) \) then allows (34) to be written as

\[
(\tilde{A} - \lambda \tilde{B}) \hat{s}' = 0
\]
Figure 2: Example calculations for flows with variable upstream wind and static stability profiles. All panels show vertical velocity $-\psi_x$ (colors; blue colors negative, white and red colors positive) and potential temperature $\zeta + \phi$ (black lines). (a) A two-layer stability case with $\delta = 0.5$, $R = 0$ and $\epsilon = 0.5$, as computed using the Newton solver. (See text for details.) (b) As in (a), but computed from a time integration to steady state. (c) A Gaussian jet flow with $\delta = 0.5$, $R = 0$ and $\epsilon = 0.5$, as computed using the Newton solver. (d) As in (c), but computed from a time integration to steady state.

which has the form of a large-scale, generalized eigenvalue problem for $\lambda$ and $s'$.

The most unstable mode is located using a shifted-inverse method, combined with a search over a set of realistic target eigenvalues. For a given target eigenvalue $\sigma$, (35) is reformulated as the shifted-inverse problem

$$\left((\hat{A} - \sigma \hat{B})^{-1} \hat{B} - \nu \hat{I}\right) s' = 0 \quad \text{(36)}$$

whose eigenvectors match those of (35), and whose eigenvalues are related to those of (35) by $\nu = 1/(\lambda - \sigma)$. The largest eigenvalue of the shifted problem is found using an Arnoldi method. Given the largest $\nu$, the eigenvalue $\lambda$ closest to $\sigma$ is then determined by

$$\lambda = \frac{1}{\nu} + \sigma.$$

This process is repeated for all target eigenvalues $\sigma$ in the search domain, with the most unstable mode being the mode with largest $\text{Re}(\lambda)$.

6.1. An example calculation

An illustration of the combined Newton solver and eigenvalue analysis is shown in Fig. 3, as applied to a strongly nonhydrostatic flow with $\delta = 1$, $R = 0$ and $\epsilon = 1.2$. The solution is computed on a $337 \times 241$ grid, with $\Delta x = \Delta q = \pi/16$. Damping layers are applied on the outer $4\pi$ of the grid in the horizontal, with the upstream boundary at $x = -8\pi$. A radiation condition applies at the upper boundary, with a damping layer above $z = 8\pi$.

Figure 3a shows the steady-state solution produced by the Newton solver. The isentropes are strongly steepened above the lee slope, suggesting a high degree of nonlinearity. (Critical overturning for this case occurs at roughly $\epsilon \approx 1.4$; see [12].) Using this steady-state solution in the eigenvalue analysis described above reveals a number of unstable modes, with the fastest growing mode illustrated in Fig. 3b (superimposed on the steady-state
Figure 3: Example calculation for the combined Newton solver and eigenvalue analysis, as applied to a flow with $\delta = 1.0$, $R = 0$ and $\epsilon = 1.2$. Color contours show either (a) steady-state vertical velocity or (b), (d) eigenmode vertical velocity (blue colors negative, white and red colors positive). Black contours in each case show isentropes from the steady-state solution. (a) Steady solution as computed by the Newton solver. (b) The most unstable linear mode as determined by the eigenvalue analysis, shown at a fixed point in the oscillation. (c) Eigenmode streamfunction at a fixed point on the grid, as computed by a time integration linearized about the steady-state solution. Overlayed (dashed line) is a curve fit using an exponential sinusoid. (See text for details.) (d) The most unstable mode from the time integration, for the same point in the oscillation as in (b).

The cellular appearance of the mode in Figs. 3b and d suggests that the instability is likely the result of wave-wave interactions, much as in the hydrostatic case of [12]. Assuming characteristic dimensional values of $U = 10$ m/s and $L = 1$ km gives a dimensional oscillation period of roughly 1.7 hours, and a dimensional e-folding time of roughly 1.0 hours, both of which are much faster than the time scales on which the background state varies.

6.2. Limitations

It should be kept in mind that since the present solver is based on a 2D Jacobian, the resulting instability modes will be strictly 2D as well. Of course, even for a 2D basic state, there is no guarantee that these 2D modes
will be the fastest growing modes in the problem. Previous work with monochromatic gravity waves suggests that for resonant-type wave-wave instabilities, the fastest growing modes are in fact typically 2D modes [18]. Furthermore, these resonant modes tend to dominate at small to moderate wave amplitudes. Nonetheless, at larger amplitudes, faster growing 3D modes appear, particularly the 3D modes associated with wave overturning and breaking [e.g., 7, 9]. A more complete study of mountain-wave instability will thus require an extension of the solver to 3D.

Further results, including a parameter mapping for the full $R$, $\delta$ and $\epsilon$ parameter space and consideration of 3D effects, will be presented in future studies.

7. Summary

A method was developed for computing nonlinear internal gravity wave solutions at steady state, as well as for computing the stability of the steady-state solutions. The method was formulated for topographic waves but applies equally well to waves forced by other forcing mechanisms.

Steady-state solutions are obtained using Newton’s method, as applied to a finite-difference discretization in terrain-following coordinates. The method is limited to 2D (although 3D extensions are possible) but is otherwise quite general, allowing both nonhydrostatic and rotating effects, as well as varying background wind and static stability profiles. The method is initialized using a boundary-inflation scheme, in which the height of the terrain is increased gradually over the first few Newton steps. The resulting solver is shown to be robust for both rotating and nonhydrostatic flows, and allows solutions for obstacle heights up to (and even slightly exceeding) the height at which the steady-state fields overturn.

By leveraging the Jacobian matrix, an extension of the Newton solver leads to a stability analysis of the steady-state flow fields. Unstable eigenmodes are computed using a shifted-inverse method for assumed target eigenvalues, with the fastest growing mode identified through a parameter-space search. An example calculation was presented for the case of strongly nonhydrostatic flow past an isolated ridge, which was shown to be vigorously unstable. A more complete stability analysis, exploring modes for both rotating and nonhydrostatic flows, will be presented in upcoming work.

Finally, it should be noted that the present solver was implemented using direct matrix inversions. The resulting calculations are robust and fast (as compared to time integrations) but are at the same time somewhat memory-intensive. (The example cases require at least 8 GB of RAM, which is far more than an equivalent time integration.) Extending the solver to three dimensions (3D) will thus likely depend on iterative matrix inversions, which in turn requires an appropriate preconditioner. The authors are currently exploring preconditioning options in the pressure-velocity framework, with the intention of eventually extending the solver to 3D.

Acknowledgements The authors thank Dave Muraki for graciously providing the Fourier-based Long’s theory code used in section 4. Part of this work was completed while the first author was affiliated first with Texas A&M University and later with the National Research Council. Early work on this project was supported by NSF grant ATM-0242228. The first and third authors acknowledge support through ONR P0 0601153N, while the first author also acknowledges support through the NRL Jerome and Isabella Karle Fellowship.

Appendix A. The $v_q$ terms at the boundary

The $v_q$ and $\delta v_q$ terms in (9) and (19) require special treatment at the lower boundary. A straightforward one-sided difference of these terms was found to corrupt the wave response downstream of the mountain. The source of this corruption can be inferred from the linear (or small $\epsilon$) form of (11), specifically

$$v_X = -\psi_q q_z$$

(A.1)

where it has been recognized that $q_z \sim O(\epsilon)$ and thus vanishes in the linear limit. According to (A.1), a $v_q$ difference is the same as an $X$ integral of $\psi_{qq} q_z$. Since $\psi_{qq}$ is itself not computable at the boundary (at least not directly), the $v_q$ term at the boundary is also problematic.

To avoid the boundary differencing for $v_q$, a separate equation is added for the $x$-component of vorticity (28) at $q = 0$. After some cancellation, the appropriate condition is

$$\left(1 + \psi_q q_z\right) q_X + (q_x - \psi_X q_z) \xi_q - (1 + v_X) \psi_{qq}(q_z)^2 - \left(q_{qq} q_z + q_q q_z \right) \xi_X = 0$$

(A.2)

or in Newton update form

$$\left(1 + \psi_q q_z\right) \xi_{qX} + (q_x - \psi_X q_z) \xi_q + \xi_X q_z \delta \psi_q - \xi_q q_z \delta \psi_X - (1 + v_X) \psi_{qq}(q_z)^2 - \psi_{qq}(q_z)^2 \delta v_X$$

$$-\left(q_{qq} q_z + q_q q_z \right) \delta \xi - \xi_q q_z \delta \psi_X - \xi q_z \delta \psi_q = -F_5(s)$$

(A.3)
where \( F_z(s) \) refers to (A.2) applied at the current iteration. As with \( \eta \) and \( \delta n \), the variables \( \xi \) and \( \delta \xi \) are stored only at \( q = 0 \). If \( \delta \varphi_{qq} \) and \( \delta \psi_{qq} \) are eliminated from (A.2) and (A.3) using (12) and (22), then all resulting terms can be computed with one-sided differencing.

With (A.2) and (A.3) added, the problematic terms in (9) and (19) can be evaluated in terms of \( \xi \) and \( \delta \xi \) at \( q = 0 \). The same applies for the \( v_q \) and \( \delta v_q \) terms in (11) and (21), although in this case the terms are explicitly zero once the inflation steps have completed.

Appendix B. The radiation condition

To implement the radiation condition, a series of horizontal Fourier transforms and inverses must be embedded into the operator \( A \) in (29). Consider the gridded variable \( \delta \psi_{jk} \) (where subscripts denote grid indices, with \( j \) the horizontal index and \( k \) the vertical index) and let the upper boundary of the grid be \( k = p \). For any given vertical grid level, let the horizontal array of points at fixed \( k \) be denoted by \( \delta \psi_{k} \), and let the associated discrete Fourier transform of \( \delta \psi_{k} \) be \( \hat{\delta \psi}_{k} \). Suppose the number of points to be transformed is \( M \) and let the \( n \)-th horizontal wavenumber on the grid be denoted by \( \kappa_n \), with \( 1 \leq n \leq M \). The discrete transform of \( \delta \psi_{k} \) can then be written in matrix form as

\[
\delta \hat{\psi}_{k} = T \delta \psi_{k}
\]

where \( T \) is the \( M \times M \) transform operator with elements

\[
T_{n,j} = \exp(-i\kappa_n(j-1)\Delta \chi).
\]

In continuous form, the radiation condition is expressed as

\[
\frac{\partial}{\partial q} \delta \hat{\psi} \left. \right|_{q=0} = im \delta \hat{\psi} - i \frac{\kappa}{\Delta \chi} \delta \psi_{n},
\]

where \( m \) is the vertical wavenumber. The wavenumber \( m \) depends on \( \kappa \) and is chosen to give either upward energy propagation (for \( R < |\kappa| < 1/\delta \)) or else vertical decay (for \( |\kappa| \leq R \) or \( |\kappa| \geq 1/\delta \)) in the linear limit (see below). Differencing (B.3) trapezoidally gives

\[
\delta \hat{\psi}_{n,k} = \frac{\partial z}{\partial q} \left( \frac{m_n \Delta q / 2}{\delta \hat{\psi}_{n,k-1} - \lambda_n \delta \hat{\psi}_{n,k-1}} \right)
\]

where \( m_n \) is the value of \( m \) corresponding to \( \kappa_n \), and where (B.4) is implicitly applied at the upper boundary \( k = p \). Defining the \( M \times M \) diagonal matrix

\[
\Lambda_{n,j} = \delta_{n,j} \kappa_n,
\]

where \( \delta_{n,j} \) is the Kronecker delta, then allows the full forward transform (B.1), radiation condition (B.4), and inverse transform steps to be written as a single matrix operation as

\[
\delta \psi_{k} = \frac{1}{M} T^{\ast} A T \delta \psi_{k-1} \quad \text{at } k = p,
\]

where the discrete inverse transform operator \( T^{\ast} \) is the Hermitian transpose of (B.2).

According to (B.5), the value \( \delta \psi_{j,k} \) at any given point \( j \) on the upper boundary is simply a linear combination of the values one level below (and similarly for \( \delta \phi \) and \( \delta \psi \)). This condition is then imposed in (29) by embedding the \( j \)-th row of the operator \( (1/M) T^{\ast} A T \) in (B.5) into the appropriate row of \( A \) in (29).

For the present study, the wavenumber \( m \) in (B.3) is modified to include the effects of the upper sponge layer and horizontal filter. The relevant dispersion relation is then

\[
m^2 = \frac{ik(\kappa + \alpha')}{{\left(\frac{\kappa + \alpha'}{2} + R^2\right)}}
\]

where \( \alpha' = \alpha + \gamma k^4 \), with \( \alpha \) being the sponge coefficient and \( \gamma \) the fourth-order filter coefficient. The branch of the square root is chosen so that \( \text{Im}(m) > 0 \), thus guaranteeing vertical decay. The limit of small \( \alpha' \) shows that this branch choice matches the standard linear theory choice when \( m^2 \) is real. As in (19), somewhat better results are achieved at short wavelengths by replacing \( \kappa \) in (B.6) by its discretized counterpart for centered differencing.

Finally, the differencing of (9) and (19) at grid level \( k = p - 1 \) relies on the availability of \( \eta \) at the upper boundary, in addition to \( \psi, \phi \) and \( \psi \). Under the radiation condition, \( \eta \) at the upper boundary is obtained spectrally from \( \psi \) using [cf. (6)]

\[
\delta \eta_{n,k} = \left( -\delta \kappa_n^2 - m_n^2 \right) \delta \hat{\psi}_{n,k} \quad \text{at } k = p
\]

where \( m_n^2 \) is again given by (B.6). As before, somewhat better results are obtained at short wavelengths by replacing \( \kappa_n^2 \) and \( m_n^2 \) by their counterparts for centered differencing. Combining with (B.4) then allows (B.7) to be implemented much as described for (B.5).
References