Projektbericht

Finite Elemente Methode auf einer
2-dimensionalen Helmholtz Gleichung

Juri Chomé

8. Februar 2010

Zusammenfassung

Wir betrachten die Maxwell-Gleichungen für das Elektromagnetische Feld in einem Mikrowellenherd und interessieren uns für die dritte Komponente des Elektrischen Feldes. Diese entspricht einer 2-dimensionalen Helmholtz Gleichung. Mit Hilfe der Finiten-Elemente-Methode geht es dann darum, die Lösung dieser Gleichung in *Matlab* zu implementieren. Dazu muss zuerst die schwache Formulierung des Problems hergeleitet werden und die Steifigkeitsmatrix und die Massenmatrix aus den jeweiligen Bilinearformen berechnet werden. Gegeben ist eine Triangulierung eines Fisches und der restlichen Mikrowelle als Liste von Knoten mit Materialinformationen. Die Triangulierung kann verfeinert werden, und die Matrizen werden möglichst effizient berechnet und gelöst. Zuletzt wird der Realteil der Lösung gezeichnet.

1 Formulierung des Problems

1.1 Die Ausgangslage

Gegeben sind die Maxwell-Gleichungen auf $\Omega \subset \mathbb{R}^3$ für das elektrische Feld $\mathbf{E} : \Omega \to \mathbb{C}^3$ und das magnetische Feld $\mathbf{H} : \Omega \to \mathbb{C}^3$:

$$\nabla \times \mathbf{E} = -\mu \mathbf{H}_t \tag{1}$$
$$\nabla \times \mathbf{H} = \varepsilon \mathbf{E}_t + \sigma \mathbf{E}. \tag{2}$$

Hier entsprechen ε, σ und μ der dielektrischen Leitfähigkeit, der Konduktivität und der Permeabilität und sind im Aufgabenblatt jeweils in der Luft und im Fisch konkret definiert. Vorgegeben wird zudem, dass die Lösung zeit-harmonisch ist:

$$\mathbf{E} = \mathbf{E}(x, y, z)e^{i\omega t} \text{ und } \mathbf{H} = \mathbf{H}(x, y, z)e^{i\omega t}.$$

1.2 Herleitung

Mit dem Kreuzprodukt und der Periodizität der Gleichungen erhalten wir

$$(\nabla \times \mathbf{E}) = \begin{pmatrix} \partial_y \mathbf{E}_3 - \partial_z \mathbf{E}_2 \\ \partial_z \mathbf{E}_1 - \partial_x \mathbf{E}_3 \\ \partial_x \mathbf{E}_2 - \partial_y \mathbf{E}_1 \end{pmatrix} = (\nabla \times \mathbf{E}) \cdot e^{i\omega t}$$

auf der linken Seite und

$$-\mu \mathbf{H}_t = -(i\omega)\mu \mathbf{H} \cdot e^{i\omega t}$$

auf der rechten Seite. Also lautet (1) nun

$$\nabla \times \mathbf{E} = -i\omega\mu \mathbf{H}.$$

Das Gleiche machen wir für (2) und erhalten so

$$(\nabla \times \mathbf{H}) \cdot e^{i\omega t} = (\varepsilon i\omega + \sigma)\mathbf{E} \cdot e^{i\omega t}.$$

Also lautet unser Gleichungssystem nun

$$\begin{aligned} \nabla \times \mathbf{E} &= -i\omega\mu \mathbf{H} \\ \nabla \times \mathbf{H} &= i\omega\tilde{\varepsilon}\mathbf{E} \qquad\qquad \text{in } \Omega \subset \mathbb{R}^3 \end{aligned}$$

für $\tilde{\varepsilon} := \varepsilon + \frac{\sigma}{i\omega}$. Als zweite Annahme wird angegeben, dass $\partial_z \mathbf{E} = \partial_z \mathbf{H} = \vec{0}$ ist. Also gilt mit dem Kreuzprodukt jetzt

$$(\nabla \times \mathbf{E}) = \begin{pmatrix} \partial_y \mathbf{E}_3 \\ -\partial_x \mathbf{E}_3 \\ \partial_x \mathbf{E}_2 - \partial_y \mathbf{E}_1 \end{pmatrix} = -i\omega\mu \mathbf{H} \tag{3}$$

und

$$(\nabla \times \mathbf{H}) = \begin{pmatrix} \partial_y \mathbf{H}_3 \\ -\partial_x \mathbf{H}_3 \\ \partial_x \mathbf{H}_2 - \partial_y \mathbf{H}_1 \end{pmatrix} = i\omega\tilde{\varepsilon}\mathbf{E}. \tag{4}$$

Aus (3) und (4) ergeben sich folgende Identitäten:

$$-\partial_y \mathbf{E}_3 = i\omega\mu\mathbf{H}_1 \tag{5}$$
$$\partial_x \mathbf{E}_3 = i\omega\mu\mathbf{H}_2 \tag{6}$$
$$\partial_x \mathbf{H}_2 - \partial_y \mathbf{H}_1 = i\omega\tilde{\varepsilon}\mathbf{E}_3. \tag{7}$$

Setze (5) und (6) in (7) ein und multipliziere mit $i\omega\mu$:

$$\begin{aligned} & \partial_x^2 \mathbf{E}_3 + \partial_y^2 \mathbf{E}_3 = -\omega^2 \mu\tilde{\varepsilon}\mathbf{E}_3 \\ \iff & \Delta\mathbf{E}_3 + \omega^2\mu\tilde{\varepsilon}\mathbf{E}_3 = 0. \end{aligned}$$

Also stehen wir nun vor der 2-dimensionalen Helmholtz-Gleichung

$$\Delta u + (\omega^2\mu\tilde{\varepsilon})u = 0 \qquad \text{in } \Omega \subset \mathbb{R}^2 \tag{8}$$
$$u = g \qquad \text{auf } \partial\Omega. \tag{9}$$

2 Schwache Formulierung

Multipliziere wie immer (8) mit einer Testfunktion $v \in H_0^1(\Omega)$, integriere und nutze die Greensche Formeln:

$$\begin{aligned} & \int_\Omega \Delta u \cdot v + \omega^2\mu\tilde{\varepsilon} \int_\Omega u \cdot v = 0 \\ \iff & \underbrace{-\int_\Omega \nabla u \cdot \nabla v}_{=:\, a_1(u,v)} + \underbrace{\int_{\partial\Omega} v \cdot \frac{\partial u}{\partial \vec{N}}}_{\equiv 0,\ \text{da } v \equiv 0 \text{ auf } \partial\Omega} + \omega^2\mu\tilde{\varepsilon} \underbrace{\int_\Omega u \cdot v}_{=:\, a_2(u,v)} = 0. \end{aligned}$$

Wir erhalten also die Gleichung

$$a(u,v) = a_1(u,v) - \omega^2\mu\tilde{\varepsilon} \cdot a_2(u,v) = 0$$

und somit die schwache Formulierung: „Suche $u \in H^1(\Omega)$, so dass $a(u,v) = 0$ $\forall v \in H_0^1(\Omega)$ und $u \equiv g$ auf $\partial\Omega$“.

3 Implementierung

3.1 Methode

Wir benutzen eine lineare Finite-Elemente Methode auf Dreiecken welche die Eckpunkte nutzt. Dazu ist eine Liste N von Knotenpunkten gegeben (x

und y Koordinaten) und eine Liste von Dreiecken T, die sich auf N beruft um die Eckpunkte der Dreiecke zu definieren. Zusätzlich enthält T noch drei binärwertige Kolonnen, die angeben, ob sich die dazugehörige Kante auf dem Rand befindet oder nicht. Zuletzt enthält ein Vektor P für jedes Dreieck die dazugehörigen Materialkonstante $\mu(\varepsilon + \frac{\sigma}{i\omega})$, in unserem Fall entweder die für den Fisch oder die in der Luft.

In Abbildung 1 sieht man als Beispiel vier Dreiecke, von denen drei eine Kante am Rand besitzen (fett gedruckte Linie) und die insgesamt sechs verschiedene Punkte benötigen. Die Listen N und T sehen in dem Fall folgendermassen aus:

```
N =                 T =

    0         0         1     4     6     1     0     1
    1         0         4     5     6     0     0     0
    0         1         4     2     5     1     0     0
  0.5         0         6     5     3     0     0     1
  0.5       0.5
    0       0.5
```

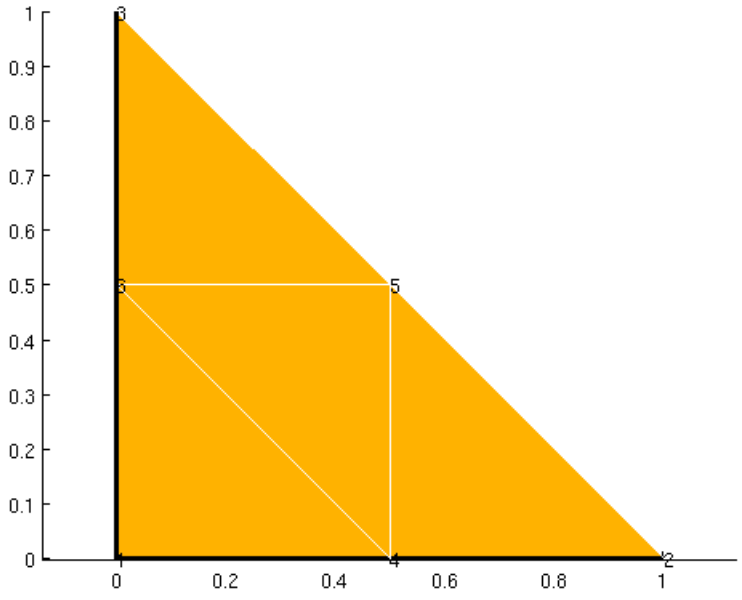

Abbildung 1: Ein 4-fach unterteiltes Dreieck

3.2 Gitterverfeinerung

Die Prozedur der Gitterverfeinerung erhält die Listen der Knotenpunkte N, die der Dreiecke T (inklusive Randinformationen) und die der dazugehörigen Materialkonstanten P. Das Ziel ist es ein gegebenes Gitter zu verfeinern indem jedes Dreieck in vier kleinere unterteilt wird und die dazu nötigen Knotenpunkte zu berechnen. Um die neue Liste der Punkte zu erstellen muss allerdings vermieden werden, Punkte doppelt zu definieren. Dazu wird eine einfache *for*-Schleife über die drei möglicherweise redundanten Punkte erstellt. In pseudo-Matlabcode sieht dies dann folgendermassen aus:

```
for i=1:size(T,1),
  uebernimm die gueltigen alten Eckpunkte;
  uebernimm Materialkonstanten;

  punkte_neu(1,1)=(x1+x2)/2; punkte_neu(1,2)=(y1+y2)/2;
  punkte_neu(2,1)=(x3+x2)/2; punkte_neu(2,2)=(y3+y2)/2;
  punkte_neu(3,1)=(x1+x3)/2; punkte_neu(3,2)=(y1+y3)/2;

  for j=1:3
    dupe=find(ismember(Nr,punkte_neu(j,:),'rows'));
    if isempty(dupe)
      Nr(size(N,1)+1,:)=punkte_neu(j,:);
      uebernimm die neuen Punkte;
    else
      uebernimm die alten Punkte;
    end
  end
end
```

Als Beispiel sieht man das Einheitsquadrat nach dreifacher Verfeinerung in Abbildung 2.

3.3 Matrizen

Um nicht für jedes Element die Basisfunktionen neu definieren zu müssen, wird eine Transformation auf das Referenzdreieck $\hat{K}$ vollzogen. Die Basisfunktionen lauten dort

$$\varphi_1(\xi_1, \xi_2) = 1 - \xi_1 - \xi_2$$
$$\varphi_2(\xi_1, \xi_2) = \xi_1$$
$$\varphi_3(\xi_1, \xi_2) = \xi_2.$$

Für die drei Eckpunkte P_1, P_2, P_3 von K lautet die affine Transformation $F_K := P_1 + (P_2 - P_1)\xi_1 + (P_3 - P_1)\xi_2$.

Elementsteifigkeitsmatrix

Mit der Herleitung die wir in der Vorlesung gesehen haben, lässt sich für a_1 die Elementsteifigkeitsmatrix berechnen mit

$$A_e = \frac{1}{2}|\det J| \cdot DD'$$

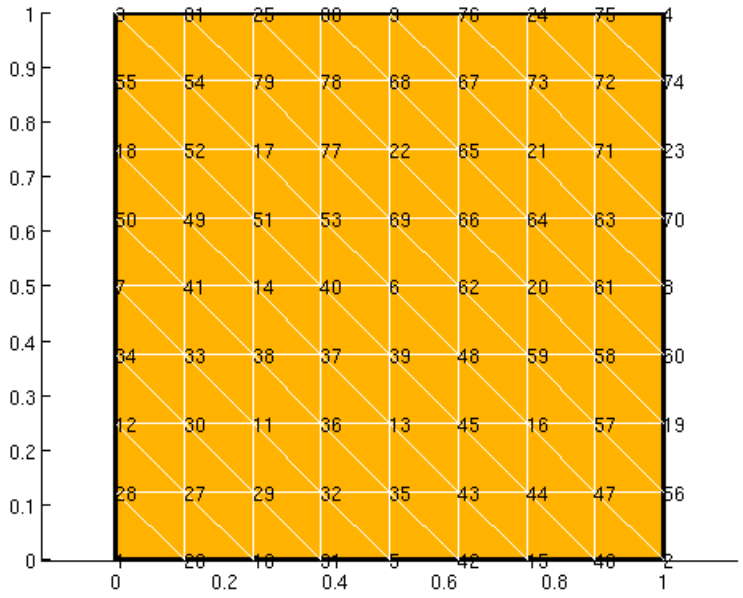

Abbildung 2: Das Gitter des dreifach verfeinerten Einheitsquadrates

wobei

$$J := \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}$$

und

$$D := \begin{pmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot J^{-1}.$$

In *Matlab* sieht das dann folgendermassen aus:

```
J=[N(T(el,2),1)-N(T(el,1),1)  N(T(el,3),1)-N(T(el,1),1);
   N(T(el,2),2)-N(T(el,1),2)  N(T(el,3),2)-N(T(el,1),2)];

D=[-ones(1,2);eye(2)]/J;

Ae=1/2*abs(det(J))*D*D';
```

Elementmassenmatrix

Auf $\hat{K}$ lauten die Einträge

$$a_2(i,j) := \int_{\hat{K}} \varphi_i \varphi_j \qquad\qquad i,j = 1,2,3.$$

Das Resultat ist $1/12$ auf den Diagonalen und $1/24$ sonst. Damit berechnen wir für jedes Element die Elementmassenmatrix

$$M_e = |\det J| \cdot a_2(i,j).$$

Der *Matlab* Code ist somit

```
J=[N(T(el,2),1)-N(T(el,1),1) N(T(el,3),1)-N(T(el,1),1);
   N(T(el,2),2)-N(T(el,1),2) N(T(el,3),2)-N(T(el,1),2)];

Me=[1/12 1/24 1/24;
    1/24 1/12 1/24;
    1/24 1/24 1/12];

Me=abs(det(J))*Me;
```

Assemblierung

Die Assemblierungsroutine läuft über alle Dreiecke, berechnet die beiden dazugehörigen Elementmatrizen und multipliziert M_e mit der entsprechenden Materialkonstante aus P. Um die Matrizen zusammenzustellen wird im Prinzip nur für jedes Element die Elementmatrix dazu addiert:

```
k=T(i,1:3);
A(k,k)=A(k,k)-Ae;
M(k,k)=M(k,k)+Me;
```

Das negative Vorzeichen der Steifigkeitsmatrix kommt von dem Minus und von der homogenität der Gleichung. Schlussendlich müssen noch die Randbedingung (9) beachtet werden. Man macht also eine Liste die prüft, ob ein Dreieck einen Randpunkt besitzt und erstellt eine Schleife über alle Knotenpunkte um die Matrizen zu modifizieren:

```
n=size(T,1);
m=size(N,1);
for i=1:n,
  bn(T(i,1))=bn(T(i,1)) | T(i,4) | T(i,6);
  bn(T(i,2))=bn(T(i,2)) | T(i,4) | T(i,5);
  bn(T(i,3))=bn(T(i,3)) | T(i,5) | T(i,6);
end
for j=1:m
  if bn(j)==1
    A(j,:)=0;
    A(j,j)= 1;
    M(j,:)=0;
    b(j)=feval(g,N(j,1),N(j,2));
  end
end
```

3.4 Ergebnisse

Testproblem

Mit $\omega = 0$ kann man die Implementierung testen, da die Gleichung dann ein Poisson-Problem ist. Gibt man sich die Randbedingung $g := x+y$, dann lau-

tet die exakte Lösung auf dem Einheitsquadrat ganz einfach $u(x, y) = x + y$, da $\Delta u = 0$ ergibt. In Abbildung 3 kann man sich davon überzeugen, dass *mainPoisson.m* tatsächlich genau diese Lösung ausspuckt. Die Gitterverfeinerung ist hier viermal auf das Einheitsquadrat angewendet worden.

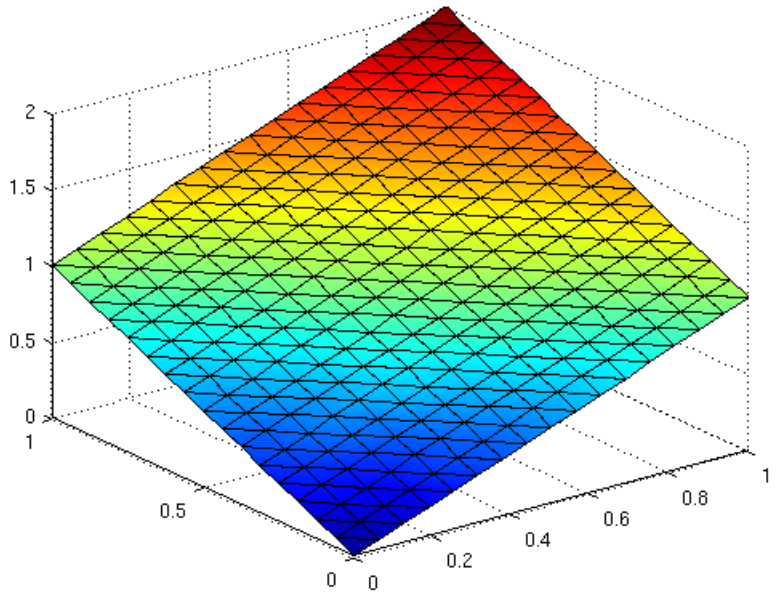

Abbildung 3: Die Lösung der Poisson Gleichung mit Matlab

Fisch

Die Randbedingung g ist in diesem Fall gegeben durch

$$g(x, y) = 100 \text{ auf dem Intervall } 0,5 \times [0,1; 0,2]$$
$$g(x, y) = 0 \text{ sonst.}$$

Die Konstanten lauten

$$\mu_F = 2\pi \cdot 10^{-7},$$
$$\varepsilon_F = 6,44 \cdot 10^{-10},$$
$$\sigma_F = 3 \cdot 10^{-11}$$
$$\mu_L = 2\pi \cdot 10^{-7},$$
$$\varepsilon_L = 8,85 \cdot 10^{-12},$$
$$\sigma_L = 0$$

jeweils auf dem Fisch und in der Luft.

Schliesslich lässt man die Routinen auf den Fisch in der Mikrowelle los und erhält nach vier Gitterverfeinerungen als Lösung die reellen Komponenten des elektrischen Feldes, die man in Abbildung 4 sehen kann, einmal nur auf dem Fisch, einmal aus isometrischer dreiviertel-Ansicht, und einmal von oben.

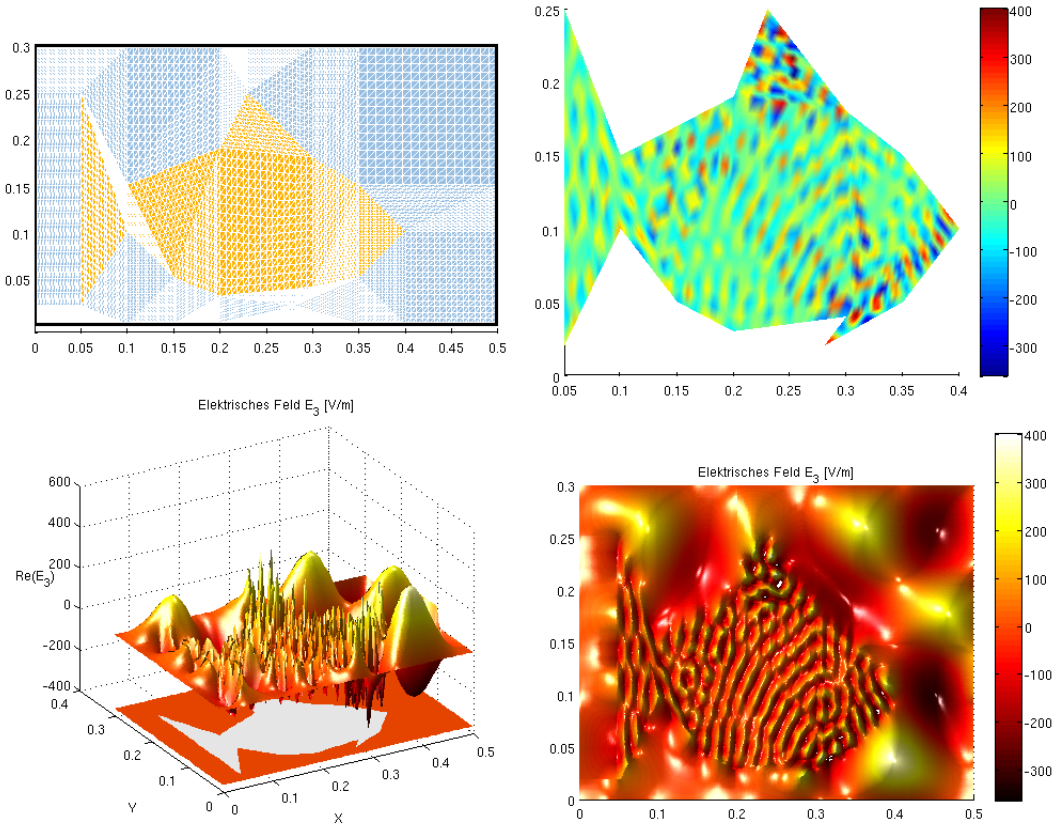

Abbildung 4: Das Resultat nach 4 Verfeinerungen mit dem dazugehörigen Gitter