Open-Channel Computation with R

by Michael C. Koohafkan and Bassam A. Younis

Abstract The **rivr** package provides a computational toolset for simulating steady and unsteady one-dimensional flows in open channels. It is designed primarily for use by instructors of undergraduate and graduate-level open-channel hydrodynamics courses in such diverse fields as river engineering, physical geography and geophysics. The governing equations used to describe open-channel flows are briefly presented, followed by example applications. These include the computation of gradually-varied flows and two examples of unsteady flows in channels—namely, the tracking of the evolution of a flood wave in a channel and the prediction of extreme variation in the water-surface profile that results when a sluice gate is abruptly closed. Model results for the unsteady flow examples are validated against standard benchmarks. The article concludes with a discussion of potential modifications and extensions to the package.

Introduction

Hydraulic engineers routinely rely on computational models to simulate the flow and conveyance of water for planning and designing channels and in-stream structures, managing reservoirs, and conducting flood damage and risk analyses. The simulation of fluid flow is a difficult and computationally-intensive endeavor, and simplifications are often made in order to reduce the task to one that is feasible with regard to time management and data availability. One of the most frequently-made assumptions is that of one-dimensional flow. This greatly simplifies the computation of a wide range of practically-relevant flows such as e.g. the prediction of the movement of a flood wave in a waterway, or the consequences of a sudden opening or closure of a control gate on the water level in a navigation channel. In making this assumption, the flow velocity—which in reality can vary in all three coordinate directions—is both depth- and laterally-averaged to produce a bulk or mean channel velocity that is a function of only time and distance along the channel:

$$\overline{\overline{u}} = \frac{\int_{y=0}^{y=H}\int_{z=0}^{z=S} u(x,y,z,t)\,dy\,dz}{\iint dy\,dz} = f(x,t) \tag{1}$$

where u is velocity, y is the vertical dimension, z is the lateral (cross-channel) dimension, t is time and x is the along-channel direction. The integral bounds H and S are the flow depth and span, respectively. Treatment of bed friction is also frequently simplified in practical applications by relating the bed shear stresses to the "friction slope", which is derived from the Energy Grade Line (EGL) equation (Leopold, 1953) and expressed using the semi-empirical Manning's equation

$$S_f = \left(\frac{n\,u}{C_m R^{2/3}}\right)^2 \tag{2}$$

where S_f is the friction slope, n is the empirical Manning's coefficient and R is the hydraulic radius. Manning's equation and the EGL equation are widely used in the discipline to predict steady and gradually-varied flow conditions. The prediction of unsteady flows is somewhat more complex; in addition to the fact that the flow area is different from the area of the conveying channel, the governing equations are both non-linear and coupled. This complicates the task of obtaining solutions that are both numerically stable and accurate without the need for excessive computational resources.

A wide variety of software suites are available for computing steady and unsteady flows; some of the most well-known packages are listed in Table 1. These packages are typically accessed via a graphical user interface and often have specialized input and output data formatting requirements. These software suites are capable of modeling complex networks of irregular channels and are therefore of great utility to engineers and practitioners working on large-scale projects, but they also pose high barriers to entry in terms of cost, usability and computer hardware requirements as well as in terms of ancillary data needs. These software suites are so complex that many developers provide training courses for utilizing the software, with the assumption that users already possess an education or background in river mechanics and hydraulics.

The **rivr** package was designed to provide rapid solutions to steady and unsteady one-dimensional flows in open channels. The primary purpose of the package is to provide educators with an accessible open-source toolset that can be used to supplement lectures with interactive examples, and to facilitate the design of assignments that encourage students to explore open-channel flow concepts rather than focusing on the actual implementation of solutions. The R Project for Statistical Computing provides

Table 1: Selection of popular software packages for modeling steady and unsteady flow.

Name	Developer	Free for use	Open Source	Cross-platform
HEC-RAS	USACE	✓	✗	✗
SRH 1D	USBR	✓	✗	✗
ESTRY	BMT	✗	✗	✗
Mike 11	DHI	✗	✗	✗
SOBEK D-Flow 1D	Deltares	✗	✗	✗

an ideal platform for providing **rivr** (and educational tools in general) to students and instructors. R is both cross-platform and free and open-source software, meaning students can run the software on personal computers and educational institutions do not need enter expensive licensing agreements to provide it on institution-owned computers. Moreover, R provides a large and comprehensive library of functions for data visualization, analysis and export both through its core functionality and through a diversity of packages developed and supported by an active global community of over 2 million users, developers and scientists. R is often referred to as a "scripting language", meaning that code blocks or snippets can be run either in an automated fashion or piecewise by users; scripting languages are ideal for teaching because they support interactive programming and exploration of data. Furthermore, support for dynamic document generation with R via packages such as **knitr** (Xie, 2014) provides an excellent mechanism for instructors to develop teaching materials that place concept and implementation side by side, as well as a promising format for students to submit assignments and reports. Finally, R provides the potential for instructors with virtually no web development proficiency to create interactive web-based tutorials and demonstrations using the **shiny** web application framework (Chang et al., 2015).

The flow algorithms provided by **rivr** are implemented in C++ and integrated into the package using **Rcpp** (Eddelbuettel and Francois, 2011) to leverage the speed advantage of compiled code. All algorithms were developed from the ground up and are based on well-established, peer-reviewed contributions to the field. Functions are provided for computing normal and critical depth, channel geometry and conveyance, and gradually-varied flow profiles. The package also provides an advanced function which implements two canonical formulations of unsteady flow—the Kinematic Wave Model and the Dynamic Wave Model—which are routinely taught in graduate-level courses in hydrology and open-channel hydraulics. The numerical schemes implemented in **rivr** are similarly well-established and reflect varying degrees of sophistication that are common topics in numerical methods courses, including implementations of iterative solutions to implicit equations; a solution scheme for nonlinear ordinary differential equations; and numerical methods for solving sets of partial differential equations ranging from one of the simplest formulations of an explicit finite-difference scheme to an advanced two-step predictor-corrector scheme that illustrates the trade-offs between performance and complexity of implementation. Detailed explanation of the theoretical foundations and numerical schemes is provided in the form of a technical vignette, and the source code is carefully organized and commented to facilitate modifications and extensions to the package such as new numerical schemes and alternative formulations for steady and unsteady flow modeling.

This article provides use examples for computing both gradually-varied and unsteady flows with **rivr**. In the first section, solutions to gradually-varied flow or "backwater curve" problems using the standard-step method are discussed. In the second section, solutions to unsteady flow problems using different theoretical formulations are described, followed by a comparison of two numerical solutions to an unsteady flow problem with discontinuous boundary conditions. The article concludes with a discussion of potential opportunities for extensions and contributions to the package.

Calculation of back-water curves

The water-surface profile resulting from non-uniform steady flow conditions is generally referred to as a gradually-varied flow (GVF) profile or "backwater curve". Backwater curves are commonly used for floodplain delineation and assessing upstream effects of bridges, weirs and other channel structures. The GVF concept is based on a simplified form of the St. Venant equations (discussed in the following section) and is expressed as

$$\frac{dy}{dx} = \frac{S_0 - S_f}{1 - Fr^2} \tag{3}$$

where dy/dx is the water-surface slope, S_0 is the channel slope and Fr is the Froude Number. Eq. (3) describes steady-state conditions, i.e. the channel flow rate $Q = uA$ is constant. When $dy/dx = 0$, the flow depth is uniform and $S_0 = S_f$; the flow depth under these conditions is referred to as the

"normal depth" and can be calculated using Manning's equation. The relative positioning of the normal depth and the "critical depth"—the depth at which the flow energy is minimized—determine the flow behavior. GVF profiles are classified based on the flow regime, the channel slope and the nature of the divergence from the steady-flow water-surface profile at a specific location or "control section" (Doubt, 1971). An M1 profile refers to sub-critical flows where the water depth at the control section is greater than the normal depth. An M2 profile refers to a water-surface profile where the water depth at the control section is below the normal depth, but above the critical depth. Other classes include "S" (steep channels where the normal depth is shallower than the critical depth), "C" (flow regimes where the normal depth and critical depth coincide), "H" (flat channels) and "A" (channels of "adverse" slope, i.e. sloped against the direction of flow).

The **rivr** package computes the GVF profile using the standard-step method (Chaudhry, 2007). A Newton-Raphson scheme is used where iterative methods are required to find the roots of an equation. The following example illustrates how to use **rivr** to calculate the variation of the water-surface profile with distance, and its return to the normal-depth level far upstream of a perturbation in flow depth. Both an M1 and an M2 profile are considered for a rectangular channel with a width of 100 feet, a slope of 0.001, a flow of 250 cubic feet per second, and a Manning's roughness coefficient of 0.045. The water depth y at the control section is specified as 1 foot above the normal depth y_n for the M1 profile (2.71 feet) and 1.1 times the critical depth y_c for the M2 profile (0.64 feet). The elevation and x-coordinate of the control section are both specified as zero. Example code for computing the water-surface profile using the function `compute_profile` is shown below.

```
g <- 32.2          # gravitational acceleration (ft/s^2)
Cm <- 1.486        # conversion factor for Manning's equation in US customary units
slope <- 0.001     # channel slope (vertical ft / horizontal ft)
mannings <- 0.045  # Manning's roughness
flow <- 250        # channel flow rate (ft^3/s)
width <- 100       # channel bottom width (ft)
sideslope <- 0     # channel sideslope (horizontal ft / vertical ft)

# calculate normal depth for channel, initial guess of 2 feet
yn <- normal_depth(slope, mannings, flow, 2, Cm, width, sideslope)

# calculate critical depth, initial guess of 1 ft
yc <- critical_depth(flow, 1, g, width, sideslope)

# compute the M1 profile 3000 ft upstream with a step size of 50 ft
m1 <- compute_profile(slope, mannings, flow, yn + 1, Cm, g,
  width, sideslope, z0 = 0, x0 = 0, stepdist = 50, totaldist = 3000)

# compute the M2 profile under the same assumptions
m2 <- compute_profile(slope, mannings, flow, 1.1*yc, Cm, g,
  width, sideslope, z0 = 0, x0 = 0, stepdist = 50, totaldist = 3000)
```

The effect of model resolution on results is shown in Figure 1. For the M1 profile, the solution is quite stable even for very coarse ($>$ 100 feet) resolutions. In contrast, solutions to the M2 profile are quite sensitive to model resolution owing to the steep gradient in dy/dx where $y \to y_c$. However, the solution to the M2 profile is not sensitive to resolutions finer than 10 feet. A step size of 10 feet is used in subsequent analyses to ensure smooth calculations of the water-surface slope for both profiles.

Manning's coefficient is an empirical quantity that cannot be directly measured, and engineers often rely on tabulated values to inform their selection of a roughness value based on channel bank material and conditions such as vegetation and compaction. It is generally necessary to investigate the sensitivity of the solution to the choice of Manning's roughness coefficient, or to calibrate a GVF model based on channel depth observations. The next example considers the GVF profile of the control section specified above for a range of Manning's roughness values spanning 0.0225 to 0.0675. Figure 2 shows that the normal depth (apparent as the section where the water-surface slope is linear) changes depending on the value of the roughness coefficient; this comes as no surprise, as the concept of normal depth is itself derived from Manning's equation. The specification of a fixed control section depth therefore confounds analysis of the relationship between Manning's coefficient and the nature of the GVF profile.

In order to control for the dependence of the normal depth on Manning's coefficient, the analysis is repeated using a control section depth of $1.25y_n$ for the M1 profile and $0.75y_n$ for the M2 profile, where y_n is the normal depth associated with each roughness value. Note that the critical depth is by definition independent of Manning's coefficient. Figure 3 shows the water-surface elevation profile as a percent difference from the normal depth for each roughness value; it is clear from the figure that bed

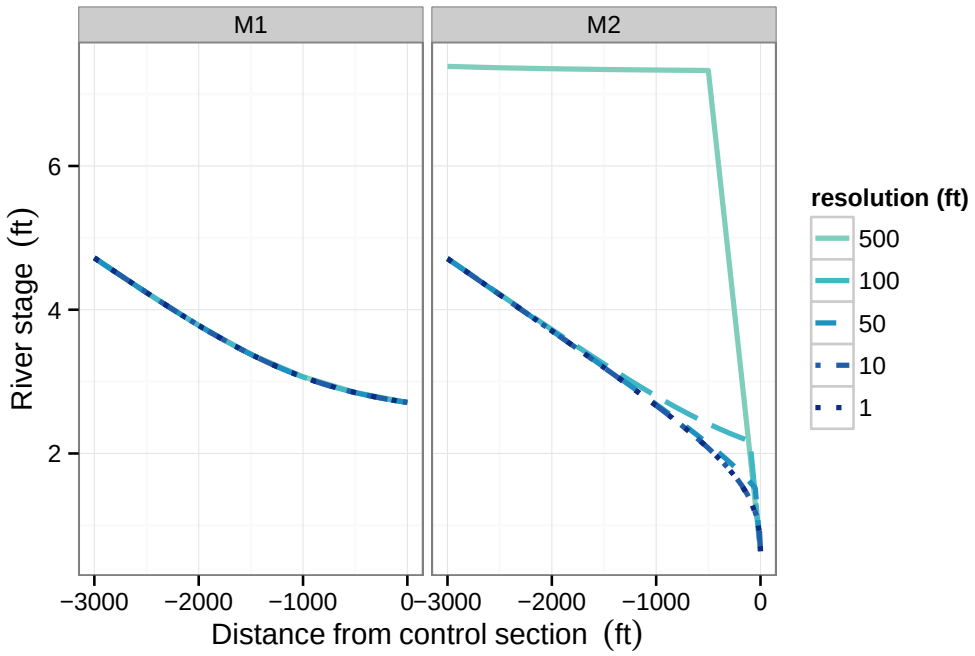

Figure 1: M1 and M2 water-surface elevation profiles computed under different resolutions. When steep gradients in the water-surface profile are present, finer spatial resolutions are needed to provide accurate solutions.

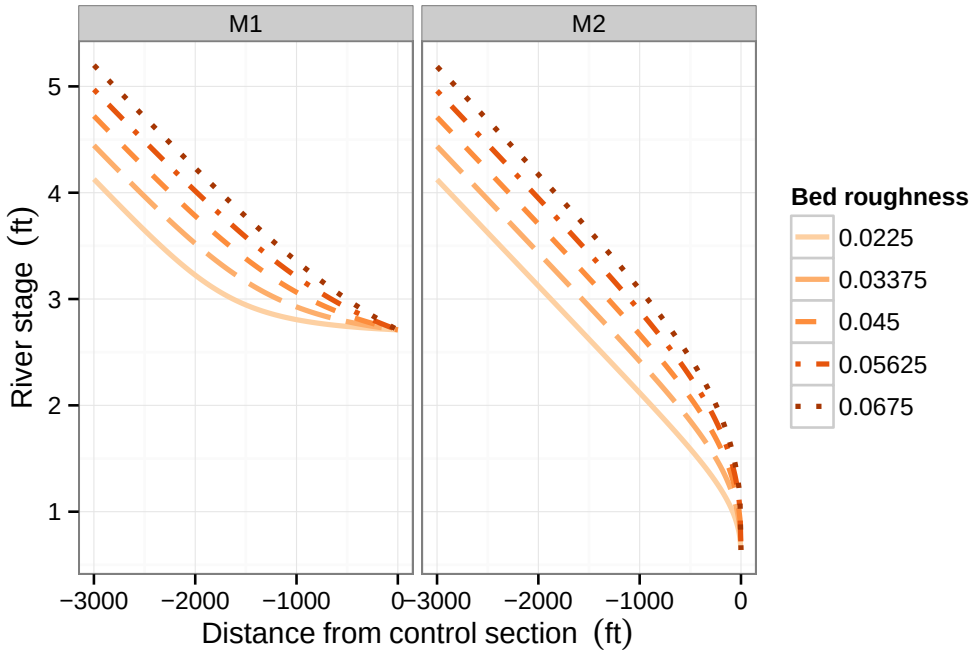

Figure 2: M1 and M2 water-surface elevation profiles for a range in values for Manning's roughness coefficient.

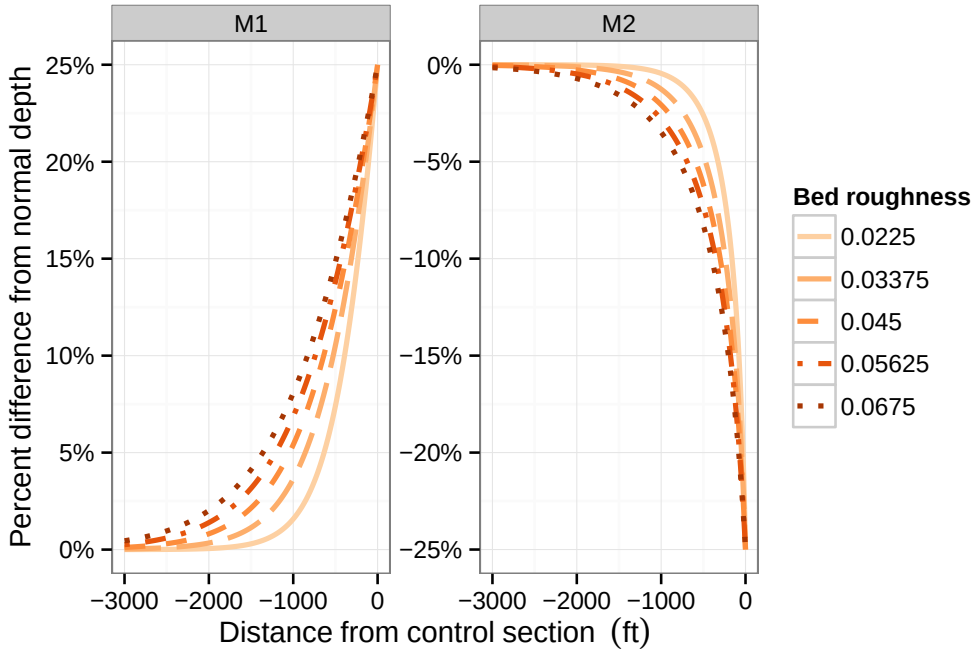

Figure 3: M1 and M2 water-surface elevation profiles as percent difference from the normal depth under different roughness conditions.

roughness affects the distance that a depth disturbance propagates upstream under sub-critical flow conditions. From this we can interpret Manning's coefficient to not only determine the steady-state water-surface profile, but also indicate the degree to which a disturbed channel resists adjustment towards said profile. Note that the M1 and M2 profiles are not symmetric; that is, the point at which the water depth returns to the normal depth from a 25% increase in water depth at the control section is further upstream compared to that for a 25% decrease in depth. This is because the velocity term in the energy equation is non-linear (u^2) and therefore the energy gradient is steeper for the M2 profile.

Computation of unsteady flows

Open-channel unsteady flows are typically described using the St. Venant equations, also known as the shallow water equations. The St. Venant equations are derived from the Navier-Stokes equations based on a number of simplifying assumptions, including uniform density and hydrostatic pressure conditions. The one-dimensional St. Venant equations are written as

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{4}$$

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(Qu + g\bar{y}A\right) - gA\left(S_0 - S_f\right) = 0 \tag{5}$$

where Eq. (4) is the continuity, or mass conservation, equation (assuming no infiltration or precipitation) and Eq. (5) is the momentum equation. The terms Q, u, A and S_0 and S_f are as defined previously and g is gravitational acceleration. The term $\bar{y}$ results from averaging the pressure component and refers to the distance from the free surface to the centroid of the cross section. The **rivr** package provides multiple methods for solving these equations to compute unsteady flows through a prismatic channel. These methods are discussed in the following sections.

Simulations with the Kinematic Wave Model

The Kinematic Wave Model (KWM) provides a simplified solution to the St. Venant equations based on a truncated momentum equation that ignores inertial terms (Lighthill and Whitham, 1955). Eq. (5)

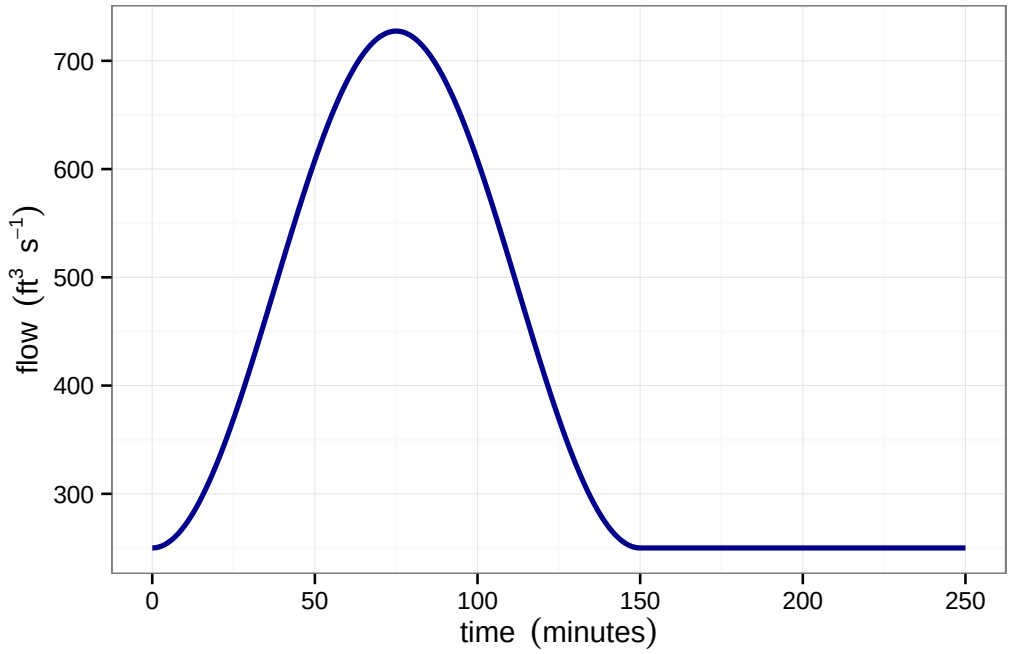

Figure 4: Flood wave hydrograph specified as the initial condition of the unsteady flow examples.

is instead expressed through the relation

$$A = \left(\frac{nQP^{2/3}}{C_m \sqrt{S_0}} \right)^{3/5} \tag{6}$$

where P is the wetted perimeter and S_0 is substituted for S_f in Manning's equation.

The KWM engine (and other engines, as we show later) provided by **rivr** can be called via the function `route_wave`. The behavior of the KWM is demonstrated by using it to calculate the evolution of a flood wave down a 100-foot wide rectangular channel that is 150,000 feet long, with a slope of 0.001 and a Manning's roughness of 0.045. The `boundary.condition` argument is a vector specifying the flow at the upstream boundary for an arbitrary period of time in intervals equal to the `timestep` argument, in seconds. The `monitor.nodes` and `monitor.times` arguments are used to specify the function outputs. The hydrograph (change in flow over time) at the upstream boundary, shown in Figure 4, is modeled as a symmetrical flood wave and is conveniently expressed as

$$Q(t) = \begin{cases} 250 + \dfrac{750}{\pi} \left(1 - \cos \dfrac{\pi t}{4500} \right) & \text{if } t \leq 9000 \\ 250 & \text{if } t \geq 9000 \end{cases} \tag{7}$$

where t is in seconds.

```
slope <- 0.001       # channel slope (vertical ft / horizontal ft)
extent <- 150000     # channel length (ft)
mannings <- 0.045    # Manning's roughness
width <- 100         # channel bottom width
sideslope <- 0       # channel side slope (horizontal ft / vertical ft)
Cm <- 1.486          # conversion factor for Manning's equation
g <- 32.2            # gravitational acceleration (ft/s^2)

numnodes <- 601                    # number of finite-difference nodes
dx <- extent/(numnodes - 1)        # distance between nodes (ft)
dt <- 100                          # time interval (s)

monpoints <- c(1, 201, 401, 601)                # Nodes to monitor
montimes <- seq(1, length(boundary), by = 10)   # time steps to monitor
```

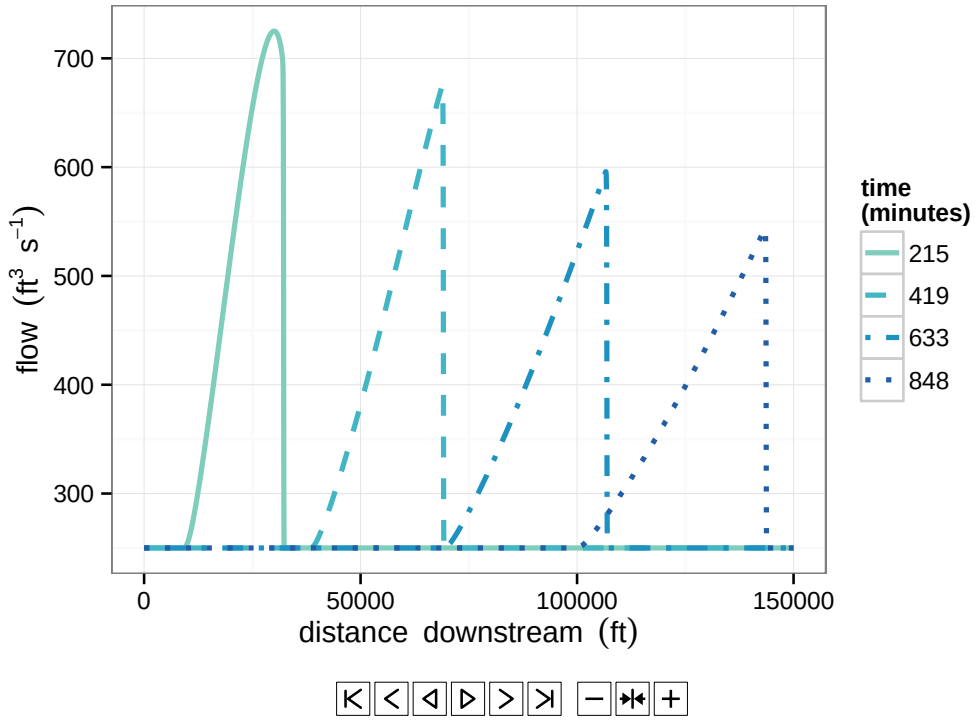

Figure 5: Progression of the flood wave calculated using the KWM engine with a spatial resolution of 25 feet and a temporal resolution of 12 seconds.

```
initflow <- 250                                  # initial flow condition (ft^3/s)
times <- seq(0, 76000, by = dt)
boundary <- ifelse(times < 9000,                 # upstream hyrograph (ft^3/s)
  250 + (750/pi)*(1 - cos(pi*times/(4500))), 250)

route_wave(slope, mannings, Cm, g, width, sideslope, initflow, boundary,
  timestep = dt, spacestep = dx, numnodes = numnodes,
  monitor.nodes = monpoints, monitor.times = montimes,
  engine = "Kinematic")
```

While the KWM is easy to use and implement, it ignores wave damping and attenuation resulting in unrealistic predictions of the shape of the flood wave as it progresses downstream. The truncation of Eq. (5) results in what is known as "kinematic shock", i.e. the shape of the flood wave becomes increasingly asymmetrical as it progresses downstream. This effect is apparent in Figure 5, which shows the wave front approaching a vertical line as the flood wave travels along the channel.

The **rivr** package implements the KWM with an explicit, first-order accurate backwards-difference scheme. While such schemes are simple to conceptualize and implement, they are sensitive to numerical error and can be unstable. A necessary condition for stability is that the Courant Number C is less than unity, i.e.

$$C \equiv \frac{u\Delta t}{\Delta x} \leq 1 \tag{8}$$

where u is defined here as the initial velocity at the upstream boundary and Δx and Δt are the spatial and temporal resolution of the model. Numerical error is propagated through successive calculations and results in a systematic decrease in the accuracy of the solution as the flood wave progresses downstream, apparent in Figure 5 as a decreasing peak flow magnitude. The sensitivity of the solution generated by the KWM engine to model resolution is assessed by varying Δx and Δt while maintaining a Courant number of 0.7 to ensure numerical stability. The effect of numerical error on model accuracy is inferred by evaluating the shape of the hydrograph at a cross-section 50,000 feet downstream. Figure 6 shows the shape of the flood wave at said cross-section for selected spatial resolutions. As resolution decreases, the peak flow downstream decreases due to the effects of numerical error. In addition, the severity of the kinematic shock decreases with decreasing resolution due to greater numerical dispersion. Numerical error can be reduced by applying finer model resolutions at the cost of increased computation time. Table 2 summarizes the effect of model resolution on the peak flow value 50,000 feet downstream; for the example shown here, increasing the model resolution incurs

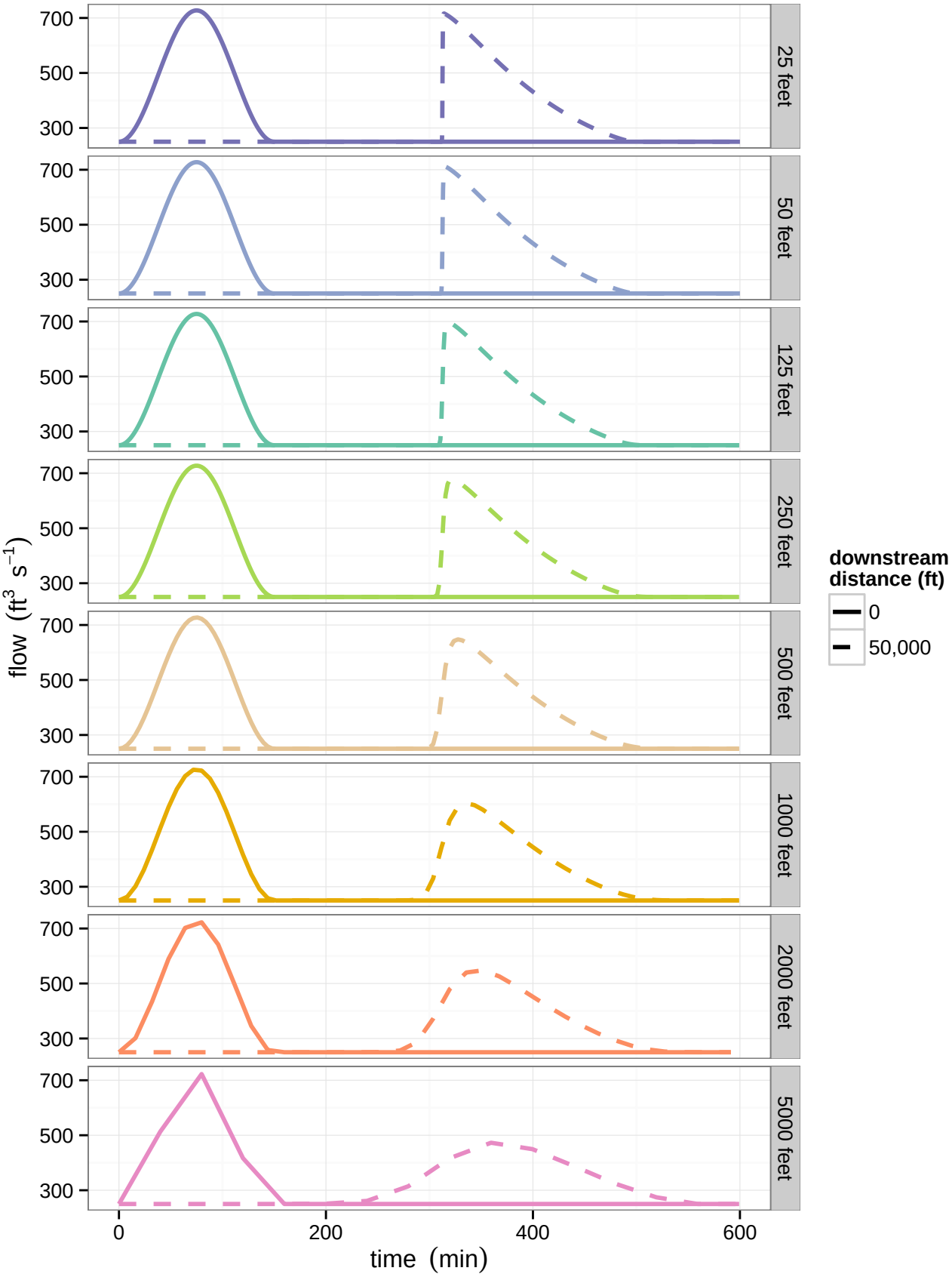

Figure 6: Evolution of the flood wave hydrograph 50,000 feet downstream using the KWM engine, for a variety of model resolutions. Note that at coarser resolutions the wave appears to attenuate, but this is solely the result of numerical error.

Table 2: KWM engine resolution and accuracy 50,000 feet downstream. Temporal resolution is matched to the selected spatial resolution to maintain a Courant number of 0.7. Finer resolutions reduce the impact of numerical error but show diminishing returns for resolutions finer than 50 feet. Computation times are for R v3.2.2 (x64) running on an Intel i7-4500U CPU with 8GB of RAM.

Δx (ft)	Δt (s)	% error (peak flow)	cost (s)
25	11.98	-1.53	221.02
50	23.97	-2.19	53.25
125	59.92	-4.04	8.48
250	119.84	-6.68	2.48
500	239.68	-10.96	0.50
1,000	479.36	-17.02	0.14
2,000	958.72	-24.69	0.05
5,000	2,396.80	-35.02	0.02

substantial costs in computation time with diminishing returns on accuracy for spatial resolutions finer than 50 feet.

Simulations with the Dynamic Wave Model

The previous example showed that the KWM gives unrealistic predictions of a flood wave routed through a prismatic channel with mild slope. This is largely due to the exclusion of inertial forces by truncating the momentum equation. Unsteady flow models that solve complete forms of Eq. (5), known as dynamic wave models (DWM), can provide more realistic simulations of unsteady flow. The **rivr** package provides a DWM engine accessed in a similar manner to the KWM engine. The DWM engine uses the MacCormack predictor-corrector scheme (MacCormack, 2003) to provide solutions that are second-order accurate in space. As shown in Figure 7, flood waves modeled with the DWM attenuate as they travel through the channel.

The DWM engine differs from the KWM engine in that the downstream boundary condition must be known. The **rivr** package employs the Method of Characteristics (MOC) to resolve fluid flow across the upstream and downstream boundaries. By using MOC, the DWM engine can accommodate a variety of boundary specifications; the upstream and downstream boundaries can be individually specified in terms of either flows or water depths. The downstream boundary can alternatively be specified as a zero-gradient condition, which allows flow to be routed out of the channel by assuming that the flow or water depth is constant across the final (downstream) two nodes in the model domain. This results in "smearing" of the flood wave at the downstream boundary, but in some practical cases this may be preferable to direct specification of depth or flow. When depths are specified at the boundaries, the characteristic equations are solved directly; when flows are specified, a Newton-Raphson scheme is used to iteratively solve for depths and velocities simultaneously.

To illustrate use of the DWM, the example from the previous section is repeated. The channel geometry, extent and upstream boundary condition are unchanged. A zero-gradient condition is specified at the downstream boundary. Example code using the DWM engine is shown below.

```
slope <- 0.001       # channel slope (vertical ft / horizontal ft)
extent <- 150000     # channel length (ft)
mannings <- 0.045    # Manning's roughness
width <- 100         # channel bottom width
sideslope <- 0       # channel side slope (horizontal ft / vertical ft)
Cm <- 1.486          # conversion factor for Manning's equation
g <- 32.2            # gravitational acceleration (ft/s^2)

numnodes <- 601                  # number of finite-difference nodes
dx <- extent/(numnodes - 1)      # distance between nodes (ft)
dt <- 100                        # time interval (s)

monpoints <- c(1, 201, 401, 601)                # Nodes to monitor
montimes <- seq(1, length(boundary), by = 10)   # time steps to monitor

initflow <- 250                                 # initial flow condition (ft^3/s)
times <- seq(0, 76000, by = dt)
```

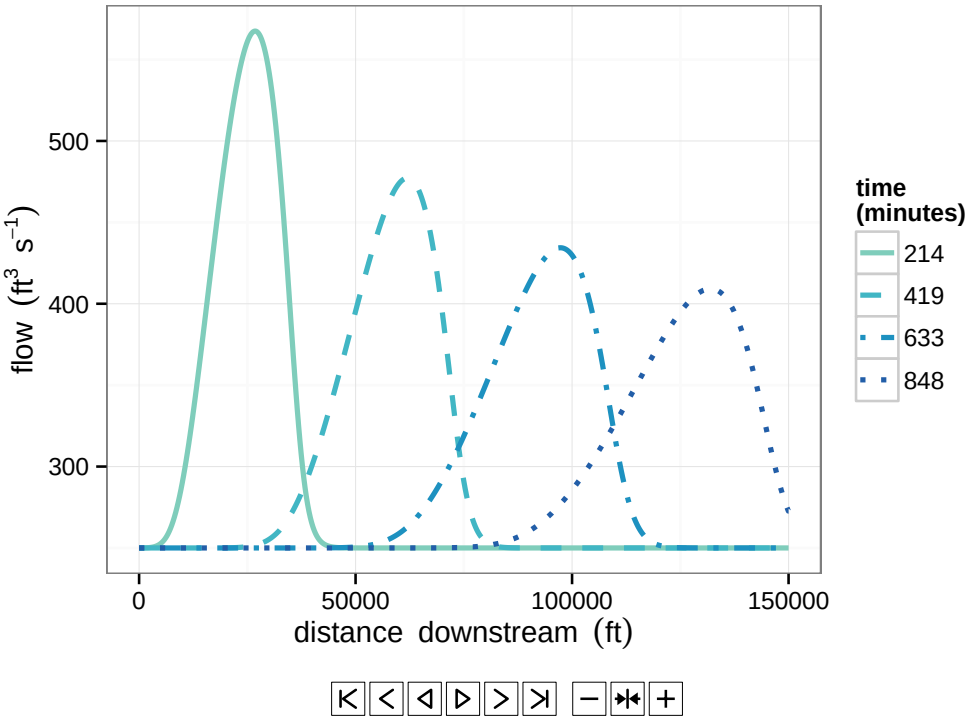

Figure 7: Progression of the flood wave simulated using the DWM engine with a spatial resolution of 50 feet and a temporal resolution of 2.1 seconds. Flow is routed out of the channel by assuming the flow gradient is zero at the downstream boundary.

Table 3: DWM engine resolution and accuracy 50,000 feet downstream. Error values for peak flow magnitude and timing are based on a standard benchmark (Sobey, 2001).

Δx (ft)	Δt (s)	% error (peak flow)	% error (time to peak)	cost (s)
50	2.05	0.79	0.98	1,690.05
125	5.13	0.77	1.03	268.67
250	10.27	0.74	1.11	67.32
500	20.54	0.68	1.26	16.87
1,000	41.07	0.44	1.96	4.28

```
boundary <- ifelse(times < 9000,                  # upstream hyrograph (ft^3/s)
  250 + (750/pi)*(1 - cos(pi*times/(4500))), 250)
downstream <- rep(-1, length(boundary))           # values < 0 specify zero-gradient

route_wave(slope, mannings, Cm, g, width, sideslope, initflow, boundary,
  downstream, timestep = dt, spacestep = dx, numnodes = numnodes,
  monitor.nodes = mpoints, monitor.times = mtimes, engine = "Dynamic",
  boundary.type = "QQ")
```

The DWM engine, like the KWM engine, uses an explicit scheme and is subject to stability constraints. In fact, the DWM engine is more prone to numerical instability compared to the KWM engine and typically requires Courant numbers on the order of 0.06 (i.e. an order-of-magnitude increase in the required temporal resolution compared to the KWM engine for a given spatial resolution). This disadvantage is offset by a significant reduction in numerical error provided by the advanced finite-differencing scheme. As shown in Figure 8, spatial resolutions of 1,000 feet provide accurate predictions at 50,000 feet downstream, at the cost of severe instability near the downstream boundary. Resolutions of 500 feet provide stable solutions throughout the full channel extent for a modest computation cost, as shown in Table 3.

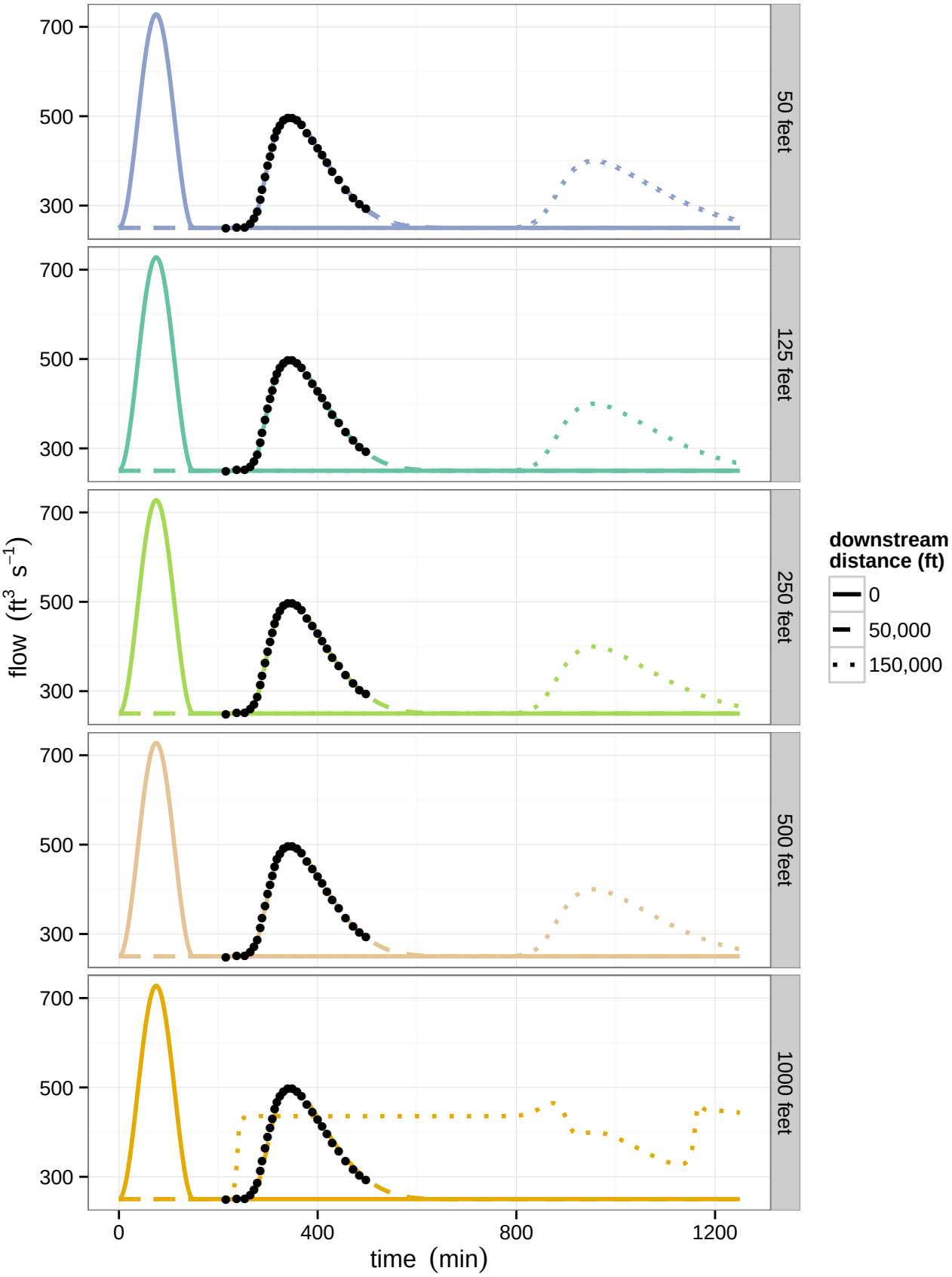

Figure 8: Evolution of the flood wave across both boundaries and at 50,000 feet downstream, for a variety of model resolutions. Validation data from Sobey (2001) are shown as points. The solution becomes unstable at very coarse resolutions but otherwise maintains a high degree of accuracy.

Simulations with discontinuous boundary conditions

The DWM engine can accommodate boundary conditions that specify zero flows (velocities) but non-zero depths, such as can occur by the sudden closure of sluice gates. The last unsteady-flow example illustrates use of the DWM engine to simulate unsteady flow conditions following sudden closure of a sluice gate at the downstream end of a trapezoidal spillway. This example mimics example 14-1 in Chaudhry (2007). The upstream boundary condition is specified as a constant depth to represent water level in a large reservoir. Example code for specifying discontinuous boundary conditions is shown below.

```
slope <- 0.00008          # channel slope (vertical m / horizontal m)
extent <- 5000            # channel length (m)
mannings <- 0.013         # Manning's roughness
width <- 6.1              # channel width (m)
sideslope <- 1.5          # channel sideslope (horizontal m / vertical m)
g <- 9.81                 # gravitational acceleration (m/s^2)
Cm <- 1.0                 # conversion factor for Manning's equation (SI units)
extent <- 5000            # channel length (m)

numnodes <- 51                          # number of nodes
dx <- extent/(numnodes - 1)             # spatial resoution (m)
dt <- 10                                # temporal resolution (s)

mp <- c(1, 16, 26, 31, 51)              # monitor nodes
mt <- c(1, 61, 101, 161, 201)           # monitor time steps

ic <- 126                               # initial condition (m^3/s)
bctime <- 2000
bc <- rep(5.79, round(bctime/dt) + 1)   # constant depth upstream (m)
dc <- rep(0, length(bc))                # flow dropped to 0 on first timestep

results <- route_wave(slope, mannings, Cm, g, width, sideslope, ic, bc, dc,
  timestep = dt, spacestep = dx, numnodes = numnodes, engine = "Dynamic",
  boundary.type = "yQ", monitor.nodes = mp, monitor.times = mt)
```

Figure 9 shows that the MacCormack scheme can produce small instabilities in the solution where gradients are high. The **rivr** package provides access to an alternative DWM engine—the Lax diffusive scheme (Lax, 1954)—which is also second-order accurate in space. Whereas the Lax scheme is less accurate than the MacCormack scheme at coarser resolutions, it can provide smoother solutions to discontinuous boundary problems. The Lax scheme can be accessed with `route_wave` via the argument `scheme = "Lax"`.

Opportunities for extension

The **rivr** package was designed such that additional analysis capabilities could be implemented in future versions with minimal modification to the existing code base. A wide variety of extensions could be supported by the existing interface, both in terms of numerical methods and problem complexity. In particular, there are many avenues for contribution to the unsteady flow analysis capabilities of **rivr**.

Additional numerical schemes for unsteady flow analysis in prismatic channels—such as implicit finite-difference schemes and finite-element methods—could be implemented with only superficial changes to the current function interface provided by `route_wave`, i.e. new algorithms could be made accessible through additional keyword options in existing arguments provided the underlying engines followed the input template used by the KWM and DWM engines. Variable-step methods could also be implemented with minimal changes, most simply by allowing the user to define an initial or "default" time step and using an interpolation function to compute the upstream boundary condition at intermediate time steps when required. The authors consider the degree of effort required to implement such additional numerical schemes in **rivr** to be on par with typical expectations for an individual project assignment in a graduate-level computational hydraulics or numerical methods course.

Support for prismatic channels with arbitrary or compound geometry could be accomplished with only minor restructuring of the channel geometry function, but would introduce additional complexity to the specification of channel roughness (Yen, 2002) and the computation of normal and

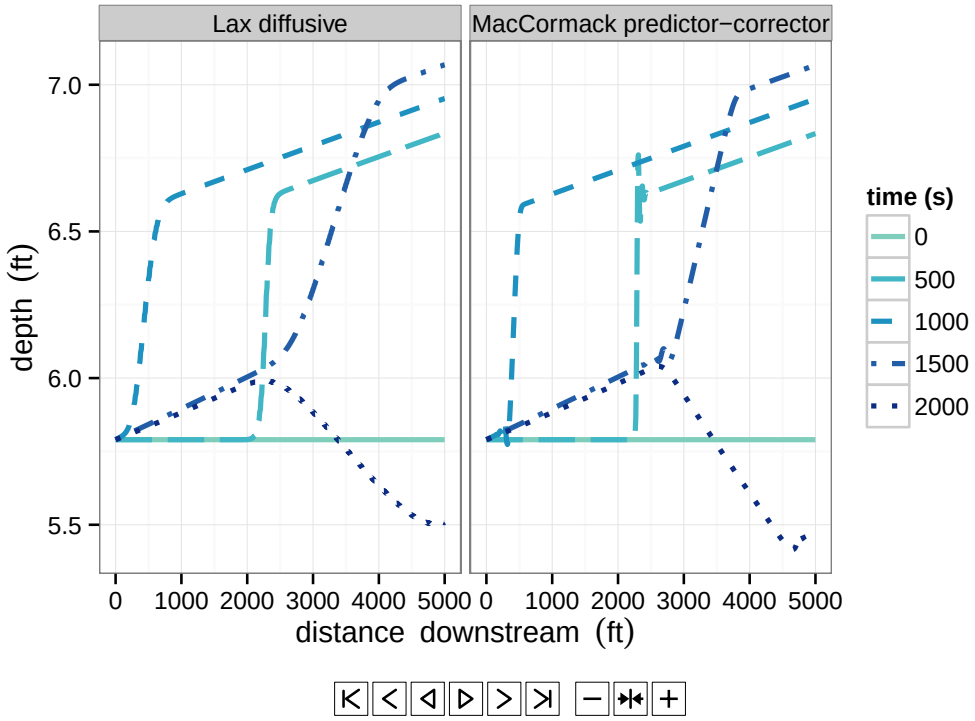

Figure 9: Evolution of the water-surface profile through time following sudden closure of a control gate. The Lax scheme produces smoother profiles, while the MacCormack scheme predicts steeper gradients and produces small numerical instabilities.

critical depth (Chaudhry and Bhallamudi, 1988; Younis et al., 2009). Support for compound channels is currently being considered for the next major version release of **rivr**.

Specification of variable channel geometry and slope, additional friction terms (e.g. bedform drag, hydraulic structures) and additional source or sink terms (e.g. hillslope runoff, infiltration) would require some modification to the current implementation but would largely consist of allowing users to specify channel properties as vectors rather than as single values, and modifying the underlying C++ code to access the appropriate vector elements when computing solutions at individual nodes. These modifications would best be developed prior to, or in conjunction with, the implementation of an implicit scheme. Such changes would, however, add considerable complexity to the formulation of unsteady flow problems and could potentially distract from the primary purpose of the package as an accessible tool for teaching.

Support for channel networks would likely require the development of a new function interface that would essentially couple three or more instances of `route_wave` (e.g. two channels merging into one) and provide a keyword argument for selecting different methods for resolving flow depth and velocity at the junctions. Implementing such functionality would likely be a much larger endeavor compared to the other extensions discussed, and could pose additional challenges in terms of obtaining reasonably-fast solutions for moderately-large channel networks. It is important to note, however, that discussion of channel network modeling is often neglected in open-channel hydraulics courses due to the high complexity of problem formulation and solution algorithms; making channel network solution algorithms demonstrable through **rivr** would therefore be of considerable educational value.

Concluding remarks

This article describes **rivr**, a first-of-its-kind R/C++ package for one-dimensional open-channel flow computation with educational applications to hydraulic engineering degree programs and related disciplines. The package provides a reliable and flexible toolset for modeling open-channel flows via a simple function interface. Model outputs are formatted to facilitate analysis, tabulation, visualization and export. Example computations of backwater curves and flood wave routing are provided to demonstrate functionality and validate model results, and potential extensions to the package are discussed. This article, along with the package documentation, provides information on the governing equations and numerical implementations as well as a suite of use examples.

Michael C. Koohafkan
Hydrologic Sciences Graduate Group
University of California, Davis, CA 95616, USA
koohafkan@ucdavis.edu

Bassam A. Younis
Department of Civil and Environmental Engineering
University of California, Davis, CA 95616, USA
bayounis@ucdavis.edu

Bibliography

W. Chang, J. Cheng, J. Allaire, Y. Xie, and J. McPherson. *shiny: web application framework for R*, 2015. URL http://CRAN.R-project.org/package=shiny. R package version 0.12.2. [p2]

M. H. Chaudhry. *Open-channel flow*. Springer, 2007. [p3, 12]

M. H. Chaudhry and S. M. Bhallamudi. Computation of critical depth in symmetrical compound channels. *Journal of Hydraulic Research*, 26(4):377–396, 1988. [p13]

P. D. Doubt. Classification system for varied flow in prismatic channels. Technical Release 47, United States Department of Agriculture, Soil Conservation Service, Engineering Division, Littleton, Colorado, USA, feb 1971. [p3]

D. Eddelbuettel and R. Francois. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011. URL http://www.jstatsoft.org/v40/i08/. [p2]

P. D. Lax. Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Communications on Pure and Applied Mathematics*, 7(1):159–193, 1954. [p12]

L. B. Leopold. Downstream change of velocity in rivers. *American Journal of Science*, 251(8):606–624, 1953. [p1]

M. Lighthill and G. Whitham. On kinematic waves. I. flood movement in long rivers. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 229(1178):281–316, 1955. [p5]

R. MacCormack. The effect of viscosity in hypervelocity impact cratering. *Journal of Spacecraft and Rockets*, 40(5):757–763, 2003. [p9]

R. J. Sobey. H11: Hydrograph routing. In *Review of One-Dimensional Hydrodynamic and Transport Models*, chapter 14. Bay-Delta Modeling Forum, jun 2001. URL http://www.cwemf.org/1-DReview/. [p10, 11]

Y. Xie. knitr: A comprehensive tool for reproducible research in R. In V. Stodden, F. Leisch, and R. D. Peng, editors, *Implementing Reproducible Computational Research*. Chapman and Hall/CRC, 2014. URL http://www.crcpress.com/product/isbn/9781466561595. ISBN 978-1466561595. [p2]

B. C. Yen. Open channel flow resistance. *Journal of Hydraulic Engineering*, 128(1):20–39, 2002. [p12]

B. Younis, V. Sousa, and I. Meireles. Prediction of the asymptotic water depth in rough compound channels. *Journal of Irrigation and Drainage Engineering*, 135(2):231–234, 2009. [p13]