Computers and Mathematics with Applications 59 (2010) 2168–2177

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

Pressure condition for lattice Boltzmann methods on domains with curved boundaries

Zhaoxia Yang

FB Mathematik und Statistik, Universität Konstanz, Postfach D194, 78457 Konstanz, Germany

ARTICLE INFO

Keywords:
Outflow condition
Pressure condition
Curved boundary
Lattice Boltzmann method

ABSTRACT

We propose a lattice Boltzmann algorithm for an average pressure boundary condition at outlets in pipe flow systems. The advantage of this boundary condition is that only the average pressure is used to recover the non-trivial flow fields. The asymptotic analysis shows that this algorithm works for general curved boundaries and renders a second order accurate velocity and a first order accurate pressure approximation of the incompressible Navier–Stokes solution. Here, we verify the accuracy by numerical simulations of a Poiseuille flow and a less symmetric flow with non-trivial pressure field in channels inclined with arbitrary angle, and flows in a pipe with three outlets.

1. Introduction

For incompressible Navier–Stokes flows, the fluid velocity $\boldsymbol{u}$ and pressure p are found from the governing equations (in non-dimensional form)

$$\nabla\cdot\boldsymbol{u}=0,\qquad \partial_t\boldsymbol{u}+(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}+\nabla p=\nu\nabla^2\boldsymbol{u}+\boldsymbol{G},\qquad \boldsymbol{u}|_{t=0}=\boldsymbol{\psi}, \tag{1}$$

where ν is the fluid shear viscosity, $\boldsymbol{\psi}:\Omega\to\mathbb{R}^d$ and $\boldsymbol{G}:[0,T]\times\Omega\to\mathbb{R}^d$ represent a divergence-free initial velocity field and a source term respectively. One of the non-Dirichlet boundary conditions is an outflow condition. Since this kind of boundary condition is not set by nature and the physics just provides some observable guidelines, appropriate outflow conditions are difficult to construct. Based on the experience of successful simulations, Gresho [1,2] proposed a type of open boundary condition

$$-p\boldsymbol{n}+\nu\frac{\partial\boldsymbol{u}}{\partial\boldsymbol{n}}=\boldsymbol{F}, \tag{2}$$

where $\boldsymbol{F}$ is a prescribed physical quantity depending on the flow type which contains also curvature terms in the case of non-planar boundaries. Further, Heywood et al. [3] carried out a systematic numerical investigation of flows around junctions in pipe systems, which need to be truncated at outlets where appropriate artificial boundary conditions must be imposed. They found that $\boldsymbol{F}$ is related to the average pressure, leading to a hydrodynamic pressure drop condition

$$-p\boldsymbol{n}+\nu\frac{\partial\boldsymbol{u}}{\partial\boldsymbol{n}}=-\bar{P}\boldsymbol{n}. \tag{3}$$

Relation (3) is a natural boundary condition in the proper variational form of the Navier–Stokes equation (1), when the finite element method is applied. The numerical experiments in [3] show that the position of the truncation does not significantly influence the inner flow with this type of pressure boundary condition. Typically $\bar{P}$ is a given constant for stationary flows

E-mail address: zhaoxia.yang@uni-konstanz.de.

0898-1221/$ – see front matter
doi:10.1016/j.camwa.2009.08.074

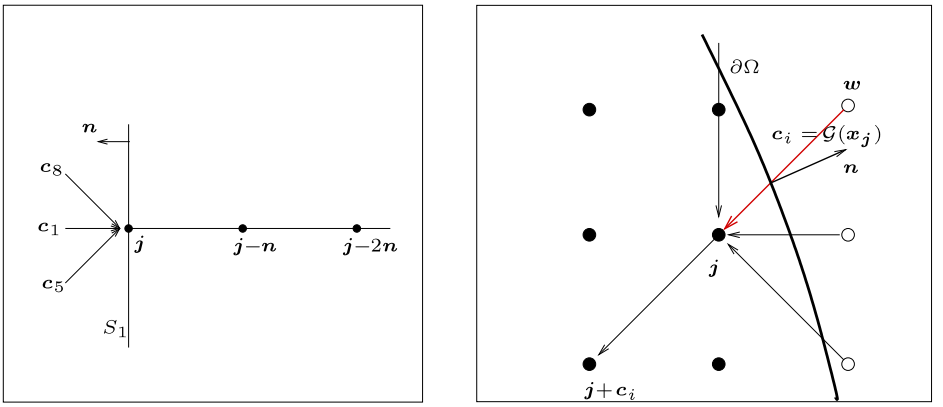

Fig. 1. Left: boundary aligned with grid lines. In the case of the D2Q9 discrete velocity model, there are three incoming directions $\boldsymbol{c}_1$, $\boldsymbol{c}_5$ and $\boldsymbol{c}_8$ at each boundary node on the boundary segment S_1, in particular $c_1 = -\boldsymbol{n}$. Right: general curved boundary $\partial\Omega$. The velocity $\boldsymbol{c}_i$ is incoming at $\boldsymbol{x_j}$ and $\boldsymbol{x_w}$ is the corresponding outer-neighbor node. $\boldsymbol{n}$ is the unit outer normal direction at the intersection point $\tilde{\boldsymbol{x}}_{\boldsymbol{j}i}$. $\mathcal{G}(\boldsymbol{x_j})$ is the closest direction to $-\boldsymbol{n}$ at the boundary segment near $\boldsymbol{x_j}$. The black and hollow balls represent fluid and wall nodes respectively.

and a prescribed function of time for transient flows. Moreover, the mathematical well-posedness of the initial boundary value problem (1), (3) together with the no-slip condition of the bounding pipe walls has been rigorously proved by Galdi [4].

It is worth mentioning that the outflow condition (3) only demands prescription of the average pressure and yields a non-trivial pressure profile on the boundary. Its success in simulations of pipe systems with the finite element method suggests that it is a reasonable outflow condition for the incompressible Navier–Stokes equation. This is our motivation to construct its lattice Boltzmann implementation.

Most previous research concerned with pressure conditions concentrates on Dirichlet conditions where the pressure profile is prescribed along the boundary. One can refer to extrapolation methods proposed by Guo et al. [5,6] and the algorithm PAB (pressure anti-bounce back) suggested by Ginzburg et al. [7]. However, the Dirichlet condition for pressure (see the useful references [5–7]) is not the in the focus of this paper.

In [8], Verhaeghe et al. propose a lattice Boltzmann algorithm by using the average pressure alone as outflow condition in microchannels. There, the unknown lattice Boltzmann distributions are extrapolated and renormalized to ensure that the pressure is attained on average along the boundary while being consistent to the flow field inside the channel.

A lattice Boltzmann realization for the pressure drop outflow condition (3) is also presented in [9]. A key requirement is that the normal direction is one of the discrete velocity directions (for a 2D sketch see the left part of Fig. 1). The velocity at the boundary node $\boldsymbol{x_j}$ is extrapolated using two fluid neighbor nodes $\boldsymbol{x_{j-n}}$ and $\boldsymbol{x_{j-2n}}$, while the density is determined using the boundary condition (3). As a result, the velocity and pressure field are obtained with accuracy order 2 and 1 respectively.

However, both approaches [8,9] require that channels or pipes are parallel to a coordinate axis with perpendicular inlet and outlet in order to guarantee that the boundary is aligned with the lattice grid lines. Along a general curved boundary, the normal direction varies and is no longer one of the particle velocity directions. The ideas in [8,9] thus fail and must be modified. In this paper, we intend to give a lattice Boltzmann algorithm which yields the outflow condition (3) on general curved boundaries. The structure of the paper is as follows. Section 2 introduces the lattice Boltzmann methods which are compatible to the boundary algorithm proposed in Section 3. Section 4 shows three numerical tests. The conclusion and discussion are given in Section 5.

2. Lattice Boltzmann methods

We consider the standard lattice Boltzmann equation,

$$f_i(n+1, \boldsymbol{j}+\boldsymbol{c}_i) = f_i(n, \boldsymbol{j}) + (A(\boldsymbol{f}^{eq} - \boldsymbol{f}))_i(n, \boldsymbol{j}) + g_i(n, \boldsymbol{j}), \tag{4}$$

where $f_i(n, \boldsymbol{j})$ represent the density distributions of the particles which are moving with velocity $\boldsymbol{c}_i$ at time level $t_n = n\triangle t$ and node $\boldsymbol{x_j} = h\boldsymbol{j}$ for a given temporal step Δt and a spatial grid size h with $n \in \mathbb{N}$ and $\boldsymbol{j} \in \mathbb{Z}^d$. The time step is coupled with the grid size by $\Delta t \sim h^2$ (see the detailed discussion in [10,11]). The discrete velocity $\boldsymbol{c}_i$ is taken from the set $\mathbb{V} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N\} \subset \mathbb{R}^d$ which possesses the symmetry property $\mathbb{V} = -\mathbb{V}$ and which is compatible with the spatial lattice $h\mathbb{Z}^d$ in the sense that $\boldsymbol{j} + \boldsymbol{c}_i \in \mathbb{Z}^d$ for every $\boldsymbol{j} \in \mathbb{Z}^d$ and every $\boldsymbol{c}_i \in \mathbb{V}$. The bold face quantities $\boldsymbol{f}$ and $\boldsymbol{f}^{eq}$ are vectors with components f_i and f_i^{eq} respectively.

The equilibrium distribution f_i^{eq} is taken to be

$$f_i^{eq} = F_i(\hat{\rho}, \hat{\boldsymbol{u}}), \qquad F_i(\hat{\rho}, \hat{\boldsymbol{u}}) = f_i^* \left(\hat{\rho} + 3\hat{\boldsymbol{u}} \cdot \boldsymbol{c}_i + \frac{9}{2}(\hat{\boldsymbol{u}} \cdot \boldsymbol{c}_i)^2 - \frac{3}{2}|\hat{\boldsymbol{u}}|^2 \right) \tag{5}$$

which is a function of the total mass density and the average velocity of the particles

$$\hat{\rho} = \sum_{i=1}^{N} f_i, \qquad \hat{\boldsymbol{u}} = \sum_{i=1}^{N} \boldsymbol{c}_i f_i. \tag{6}$$

Here $f_i^* = F_i(1, 0)$ are the standard lattice Boltzmann weights obeying the symmetry property $f_i^* = f_{i^*}^*$ where i^* is the index of the velocity vector $\boldsymbol{c}_{i^*} = -\boldsymbol{c}_i$, and

$$\sum_{i=1}^{N} f_i^* = 1, \qquad \sum_{i=1}^{N} c_{i\alpha} c_{i\beta} f_i^* = \frac{1}{3}\delta_{\alpha\beta},$$
$$\sum_{i=1}^{N} c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} f_i^* = \frac{1}{9}(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma} + \delta_{\alpha\gamma}\delta_{\beta\delta}).$$

Note that the standard D2Q9, D3Q15, D3Q19, and D3Q27 weights fall into this class [12,13].

The collision matrix $A \in \mathbb{R}^{N\times N}$ used in (4) should satisfy the following properties: it should be symmetric and positive semi-definite with kernel generated by $\{\mathbf{1}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_d\}$. Here, the components of the vector $\mathbf{1} \in \mathbb{R}^N$ are all 1 and $\boldsymbol{v}_\alpha \in \mathbb{R}^n$ is a vector with components $(c_{i\alpha})_{i=1,\ldots,N}$. Further we require that for every $\alpha, \beta \in \{1, \ldots, d\}$ the vectors $\boldsymbol{\Lambda}_{\alpha\beta}\boldsymbol{f}^*$ with components

$$(\boldsymbol{\Lambda}_{\alpha\beta}\boldsymbol{f}^*)_i = \left(c_{i\alpha}c_{i\beta} - \frac{1}{d}|\boldsymbol{c}_i|^2\right) f_i^*, \quad i = 1, \ldots, N$$

are eigenvectors of the collision matrix A with eigenvalue $1/(3\mu)$, i.e.

$$A\boldsymbol{\Lambda}_{\alpha\beta}\boldsymbol{f}^* = \frac{1}{3\mu}\boldsymbol{\Lambda}_{\alpha\beta}\boldsymbol{f}^*, \quad \alpha, \beta \in \{1, \ldots, d\}. \tag{7}$$

As shown in [10], this establishes the link to the fluid viscosity ν with the relation $\nu = \mu - 1/6$.

The required conditions on the matrix A are satisfied for many widely used collision models, for example MRT models described in [14]. Also, the well-known BGK model with single relaxation time $\tau = 3\mu$ can be formulated with an appropriate matrix A as shown, for example, in [10,15].

Finally, the function g_i models the body force term in (1) with a scaling proportional to h^3 (a detailed explanation is given in [10])

$$g_i(n, \boldsymbol{j}) = c_s^{-2} h^3 f_i^* \boldsymbol{c}_i \cdot \boldsymbol{G}(t_n, \boldsymbol{x}_{\boldsymbol{j}}).$$

3. Boundary algorithm

In the following, we always assume that $\boldsymbol{x}_{\boldsymbol{j}}$ is some boundary node, i.e. a node in the domain Ω for which at least one neighbor is outside of Ω. The lattice Boltzmann equation (4) does not prescribe how to compute $f_i(n+1, \boldsymbol{j})$ for those indices i which satisfy $\boldsymbol{x}_{\boldsymbol{j}-\boldsymbol{c}_i} \notin \Omega$. The corresponding directions $\boldsymbol{c}_i$ are called incoming directions at $\boldsymbol{x}_{\boldsymbol{j}}$.

In order to specify the required values $f_i(n+1, \boldsymbol{j})$, we use the boundary condition (3) which can also be split into a normal and a tangential component

$$-p + \nu\frac{\partial u_{\boldsymbol{n}}}{\partial \boldsymbol{n}} = -\bar{P}, \tag{8a}$$
$$\frac{\partial u_{\boldsymbol{t}}}{\partial \boldsymbol{n}} = 0, \tag{8b}$$

in which $\boldsymbol{t}$ and $\boldsymbol{n}$ indicate the tangential and outer normal directions and components; both $\boldsymbol{t}$ and $\boldsymbol{n}$ are unit vectors.

The normal component of (3), i.e. Eq. (8a), is a natural candidate to specify the incoming information $f_i(n+1, \boldsymbol{j})$ for the direction $\bar{\boldsymbol{c}}_i$ which is *as normal as possible*. In order to make this notion more precise, we construct a function $\mathcal{G}$ with the property that $\boldsymbol{c}_i = \mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}})$ is as normal as possible in the following sense: at $\boldsymbol{x}_{\boldsymbol{j}}$ we check the intersection points $\bar{\boldsymbol{x}}_{\boldsymbol{j}k}$ of all the links with $\partial\Omega$ and compare $-\boldsymbol{n}(\boldsymbol{x}_{\boldsymbol{j}k})$ and $\bar{\boldsymbol{c}}_k = \boldsymbol{c}_k/\|\boldsymbol{c}_k\|$. The index i for which the least norm difference is encountered wins and we set $\mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}}) = \boldsymbol{c}_i$ (see Appendix A for a more careful definition).

Usually $\mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}})$ is an incoming direction at node $\boldsymbol{x}_{\boldsymbol{j}}$. A rare exception is illustrated in the right plot of Fig. 2. Here, the boundary node has only one incoming direction $\boldsymbol{c}_i$ but $\mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}})$ is equal to $\boldsymbol{c}_n$.

A second direction selection function is required to deal with the tangential component (8a) of (3). Here, our basic idea is to implement the zero slope condition in normal direction by imposing suitably reflected values of the tangential velocity outside the computational domain so that the normal derivative approximately vanishes at the boundary. In particular, the value to be prescribed at some outside point $\boldsymbol{x}_{\boldsymbol{w}} \in \Omega^c$ is computed from values inside the domain in a direction which is as normal as possible. A velocity $\boldsymbol{c}_p$ which satisfies these requirements will be denoted $\boldsymbol{c}_p = \mathcal{P}(\boldsymbol{x}_{\boldsymbol{w}})$. More specifically, at $\boldsymbol{x}_{\boldsymbol{w}}$ we follow the velocity directions which point into the domain and select those for which at least two neighboring nodes

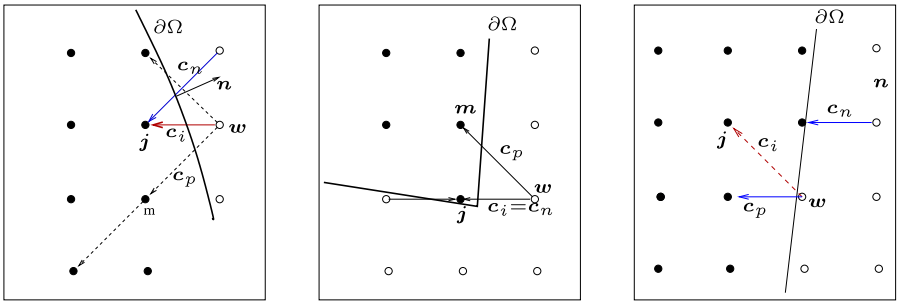

Fig. 2. Illustration of various boundary scenarios for the D2Q9 velocity model. The outer nodes and fluid nodes are represented by ○ and ● respectively. At a boundary node $\boldsymbol{x}_j$, $\boldsymbol{c}_n = \mathcal{G}(\boldsymbol{x}_j)$ denotes a velocity which is as normal as possible and $\boldsymbol{c}_p = \mathcal{P}(\boldsymbol{x}_w)$ is a velocity which is most appropriate for extrapolating the velocity at $\boldsymbol{x}_w$. For an incoming direction $\boldsymbol{c}_i$, the left and right plot illustrate the case $\boldsymbol{c}_i \neq \boldsymbol{c}_n$ while $\boldsymbol{c}_i = \boldsymbol{c}_n$ and $\boldsymbol{c}_p \neq \boldsymbol{c}_n$ is shown in the middle.

in Ω are available. If there are several such directions, we choose the one which is closest to $-\boldsymbol{n}$ at the corresponding link intersection with $\partial\Omega$. The corresponding discrete velocity is denoted $\mathcal{P}(\boldsymbol{x}_w)$. A more formal definition of $\mathcal{P}$ can be found in Appendix B.

Of course, it may happen that no direction is found for which two neighboring points of $\boldsymbol{x}_w$ are available in Ω. In this case, the geometry is either very thin or strongly curved and it is recommended to use a finer grid to better resolve the situation. If this is not feasible, other specifically tailored modifications have to be used. A remedy is also given in Appendix B.

Having introduced the velocity selection functions $\mathcal{P}$ and $\mathcal{G}$, we can now distinguish the relevant cases which can appear at a general boundary point. To this end, we assume that $\boldsymbol{x}_j$ is a boundary node and that $\boldsymbol{c}_i$ is an incoming velocity. The corresponding neighbor outside the domain is denoted $\boldsymbol{x}_w = \boldsymbol{x}_{j-c_i}$. We distinguish the case

$$\boldsymbol{c}_i = \mathcal{G}(\boldsymbol{x}_j) \quad \text{and} \quad \mathcal{G}(\boldsymbol{x}_j) = \mathcal{P}(\boldsymbol{x}_w) \tag{9}$$

and the opposite situation

$$\boldsymbol{c}_i \neq \mathcal{G}(\boldsymbol{x}_j) \quad \text{or} \quad \mathcal{G}(\boldsymbol{x}_j) \neq \mathcal{P}(\boldsymbol{x}_w). \tag{10}$$

3.1. Case: $\boldsymbol{c}_i = \mathcal{G}(\boldsymbol{x}_j)$ and $\mathcal{G}(\boldsymbol{x}_j) = \mathcal{P}(\boldsymbol{x}_w)$

By construction, $\boldsymbol{c}_i$ is closest to the inner normal direction $-\boldsymbol{n}$. A 2D example is illustrated in the right plot of Fig. 1. The particular case that $\bar{\boldsymbol{c}}_i = \boldsymbol{c}_i / \|\boldsymbol{c}_i\|$ is exactly equal to $-\boldsymbol{n}$, has been studied in [9], however, under the additional restriction that the grid nodes are located on the boundary. In the boundary algorithm proposed below, this restriction has been removed.

At the node $\boldsymbol{x}_j$, the incoming direction $\boldsymbol{c}_i$ will be responsible for the realization of Eq. (8a) using the following boundary scheme,

$$f_i(n+1, \boldsymbol{j}) + f_{i^*}(n+1, \boldsymbol{j}) = f_i(n, \boldsymbol{j}) + f_{i^*}(n, \boldsymbol{j}) + 2f_i^*(\tilde{\rho}(n, \boldsymbol{j}) - \hat{\rho}(n, \boldsymbol{j})), \tag{11}$$

in which $f_{i^*}(n+1, \boldsymbol{j})$ is already obtained from the lattice Boltzmann equation (4). The density $\tilde{\rho}(n, \boldsymbol{j})$ represents the estimated density at $\boldsymbol{x}_j$ after imposing the boundary condition (8a). More precisely, following the idea in [9], $\tilde{\rho}(n, \boldsymbol{j})$ is obtained by interpolating between the values at the intersection point $\bar{\boldsymbol{x}}_{ji}$ and the neighbor node $\boldsymbol{x}_{j+c_i}$. This gives rise to

$$\tilde{\rho}(n, \boldsymbol{j}) = \frac{1}{1+q_{ji}} \rho(n, \bar{\boldsymbol{x}}_{ji}) + \frac{q_{ji}}{1+q_{ji}} \hat{\rho}(n, \boldsymbol{x}_{j+c_i}). \tag{12}$$

Here q_{ji} is the scaled distance between $\bar{\boldsymbol{x}}_{ji}$ and the node $\boldsymbol{x}_j$, namely $\boldsymbol{x}_j = \bar{\boldsymbol{x}}_{ji} + hq_{ji}\boldsymbol{c}_i$. Further, the pressure at the intersection point $\bar{\boldsymbol{x}}_{ji}$ is calculated from the boundary condition,

$$\rho(n, \bar{\boldsymbol{x}}_{ji}) = 1 + 3h^2 \left(\bar{P} + \nu(\boldsymbol{n} \cdot \nabla)(\boldsymbol{u} \cdot \boldsymbol{n})(\bar{\boldsymbol{x}}_{ji}) \right). \tag{13}$$

The unknown normal stress term is approximated by averaging the non-equilibrium distribution (a variation of the idea introduced in [16]),

$$(\boldsymbol{n} \cdot \nabla)(\boldsymbol{u} \cdot \boldsymbol{n})(\bar{\boldsymbol{x}}_{ji}) \approx -\frac{1}{2h^2} \sum_{k=1}^{N} B_k(\boldsymbol{n})(A_k + A_{k^*}) \boldsymbol{f}^{neq}(t_n, \bar{\boldsymbol{x}}_{ji}), \tag{14}$$

in which A_k is the kth row of the collision matrix A and B_k are functions from $\mathbb{R}^d$ to $\mathbb{R}$ defined by

$$B_k(\boldsymbol{s}) = \frac{c_s^{-2}}{2} \left[(\boldsymbol{s} \cdot \boldsymbol{c}_k)^2 - \frac{1}{d} \|\boldsymbol{s}\|^2 \|\boldsymbol{c}_k\|^2 \right], \quad k = 1, \ldots, N. \tag{15}$$

Finally, the non-equilibrium distribution at the intersection point $\bar{\boldsymbol{x}}_{\boldsymbol{j}i}$ is obtained by

$$\boldsymbol{f}^{neq}(t_n, \bar{\boldsymbol{x}}_{\boldsymbol{j}i}) \approx (1 + q_{\boldsymbol{j}i})\boldsymbol{f}^{neq}(n, \boldsymbol{j}) - q_{\boldsymbol{j}i}\boldsymbol{f}^{neq}(n, \boldsymbol{j} + \boldsymbol{c}_i). \tag{16}$$

The reason why (11) implies that Eq. (8a) can be seen from an asymptotic analysis [10,16,17,9]. Since the difference of $f_i + f_{i^*}$ between two consecutive time level is of high order h^4 and the correction term $2\boldsymbol{f}^*(\tilde{\rho} - \hat{\rho})$ is of order h^2, the scheme (11) leads to the condition (8a) in leading order.

3.2. Case: $\boldsymbol{c}_i \neq \mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}})$ or $\mathcal{G}(\boldsymbol{x}_{\boldsymbol{j}}) \neq \mathcal{P}(\boldsymbol{x}_{\boldsymbol{w}})$

This case generally occurs if the incoming velocity $\boldsymbol{c}_i$ is rather tangential than normal. However, the opposite may also happen in some extreme situations as depicted in the middle plot of Fig. 2.

The update formula has the basic form

$$f_i(n+1, \boldsymbol{j}) = F_i(\hat{\rho}, \hat{\boldsymbol{u}})(n, \boldsymbol{w}) + f_i^{neq}(n, \boldsymbol{w}) - (A\boldsymbol{f}^{neq})_i(n, \boldsymbol{w}) + g_i(n, \boldsymbol{w}), \tag{17}$$

which is obtained by performing a lattice Boltzmann step (4) at the boundary node $\boldsymbol{x}_{\boldsymbol{j}}$ and applying $\boldsymbol{f} = \boldsymbol{f}^{eq} + \boldsymbol{f}^{neq}$. The required values at the outer-neighbor node $\boldsymbol{x}_{\boldsymbol{w}}$ are extrapolated. The idea to use extrapolation at the non-fluid node $\boldsymbol{x}_{\boldsymbol{w}}$ and not at $\boldsymbol{x}_{\boldsymbol{j}}$ is inspired by the paper [6].

Since $\mathcal{P}(\boldsymbol{x}_{\boldsymbol{w}})$ maps the outer-neighbor node $\boldsymbol{x}_{\boldsymbol{w}}$ to an incoming direction $\boldsymbol{c}_p$ which guarantees two fluid neighbor nodes on the corresponding grid line, the fluid velocity and density can be extrapolated to the node $\boldsymbol{x}_{\boldsymbol{w}}$ imposing the Neumann condition of the tangential velocity component (Eq. (8b)). For convenience, we denote the first fluid neighbor node as $\boldsymbol{x}_{\boldsymbol{m}} = \boldsymbol{x}_{\boldsymbol{w}+\boldsymbol{c}_p}$. Then we set

$$\begin{aligned}
&\hat{\rho}(n, \boldsymbol{w}) = 2\hat{\rho}(n, \boldsymbol{m}) - \hat{\rho}(n, \boldsymbol{m} + \boldsymbol{c}_p),\\
&g_i(n, \boldsymbol{w}) = 2g_i(n, \boldsymbol{m}) - g_i(n, \boldsymbol{m} + \boldsymbol{c}_p),\\
&\hat{\boldsymbol{u}}(n, \boldsymbol{w}) = \frac{4q_{\boldsymbol{m}p}}{1 + 2q_{\boldsymbol{m}p}}\hat{\boldsymbol{u}}(n, \boldsymbol{m}) + \frac{1 - 2q_{\boldsymbol{m}p}}{1 + 2q_{\boldsymbol{m}p}}\hat{\boldsymbol{u}}(n, \boldsymbol{m} + \boldsymbol{c}_p) - \frac{1}{1 + 2q_{\boldsymbol{m}p}}\left[\gamma_1\boldsymbol{t}_{\boldsymbol{m}p} + \frac{1}{a}(\gamma_2 - b\gamma_1)\boldsymbol{n}_{\boldsymbol{m}p}\right],
\end{aligned} \tag{18}$$

where $\boldsymbol{n}_{\boldsymbol{m}p}$ and $\boldsymbol{t}_{\boldsymbol{m}p}$ are defined as follows: at $\bar{\boldsymbol{x}}_{\boldsymbol{m}p}$, we split $\boldsymbol{c}_p = a\boldsymbol{n}_{\boldsymbol{m}p} + b\boldsymbol{t}_{\boldsymbol{m}p}$, where $\boldsymbol{n}_{\boldsymbol{m}p} = \boldsymbol{n}(\bar{\boldsymbol{x}}_{\boldsymbol{m}p})$ is the unit outer normal direction and $\boldsymbol{t}_{\boldsymbol{m}p}$ is a tangential direction. In addition, γ_1 and γ_2 are defined by

$$\begin{aligned}
\gamma_1 &= -b\sum_{k=1}^{N} B_k(\boldsymbol{t}_{\boldsymbol{m}p})(A_k + A_{k^*})\boldsymbol{f}^{neq}(t_n, \bar{\boldsymbol{x}}_{\boldsymbol{m}p}),\\
\gamma_2 &= -\sum_{k=1}^{N} B_k(\boldsymbol{c}_p)(A_k + A_{k^*})\boldsymbol{f}^{neq}(t_n, \bar{\boldsymbol{x}}_{\boldsymbol{m}p}),
\end{aligned} \tag{19}$$

where the non-equilibrium distribution at the intersection point $\bar{\boldsymbol{x}}_{\boldsymbol{m}p}$ is extrapolated similar to (16).

The remaining task is to approximate the non-equilibrium distributions in (17). There are two ways to extrapolate $\boldsymbol{f}^{neq}$ to $\boldsymbol{x}_{\boldsymbol{w}}$. First, analogous to the approach for the density $\hat{\rho}(n, \boldsymbol{w})$, we can approximate the non-equilibrium at $\boldsymbol{x}_{\boldsymbol{w}}$ by

$$\boldsymbol{f}^{neq}(n, \boldsymbol{w}) = 2\boldsymbol{f}^{neq}(n, \boldsymbol{m}) - \boldsymbol{f}^{neq}(n, \boldsymbol{m} + \boldsymbol{c}_p). \tag{20}$$

Secondly, if the boundary node $\boldsymbol{x}_{\boldsymbol{j}}$ has a fluid neighbor node along the incoming direction $\boldsymbol{c}_i$, the following approach used in [6], is also feasible:

$$\boldsymbol{f}^{neq}(n, \boldsymbol{w}) = 2\boldsymbol{f}^{neq}(n, \boldsymbol{j}) - \boldsymbol{f}^{neq}(n, \boldsymbol{j} + \boldsymbol{c}_i). \tag{21}$$

Note that the second approximation is always possible except in the rare case that $\boldsymbol{c}_i$ and $\boldsymbol{c}_{i^*}$ is a pair of opposite, incoming directions. In numerical simulations we have found that the second approximation appears to be more stable.

Using standard asymptotic analysis [10,16,17] we can check that the boundary algorithm (11), (17) yields a second order accurate velocity and a first order accurate pressure.

4. Numerical simulations

4.1. Pressure driven Poiseuille flow in inclined channels

Let θ be the angle between the inclined centerline of the channel and the x-axis. The exact velocity and pressure fields of the 2D pressure driven Poiseuille flows are given by

$$\begin{aligned}
u(x, y, t) &= \bar{y}(1 - \bar{y})\triangle P/(2\nu)\cos(\theta),\\
v(x, y, t) &= -\bar{y}(1 - \bar{y})\triangle P/(2\nu)\sin(\theta),\\
p(x, y, t) &= (x\cos(\theta) - y\sin(\theta))\triangle P + P_0,
\end{aligned} \tag{22}$$

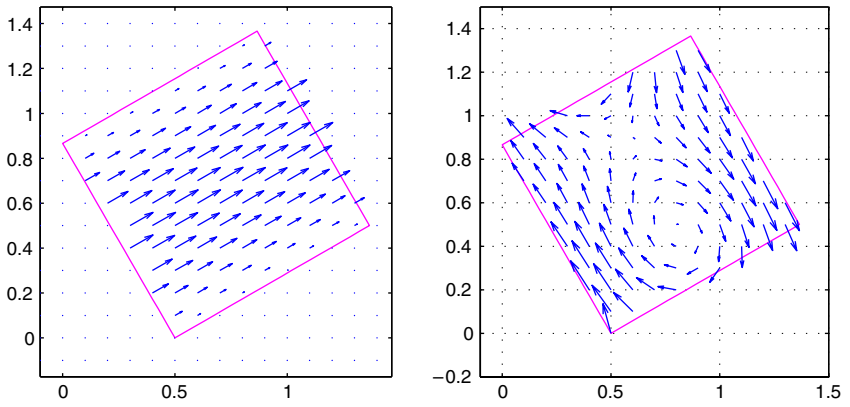

Fig. 3. Velocity profiles of Poiseuille flow (left) and a flow with non-trivial pressure (right) in a channel inclined with an angle $\theta = \pi/6$. The arrows indicate the velocity at each lattice node on a coarse grid $h = 1/10$ in the computational domain.

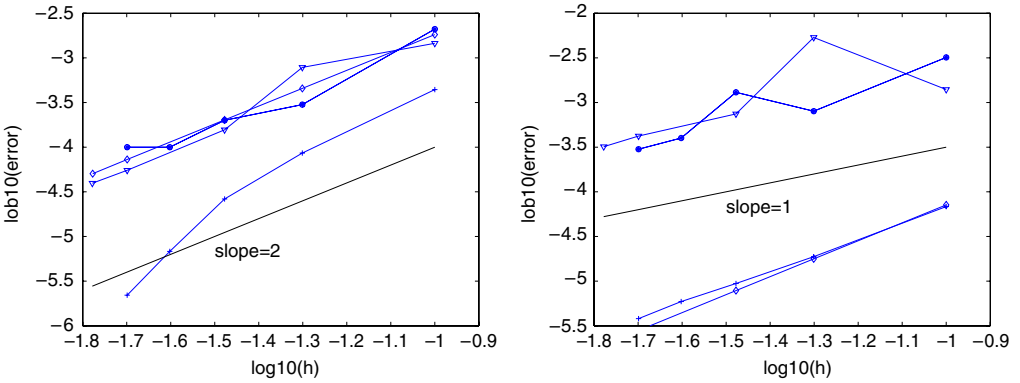

Fig. 4. Error plot of velocity (left) and pressure (right) for Poiseuille flow. The black straight line is a reference line with prescribed slope. The different symbols refer to $\theta = 0$ ($\diamond$), $\theta = \frac{\pi}{6}$ ($\circ$), $\theta = \frac{\pi}{11}$ ($\star$), $\theta = \frac{\pi}{4}$ ($+$) and $\theta = \frac{11}{26}\pi$ ($\triangledown$).

where $\triangle P$ is a constant pressure drop and $\bar{y} = x\sin(\theta) + y\cos(\theta)$. Obviously the isolines of the exact pressure are $x\cos(\theta) - y\sin(\theta) = C$. The velocity profile is displayed in the left plot of Fig. 3.

The computational region is restricted to an inclined unit square. At the channel's rigid walls, the BFL method [18] is implemented to recover the no-slip boundary condition. The proposed algorithm (11), (17) is implemented at the left and right boundaries $x\cos(\theta) - y\sin(\theta) = 0$ and $x\cos(\theta) - y\sin(\theta) = 1$. The average pressure is computed from the exact pressure: $\bar{P}_0 = P_0$, $\bar{P}_1 = P_0 + \triangle P$. Changing the size of the constants P_0 and $\triangle P$, we can set the average pressure values at the boundaries. By determining the numerical error on several grids $h \in \{1/10,\ 1/20, \ldots,\ 1/70\}$ with different inclination angles $\theta \in \{0, \frac{\pi}{6}, \frac{\pi}{11}, \frac{\pi}{4}, \frac{11}{26}\pi\}$, we find that the velocity is second order accurate and the pressure first order accurate (see Fig. 4), which coincides with the predicted accuracy. In cases where the outer normal direction is opposite to an incoming direction, i.e. $\theta = 0, \frac{\pi}{4}$, the accuracy is better than the analytic prediction which may be due to the special nature of the Poiseuille flow: the average pressure is the exact pressure and $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ along the boundary for any angle θ.

4.2. A nonlinear flow

Since the Poiseuille flow is very special, we also test our scheme with a less symmetric flow which has a nonlinear pressure profile on the boundary. The example has been introduced in [9]. The divergence-free velocity field and the pressure are prescribed in the domain $\Omega = [0, 1]^2$,

$$\begin{aligned} u(x, y) &= \frac{a}{\nu\pi}\left(\sin(\pi y) - \frac{2}{\pi}\right)\sin(\pi x), \\ v(x, y) &= \frac{a}{\nu\pi}(\cos(\pi y) + 2y)\cos(\pi x). \\ p(x, y) &= a\sin(\pi(x + y)). \end{aligned} \tag{23}$$

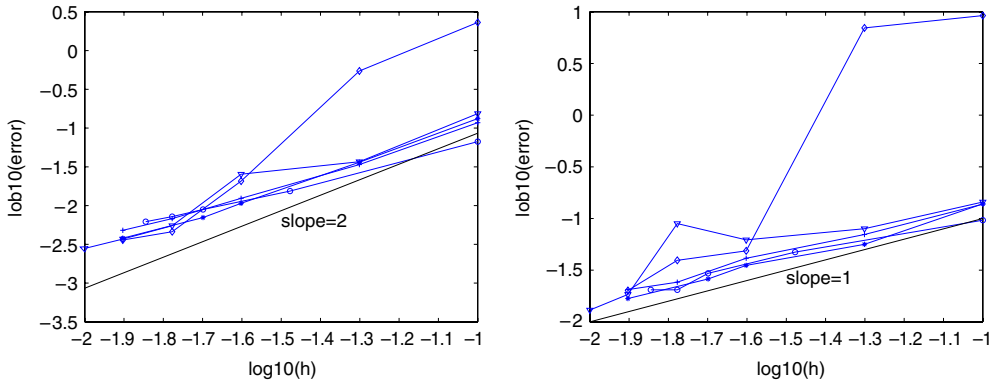

Fig. 5. Error plot of velocity (left) and pressure (right) for nonlinear constructed flows. The black straight lines are reference lines with prescribed slopes. The symbols represent $\theta = \frac{3}{8}\pi$ ($\diamond$), $\theta = \frac{\pi}{17}$ ($\circ$), $\theta = \frac{\pi}{7}$ ($\star$), $\theta = \frac{\pi}{4}$ ($+$), $\theta = \frac{\pi}{8}$ ($\triangledown$).

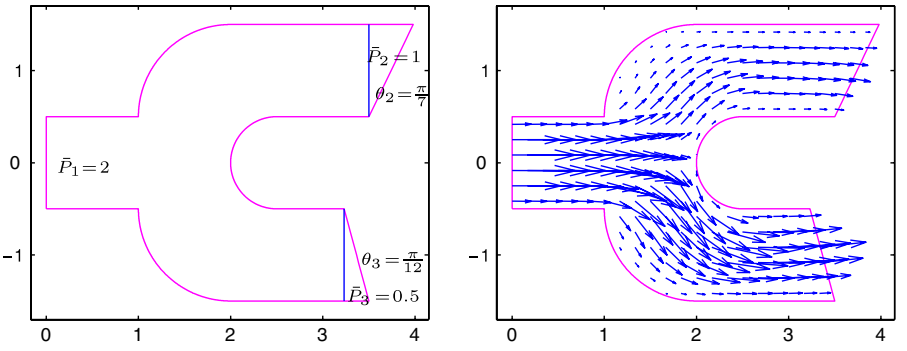

Fig. 6. Left: a schematic explanation of the geometry and the parameters for a pipe flow with three outlets. Right: the corresponding velocity profile.

Here a is an arbitrary constant which we set to 3ν in the following tests. Together with the acceleration $\boldsymbol{G} = (G_x,\ G_y)^{\mathsf{T}}$

$$\begin{aligned} G_x &= u\partial_x u + v\partial_y u + a\pi\cos\pi(x-y) - 2a\sin(\pi x), \\ G_y &= u\partial_x v + v\partial_y v + a\pi\cos\pi(x+y) + 2a\pi\cos(\pi x)(\cos(\pi y) + y), \end{aligned} \tag{24}$$

we obtain a solution of the incompressible Navier–Stokes equation (1) with suitable Dirichlet velocity conditions at $y = 0$ and $y = 1$. The pressure satisfies relation (3) at $x = 0$ and $x = 1$. The average pressures are $\bar{P}_0$ at $x = 0$ and $\bar{P}_1$ at $x = 1$ with

$$\bar{P}_0 = \frac{2a}{\pi}, \qquad \bar{P}_1 = -\frac{2a}{\pi}. \tag{25}$$

Similar to the Poiseuille flow, we simulate this nonlinear flow in a coordinate system rotated with an angle θ. An exemplary flow profile with $\theta = \pi/6$ is indicated in the right plot of Fig. 3.

The Dirichlet velocity boundary condition is obtained from the exact values of the velocity field and is realized by the BFL rule [18]. At the remaining boundaries, the proposed scheme is applied.

In the setup of the lattice Boltzmann method, the velocity set $\mathbb{V}$ is of type D2Q9. As collision model, we have used the MRT model as in [19] with eigenvalues $s_8 = s_9 = 1/\tau$ and the other nonzero eigenvalues close to 1, for example, $s_2 = 1.13$, $s_3 = 1.14$, $s_5 = s_7 = 1.2$.

The numerical tests on several grids show a second order accurate velocity and a first order accurate pressure. A numerical order study is given in Fig. 5. The right part of Fig. 5 indicates that the non-trivial pressure on the boundary is resolved using condition (3) with the given average pressure.

4.3. Pipe flows with three outlets

In this section we simulate stationary flows around a junction in a 2D pipe system. Fig. 6 shows a schematic explanation. All three outlets have identical width, but are assigned different average pressure values. The right upper and lower outlets are inclined with different angles, as a model for cases where outlets are not perpendicular to the pipe axis (e.g., systems of blood vessels which leave the computational box at various angles.).

The lattice Boltzmann setup is the same as in Section 4.2. The flow is initially at rest. At rigid walls, the BFL rule is used to recover the no-slip condition. The newly developed algorithm is applied to realize the average pressure condition at the outlets.

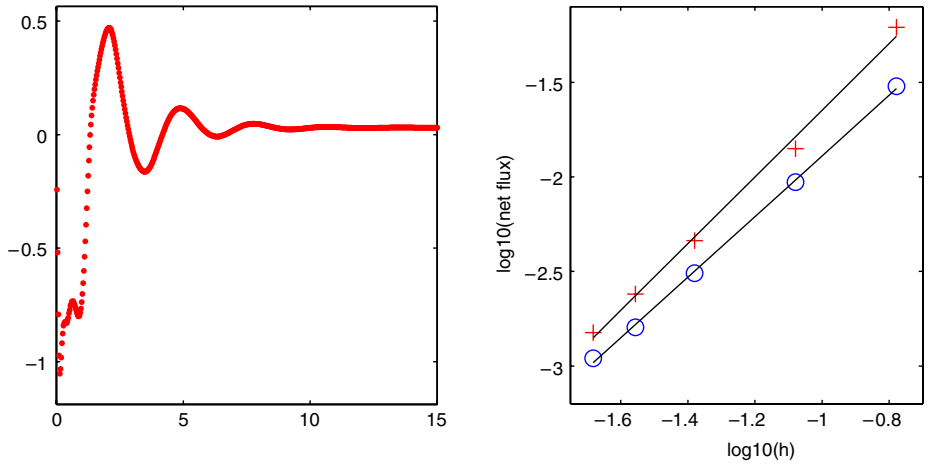

Fig. 7. Left: the behavior of the net flux for the flow in Fig. 6 before the stationary state is reached. Right: the net flux on several grids.The convergence rate is 1.6 for the flow in Fig. 6 (marked with ∘), and 1.8 (+) for a flow with parameters: $\theta_2 = \frac{\pi}{8}, \theta_3 = \frac{\pi}{12}, \bar{P}_1 = 0, \bar{P}_2 = 1.5, \bar{P}_3 = 2$.

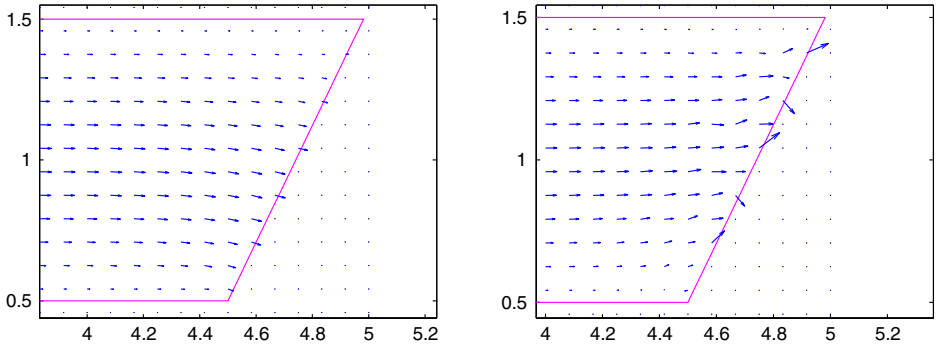

Fig. 8. The velocity profile at the right upper outlet in the pipe system by applying the boundary algorithm (11), (17) with (left) and without (right) the viscous term. The configuration of the flow is similar to the one in Fig. 6 with longer pipes.

The right plot of Fig. 6 illustrates the numerical velocity profile of the flow with a mild viscosity $\nu = 0.1$ on a coarse grid $h = 1/6$. The flow streams in at the left outlet (high pressure) and leaves mostly at the right lower outlet due to the bigger pressure drop compared to the right upper one.

We also compare the net flux with its theoretical value zero on several grids (see right plot of Fig. 7). For several flows with different shear viscosities, we find that the convergence rate is about 2. Since the net flux is a linear function of the boundary velocity, the numerical convergence rate supports the analytically predicted accuracy very well. The left plot of Fig. 7 shows the net flux behavior on the coarse grid $h = 1/6$ before the flow reaches the stationary state. The net flux is eventually close to zero after the initial oscillation.

5. Conclusion and discussion

We have constructed a lattice Boltzmann algorithm for the average pressure outflow boundary condition (3). This algorithm applies to arbitrary boundary shapes and requires only one neighbor fluid node in an incoming direction. Theoretically, second order accurate velocity and first order accurate pressure fields are recovered for both 2D and 3D flows.

For the practically relevant case of inclined boundaries, the numerical verification is carried out in several 2D tests. The results for the Poiseuille flow and a more general flow demonstrate the advantage of this boundary condition: the correct pressure profile on the boundary is recovered, while only the average pressure is prescribed.

We note that, if the viscous term in (3) is dropped, the proposed algorithm actually recovers the following hydrodynamic condition

$$p = \bar{P}, \qquad \frac{\partial u_t}{\partial n} = 0, \tag{26}$$

and thus imposes a constant pressure on the boundary. Applied to the flow in Fig. 6 on several grids, it leads to a different flow at the inclined outlets (see Fig. 8). The magnitude of the velocity difference and pressure difference does not decrease when the grid becomes finer. However, the difference is obvious only in the regions close to the inclined boundaries. At the

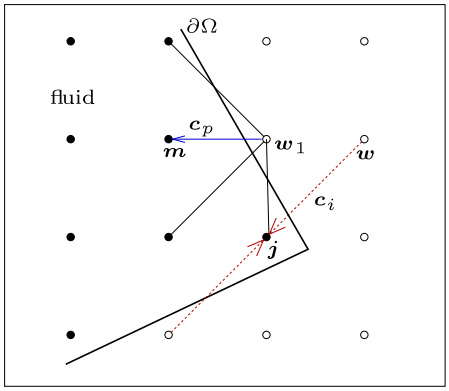

Fig. B.1. A rare situation where the definition of $\mathcal{P}(\boldsymbol{x}_w)$ is not straightforward. The outer-neighbor nodes and fluid nodes are represented by ∘ and • respectively. For the outer node $\boldsymbol{x}_w$, only on the $\boldsymbol{c}_i$ grid line there is an associated fluid node $\boldsymbol{x}_j$. No second fluid node exists in direction $\boldsymbol{c}_i$.

left outlet, no obvious difference occurs which may be due to the fact that the pressure is almost constant in the stationary flow.

Acknowledgments

The author is grateful to Prof. Junk for revising this paper and Prof. Li-Shi Luo for the helpful discussions. This work is supported by the Deutsche Forschungsgemeinschaft through the grant DFG JU440/1–4.

Appendix A. Definition of the mapping $\mathcal{G}$

For an arbitrary boundary node $\boldsymbol{x}_j$, we select a velocity direction $\boldsymbol{c}_n$ according to the rule

$$\left\| \frac{\boldsymbol{c}_n}{\|\boldsymbol{c}_n\|} + \boldsymbol{n}_{jn} \right\| = \min_{\boldsymbol{c}_k \in \mathbb{V}_j} \left\{ \left\| \frac{\boldsymbol{c}_k}{\|\boldsymbol{c}_k\|} + \boldsymbol{n}_{jk} \right\| \right\}, \tag{A.1}$$

where $\mathbb{V}_j$ is the set of the incoming velocities $\boldsymbol{c}_i$ at the boundary node $\boldsymbol{x}_j$ or at a close-by neighbor node of $\boldsymbol{x}_j$, $\boldsymbol{n}_{jk} = \boldsymbol{n}(\bar{\boldsymbol{x}}_{jk})$ is the outer normal direction at the intersection point $\bar{\boldsymbol{x}}_{jk}$ of the line $\boldsymbol{x}_j - \lambda \boldsymbol{c}_k$ ($\lambda \geq 0$) and the boundary $\partial\Omega$. This give rise to the definition

$$\mathcal{G}(\boldsymbol{x}_j) = \boldsymbol{c}_n. \tag{A.2}$$

Note that $\boldsymbol{c}_n$ is closest to the inner normal direction $-\boldsymbol{n}$ at the boundary segment near the node $\boldsymbol{x}_j$.

Appendix B. Definition of the mapping $\mathcal{P}$

In general, an outer-neighbor node $\boldsymbol{x}_w$ is related to several pairs of boundary nodes and incoming directions, so that we can define a mapping $\mathcal{P}$ which associates $\boldsymbol{x}_w$ to an incoming direction $\boldsymbol{c}_p$ ensuring that on the corresponding grid line both nodes $\boldsymbol{x}_{w+c_p}$ and $\boldsymbol{x}_{w+2c_p}$ are in Ω. Moreover, we select the direction $\boldsymbol{c}_p$ to be relatively close to the normal direction. Altogether,

$$\mathcal{P}(\boldsymbol{x}_w) = \boldsymbol{c}_p. \tag{B.1}$$

is defined by

$$\left\| \boldsymbol{n}_{wp} + \frac{\boldsymbol{c}_p}{\|\boldsymbol{c}_p\|} \right\| = \min_{\boldsymbol{c}_k \in \mathbb{V}} \left\{ \left\| \boldsymbol{n}_{wk} + \frac{\boldsymbol{c}_k}{\|\boldsymbol{c}_k\|} \right\| \,|\, \boldsymbol{x}_{w+c_p} \in \Omega,\ \boldsymbol{x}_{w+2c_p} \in \Omega \right\}. \tag{B.2}$$

A very rare situation might occur (for example at a corner), when the node $\boldsymbol{x}_w$ is related to opposite, incoming directions. Then the set in (B.2) is empty and $\mathcal{P}$ is not defined. In 2D, such an example is displayed in Fig. B.1.

To cope with this special case, we seek a velocity direction $\boldsymbol{c}_p$ so that the node $\boldsymbol{x}_{w_1} = \boldsymbol{x}_{w+c_p}$ is another outer-neighbor node which has two fluid neighbor nodes on the link of $\boldsymbol{c}_p$. Then we define

$$\mathcal{P}_1(\boldsymbol{x}_w) = \boldsymbol{c}_p. \tag{B.3}$$

Accordingly, for achieving the values at $\boldsymbol{x}_{\boldsymbol{w}}$ used in the boundary scheme (17), the extrapolation coefficients change. Now let $\boldsymbol{x}_{\boldsymbol{m}} = \boldsymbol{x}_{\boldsymbol{w}+2\boldsymbol{c}_p}$, we then set

$$\begin{aligned}
\hat{\rho}(n, \boldsymbol{w}) &= 3\hat{\rho}(n, \boldsymbol{m}) - 2\hat{\rho}(n, \boldsymbol{m} + \boldsymbol{c}_p),\\
\boldsymbol{g}(n, \boldsymbol{w}) &= 3\boldsymbol{g}(n, \boldsymbol{m}) - 2\boldsymbol{g}(n, \boldsymbol{m} + \boldsymbol{c}_p),\\
\boldsymbol{f}^{neq}(n, \boldsymbol{w}) &= 3\boldsymbol{f}^{neq}(n, \boldsymbol{m}) - 2\boldsymbol{f}^{neq}(n, \boldsymbol{m} + \boldsymbol{c}_p),\\
\hat{\boldsymbol{u}}(n, \boldsymbol{w}) &= \frac{6q_{\boldsymbol{m}p} - 3}{2q_{\boldsymbol{m}p} + 1}\hat{\boldsymbol{u}}(n, \boldsymbol{j}) + \frac{4 - 4q_{\boldsymbol{m}p}}{2q_{\boldsymbol{m}p} + 1}\hat{\boldsymbol{u}}(n, \boldsymbol{m} + \boldsymbol{c}_p) - \frac{3}{2q_{\boldsymbol{m}p} + 1}\left[\gamma_1 \boldsymbol{t}_{\boldsymbol{m}p} + \frac{1}{a}(\gamma_2 - b\gamma_1)\boldsymbol{n}_{\boldsymbol{m}p}\right].
\end{aligned} \tag{B.4}$$

The formula for γ_1 and γ_2 is the same as in (19).

In principle, this idea can be repeated by considering $\boldsymbol{x}_{\boldsymbol{w}_2} = \boldsymbol{x}_{\boldsymbol{w}+2\boldsymbol{c}_p}$ and $\mathcal{P}_2(\boldsymbol{x}_{\boldsymbol{w}}) = \boldsymbol{c}_p$. However, if $\mathcal{P}_2$ is required, a finer grid is definitely recommended.

References

[1] P.M. Gresho, Some current CFD issues relevant to the incompressible Navier–Stokes equations, Comput. Methods Appl. Mech. Eng. 87 (1991) 211–252.
[2] P.M. Gresho, Incompressible fluid dynamics: Some fundamental formulation issues, Annu. Rev. Fluid Mech. 23 (1991) 413–453.
[3] J. Heywood, R. Rannacher, S. Turek, Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations, Int. J. Numer. Methods Fluids 22 (1996) 325–352.
[4] G.P. Galdi, A.M. Robertson, R. Rannacher, S. Turek, Mathematical Problems in Classical and Non-Newtonian Fluid Mechanics, hemodynamical flows edition, in: Oberwolfach Seminars, vol. 37, Birkhäuser Basel, 2008.
[5] Q. Zou, X. He, On pressure and velocity boundary conditions for the lattice Boltzmann BGK model, Phys. Fluids 9 (6) (1997) 1591–1598.
[6] Z. Guo, C. Zheng, B. Shi, An extrapolation method for boundary conditions in lattice Boltzmann method, Phys. Fluids 14 (6) (2002) 2007–2010.
[7] I. Ginzburg, F. Verhaeghe, D. d'Humières, Two-relaxation-time lattice Boltzmann schemes: About parametrization, velocity, pressure and mixed boundary conditions, Commun. Comput. Phys. 3 (2) (2008) 422–478.
[8] F. Verhaeghe, L.-S. Luo, B. Blanpain, Lattice Boltzmann modeling of microchannel flow in slip flow regime, J. Comput. Phys. 228 (2009) 147–157.
[9] M. Junk, Z. Yang, Pressure boundary conditions for the lattice Boltzmann method, Comput. Math. Appl. 58 (5) (2009) 922–929.
[10] M. Junk, A. Klar, L.-S. Luo, Asymptotic analysis of the lattice Boltzmann equation, J. Comput. Phys. 210 (2005) 676–704.
[11] M. Junk, D. Kehrwald, On the relation between lattice variables and physical quantities in lattice Boltzmann simulations, ITWM Report, 2006.
[12] Y. Qian, D. d'Humières, P. Lallemand, Lattice BGK models for Navier-Stokes equation, Europhys. Lett. 17 (1992) 479–484.
[13] R. Mei, L.-S. Luo, W. Shyy, An accurate curved boundary treatment in the lattice Boltzmann method, J. Comput. Phys. 155 (1999) 307–330.
[14] D. d'Humières, I. Ginzburg, M. Krafczyk, P. Lallemand, L.-S. Luo, Multiple-relaxation-time lattice Boltzmann models in three dimensions, Phil. Trans. R. Soc. Lond. A 360 (2002) 437–451.
[15] M. Junk, Z. Yang, Outflow boundary conditions for the lattice Boltzmann method, Prog. Comput. Fluid Dyn. 8 (2008) 38–48.
[16] M. Junk, Z. Yang, Asymptotic analysis of lattice Boltzmann boundary conditions, J. Statist. Phys. 121 (2005) 3–35.
[17] Z. Yang, Analysis of lattice Boltzmann boundary conditions, Ph.D. Thesis, Universität Konstanz, Konstanz, 2007.
[18] M. Bouzidi, M. Firdaouss, P. Lallemand, Momentum transfer of a Boltzmann-lattice fluid with boundaries, Phys. Fluids 13 (11) (2001) 3452–3459.
[19] P. Lallemand, L.-S. Luo, Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability, Phys. Rev. E 61 (2000) 6546–6562.