# A Simple Finite Volume Method Discretization of the Magnetostatic Potential Problem

Kenny Erleben
Department of Computer Science
University of Copenhagen
kenny@diku.dk

## Abstract

Here I outline how to get the second order boundary condition $\frac{\partial^2 \phi}{\partial n^2} = 0$ into a simple finite volume method discretization.

***Keywords*** Magnetostatic Potentialss, Boundary Conditions

***Categories and Subject Descriptors*** I.3.5 [*Computer Graphics*]: Computational Geometry and Object Modeling—Physically based modeling; G.1.0 [*Mathematics of Computing*]: General—Numerical Algorithms

## 1. The Mathematical Model of Magnetostatic Problems

Omitting the physics details for brevity we simply state the mathematical model here as a partial differential equation (PDE) problem. We are solving for the unknown potential field $\phi : \mathbb{R}^2 \mapsto \mathbb{R}$ given the govering equation

$$\nabla^2 \phi(\mathbf{x}) = \nabla \cdot \mathbf{m}(\mathbf{x}) \tag{1}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})^T$ is the differential vector operator and the position field is given by $\mathbf{x} = (x, y) \in \mathbb{R}^2$. Here $\mathbf{m}(\mathbf{x})$ is a known dipole field assumed constant inside and on the surface of the magnet and zero out-side the magnet. Let the magnet be given by the closed point set $\mathcal{A} \equiv \left\{\mathbf{x} \in \mathbb{R}^2 \,|\, \mathbf{x} \in \text{magnet}\right\}$ then we may write

$$\mathbf{m}(\mathbf{x}) = \begin{cases} \mathbf{k} & \text{if } \mathbf{x} \in \mathcal{A}, \\ \mathbf{0} & \text{if } \mathbf{x} \notin \mathcal{A} \end{cases} \tag{2}$$

where $\mathbf{k}$ is a user-specified constant. typically $\mathbf{k} = (1, 0)^T$.

To make the model well-posed we need one Dirichlet condition at a single point in our domain.

$$\phi(\mathbf{x}_{\text{DC}}) = 0 \tag{3}$$

where $\mathbf{x}_{\text{DC}}$ is some known location in our domain. We assume our domain given by the point set $\Omega$ is a square domain. Hence we typically apply the Dirichlet condition at the lower-left corner point of the domain.

On our domain boundaries we need to specify the behaviour of $\phi(\mathbf{x})$ to make our model well-posed. Given that we wish to model an infinite big world we wish for the potential field to expand/extrapolate across boundaries unchanged. This implies that the slope of $\phi(\mathbf{x})$ across the boundaries should be unchanged. Let the boundaries be given by the point set $\Gamma$ then we can now write the desired boundary conditions as follows

$$\frac{\partial^2 \phi(\mathbf{x})}{\partial^2 \mathbf{n}} = 0 \quad \forall \mathbf{x} \in \Gamma \tag{4}$$

where $\mathbf{n}$ is the outward unit normal at the boundary point.

## 2. Discretization using a Triangle Mesh

To numerically solve for $\phi$ we discretize the model on a triangle mesh using the finite volume method. Our mesh layout uses the triangles as control volumes and stores a single $\phi$ value et the geometric center of each triangle. We assume that the boundary of the magnet to be conforming with the edges of our triangles. Hence a triangle is either completely inside the magnet or totally outside the magnet. As the magnet is considered a closed point set, we consider the edge between a air-triangle and magnet-triangle to be a magnet-edge.

Let us consider the $C^{\text{th}}$ triangle with the counter-clockwise vertex labels $i$, $j$, and $k$. Let the adjacent triangles be labeled by $I$, $J$, and $K$ in such a way that triangle $I$ is opposite vertex $i$. That is triangle $I$ is sharing edge $jk$ with triangle $C$. Let $\mathcal{V} \equiv \{i, j, k\}$ and $\mathcal{N} \equiv \{I, J, K\}$. We denote the geometric centers of the four triangles as $\mathbf{x}_C$, $\mathbf{x}_I$, $\mathbf{x}_J$, and $\mathbf{x}_K$, similar vertex positions are $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{x}_k$. The vector between two positions are written as $\mathbf{l}_{ik} = \mathbf{x}_i - \mathbf{x}_k$ and the length between two positions are $l_{ik} = \| \mathbf{x}_i - \mathbf{x}_k \|$, similar $\mathbf{l}_{ij}$ and $l_{ij}$ would have the same semantics.

To derive the finite volume method we first write the partial differential equation into volume form

$$\int_{\Omega_c} \nabla \cdot \nabla \phi(\mathbf{x}) dA = \int_{\Omega_c} \nabla \cdot \mathbf{m}(\mathbf{x}) dA \tag{5}$$

where $\Omega_c$ is the closed point-set of the $C^{\text{th}}$ triangle and $dA$ is the differential area element. Next we apply the Gauss divergence theorem

$$\oint_{\Gamma_c} \mathbf{n} \cdot \nabla \phi(\mathbf{x}) dS = \oint_{\Gamma_c} \mathbf{n} \cdot \mathbf{m}(\mathbf{x}) dS \tag{6}$$

and $dS$ is the differential arc-length element. Since the triangle has three edges of constant outward unit normals we use this knowledge to rewrite into

$$\begin{aligned} &\int_{\Gamma_I} \mathbf{n}_I \cdot \nabla \phi dS + \int_{\Gamma_J} \mathbf{n}_J \cdot \nabla \phi dS + \int_{\Gamma_K} \mathbf{n}_K \cdot \nabla \phi dS \\ &= \int_{\Gamma_I} \mathbf{n}_I \cdot \mathbf{m} dS + \int_{\Gamma_J} \mathbf{n}_J \cdot \mathbf{m} dS + \int_{\Gamma_K} \mathbf{n}_K \cdot \mathbf{m} dS \end{aligned} \tag{7}$$

Here $\mathbf{n}_I$ is the outward unit normal of the edge between triangle $C$ and triangle $I$. Hence it is given by

$$\mathbf{n}_I = -\frac{\hat{\mathbf{l}}_{kj}}{l_{kj}} \tag{8}$$

Next we apply the mid-point rule approximation that is

$$\begin{aligned}&\mathbf{n}_I \cdot \nabla\phi(\mathbf{e}_I) l_{kj} + \mathbf{n}_J \cdot \nabla\phi(\mathbf{e}_J) l_{ik} + \mathbf{n}_K \cdot \nabla\phi(\mathbf{e}_K) l_{ji} \\ &= \mathbf{n}_I \cdot \mathbf{m}(\mathbf{e}_I) l_{kj} + \mathbf{n}_J \cdot \mathbf{m}(\mathbf{e}_J) l_{ik} + \mathbf{n}_K \cdot \mathbf{m}(\mathbf{e}_K) l_{ji}\end{aligned} \tag{9}$$

where $\mathbf{e}_I$ means the midpoint of edge $I$. That is $\mathbf{e}_I = \frac{\mathbf{x}_k + \mathbf{x}_j}{2}$. Now the problem is that there do not exist a $\phi$-value at the $\mathbf{e}$'s.. However, we may approximate the directional derivative at $\mathbf{e}_I$ in the direction of $\mathbf{n}_I$ by a Taylor approximation (see Appendix A for details),

$$\mathbf{n}_I \cdot \nabla\phi(\mathbf{e}_I) \approx \frac{\phi_I - \phi_C}{l_{CI}} \tag{10}$$

So we finally have the numerical stencil

$$\underbrace{\sum_{I \in \mathcal{N}} l_{kj} \frac{\phi_I - \phi_C}{l_{CI}}}_{\equiv \mathbf{A}_{CC}\phi_C + \mathbf{A}_{CI}\phi_I + \mathbf{A}_{CJ}\phi_J + \mathbf{A}_{CK}\phi_K} = \underbrace{\sum_{I \in \mathcal{N}} l_{kj} \mathbf{n}_I \cdot \mathbf{m}(\mathbf{e}_I)}_{\equiv \mathbf{b}_C} \tag{11}$$

That is the numerical stencil results in a four banded symmetric positive semi definite matrix given by the $A$-terms below

$$\mathbf{A}_{CC}\phi_C + \mathbf{A}_{CI}\phi_I + \mathbf{A}_{CJ}\phi_J + \mathbf{A}_{CK}\phi_K = \mathbf{b}_C \tag{12}$$

where

$$\mathbf{A}_{CC} = -\frac{l_{kj}}{l_{CI}} - \frac{l_{ik}}{l_{CJ}} - \frac{l_{ji}}{l_{CK}}, \tag{13a}$$

$$\mathbf{A}_{CI} = \frac{l_{kj}}{l_{CI}}, \tag{13b}$$

$$\mathbf{A}_{CJ} = \frac{l_{ik}}{l_{CJ}}, \tag{13c}$$

$$\mathbf{A}_{CK} = \frac{l_{ji}}{l_{CK}}. \tag{13d}$$

We omit further details on the right hand side vector $\mathbf{b}_c$ as it is trivial to evaluate given the closed-form solution of a given known constant $\mathbf{m}$-field. Defining triangle boundaries to coincide perfectly with the magnet boundary is the unique trait that allow us to treat $\mathbf{b}_C$ in this simple fashion. What we are missing now is how to deal with the higher order Neumann boundary conditions. This takes a little more work.

### 2.1 Higher Order Neumann Condition

We will now address the case of having a triangle $C$ with one open boundary edge. We assume without loss of generality that the boundary edge is $I$. On this edge we must apply the second order Neumann condition. From (12) we immediately notice a problem as the numerical stencil is referencing the value $\phi_I$ and that value does actually not exist in the mesh. To circumvent this we imagine a ghost triangle (cell) being placed across the boundary. Hence $\phi_I$ are the imaginary value stored in this ghost cell.

We will now derive the closed-form solution of the discrete ghost value $\phi_I$. We extrapolate this value under the assumption that the gradient $\nabla\phi$ remains unchained across the solid boundary. This assumption is equivalent to the second order Neumann condition we wish to apply on that boundary.

We first recover the value of $\nabla\phi$ at cell $C$ using the known directional gradient derivatives

$$\mathbf{n}_J^T \nabla\phi = \frac{\phi_J - \phi_C}{l_{JC}} \tag{14}$$

$$\mathbf{n}_K^T \nabla\phi = \frac{\phi_K - \phi_C}{l_{KC}} \tag{15}$$

and then use it to compute the value of $\phi_I$ from

$$\mathbf{n}_I^T \nabla\phi = \frac{\phi_I - \phi_C}{l_{IC}}. \tag{16}$$

The two expressions for directional derivatives (14)–(15) form a system of two linear equations with two unknowns (the two coordinates of $\nabla\phi$). By rewriting them in scalar form, we obtain

$$n_J^x \nabla\phi^x + n_J^y \nabla\phi^y = \frac{\phi_J - \phi_C}{l_{JC}} \tag{17}$$

$$n_K^x \nabla\phi^x + n_K^y \nabla\phi^y = \frac{\phi_K - \phi_C}{l_{KC}}. \tag{18}$$

By multiplying Eq. (17) by $n_K^y$ and Eq. (18) by $n_J^y$, and then subtracting (18) from (17) we eliminate $\nabla\phi^y$ from the system and find

$$\begin{aligned}\left(n_J^x n_K^y - n_J^y n_K^x\right) \nabla\phi^x = &\left(\frac{\phi_J - \phi_C}{l_{JC}}\right) n_K^y \\ &- \left(\frac{\phi_K - \phi_C}{l_{KC}}\right) n_J^y.\end{aligned} \tag{19}$$

Let us stop and look at the coefficient at $\nabla\phi^x$. By definition of the cross product between $\mathbf{n}_J$ and $\mathbf{n}_K$ we obtain

$$n_J^x n_K^y - n_J^y n_K^x = \mathbf{n}_J \times \mathbf{n}_K = \mathbf{n}_K^T \hat{\mathbf{n}}_J, \tag{20}$$

Clearly if the triangle is non-degenerate cross products of normal will be non-zero as any two normal vectors never will be parallel vectors. Hence we can make a division and write

$$\nabla\phi^x = \frac{n_K^y}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_J - \phi_C}{l_{JC}}\right) - \frac{n_J^y}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_K - \phi_C}{l_{KC}}\right). \tag{21}$$

By analogy, we obtain

$$\nabla\phi^y = -\frac{n_K^x}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_J - \phi_C}{l_{JC}}\right) + \frac{n_J^x}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_K - \phi_C}{l_{KC}}\right). \tag{22}$$

By putting these values together, we find that

$$\nabla\phi = \frac{\hat{\mathbf{n}}_J}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_K - \phi_C}{l_{KC}}\right) - \frac{\hat{\mathbf{n}}_K}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_J - \phi_C}{l_{JC}}\right). \tag{23}$$

Now we can substitute this value into (16)

$$\frac{\phi_I - \phi_C}{l_{IC}} = \mathbf{n}_I^T \nabla\phi \tag{24}$$

$$= -\frac{\mathbf{n}_I^T \hat{\mathbf{n}}_K}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_J - \phi_C}{l_{JC}}\right) - \frac{\mathbf{n}_J^T \hat{\mathbf{n}}_I}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \left(\frac{\phi_K - \phi_C}{l_{KC}}\right). \tag{25}$$

Therefore, the closed-form solution for $\phi_I$ reads

$$\begin{aligned}\phi_I = &\left(1 + \frac{\mathbf{n}_I^T \hat{\mathbf{n}}_K}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \frac{l_{IC}}{l_{JC}} + \frac{\mathbf{n}_J^T \hat{\mathbf{n}}_I}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \frac{l_{IC}}{l_{KC}}\right) \phi_C \\ &- \frac{\mathbf{n}_I^T \hat{\mathbf{n}}_K}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \frac{l_{IC}}{l_{JC}} \phi_J \\ &- \frac{\mathbf{n}_J^T \hat{\mathbf{n}}_I}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} \frac{l_{IC}}{l_{KC}} \phi_K.\end{aligned} \tag{26}$$

Kenny says: Observe that $l_{IC}$ is actually not known as $\mathbf{x}_I$ does not exist. For now in our derivation we simply use the edge midpoint $\mathbf{x}_I$ to define $l_{IC}$. Later we will observe that the $l_{IC}$ terms cancel out. An alternative route leading to the same solution is by expressing $\mathbf{n}_I$ as a linear combination of $\mathbf{n}_J$ and $\mathbf{n}_K$:

$$\mathbf{n}_I = a\,\mathbf{n}_J + b\,\mathbf{n}_K. \tag{27}$$

We find $a$ by transposing the last equation and multiplying both sides by $\hat{\mathbf{n}}_K$ (orthogonal to $\mathbf{n}_K$)

$$\mathbf{n}_I^T \hat{\mathbf{n}}_K = a\, \mathbf{n}_J^T \hat{\mathbf{n}}_K + b\, \mathbf{n}_K^T \hat{\mathbf{n}}_K = a\, \mathbf{n}_J^T \hat{\mathbf{n}}_K, \tag{28}$$

hence

$$a = \frac{\mathbf{n}_I^T \hat{\mathbf{n}}_K}{\mathbf{n}_J^T \hat{\mathbf{n}}_K} = -\frac{\mathbf{n}_I^T \hat{\mathbf{n}}_K}{\mathbf{n}_K^T \hat{\mathbf{n}}_J}. \tag{29}$$

Now by transposing (27) and multiplying onto $\hat{\mathbf{n}}_J$ we find

$$b = \frac{\mathbf{n}_I^T \hat{\mathbf{n}}_J}{\mathbf{n}_K^T \hat{\mathbf{n}}_J} = -\frac{\mathbf{n}_J^T \hat{\mathbf{n}}_I}{\mathbf{n}_K^T \hat{\mathbf{n}}_J}. \tag{30}$$

Having found $a$ and $b$, we can substitute them into (16)

$$\frac{\phi_I - \phi_C}{l_{IC}} = \mathbf{n}_I^T \nabla\phi = a\, \mathbf{n}_J^T \nabla\phi + b\, \mathbf{n}_K^T \nabla\phi \tag{31}$$

Now substituting (14) and (15) yields the exact same closed-form formula as the one in (26). Note that both derivations hinge on the assumption that the expressions for directional derivatives (14)–(16) are accurate, i.e. when all triangles in the mesh are nearly equilateral.

Having two open boundary edges is a degenerate case. However we can resolve this easily in the moving mesh framework by applying an edge-split on the inner edge on all triangles having two boundary edges. Hence we can safely assume that a triangle cell can have at most one boundary edge.

### 2.2 An Implementation Friendly Version

Let us now derive a generic implementation friendly equation. First we introduce the auxiliary notation for (26) as follows

$$\phi_I = B_{IC}\phi_C + B_{IJ}\phi_J + B_{IK}\phi_K \tag{32}$$

We refer to the $B$-values as the boundary coefficients. In case $J$ or $K$ where boundary edges we would find

$$\phi_J = B_{JC}\phi_C + B_{JI}\phi_I + B_{JK}\phi_K, \tag{33a}$$
$$\phi_K = B_{KC}\phi_C + B_{KJ}\phi_J + B_{KI}\phi_I \tag{33b}$$

Given that edge $I$ is a boundary edge one must substitute (32) into (12) to obtain the final modified stencil that takes the boundary conditions into account. The result is

$$\mathbf{A}'_{CC}\,\phi_C + \mathbf{A}'_{CI}\,\phi_I + \mathbf{A}'_{CJ}\,\phi_J + \mathbf{A}'_{CK}\,\phi_K = \mathbf{b}_C \tag{34}$$

where

$$\mathbf{A}'_{CC} = \mathbf{A}_{CC} + \mathbf{A}_{CI}\,B_{IC}, \tag{35a}$$
$$\mathbf{A}'_{CI} = 0, \tag{35b}$$
$$\mathbf{A}'_{CJ} = \mathbf{A}_{CJ} + \mathbf{A}_{CI}\,B_{IJ}, \tag{35c}$$
$$\mathbf{A}'_{CK} = \mathbf{A}_{CK} + \mathbf{A}_{CI}\,B_{IK}. \tag{35d}$$

Kenny says: Recall our previous comment about $l_{IC}$ terms cancel out. Observe above that the $\mathbf{B}$-terms and $l_{IC}$ in numerators whereas the $\mathbf{A}$-terms have $l_{IC}$-terms in denominators. Hence multiplication of $\mathbf{A}$ and $\mathbf{B}$ terms will cancel out $l_{IC}$-terms. Therefore the actual value of $l_{IC}$ is of lesser importance. Now let us generalize the concept and define the $C$ modification coefficients as follows

$$C_{AB} = \begin{cases} -1 & \text{if } A = B \text{ and } A \text{ is on boundary} \\ B_{AB} & \text{if } A \text{ is on boundary} \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

where $A \in \{I, J, K\}$ and $B \in \{I, J, K, C\}$. Now all cases can be handled using

$$\mathbf{A}'_{CC} = \mathbf{A}_{CC} + \mathbf{A}_{CI}\,C_{IC} + \mathbf{A}_{CJ}\,C_{JC} + \mathbf{A}_{CK}\,C_{KC}, \tag{37a}$$
$$\mathbf{A}'_{CI} = \mathbf{A}_{CI} + \mathbf{A}_{CI}\,C_{II} + \mathbf{A}_{CJ}\,C_{JI} + \mathbf{A}_{CK}\,C_{KI}, \tag{37b}$$
$$\mathbf{A}'_{CJ} = \mathbf{A}_{CJ} + \mathbf{A}_{CI}\,C_{IJ} + \mathbf{A}_{CJ}\,C_{JJ} + \mathbf{A}_{CK}\,C_{KJ}, \tag{37c}$$
$$\mathbf{A}'_{CK} = \mathbf{A}_{CK} + \mathbf{A}_{CI}\,C_{IK} + \mathbf{A}_{CJ}\,C_{JK} + \mathbf{A}_{CK}\,C_{KK}. \tag{37d}$$

The benefit is that all cases are encoded into the $C$-definition which can be implemented using question-mark statements. Hence by paying for 12 additional temporary local variables one can avoid witting a lot of if-statements causing thread divergence. This is an advantage as the matrix assembly is data parallel and the above computations can be done naively parallel. Even in the case of matrix free methods one can make a massively parallel operator for evaluating the matrix-vector product $\mathbf{A}\,\phi$ without any thread divergence. Of course using locals could have the side-effect of register usage becoming the limiting resource bottleneck on certain GPUs.

Kenny says: Obviously the magneto static potential problem with 1st order Neumann conditions will be symmetric. Clearly the fill-in for the 2nd order Neumann condition will be symmetric too (same memory layout), however the values stored in the modified system are no longer symmetric. (Can we prove this formally?)

## A. The Approximation of Directional Derivative

Let us revisit the approximation

$$\mathbf{n}_I \cdot \nabla\phi(\mathbf{e}_I) \approx \frac{\phi_I - \phi_C}{l_{CI}}. \tag{38}$$

We assume that the geometry of the mesh is such that

$$\mathbf{x}_I = \mathbf{e}_I + \mathbf{n}_I\, l_{CI}\,(1-\gamma)\,, \tag{39a}$$
$$\mathbf{x}_C = \mathbf{e}_I - \mathbf{n}_I\, l_{CI}\,\gamma, \tag{39b}$$

where $0 < \gamma < 1$ is some known constant. Now we use Taylor series approximations around $\phi(\mathbf{e}_I)$

$$\phi(\mathbf{x}_I) = \phi(\mathbf{e}_I) + \nabla\phi(\mathbf{e}_I)\cdot\mathbf{n}_I\, l_{CI}\,(1-\gamma) + \mathcal{O}\left(l_{CI}^2\right), \tag{40a}$$
$$\phi(\mathbf{x}_C) = \phi(\mathbf{e}_I) - \nabla\phi(\mathbf{e}_I)\cdot\mathbf{n}_I\, l_{CI}\,\gamma + \mathcal{O}\left(l_{CI}^2\right). \tag{40b}$$

Subtracting the two approximations we have

$$\phi(\mathbf{x}_I) - \phi(\mathbf{x}_C) = \nabla\phi(\mathbf{e}_I)\cdot\mathbf{n}_I\, l_{CI} + \mathcal{O}\left(l_{CI}^2\right). \tag{41}$$

Cleaning up we find

$$\nabla\phi(\mathbf{e}_I)\cdot\mathbf{n}_I = \frac{\phi(\mathbf{x}_I) - \phi(\mathbf{x}_C)}{l_{CI}} + \mathcal{O}\left(l_{CI}\right). \tag{42}$$

Hence we find the first-order approximation of the directional derivative to be

$$\nabla\phi(\mathbf{e}_I)\cdot\mathbf{n}_I \approx \frac{\phi(\mathbf{x}_I) - \phi(\mathbf{x}_C)}{l_{CI}}. \tag{43}$$

Besides our discretization error we are making the assumptions that the line connecting $\mathbf{x}_I$ and $\mathbf{x}_C$ goes orthogonal through the edge mid-point $\mathbf{e}_I$. This requires a very special mesh with only equilateral triangles. Real-life meshes do not have this property all over. Hence we are making some errors too due to our geometry assumptions.

If we assume the size of the elements is sufficiently small that we can assume a linear approximation over the elements then a reasonable measure of the error we make by assuming the line between the cell centers are orthogonal can be

defined as

$$\varepsilon_I \equiv \sin(\theta_I) = \mathbf{n}_I \cdot \frac{\mathbf{l}_{IC}}{l_{IC}} \tag{44}$$

## B. Mesh Control

The error measure $\varepsilon_I$ can be used for mesh control in such a way to make $\varepsilon_I \to 0$. Another way to control the mesh is such that the assumption that $\phi$ behaves sufficiently linear on the triangle $C$. One way to measure if this property is fulfilled is to make a linear model based on a small patch of $\phi$ samples and then measure how well $\phi_C$ fits with this model. For an arbitrary triangle let $\mathcal{V}_T$ denote its vertex set. hence $\mathcal{V}_C$ is $\{i, j, k\}$ and let label$(T)$ be a mapping of triangles to whether they belong to the magnetic or non-magnetic region of the computational domain. Now we can define the one ring restricted neighbourhood patch of $C$ to be

$$\mathcal{P}_c \equiv \{n \,|\, \mathcal{V}_n \cap \mathcal{V}_c \neq \emptyset \wedge \text{label}(n) = \text{label}(c)\} \tag{45}$$

Define the coefficient vector as $\mathbf{c} \equiv (c_0, c_1, c_2)^T$ then a linear model of the patch is computed using linear least squares error

$$\mathbf{c}^* \equiv \arg\min_{\mathbf{c}} \underbrace{\frac{1}{2} \sum_{T \in \mathcal{P}_C} (c_0 + c_1\, x_T + c_2\, y_T - \phi_T)^2}_{f(\mathbf{c})} \tag{46}$$

Knowing $\mathbf{c}^*$ we have a linear "best" model of the patch

$$L(x, y; \mathbf{c}^*) \equiv c_0^* + c_1^* x + c_2^* y \tag{47}$$

From which we may estimate the sufficiently small assumption as

$$\epsilon_C = \left(L(x_C, y_C; \mathbf{c}^*) - \phi_C\right)^2 A_c \tag{48}$$

Here we use the square error to get positive measures and we weight by area to penalize too large regions. This is a somewhat heuristic based measure found through trial and error but in our experience mesh quality seems to behave nice with this measure of quality. Alternatively one may simply use the value of $f(\mathbf{c}^*)$ as the error measure. This is by design non-negative and bounded from below by zero. The objective function includes the "area" too as all samples of the patch region contributes to the error measure.

To solve for the first order optimal coefficients $\mathbf{c}^*$ we need to find a formula for the gradient of the objective function which we derive from calculus to be

$$\nabla f \equiv \sum_{T \in \mathcal{P}_C} L(x_T, y_T, \mathbf{c}) \begin{bmatrix} 1 \\ x_T \\ y_T \end{bmatrix} \tag{49}$$

Without loss of generality assume indices of $\mathcal{P}_C$ to be $\{1, 2, \ldots, T\}$ then we may introduce the matrix and vector

$$\mathbf{B} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ & \vdots & \\ 1 & x_T & y_T \end{bmatrix} \quad \text{and} \quad \mathbf{\Phi} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_T \end{bmatrix}. \tag{50}$$

We may now write the objective gradient using this new notation as

$$\nabla f = \mathbf{B}^T \mathbf{B} \mathbf{c} + \mathbf{B}^T \mathbf{\Phi} \tag{51}$$

Solving for $\nabla f = \mathbf{0}$ is now done solving the 3-by-3 system

$$\left(\mathbf{B}^T \mathbf{B}\right) \mathbf{c} = -\mathbf{B}^T \mathbf{\Phi} \tag{52}$$

To be numerically well posed this requires the patch to have at least three elements, $\| \mathcal{P}_C \| \geq 3$. If this is the case then $\left(\mathbf{B}^T \mathbf{B}\right)$ is guaranteed to be symmetric positive definite matrix and hence non-singular. If less than three elements the matrix will be positive semi-definite and hence singular. In other words the patch is too small for the linear model to make sense the most sensible solution would be to grow the patch region or add some other assumptions for this specific case. Note that a patch with exactly three elements will always yield a perfect fit to a linear model. Hence it makes sense to require that a patch has at least 4 (or more) elements to be able to test how well a linear model fits with the $\phi$-field.

## C. Jumping Gradients

One should observe that $\nabla\phi$ is non-smooth across magnet and void boundaries hence our normal derivative approximation across those edges are strictly speaking not valid in classical sense. If we apply non-smooth theory and replace directional derivatives with B-differentials then everything holds. Alternatively we could use our open boundary approach on those edges to force a one-sided boundary condition on the gradient from magnet region and void region respectively. However, we would need to add a Dirichlet condition as well to ensure that the $\phi$-field stays continuous and do not jump across the magnet and void boundaries. We decided to go with the non-smooth analysis argumentation although this could potentially add some smoothness to the solution on the magnet-void boundaries. However, on those boundaries we do control the mesh element size to become very small hence the error we make can be made as sufficiently small as we desire.