1 Einführung in Numerische Methoden

1.1 Inhalt und Ziel des Versuchs

Im ersten Versuch des Praktikums werden allgemeine Grundlagen numerischer Methoden behandelt sowie Grundsätze zur Visualisierung veranschaulicht. Im Vordergrund steht hierbei die numerische Nachbildung der Differentiation am Beispiel der eindimensionalen Wellengleichung sowie die dreidimensionale Darstellung von Körpern am Beispiel der Dreiecksdarstellung von Oberflächen unter Verwendung der *Standard Transformation Language*.

1.2 Theorie

1.2.1 Differenzenverfahren

Die Simulation physikalischer und technischer Prozesse im Rechner, die in weiten Bereichen von Forschung und Entwicklung eine zentrale Schlüsselposition eingenommen hat, ist ein zweistufiger Prozess: Zum einen wird ein abstrahiertes Abbild der tatsächlichen Problemstellung geschaffen (*Modellbildung*), die die *Diskretisierung* der tatsächlichen Konfiguration sowie die Erstellung einer mathematischen Beschreibung (*Abstraktion, Systementwurf*) erfordert.

In diesem Prozess können veschiedene *Modellierungsfehler* auftreten: der Geometriefehler, der durch die vereinfachte Darstellung der tatsächlichen Konfiguration entsteht, der *Diskretisierungsfehler*, der bei fehlerhafter physikalischer Beschreibung (beispielsweise bei einem Fehler in der Materialbeschreibung) auftritt und der sogenannte *Abstraktionsfehler*, hervorgerufen durch die Simplifikation der mathematischen Darstellung (Vernachlässigung von Prozessen wie beispielsweise Temperaturabhängigkeit der Materialparameter, Verformung der Geometrie etc.).

Im zweiten Schritt wird mit Hilfe numerischer Methoden die modellierte Problemstellung gelöst. Aufgrund der Eigenschaften der verwendeten Lösungsmethoden kann es hierbei zu *Verfahrensfehlern* sowie zu *numerischen Fehlern* durch die begrenzte Darstellungsgenauigkeit des Rechners kommen.

Numerische Methoden in der Feldtheorie lassen sich in zwei Klassen unterteilen:

- integrale Methoden
- differentielle Methoden

Während die integralen Verfahren wie die *Momentenmethode* (MoM von engl. *Method of Moments*) oder die *Randelementemethode* (BEM von engl. *Boundary Element Method*) die Wirkung von Feldquellen (Ladungen oder Ströme) auf jeden Raumpunkt meist mit Hilfe semianalytischer Methoden und Superpositionsprinzipien berechnen und kontinuierliche Lösungsbeschreibungen liefern, zielen differentielle Methoden wie *Finite Integration* (FIT), *Finite Differenzen* (FD) oder *Finite Elemente* (FE) auf die Berechnung des Feldes mit Hilfe eines Rechengitters an diskreten Stützstellen (diskrete Feldlösung) oder in jedem Raumpunkt (kontinuierliche Lösung) ab.

Differenzenverfahren wie Sie die Methode der Finiten Differenzen und in gewissem Aspekt auch die Finite Integrationsmethode darstellt, lösen Differentialgleichungen, indem die Differentiationsvorschriften numerisch nachgebildet und an Stützstellen ausgewertet werden. Sie sind relativ einfach zu konstruieren, robust und effizient zu implementieren.

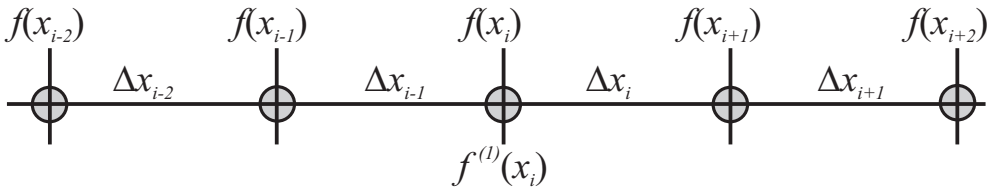

Abbildung 1.1: Allokation der Funktionswerte $f(x_i)$. Die Funktionswerte $f(x_i)$ werden an den diskreten Stützstellen x_i allokiert, die Bezeichnung der Abstände zwischen den Stützstellen sind durch Δx_i gegeben.

Die Grundlage dieser Methode ist die Darstellung von Werten einer Funktion an diskreten Stützstellen x_i (siehe Abb. 1.1). Mit Hilfe der Taylor-Entwicklung um Entwicklungspunkt a

$$f(a) + \frac{(x-a)}{1!}\left.\frac{\mathrm{d}f(x)}{\mathrm{d}x}\right|_{x=a} + \frac{(x-a)^2}{2!}\left.\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2}\right|_{x=a} + \ldots, \tag{1.1}$$

lässt sich, ausgehend von einem Entwicklungspunkt x_i, die Größe des Feldwertes an der Stelle $x_{i+1} = x_i + \Delta x_i$ mittels

$$f(x_{i+1}) = \sum_{n=0}^{N} \frac{\Delta x_i^n}{n!}\left.\frac{\mathrm{d}^n f(x)}{\mathrm{d}\,x^n}\right|_{x=x_i} + \mathcal{O}(\Delta x^{N+1}) \tag{1.2}$$

ausdrücken. Der Abbruchfehler skaliert mit der Größe Δx^{N+1} und stellt sich somit als eine in Abhängigkeit der Gitterschrittweite definierte Größe dar, die im Grenzübergang $\lim_{\Delta x \to 0}$ verschwindet. Die Potenz des Abbruchfehlers, hier $N+1$, bezeichnet man als *Fehlerordnung* der Näherung.

Numerische Differentiation erster Ordnung

Sind die Stützwerte einer Funktion bekannt, so kann mit Hilfe von Gl. (1.2) der Wert der ersten Ableitung $f^{(1)}(x_i)$ am Entwicklungspunkt x_i ermittelt werden. Je nach Konstruktion des Verfahrens unterscheidet man zwischen:

- **Vorwärts-Differenzenquotient**: Bringt man $f(x_i)$ auf die linke Seite von Gl. (1.2) und löst nach der ersten Ableitung $f^{(1)}(x_i)$ auf, erhält man die Näherung für die erste Ableitung

$$f^{(1)}(x_i) = \frac{f(x_{i+1}) - f(x_i)}{\Delta x_i} + \mathcal{O}(\Delta x)\,. \tag{1.3}$$

- **Rückwärts-Differenzenquotient:** Analog zum Vorwärts-Differenzenquotienten entwickelt man eine Taylor-Reihe zum Feldwert an der Stelle x_{i-1}. Durch Subtraktion von $f(x_i)$ erhält man

$$f^{(1)}(x_i) = \frac{f(x_i) - f(x_{i-1})}{\Delta x_{i-1}} + \mathcal{O}(\Delta x)\,. \tag{1.4}$$

- **Zentraler Differenzenquotient:** Werden zwei Taylor-Entwicklungen vom Startpunkt x_i zu den Punkten x_{i+1} und x_{i-1} voneinander subtrahiert, ergibt sich der sogenannte zentrale Differenzenquotient

$$f^{(1)}(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{\Delta x_{i-1} + \Delta x_i} + \mathcal{O}(\Delta x^{1\cdots 2})\,, \tag{1.5}$$

der die erste Ableitung bei jeweils gleichem Rechenaufwand mit einer Genauigkeit erster bis zweiter Ordnung nähert. Er approximiert die Ableitung, ohne den Wert der Funktion an der Stützstelle zu berücksichtigen. Eine Näherung zweiter Ordnung wird bei äquidistanten Gitterschrittweiten ($\Delta x = \Delta x_i = \Delta x_{i+1} = \ldots$) erzielt.

Der zentrale Differenzenquotient findet häufig Anwendung bei der numerischen Lösung zweier verkoppelter Differentialgleichungen erster Ordnung, wie zum Beispiel bei den ersten beiden Maxwellschen Gleichungen. Diese verknüpfen die räumlichen Ableitungen elektrischer mit der zeitlichen Ableitung magnetischer Felder und umgekehrt. Folglich müssen an den Punkten, an denen die räumlichen Ableitungen einer Größe ausgewertet werden, die hierzu gehörigen anderen Größen allokiert sein. Dies motiviert die Einführung eines sogenannten *dualen Gitters*.

In Anlehnung an das weitverbreitete Verfahren der Finiten Differenzen und der Methode der Finiten Integration wird ein duales Gitter eingeführt, das in Abb. 1.2 dargestellt ist. Der zentrale Differenzenquotient liefert an der Stelle $\tilde{x}_i = x_{i+1/2} = (x_i + x_{i+1})/2$ des dualen Gitters die Bestimmungsgleichung zweiter Ordnung:

$$f^{(1)}(\tilde{x}_i) = \frac{f(x_{i+1}) - f(x_i)}{\Delta x_i} + \mathcal{O}(\Delta x^2)\,. \tag{1.6}$$

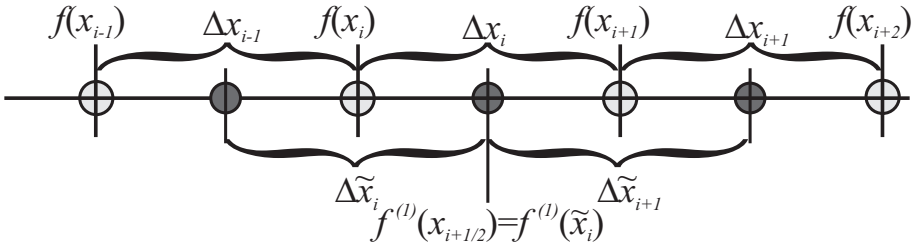

Abbildung 1.2: Einführung eines dualen Gitters. Zwischen den Stützstellen x_i werden Zwischenstellen $x_{i+1/2}$ geschaffen, an denen die Ableitungen $f^{(1)}(x_{i+1/2})$ und $f^{(2)}(x_{i+1/2})$ allokiert werden. Die Lage und die Abstände dieser dualen Gitterpunkte werden durch $\tilde{x}_i = (x_i + x_{i+1})/2$ sowie $\Delta\tilde{x}_i = (\Delta x_i + \Delta x_{i-1})/2$ festgelegt.

Um eine bessere Genauigkeit bei der Berechnung der Ableitungen zu erhalten, können Differenzenquotienten höherer Ordnung konstruiert werden. Ausgehend von einem Startpunkt $f(\tilde{x}_i)$ können Taylor-Entwicklungen zu den vier benachbarten Punkten $f(x_i)$, $f(x_{i-1})$, $f(x_{i+1})$ und $f(x_{i+2})$ aufgestellt und durch Auflösung des resultierenden Gleichungssystems eine Näherung der ersten Ableitung bis zur vierten Ordnung ermittelt werden.

Numerische Differentiation zweiter Ordnung

Ableitungen zweiter Ordnung werden bei der direkten Lösung der Wellengleichung (Helmholtz-Gleichung) benötigt:

$$\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2} = -k^2 f(x) \tag{1.7}$$

k ist hierbei die Wellenzahl. Zu Gl. (1.7) kann durch die Addition zweier Taylor-Entwicklungen, die vom Startpunkt x_i zu den Punkten x_{i+1} sowie x_{i-1} entwickelt werden, eine Näherung

$$f^{(2)}(x_i) = 2\frac{f(x_{i+1})\Delta x_{i-1} + f(x_{i-1})\Delta x_i - f(x_i)(\Delta x_i + \Delta x_{i-1})}{\Delta x_i \Delta x_{i-1}(\Delta x_i + \Delta x_{i-1})} + \mathcal{O}(\Delta x^{1\cdots 2}) \tag{1.8}$$

bestimmt werden. Die höhere Fehlerordnung bezieht sich wiederum auf gleiche Schrittweiten $\Delta x = \Delta x_i = \Delta x_{i-1} = \ldots$, hiermit ergibt sich die zweite Ableitung zu

$$f^{(2)}(x_i) = \frac{f(x_{i+1}) + f(x_{i-1}) - 2f(x_i)}{\Delta x^2} + \mathcal{O}(\Delta x^2)\,. \tag{1.9}$$

Die Erweiterung zu einem Differenzenquotienten vierter Ordnung geschieht durch Umformung und Linearkombination von Taylorentwicklungen vom Startpunkt x_i zu den vier Punkten x_{i+1}, x_{i-1}, x_{i+2} und x_{i-2}. Im Fall äquidistanter Stützstellen resultiert dieses Vorgehen in der Bestimmungsgleichung

$$f^{(2)}(x_i) = \frac{16(f(x_{i+1}) + f(x_{i-1})) - (f(x_{i+2}) + f(x_{i-2})) - 30f(x_i)}{12\Delta x^2} + \mathcal{O}(\Delta x^4)\,. \tag{1.10}$$

Eindimensionale Wellengleichung

Als möglichst einfaches Beispiel für die Betrachtung der Differenzenverfahren wollen wir die TE-Moden eines Parallelplattenleiters (PPL) berechnen. Der PPL soll in x- und y-Richtung unendlich ausgedehnt sein und die zeitlich harmonisch angeregte Welle breite sich in x-Richtung aus. Bei diesen Annahmen hat das elektrische Feld keine x-Komponente und alle Ableitungen in y-Richtung sind null. Aus den ersten beiden Maxwellschen Gleichungen ($\underline{\vec{J}} = 0$)

$$\operatorname{rot}\underline{\vec{E}} = -j\omega\mu_0\underline{\vec{H}} \qquad\qquad \operatorname{rot}\underline{\vec{H}} = j\omega\varepsilon_0\underline{\vec{E}} \tag{1.11}$$

folgt daher komponentenweise, dass

$$\frac{\mathrm{d}\underline{E}_y}{\mathrm{d}x} = -j\omega\mu_0\underline{H}_z \qquad\qquad -\frac{\mathrm{d}\underline{H}_z}{\mathrm{d}x} = j\omega\varepsilon_0\underline{E}_y. \tag{1.12}$$

Durch Ableiten und ineinander Einsetzen lässt sich die eindimensionale Wellengleichung für die y-Komponente des elektrischen Feldes herleiten zu

$$\frac{\mathrm{d}^2\,\underline{E}_y}{\mathrm{d}\,x^2} = -\omega^2\varepsilon_0\mu_0\underline{E}_y = -k_x^2\underline{E}_y, \tag{1.13}$$

mit der Wellenzahl $k_x = 2\pi/\lambda = \omega/c$. Nach erfolgter Diskretisierung mit einem Rechengitter der Schrittweite Δx können die Werte der elektrischen Feldstärke aller Stützstellen zu einem Vektor $\mathbf{e}$ zusammengefasst werden. Fasst man die numerische Approximation der zweiten Ableitung nach Gl. (1.9) bzw. Gl. (1.10) jedes Punktes in einer Matrix $\widetilde{\mathbf{C}}\mathbf{C}$ zusammen, so kann das Gleichungssystem

$$\widetilde{\mathbf{C}}\mathbf{C}\mathbf{e} = -\Delta x^2 k_x^2\mathbf{e} \tag{1.14}$$

aufgestellt werden. Dieses Eigenwertproblem (bzw. die Matrix $\widetilde{\mathbf{C}}\mathbf{C}$) besitzt die Eigenwerte $-\Delta x^2 k_{x,i}^2$ und die dazugehörigen Eigenvektoren $\mathbf{e}_i$, die das Feldbild des jeweiligen Modes beinhalten. Die sich ergebenden Moden sind – aufgrund der Symmetrie der Matrix – zueinander orthogonal, d.h. es gilt $\mathbf{e}_i^\top \cdot \mathbf{e}_j = \mathbf{0}$ für $i \neq j$. Wird beispielsweise die Ableitung wie in Gl. (1.9) benutzt, so stellt sich eine Zeile des Gleichungssystems nach Durchmultiplizieren mit $-\Delta x^2$ wie folgt dar:

$$\begin{pmatrix} \cdots & 0 & -1 & 2 & -1 & 0 & \cdots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ f(x_{i-2}) \\ f(x_{i-1}) \\ f(x_i) \\ f(x_{i+1}) \\ f(x_{i+2}) \\ \vdots \end{pmatrix} = \Delta x^2 k_x^2 f(x_i)\,. \tag{1.15}$$

Da die numerischen Operatoren (1.9) und (1.10) Stützstellen außerhalb des Rechengebietes benötigen (siehe Abb. 1.3), müssen am Rand des Rechengebietes Sonderbehandlungen eingeführt werden. Prinzipiell unterscheidet man zwischen zwei sehr einfach zu behandelnden Randbedingungen:

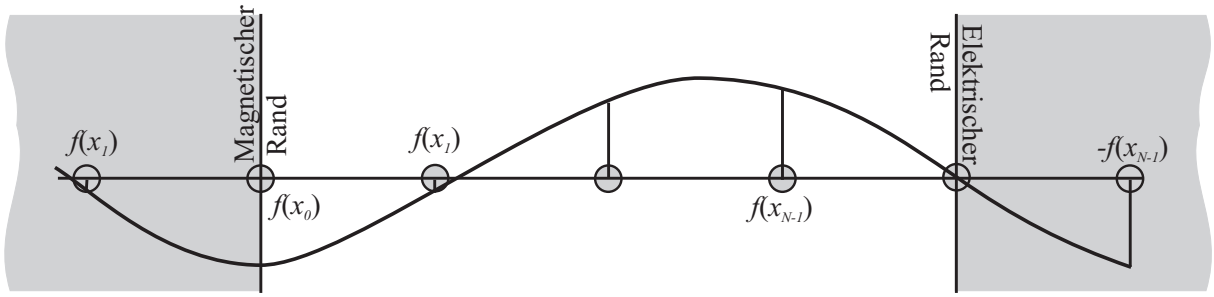

Abbildung 1.3: Randbedingungen. Elektrische Randbedingungen erzwingen eine ungerade Symmetrie ($f(x_{N-1}) = -f(x_{N+1})$) der tangentialen elektrischen Feldstärken, magnetische Randbedingungen eine gerade Symmetrie ($f(x_{-1}) = f(x_1)$). Hierdurch können Vorschriften zur Bestimmung nichtexistierender Feldwerte, die zur Auswertung über den Rand hinausgreifender Differenzenquotienten heranzuziehen sind, formuliert werden.

- **Elektrischer Rand:** ungerade Symmetrie des tangentialen elektrischen Feldes am Rand
- **Magnetischer Rand:** gerade Symmetrie des elektrischen Feldes am Rand

Für das hier gewählte Beispiel der Bestimmung von TE-Moden wäre also die physikalisch korrekte Randbedingung, bei perfekt elektrischen Rändern, die elektrische Randbedingung, da hierbei die $\underline{E}_y$-Komponente am Rand verschwinden muss. Dies entspräche dem rechten Rand in Bild 1.3.

Aufgrund der ungeraden Symmetrie der tangentialen Feldstärke ($f(x_{N-1}) = -f(x_{N+1})$), ergibt sich für die Differenzenvorschrift (1.9) und damit für die Modifikation der Matrixeinträge am rechten Rand:

$$2f(x_N) - f(x_{N-1}) - f(x_{N+1}) = 2f(x_N) = \Delta x^2 k_x^2 f(x_N)\,. \tag{1.16}$$

Die physikalische Bedeutung der Moden und Eigenwerte wird im folgenden kurz skizziert:
Ein Mode ist die charakteristische, räumliche Feldverteilung in einer Struktur mit vollständiger Berandung. Die Moden können mithilfe der Eigenwertgleichung (1.14) berechnet werden und entsprechen den Eigenwerten. Zu jedem Eigenwert (Mode) gibt es einen Eigenvektor (Modenbild), der die numerischen Werte des Feldes (z.B. y-Komponente der elektrischen Feldstärke) beinhaltet. Der niedrigste Mode beschreibt die Eigenschwingung mit der größten Wellenlänge bzw. kleinsten Frequenz. Bei elektrischen Rändern wäre dies im gegebenen Beispiel durch einen „halben Sinusbogen“ in der Struktur gekennzeichnet.

1.2.2 Darstellung von Körpern mit Oberflächengittern

Zur Simulation einer realistischen Anordnung mit Hilfe eines numerischen Verfahrens muss zuerst die Geometrie der Anordnung in das Programm eingegeben werden[1]. Die einfachste Möglichkeit dafür ist die Geometrie des Problems als eine Kombination von analytisch beschreibbaren Körpern (z.B. Kugeln, Zylindern, Ellipsoide, Quader) zu modellieren. Der Nachteil dieser Methode besteht darin, dass sich nur Körper mit beschränkter Komplexität als Kombination kanonischer geometrischer Körper beschreiben lassen. Eine andere Möglichkeit ist, die Oberfläche der Körper des Problems mit Hilfe eines Dreiecksgitters zu diskretisieren und als Menge dieser Dreiecke darzustellen. Zusammen mit den Normalvektoren der Oberflächen lassen sich allgemeine Geometrien beschreiben. Das diskrete Modell kann mit Hilfe eines allgemeinen CAD-Programms hergestellt oder direkt vom Output eines 3D-Scanners eingelesen

[1] Dies hat in erster Hinsicht nichts mit dem numerischen Gitter zu tun. Diese Diskretisierung erfolgt später.

werden. Die Information, die durch die Diskretisierung verloren geht, kann mit Hilfe mathematischer Algorithmen teilweise wiederhergestellt werden, sofern das für den Solver notwendig ist. Die meist verwendete Methode dafür ist die Interpolation mit *Splines*. Es ist selbstverständlich, dass je feiner das Dreiecksgitter ist, desto genauer lässt sich die Geometrie approximieren und desto größer ist gleichzeitig der Rechenaufwand. Man muss deshalb ein Optimum zwischen der Genauigkeit der Approximation und dem Rechenaufwand finden.

Das .stl (von engl. *stereolithography*, oder auch *Standard Transformation Language*) Format ist ein universelles Format zur Darstellung von Körpern in CAD-Anwendungen. Es kann entweder eine Binär- oder ASCII-Datei sein. Die ASCII-.stl-Datei beginnt immer mit dem Schlüsselwort *solid* und endet mit

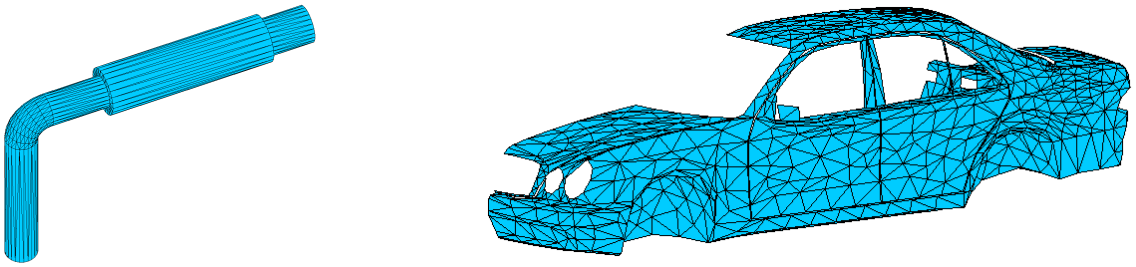

Abbildung 1.4: Durch eine Triangulierung der Oberfläche lassen sich Körper beliebiger Geometrie genau beschreiben. Hier wird ein Koaxial-Kabel (links) und ein Auto (rechts) mit Hilfe eines Oberflächengitters dargestellt.

dem Wort *endsolid*. Zwischen diesen Schlüsselwörtern wird die Liste der Dreiecke, die die Oberfläche des Körpers beschreiben, eingefügt. Zur Beschreibung jedes verschiedenen Dreiecks wird zuerst der normale Einheitsvektor definiert und danach folgen die xyz-Koordinaten jedes Punktes des Dreiecks. Wenn die Normalenvektoren nicht eingegeben worden sind, werden Sie von der Software durch die Rechte-Hand-Regel während des Einlesens der Datei berechnet. Falls die Information für die Normalenvektoren nicht verfügbar ist, müssen die drei Werte zu Null gesetzt werden. Die allgemeine Struktur einer .stl-Datei lautet:

```
1   solid
2       ...
3       facet normal 0.00 0.00 1.00
4          outer loop                    % x-, y-, z- Koordinate des
5            vertex  2.00  2.00  0.00   % ersten Punktes des Dreiecks
6            vertex -1.00  1.00  0.00   % zweiten Punktes des Dreiecks
7            vertex  0.00 -1.00  0.00   % dritten Punktes des Dreiecks
8          endloop
9       endfacet
10      ...
11  endsolid
```

Nach STL-Format-Konvention müssen alle benachbarten Dreiecke zwei gemeinsame Punkte haben (engl. *vertex-to-vertex rule*). Diese Regel wird in Abb. 1.5 erklärt.

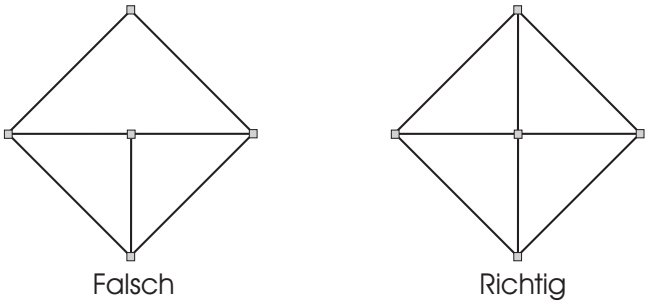

Abbildung 1.5: Vertex-to-vertex rule. Bei der *vertex-to-vertex rule* wird ein Viereck in Dreiecke zerlegt. In der Triangulierung auf der linken Seite liegt ein Punkt in der Mitte einer Kante des oberen Dreiecks und verletzt somit die STL-Format-Konvention.

1.3 Versuchsdurchführung

Der Versuch ist in zwei völlig unabhängige Abschnitte geteilt. Im ersten Abschnitt soll die eindimensionale Wellengleichung mithilfe des **Differenzenverfahren** für das vorgestellte Beispiel des Parallelplattenleiters gelöst werden. Hierbei sind die Abbildungen 1.1-1.3 als Anschauung nützlich. Die Ausdehnung in x-Richtung soll $L = 1\,\mathrm{m}$ betragen, wobei die Stützstellenanzahl n je nach Versuchsteil variiert. Die Materialwerte ε und μ werden als homogen angenommen. Kernpunkt bildet die Betrachtung des Konvergenzverhaltens für verschiedene Verfahrensordnungen sowie für fehlende Randbedingungen. Der zweite Abschnitt **Dreidimensionale Darstellung** besteht hauptsächlich aus der Visualisierung verschiedener Geometrien sowie der Bestimmung des Fehlers der diskreten Modellierung.

1.3.1 Vorbereitungsaufgaben

Differenzenverfahren

1. Zeigen Sie, dass der zentrale Differenzenquotient (1.5) aus der Subtraktion zweier Taylor-Entwicklungen zu den Punkten x_{i+1} und x_{i-1} folgt. Vergessen Sie dabei nicht, die Fehlerterme zu berücksichtigen und kommentieren Sie die Ordnung des Fehlers im Vergleich zu vorwärts- und rückwärts-Differenzenquotient.

2. Berechnen Sie ausgehend von einem Startpunkt $f(\tilde{x}_i)$ auf einem dualen Gitter mit Hilfe von Taylorentwicklungen analog zur Differenzenvorschrift (1.6) im Fall äquidistanter Gitter eine zentrale Differenzenvorschrift vierter Ordnung zur Berechnung der ersten Ableitung. Verwenden Sie hierfür $f(x_i), f(x_{i+1}), f(x_{i+2})$ sowie $f(x_{i-1})$.

3. Bestimmen Sie aus der Differenzenvorschrift (1.10) die Matrix $\tilde{\mathbf{CC}}$ für einen 1D-Resonator mit 5 Stützstellen für den Fall elektrischer bzw. magnetischer Randbedingungen an beiden Rändern (siehe Abb. 1.3, Gl. (1.15) und Gl. (1.16)).

4. Betrachtet werden soll exemplarisch die Berechnung der diskreten Wellenzahlen bei vorgegebener Länge des Rechengebietes nach Gl. (1.14). Geben Sie den Abbruchfehler in Abhängigkeit von der Anzahl der verwendeten Stützstellen n bzw. der Diskretisierungsschrittweite Δx an, wenn der Differenzenquotient nach Gl. (1.9) bzw. Gl. (1.10) verwendet wird. Wie müssen Sie ein entsprechendes Diagramm (Abbruchfehler vs. Anzahl der Stützstellen) skalieren, um einen geradlinigen Verlauf zu erhalten?

5. Stellen Sie eine Formel auf, mit der die analytischen Wellenzahlen $k_{x,\mathrm{ana}}$ für die eindimensionale Wellengleichung und einem Resonator der Länge L einmal mit rein elektrischer Berandung und

einmal mit unterschiedlichen Randbedingungen (eine Seite elektrisch – eine Seite magnetisch) berechnet werden kann. Bestimmen Sie dabei auch die jeweils kleinste Wellenzahl. Geben Sie bei einer numerischen Berechnung eine Formel für den relativen Wellenzahlfehler Δk_x an.

6. Wie kann man die Orthogonalität zweier Eigenvektoren testen und was sagen orthogonale Eigenvektoren über die Lösungen eines Eigenwertproblems (Moden) aus?

Dreidimensionale Darstellung

Gegeben sei in Abbildung 1.6 die Diskretisierung eines Kreiszylinders der Höhe h und des Radius r mit n_D Deckflächendreiecken der Grundlänge $2a_\mathrm{D}$ und Höhe h_D.

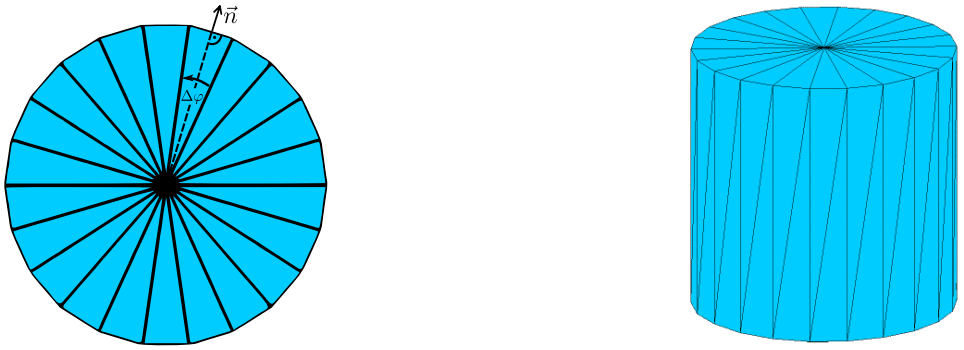

Abbildung 1.6: Diskretisierung einer Kreiszylinderoberfläche. Dargestellt sind die Dreiecksgitter der Diskretisierung der Deckelfläche (links) und des diskretisierten Zylinders in Perspektivansicht (rechts).

7. Überlegen Sie sich einen Pseudocode mit For-Schleifen, mit welchem Sie die Koordinaten aller Dreiecke der Oberflächendiskretisierung (siehe Abb. 1.6) bestimmen können. Die Anzahl der Dreiecke n_D soll dabei beliebig sein. Beachten Sie, dass zur Oberfläche Deck-, Mantel- und Bodenfläche gehören. Außerdem besitzen die Dreiecke auf der Mantelfläche „unterschiedliche Orientierungen“.

8. In der Theorie wurde erwähnt, dass eine feinere Diskretisierung die Genauigkeit der Darstellung steigert[2], aber dass auch der Rechenaufwand und der geforderte Speicher zunehmen. Berechnen Sie den Speicherplatz, der zur Speicherung des oben abgebildeten, diskretisierten Zylinders im STL-Format notwendig ist, als Funktion der Anzahl der Dreiecke n_D (Normalenvektoren müssen auch berücksichtigt werden). Beachten Sie hierbei lediglich die notwendige Anzahl der `double`-Zahlen, wobei eine `double`-Zahl 8 Byte benötigt.

9. Gegeben ist folgender Zusammenhang für die Fläche eines Deckflächendreieckes:

$$A_\mathrm{D} = 2\left[\frac{1}{2}a_\mathrm{D}h_\mathrm{D}\right] = 2\left[\frac{1}{2}\cdot r\cos\left(\frac{\Delta\varphi}{2}\right)\cdot r\sin\left(\frac{\Delta\varphi}{2}\right)\right] = \frac{1}{2}r^2\sin(\Delta\varphi) \tag{1.17}$$

Überlegen sie sich geometrisch, wie diese Formel zustande kommt und dokumentieren Sie die einzelnen Schritte. Leiten Sie anschließend eine Formel zur Berechnung des Volumens und der Fläche des gezeigten Zylinders in Abhängigkeit von der Anzahl der Deckflächendreiecke n_D her. Bestimmen Sie den relativen Oberflächenfehler ΔA bzw. Volumenfehler ΔV dieser Diskretisierung. Bei welchen Körpern wäre eine solche Oberflächendiskretisierung mit Dreiecken exakt?

[2] Es gibt jedoch Fälle, bei denen eine Verdichtung des Oberflächengitters nicht gegen die exakte Lösung konvergiert.

10. Bei einigen Anwendungen ist es wichtig, dass die diskretisierte Fläche möglichst glatt bleibt. Ein wichtiges Beispiel hierbei ist die Streuung hochfrequenter elektromagnetischer Felder, wobei Kanten und Ecken das gestreute Feld verändern können. Angenommen bei der oberen Diskretisierung des Zylinders ist der Winkel zwischen den Normalenvektoren benachbarter Dreiecke der Deckflächen $\Delta\varphi \leq 5°$ gefordert. Berechnen Sie die minimale Anzahl an Dreiecken zur Erfüllung dieser Forderung.

1.3.2 Aufgaben während der Praktikumssitzung

Differenzenverfahren

1. Implementieren Sie eine Methode

   ```
   [cc] = createCC(n, ord, bc)
   ```
   (1.18)

   welche die $\tilde{\mathbf{C}}\mathbf{C}$-Matrix mit der Stützstellenanzahl `n`, der Ordnung des Differenzenverfahrens `ord` (`2` = zweite und `4` = vierte Ordnung) und der für beide Ränder identischen Art der Randbedingung `bc` (`0`=fehlende, `1`=elektrische und `2`=magnetische) erstellt. Rückgabewert ist die $\tilde{\mathbf{C}}\mathbf{C}$-Matrix `cc`. Nutzen Sie hierfür das vorgefertigte Template `createCC.m`.

2. Ein einfacher Solver soll in

   ```
   [kx, modes] = solveCC(cc, dx)
   ```
   (1.19)

   implementiert werden, wobei die Schrittweite `dx` als zusätzlicher Eingabeparameter übergeben wird. `kx` ist hier ein Vektor mit den geordneten Wellenzahlen, angefangen mit der kleinsten Wellenzahl. In gleicher Reihenfolge sollen auch die Eigenvektoren in der Matrix `modes` zurückgegeben werden. Das Eigenwertproblem lässt sich durch die MATLAB®-Funktion `eig` lösen, das Sortieren kann mit `sort` erfolgen.

3. Verwenden Sie die Routine `createCC` mit `n=6`, `ord=2` und `bc=0` und anschließend `solveCC`. Überprüfen Sie die Orthogonalität der Eigenvektoren. Wie viele Eigenmoden können bei dieser Parameterwahl bestimmt werden? Nutzen Sie hierfür die bereits gegebene Datei `check_orth.m`.

   Hinweis: Wenn Sie die Eigenvektoren geschickt miteinander multiplizieren, erhalten Sie eine Matrix, in der jeder Eintrag einem Produkt zweier Eigenvektoren entspricht. Diese sich ergebene Matrix lässt sich dann bequem mit dem Befehl `imagesc` (siehe Abschnitt 1.3.3) darstellen.

4. Stellen Sie die zwei niedrigsten Moden in einem Skript `plotModes` grafisch dar. Verwenden Sie `n=100`, `bc=1` und `ord=4` sowie die Länge des eindimensionalen Gebietes L=5 m.

5. Als nächstes sollen Sie das Konvergenzverhalten betrachten. Schreiben Sie ein Skript `plotConv`, welches das Konvergenzverhalten in Abhängigkeit von der Stützstellenanzahl `n` Ihrer verschiedenen Implementierungen in zwei Grafiken dokumentiert:

   a) Lineare Darstellung der Wellenzahl des Grundmodes über der Stützstellenanzahl n im Fall elektrischer Randbedingungen, sowohl analytisch als auch zweite und vierte Ordnung.

   b) Doppelt-logarithmische Darstellung des relativen Wellenzahlfehlers des Grundmodes über die Gitterschrittweite bei elektrischen und fehlenden Randbedingungen für jeweils beide Ordnungen.

   Verwenden Sie die Grafiken um die Ordnung der verschiedenen Implementierungen graphisch zu bestimmen. Vergleichen Sie Ihr Ergebnis mit Aufgabe 4 aus der Vorbereitung. Wie verändert sich das Konvergenzverhalten, wenn keine Randbedingungen implementiert sind?

Dreidimensionale Darstellung

6. Schreiben Sie ein Skript `plotCyl`, welches den Zylinder aus der Vorbereitung visualisiert. Bauen Sie hierzu auf der MATLAB®-Funktion `patch` auf. Verwenden Sie die Anzahl der Dreiecksflächen in einer Deckelfläche `nd=20`, den Radius `r=1` und die Höhe `h=1`.

7. Stellen Sie für einen Zylinder Ihrer Wahl den Oberflächenfehler ΔA sowie den Volumenfehler ΔV in Abhängigkeit von der Anzahl der Dreiecksflächen je Deckel n_D (Vorbereitungsaufgabe 9) in einem Skript `plotVisErr` doppelt-logarithmisch dar. Mit welcher Ordnung konvergieren die Fehler? Aus der Vorbereitung wissen Sie zusätzlich, wie der Speicherbedarf der Darstellung skaliert.
   Wie viele Dreiecke sind notwendig um einen Diskretisierungsfehler kleiner als 10^{-5} zu garantieren?

8. Verwenden Sie die bereitgestellte Methode `read_stl` um zwei der bereitgestellten Geometrien im STL-Format (vgl. Abb. 1.4) einzulesen und dann erneut mit `patch` darzustellen. Nennen Sie Ihr Skript `plotStl`.

1.3.3 Nützliche MATLAB®-/GNU OCTAVE Befehle und Hilfsroutinen

Allgemeine Befehle	
`clear all`	Löscht alle Variablen.
`help` *Befehl*	Gibt die Hilfe zu einem Befehl zurück.
`for i=1:n;` *Befehle...*; `end`	For-Next Schleifen-Konstruktion.
`if i>j;` *Befehle...*; `end`	If-Then Konstruktion.
Befehle für Vektoren	
`vec=n:m`	Dem Vektor `vec` wird die Zahlenfolge $n, n+1, \ldots, m$ zugewiesen.
`vec=n:p:m`	Dem Vektor `vec` wird die Zahlenfolge $n, n+p, \ldots, m$ zugewiesen.
`vec3=[vec1, vec2]`	Der Vektor `vec` wird aus der Aneinanderreihung der Vektoren $vec1$ und $vec2$ gebildet.
`vec(i)`	Referenziert das i-te Element des Vektors `vec`.
`vec2=vec1(idx)`	Referenziert mit Hilfe eines Indexvektors `idx` mehrere Elemente aus Vektor `vec1` und gibt diese als Vektor `vec2` zurück.
`vec'`	Gibt den transponierten Vektor des Vektors `vec`.
`abs(vec)`	Liefert einen Vektor, der die Absolutbeträge der Elemente von `vec` enthält.
`min(vec)`	Liefert den minimalen Wert des Vektors `vec`.
`[vec2,idx]=sort(vec1)`	Sortiert den Vektor `vec1` beginnend mit dem kleinsten Wert zu `vec2`. Die Reihenfolge der Indizes wird in `idx` zurückgegeben, so dass `vec2=vec1(idx)`.
Befehle für Matrizen	
`mat=[vec1; vec2]`	Zwei Vektoren `vec1` und `vec2` der Länge `n` werden übereinander gelegt, so dass eine $2 \times n$ Matrix `mat` entsteht.
`mat=zeros(m,n), mat=ones(m,n)`	Erzeugt eine $m \times n$ Matrix `mat`, gefüllt mit Nullen bzw. Einsen.
`S = sparse(m,n)`	Erzeugt eine Nullmatrix `S` der Dimension `(m,n)`. Dieser Befehl ist bei den in diesem Praktikum verwendeten Matrizen immer dem Befehl `zeros(m,n)` vorzuziehen, da er sehr viel weniger Ressourcen beansprucht.
`mat=sparse(i,j,s,m,n)`	Erzeugt eine $m \times n$ dünnbesetzte Matrix `mat`, deren Einträge durch die Vektoren `i,j,s` der Länge L bestimmt sind. Es gilt: `mat(i(k),j(k))=s(k)` für $k = 1:L$.
`mat=spdiags(B,d,m,n)`	Erzeugt eine $m \times n$ dünnbesetzte Matrix `mat` indem die Spalten von `B` auf die durch `d` spezifizierten Diagonalen geschrieben werden.
`mat(i,j)`	Referenziert das i,j-te Element der Matrix `mat`.
`mat(:,i), mat(i,:)`	Referenziert den i-ten Spalten- bzw. Zeilenvektor der Matrix `mat`.
`mat(i:j,k:l)`	Referenziert einen $(i-j) \times (l-k)$ großen Blockausschnitt aus der Matrix `mat`.
`diag(mat)`	Liefert die Diagonale der Matrix `vec` als Vektor.

`[V,D]=eig(A)`	Berechnet Eigenvektoren und Eigenwerte der Matrix `A` und weist die Eigenvektoren spaltenweise der Matrix `V` sowie die Eigenwerte der Diagonalen der Matrix `D` zu.
`[V,D]=eigs(A)`	Berechnet Eigenvektoren und Eigenwerte der Matrix `A` und weist die Eigenvektoren spaltenweise der Matrix `V` sowie die Eigenwerte der Diagonalen der Matrix `D` zu. Dieser Befehl ist speziell für große, dünn besetzte und quadratische Matrizen konzipiert.
Befehle zum Zeichnen von Grafiken	
`plot(x,y)`	Trägt die Werte des Vektors y gegen x auf.
`loglog(X1,Y1,X2,Y2)`	Zeichnet `Y1` gegen `X1` sowie `Y2` gegen `X2` doppeltlogarithmisch auf.
`patch(X,Y,Z,C)`	Zeichnet eine triangulierte Fläche. X,Y,Z sind Matrizen, deren Spalten die Koordinaten jedes Dreiecks enthalten. Zwei aneinanderhängende Dreiecke können z.B. so modelliert werden: X=[[0,0];[0,1];[0,0]]; Y=[[0,0];[1,0];[0,1]]; Z=[[0,0];[0,0];[1,0]] .
`grid on`	Schaltet in einem Graph die automatische Hilfslinien-Darstellung ein.
`read_stl(filename)`	Liest die filename.stl-Datei und gibt die X,Y,Z-Koordinaten-Matrizen (siehe `patch(X,Y,Z,C)`), die Normalenvektoren und den Namen des Körpers zurück.
`imagesc(matrix)`	Visualisiert eine Matrix in einer farblichen Darstellung. Die Farben sind so ausgewählt, dass die Einträge der Matrix das komplette Farbspektrum der aktuellen Farbgebung abdecken.