# Automated Generation of Hydraulic Models and its use for Sensitivity Analysis
## Response Time Variation with Channel Parameters

Van Nooijen, Ronald; Kolechkina, Alla

**DOI**
10.1016/j.ifacol.2024.07.091

**Publication date**
2024

**Document Version**
Final published version

**Published in**
IFAC-PapersOnline

**Citation (APA)**
Van Nooijen, R., & Kolechkina, A. (2024). Automated Generation of Hydraulic Models and its use for Sensitivity Analysis: Response Time Variation with Channel Parameters. *IFAC-PapersOnline*, *58*(2), 55-60. https://doi.org/10.1016/j.ifacol.2024.07.091

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 58-2 (2024) 55–60

# Automated Generation of Hydraulic Models and its use for Sensitivity Analysis: Response Time Variation with Channel Parameters

**Ronald van Nooijen** [*] **Alla Kolechkina** [**]

[*] *Water Management Department, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, Netherlands*

[**] *Delft Center for Systems and Control, Faculty of Mechanical, Maritime and Materials Engineering, Delft University of Technology, Delft, Netherlands*

**Abstract:** A first prototype of a Python package that can generate input for, perform runs of, and the analyse output of the Delft3D FM 1D2D simulation software corresponding to both simple and complex canal networks is demonstrated. The package can, for instance, generate input files for a canal system with a wide range of friction values and a range of variations on the design cross sections and structures used. It can then start parallel Delft3D runs for these input files. Finally, results can be analysed from Python. Results are presented for the rise time associated with a step change in the discharge for a prismatic canal with a trapezoidal cross section and either a weir or a gate at the downstream end. The size of the step change, the parameters of the cross section, the friction, and the canal slope are varied.

*Keywords:* Modeling and identification of environmental systems, Real time control of environmental systems, Hydroinformatics, Open channel flow, Weir, Gate

## 1. INTRODUCTION

This study contains example code from a Python package written to automate model building and results on delay times in open channels, and a plea for standardized tests of hydraulic simulation software and standard water systems to be used to evaluate control methods, verify their robustness to small system changes, and compare their performance. To realize such tests for non-trivial systems and non-trivial test sets, it is necessary that model building is automated. While (almost) all suppliers of simulation software provide the means to import Geographical Information System (GIS) data, and some include program specific means of including control actions, the automated building models is decidedly less well documented, if it is supported at all. The paper starts off by emphasizing the importance of control of water systems by linking water to the 17 Sustainable Development Goals (SDGs) set by the United Nations (UN, 2022). Next, a short overview of the documented capabilities of several simulation packages with respect to control and model building is given. Then, an example of use of the proposed Python package is provided. Subsequently, the setup of an experiment to determine the response time of a set of canals that differ in cross section and downstream boundary condition (fixed weir or fixed gate) is described, and the results are discussed. Finally, some conclusions are stated.

### *1.1 Water, the SDGs, and control*

The 17 Sustainable Development Goals (SDG) summarize the problems the world faces today. For Clean Water and Sanitation (SDG 6), the link to water management is clear. Apart from its intrinsic importance, water is also essential to food production, linking it to Zero Hunger (SDG 2). For many communities, safety from floods, which is part of Sustainable Cities and Communities (SDG 11), depends on proper water management. Moreover, to achieve Peace Justice and Strong Institutions (SDG 16), peaceful, equitable, and harmonious distribution of water resources is a prerequisite. Meeting these goals will depend strongly on the ability of societies and of the international community to reach agreements on the measures needed to achieve the goals. Many aspects of that process lie outside the scope of science and engineering. However, some of the problems are of a scientific or technical nature. In water management, in particular, translating an agreement in principle between the many different stakeholders into practice may require tools that are not yet available.

More specifically, work is needed on automatic control in large irrigation and drainage networks. Such water systems often require the use Model Predictive Control (MPC) to accommodate the many and sometimes conflicting needs of different users (Mayne, 2014; van Nooijen et al., 2021; Horváth et al., 2015). The uncertainties in the input and complexity of the goals lead to complex optimization problems. If such problems are formulated in terms of traditional objective functions, then many runs of the internal model in MPC may be needed to solve this optimization problem. This in turn means it is necessary to reduce the time needed to run the internal model in MPC. A full hydraulic model of the system is unlikely to be fast enough. However, the simplified model will need to be validated. Preferably under different circumstances and for a range of different states of system maintenance.

10.1016/j.ifacol.2024.07.091

The large number of different systems more or less forces an approach where a software package for hydrodynamic simulation is used as a model of reality, because such data is rarely available even for existing systems, and the construction of physical scale models would be prohibitively expensive. To accommodate a large number of different models, model parameters and scenarios, an approach where the software package is only accessible through a Graphical User Interface (GUI) without scripting support is not very practical, unless the authors of the package anticipated all the needs of this particular use case. Even with scripting support, a package that is operated through a GUI may not be the best way to go, as it is more difficult to guarantee that all actions are reproducible. In other words, facilities to perform large numbers of runs with slightly different sets of canal system parameters are needed. This holds for tests of the accuracy of simplified canal models on single canal reaches, as well as for tests of robustness of control methods on systems of canals. That type of support is either not present or not well documented for most of the programs that provide 1D shallow water flow simulation (Table 1). These packages tend to focus on an easy-to-use GUI and GIS data import.

This lack of support for large scale studies is reflected by the literature. There are not that many studies on testing control systems for open channel networks on standard problems. For instance, for the standard test canals proposed in Clemmens et al. (1998), a Web of Science search for studies using these channels for control testing employing the search criterion “ALL=(”ASCE test canal”)” turned up only 9 hits. There are many articles that investigate particular forms of control, but they apply the method to a system particular to the paper and tend not to perform extensive sensitivity tests. As another example, the search for studies that test or use the simplified canal model introduced in Litrico and Fromion (2004) “(ALL=(Integrator Delay Zero) OR ALL=(IDZ)) AND (ALL=(Canal) OR ALL=(Channel))” resulted in just 18 hits; and the number of canals used to test the model tended to be below 10. Varying the keywords did not result in a large number of additional studies appearing. Given the low number of affordable and reliable software packages for open channel flow and the effort needed to build a system model in such a package, this is not surprising.

With this in mind, a prototype of a Python package was developed that can be used to quickly generate models of water systems with, for example, different friction values, different canal cross sections, different structures, and different slopes. It generates output for the Delft3D FM simulation software (Deltares, 2023) and contains code to perform the corresponding runs using the command line version of Delft3D. Analysis of the output of Delft3D from Python is already possible as Delft3d provides output in standard NETCDF3 format.

A second application of such a package would be in education, where students could design a simple irrigation system and run a full hydrodynamic simulation of that system without having to go through a GUI to produce their layout and variations thereon.

### 1.2 Some of the available software

Several large institutes provide packages that can simulate 1D shallow water flow in channels. Some of these packages are: MIKE 1D (DHI, 2024), Infoworks ICM (Autodesk, 2024), ESTRY (TUFLOW, 2024) , HWC-RAS (U.S. Army Corps of Engineers, 2024), Delft3D (Deltares, 2024). Some characteristics of these packages can be found in Table 1. Delft3D FM 1D2D version 2023.03 was used in this paper because at the time of writing this paper, it was the package the authors were most familiar with.

## 2. METHOD

### 2.1 Python code example

An example of how the package could be used to automate creation of sets of input files for Delft3D can be found below. To fit the code into one column, some details have been omited or replaced by “...”.

```
for cs in cross_sections:
  for ft in friction_terms:
    wnb = WaterwayNetworkBuilder(...)
    arc_reach = PrismaticReach(
      ..., friction_term=ft,
      cross_section=cs)
    a = Arc.fixed_from(
      ..., reach=arc_reach, direction='E')
    wnb.add_arc(a)
    a.add_cs_location_and_definition(...)
    a.arc.add_cs_location(...)
    wnb.force_add_simple_weir(...)
    a.add_cs_location(...)
    a.add_cs_location(...)
    a.from_vertex.boundary_condition = \
      DischargeTimeSeriesBC(...)
    a.to_vertex.boundary_condition = \
      ConstantWaterLevelBC(...)
    a.mesh = numpy.linspace(...)
    a.initial_condition = \
      InitialWaterLevel(...)
    wnb.write_d3d_files(...)
```

### 2.2 Channel configurations

In this paper, fixed slope prismatic channels with trapezoidal cross sections were considered. All canals were 10 km long. The trapezoidal cross sections were parametrized by width $B$ and side slope $m$, see Fig. 1. The channel parameter set was completed by the slope $S_\mathrm{b}$ of the channel bed, given as the ratio of vertical drop in bed level to horizontal distance, the friction term used, and the appropriate friction coefficient. For trapezoidal canals, combinations of $b = 0.8, 0.9, 1.0, 1.1, 1.2$ m, $m = 1.00, 1.25, 1.75, 2.00$ m/m, gradient $S_\mathrm{b} = 9.4, 10, 10.6$ cm/km, and a Strickler friction term with coefficient $k_\mathrm{S} = 40, 41, \ldots, 50$, $\mathrm{m}^{1/3}/\mathrm{s}$ were used. For each combination of parameters, simulations were performed for downstream depths of 1.25 times the normal depth. Simulations were carried out with either a fixed weir or a fixed gate at the downstream end of the canal, where the dimensions of the structure were chosen in such a way that the desired

Table 1. 1D channel flow packages (information from product websites consulted on 2022/11/9)

| Package | Scripting Language | API | Open Source | Free version |
|---|---|---|---|---|
| ESTRY | For structure operation | Unknown | No | Limited demo |
| MIKE 1D | C# | Yes | No | Unknown |
| Delft3D FM 1D2D | IronPython | Partial | | Partial |
| HEC-RAS | Not built-in | HECRASController | No | Yes |
| Infoworks ICM | Unknown | Unknown | No | No |

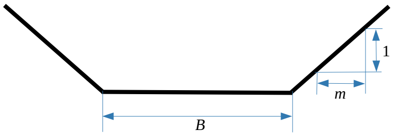

Fig. 1. Trapezoidal cross section

downstream water level was achieved for the initial flow rate. All hydrodynamic simulations were started with an approximately correct initial steady state for a flow rate corresponding to a normal depth of 1 m and were then run for a warm-up period to reach a true steady state. Next, a discharge change of -6%, -4%, -2%, 2%, 6% was imposed at the upstream end of the channel. The process of building and running the hydrodynamic models in Delft3D FM was automated.

### *2.3 Actuators used*

Please note that in water management, levels are always given relative to a fixed vertical (or altimetric) datum; the term depth is reserved for the height of the water above the canal bottom. Two actuators were used: a rectangular weir in free flow and a rectangular sluice gate in free flow. The stage discharge relations given here are fairly standard and correspond to the implementation in Delft3D. For a weir in free flow, the discharge $q$ is a function of the water level $h$ immediately upstream of the weir

$$q = c_{\mathrm{w}} b \sqrt{g} \left( \frac{2}{3} \max(0, h - h_{\mathrm{crest}}) \right)^{3/2} \tag{1}$$

where $g$ is the gravitational acceleration ($9.81\,\mathrm{m\,s^{-2}}$), $b$ is the length of the weir crest measured orthogonally to the flow, $h_{\mathrm{crest}}$ is the level of the crest, and $c_{\mathrm{w}}$ is a constant that depends on the shape of the weir crest ($c_{\mathrm{w}} = 1$ was used). The water level downstream of the weir is assumed to be below the level of the weir crest. The inverse relation is

$$\frac{3}{2} \left( \frac{q}{c_{\mathrm{w}} b \sqrt{g}} \right)^{2/3} = h - h_{\mathrm{crest}} \tag{2}$$

For a gate in free flow, the discharge $q$ is a function of the water level $h$ immediately upstream of the gate

$$q = \mu c_{\mathrm{g}} b w \sqrt{2g} \left( \max(0, h - h_{\mathrm{sill}} - \mu w) \right)^{1/2} \tag{3}$$

where $w$ is the opening height, $b$ is the length of the gate sill measured orthogonally to the flow, $h_{\mathrm{sill}}$ is the level of the sill, $\mu$ is the contraction coefficient ($\mu = 1$ was used), and $c_{\mathrm{g}}$ is a constant that depends on the shape of the gate ($c_{\mathrm{g}} = 1$ was used). The water level downstream of the gate is assumed to be below the level of the sill of the gate. The inverse relation is

$$\frac{q^2}{2g \left( \mu c_{\mathrm{g}} b w \right)^2} = h - h_{\mathrm{sill}} - \mu w \tag{4}$$

## 3. RESULTS AND DISCUSSION

The main reason for developing this code was to automate the large numbers of runs needed to perform parameter sweeps on massively parallel hardware. The figures shown were selected from a much larger set that was generated both to test the Python package and to examine the effects of changes in cross section friction and canal gradient on the rise time.

The rise time is the time needed to go from 10% to 90% of the expected discharge change for a number of different canal cross sections. Figures 2 and 3 show rise times in a canal with a weir for gradients of 5 and 15 cm per km respectively. Figure 4 shows rise times in a canal with a gate for a gradients of 5 cm per km and a Strickler friction coefficient of $70\,\mathrm{m^{1/3}/s}$. Each subplot represents 25 Delft3D FM runs for a total of 200 runs.

Comparison of Fig. 2(a,c,e) with Fig. 2(b,d,f) and Fig. 3(a,c,e) with Fig. 3(b,d,f) shows that, for a canal with a fixed weir at the downstream end, the rise time does not differ very much between a step decrease and a step increase in inflow. This is very different for a canal with a fixed gate at the end. A comparison of Fig. 4(a) with rise times for a step decrease and Fig. 4(b) with rise times for a step increase shows a large difference.

Comparison of, for instance, Fig. 2(a) with Fig. 2(c) shows that friction has considerable influence on rise time. Next consider 3(b). It shows that for a trapezoidal cross section the dependence of rise time on both bottom width and side slope is non-linear. This is less obvious in Fig. 4(b).

Comparison of, for instance, Fig. 2(a) with Fig. 3(a) shows that for the lower gradient the rise time increases with increasing $m$, while for the higher gradient the rise time decreases with increasing $m$. The same holds for Fig. 2(b,c) and 3(b,c).

Interestingly, Fig. 2(a,c,e) and Fig. 2(b,d,f) show that for the weir, the rise time for a step increase is slightly lower than for a step decrease, while Fig. 4(a,c,e) and Fig. 4(b,d,f) show that for the gate, the rise time for a step increase is much higher than for a step decrease.

The difference in rise times between weir and gate is to be expected. As can be seen from the exponents of $q$ in (2) and (4), a change in $q$ will cause a change in level that is larger for a gate than for a weir.

## 4. CONCLUSIONS

The example given here applies to irrigation and drainage canals. The package can also be applied to networks of canals. Moreover, once the relevant parts of Delft 1D2D are available, river and sewer networks can also be constructed. This allows automated generation of

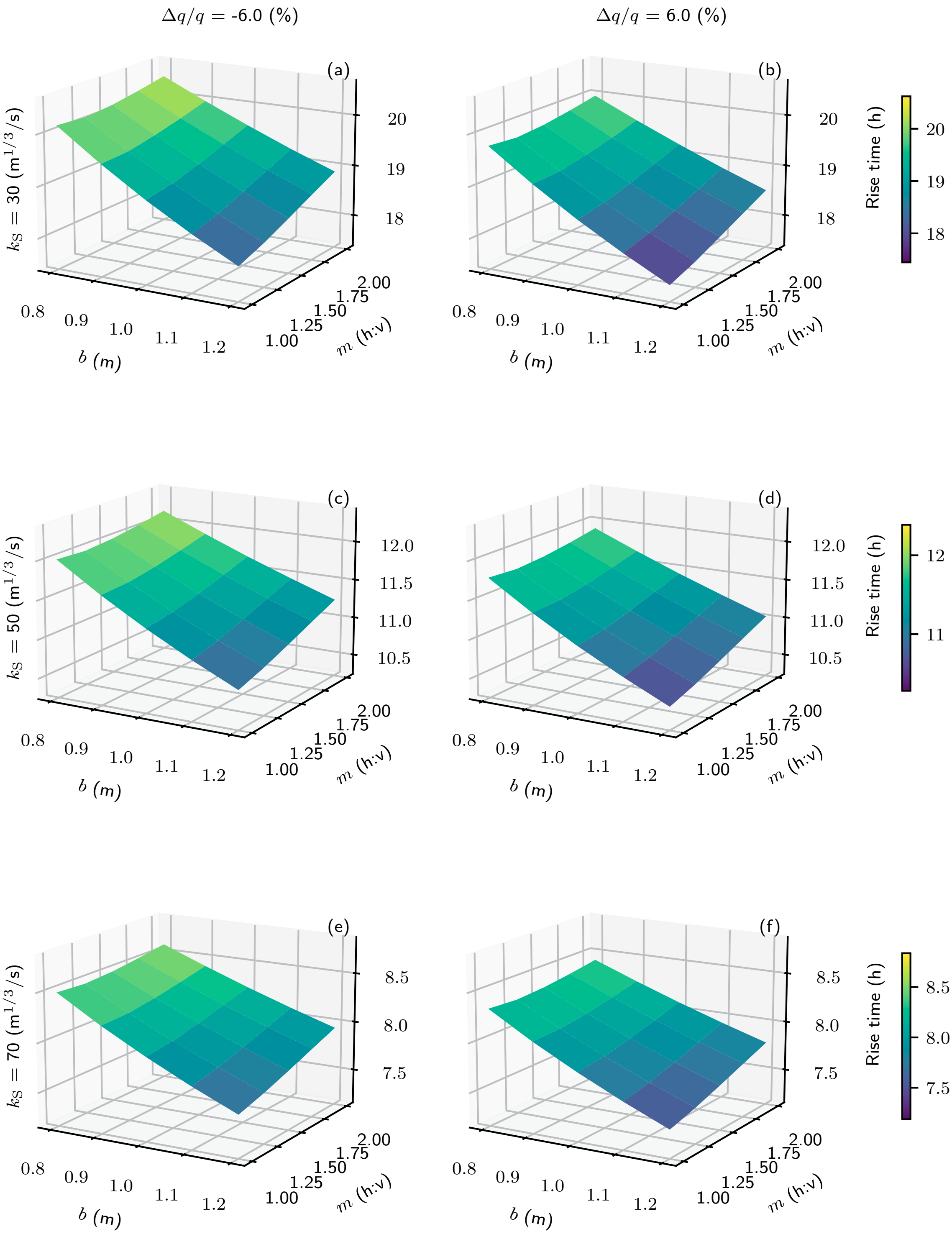

Fig. 2. Rise time for different friction values and discharge changes for a trapezoidal canal with a slope of 5 cm per km and a weir at the end

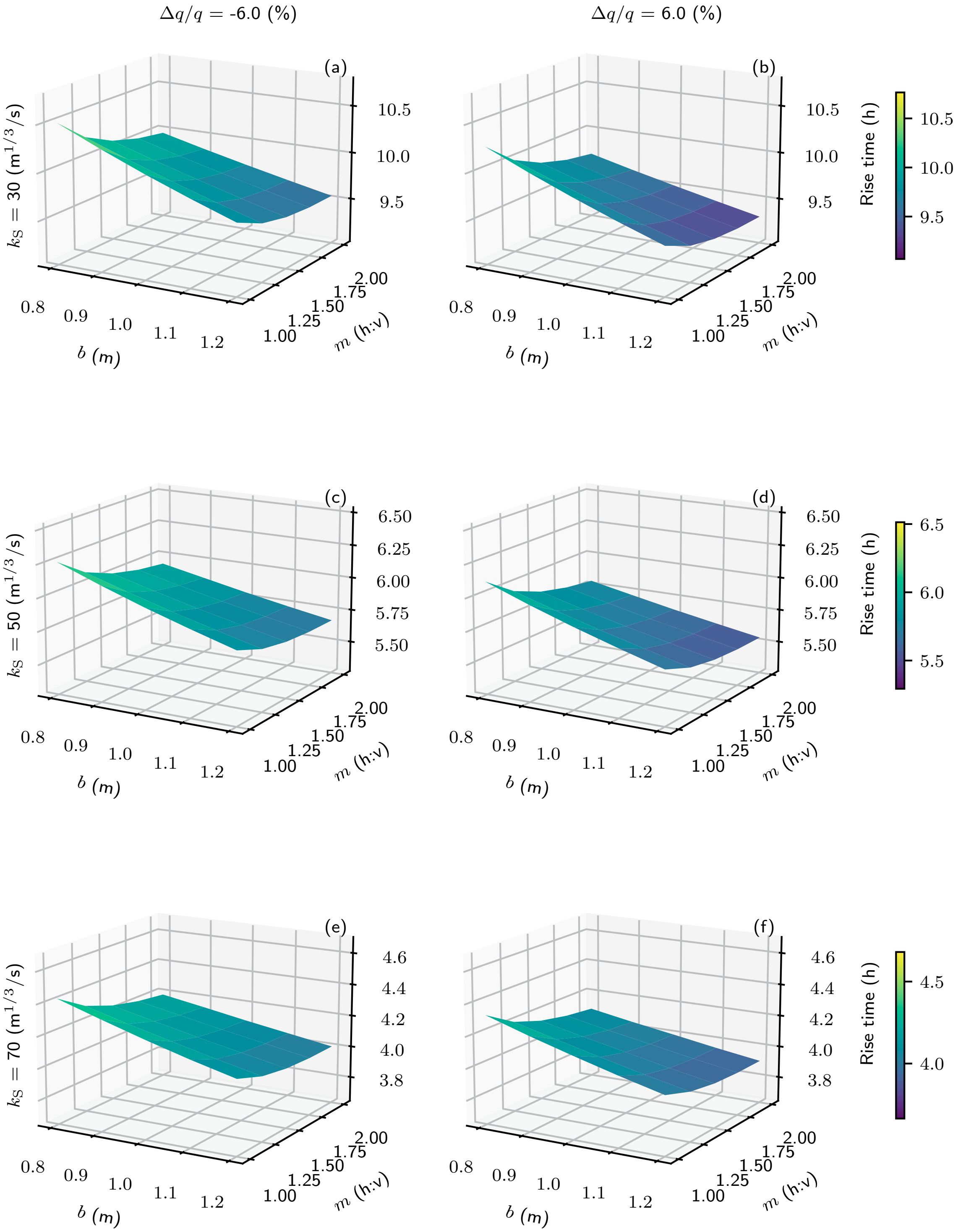

Fig. 3. Rise time for different friction values and discharge changes for a trapezoidal canal with a slope of 15 cm per km and a weir at the end

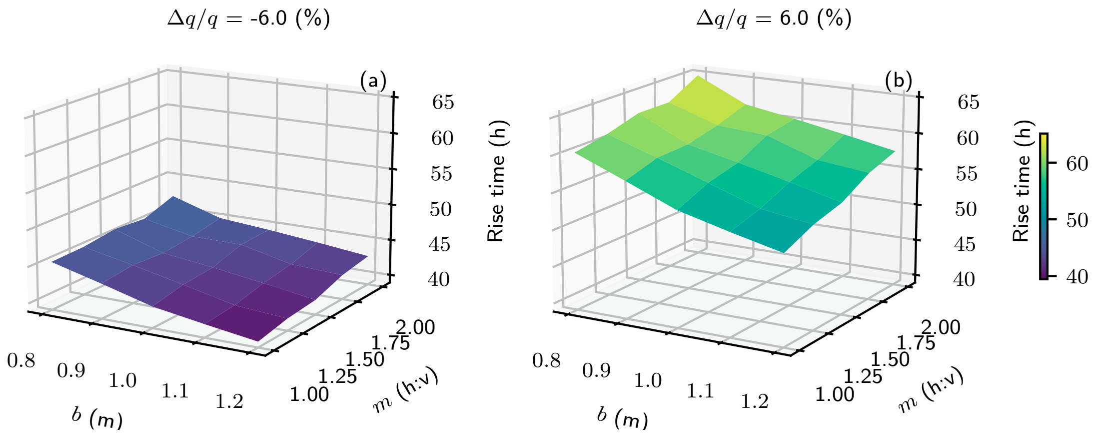

Fig. 4. Rise time for nine different cross sections in a canal with a slope of 5.0 cm per km, a Strickler friction coefficient of 70 $\mathrm{m}^{1/3}$/s and a gate at the end

such networks with varying parameters. Once the Python package is stable, it will be made available under an open source license. Deltares already makes available Delft3D FM under an open source license.

In future research, the results obtained for these canals will be compared with those of simplified models such as the Integrator Delay (ID)(Schuurmans et al., 1995) and Integrator Delay Zero (IDZ) approximations (Litrico and Fromion, 2004, 2009) of the one dimensional (1D) Saint Venant (or shallow water) equations (Chaudhry, 2022). Given the relative ease with which a framework could be set up for Delft3D, it would be interesting to repeat the test with another package in future, for example HEC-RAS or Mike 1D. With respect to possible applications in education, a similar package that was written for Sobek 3 was made available to students taking a course on water systems at Delft University of Technology.

## ACKNOWLEDGEMENTS

The authors wish to thank Deltares for making Delft3d FM 1D2D available for research use at Delft University of Technology. We also thank the referees for the care with which they review this work and for their valuable comments.

## REFERENCES

Autodesk (2024). Infoworks ICM. URL `https://www.autodesk.com/campaigns/infoworks-icm`. Consulted 28 April 2024.

Chaudhry, M.H. (2022). *Open-Channel Flow*. Springer Nature Switzerland AG, third edition. doi:10.1007/978-3-030-96447-4.

Clemmens, A.J., Kacerek, T.F., Grawitz, B., and Schuurmans, W. (1998). Test cases for canal control algorithms. *Journal of Irrigation and Drainage Engineering*, 124(1), 23–30. doi:10.1061/(asce)0733-9437(1998)124:1(23).

Deltares (2023). D-flow flexible mesh: Computational cores and user interface. Technical report. Released for: D-HYDRO Suite 1D2D 2023.

Deltares (2024). Delft3D 1D2D. URL `https://www.deltares.nl/en/software-and-data/products/sobek-and-delft3d-fm-suite-1d2d`. Consulted 28 April 2024.

DHI (2024). MIKE 1D. URL `https://docs.mikepoweredbydhi.com/engine_libraries/mike1d/mike1d_api/`. Consulted on 28 April 2024.

Horváth, K., Rajaoarisoa, L., Duviella, E., Blesa, J., Petreczky, M., and Chuquet, K. (2015). Enhancing inland navigation by model predictive control of water levels: The Cuinchy-Fontinettes case. In C. Ocampo-Martinez and R.R. Negenborn (eds.), *Transport of Water versus Transport over Water: Exploring the Dynamic Interplay of Transport and Water*, 211–234. Springer International Publishing, Cham. doi:10.1007/978-3-319-16133-4_12.

Litrico, X. and Fromion, V. (2004). Analytical approximation of open-channel flow for controller design. *Applied Mathematical Modelling*, 28(7), 677–695. doi:10.1016/j.apm.2003.10.014.

Litrico, X. and Fromion, V. (2009). *Modeling and Control of Hydrosystems*. Springer, London. doi:10.1007/978-1-84882-624-3.

Mayne, D.Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, 50(12), 2967–2986. doi:10.1016/j.automatica.2014.10.128.

Schuurmans, J., Bosgra, O.H., and Brouwer, R. (1995). Open-channel flow model approximation for controller design. *Applied Mathematical Modelling*, 19, 525–530. doi:10.1016/0307-904X(95)00053-M.

TUFLOW (2024). ESTRY. URL `https://www.tuflow.com/products/estry/`. Consulted on 28 April 2024.

UN. General Assembly (70th sess. : 2015-2016) (2015). Transforming our world : the 2030 agenda for sustainable development : resolution / adopted by the general assembly. Technical report.

U.S. Army Corps of Engineers (2024). HEC-RAS. URL `https://www.hec.usace.army.mil/software/hec-ras/`. Consulted 28 April 2024.

van Nooijen, R., Koutsoyiannis, D., and Kolechkina, A. (2021). Optimal and real-time control of water infrastructures. In *Oxford Research Encyclopedia of Environmental Science*. Oxford University Press. doi:10.1093/acrefore/9780199389414.013.627.