KiSSAM User Guide and Documentation. Principal basic physical models and approaches

Andrey Zakirov, Sergei Belousov, Anastasia Perepelkina, Boris Korneev, Maria Bogdanova

2023

Contents

1 Introduction

KiSSAM stands for Kintech Simulation Software for Additive Manufacturing. As its name clearly suggests, KiSSAM is a multiphysics simulation software tool designed primarily for high-performance and high-fidelity modeling of metal powder bed fusion (PBF) additive manufacturing processes at the mesoscale level. The current version of KiSSAM supports simulations of both direct metal laser melting (DMLM) and electron beam melting (EBM) varieties of PBF.

At the core of KiSSAM lies a powerful hydrodynamic solver for weakly compressible fluids based on the Thermal Lattice Boltzmann Method (TLBM) with Volume of Fluid (VoF) free surface tracking. This solver is used to simulate melt pool dynamics during the PBF process. In order to facilitate the high-fidelity modeling of the PBF, the following physical processes are also modeled in KiSSAM:

- heat transfer,
- melting and solidification phase transitions,
- surface wetting,
- Marangoni convection,
- recoil pressure of evaporating metal,
- evaporation and radiative cooling of melt,
- propagation and absorption of laser beam in both the powder bed and curved melt surface accounting for the multiple reflections of the beam,
- propagation and absorption of electron beam in both the powder bed and curved melt surface accounting for the electron reflection at the material surface.

The software code is written in CUDA C++ and is designed for high-performance computing at workstations equipped with Nvidia General Purpose Graphics Processing Units (GPGPU).

Detailed description of the underlying physical models and equations can be found in [].

2 Hardware and Software System Reqirements

KiSSAM requires a workstation with Linux OS and at least one NVidia GPU. Minimal hardware requirements are:

- x86 CPU (Intel i5 processor or AMR Ryzen 5 desktop or server);
- NVidia GPU Pascal or newer architecture with more than 8 GB GPU memory;
- RAM 32 GB RAM.

Software requirements:

- Ubuntu 18.04+
- X server
- NVidia latest driver
- NVidia Cuda toolkit ver. 11 (preferable)

- python 3.6
- numpy
- python pillow
- GLut library (libglut 3 or freeglut3 package);
- VirtualGL (recommended);
- OpenVDB library (recommended).

RECOMMENDED THIRD PARTY SOFTWARE:

- A text editor or JSON editor for configuration.
- vdb_view or Blender for viewing output geometry.
- Preferred data analysis software: MS Excel, Matlab, etc.

3 Description of physics and modelling approaches

3.1 Simulation domain and grids

KiSSAM has a cuboidal simulation domain. There are 3 types of grid: **Meltpool Grid** , **Global Geometry Grid**, **Tractile Mesh**.

Meltpool Grid is a smallest grid which covers the meltpool region only. This grid is rectangular and contains of cubic cells with $\Delta r \times \Delta r \times \Delta r$ size each. The grid is adaptive and follows the liquid melt pool. The melt pool is always enclosed by the Meltpool Grid.

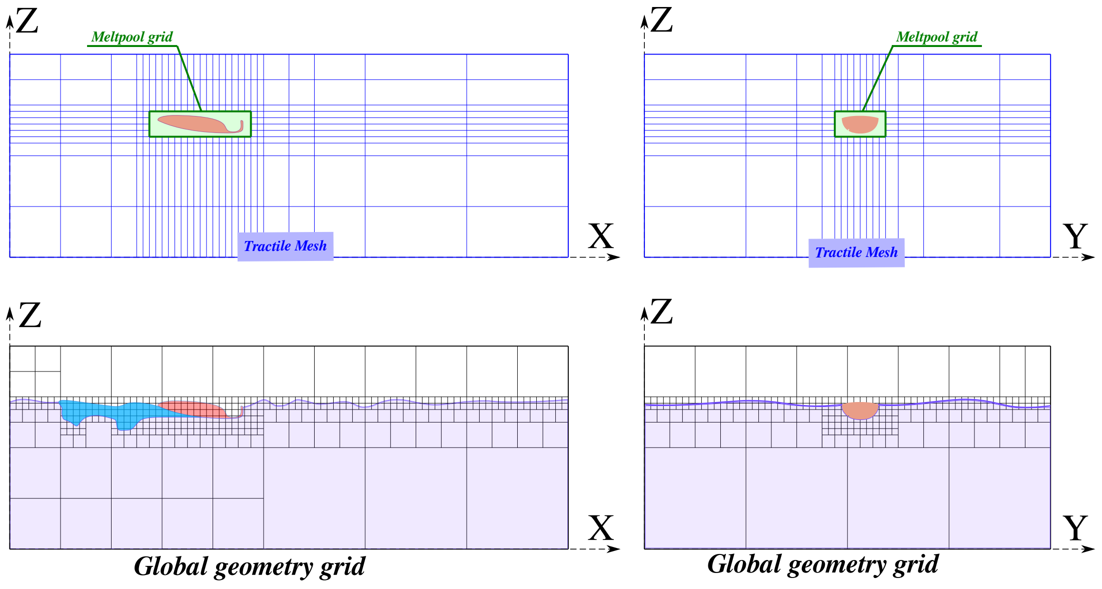

Figure 1: Types of grids used in **KiSSAM**

3.2 Underlying physical models

3.2.1 Hydrodynamic model

For simulation of the liquid meltpool the weakly compressible solver based on Thermal Lattice Boltzmann Method (TLBM) with Volume of Fluid (VoF) free surface tracking is used. The model includes several important extensions to the standard LBM: free surface tracking with tension, wetting and other acting forces; heat transfer; and melting and solidification phase transitions.

In the simulation, the computational domain (**Meltpool Grid**) is divided into equal cubic cells with $\Delta x = \Delta y = \Delta z = \Delta r$. Each cell has a type that defines its phase: *Solid*, *Fluid*, or *Void*. Some *Fluid* cells can be also *Interface* sub-type. *Interface* cells separates *Fluid* and *Void* cells assure them not to be adjacent.

In each *Fluid* cell, the lattice Boltzmann equation (LBE) is solved for a discrete set of distribution functions (DFs) f_i. We use the LBM with the D3Q27 model and single-relaxation time BGK scheme [11]:

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) - \frac{f_i(\vec{x}, t) - f_i^{eq}(\rho, \vec{u})}{\tau_f} + F_i, \tag{1}$$

where $\vec{c}_i = \{l, m, n\}, l, m, n \in 0, \pm 1, i = 0, ..., 26$ are discrete velocity vectors corresponding to discrete DF values f_i, F_i is the force term. The equilibrium distribution is taken as the basic polynomial form

$$\vec{f}_i^{eq}(\rho, \vec{u}) = w_i \rho \left(1 + \frac{\vec{c}_i \vec{u}}{c_s^2} + \frac{1}{2}\frac{(\vec{c}_i \vec{u})^2}{c_s^4} - \frac{1}{2}\frac{\vec{u}^2}{c_s^2}\right), \tag{2}$$

where $\rho(\vec{x}, t) = \sum_i f_i$ and $\rho(\vec{x}, t)\vec{u}(\vec{x}, t) = \sum_i \vec{c}_i f_i$ represent the local fluid density and velocity, w_i are the constant weights defined in the LBM, $c_s^2 = c^2/3$, with $c = \Delta x/\Delta t$ is the lattice step, and the fluid viscosity is $\mu = \dfrac{\tau_f - 1/2}{3}$, which gives the unit conversion system.

The basic LBM algorithm consists of two simple steps, corresponding to the first and second terms in (1). The streaming step copies each DF f_i from one cell to its neighbor in the $\vec{c}_i$ direction. The collision step locally computes the macro variables and updates f_i.

For heat transfer and energy distribution in **Meltpool Grid** in *Solid* and *Fluid* cells the double distribution function (DDF) approach is implemented, where one more set of 27 LBEs is solved:

$$h_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = h_i(\vec{x}, t) - \frac{h_i(\vec{x}, t) - h_i^{eq}(\rho, \vec{u})}{\tau_h} + Q_i, \tag{3}$$

where Q_i is the source term and the equilibrium distribution is

$$\vec{h}_i^{eq} = w_i E \left(1 + \frac{\vec{c}_i \vec{u}}{c_s^2} + \frac{1}{2}\frac{(\vec{c}_i \vec{u})^2}{c_s^4} - \frac{1}{2}\frac{\vec{u}^2}{c_s^2}\right),$$

Here, $E = \sum_i h_i = E(\vec{x}, t)$ is the local energy density. The thermal diffusivity is $k = \dfrac{\tau_h - 1/2}{3}$. The energy (or enthalpy) E is strictly associated with temperature T as:

$$E(T) = \begin{cases} E_{solid}(T) & \text{if } T \le T_{solidus}; \\ E_{solidus} + L\dfrac{T - T_{solidus}}{T_{liquidus} - T_{solidus}} & \text{if } T_{solidus} < T \le T_{liquidus}; \\ E_{solidus} + L + E_{liquid}(T) & \text{if } T > T_{liquidus}. \end{cases} \tag{4}$$

Here $E_{solid}(T)$ and $E_{liquid}(T)$ are temperature-dependent piecewise linear enthalpy approximation in solid and in liquid phases, L is latent heat of melting, $T_{solidus}$ and $T_{liquidus}$ are solidus and liquidus temperatures, $E_{solidus} = E_{solid}(T_{solidus})$ is enthalpy at solidus point.

The source term Q_i is either volumetric heat source (for electron beam) or surface energy normal gradient (for laser beam). Volumetric heat source has the following form: $Q_i{=}w_i E_{source}$, where E_{source} is the source volumetric energy density. The surface energy normal gradient is calculated similarly to [1]:

$$Q_i = w_i \left(\frac{2}{c_s^2} \left(\vec{c}_i, \vec{Q}\right) + \frac{4}{c_s^4} (\vec{c}_i, \vec{u}) \left(\vec{c}_{aver}, \vec{Q}\right) \right) \quad \text{if } (\vec{c}_i - \vec{u}, \vec{n}) \leq 0,$$

where $\vec{c}_{aver} = \sum\limits_{(\vec{c}_i - \vec{u}, \vec{n}) \leq 0} w_i \vec{c}_i$, $\vec{Q} = -\vec{n} E_{source}$ and $\vec{n}$ is a normal at the surface directed from fluid or solid.

Cell types change algorithm

The free surface is simulated with the VoF method, where the filled fraction of every cell is tracked. This approach allows us to consistently describe the sharp change of density across the fluid surface boundary.

Each cell of **Meltpool Grid** stores the cell filling fraction φ value ($0 \leqslant \varphi \leqslant 1$). In *Void* cells $\varphi = 0$, in *Fluid* cells not of the *Interface* sub-type $\varphi = 1$, in *Solid* and *Interface* cells φ ranges from 0 to 1. Additionally the intermediate value Δm (cell mass change) is tracked in every cell of the **Meltpool Grid**.

The step Volume of Fluid step consists of 4 sub-steps:

1. Calculate Δm only in the *Interface* cells. This is performed in according with the following equation:

$$\begin{aligned} \Delta m &= \sum_i^Q F_i + \Delta m^{evap}; \\ F_i &= F_i^{\{n\}} + k_{ad} F_i^{\{ad\}}. \end{aligned} \tag{5}$$

$$F_i^{\{n\}}(\vec{x}) = \begin{cases} -f_i(\vec{x} + \vec{c}_i) + f_{\bar{i}^r}(\vec{x}) & \text{if } (\vec{x} + \vec{c}_i) \text{ is } \textit{Fluid} \text{ cell };\\ \dfrac{(\varphi(\vec{x}) + \varphi(\vec{x} + \vec{c}_i))}{2} (-f_i(\vec{x} + \vec{c}_i) + f_{\bar{i}^r}(\vec{x})) & \text{if } (\vec{x} + \vec{c}_i) \text{ is } \textit{Interface} \text{ cell };\\ 0 & \text{if } (\vec{x} + \vec{c}_i) \text{ is } \textit{Void} \text{ or } \textit{Solid} \text{ cell }. \end{cases} \tag{6}$$

Here $\Delta m^{\{ev\}}$ denotes mass loss during evaporation, which is evaluated from the evaporation solver, see sec. 3.2.6. And $\bar{i}^r$ stands for the inverse direction ($\vec{c}_{\bar{i}^r} = -\vec{c}_i$). The term $F_i^{\{ad\}}$ designates as anti-diffusion correction, which is necessary to make the surface sharper. The constant coefficient k_{ad} regularizes the anti-diffusion magnitude (default 0.1).

The $F_i^{\{ad\}}$ is evaluated from the following table [10]:

cell $\vec{x}$ type (sub-type)	cell $(\vec{x}+\vec{c}_i)$ type (sub-type)	$(\vec{c}_i \cdot \vec{n}_{aver})$	$F_i^{\{ad\}}$
Interface (sub-type **F**)	*Interface* (sub-type **I**)	> 0	$+\frac{(\vec{c}_i \cdot \vec{n}_{aver})}{\lvert\vec{c}_i\rvert\lvert\vec{n}_{aver}\rvert}\varphi(\vec{x}+\vec{c}_i) f_{\bar{i}^r}(\vec{x})$
Interface (sub-type **V**)	*Interface* (sub-type **I**)	< 0	$+\frac{(\vec{c}_i \cdot \vec{n}_{aver})}{\lvert\vec{c}_i\rvert\lvert\vec{n}_{aver}\rvert}\left(1-\varphi(\vec{x}+\vec{c}_i)\right) f_i(\vec{x}+\vec{c}_i)$
Interface (sub-type **I**)	*Interface* (sub-type **F**)	< 0	$+\frac{(\vec{c}_i \cdot \vec{n}_{aver})}{\lvert\vec{c}_i\rvert\lvert\vec{n}_{aver}\rvert}\varphi(\vec{x}) f_i(\vec{x}+\vec{c}_i)$
Interface (sub-type **F**)	*Interface* (sub-type **F** of **V**)	< 0	$-\varphi(\vec{x}) f_i(\vec{x}+\vec{c}_i)$
Interface (sub-type **F**)	*Interface* (sub-type **F**)	> 0	$+\varphi(\vec{x}+\vec{c}_i) f_{\bar{i}^r}(\vec{x})$
Interface (sub-type **F**)	*Interface* (sub-type **V**)	> 0	$+\left(1-\varphi(\vec{x})\right) f_{\bar{i}^r}(\vec{x})$
Interface (sub-type **V**)	*Interface* (sub-type **F** or **V**)	< 0	$-\left(1-\varphi(\vec{x}+\vec{c}_i)\right) f_i(\vec{x}+\vec{c}_i)$
Interface (sub-type **V**)	*Interface* (sub-type **F**)	> 0	$+\varphi(\vec{x}+\vec{c}_i) f_{\bar{i}^r}(\vec{x})$
Interface (sub-type **V**)	*Interface* (sub-type **V**)	> 0	$+\left(1-\varphi(\vec{x})\right) f_{\bar{i}^r}(\vec{x})$
Interface (sub-type **I**)	*Interface* (sub-type **V**)	> 0	$+\frac{(\vec{c}_i \cdot \vec{n}_{aver})}{\lvert\vec{c}_i\rvert\lvert\vec{n}_{aver}\rvert}\left(1-\varphi(\vec{x})\right) f_{\bar{i}^r}(\vec{x})$
Otherwise			0

Here $\vec{n}_{aver} = \frac{1}{2}(\vec{n}(\vec{x}) + \vec{n}(\vec{x}+\vec{c}_i))$, where $\vec{n}$ is a surface normal directed from fluid, evaluated from curvature calculation algorithm (see sec. 3.2.3).

Interface cell has three sub-types: **F**,**V**,**I**:
sub-type **F** cell surrounded only by *Fluid* and *Interface* adjacent cells,
sub-type **V** cell surrounded only by *Void* and *Interface* adjacent cells, and sub-type **I** are all other *Interface* cells (namely has both *Fluid* and *Void* cells among neighbouring cells).

2. Solving the following equation for $\varphi(\vec{x}, t+\Delta t)$ at every cells (also *Fluid* and *Void*) with unknown all the $\Delta\Pi_i$ and preserve the total mass conservation:

$$\begin{aligned}
&\rho(\vec{x}, t+\Delta t)\varphi(\vec{x}, t+\Delta t) = \rho(\vec{x}, t)\varphi(\vec{x}, t) + \Delta m(\vec{x}, t) + \sum_i^Q \Delta\Pi_i(\vec{x}, \vec{x}+\vec{c}_i, t);\\
&\Delta\Pi_i(\vec{x}, \vec{x}+\vec{c}_i, t) = -\Delta\Pi_i(\vec{x}+\vec{c}_i, \vec{x}, t);\\
&\sum_{i,\vec{x}} ||\Delta\Pi_i|| \to 0 \text{ at condition } 0 \le \varphi(\vec{x}, t+\Delta t) \le 1.
\end{aligned} \tag{7}$$

Here $\rho(\vec{x}, t+\Delta t)$ is a new density value after LBM streaming step. $\Delta\Pi_i(\vec{x}, \vec{x}+\vec{c}_i, t)$ are additional mass fluxes between all adjacent cells, and they has to be such that the new value $\varphi(\vec{x}, t+\Delta t)$ will be between 0 and 1 in all cells. It it also necessary to keep all $\Delta\Pi_i$ rather small in any reasonable norm $||\cdot||$.

Equations (7) with necessary conditions requires implicit solver. Actually there is no guarantee that the solution is exist or any implicit special solver will converge fast. **KiSSAM** uses the following iteration procedure to find all $\Delta\Pi_i$. The procedure is called "Mass redistribution" and consists of exchange of Δm between all neighbouring cells:

- preliminary iteration $j+1$ changes Δm:

$$\Delta m\left(\vec{x},t^{j+1}\right)=\begin{cases}\Delta m\left(\vec{x},t^{j}\right)-\delta^{+}(\vec{x}), & \text{if } \Delta m^{\{extra\}}(\vec{x})>\delta^{+}(\vec{x});\\ \Delta m\left(\vec{x},t^{j}\right)-\delta^{-}(\vec{x}), & \text{if } \Delta m^{\{extra\}}(\vec{x})<\delta^{-}(\vec{x});\\ \Delta m\left(\vec{x},t^{j}\right), & \text{otherwise };\end{cases}$$

$$\Delta m\left(\vec{x}+\vec{c}_i,t^{j+1}\right)=\begin{cases}\Delta m\left(\vec{x}+\vec{c}_i,t^{j}\right)+\delta^{+}(\vec{x})/N_v(\vec{x}), & \text{if } \Delta m^{\{extra\}}(\vec{x})>\delta^{+}(\vec{x});\\ \Delta m\left(\vec{x}+\vec{c}_i,t^{j}\right)+\delta^{-}(\vec{x})/N_f(\vec{x}), & \text{if } \Delta m^{\{extra\}}(\vec{x})<\delta^{-}(\vec{x});\\ \Delta m\left(\vec{x}+\vec{c}_i,t^{j}\right), & \text{otherwise}\end{cases}$$

$$\delta^{+}(\vec{x})=\begin{cases}+10^{-7}\rho_0 & \text{if } N_v(\vec{x})>0 \text{ and } \vec{x} \text{ is } \textit{Interface} \text{ cell;}\\ 0 & \text{if } N_v(\vec{x})=0 \text{ or } \vec{x} \text{ is not } \textit{Interface} \text{ cell;}\end{cases}$$

$$\delta^{-}(\vec{x})=\begin{cases}-10^{-7}\rho_0 & \text{if } N_f(\vec{x})>0 \text{ and } \vec{x} \text{ is } \textit{Interface} \text{ cell;}\\ 0 & \text{if } N_f(\vec{x})=0 \text{ or } \vec{x} \text{ is not } \textit{Interface} \text{ cell;}\end{cases}$$

$$N_v(\vec{x})=\sum_i^Q\begin{cases}1 & \text{if } (\vec{x}+\vec{c}_i) \text{ is } \textit{Void} \text{ cell}\\ 0 & \text{if } (\vec{x}+\vec{c}_i) \text{ is not } \textit{Void} \text{ cell}\end{cases}$$

$$N_f(\vec{x})=\sum_i^Q\begin{cases}1 & \text{if } (\vec{x}+\vec{c}_i) \text{ is } \textit{Fluid} \text{ cell}\\ 0 & \text{if } (\vec{x}+\vec{c}_i) \text{ is not } \textit{Fluid} \text{ cell}\end{cases}$$

$$\Delta m^{\{extra\}}(\vec{x})=$$
$$\begin{cases}\left(\varphi^{\{test\}}-1\right)\rho(\vec{x},t+\Delta t)=\Delta m(\vec{x},t)+\rho(\vec{x},t)\varphi(\vec{x},t)-\rho(\vec{x},t+\Delta t), & \\ \qquad\qquad\qquad \text{if } \varphi^{\{test\}}-1>10^{-7}; & \\ \varphi^{\{test\}}\rho(\vec{x},t+\Delta t)=\Delta m(\vec{x},t)+\rho(\vec{x},t)\varphi(\vec{x},t), & \\ \qquad\qquad\qquad \text{if } \varphi^{\{test\}}<-10^{-7}; & \\ 0, \quad \text{otherwise;} & \end{cases}$$
$$\varphi^{\{test\}}=\frac{\rho(\vec{x},t)\varphi(\vec{x},t)+\Delta m(\vec{x},t)}{\rho(\vec{x},t+\Delta t)} \tag{8}$$

First of all, if the *Interface* cell current value $\varphi^{\{test\}}=\dfrac{\rho(\vec{x},t)\varphi(\vec{x},t)+\Delta m(\vec{x},t)}{\rho(\vec{x},t+\Delta t)}>1+10^{-7}$ we put very small amount of Δm (about $\delta=10^{-7}\rho_0$) evenly in all neighboring *Void* cells (of course if they are exist). Similarly if $\varphi^{\{test\}}<-10^{-7}$ then the very small amount of $\Delta m=10^{-7}\rho_0$ is taken away from all neighboring *Fluid* cells.

- At uneven iteration $k+1$ the Δm is changed according with following equations:

$$\Delta m\left(\vec{x},t^{k+1}\right)=\begin{cases}\Delta m\left(\vec{x},t^{k}\right)-\Delta m^{\{extra\}}(\vec{x},t^{k}), & \text{if } \vec{x} \text{ is \textit{Interface} cell} \\ & \quad\text{and } \varphi^{\{test\}}(\vec{x},t^{k})-1>10^{-7} \\ & \quad\text{and } N_I(\vec{x})>0; \\ \Delta m\left(\vec{x},t^{k}\right), & \text{otherwise ;}\end{cases}$$

$$\Delta m\left(\vec{x}+R\vec{c}_i,t^{k+1}\right)=\begin{cases}\Delta m\left(\vec{x}+R\vec{c}_i,t^{k}\right)+\dfrac{\Delta m^{\{extra\}}(\vec{x},t^{k})}{N_I(\vec{x},R)}, & \text{if } \vec{x} \text{ is \textit{Interface} cell} \\ & \quad\text{and } \varphi^{\{test\}}(\vec{x},t^{k})-1>10^{-7} \\ & \quad\text{and } N_I(\vec{x})>0; \\ & \quad\text{and } \vec{c}_i\neq(0,0,0); \\ \Delta m\left(\vec{x}+R\vec{c}_i,t^{k}\right), & \text{otherwise;}\end{cases}$$

$$N_I(\vec{x},r)=\sum_i^Q\begin{cases}1 & \text{if } (\vec{x}+r\vec{c}_i) \text{ is \textit{Interface} cell and } \varphi(\vec{x}+r\vec{c}_i)<1, \vec{c}_i\neq(0,0,0); \\ 0 & \text{if } (\vec{x}+r\vec{c}_i) \text{ otherwise;}\end{cases}$$

$$R=\min_{1\le r\le 3} r|_{N_I(\vec{x},r)>0};$$

$$\Delta m^{\{extra\}}(\vec{x},t^{k})=\left(\varphi^{\{test\}}(\vec{x},t^{k})-1\right)\rho(\vec{x},t+\Delta t)= \\ \Delta m(\vec{x},t^{k})+\rho(\vec{x},t)\varphi(\vec{x},t^{k})-\rho(\vec{x},t+\Delta t);$$

$$\varphi^{\{test\}}(\vec{x},t^{k})=\frac{\rho(\vec{x},t)\varphi(\vec{x},t^{k})+\Delta m(\vec{x},t^{k})}{\rho(\vec{x},t+\Delta t)}. \tag{9}$$

At uneven iteration number the ***Flawed*** cells (i.e. *Interface* cells with $\varphi^{\{test\}} > 1+10^{-7}$) equally redistribute their $\Delta m^{\{extra\}}$ between neighbouring *Interface* cells. If no appropriate neighbouring cell are found the neighbourhood distance is increased up to maximum radius $R_{max} = 3$;

- At even iteration $k+2$ the Δm is changed according with following equations:

$$\Delta m\left(\vec{x}, t^{k+2}\right) = \begin{cases} \Delta m\left(\vec{x}, t^{k+1}\right) + \Delta m^{\{lack\}}\left(\vec{x}, t^{k+1}\right), & \text{if } \vec{x} \text{ is } \textit{Interface} \text{ cell} \\ & \quad \text{and } \varphi^{\{test\}}(\vec{x}, t^{k+1}) < -10^{-7} \\ & \quad \text{and } N_I(\vec{x}) > 0; \\ \Delta m\left(\vec{x}, t^{k+1}\right), & \text{otherwise ;} \end{cases}$$

$$\Delta m\left(\vec{x} + R\vec{c}_i, t^{k+2}\right) = \begin{cases} \Delta m\left(\vec{x} + R\vec{c}_i, tk+1\right) - \dfrac{\Delta m^{\{lack\}}\left(\vec{x}, t^{k+1}\right)}{N_I(\vec{x}, R)}, & \text{if } \vec{x} \text{ is } \textit{Interface} \text{ cell} \\ & \quad \text{and } \varphi^{\{test\}}\left(\vec{x}, t^{k+1}\right) < -10^{-7} \\ & \quad \text{and } N_I(\vec{x}) > 0; \\ & \quad \text{and } \vec{c}_i \neq (0,0,0); \\ \Delta m\left(\vec{x} + R\vec{c}_i, t^{k+1}\right), & \text{otherwise;} \end{cases}$$

$$N_I(\vec{x}, r) = \sum_i^Q \begin{cases} 1 & \text{if } (\vec{x} + r\vec{c}_i) \text{ is } \textit{Interface} \text{ cell and } \varphi(\vec{x} + r\vec{c}_i) > 0, \vec{c}_i \neq (0,0,0); \\ 0 & \text{if } (\vec{x} + r\vec{c}_i) \text{ otherwise;} \end{cases}$$

$$R = \min_{1 \le r \le 3} r|_{N_I(\vec{x}, r) > 0};$$

$$\Delta m^{\{lack\}}\left(\vec{x}, t^{k+1}\right) = \\ -\varphi^{\{test\}}\left(\vec{x}, t^{k+1}\right)\rho(\vec{x}, t + \Delta t) = -\Delta m(\vec{x}, t^{k+1}) - \rho(\vec{x}, t)\varphi(\vec{x}, t^{k+1});$$

$$\varphi^{\{test\}}(\vec{x}, t^{k+1}) = \frac{\rho(\vec{x}, t)\varphi(\vec{x}, t^{k+1}) + \Delta m(\vec{x}, t^{k+1})}{\rho(\vec{x}, t + \Delta t)}. \tag{10}$$

Similarly at even iterations the ***Flawed*** cell (*Interface* cells with $\varphi^{\{test\}} < -10^{-7}$) collect their Δm from neighbouring cells up to the neighbourhood distance $R_{max} = 3$.

- Repeat uneven and even iterations up to maximum 20 iterations or until the total number of ***Flawed*** cells > 0.

3. Changing φ in all cells (*Interface*,*Fluid* and *Void*):

$$\varphi(\vec{x}, t + \Delta t) = \begin{cases} 0 & , \text{ if } \varphi^{\{test\}}(\vec{x}, t + \Delta t) \le 10^{-7}; \\ 1 & , \text{ if } \varphi^{\{test\}}(\vec{x}, t + \Delta t) - 1 \ge -10^{-7}; \\ \varphi^{\{test\}}(\vec{x}, t + \Delta t) & , \text{ otherwise .} \end{cases}$$

$$\varphi^{\{test\}}(\vec{x}, t + \Delta t) = \begin{cases} 1 + \dfrac{\Delta m(\vec{x}, t^{k_{last\_iteration}})}{\rho(\vec{x}, t + \Delta t)} & , \text{ if } \vec{x} \text{ is } \textit{Fluid} \text{ cell ;} \\ 0 & , \text{ if } \vec{x} \text{ is } \textit{Void} \text{ cell ;} \\ \dfrac{\varphi(\vec{x}, t)\rho(\vec{x}, t) + \Delta m(\vec{x}, t^{k_{last\_iteration}})}{\rho(\vec{x}, t + \Delta t)} & , \text{ otherwise.} \end{cases} \tag{11}$$

Change cells types:

Cell type	$\varphi(\vec{x}, t+\Delta t)$	Cell enthalpy E	New Cell type
Interface	$=1$		*Fluid*
Interface	$=0$		*Void*
Fluid	$\neq 1$		*Interface*
Void	$\neq 0$		*Interface*
Solid		$E > E(T_{liquidus})$	*Fluid*
Fluid or *Interface*		$E < E(T_{solidus})$ and there are adjacent *Solid* cells or $E < E(T_{uncsolidus})$	*Solid* (*Remelted* sub-type)

Table 1: Cell types change conditions

After that all Δm in all cells is set to zero, except the cells of *Void* type with $\Delta m \neq 0$, because it is not possible to evaluate correct $\varphi(\vec{x}, t+\Delta t)$ there due to unknown $\rho(\vec{x}, t+\Delta t)$ values. The updated value of $\rho(\vec{x}, t+\Delta t)$ (as like as all DFs f_i) in such cells will be extrapolated from the neighbouring cells only after memory allocation and initialization, and the new $\varphi(\vec{x}, t+2\Delta t)$ will be calculated further at the next step.

4. Fixing cell types. The algorithm finds the "bad" *Fluid* cells with adjacent *Void* cells and changes their types from *Fluid* to *Interface*. Besides that additionally at the same time this algorithm also carried out the special sub-types *Interface* **F**,**V** and **I** required by anti-diffusion algorithm (see above).

3.2.2 LBM boundary conditions

The usual streaming step $f_i(\vec{x} + \vec{c}_i\Delta t, t+\Delta t) = f_i(\vec{x}, t+\Delta t)$ is not correct if the coordinate $(\vec{x} + \vec{c}_i)$ falls into the *Void* cell type. The pressure boundary conditions are applied in this case:

$$
\begin{aligned}
&f_{\bar{i}^r}(\vec{x}, t+\Delta t) = (f_i^{eq}(\rho_{ext}, \vec{u}(\vec{x}, t)) + f_{\bar{i}^r}(\rho_{ext}, \vec{u}(\vec{x}, t))) - f_i(\vec{x}, t), \\
&\rho_{ext} = \rho_0 + \frac{P_l + P_{ev}}{c_s^2}.
\end{aligned} \tag{12}
$$

where f_i^{eq} is evaluated from eq. (2), ρ_0 is quasi-equilibrium density of fluid, additional pressure caused by Laplace pressure P_l and evaporation recoil pressure P_{ev}(see sec. 3.2.6).

The Laplace pressure can be estimated with the Young–Laplace equation:

$$P_a = \sigma\kappa, \tag{13}$$

where σ is the surface tension and κ is the mean curvature of surface.

The bounce-back boundary condition [14, 11] is applied near solid boundaries:

$$f_{\bar{i}^r}(\vec{x}, t+\Delta t) = f_i(\vec{x}, t) \quad \text{if } (\vec{x} + \vec{c} + i) \text{ falls into } \textit{Solid} \text{ cell type.} \tag{14}$$

For the distribution functions h_i from eq. (3) used for heat transfer in liquid the following boundary conditions at the free surface is applied:

$$h_{\bar{i}^r}(\vec{x}, t+\Delta t) = h_i(\vec{x}, t) + \frac{E}{\rho_{ext}}(f_{\bar{i}^r}(\vec{x}, t+\Delta t) - f_i(\vec{x}, t)) \tag{15}$$

if $(\vec{x} + \vec{c}_i)$ falls into the *Void* cell type. The value of $f_{\bar{i}^r}(\vec{x}, t+\Delta t)$ is obtained from eq. (12) and $E = \sum_i h_i(\vec{x}, t)$ is the energy in the considered cell $\vec{x}$. If the cell $\vec{x}$ is *Solid* than the boundary condition for $\vec{h}_i$ degenerates into the simple bounce-back.

3.2.3 Curvature, wetting and surface normal calculation

Curvature is evaluated only at *Interface* cells by help of the "template sphere" approach [5]. The direction of surface normal is also calculated in some *Solid* cells, which has *Void* cells among adjacent cells.

Let the interface be given by the sphere with radius R, i.e. the interface curvature is given by $\kappa = 2/R$. The template sphere has the radius r and center M_r. Considering a general case, the interface does not pass through the center M_r. The distance is labeled δ, so the portion of non-fluid void in the template sphere is labeled V(see fig. 2). The curvature should be expressed as a function of r, δ, and V .

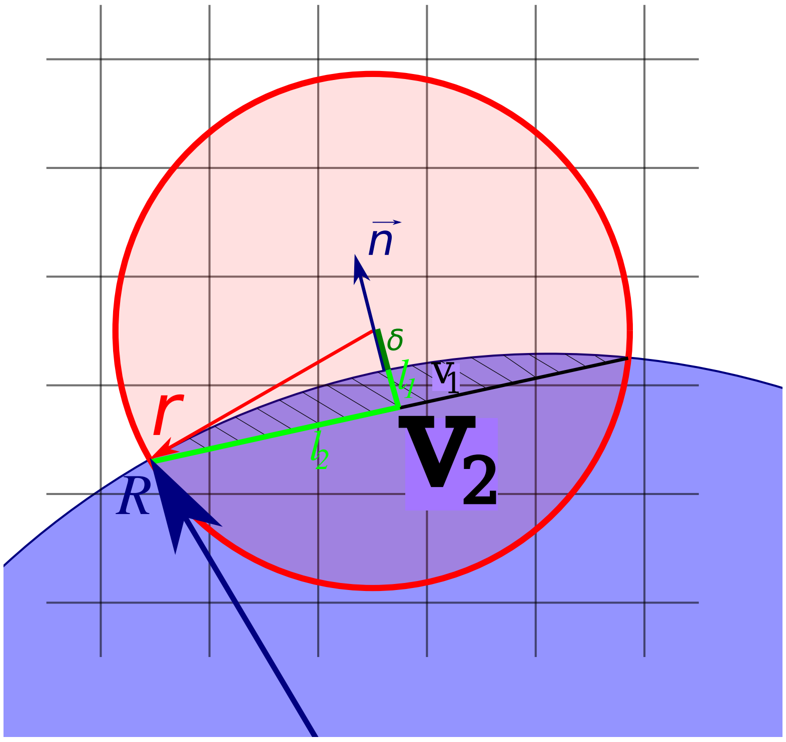

Figure 2: Geometric scheme for curvature calculation.

After some math manipulations (see appendix), the radius R can be obtained:

$$R = \frac{3(r^2-\delta^2)^2}{4(2r^3-3r^2\delta+\delta^3)-12V/\pi} - \delta \tag{16}$$

and $\kappa = 2/R$. The volume V cannot be evaluated accurately because of the discrete nature of grid so it is calculated according with following formula:

$$\begin{gathered} V = \sum_{\substack{-L\leqslant p_x\leqslant +L\\ -L\leqslant p_y\leqslant +L\\ -L\leqslant p_z\leqslant +L}} w\left(|p_x|,|p_y|,|p_z|\right) k_{wet}(\vec{x}_c+\vec{p})\varphi(\vec{x}_c+\vec{p}), \\ k_{wet}(\vec{x}) = \begin{cases} 1-\dfrac{\theta^{eq}}{180^\circ} & \text{if cell } \vec{x} \text{ is type } \textit{Solid} \text{ but not } \textit{Remelted} \text{ sub-type;} \\ 1 & \text{otherwise;} \end{cases} \end{gathered} \tag{17}$$

where L is the stencil half size, **KiSSAM** uses $L = 4$ (in fig. 2 $L = 2$), w are the weights dependent on the distance from point $(\vec{x}_c + \vec{p})$ to the center of template sphere $\vec{x}_c$. These weights are constants pre-calculated as the volume of sphere-cube intersection (with sphere of radius r and center at $(0, 0, 0)$ and cube is the cell $\vec{p} \pm (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$).

The template sphere radius r is $\sqrt[3]{\frac{3}{4\pi}389} \approx 4.5285$, where the number 389 is the total number of weights w with value > 0.5 for $L = 4$.

The wetting coefficient k_{wet} simulates the effect of the tendency to the equilibrium angle θ^{eq} when the liquid surface get close to the solid surface. Note that **KiSSAM** uses three types of equilibrium angle θ^{eq}: for solid substrate wetting, solid powder particles wetting and solid remelted material (e.g. wetting of solidified meltpool, this value is usually $\theta^{eq} = 0°$ to get smooth solidification).

Volume φ of fluid inside the cell is known at the each time step, and the distance δ between the fluid interface and the center of the considered interface cell is calculated from the following approximate formula of the position of the interface:

$$\delta = \frac{1 - 2\varphi}{2\max\{n_i\}}, \tag{18}$$

where n_i is the component of the normal vector to the interface at this point, which is calculated at the each time step (see below). The expression (18) can be obtained from the exact equations (e.g., see [13]). Another approximation is proposed in [7]. Thus, with formulas (16) and (18) the approximation of the curvature is obtained for every *Interface* cell. The analysis of the error produced by this approach is given in the Appendix. The approximation appears to be quite accurate, giving the mean curvature with the relative error less then 2%.

The unit normal vector $\vec{n}(\vec{x}_c)$ in the cell $\vec{x}_c$ is calculated based on the φ of the neighbor cells:

$$\vec{n}'(\vec{x}_c) = \sum_{\vec{x} \in C_{norm}} (1 - \varphi(\vec{x}))\,(\vec{x} - \vec{x}_c);$$

$$C_{norm} = \vec{x}|_{\vec{x} = \vec{x}_c + (i,j,k)^T, i,j,k \in \{0, \pm 1\}};$$

$$\vec{n}(\vec{x}_c) = \frac{\vec{n}'(\vec{x}_c)}{|\vec{n}'(\vec{x}_c)|}.$$

3.2.4 Marangoni force

High temperature gradients along the liquid metal surface lead to Marangoni convection. This tangential Marangoni force acts along the surface tension gradient, and the following equation should be satisfied for force $\vec{F}$:

$$\vec{F} \cdot \vec{\tau} = \nabla\sigma(\vec{x}, T) \cdot \vec{\tau}\Delta S,$$

where $\vec{\tau}$ is the unit tangential vector, $\sigma(\vec{x}, T)$ is the temperature-dependent surface tension, and ΔS is the unit surface area equals to 1 in LBM units. We note here that the recoil pressure and the Laplace pressure are included in the pressure drop at the fluid/gas interface boundary condition, while the Marangoni effect acts as a shear force.

The force $\vec{F}$ for equation (1) is computed as

$$\vec{F}(\vec{x}) = \nabla\sigma(\vec{x}, T) - (\nabla\sigma(\vec{x}, T) \cdot \vec{n})\,\vec{n}, \tag{19}$$

where $\vec{n}$ is the unit normal vector (how it is calculated see in section 3.2.3), and the surface tension gradient $\nabla\sigma$ is calculated with finite-difference scheme.

3.2.5 Drag force in mushy zone

KiSSAM additionally takes into account Darcy's damping force due to the flow of liquid metal through the mushy zone represented as a porous medium of permeability K [3]:

$$\vec{F} = -\mu K \vec{u} \rho, \tag{20}$$

where μ is the fluid viscosity, $\vec{u}$ is the fluid velocity and K can be obtained through the Carman–Kozeny equation derived from the Darcy model by assuming that the mushy zone can be regarded as a porous medium [2]:

$$K = \frac{180}{D^2} \left(\frac{1}{f_l} - 1 \right)^2 \frac{1}{f_l}, \tag{21}$$

where f_l is the liquid fraction in mushy zone, assuming it linearly dependent from temperature:

$$f_l = \min \left(1, \max \left(10^{-5}, \frac{T - T_{solidus}}{T_{liquidus} - T_{solidus}} \right) \right). \tag{22}$$

The parameter D has the meaning of the characteristics length approximately equal to the primary dendrite arm spacing (PDAS).

3.2.6 Evaporation solver

Evaporation solver follows the Knight and Klassen evaporation model [9, 8] to estimate the energy and mass carried away by the vapor and to evaluate the recoil pressure force. In this model, the additional pressure P_v, energy loss J_v and mass loss M_v are introduced at the surface of the fluid with temperature T_{surf}:

$$\begin{aligned}
&P_v = \max \left(\frac{P_{sat}(T_{surf})}{2} \left(1 + \frac{1-\psi}{2} \left(1 + \frac{\rho_{Kn}}{\rho_{surf}} \frac{T_{Kn}}{T_{surf}} \right) \right) - P_{atm}, 0 \right), \\
&M_v = -\psi j_+ \cdot \Delta t \Delta x^2, \\
&J_v = M_v \left(L_{vap}(T_{surf}) + E(T_{surf})/\rho \right), \\
&j_+ = P_{sat}(T_{surf}) \sqrt{\frac{m_a}{2\pi k_B T_{surf}}}, \\
&\psi = \sqrt{2\pi\gamma} M \frac{\rho_{Kn}}{\rho_{surf}} \sqrt{\frac{T_{Kn}}{T_{surf}}},
\end{aligned} \tag{23}$$

where ψ is the evaporation coefficient, $\dfrac{\rho_{Kn}}{\rho_{surf}}$ and $\dfrac{T_{Kn}}{T_{surf}}$ is the vapor density and temperature jump across the Knudsen layer:

$$\begin{aligned}
\sqrt{\frac{T_{Kn}}{T_{surf}}} &= \sqrt{1+\pi\left(\frac{1}{2}\frac{(\gamma-1)}{(\gamma+1)}\mathcal{M}_{Kn}\right)^2} - \frac{\sqrt{\pi}}{2}\frac{(\gamma-1)}{(\gamma+1)}\mathcal{M}_{Kn}, \\
\frac{\rho_{Kn}}{\rho_{surf}} &= \sqrt{\frac{T_{surf}}{T_{Kn}}}\left(\frac{1}{2}(2\mathcal{M}_{Kn}^2+1)\exp\left(\mathcal{M}_{Kn}^2\right)\mathrm{erfc}(\mathcal{M}_{Kn}) - \frac{\mathcal{M}_{Kn}}{\sqrt{\pi}}\right) + \\
&\qquad \frac{1}{2}\frac{T_{surf}}{T_{Kn}}\left(1-\sqrt{\pi}\mathcal{M}_{Kn}\exp\left(\mathcal{M}_{Kn}^2\right)\mathrm{erfc}\left(\mathcal{M}_{Kn}\right)\right), \\
\mathcal{M}_{Kn} &= M\sqrt{\frac{\gamma}{2}},
\end{aligned} \tag{24}$$

with M is Mach number for evaporated material near the surface and γ is the vapor adiabatic index (usually metals vapors are monoatomic with $\gamma = 5/3$), m_a is the vapor atomic mass, k_B is the Boltzmann constant, Δt and Δx are space and time steps (equals to 1 in LBM units), $E(T)/\rho$ is enthalpy density at temperature T (see eq. (4)), $L_{vap}(T)$ is the latent heat of vaporization, $P_{sat}(T)$ is the saturated vapor pressure at temperature T, P_atm is the pressure of ambient atmosphere.

The saturated vapor pressure can be can be found with different models in **KiSSAM**: *Clausius-Clapeyron*, *Antonie* and *Antonie Extended*.

For *Clausius-Clapeyron* model [16, 8]:

$$\begin{aligned}
P_{sat}(T)\ [\text{bars}] &= \exp\left\{-m_a\frac{L_{vap0}}{k_B}\left(\frac{1}{T}\sqrt{1-\left(\frac{T}{T_{crit}}\right)^2} - \frac{1}{T_{boil}}\sqrt{1-\left(\frac{T_{boil}}{T_{crit}}\right)^2} + \right.\right. \\
&\qquad \left.\left.\frac{1}{T_{crit}}\left(\arcsin\left(\frac{T}{T_{crit}}\right) - \arcsin\left(\frac{T_{boil}}{T_{crit}}\right)\right)\right)\right\}; \\
L_{vap}(T) &= L_{vap0}\sqrt{1-\left(\frac{T}{T_{crit}}\right)^2}, \\
L_{vap0}(T) &= L_{vap_B}/\sqrt{1-\left(\frac{T_{boil}}{T_{crit}}\right)^2},
\end{aligned} \tag{25}$$

where L_{vap0} is the specific heat of vaporization at absolute zero temperature (it can be carried out from L_{vap_B} – the heat of vaporization at boiling point which is known more often), T_{boil} is the boiling temperature at standard atmosphere (1 bar) and T_{crit} is the critical temperature.

For *Antonie* model:

$$\begin{aligned}
P_{sat}(T)\ [\text{Pascal}] &= 10^{\mathcal{A}-\mathcal{B}/(\mathcal{C}+T)}, \\
L_{vap}(T) &= \frac{\mathcal{B}\ln 10}{(\mathcal{C}+T)^2}\cdot\frac{k_B T^2}{m_a};
\end{aligned} \tag{26}$$

and for *Antonie Extended*:

$$
\begin{aligned}
P_{sat}(T)\ [\text{Pascal}] &= 10^{\mathcal{A}-\mathcal{B}/(\mathcal{C}+T)+\mathcal{D}T+\mathcal{E}T^2+F\log_{10}(T)},\\
L_{vap}(T) &= \frac{k_B T^2}{m_a}\cdot\left(\frac{\mathcal{B}}{(\mathcal{C}+T)^2}+\mathcal{D}+2\mathcal{E}+\frac{\mathcal{F}}{T\ln 10}\right)\ln 10.
\end{aligned}
\tag{27}
$$

The coefficients $\mathcal{A}$,$\mathcal{B}$,$\mathcal{C}$ usually can be found from any reference book with saturated vapor data (for example see https://en.wikipedia.org/wiki/Vapor_pressures_of_the_elements_%28data_page%29). Additional coefficients $\mathcal{D}$,$\mathcal{E}$,$\mathcal{F}$ can be used for more precise fitting of saturated vapor curves.

The recommended model in **KiSSAM** is *Antonie* model.

For eq. (23) it is still not clear how to found Mach number M. We use three regimes for evaluation the M: diffusive evaporation when $M = 0$, moderate evaporation (when $0 < M < 1$) and intensive sonic evaporation with $M = 1$:

$$
M = \begin{cases} 0, & \text{if } T_{surf} \leqslant T_{lim_0} \text{ or } P_{sat} < P_{atm};\\ 1, & \text{if } T_{surf} \geqslant T_{lim_1};\\ \dfrac{T_{surf}-T_{lim_0}}{T_{lim_1}-T_{lim_0}}, & \text{otherwise.} \end{cases}
\tag{28}
$$

The crucial points T_{lim_0} and T_{lim_1} are calculated once as the temperature when saturated vapor pressure equals to ambient pressure, and the temperature when the evaporation becomes sonic. In other words T_{lim_0} is the root of the equation $P_{sat}(T_{lim_0}) = P_{atm}$ and T_{lim_1} is the root of the equation

$$
P_{sat}(T_{lim_1}) = P_{atm}\cdot P_{rel}\left(T = T_{lim_1}, M = 1\right),
\tag{29}
$$

where $P_{rel}(T, M)$ is the pressures relation, which can be calculated from the Rankine-Hugoniot shock relations [9] (see appendix).

Additionally in **KiSSAM** for stability purpose limit P_v,M_v and J_v are limited with high absolute values.

3.2.7 Radiation cooling

Radiation cooling of liquid surface with temperature T is expressed as additional energy losses equal to $-\sigma_{SB}T^4$, where σ_{SB} is the Stefan-Boltzmann constant.

3.2.8 Convective cooling

One more reason for cooling is the energy loses due to forced convection, when the ambient gas is forced to flow over a meltpool or liquid droplets by external means such as a pump in camera. If the local curvature radius of the surface (either meltpool or liquid droplet) is R, then the cooling flux equals to

$$
E_{cc} = -Nu\frac{k}{2R}\left(T - T_{ref}\right),
\tag{30}
$$

where Nu is the Nusselt number, k is the gas thermal conductivity, $T_{ref} = 300\ K$ is the reference temperature T is the local temperature of the surface.

To estimate Nusselt number **KiSSAM** uses Whitaker model for flow over the sphere [15]:

$$
Nu = 2 + \left(0.4Re^{1/2} + 0.06Re^{2/3}\right)Pr^{0.4}\left(\mu_\infty/\mu_s\right)^{1/4},
\tag{31}
$$

where Reynolds number $Re = \dfrac{2R\rho_{gas}V_{gas}}{\mu_\infty}$, Pr is the gas Prandtl number, μ_∞ is the gas viscosity at ambient temperature, ρ_{gas} is the gas density, V_{gas} is the camera gas absolute flow velocity, μ_s is the value of the gas viscosity at the surface temperature.

The value of μ_s can be carried out from the Sutherland's law:

$$\mu_s = \mu_{ref}\left(\frac{T}{T_{ref}}\right)^{3/2} \cdot \frac{T_{ref}+T_\mu}{T+T_\mu}, \tag{32}$$

where μ_{ref} is the gas reference viscosity ($= \mu_\infty$) and T_μ is an effective temperature (the Sutherland constant).

3.2.9 Bubbles tracking solver

By default the vacuum is assumed in any *Void* cells and regions where the gas pressure is 0. But it is possible to enable the algorithm of bubbles tracking (flag `config.calcBubbles` in *input.json* file). This algorithm of Connected-component labeling [4] at every iteration finds all enclosed regions of connected cells with filling fraction of $\varphi < 1$ (*Void*,*Interfaces* and some *Solid* cells).

TODO

3.2.10 Gas solver

Now the gas and vapour are not tracked in camera and their flow is not simulated.

3.2.11 Heat solver

In **Tractile Mesh** only Solid or Void regions exist and heat equation is solved in Solid region:

$$\frac{\partial E(T)}{\partial t} = \nabla \cdot \left(k(T)\nabla E(T)\right), \tag{33}$$

where $k(T)$ is temperature-dependent thermal diffusivity, $E(T)$ is the enthalpy. At the Solid-Void boundary and on the boundaries of simulation domain adiabatic boundary conditions are set.

The cells in **Tractile Mesh** with coordinates (x_1, y, z) and (x_2, y, z) are possibly merged if maximum temperature difference between cells (x_1, y_a, z_a) and (x_2, y_a, z_a) for any y_a, z_a is less than 10 K. Also the cell is split if the temperature difference between neighbouring cells is less than 10 K. The same criteria works for all coordinates.

3.2.12 Ray tracing

Laser Beam
For $\sim 1\mu m$ wavelength lasers typically used in L-PBF, the laser energy is absorbed in a thin skin layer of the metal ($< 100\ nm$). Therefore **KiSSAM** implements the laser energy deposition as a heat flux boundary condition for every ray being reflected from the liquid or solid material surface in the corresponding cell. To model multiple reflections, the surface normal at each cell at the material interface is used (see sec. 3.2.3). For every ray we track its propagation with an unlimited number of reflections until the energy of the ray reaches a given small threshold value.

The absorption coefficient (the part of ray power absorbed after the single reflection) is depend on the local temperature of the cell.

The laser ray polarization and defocusing is neglected now.

Electron Beam

The electron beam energy deposition behaves differently from the laser beam. For the electron beam, the similar ray tracing method is used, although the rays are not reflected from the surface and can penetrate into the solid or liquid metal. We trace each ray as it is scattered inside the material, reducing its kinetic energy along the path. This energy is deposited as a volume heat source at the corresponding cells for LBM temperature model.

The Monte-Carlo model simulates electrons trajectories by considering both elastic and inelastic scattering processes. In the energy range of 30–120 keV for the materials of interest (with relatively large atomic number), we can separately consider the nearly elastic scattering of the electrons by atomic nuclei and its energy loss (attenuation) via the scattering by the electrons in the atomic shells.

The nearly elastic scattering is governed by the nuclear scattering cross-section. For a nucleus with the charge number Z, in the approximation of the screened Rutherford scattering, the single differential scattering cross-section has the form [12]:

$$\frac{d\sigma(\theta)}{d\Sigma} = \frac{Z(Z+1)e^4}{p^2v^2}\frac{1}{(1+\cos\theta+2\beta)^2}, \tag{34}$$

where v,p,θ are the electron velocity, momentum, and scattering angle respectively, e is the elementary charge, and β is the screening parameter:

$$\beta = \left(\frac{\hbar Z^{1/3}}{pa_B}\right)^2, \tag{35}$$

where a_B is the Bohr radius and $\hbar$ is the reduced Planck's constant.

The scattering length Λ is determined by the total scattering cross-section and the material density ρ:

$$\Lambda = \frac{2\beta(1+\beta)p^2v^2A}{2\pi N_A\rho Z(Z+1)e^4}, \tag{36}$$

where A is the atomic weight, and N_A is the Avogadro number.

The differential energy loss (stopping power) due to the electron-electron scattering is computed as in [6]:

$$\frac{dE}{ds} = -785\frac{\rho Z}{AE}\ln\left(\frac{1.166E}{J}\right) [\text{eV/Å}], \tag{37}$$

where E is the instantaneous electron energy in eV, s is the path traveled in the material in Å, ρ is the density given in g/cm3, and $J = 10Z$ eV is the mean ionization potential of the atom.

In the Monte-Carlo model, the electron trajectory in the material is approximated by a series of straight line segments with a randomly chosen length L_r, which is chosen based on the scattering length (36):

$$L_r = -\Lambda\log R_1, \tag{38}$$

where R_1 is a random number uniformly distributed in the interval between 0 and 1.

The polar angle of collision θ is determined by solving the equation:

$$R_2 = \frac{\int\limits_0^{\theta} \frac{d\sigma(\theta')}{d\theta'} \sin\theta' d\theta'}{\int\limits_0^{\pi} \frac{d\sigma(\theta')}{d\theta'} \sin\theta' d\theta'}, \tag{39}$$

where R_2 is a random number uniformly distributed in the interval between 0 and 1. The azimuthal angle of collision is chosen randomly in the interval between 0 and 2π.

The energy loss is modeled by reducing the electron energy at each line segment between two collisions i and $i+1$ in the following manner:

$$E_{i+1} = E_i + L_r \frac{dE}{ds}, \tag{40}$$

where L_r is determined from eq. (38). The deposited energy $L_r \frac{dE}{ds}$ is divided among the lattice cells corresponding to the given line segment.

3.3 Sensors

Radiation sensors (photo-detectors) can be specified to collect emission from the melted and solid area. The collection scheme of one sensor is shown in the fig. 3.

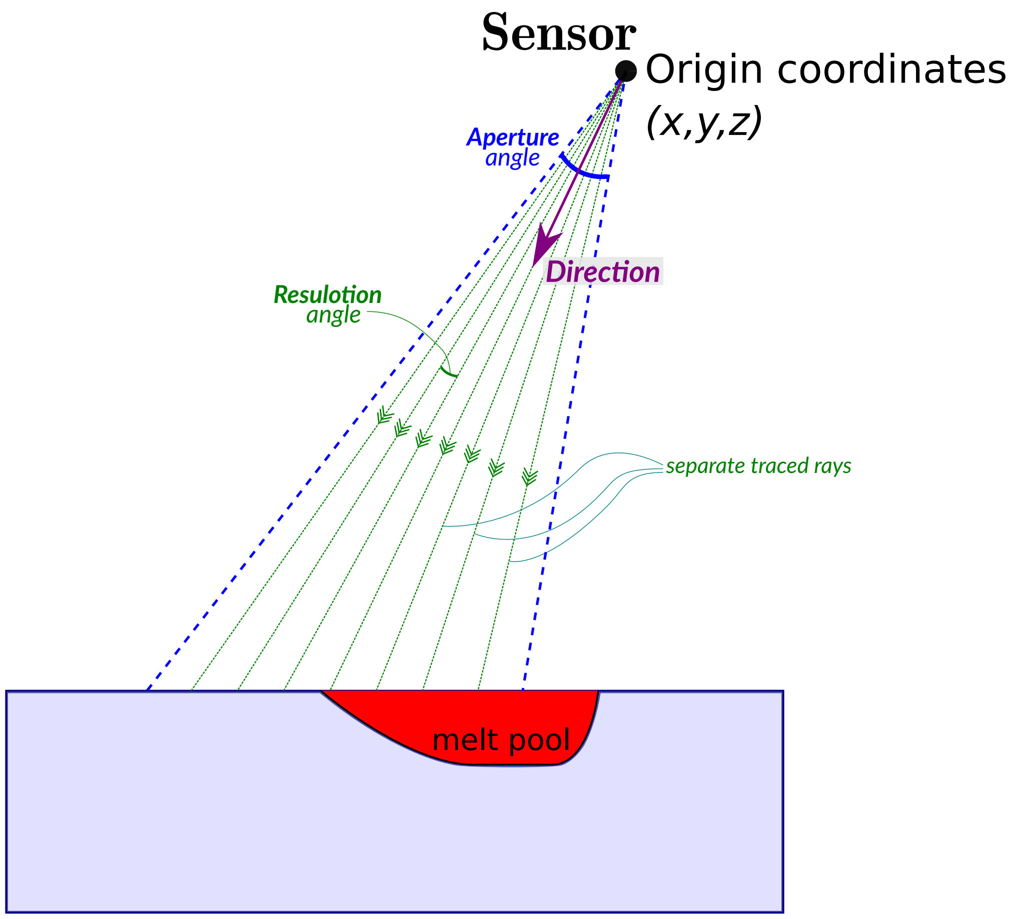

Figure 3: The sensor (photo-detector) design **KiSSAM**

Each sensor is a point with coordinates *(x,y,z)* (origin), *Direction* vector, *Aperture* view angle, *Resolution* angle between individual rays traced from the origin point inside the cone and *Wavelength band* response. Each separate ray inside the sensor's cone is traced until the collision point with the meltpool or solid surface and the emission power of this point is calculated from temperature (in accordance with Plank's law) and collected by the sensor. So the sensors signal is calculated at every time step with the following formula:

$$\text{Emission energy [J]} = \sum_i \Delta t \varepsilon \frac{\pi |\vec{r}_i|^2 \beta^2}{4} \int\limits_{\lambda_1}^{\lambda_2} \frac{2hc^2}{\lambda^5 \left(e^{\frac{hc}{\lambda k_B T_i}} - 1 \right)} d\lambda, \tag{41}$$

where Δt is time step, $\vec{r}_i = \vec{p}_i - \vec{s}_i$ is a ray vector pointed from the sensor's origin point $\vec{s}_i$ to the ray's intersection point with the surface $\vec{p}_i$, T_i is the surface temperature of the point $\vec{p}_i$, β is the sensor's resolution angle, h, c, k_B are Plank's constant, light speed and Boltzmann constant, and $[\lambda_1, \lambda_2]$ is the wavelength band of the sensor's response. Emissivity constant ε is assumed as 1 now. The sum $\sum\limits_i$ is accumulated over the full set of the rays back-traced from the sensor as shown in the fig. 3.

See the User Guide below (section **??**) to study how to specify individual sensors parameters.

4 Supplementary instructions (to User guide and tutorial)

4.1 Recommended parameters sets

4.1.1 Space and time steps

LBM generally has no strictly defined stability criteria. Standard BGK LBM usually stable if Reynolds number Re $<$ 1000 and Mach number $Ma \lesssim 0.3$. Here the Mach number not the phisical value but the following value: $Ma = \frac{u}{c_s} = \frac{u\sqrt{3}\Delta t}{\Delta x}$. LBM also has the second order accuracy over the Mach number, so it must be as low as possible. So finally we have two restrictions when space and time steps (Δx and Δt) have to be chosen:

$$\begin{aligned} &1000 > Re = \frac{uL}{\nu} \overset{\substack{\text{convert to}\\\text{dimensionless}\\\text{LBM units}}}{\approx} \frac{c_s Ma \cdot 10\Delta x}{(\tau - 1/2)c_s^2} = \frac{Ma \cdot 10\Delta t\sqrt{3}}{\tau - 1/2} \lesssim \frac{10}{\tau - 1/2} < 1000 \\ &Ma = \frac{u\Delta t}{\Delta x} \ll 1 \end{aligned} \tag{42}$$

First condition requires the relaxation parameter τ should be more than 0.51, i.e. $3\hat{\nu} + 0.5 = \frac{3\nu\Delta t}{\Delta x^2} + 0.5 > 0.51$, where $\hat{\nu}$ is kinematic viscosity in numerical LBM units and ν is the kinematic viscosity in physical units (SI). So finally we got the following simple condition:

$$\Delta t \gtrsim 0.003\frac{\Delta x^2}{\nu}. \tag{43}$$

From the second condition $Ma \ll 1$ from eq (42) and assuming that typical velocities in melt-pool are about 1 m/s we got one more condition:

$$\Delta t \ll \frac{\Delta x}{1m/s}. \tag{44}$$

And the third arrangement also must be satisfied — the CFL condition for heat solver:

$$\Delta t < \frac{1}{2}\frac{\Delta x^2}{\kappa}, \tag{45}$$

where κ is the thermal diffusivity of material in solid and low-temperature liquid phases.

Combining this the time step Δt should be chosen after the space step Δx as

$$\Delta t = \min\left(\frac{0.003\Delta x^2}{\nu}, \quad \frac{\Delta x^2}{2\kappa_{max}}\right) \tag{46}$$

where κ_{max} is the maximum value of thermal diffusivity for temperature ranges $T < 1.1T_l$ (T_l is a liquidus temperature).

The recommended space/time steps parameters set for the most common materials are following:

$\Delta x = 2\ \mu m$	$\Delta t = 15$ ns
$\Delta x = 3\ \mu m$	$\Delta t = 30$ ns
$\Delta x = 4\ \mu m$	$\Delta t = 50$ ns
$\Delta x = 5\ \mu m$	$\Delta t = 80$ ns
$\Delta x = 6\ \mu m$	$\Delta t = 120$ ns
$\Delta x = 7\ \mu m$	$\Delta t = 160$ ns

Figure 4: Recommended space and time steps for most common metals and alloys. Nevertheless don't forget to check the condition (46)

For laser beam melting recommended space step is $\Delta r = 3\ \mu m$ (`dr=3e-6`) and time step is $\Delta t = 30\ ns$ (`dt=30e-9`).
For electron beam melting recommended space step is $\Delta r = 5\ \mu m$ (`dr=5e-6`) and time step is $\Delta t = 80\ ns$ (`dt=80e-9`).

References

[1] Elham Attar and Carolin Körner. "Lattice Boltzmann model for thermal free surface flows with liquid–solid phase transition". In: *International Journal of Heat and Fluid Flow* 32.1 (2011), pp. 156–163.

[2] Phillip C Carman. "Fluid flow through granular beds". In: *Chemical Engineering Research and Design* 75 (1997), S32–S48.

[3] Jonathan A Dantzig and Michel Rappaz. *Solidification: -Revised & Expanded*. EPFL press, 2016.

[4] Michael B. Dillencourt, Hanan Samet, and Markku Tamminen. "A General Approach to Connected-Component Labeling for Arbitrary Image Representations". In: *J. ACM* 39.2 (Apr. 1992), pp. 253–280. ISSN: 0004-5411. DOI: 10.1145/128749.128750. URL: https://doi.org/10.1145/128749.128750.

[5] E. Attar. "Simulation of selective electron beam melting processes". In: (2011).

[6] DC Joy and Suichu Luo. "An empirical stopping power relationship for low-energy electrons". In: *Scanning* 11.4 (1989), pp. 176–180.

[7] Akio Kawano. "A simple volume-of-fluid reconstruction method for three-dimensional two-phase flows". In: *Computers Fluids* 134-135 (2016), pp. 130–145. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2016.05.014. URL: https://www.sciencedirect.com/science/article/pii/S0045793016301608.

[8] Alexander Klassen. *Simulation of evaporation phenomena in selective electron beam melting*. FAU University Press, 2018.

[9] Charles J Knight. "Theoretical modeling of rapid surface vaporization with back pressure". In: *AIAA journal* 17.5 (1979), pp. 519–523.

[10] M. Thies. "Lattice Boltzmann modeling with free surfaces applied to in-situ gas generated foam formation". In: (2005).

[11] Sukop M.C. and Thorne D.T. *Lattice Boltzmann modeling*. Springer, 2006.

[12] Kenji Murata, Takayuki Matsukawa, and Ryuichi Shimizu. "Monte Carlo calculations on electron scattering in a solid target". In: *Japanese Journal of Applied Physics* 10.6 (1971), p. 678.

[13] Ruben Scardovelli and Stephane Zaleski. “Analytical Relations Connecting Linear Interfaces and Volume Fractions in Rectangular Grids”. In: *Journal of Computational Physics* 164.1 (2000), pp. 228–237. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2000.6567. URL: https://www.sciencedirect.com/science/article/pii/S0021999100965677.

[14] Sauro Succi. *The lattice Boltzmann equation: for fluid dynamics and beyond*. Oxford university press, 2001.

[15] Stephen Whitaker. “Forced convection heat transfer correlations for flow in pipes, past flat plates, single cylinders, single spheres, and for flow in packed beds and tube bundles”. In: *AIChE Journal* 18 (Mar. 1972), pp. 361–371. DOI: 10.1002/aic.690180219.

[16] BS Yilbas. “Laser heating process and experimental validation”. In: *International journal of heat and mass transfer* 40.5 (1997), pp. 1131–1143.

Appendices

A Curvature calculation

A.1 Curvature radius formula

The volume V is given by intersection of the template sphere and the fluid domain as described in Figure 2.

$$V = V_1 + V_2,$$

where

$$V_1 = \frac{1}{3}\pi x^2(3R - x);$$

$$V_2 = \frac{1}{3}\pi(r - \delta - x)^2(3r - (r - \delta - x))$$

are volumes of the spherical caps which compose the V.

By the similarity of the triangles

$$\frac{l_1}{l_2} = \frac{l_2}{2R - l_1},$$

where $l_1,\ l_2$ are as labeled in figure 2. Also by the Pythagoras theorem we have

$$r^2 = l_2^2 + (\delta + l_1)^2$$

and excluding l_2 we find

$$l_1 = \frac{r^2 - \delta^2}{2(R + \delta)}.$$

Transforming the expression for the volume we get

$$V = \frac{\pi}{3}\left[3l_1^2R - l_1^3 + (r^2 + \delta^2 + l_1^2 - 2r\delta - 2rl_1 + 2\delta l_1)(2r + \delta + l_1)\right] =$$

$$= \frac{\pi}{3}\left[2r^3 - 3r^2\delta + \delta^3 + 3(\delta^2 - r^2)l_1 + 3(\delta + R)l_1^2\right].$$

After inserting l_1 in this equation for V we obtain

$$\frac{3V}{\pi} = 2r^3 - 3r^2\delta + \delta^3 - \frac{3(r^2-\delta^2)^2}{4(R+\delta)}.$$

Simplifying the above we can derive the radius R as follows,

$$R = \frac{3(r^2-\delta^2)^2}{4(2r^3 - 3r^2\delta + \delta^3) - 12V/\pi} - \delta \tag{47}$$

and $\kappa = 2/R$.

A.2 Curvature approximation formula error

To understand how rude approximation the formula (16) is, one could remind that it is divided under the assumption that the surface is close to the sphere. In the common case, the surface has two main curvatures R_x and R_y and $\kappa = \frac{1}{R_x} + \frac{1}{R_y}$. The formula is obtained for $R_x = R_y$. Due to the symmetry of the formula, it will produce correct result $\kappa = 0$ when $R_x = -R_y$. So we believe that the error of the formula 16 will be the biggest when one of the $R_i = 0$, which corresponds to the case when the surface is a cylinder.

Let us consider again the figure 2. Now the volume of intersection can be calculated the following way

$$V = 2\int_0^{r_0} S(x)dx,$$

where

$$r(x) = \sqrt{r_0^2 - x^2},$$

while $\delta = 0$ is considered for simplicity.

$$S(x) = r^2(x)\arccos(\frac{r(x)}{2R}) + R^2\arccos(1 - \frac{r^2(x)}{2R^2}) - \frac{r(x)}{2}\sqrt{4R^2 - r^2(x)}$$

as the area of two circles intersection.

Performing Taylor expansion up to $o(r^3)$ for $S(x)$ we get

$$r^2(x)\arccos(\frac{r(x)}{2R}) = r^2(x)(\pi/2 - \frac{r(x)}{2R}) + o(r^3).$$

Using the formula $\arccos(1-x) = \sqrt{2x} + \frac{(2x)^{3/2}}{24} + o(x^2)$ we get

$$R^2\arccos(1 - \frac{r^2(x)}{2R^2}) = r(x)R + \frac{r^3(x)}{24R} + o(r^3)$$

and for the last member

$$\frac{r(x)}{2}\sqrt{4R^2 - r^2(x)} = r(x)R - \frac{r^3(x)}{8R} + o(r^3).$$

So

$$S(x) = r^2(x)(\pi/2 - \frac{r(x)}{2R}) + \frac{r^3(x)}{6R} + o(r^3) = \pi/2r^2(x) - \frac{r^3(x)}{3R} + o(r^3).$$

Using expression for $r(x)$ and the equality

$$r(x) = r_0 - \frac{x^2}{2r_0} + o(x^3)$$

we derive

$$S(x) = \pi/2(r_0^2 - x^2) - \frac{(r_0^2 - x^2)(r_0 - \frac{x^2}{2r_0})}{3R} + o(x^3).$$

Simplifying this we obtain

$$S(x) = \pi/2(r_0^2 - x^2) - \frac{1}{3R}(r_0^3 - 3/2x^2 r_0 + \frac{x^4}{2r_0}) + o(x^3).$$

So

$$V = 2\int_0^{r_0} \left[\pi/2(r_0^2 - x^2) - \frac{1}{3R}(r_0^3 - 3/2x^2 r_0 + \frac{x^4}{2r_0}) + o(x^3)\right] dx =$$

$$= 2\left[\pi r_0^3/3 - \frac{1}{3R}(r_0^4 - 1/2r_0^4 + \frac{r_0^4}{10}) + o(r_0^4)\right] = 2\pi r_0^3/3 - 2/5 r_0^4/R + o(r_0^4).$$

While using the formula 16 we will get

$$R' = \frac{3r^4}{8r^3 - \frac{12}{\pi}(2\pi r^3/3 - 2/5r^4/R + o(r^4))} = \frac{R}{\frac{8}{5\pi} + o(1)}$$

and $\kappa' = 2/R' = \frac{\frac{16}{5\pi} + o(1)}{R}$. Comparing this result with the correct $\kappa = 1/R$ for the cylinder we get the relative error

$$\varepsilon = \frac{\kappa' - \kappa}{\kappa} = \frac{16}{5\pi} - 1 + o(1) \approx 1.86\%$$

So we approximate the curvature having the error less than 2% in the worst case. Note that no mesh effects are taken into the account and they might corrupt the result much more, especially when the grid is quite coarse.

B Pressure ratio at evaporation

The pressure ratio P_{rel} between the vapour pressure directly above the surface of the evaporated metal P_{surf} and the pressure of the ambient atmosphere P_{atm} is necessary for the eq. (29). This ratio can be calculated with help of the Rankine-Hugoniot shock relations [9]:

$$\begin{aligned}
P_{rel}(T_{surf}, M) &= \frac{P_{surf}}{P_{atm}} = \frac{\rho_{surf}}{\rho_{Kn}} \cdot \frac{T_{surf}}{T_{Kn}} \cdot \left(1 + \gamma_{amb} M \frac{c_{Kn}}{c_{amb}}\left(M_{corr} + \sqrt{1 + M_{corr}^2}\right)\right), \\
M_{corr} &= \frac{\gamma_{amb} + 1}{4} M \frac{c_{Kn}}{c_{amb}}, \\
\frac{c_{Kn}}{c_{amb}} &= \sqrt{\frac{m_a^{amb} \gamma_{vap} T_{surf} \frac{T_{Kn}}{T_{surf}}}{m_a^{vap} \gamma_{amb} T_{amb}}},
\end{aligned} \tag{48}$$

where $\dfrac{\rho_{Kn}}{\rho_{surf}}$ and $\dfrac{T_{Kn}}{T_{surf}}$ are Mach-dependent density and temperature jumps across the Knudsen layer and can be found from eq. (24). Ambient gas has temperature T_{amb}, adiabatic index γ_{amb} and atomic mass m_a^{amb}, whilst metal vapor has adiabatic index γ_{vap} and atomic mass m_a^{vap}, M is the Mach number.

If ambient gas has non-zero velocity V_{amb} directed to the normal to the evaporated surface (this is necessary for special gas solver), than:

$$P_{rel}(T_{surf}, M) = \begin{cases} \dfrac{\rho_{surf}}{\rho_{Kn}} \cdot \dfrac{T_{surf}}{T_{Kn}} \cdot \left(1 - \dfrac{\gamma_{amb} - 1}{2} M_{rel}\right)^{\dfrac{2\gamma_{amb}}{1 - \gamma_{amb}}}, & \text{if } M_{rel} \leqslant 0; \\ \dfrac{\rho_{surf}}{\rho_{Kn}} \cdot \dfrac{T_{surf}}{T_{Kn}} \cdot \left(1 + \gamma_{amb} M_{rel} \left(M_{corr_1} + \sqrt{1 + M_{corr_1}^2}\right)\right), & \text{otherwise ;} \end{cases}$$
$$M_{corr_1} = \frac{\gamma_{amb} + 1}{4} M_{rel};$$
$$M_{rel} = M \frac{c_{Kn}}{c_{amb}} - M_{amb};$$
$$M_{amb} = V_{amb} \sqrt{\frac{m_a^{amb}}{\gamma_{amb} k_B T_{amb}}}. \tag{49}$$

When $M_{rel} \leqslant 0$ this is isentropic rarefaction wave conditions.