Analyzing Influence of Non-neutral Contact Angle on the Cahn-Hilliard Model

Project Thesis

Jonathan Ulmer

November 20, 2024

Contents

Abstract

This work shows sensitivity of boundary conditions for two different finite difference approaches to solving the cahn hilliard equation

1 Fundamentals

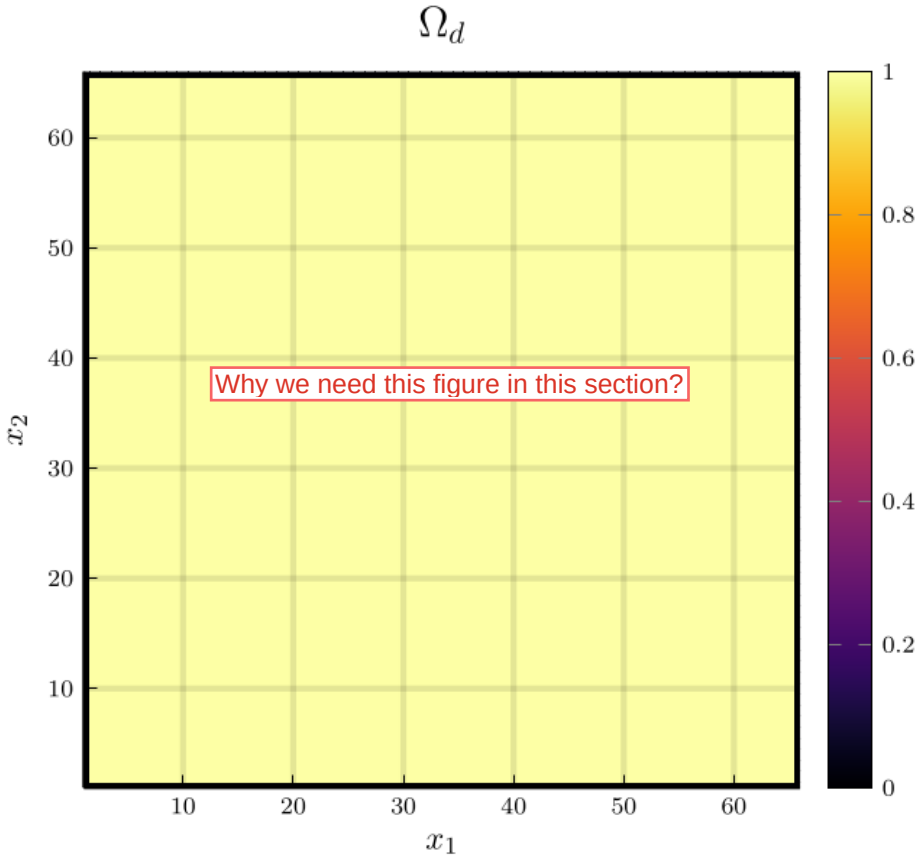
The Cahn-Hilliard (CH) equation is a diffuse interface coupling equation used for interpolating between different approach in different formulas. We use it in the following form:

$$\begin{aligned} \partial_t \phi(x,t) &= \nabla \cdot (M(\phi)\nabla\mu) \\ \mu &= -\varepsilon^2 \Delta\phi + W'(\phi) \end{aligned} \tag{1.1}$$

Furthermore our solver implements the ansatz proposed by the autors[1].

$$\begin{aligned} \frac{\phi_{ij}^{n+1} - \phi_{ij}^{n}}{\Delta t} &= \nabla_d \cdot (G_{ij} \nabla_d \mu_{ij}^{n+\frac{1}{2}}) \\ \mu_{ij}^{n+\frac{1}{2}} &= 2\phi_{ij}^{n+1} - \varepsilon^2 \nabla_d \cdot (G_{ij} \nabla_d \phi_{ij}^{n+1}) + W'(\phi_{ij}^{n}) - 2\phi_{ij}^{n} \end{aligned} \tag{1.2}$$

Although we use as slightly different implementation as in the bachelor thesis[2]., use the same discretizations and notation.



1.1 Notation

We use the following differential quotients:

$$D_x f_{i+\frac{1}{2}j} = \frac{f_{i+1j} - f_{ij}}{h} \qquad D_y f_{ij+\frac{1}{2}} = \frac{f_{ij+1} - f_{ij}}{h} \tag{1.3}$$

and define a discrete gradient as.

$$\nabla_d f_{ij} = (D_x f_{i+1j},\ D_y f_{ij+1}) \tag{1.4}$$

see[2]

2 Adaptation to second order Neumann conditions

The solver we use as reference guaranties no flux boundary conditions at a discrete level by setting $\nabla\phi_{ij} = 0$ for $\phi_{ij} \in \partial\Omega_d$ this is done by multiplying with the Characteristic function of Ω_d

$$G_{ij} = \begin{cases} 1\,, x_{ij} \in \Omega \\ 0\,, x_{ij} \notin \Omega \end{cases} \tag{2.1}$$

~~The~~ accommodate different boundary conditions, we bias $\nabla_d \cdot (G_{ij}\nabla_d\phi_{ij})$ in grid points next to the boundary. We determine those points using a ~~centred~~ difference scheme on G

$$B_{ij} = \max\Big(|G_{i+\frac{1}{2}j} - G_{i-\frac{1}{2}j}|, |G_{ij+\frac{1}{2}} - G_{ij-\frac{1}{2}}|\Big) * C \tag{2.2}$$

where C is a constant ~~we~~ chose freely. For example ~~on~~ a 32x32 ~~Domain~~ with $C = 1$ the ~~Boundary fiels~~ $\mathbf{B}$ appears as ~~follows~~

~~We then state the adapted approach as:~~

$$\begin{aligned} \frac{\phi_{ij}^{n+1} - \phi_{ij}^{n}}{\Delta t} &= \nabla_d \cdot (G_{ij}\nabla_d \mu_{ij}^{n+\frac{1}{2}}) \\ \mu_{ij}^{n+\frac{1}{2}} &= 2\phi_{ij}^{n+1} - \varepsilon^2\nabla_d \cdot (G_{ij}\nabla_d\phi_{ij}^{n+1}) + B_{ij} + W'(\phi_{ij}^{n}) - 2\phi_{ij}^{n} \end{aligned} \tag{2.3}$$

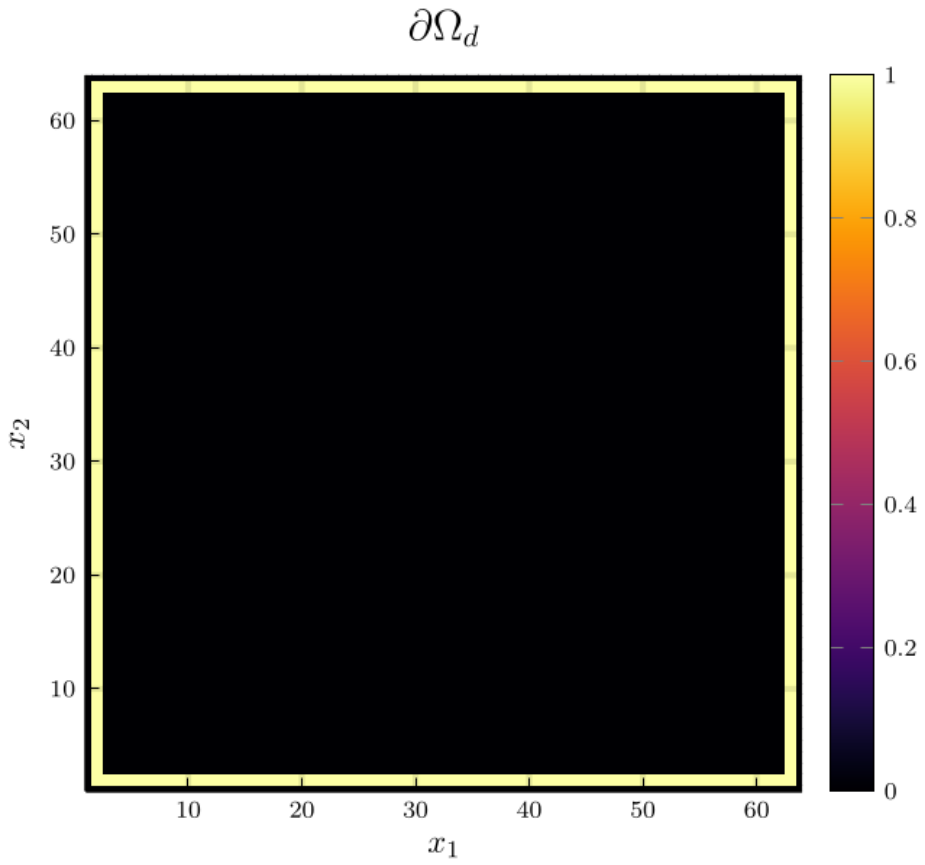


Figure 2.1: visualization of all grid-cells adjacent to the boundary $\partial\Omega_d$ of a square domain

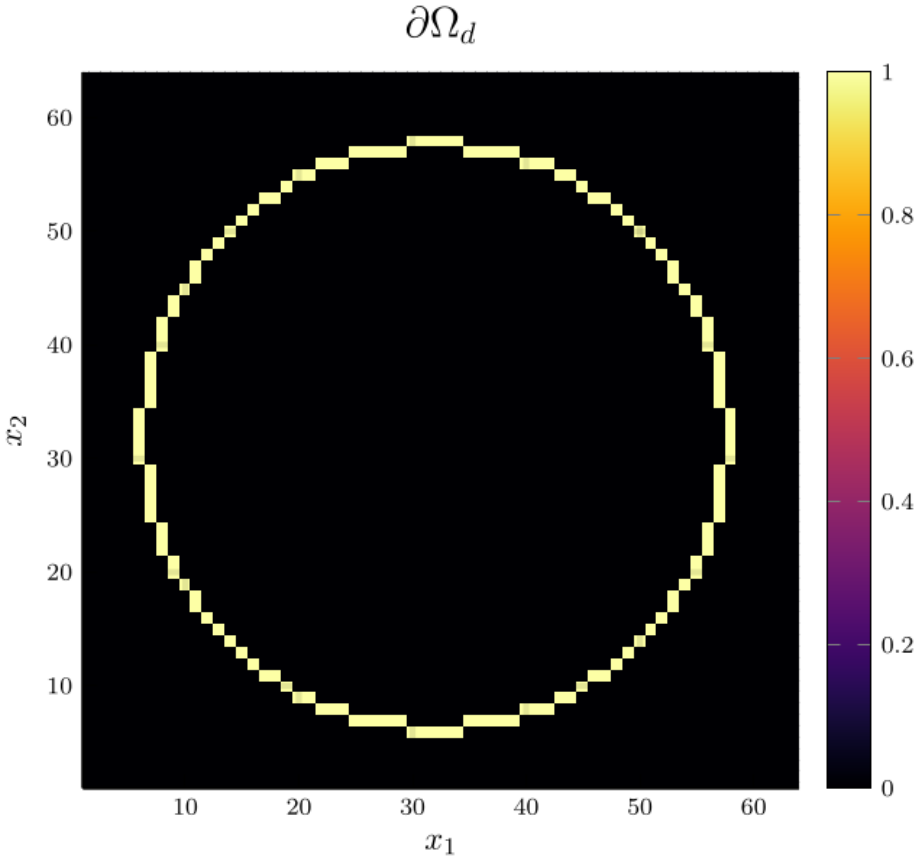


Figure 2.2: visualization of all grid-cells adjacent to the boundary $\partial\Omega_d$ of a circular domain

3 Numerical solver

contrary to the solver proposed in[2] we do not use a multi-grid Gauss-Seidel Solver to solve the linear system, and use a Jacoby solver instead, since it is easier to parrallize. Similar to [2] we linearise (2.3) to

$$\begin{aligned} \frac{\phi_{ij}^{n+1}}{\Delta t} - \nabla_d \cdot (G_{ij} \nabla_d \mu_{ij}^{n+\frac{1}{2}}) &= \frac{\phi_{ij}^{n}}{\Delta t} \\ \mu_{ij}^{n+\frac{1}{2}} - 2\phi_{ij}^{n+1} + \varepsilon^2 \nabla_d \cdot (G_{ij} \nabla_d \phi_{ij}^{n+1}) + B_{ij} &= 2\phi_{ij}^{n} - W'(\phi_{ij}^{n}) \end{aligned} \tag{3.1}$$

after some rearranging we note, that the left hand side is linear and, the right hand side is solely dependent on the previous time step. Therefore this constitutes a linear system, wich we solve with a Jacoby method, the element wise formula of wich is given as follows: Provided the mth Jacoby iteration has been computed, the $m+1th$ iteration is computed by solving

$$\begin{aligned} \frac{\phi_{ij}^{n+1,m+1}}{\Delta t} - \nabla_d \cdot (G_{ij} \nabla_d \mu_{ij}^{n+\frac{1}{2},m+\frac{1}{2}}) &= \frac{\phi_{ij}^{n}}{\Delta t} \\ \mu_{ij}^{n+\frac{1}{2},m} - 2\phi_{ij}^{n+1,m} + \varepsilon^2 \nabla_d \cdot (G_{ij} \nabla_d \phi_{ij}^{n+1,m+\frac{1}{2}}) + B_{ij} &= 2\phi_{ij}^{n} - W'(\phi_{ij}^{n}) \end{aligned} \tag{3.2}$$

for $\phi_{ij}^{n+1,m+1}, \mu_{ij}^{n+\frac{1}{2},m+1}$, where $\nabla_d \cdot (G_{ij} \nabla_d \mu_{ij}^{n+\frac{1}{2},m+\frac{1}{2}})$ and $\nabla_d \cdot (G_{ij} \nabla_d \phi_{ij}^{n+1,m+\frac{1}{2}})$. Use the results from the previous jacoby step for values off the center. eg.

$$\begin{aligned} \nabla_d \cdot (G_{ij} \nabla_d \phi_{ij}^{n+1,m+\frac{1}{2}}) = &\frac{1}{h^2} (G_{i+\frac{1}{2}j} \phi_{i+1j}^{n+1,m} + G_{i-\frac{1}{2}j} \phi_{i-1j}^{n+1,m} \\ &+ \quad G_{ij+\frac{1}{2}} \phi_{ij+1}^{n+1,m} + G_{ij-\frac{1}{2}} \phi_{ij-1}^{n+1,m}) \\ &- \left(G_{i+\frac{1}{2}j} + G_{i-\frac{1}{2}j} + G_{ij+\frac{1}{2}} + G_{ij-\frac{1}{2}}\right) \phi_{ij}^{n+1,m+1} \end{aligned} \tag{3.3}$$

our implementation makes use of julia to dispatch the solution for each element in paralell on the GPU. The full implementation of the jacoby iteration is given as:

```
@kernel function jacoby!(
    Φ,
```

```
    M,
    @Const(Ξ),
    @Const(Ψ),
    @Const(h),
    @Const(ε),
    @Const(Δt),
    @Const(iterations)
)
    I   = @index(Global, Cartesian)
    Id  = oneunit(I)
    Ids = CartesianIndices(M)
    Ix = CartesianIndex(1, 0)
    Iy = CartesianIndex(0, 1)
    if I in (Ids[begin]+Id:Ids[end]-Id)
        g = G(2 * I + Ix, Ids) + G(2 * I + Iy, Ids) + G(2 * I - Ix, Ids) + G(2 * I - Iy,
        ↪  Ids)
        a1 = 1/Δt
        a2 = -1* ε^2/h^2 * g  - 2
        b1 = 1/h^2 * g
        b2 = 1
        for _ = 1:iterations

            Σμ = G(2 * I + Ix, Ids) * M[I+Ix] + G(2 * I + Iy, Ids) * M[I+Iy] + G(2 * I -
            ↪  Ix, Ids) * M[I-Ix] + G(2 * I - Iy, Ids) * M[I-Iy]

            Σϕ = G(2 * I + Ix, Ids) * Φ[I+Ix] + G(2 * I + Iy, Ids) * Φ[I+Iy] +G(2 * I -
            ↪  Ix, Ids) * Φ[I-Ix] +G(2 * I - Iy, Ids) * Φ[I-Iy]

            c1 = Ξ[I] + 1/h^2   * Σμ
            c2 = Ψ[I] - ε^2/h^2 * Σϕ

            # stupid matrix solve
            @inline Φ[I] = (c1*b2 - c2*b1) / (a1*b2 - a2*b1)
            @inline M[I] = (a1*c2 - a2*c1) / (a1*b2 - a2*b1)
            #
            @synchronize()
        end

    end
end
```

4 Numerical evaluation

4.1 Experiments

to begin our evaluations we tested constant values for B_{ij} on the boundary. initially we tested setting $C = 0$ as it is equivalent to the no-flux boundary condition of the unmodified solver. For $C = 0$,as seen in 4.1 the interface lies orthogonal on the boundary as expected for a CH solver with no-flux boundary conditions. For $B_{ij} \in \{-1, 1\}$ we observed behaviour expected of hydrophobic / hydrophilic substances on the boundary, where $B_{ij} = 1$ resulted in the one phase pearling of the boundary, while the other semed attracted. this manifested on apparent contact angles of 180° and 0° respectivly. Using $B_{ij} = -1$ results in the opposite behavior.

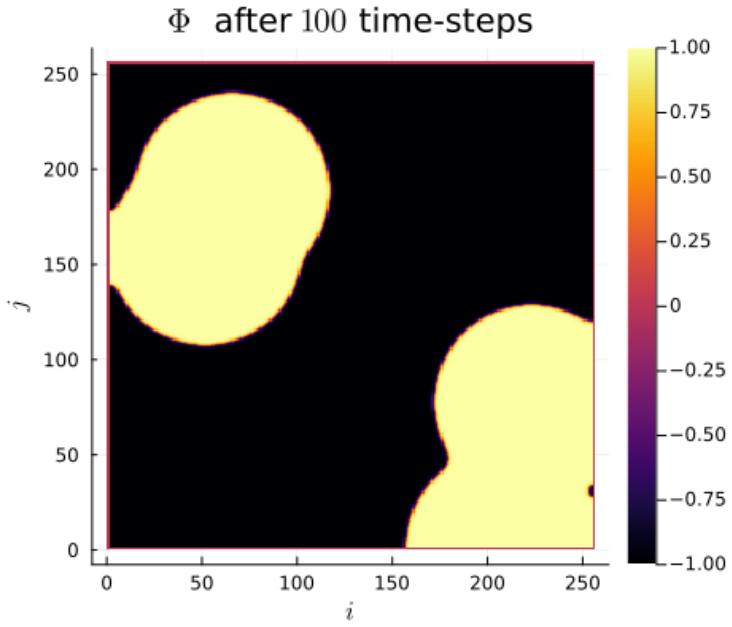
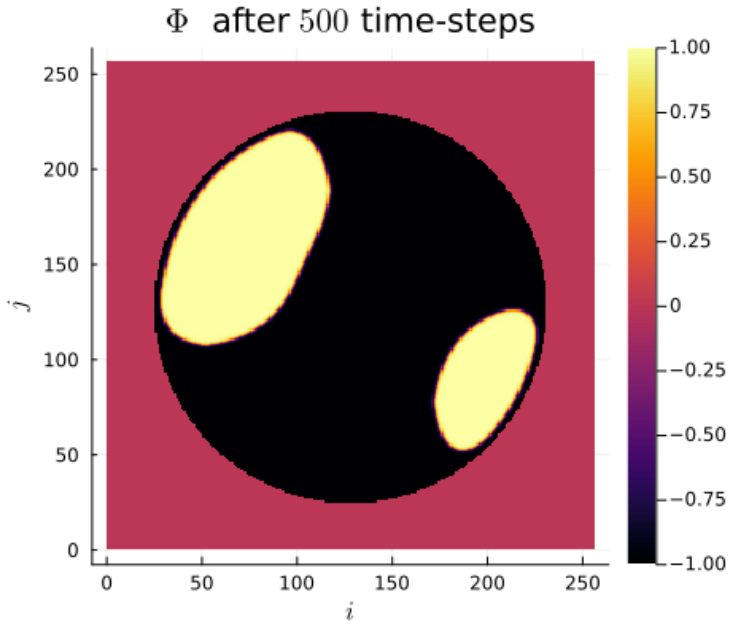


Figure 4.1: phase-field ϕ after 100 time-steps with $C = 0$ emmulating no-flux boundary.

To showcase the relative stability, and the effect off the constant boundary proposed in TODO we evaluate our solver with different constant values C. In 4.1 we

employ a constant value of $C = 1$ and observe the phase corresponding to $\phi = 1$ puling away from the boundary. the contact angle between phase 1 and the boundary approaches 180° ie. the interface runs parallel to the boundary.



In 4.2 we try the opposite as bevore. And we obsesrve corresponding behaviour. When using a value of $C = -1$ we observe behaviour opposite to before. Where the contact angle on the boundary lies at 0°, the interface runs parallel to the boundary again.

we observe the most interessting behaviour for values between $(-1, 1)$, where we observe the contact angle of the interafce on the boundary to change from parallel 0° to parallel 180°.

```
using LaTeXStrings
include("solvers.jl")
θ = 5f-3
arr = cu(rand(Float32,256,256)) .* 2 .- 1
d = domain(get_backend(arr) , 256 , size(arr))
d(arr)
solution = solve(arr , 5 , θ=θ)
h1 = heatmap(Array(solution) , aspect_ratio=:equal , clims=(-1,1),
↪  lims=(0,size(solution,1)), widen=1.06 , title=L"n=5")
solution = solve(arr , 50 , θ=θ)
h2 = heatmap(Array(solution) , aspect_ratio=:equal , clims=(-1,1),
↪  lims=(0,size(solution,1)), widen=1.06 , title=L"n=50")
```

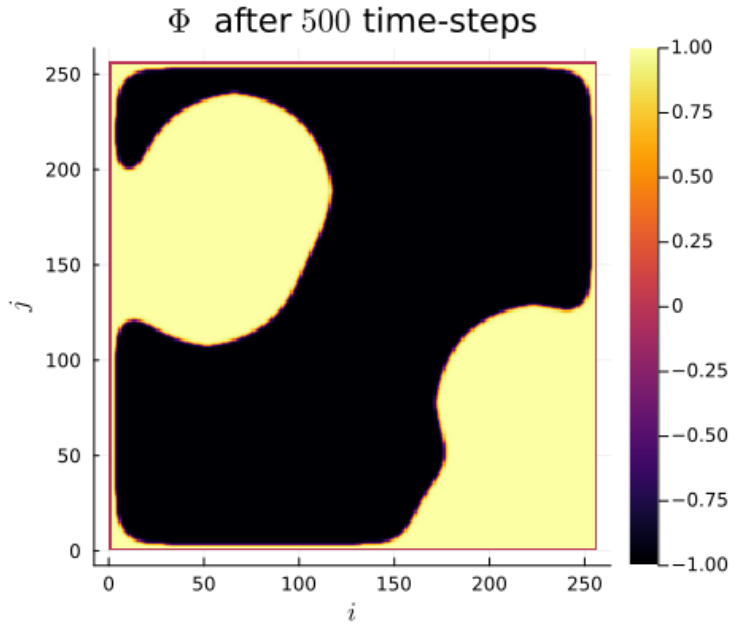


Figure 4.2: phase-field ϕ after 100 time-steps with $C = -1$

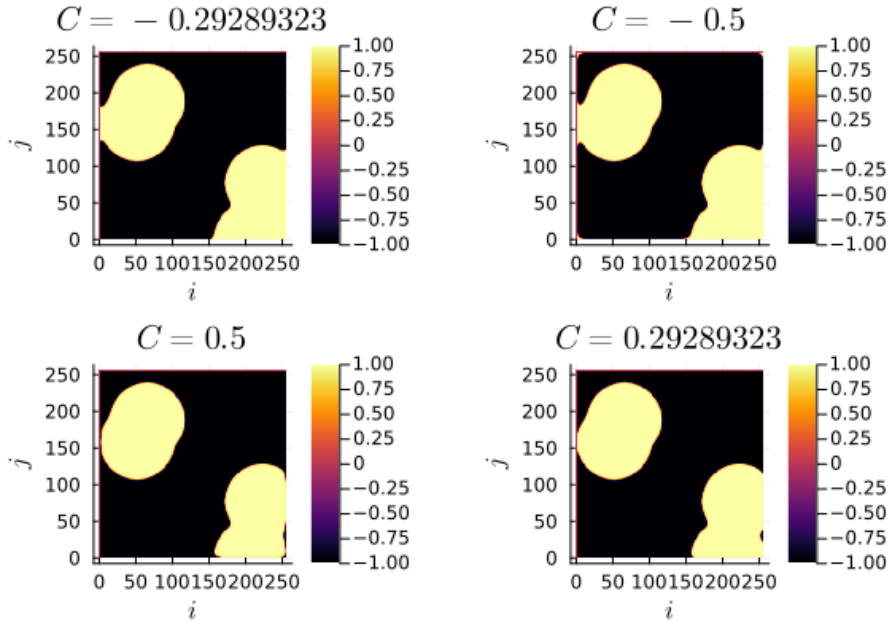


Figure 4.3: phase-field ϕ after 500 time-steps with $C \in \{-1 + \frac{\sqrt{2}}{2}, -0.5, 0.5, 1 - \frac{\sqrt{2}}{2}\}$

```
solution = solve(arr , 500 , θ=0)
h3 = heatmap(Array(solution) , aspect_ratio=:equal , clims=(-1,1),
↪  lims=(0,size(solution,1)), widen=1.06 , title=L"n=500")
solution = solve(arr , 5000 , θ=0)
h4 = heatmap(Array(solution) , aspect_ratio=:equal , clims=(-1,1),
↪  lims=(0,size(solution,1)), widen=1.06 , title=L"n=5000")
plot(h1,h2,h3,h4)
```

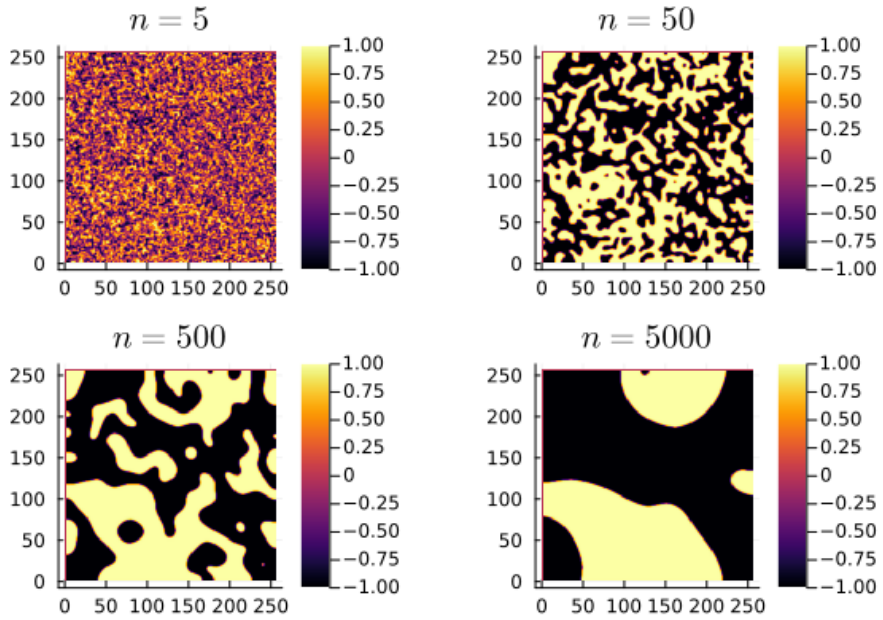


Figure 4.4: phase-field ϕ after 100 time-steps with $C = -\frac{\sqrt{2}}{2}$

4.2 more values

further testing shows that with values for $B_{ij} \in [-1, 1]$ we are able to control the contact angle freely.

4.3 Boundary Normals

we are interested in the contact angle of the interface on the boundary. ie.

$$\frac{\nabla_d \phi_{ij} \cdot \mathbf{n}}{\|\nabla_d \phi_{ij}\|} = \cos(\theta) \qquad \phi_{ij} \in \partial\Omega_d = 0 \tag{4.1}$$

ie.

$$\frac{\nabla_d \phi_{ij} \cdot \mathbf{n}}{\|\nabla_d \phi_{ij}\|} - \cos(\theta) = 0 \tag{4.2}$$

$$\frac{\nabla_d \phi_{ij} \cdot \mathbf{n}}{\|\nabla_d \phi_{ij}\|} - \cos(\theta)\mathbf{n} \cdot \mathbf{n} = 0 \tag{4.3}$$

$$\left(\frac{\nabla_d \phi_{ij}}{\|\nabla_d \phi_{ij}\|} - \cos(\theta)\mathbf{n} \right) \cdot \mathbf{n} = 0 \tag{4.4}$$

$$\left(\nabla_d \phi_{ij} - \cos(\theta)\|\nabla_d \phi_{ij}\|\mathbf{n} \right) \cdot \mathbf{n} = 0 \tag{4.5}$$

$$\tag{4.6}$$

therefore we use

$$\nabla \cdot \left(\nabla_d \phi_{ij} - \cos(\theta)\|\nabla_d \phi_{ij}\|\mathbf{n} \right) \cdot \mathbf{n} = \nabla \cdot \nabla \phi_{ij} - \cos(\theta) div(\mathbf{n}) \tag{4.7}$$

$$\tag{4.8}$$

in our implementation we modify $\nabla_d(G\nabla_d\phi_{ij})$. In essence our solver tries to ensure no flux boundary conditions ie. $\nabla\phi_{ij} \cdot \mathbf{n} = 0$. hence B_{ij} should be $-\cos(\theta)\left|\nabla_d \phi_{ij}\right| div(\mathbf{n})$

5 Numerical evaluation on a circle

5.1 Experiments

The original solver presented in[1] was able to solve the CH equation on arbitrary domains. Since the addition of our boundary function depends soley on the characteristic function of the discrete domain, we are able to use our approach on diferent Domains, by providing a different characteristic function. We present the results of which in this chapter. To show the behavior of the CH sover in 5.1, we first emplay no-flux boundary conditioons on a circular domain. We observe the interface perpendicular on the boundary, as we expect.

```
G (generic function with 1 method)
```

The results we oserve in 5.2 are similar to the reults on a square domain in 4.1. The contact angle is 18° i.e. the interface does not touch the boundary and runs parallel to it.

The results for $C = -1$ in 5.3 on the circular domain, are similar to the results in 4.2 onth esqaure domain as well ,where the interface touches the boundary and iun parallel with a contact angle of 0°.

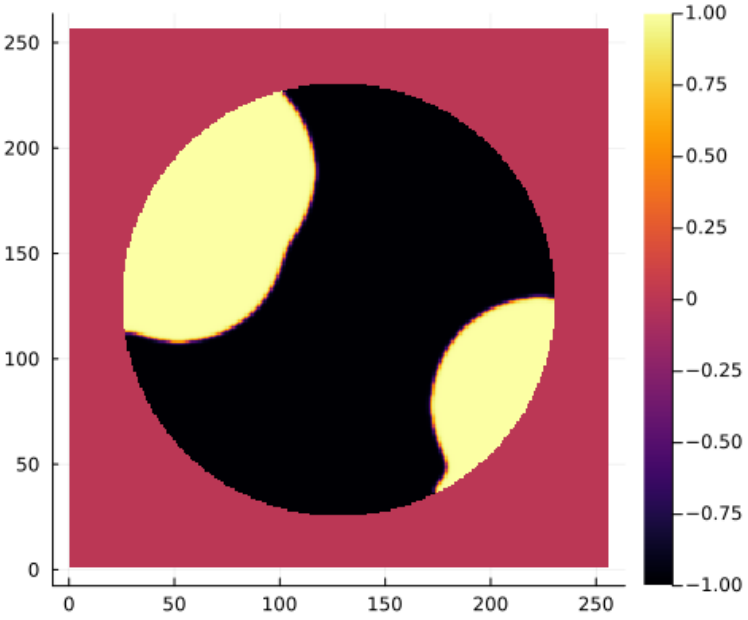


Figure 5.1: ϕ after 100 time steps on a circular domain with no-flux boundary-conditions after 100 time steps on a circular domain with no-flux

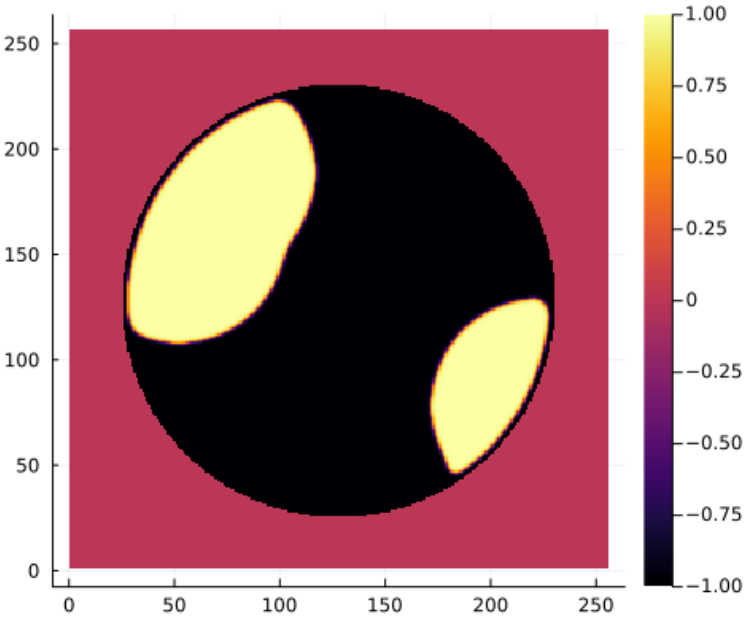


Figure 5.2: phase-field ϕ after 100 time-steps with $C = 1$

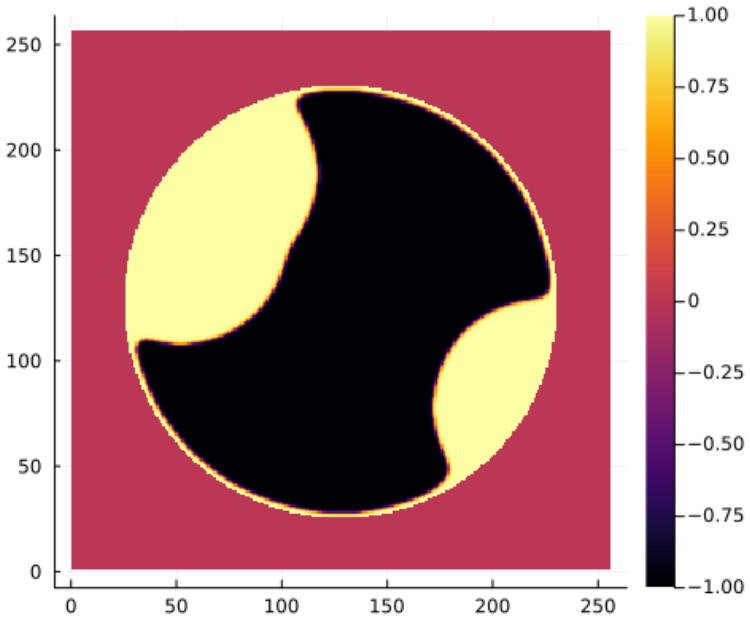


Figure 5.3: phase-field ϕ after 100 time-steps with $C = -1$

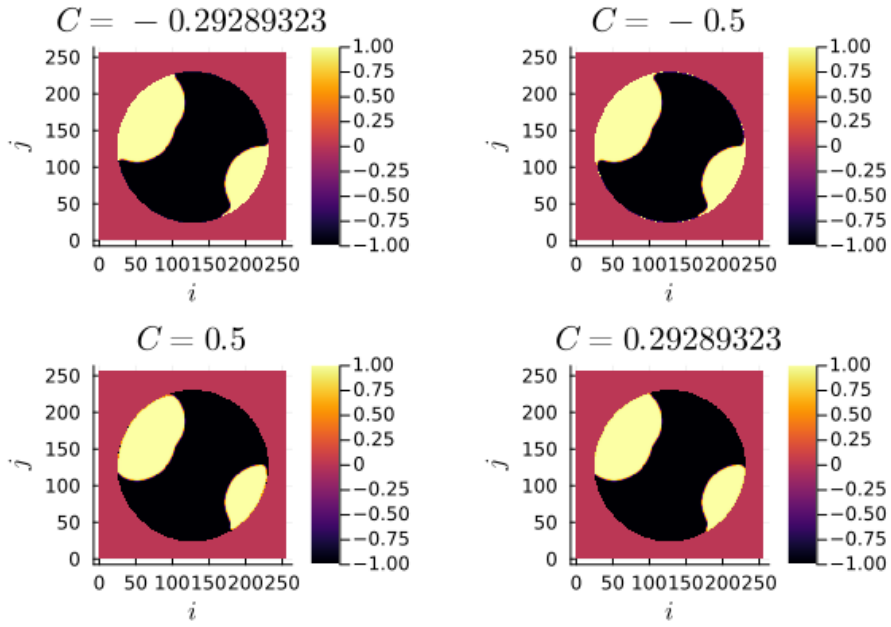


Figure 5.4: phase-field ϕ after 500 time-steps with $C \in \{-1 + \frac{\sqrt{2}}{2}, -0.5, 0.5, 1 - \frac{\sqrt{2}}{2}\}$ on a circular domain.

6 Summary and outlook

7 References

Bibliography

[1] Jaemin Shin, Darae Jeong, and Junseok Kim. "A conservative numerical method for the Cahn–Hilliard equation in complex domains". In: *Journal of Computational Physics* 230.19 (2011), pp. 7441–7455. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2011.06.009. URL: https://www.sciencedirect.com/science/article/pii/S0021999111003585.

[2] Jonathan Ulmer. "Untitled1". Bachelor Thesis. University of Stuttgart, 3000.