A Head Loss Pressure Boundary Condition for Hydraulic Systems

OpenFOAM® Journal

Volume 2, Pages 1–12
ISSN: TBD

A HEAD LOSS PRESSURE BOUNDARY CONDITION FOR HYDRAULIC SYSTEMS

JONATHAN FAHLBECK*, HÅKAN NILSSON AND SAEED SALEHI

Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg SE-412 96, Sweden

Email address: fahlbeck@chalmers.se

DOI: TBD
Version(s): OpenFOAM® v1912, v2006, v2012, v2106, v2112
Repo: –

Abstract. Despite the increase in computational power of HPC clusters, it is in most cases not possible to include the entire hydraulic system when doing detailed numerical studies of the flow in one of the components in the system. The numerical models are still most often constrained to a small part of the system and the boundary conditions may in many cases be difficult to specify. The `headLossPressure` boundary condition is developed in the present work for the OpenFOAM open-source CFD code to include the main effects caused by a large hydraulic system onto a component in the system. The main motivation is to provide a boundary condition for incompressible hydraulic systems where known properties are specified by the user and unknown properties are calculated. This paper is a guide to the developed `headLossPressure` boundary condition. It is based on the extended Bernoulli equation to calculate the kinematic pressure on the patch. An arbitrary number of minor and friction losses are considered to describe the system in terms of head losses. The boundary condition also provides the opportunity to specify the head (difference in height) in relation to a reference elevation. System changes during operations are modelled through `Function1` variables, which enables time-varying inputs. The developments are validated against experimental test data, where the varying head between two free surfaces and a valve closing and opening sequence are modelled with the boundary condition. The main effects of the system are well captured by the `headLossPressure` boundary condition. It is thus a useful and trustworthy boundary condition for incompressible flow simulations of components in a hydraulic system.

1. Introduction

Computational Fluid Dynamics (CFD) is inherently computationally demanding and requires large computer resources [1]. The demands of the numerical simulations increase drastically with the size of the computational domain. Therefore, often the computational domains are as much as possible limited to the region of the studied component. In hydraulic system simulations, the dynamics of the system that the studied component is connected to is often neglected, and approximate (unknown) static boundary conditions are in most cases applied for convenience. This puts great limitations on the usefulness and accuracy of the results. Simplified 1D methods, such as the Method of Characteristics (MOC) [2] or Method of Implicit (MOI) [3], are sometimes coupled to full 2D/3D CFD simulations to include the effects of the system. This requires access to a MOC/MOI solver and a coupling interface between the CFD solver and the MOC/MOI solver. With MOC/MOI, the pressure waves in the system are included, which in the case of hydraulic systems requires the pressure waves to be resolved also by the CFD solver. This puts enormous constraints on the time step due to the high sound speed. The aim of the present work is to provide a boundary condition for incompressible 1D hydraulic systems that is implemented using OpenFOAM libraries and can be linked to OpenFOAM using the standard linking procedures of OpenFOAM. Built on an incompressible assumption, it allows larger time steps compared to an compressible formulation, since no pressure waves need to be resolved. The present work thus provides a cost-efficient method to model the main effects from 1D incompressible hydraulic systems with one single pressure boundary condition in OpenFOAM. The motivation behind the work is the need to simulate a component in detail where both the pressure difference between the inlet and outlet of the computational domain and the flow rate are highly dependent on the system. Hence, the flow

* Corresponding author
Received: 26 October 2021, Accepted: 10 January 2022, Published: 20 January 2022

rate and pressure difference are both calculated with the developed boundary condition as a part of the CFD solution. This is accomplished by a boundary condition that considers the characteristics of an arbitrary number of components of the system which the studied component is connected to. The use of time-varying variables, i.e. `Function1`, allows the boundary condition to be utilised for transients, where some of the system characteristics change over time. The boundary condition is implemented as a subclass to the `fixedValue` type. It is developed based on the `totalPressure` boundary condition and is a further development of the initial work made by Fahlbeck [4]. The implementation is validated against measured data for an experimentally studied test case. In addition to the description and validation of the new boundary condition, the paper also provides a general description on how to incorporate head losses at the patches of the computational domain in OpenFOAM. The provided library can be employed by anyone who would like to model a specific component of a large hydraulic system while considering the effect of other components. The implementation was originally developed for OpenFOAM v2012. It has however been tested for several recent OpenFOAM versions (v1912–v2112) and for a number of incompressible solvers, including `simpleFoam`, `pisoFoam`, `pimpleFoam` and `interFoam` (only one phase at the patch is allowed).

2. Theoretical background

The essence of the developed `headLossPressure` pressure boundary condition is explained in this section through mathematical expression and theoretical assumptions. We start by considering Bernoulli's energy principle [5]. It states that the energy of the flow is constant along a streamline (if no losses are considered) as

$$p^* + \frac{u^2}{2} + gz = \text{constant}. \tag{1}$$

Here p^* is the kinematic pressure ($p^* = p/\rho$, where ρ is the fluid density), u is the velocity, g is the gravitational acceleration, and z is the position in the direction of the gravitational acceleration with respect to some reference level. The energy equation can be further extended to include the head losses along a streamline between an upstream (subscript u) and downstream (subscript d) point as

$$p^*_\mathrm{u} + \frac{u^2_\mathrm{u}}{2} + gz_\mathrm{u} = p^*_\mathrm{d} + \frac{u^2_\mathrm{d}}{2} + gz_\mathrm{d} + \Delta p^*_\mathrm{m} + \Delta p^*_\mathrm{f}. \tag{2}$$

Here Δp^*_m and Δp^*_f represent head losses due to local occurrences in the flow path (minor) and friction from the wall (major), respectively.

Consider the simple hydraulic system shown in Figure 1. The simulated computational domain is located between points 2 and 3, and the `headLossPressure` boundary condition updates and adjusts the pressure at patches 2 and 3 by applying the extended Bernoulli equation between points 1–2 and 4–3, respectively, as

$$p^*_2 = p^*_1 + \frac{1}{2}\left(u_1^2 - u_2^2\right) + g\underbrace{H_{12}}_{z_1 - z_2} - \left(\Delta p^*_\mathrm{m} + \Delta p^*_\mathrm{f}\right), \tag{3}$$

$$p^*_3 = p^*_4 + \frac{1}{2}\left(u_4^2 - u_3^2\right) + g\underbrace{H_{34}}_{z_4 - z_3} + \left(\Delta p^*_\mathrm{m} + \Delta p^*_\mathrm{f}\right). \tag{4}$$

If the reservoirs at 1 and 4 are large, it is assumed that the flow velocity is zero at the free surface, which is a common assumption. The extended energy equation, Eq. (2), for the patches at 2 and 3 can thus be expressed as

$$p^*_2 = p^*_1 - \frac{u_2^2}{2} + gH_{12} - \left(\Delta p^*_\mathrm{m} + \Delta p^*_\mathrm{f}\right), \tag{5}$$

$$p^*_3 = p^*_4 - \frac{u_3^2}{2} + gH_{34} + \left(\Delta p^*_\mathrm{m} + \Delta p^*_\mathrm{f}\right). \tag{6}$$

The head losses are subtracted at p^*_2 since it is downstream of p^*_1, and they are added at p^*_3 since p^*_4 is upstream of p^*_3, which is in accordance with Eq. (2). It should be noted that the sign of the loss terms can be made dependent on the flow direction at the boundaries, making the implementation independent of the flow direction. There may for instance be a pump in the CFD region, pumping the water from the lower reservoir to the upper reservoir. This is taken care of in the present implementation.

Local head loss occurrences in the flow path are commonly referred to as minor head losses, and friction losses from the wall as major head losses. In this work the terms *minor* and *friction* are used for those

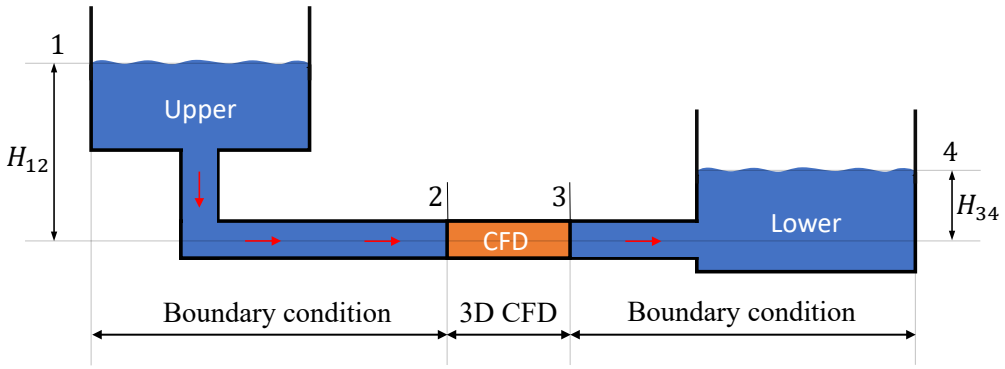


FIGURE 1. Example of a hydraulic system with upper and lower reservoirs, numerical (CFD) domain, and pipelines. The flow is here assumed to be from the upper to the lower reservoir as indicated by the arrows (could be reversed if there is a pump in the CFD region), and the markers 1–4 are in accordance with Eqs. (3)–(6).

head losses. Minor head losses are caused by sudden changes in the flow path or separation, for instance bends, valves, cross-sectional area changes, etc. The minor losses depend on the dynamic pressure and a minor loss coefficient, k. An arbitrary number of minor head losses can be added as

$$\Delta p^*_{\mathrm{m}} = \sum_{i=1}^{\mathrm{N_m}} k_i \frac{u_i^2}{2}. \tag{7}$$

Here $\mathrm{N_m}$ is the number of minor losses. The minor loss coefficient of each component, k_i, is usually found as tabulated values in textbooks, provided by the manufacturer, calculated with numerical models, or estimated based on a set of experimental data.

The head loss due to friction in an arbitrary number of pipes can be summed up as

$$\Delta p^*_{\mathrm{f}} = \sum_{i=1}^{\mathrm{N_f}} f_i \frac{L_i}{d_{\mathrm{h},i}} \frac{u_i^2}{2}, \tag{8}$$

where L_i and $d_{\mathrm{h},i}$ are the length and hydraulic diameter of each pipe, respectively. f_i is the friction loss coefficient of each pipe, and $\mathrm{N_f}$ is the number of friction losses. In a laminar case, f is calculated as a function of the Reynolds number, as

$$f = \frac{64}{\mathrm{Re}}, \tag{9}$$

where $\mathrm{Re} = u d_{\mathrm{h}}/\nu$, in which ν is the kinematic viscosity. However, if the flow is turbulent ($\mathrm{Re} > 2300$), f is solved iteratively using the implicit Colebrook equation [5],

$$\frac{1}{f^{1/2}} = -2.0 \log_{10}\left(\frac{\epsilon/d_{\mathrm{h}}}{3.7} + \frac{2.51}{\mathrm{Re} f^{1/2}}\right), \tag{10}$$

where ϵ is the surface roughness. The explicit Haaland equation [5], given by

$$f^{1/2} \approx \frac{1}{-1.8 \log_{10}\left[\left(\frac{\epsilon/d_{\mathrm{h}}}{3.7}\right)^{1.11} + \frac{6.9}{\mathrm{Re}}\right]}, \tag{11}$$

can be used as an initial guess before solving the more complex Colebrook equation (Eq. (10)).

Hence, a minor head loss is only a function of the local flow velocity and the minor loss coefficient, whereas a friction head loss is a function of the local flow velocity, the pipe length and diameter, and the surface roughness. If all the losses are arranged sequentially (without bifurcated flow paths), as is presently assumed, the local flow velocity can be calculated from the flow rate and the local cross-sectional area.

3. Implementation of the boundary condition

To reach the goal with the `headLossPressure` boundary condition, one must be able to account for any number of head losses in the system. In section 4.3 an example of user-inputs is given. To enable the user to specify an arbitrary number of minor and friction loss factors, the user-inputs `minorLossFactors` and `frictionLossFactors` are declared according to Listing 1. Each of the two variables is constructed as a `List` of `Tuple2` type, where `Tuple2` has the capability of storing two objects. For the minor losses, the `Tuple2` consists of a `vector2D` and a `word`. This ensures that the user can supply a list of the necessary information in the appropriate format (hydraulic diameter, minor loss coefficient), and a name of the particular loss. A similar construction is made for the friction loss factors, the difference being that the `Tuple2` consists of a `vector` and a `word`. This enables the user to supply the list of friction loss factors in the following format (hydraulic diameter, surface roughness, pipe length), and the name of the friction loss. At least one minor and friction loss factor must always be specified for the boundary condition to work. Specifying a minor loss coefficient of zero will disable the calculation of the minor loss. For the friction loss factor, the pipe length must be set to zero to disable all effects due to friction. The name of each loss is used by `Info` and `Warning` statements during run time, and it helps the user to keep track of the individual losses.

```
//- Tuple2 list of the minor loss factors ((d, k) name)
typedef Tuple2<vector2D, word> indexedVector2D;
const List<indexedVector2D> minorLossFactors_;

//- Tuple2 list of the friction loss factors ((d, epsilon, L) name)
typedef Tuple2<vector, word> indexedVector;
const List<indexedVector> frictionLossFactors_;
```

Listing 1. Declaration of the minor and friction loss factors

There are additional user-inputs that are required (and some that are optional) when applying the boundary condition, and in Table 1 the full list of available inputs is shown. The variable `pFar` corresponds to the kinematic pressure 'far' from the patch in the hydraulic system, e.g. up or downstream in the pipe or at a free surface. The variable `HFar` is used to calculate the hydrostatic pressure, gH, which typically is a free surface. Both `pFar` and `HFar` contribute with a constant pressure on the patch. If a *gravity* based solver (e.g `interFoam`) is used, the `HFar` at the patch must be in reference to the head specified in the `hRef` dictionary. In the case of a *non-gravity* based solver (e.g. `simpleFoam` or `pimpleFoam`) the head must be in reference to the system. Hence, if the `headLossPressure` is used at multiple patches, the pressure that drives the flow will be correct as long as the same reference level is used. However, the hydrostatic pressure on a specific patch may be incorrect if the computational domain has an extension in the direction of the gravity acceleration. The same is true if the `headLossPressure` is used only on one patch, the head specified for this single patch must be the head of the system to get the correct flow rate of the system.

The three variables `dP`, `minorLossFactors`, and `frictionLossFactors` are used to calculate the head losses. The hydraulic diameter of the patch, `dP`, and the hydraulic diameter specified for each head loss is used together with the flow rate at the patch to calculate the velocity that is needed to calculate each head loss. The entries `minorLossFactors` and `frictionLossFactors` are for the minor and friction loss factors, respectively, where the details are explained in the Table 1 footnotes.

The special variable `kDynamic` is a single *dynamic* minor loss coefficient, k, with `Function1` capabilities, since the list of minor loss coefficients in `minorLossFactors` cannot use `Function1`. This is useful when, for instance, modelling a valve closing or opening sequence with the boundary condition. Due to the fact that the valve may be located in a pipe with a different cross-sectional area than the patch, the variable `dkDynamic` must also be supplied to re-scale to an appropriate flow velocity for the loss.

The boundary condition can also model the filling or emptying of a reservoir using the parameters `flowRate` and `Ar`, as

$$H_{\mathrm{Far}} = H_{\mathrm{o}} + \frac{Q_{\mathrm{r}} - Q_{\mathrm{p}}}{A_{\mathrm{r}}}\Delta t. \tag{12}$$

Here H_{Far} is the head of the reservoir at the new time step, H_{o} is the head of the reservoir at the previous time step, Q_{r} is the user-supplied `flowRate` to the reservoir, Q_{p} is the flow rate out of the reservoir (calculated from the volumetric flux at the patch), and A_{r} is the user-supplied surface area of the reservoir, `Ar`. The calculation of a new H_{Far} is only made if the user supplies the `flowRate` and `Ar`.

TABLE 1. Available user-inputs for the `headLossPressure` boundary condition.

Variable	Description	Required	Dimension	Default	`Function1`
`pFar`	Kinematic pressure far from patch, p^*_{Far}	yes	m^2/s^2	-	no
`HFar`	Elevation far from patch , H_{Far}	yes	m	-	no
`dP`	Hydraulic diameter of patch, $d_{\text{h,p}}$	yes	m	-	no
`minorLossFactors`	Minor loss factors: ($d^\dagger_{\text{h,l}}$, $k^{\dagger\dagger}$), name‡‡	yes	(m,-), -	-	no
`frictionLossFactors`	Friction loss factors: ($d^\dagger_{\text{h,l}}$, $\epsilon^\parallel$, $L^\star$), name‡‡	yes	(m,m,m), -	-	no
`kDynamic`	Dynamic minor loss coefficient, k	no	-	-	yes
`dkDynamic`†	Hydraulic diameter of `kDynamic`, $d^\dagger_{\text{h,l}}$	no	m	-	no
`flowRate`	Flow rate to the reservoir, Q_{r}	no	m^3/s	-	yes
`Ar`‡	Reservoir surface area, A_{r}	no	m^2	-	no
`dFar`§	Hydraulic diameter far from patch, $d_{\text{h,Far}}$	no	m	0	no
`U`	Velocity field name	no	-	U	no
`phi`	Flux field name	no	-	phi	no
`g`¶	Gravitational acceleration	no	kg m/s^2	9.81	no
`Tol`	Tolerance for the Colebrook equation	no	-	10^{-6}	no
`Nitr`	Max iterations for the Colebrook equation	no	-	20	no
`fDiff`	Max difference between `f` and `fInitial`	no	-	0.05	no

† `dkDynamic` is required if `kDynamic` is supplied, ‡ `Ar` is required if `flowRate` is supplied, † hydraulic diameter at the loss, †† minor loss coefficient, ‡‡ name of the loss, $^\parallel$ surface roughness, * pipe length, § default is 0 and this gives that the far velocity is assumed to be zero (e.g. large reservoir), ¶ only used if a *non-gravity* based solver is used (otherwise the `g` dictionary is used)

The head cannot be allowed to vary within a time step, thus H_{Far} is only updated at the first inner-loop if a transient solver (e.g. `pimpleFoam` or `pisoFoam`) is used. Q_{p} is in this case from the final corrector loop at the previous time step.

If the `headLossPressure` boundary condition is used within a system where there are no free surfaces (e.g. a pipe system), the velocity far from the patch cannot be assumed to equal zero. In this case the variable `dFar` is used to calculate the velocity far from the patch as

$$u_{\text{Far}} = \frac{A_{\text{p}}}{A_{\text{Far}}} u_{\text{avg}} \Rightarrow u_{\text{Far}} = \left(\frac{d_{\text{h,p}}}{d_{\text{h,Far}}} \right)^2 u_{\text{avg}}. \tag{13}$$

Here $d_{\text{h,p}}$ is the hydraulic diameter of the patch, `dP`, $d_{\text{h,Far}}$ is the hydraulic diameter far from the patch, `dFar`, and u_{avg} is the average velocity of the patch (calculated from $u_{\text{avg}} = |\phi/A|_{\text{patch}}$, where ϕ is the volumetric flux of the patch). It is thus assumed that the local velocity far from the patch can be scaled with the ratio of hydraulic diameter, which is valid if the same type of cross-section is used. With this manipulation, the flow rate of the system will be given as part of the solution.

3.1. **Calculation of head losses.** The calculation of the sum of minor head losses is governed by Eq. (7), and the code that handles the calculation is shown in Listing 2. At line 2 the minor head loss, Δp^*_{m}, is defined and initiated either based on the present value of the `kDynamic` entry (if available) or equal to zero. Line 9 shows the general calculation of the i^{th} minor head loss as

$$\Delta p^*_{\text{m},i} = \frac{k_i}{2} \left[u_{\text{avg}} \left(\frac{d_{\text{h,p}}}{d_{\text{h,l},i}} \right)^2 \right]^2. \tag{14}$$

Here u_{avg} is the average velocity on the patch. The expression $(d_{\text{h,p}}/d_{\text{h,l}})^2$ is used to re-scale the velocity for the specific minor loss, where $d_{\text{h,p}}$ and $d_{\text{h,l}}$ are the hydraulic diameter of the patch and loss, respectively. It is thus assumed that the local velocity of the minor loss is only dependent on the relation in hydraulic diameter between the patch and the loss in square, $u_{\text{loss}} = (d_{\text{h,p}}/d_{\text{h,l}})^2 u_{\text{avg}}$. Eq. (14) is used for all the user-supplied minor loss factors to sum up the combined head loss due to minor effects.

The head loss due to friction in the pipes is calculated in a similar fashion as the minor losses, however now according to Eq. (8). The main difference here is that the friction loss coefficient, f, needs to be calculated iteratively using Eq. (10) for a turbulent flow case. The process of how f is estimated is shown in Figure 2, and the procedure is repeated for all the individual friction losses. The first step is to check if the flow is laminar or turbulent, where a critical Reynolds number of 2300 is used. If the flow is laminar, f is immediately found by Eq. (9). In the turbulent case, the Haaland equation (Eq. (11)) is used as the initial guess when estimating the friction loss coefficient by iterating the Colebrook equation (Eq. (10)).

```
scalar kDynamic = kDynamic_ ? kDynamic_->value(this->db().time().timeOutputValue())
    : 0;
scalar dpMinor = kDynamic ? kDynamic*sqr(Uavg*sqr(dP_/dkDynamic_))/2 : 0;
scalar dMinor;
scalar k;
forAll(minorLossFactors_, i)
{
    dMinor = minorLossFactors_[i].first().component(0);
    k = minorLossFactors_[i].first().component(1);
    dpMinor += k * sqr(Uavg * sqr(dP_ / dMinor))/2;
}
```

LISTING 2. Calculation of the minor head losses.

If the relation between the estimated value, f, and the previous value, f_o, is above a tolerance `Tol`, then the latest f is used as a new guess for the next iteration. The default value of `Tol` is 10^{-6}. This process is repeated a maximum of `Nitr` times to avoid slowing down the simulation, where the default value of `Nitr` is 20. If $|f - f_\mathrm{o}|/f_\mathrm{o} \leq$ `Tol` never happens, the latest f is passed on to the next step. Usually, a converged f is obtained within the default tolerance after just a few iterations. To ensure that the iteration process has not diverged, the final f is compared to the initial f_i. The solution obtained by the Haaland equation should only differ from that by the Colebrook equation by ± 2 % [5], so if the difference is more than `fDiff` (5 % is the default value) the initial value from the Haaland equation is used and an `Info` statement is printed to the command window.

3.2. **Patch pressure calculation.** Once all the minor and friction head losses are calculated, the kinematic pressure on the patch is updated according to Eq. (3) and (4) for inflow and outflow faces, respectively. Listing 3 shows the piece of code that calculates and updates the pressure for the patch faces. Note that `Up` and `phip` are velocity and volumetric flux per face on the patch, respectively, whereas `Uavg` and `sumPhi_` are integrated velocity and volumetric flux over the patch. The function `neg` returns 1 if the variable is negative, and the function `pos` returns 1 if the variable is positive only, otherwise both return 0. The mathematical expression of Listing 3 is given by

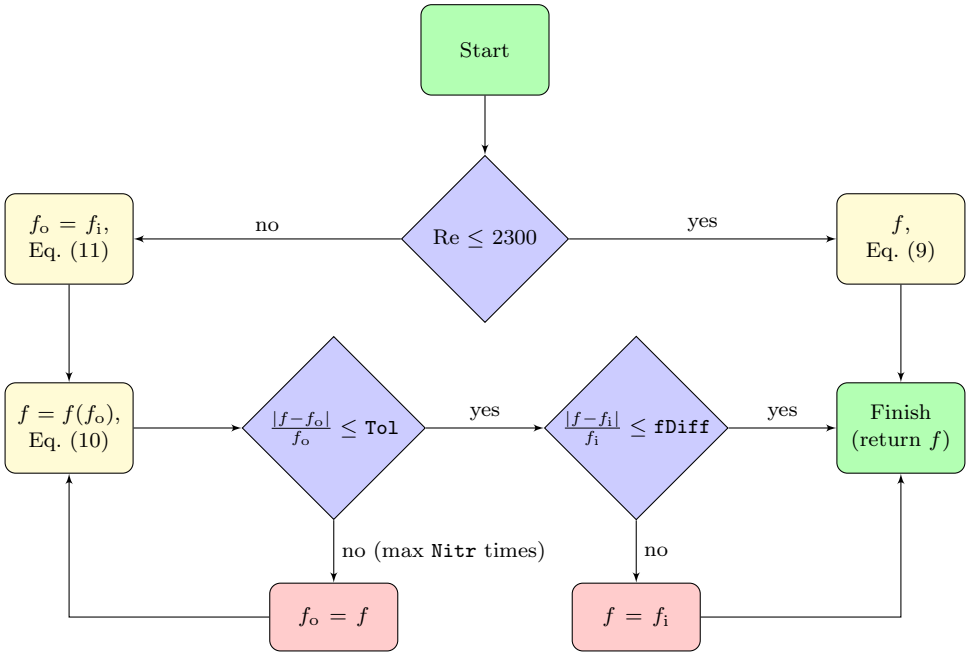


FIGURE 2. Flow chart of iteration procedure to obtain the friction loss coefficient f. Here `Tol` is a tolerance, `Nitr` is the maximum number of iterations for the Colebrook Eq. (10), and `fDiff` is the maximum allowed difference between the Haaland Eq. (11) and Colebrook Eq. (10), index i and o are for initial and old, respectively.

```
operator==
(
    pFar + gHFar + sumPhi_/mag(sumPhi_)*(dpFriction + dpMinor)
    - 0.5*neg(phip)*magSqr(Up) - 0.5*pos(sumPhi_)*magSqr(Uavg)
    + 0.5*UFarScale_*magSqr(Uavg)
);
```

LISTING 3. Patch pressure calculation

$$
\begin{aligned}
p^*_{\mathrm{p},i} = \underbrace{p^*_{\mathrm{Far}}}_{\mathrm{I}} + \underbrace{gH_{\mathrm{Far}}}_{\mathrm{II}} - \underbrace{X_1(\phi_{\mathrm{sum}})\ (\Delta p^*_{\mathrm{f}} + \Delta p^*_{\mathrm{m}})}_{\mathrm{III}} \\
- \underbrace{0.5X_2(\phi_i)\ |u_{\mathrm{p},i}|^2}_{\mathrm{IV}} - \underbrace{0.5X_3(\phi_{\mathrm{sum}})\ |u_{\mathrm{avg}}|^2}_{\mathrm{V}} \\
+ \underbrace{0.5\ u^2_{\mathrm{Far}}}_{\mathrm{VI}},
\end{aligned}
\tag{15}
$$

where the functions X are defined as (note that $\phi_{\mathrm{sum}} \neq 0$)

$$
X_1(\phi_{\mathrm{sum}}) = \begin{cases} -1 \text{ if } \phi_{\mathrm{sum}} < 0 \\ 1 \text{ if } \phi_{\mathrm{sum}} > 0 \end{cases}, \quad X_2(\phi_i) = \begin{cases} 1 \text{ if } \phi_i < 0 \\ 0 \text{ if } \phi_i \geq 0 \end{cases}, \quad X_3(\phi_{\mathrm{sum}}) = \begin{cases} 0 \text{ if } \phi_{\mathrm{sum}} < 0 \\ 1 \text{ if } \phi_{\mathrm{sum}} > 0 \end{cases}. \tag{16}
$$

Here ϕ is the volumetric face flux of the boundary patch, subscript *sum* represents the summation of all patch faces and index i corresponds to face i on the patch. Terms I and II denote the kinematic pressure and hydrostatic pressure far from the patch, respectively. To model the head-rise of a free surface in a reservoir or tank, or to find the head of the system, H_{Far} is updated once per time step/iteration with Eq. (12) (if `flowRate` and `Ar` are supplied in the `p` dictionary for the patch). Term III is where the head losses are added or subtracted according to Eq. (2). They are subtracted for an inflow patch since the losses are located upstream of the patch. For an outflow patch, the losses are added to the patch since the losses are downstream of the patch. Terms IV and V give the dynamic pressure contribution. For inflow faces (negative ϕ_i) term IV is active, and for an outflow patch (positive ϕ_{sum}) term V is switched on. Thus, the boundary condition is set on a face-by-face basis so that the kinematic pressure on the patch varies with the velocity for inflow faces with term IV. This is to ensure the preservation of continuity in combination with a proper velocity boundary condition, and it is the same procedure as is used in the `totalPressure` boundary condition. If there are only outflow faces on the patch, term IV is deactivated on the patch and a uniform dynamic pressure is obtained with term V. This means that for a patch with mixed flow (both inflow and outflow faces) the pressure will be nonphysically represented with this boundary condition. To alert the user if reversed flow is encountered on the patch, a warning message is printed in the command window. The error produced by the boundary condition will increase if adverse pressure gradients are expected on parts of the patch. However, if the flow direction is well defined, a mixed flow on the patch may indicate that the flow velocity is small or close to zero. If that is the case, the error produced by the boundary condition will also be small. This is because the average velocity, u_{avg}, is the summation of the volumetric face fluxes divided by the patch area, hence a small volumetric flow equals a small error. Term VI is the velocity far from the patch and it is calculated from Eq. (13). The variable `UFarScale_` in Listing 3 is the relation in hydraulic diameter of the patch, $d_{\mathrm{h,p}}$, and far, $d_{\mathrm{h,Far}}$, to the power of four, $(d_{\mathrm{h,p}}/d_{\mathrm{h,Far}})^4$. As mentioned in Table 1, if the variable `dFar` is not specified in the boundary condition dictionary, or provided as zero, the variable `UFarScale_` is set to zero. In that case the patch pressure calculation is given by Eqs. (5) and (6), for an inlet and outlet patch, respectively. However, if `dFar` is specified, the patch pressure calculation is according to Eq. (3) for inflow and Eq. (4) for outflow.

4. Validation Case

The implementation and functionality of the developed `headLossPressure` boundary condition is validated with experimental test data reported by Petit et al. [6] and results from numerical simulations by Wang et al. [7].

4.1. **Experimental rig and computational domain.** The experimental test rig is illustrated in Figure 3. The flow enters the upstream tank ② at a steady discharge of 50 l/s. With a partly closed upstream valve ③ and a fully open downstream valve ⑤, the free surfaces of the upstream and downstream tanks stabilise at roughly 3 m and 0.5 m above the centreline of the pipe ④, respectively. The flow in the system is thus driven by the head difference between the upstream and downstream tanks, and counteracted by the losses in the components of the system. A transient sequence was studied experimentally, where the downstream valve ⑤ closed and opened. The pipe ④ had a length of 10 m and a rectangular cross-section of 0.2 by 0.25 m (width×height). The downstream valve ⑤ and a pressure sensor were placed at 2 m and 3.245 m upstream the downstream tank ⑥, respectively.

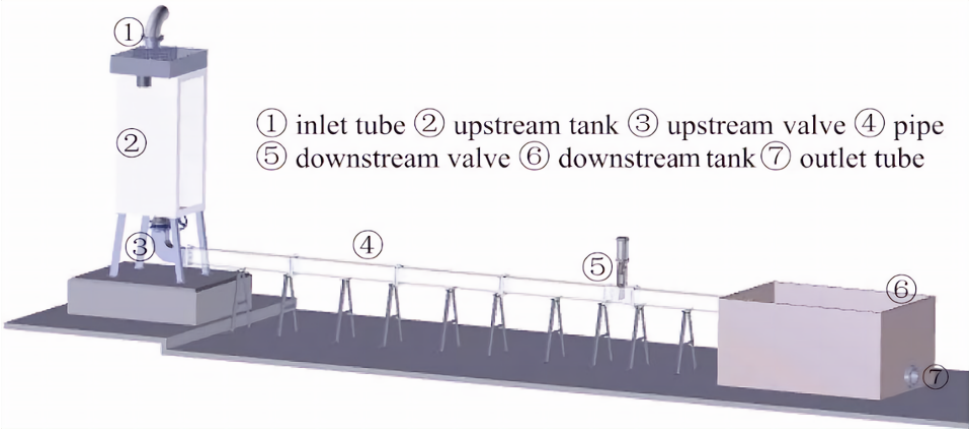


FIGURE 3. Illustration of the experimental test rig [6]. The flow is driven by gravity from the upstream tank ② to the downstream tank ⑥, while being counteracted by the losses in the components of the system. The downstream valve ⑤ is closed and opened to give a transient sequence.

To validate the `headLossPressure` boundary condition a small 1D computational domain is placed in the rectangular pipe between markers ④ and ⑤ in Figure 3. The computational domain is shown in Figure 4. Being a 1D computational domain, it has no losses and does therefore not contribute to the system balance. The entire system is thus represented in the `headLossPressure` boundary conditions at each side of the computational domain, which is the subject of the present validation. The boundary condition is fully capable of being used for a 3D computational domain, and in that case the domain is part of the system balance. The test rig was originally designed to validate simulations of water hammer and pressure oscillations in the system. The current development is restricted to incompressible flow, which limits the results to the main variations in the system and excludes the pressure waves.

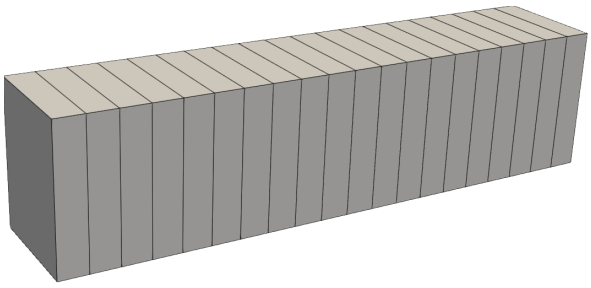


FIGURE 4. 1D computational domain. The flow is from the left to right patch, and all sides are treated with the `empty` boundary condition.

4.2. **Operational sequence.** The opening of the downstream valve (⑤ in Figure 3) in the experiment varied during a transient sequence as shown in Figure 5. The valve was fully open for the first five seconds. Then, the valve closed linearly between 5–10 s, and it was remained fully closed for eight seconds. The valve was then opened linearly between 18–23 s.

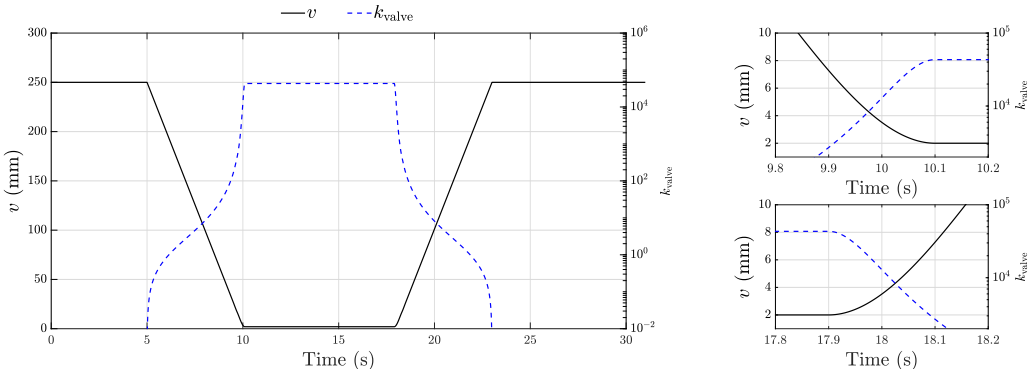


FIGURE 5. Valve opening v (values at left axis) and loss coefficient k_{valve} (values at right axis) as a function of time. Full operational sequence (left plot) and zoomed views of the smoothed parts of the operational sequence at 10 s (small upper right plot) and at 18 s (small lower right plot). The valve is fully open at 250 mm, and closed at 0 mm.

To model the closing and opening of the valve, the `Function1` entry `kDynamic` of the `headLossPressure` boundary condition at the outlet patch of the computational domain is used to prescribe the loss coefficient of the valve at different opening positions. According to the work by Wang et al. [7], the minor loss coefficient of the valve is estimated as a function of the opening as

$$k_{\text{valve}}(v) = \exp(-2.1469 \ln(v) + 12.1624) - 1.3614, \tag{17}$$

where v is in millimetres (250 mm is fully open, and 0 mm is closed). The subtraction of 1.3614 ensures that the expression is zero as the valve is fully open. In the numerical simulation it is not possible to close the valve entirely since the k_{valve} function in Eq. (17) tends to infinity. Thus, a minimum valve opening of 2 mm is used in the simulation. This gives the largest value of the dynamic minor loss coefficient, $k_{\text{valve}}(2) \approx 4.3 \times 10^4$, as shown by the k_{valve} in Figure 5. The valve opening curve in Figure 5 (left) has a smooth transition at times 10 s (small upper right plot) and at 18 s (small lower right plot) for numerical stability, which yields a corresponding smooth transition for the minor loss variation.

The discharge into the upstream tank (② in Figure 3) remained constant at 50 l/s during the entire sequence. Thus, the free surface in the upstream tank was rising during the time that the downstream valve was not fully open.

4.3. **Numerical set-up.** The `pimpleFoam` solver is used for the numerical simulation in the 1D computational domain shown in Figure 4. The mesh is constructed using the `blockMesh` utility and contains 20 cells in the streamwise direction. No turbulence model is used in the simulations. The `linear` scheme is used for gradients and convection terms for all variables. The `backward` scheme is used for temporal discretisation, with a fixed time step of 10^{-3} s. The `headLossPressure` boundary condition is applied at both the inlet and outlet patches, where the system components at each side are modelled in the boundary condition. The results are thus in this case entirely dependent on the developed boundary condition.

The inputs to the `headLossPressure` boundary condition at the inlet and outlet patches are shown in Listings 4 and 5, respectively. The variable `pFar` is given as `uniform 0` at both the inlet and the outlet. This is because the atmospheric pressure is the same at the free surfaces of the upstream and downstream tanks, and hence just a reference pressure. The test rig shown in Figure 3 indicates no water levels. However, the generic system shown in Figure 1 has a similar layout as the experimental test rig. In Figure 1, positions 1 and 4 can be thought of as the 'far' locations for the inlet and outlet, respectively.

For the inlet, Listing 4, the head `HFar` has a value of 3 m, as this is the head of the upstream tank, ② in Figure 3, when the system initially is in balance. A `flowRate` of 50×10^{-3} m^3/s and a free surface area `Ar` of 1.27 m^2 is specified. This allows the free surface of the upstream tank to vary when the valve is not fully open. The hydraulic diameter of the patch, `dP`, has the value 0.222 m and it is the same everywhere in the system, which is why this value is also seen as the first entry of all the loss factors. The `frictionLossFactors` at the inlet has a hydraulic diameter of 0.222 m and an estimated surface roughness of 0.0025 mm since the pipe is made out of plexiglass. Because of that the computational domain is 1D, no wall friction is included in the computational domain itself. Instead, the full length of the pipe is modelled with the boundary condition. The inlet of the computational domain is positioned at the location of the pressure probe from the lab (3.245 m upstream of the downstream tank), such

that the time-varying value of the pressure at that patch can be used for the validation. Thus, the `frictionLossFactors` of the inlet patch utilises a pipe length of 6.755 m, which is the remainder of the full length of the 10 m pipe. There are two minor loss factors, `minorLossFactors`, supplied at the inlet, both with the hydraulic diameter of 0.222 m. The first entry, `expData`, has a minor loss coefficient of 45.9 and includes the bend and the upstream valve, ③ in Figure 3. It was reported from the lab that the upstream valve was partly closed, hence the large value of the `expData` minor loss coefficient. In the work by Petit et al. [6] a combined minor loss coefficient of 49 was reported for the full system. The minor loss coefficient of 45.9 for the `expData` is 49 subtracted by the estimated sum of all the other losses combined (minor and friction). This is to show that the implementation of the `frictionLossFactors` and `minorLossFactors` work as intended. The second entry, `entrance`, is due to entrance effects from the upstream tank to the pipe. A value of 0.45 is assigned in accordance with White [5]. It is assumed that the friction losses in the upstream tank are negligible, since the velocity in the tank is comparatively very small.

For the outlet, Listing 5, the same atmospheric pressure `pFar` is used as for the inlet. The head of the downstream tank, ⑥ in Figure 3, was at the experimental tests 0.5 m, and the same value is thus given as the head `HFar` in the numerical simulation. As mentioned, the same hydraulic diameter `dP` is used everywhere. The dynamic minor loss coefficient `kDynamic` is specified with a table file according to the time-varying losses in the valve closing and opening, as explained in Section 4.2. The entry `dkDynamic` is the hydraulic diameter of the dynamic minor loss coefficient. The `frictionLossFactors` has a third entry of 3.245 m, which indicates that the outlet patch is located 3.245 m upstream of the downstream tank. Thus, from a hydraulic loss perspective, the inlet and outlet of the computational domain are located at the same position. This is in accordance with the fact that the 1D computational domain introduced no losses, and that all the losses of the 10 m pipe are given by the boundary conditions. The hydraulic diameter and surface roughness of the pipe downstream the outlet are the same as for the inlet (first and second entry, respectively). The final entry is the `minorLossFactors`, where only one minor loss factor is specified and it is due to exit effects, from the pipe to the downstream tank. The value of the `exit` is according to White [5]. The friction losses in the downstream tank are assumed negligible.

Note that the variable `dFar` is specified at neither the inlet nor the outlet. By not specifying this, it is assumed that 'far' is located at the surface of a large tank/reservoir, where the velocity is small compared to that in the computational domain.

For the velocity boundary condition, the `pressureInletOutletVelocity` is used at both the inlet and the outlet. All sides use the `empty` boundary condition.

```
inlet
{
    type            headLossPressure;
    pFar            uniform 0; // m2/s2
    HFar            3;         // m
    flowRate        50e-3;     // m3/s
    Ar              1.27;      // m2
    dP              0.222;     // m
    frictionLossFactors
    (
        ((0.222 0.0025e-3 6.755) pipe)
    ); // (m, m, m)
    minorLossFactor
    (
        ((0.222 45.9) expData)
        ((0.222 0.45) entrance)
    ); // (m, -)
}
```

Listing 4. Inlet patch pressure boundary condition

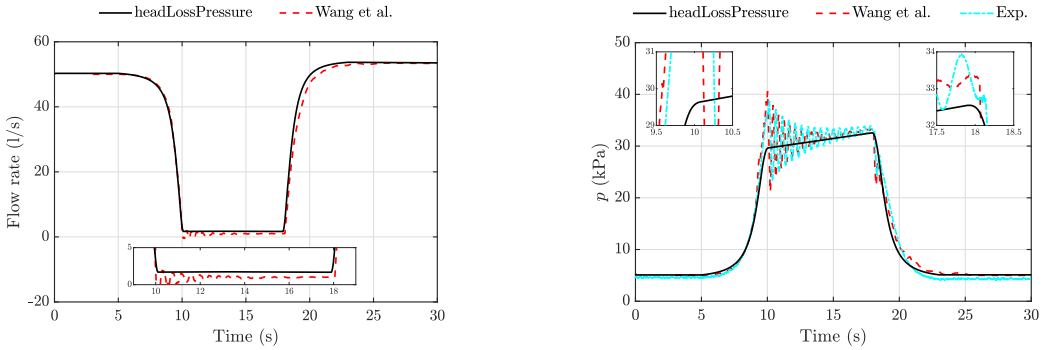
```
outlet
{
    type            headLossPressure;
    pFar            uniform 0; // m2/s2
    HFar            0.5;       // m
    dP              0.222;     // m
    kDynamic        tableFile;
    kDynamicCoeffs
    {
        file    "$FOAM_CASE/constant/kValve";
    } // (s, -)
    dkDynamic       0.222;     // m
    frictionLossFactors
    (
        ((0.222 0.0025e-3 3.245) pipe)
    ); // (m, m, m)
    minorLossFactors
    (
        ((0.222 1) exit)
    ); // (m, -)
}
```

Listing 5. Outlet patch pressure boundary condition

5. Results and discussion

Figure 6 shows the presently computed results compared to the numerical work done by Wang et al. [7] and the experimental test data (only for pressure) [6]. The flow rate is in this work extracted from the

volumetric face flux, `phi`, at the inlet patch, and the pressure probe is located at the centre of the inlet patch to match the location of the experiment. Neglecting the oscillations that are due to compressibility effects (not included in the present work), the main variations for both flow rate and static pressure are greatly captured by the boundary condition. However, a small offset is noted as the valve is fully closed, as shown in the zoomed view of Figure 6a. This is because the valve could not be fully closed with the present boundary condition since it is controlled with a loss coefficient and not with a physical valve. As the valve is closing (5–10 s), the flow rate decreases, and the pressure increases. When the valve is fully closed (10–18 s), the flow rate is constant at a small value while the pressure continues to increase linearly. This is due to the fact that the free surface of the upstream tank is rising when the flow through the pipe is lower than the flow entering the upstream tank. During the valve opening (18–23 s), the flow rate increases rapidly and the pressure decreases. The static pressure reaches its initial value fast, while the flow rate requires a longer time to develop, roughly 300 s (not shown here). All these features are captured adequately by the developed boundary condition. As previously mentioned, the current implementation cannot capture the oscillations in pressure when the valve is fully closed. This is because the boundary condition is developed and used for purely incompressible flows. The work done by Wang et al. [7] concerned water hammer effects, i.e. pressure pulsations in the system, and thus employed appropriate solver and boundary conditions for such simulations.



(A) Flow rate (B) Pressure probe

FIGURE 6. Flow rate and static pressure during the transient sequence. Wang et al. [7] is numerical results with MOC and Exp. is experimental results reported by Petit et al. [6].

As mentioned in Section 4.2, the valve opening and closing sequences are smoothed when the valve opening is near the fully closed position ($v = 2$ mm). Without the smoothing, the pressure curve of Figure 6b shows nonphysical oscillating behaviour at 10 s and 18 s. However, with the smoothing of the operational sequence, the boundary condition gives a smooth pressure variation, as shown by the zoomed views of Figure 6b.

The computational domain can easily be extended to 2D or 3D, with resolved boundary layers. Such results are not shown here, but they present equally convincing results as the 1D results. It can thus be concluded that the `headLossPressure` pressure boundary condition can be used to predict a trustworthy flow rate and pressure at the boundary patches of a computational domain that is connected to a system of sequential heads and head losses. The feasibility of the developed boundary condition when the flow field is non-uniform at an inlet patch, or if reverse/mixed flow is encountered at boundary patches needs further studies.

6. CONCLUSIONS

This paper has described and validated a newly developed pressure boundary condition for the open-source CFD toolbox OpenFOAM, named `headLossPressure`. The boundary condition is an extended version of the available `totalPressure` boundary condition. It adds the possibility to set flow-driving pressure differences and free surface elevations, and an arbitrary number of hydraulic losses that modify the pressure at the patches that the boundary condition is used on. This provides the possibility to model a hydraulic system to which the numerically analysed component is in reality part of. One of the main motivations behind the implementation is to be able to set boundary conditions for simulations of hydraulic systems with properties that are known. An example is the head between two free water surfaces (which can vary in time), which is subjected to flow-dependent (and time-dependent) losses in the system in which the studied component is located. The total losses are given by the sum of minor

losses (local in the flow path) and friction losses (occurring in pipes), and the user can supply any number of losses that are in series in a system. The boundary condition is validated against experimental data, and the numerical predictions match the relevant features of the experimental to a great extent. The top features with the boundary condition are:

- Estimates the correct flow rate and pressure at any given operating condition, provided that the system is well represented by loss factors.
- Models effects, e.g. pressure rise, from a full system to the computational domain in a cost-efficient manner.
- Models changes in the system, e.g. closing of a valve or head rise via `Function1` variables, or filling/emptying of a reservoir.
- Enables the user to specify the head and/or pressure far from the patch.
- Possible to use for both steady and unsteady computations.
- Compiles directly in OpenFOAM without additional dependencies.

The main limitation with the current development is that it is based on an incompressible assumption, and that it can therefore not capture pressure waves such as water hammer. It is thus mostly useful for studies of hydraulic components in a system with incompressible dynamics, or in incompressible systems where the boundary conditions are unknown at the computational domain boundaries.

Author contributions: Conceptualization, J.F., H.N. and S.S.; methodology, J.F.; software, J.F.; validation, J.F.; formal analysis, J.F.; investigation, J.F.; resources, H.N.; data curation, n/a; writing—original draft preparation, J.F.; writing—review and editing, J.F., H.N. and S.S.; visualization, J.F.; supervision, H.N. and S.S.; project administration, J.F.; funding acquisition, H.N. All authors have read and agreed to the published version of the manuscript. Please turn to the CRediT taxonomy (`https://casrai.org/credit/`, accessed on 11 January 2022) for term explanation.

Acknowledgement: This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 883553. The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at NSC and C3SE, partially funded by the Swedish Research Council through grant agreement No. 2018-05973.

References

[1] T. Kajishima and K. Taira, *Computational Fluid Dynamics: Incompressible Turbulent Flows*, T. Kajishima and K. Taira, Eds. Cham: Springer International Publishing, 2017.

[2] M. Zhao and M. Ghidaoui, "Godunov-type solutions for water hammer flows," *Journal of Hydraulic Engineering*, vol. 130, no. 4, pp. 341–348, 2004.

[3] C. Wang and J.-D. Yang, "Water hammer simulation using explicit-implicit coupling methods," *Journal of Hydraulic Engineering*, vol. 141, no. 4, 2015.

[4] J. Fahlbeck, "Implementation of an incompressible headLossPressure boundary condition," in *Proceedings of CFD with OpenSource Software, 2020, Edited by Nilsson. H.*, 2021, http://dx.doi.org/10.17196/OS_CFD#YEAR_2020.

[5] F. M. White, *Fluid Mechanics*, 8th ed. McGraw-Hill, 2016.

[6] O. Petit, W. Chao, M. Cervantes, J. Yang, and H. Nilsson, "Numerical and experimental investigations of a hydraulic pipe during a gate closure at a high Reynolds number," in *6 th IAHR International Meeting of the Workgroup on Cavitation and Dynamic Problems in Hydraulic Machinery and Systems*, Ljubljana, Slovenia, 2015.

[7] C. Wang, H. Nilsson, J. Yang, and O. Petit, "1D–3D coupling for hydraulic system transient simulations," *Computer Physics Communications*, vol. 210, pp. 1–9, Jan. 2017.