# Implementation of Solidification Phase Change in a Multiphase Solver

Paria Khosravifar

Department of Engineering Sciences and Mathematics
Luleå University of Technology
Luleå, Sweden

January 23, 2025

LULEÅ UNIVERSITY OF TECHNOLOGY

# Contents

# Introduction

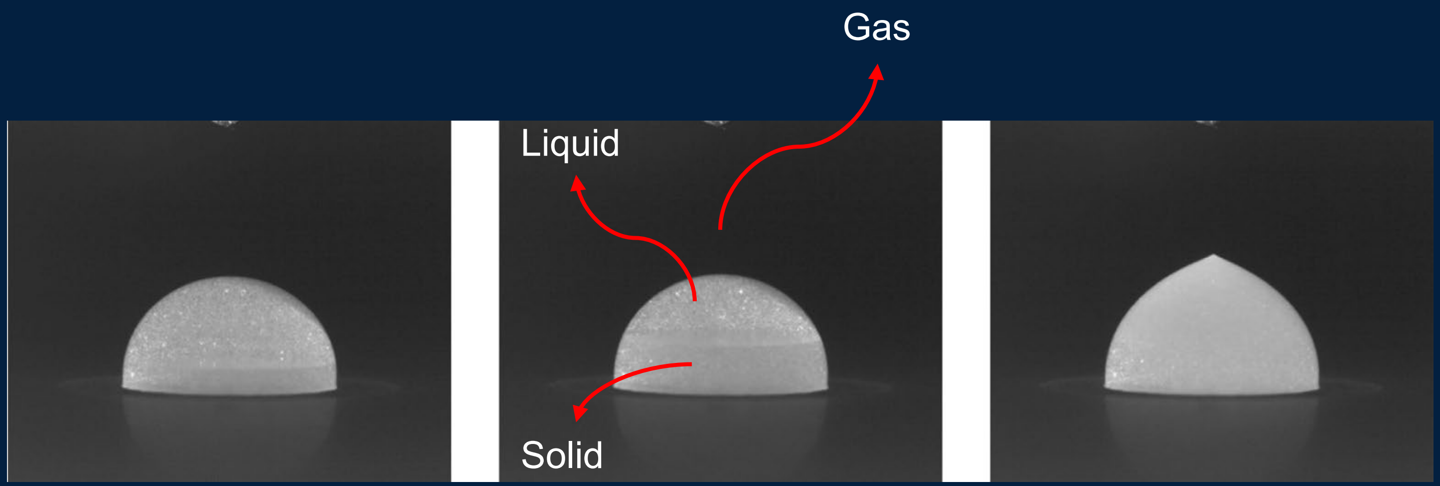
Modeling phase change phenomena, such as solidification or melting is important for understanding various natural and industrial systems.

- Ice formation
- Freezing of food
- Solidification of metal casting
- Thermal energy storage
- Crystal growth

# Different cases

Two-phase (liquid and solid)

Three-phase (gas, liquid, and solid)

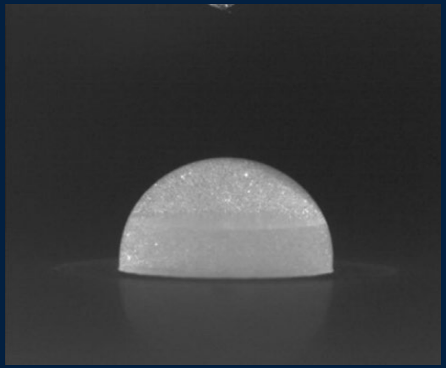



Freezing of an impinging droplet [1]

[1] Fagerström, E. and Ljung, A.L., 2023. Shape and temperature dependence on the directional velocity change in a freezing water droplet. *International Journal of Thermofluids*, *20*, p.100519.

# Main challenge

Tracking the interface between phases



- Interface between liquid and gas → VOF
- Interface between solid and liquid → Enthalpy-porosity method

# Contents

# Volume of Fraction (VOF)

- The VOF model is a numerical technique for capturing the interface between two phases

$$\alpha = \begin{cases} 0 & fluid\ 2 \\ 0 < \alpha < 1 & interface \\ 1 & fluid\ 1 \end{cases}$$

Transport equation of the volume fraction:

$$\frac{\partial \alpha}{\partial t} + \nabla.(\alpha U) + \nabla.[U_r \alpha(1-\alpha)] = 0$$

Material properties:

$$y = \alpha y_1 + (1-\alpha) y_2$$

# Enthalpy-porosity solidification model (1/3)

- This method, presented by Voller and Prakash [2], utilizes the enthalpy method to account for both sensible and latent heat, along with the porosity concept to represent the transition between the solid and liquid phases.

Energy equation:

$$\frac{\partial \rho H}{\partial t} + \nabla \cdot (\rho U H) = \nabla \cdot (k \nabla T)$$

$$H = h + \Delta H$$

$$h = h_{\mathrm{ref}} + \int_{T_{\mathrm{ref}}}^{T} C_p \mathrm{d}T$$

$$\Delta H = \gamma L$$

$$\gamma = \begin{cases} 0 & T < T_{sol} \\ \frac{T - T_{sol}}{T_{liq} - T_{sol}} & T_{sol} \leq T \leq T_{liq} \\ 1 & T > T_{liq} \end{cases}$$

?

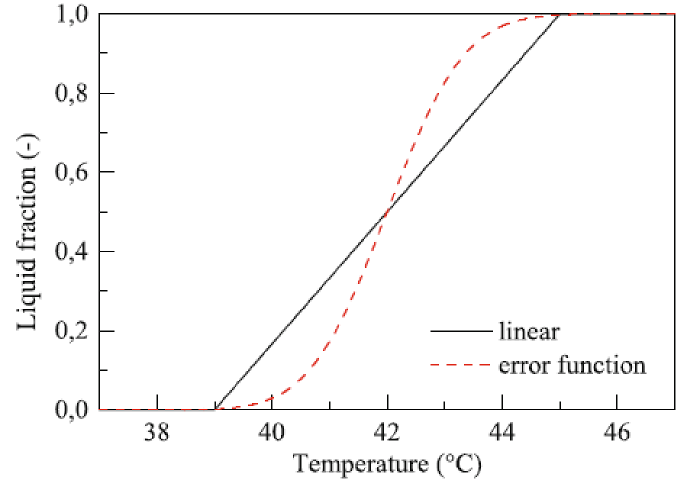
How to take account the existence of two fluids in the relation?

$$\Delta H = \alpha_l \gamma L$$

[2] Voller, V.R. and Prakash, C., 1987. A fixed grid numerical modelling methodology for convection-diffusion mushy region phase-change problems. *International journal of heat and mass transfer*

# Enthalpy-porosity solidification model (2/3)

$$\frac{\partial(\rho C_p T)}{\partial t} + \nabla\cdot(U\rho C_p T) + \alpha_l L\left[\frac{\partial \rho\gamma}{\partial t} + \nabla\cdot(U\rho\gamma)\right] = \nabla\cdot(k\nabla T)$$



Liquid fraction approximation with linear function and error function [3]

$$\gamma = 0.5\,\mathrm{erf}\left(\frac{4(T - T_{melt})}{T_{liq} - T_{sol}}\right) + 0.5$$

$$\frac{\partial(\rho C_p T)}{\partial t} + \nabla\cdot(U\rho C_p T) + S_h = \nabla\cdot(k\nabla T)$$

$$S_h = \alpha_l \rho L \frac{4\exp\left(\left(\frac{4(T - T_{melt})}{T_{liq} - T_{sol}}\right)^2\right)}{(T_{liq} - T_{sol})\sqrt{\pi}}\left(\frac{\partial T}{\partial t} + U\cdot\nabla T\right)$$


[3] Rösler, F. and Brüggemann, D., 2011. Shell-and-tube type latent heat thermal energy storage: numerical analysis and comparison with experiments. Heat and mass transfer

# Enthalpy-porosity solidification model (3/3)

Momentum equation:

$$\frac{\partial(\rho U)}{\partial t} + \nabla \cdot (\rho U U) = -\nabla p + \nabla \mu \left[\nabla U + \nabla U^{\mathrm{T}}\right] + \rho g + S_{\mathrm{st}} + S_{\mathrm{u}}$$

$$S_{\mathrm{u}} = -\frac{(1-\gamma)^2}{\gamma^3 + \mathrm{q}} A_{\mathrm{mushy}} U$$

Material properties:

$$y = \alpha_{\mathrm{l}}\left(\gamma y_{\mathrm{l}} + (1-\gamma) y_{\mathrm{s}}\right) + \left(1-\alpha_{\mathrm{l}}\right) y_{\mathrm{g}}$$

# Contents

# OpenFOAM implementation (1/8)

- interFoam solver

```
Solver for two incompressible, isothermal immiscible fluids using a VOF
(volume of fluid) phase-fraction based interface capturing approach,
with optional mesh motion and mesh topology changes including adaptive
re-meshing.
```

To read and calculate transport properties:

```
#include "immiscibleIncompressibleTwoPhaseMixture.H"

class immiscibleIncompressibleTwoPhaseMixture
:
    public incompressibleTwoPhaseMixture,
    public interfaceProperties
```

To have access to functions and properties in the class:

```
Info<< "Reading transportProperties\n" << endl;
immiscibleIncompressibleTwoPhaseMixture mixture(U, phi);
```

# OpenFOAM implementation (2/8)

- solidificationInterFoam solver

```
A solver for two incompressible, immiscible fluids using a Volume of Fluid
(VOF) phase-fraction-based interface capturing approach, with solidification
phase change in one of the fluids.
```

```
class myThermoIncompressibleTwoPhaseMixture
:
    public incompressibleTwoPhaseMixture
{
protected:
        dimensionedScalar kappa1_; //Thermal conductivity
        dimensionedScalar kappa2_;

        dimensionedScalar Cp1_; //Specific heat capacity
        dimensionedScalar Cp2_;
```

# OpenFOAM implementation (3/8)

myImmiscibleIncompressibleTwoPhaseMixture inheritance:

```
class myImmiscibleIncompressibleTwoPhaseMixture
:
    // public incompressibleTwoPhaseMixture,
    public myThermoIncompressibleTwoPhaseMixture,
    public interfaceProperties
```

Mixture object in createFields:

```
Info<< "Reading transportProperties\n" << endl;
myImmiscibleIncompressibleTwoPhaseMixture mixture(U, phi);
```

Mention `lmyThermoIncompressibleTwoPhaseMixture` and `myImmiscibleIncompressibleTwoPhaseMixture` in `Make/options`

```
-L$(FOAM_USER_LIBBIN) \
-lmyImmiscibleIncompressibleTwoPhaseMixture \
-lmyThermoIncompressibleTwoPhaseMixture
```

# OpenFOAM implementation (4/8)

Define the properties related to phase change:

```
Info<< "Reading phaseChangeProperties" << endl; //Add
IOdictionary phaseChangeProperties
(
    IOobject
    (
        "phaseChangeProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);


dimensionedScalar L //latent heat
(
    "L",
    dimensionSet(0, 2, -2, 0, 0, 0, 0),
    phaseChangeProperties
);
```

# OpenFOAM implementation (5/8)

```
volScalarField gamma
(
    IOobject
    (
        "gamma",
        mesh.time().timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedScalar("gamma", dimless, 1)
);

// Update gamma using the erf function
gamma = 0.5 * Foam::erf(4.0 * (T - Tmelt) / (Tliq - Tsol)) + 0.5;
```

# OpenFOAM implementation (6/8)

```
volScalarField rho
(
    IOobject
    (
        "rho",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT
    ),
    alpha1* (gamma*rho1+(1-gamma)*rhoS) + alpha2*rho2

);


volScalarField Cp //Add
(
    IOobject
    (
        "Cp",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT
    ),
    alpha1* (gamma*Cp1+(1-gamma)*CpS) + alpha2*Cp2
);
```

# OpenFOAM implementation (7/8)

Energy equation:

```
    volScalarField expArg = sqr(4.0 * (T - Tmelt) / (Tliq - Tsol));
    volScalarField Sh_erf = -rho * L * ((4.0 * exp(-expArg) / ((Tliq - Tsol) *
::sqrt(Foam::constant::mathematical::pi))) * (fvc::ddt(T) + (U & fvc::grad(T))));

    fvScalarMatrix TEqn
    (
          fvm::ddt(rhoCp, T)
        + fvm::div(rhoCpPhi, T)
        + alpha1*Sh_erf
        - fvm::Sp(fvc::ddt(rhoCp) + fvc::div(rhoCpPhi), T)
        - fvm::laplacian(kappaEff, T)
    );
```

# OpenFOAM implementation (8/8)

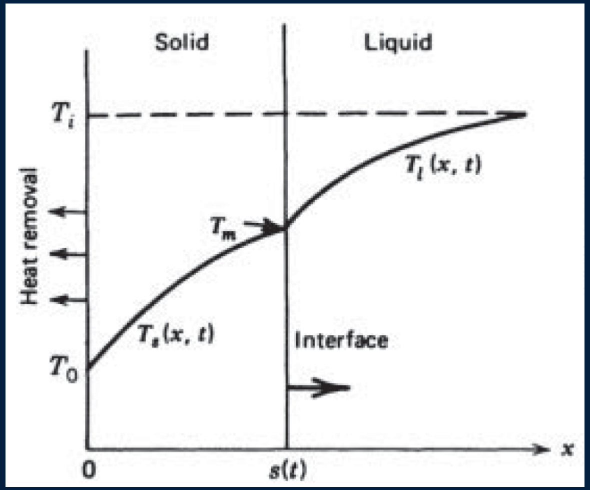
Momentum equation:

```
volScalarField prosityFunc = A_mushy *sqr(1.0 - gamma)/(pow3(gamma) + q);

fvVectorMatrix UEqn
(
    fvm::ddt(rho, U) + fvm::div(rhoPhi, U)
  + alpha1 * fvm::Sp(prosityFunc, U) //Add
  + MRF.DDt(rho, U)
  + turbulence->divDevRhoReff(rho, U)
 ==
    fvOptions(rho, U)
);

UEqn.relax();

fvOptions.constrain(UEqn);

if (pimple.momentumPredictor())
{
    solve
    (
        UEqn
     ==
        fvc::reconstruct
        (
            (
                mixture.surfaceTensionForce()
              - ghf*fvc::snGrad(alpha1*rho1+alpha2*rho2)
              - fvc::snGrad(p_rgh)
            ) * mesh.magSf()
        )
    );

    fvOptions.correct(U);
}
```

# Contents

# Stefan problem

- One-dimentional solidification problem




Geometry and coordinates for one-dimensional solidification problems [4]

Analytical solution (Neumann's solution):

$$x(t) = 2\lambda\sqrt{\frac{tk_s}{C_s\rho}}$$

$$\lambda e^{\lambda^2} = \frac{\text{Ste}}{\sqrt{\pi}}$$

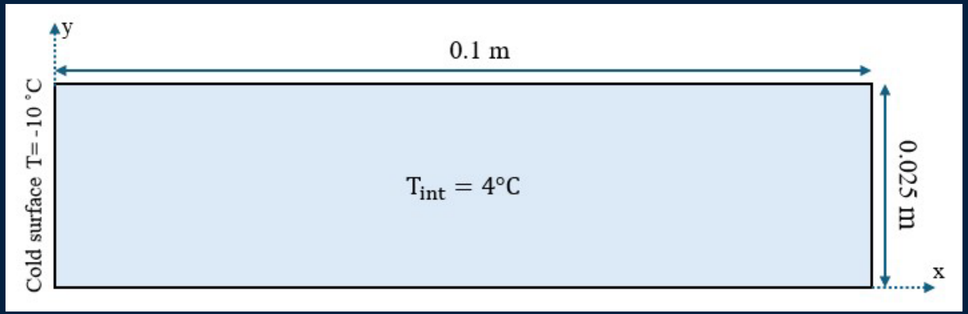
$$\text{Ste} = \frac{C_s(T_{\text{melt}} - T_0)}{\text{L}}$$

[4] D. W. Hahn and M. N. ¨Ozisik, Heat conduction. John Wiley & Sons, 2012.

# Test case 1

- Two-phase Stefan problem

```
defaultFieldValues
(
    volScalarFieldValue alpha.water 0
);

regions
(
    boxToCell
    {
        box (0 -0.001 0) (0.025 0.001 0.1);
        fieldValues
        (
            volScalarFieldValue alpha.water 1
        );
    }
);
```



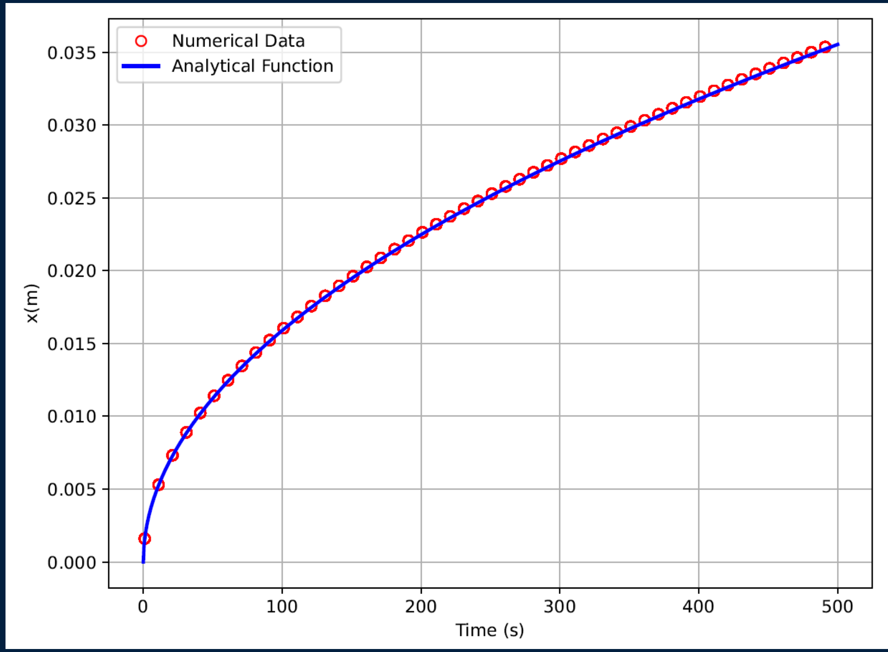

| Properties | Solid | Liquid | Gas | Interface |
|---|---|---|---|---|
| Heat capacity (J/kg·K) | 2050 | 2590 | 1 | |
| Thermal conductivity (W/m·K) | 4.02 | 2.89 | 0.025 | |
| Density (kg/m$^3$) | 1000 | 1000 | 1 | |
| Melting temperature (K) | | | | 273.15 |
| Latent heat of fusion (J/kg) | | | | 80332 |

```
Tmelt           273.15;
Tliq            273.35;
Tsol            272.95;
```

# Result

```
freezingFront
{
    type             surfaces;
    libs             (sampling);
    writeControl     writeTime;

    surfaceFormat   raw;
    fields          (T);

    interpolationScheme cellPoint;

    surfaces
    {
        isoT
        {
            type            isoSurface;
            isoField        T;
            isoValue        273.15;
            interpolate     true;
            triangulate     true;
        }

    }
}
```

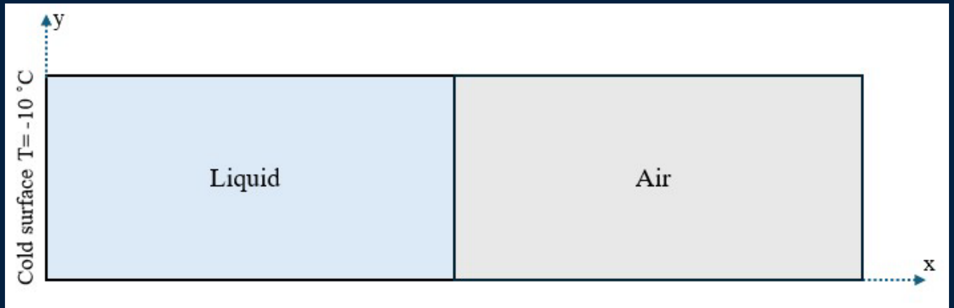
$$x(t) = 2\lambda\sqrt{\frac{tk_s}{C_s\rho}}$$

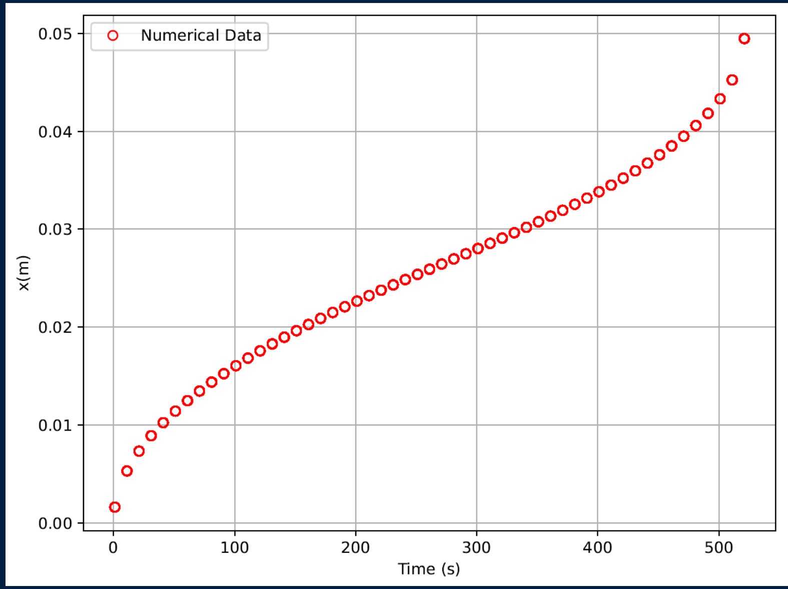


# Test case 2

- Three-phase Stefan problem

```
defaultFieldValues
(
    volScalarFieldValue alpha.water 0
);

regions
(
    boxToCell
    {
        box (0 -0.001 0) (0.025 0.001 0.050);
        fieldValues
        (
            volScalarFieldValue alpha.water 1
        );
    }
);
```



# Contents

# Conclusion and future works

- **Conclusion**:

• Energy equation and enthalpy-porosity phase change model implemented in the interFoam solver to simulate the solidification phenomenon.

• The implemented solver was validated using the analytical solution of the Stefan problem.

- **Future Work**:

• Consider the density change during phase change to better detect flow currents during phase change.

• Account for volume expansion due to phase change.

LULEÅ UNIVERSITY OF TECHNOLOGY