Ausarbeitung Übung 3

Studienarbeit von Dominik Schiller, Constanze Kramer, Simon Arnold & Tobias Lingenberg
Datum: 2. Dezember 2020

Darmstadt

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Ausarbeitung Übung 3

Studienarbeit von Dominik Schiller, Constanze Kramer, Simon Arnold & Tobias Lingenberg

Datum: 2. Dezember 2020

Darmstadt

Inhaltsverzeichnis

1 Einleitung

Dies ist die Einleitung.

2 Bearbeitung der Aufgaben

2.1 Koaxialkabel Simulator

Das elektrische Verhalten eines Koaxialkabels lässt sich durch das Ersatzschaltbild 2.1 beschreiben. Hierbei stellt R_1 den ohmschen Längswiderstand dar, der von der Leitungslänge, dem Leitungsquerschnitt und dem Material abhängt.

Das induktive Verhalten des Kabels wird durch L_1 simuliert. Jeder stromdurchflossene Leiter baut ein Magnetfeld auf, die Änderung des Magnetfelds induziert eine Spannung, die dem Stromfluss entgegen wirkt und diesen abschwächt.

Durch den Isolationswert R_2 werden Verlustströme betrachtet, die zwischen dem inneren und äußeren Leiter des Koaxialkabels auftreten. Diese Leckströme entstehen, da es in der Realität keinen idealen Isolator gibt.

Zuletzt beinhaltet das Ersatzschaltbild noch die Leitungskapazität C_1. Ist am Ende der Leitung ein Verbraucher R_f angeschlossen, so liegt an diesem eine Spannung an. In Folge dessen besteht auch zwischen Hin- und Rückleiter eine Potenzialgefälle. Beide Leiter verhalten sich, aufgrund der Ladungsdifferenz, somit wie die Platten eines Kondensators[1].

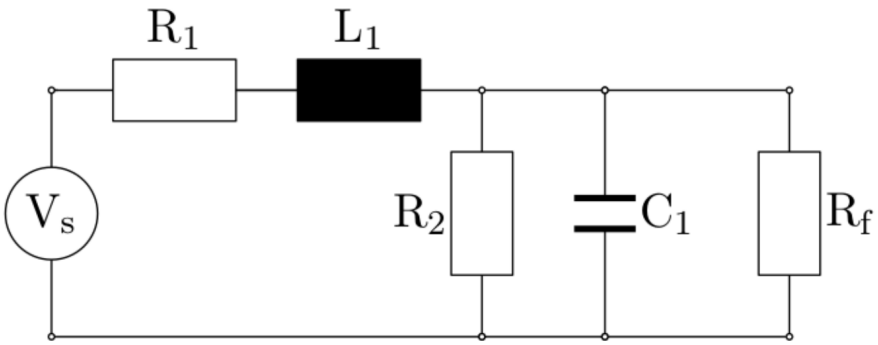


Abbildung 2.1: Ersatzschaltbild Koaxialkabel

Mit den Routinen, die in Aufgabe 2.3 entwickelt wurden lässt sich nun dieses Segment eines Koaxialkabels simulieren. Hierzu übergibt man der Methode `calculate_matrices` (siehe 4.2) die Netzliste 2.1.

[1] Quelle: Leitungstechnik, Dipl.-Ing.(FH) Christian Wolff, 2009

Listing 2.1: Netzliste Koaxialkabel

```
Vs  1  0  12
L1  2  3  0.00025
R1  1  2  0.001
C1  3  0  0.0000001
R2  0  3  1500
Rf  0  3  1500
```

Berechnet werden nun wieder die Matrizen $\mathbf{M}, \mathbf{K}$ und $\mathbf{r}$, die die Differenzialgleichung

$$\mathbf{M}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x} + \mathbf{K}\mathbf{x} = \mathbf{r} \tag{1}$$

parametrisieren. Zuletzt wird mit der Funktion `daspk` das Gleichungssystem numerisch gelöst und die Spannung an R_f gezeichnet 2.2. Das dazu verwendete Skript befindet sich im Anhang 4.3.

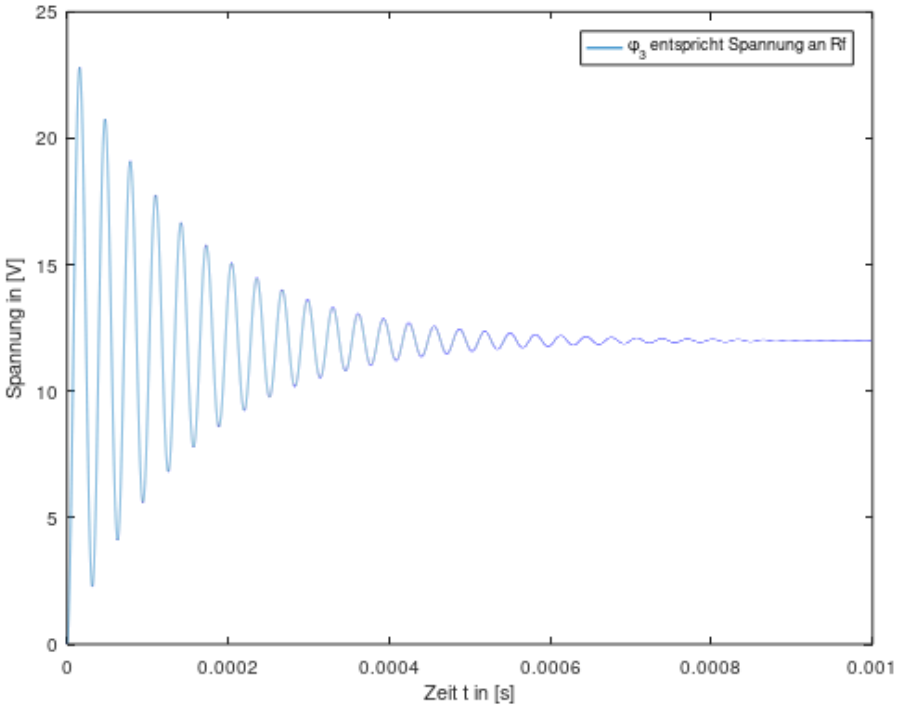


Abbildung 2.2: Spannungsverlauf an Last R_f bei konstanter Spannungsquelle mit 12V

Koaxialkabel mit n Gliedern

Nun wird das RLC-Segment in 2.1 $n = 10$ mal hintereinander in Reihe geschaltet (siehe 2.3).

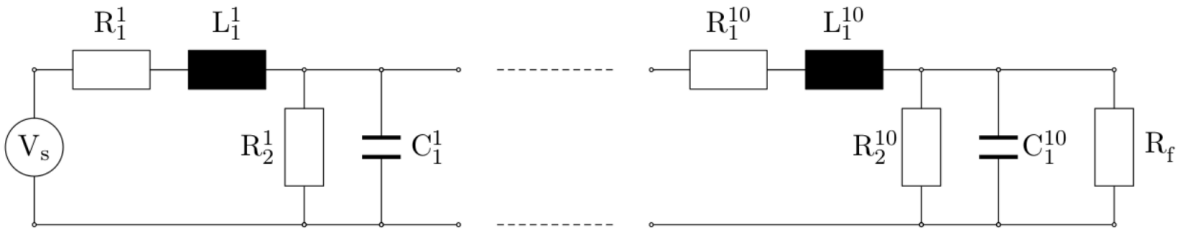


Abbildung 2.3: Ersatzschaltbild Koaxialkabel mit $n = 10$ Segmenten

Um zu untersuchen inwiefern diese Veränderung auf die Spannung an R_f Einfluss hat, wurde ein neues Skript geschrieben, das die Unterschiede der Ausgangsspannung bei $n = 1$ und $n = 10$ sichtbar macht (siehe 4.5). Hierzu wurde die Netzliste für $n = 10$ Segmente erstellt (siehe 4.4), den Methoden übergeben und die Spannung an R_f erneut berechnet. Beide Ausgangsspannungen für $n = 1$ und $n = 10$ wurden nun in das selbe Diagramm 2.4 eingezeichnet:

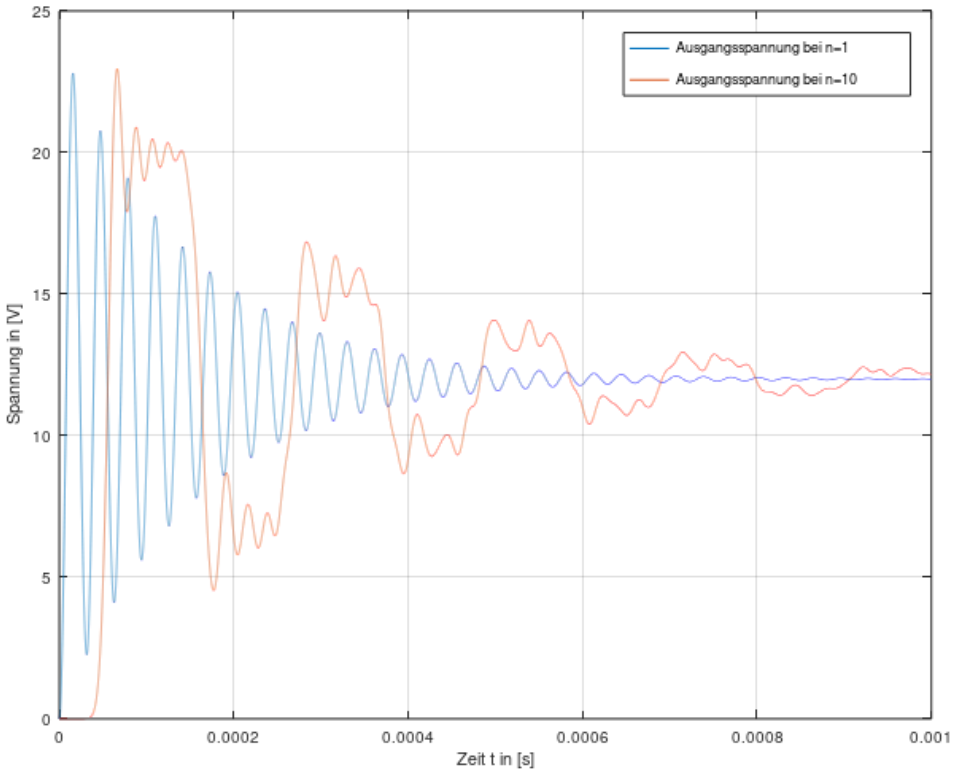


Abbildung 2.4: Spannungsverlauf an Last R_f für $n = 1$ und $n = 10$

Auffällig ist zunächst einmal, dass im Fall $n = 10$ das Signal durch Oberschwingungen gestört wird. Jedoch lässt sich die eigentliche Hauptschwingung dennoch gut erkennen. Bei Betrachtung dieser im Vergleich zum Fall $n = 1$ wird ebenso deutlich, dass die Frequenz erheblich kleiner wurde. Ein weiterer Unterschied besteht in der Dämpfung, welche bei $n = 10$ nicht so stark ist . Es dauert länger, bis die Amplitude gegen 0 geht. Zuletzt ist noch anzumerken, dass der maximal Betrag der Spannung an R_f in beiden Fällen gleich ist. Die

Spannung liegt immer zwischen 0 und $24\mathrm{V}$.

Empirische Tests mit anderen Werten von n ergaben, dass diese Aussagen für alle $n > 1, n \in \mathbb{N}$ gelten. Folglich gilt, je mehr Segmente hinzugefügt werden, umso mehr verkleinert sich die Frequenz. Der maximale Betrag der Spannung an der Last hängt jedoch nicht von n ab. Der erste Ausschlag erreicht jedes mal das maximale Niveau von knapp unter $24\mathrm{V}$ und nimmt dann mit der Zeit immer weiter ab, bis die Spannung gegen $12\mathrm{V}$ konvergiert.

2.2 Frequenzbereich

In der Elektrotechnik arbeitet man oft an Problemen mit sinus-förmigen Schwingungen. Um die Rechnungen dafür zu vereinfachen bietet es sich an diese Probleme nur im Frequenzbereich zu betrachten. Im folgenden wird allerdings zuerst eine Lösung für die skalare lineare gewöhnliche Differentialgleichung im Zeitbereich

$$x'(t) = f(t,x) := -kx(t) + r(t) \tag{2}$$

mit Startwert $x(t_0) = x_0$ und $k > 0$ ermittelt. Die Anregung ist durch $r(t) = \cos(\omega t)$ als sinus-förmige Schwingung gegeben, wobei ω die Kreisfrequenz ist. Aus der vorgegebenen allgemeinen Lösung

$$x(t) = \mathrm{e}^{-k(t-t_0)} \left(x_0 + \int_{t_0}^{t} \mathrm{e}^{k(s-t_0)} r(s) ds \right) \tag{3}$$

betrachtet man vorerst nur das Integral und löst dieses.
Für die partielle Integrationsregel

$$\int_a^b f'(x) \cdot g(x) dx = [f(x) \cdot g(x)]_a^b - \int_a^b f(x) \cdot g'(x) dx$$

setzt man nun $f'(x) = \mathrm{e}^{k(s-t_0)}$ und $g(x) = \cos(\omega s)$. Daraus ergibt sich $f(x) = \frac{1}{k}\mathrm{e}^{k(s-t_0)}$ sowie $g'(x) = -\omega \sin(\omega s)$. Eingesetzt ergibt dies

$$\int_{t_0}^{t} \mathrm{e}^{k(s-t_0)} \cos(\omega s) ds = \left[\frac{\mathrm{e}^{k(s-t_0)} \cos(\omega s)}{k} \right]_{s=t_0}^{t} - \int_{t_0}^{t} -\frac{\omega \sin(\omega s) \mathrm{e}^{k(s-t_0)}}{k} ds.$$

Unter der Annahme $f'(x) = \frac{\mathrm{e}^{k(s-t_0)}}{k}$, $g(x) = -\omega \sin(\omega s)$ wird erneut partiell integriert, damit folgt $f(x) = \frac{\mathrm{e}^{k(s-t_0)}}{k^2}$ und $g'(x) = -\omega^2 \sin(\omega s)$. Setzt man die Annahmen in die vorherige Gleichung ein, folgt

$$\int_{t_0}^{t} \mathrm{e}^{k(s-t_0)} \cos(\omega s) ds = \left[\frac{\mathrm{e}^{k(s-t_0)} \cos(\omega s)}{k} \right]_{s=t_0}^{t} - \left(- \left[\frac{\omega \mathrm{e}^{k(s-t_0)} \sin(\omega s)}{k^2} \right]_{s=t_0}^{t} - \int_{t_0}^{t} -\frac{\omega^2 \mathrm{e}^{k(s-t_0)} \cos(\omega s)}{k^2} ds \right)$$

dies weiter vereinfacht führt zu

$$\int_{t_0}^{t} \mathrm{e}^{k(s-t_0)} \cos(\omega s) ds = \left[\frac{\mathrm{e}^{k(s-t_0)} \cos(\omega s)}{k} \right]_{s=t_0}^{t} + \left[\frac{\omega \mathrm{e}^{k(s-t_0)} \sin(\omega s)}{k^2} \right]_{s=t_0}^{t} - \frac{\omega^2}{k^2} \int_{s=t_0}^{t} \mathrm{e}^{k(s-t_0)} \cos(\omega s) ds.$$

Das Ausgangsintegral findet sich nun sowohl auf der linken, als auch auf der rechten Seite der Gleichung wieder. Die Formel

$$\int_{t_0}^{t} \mathrm{e}^{k(s-t_0)}\cos(\omega s)ds \cdot \left(\frac{k^2+\omega^2}{k^2}\right) = \left[\frac{\mathrm{e}^{k(s-t_0)}\cos(\omega s)}{k}\right]_{s=t_0}^{t} + \left[\frac{\omega \mathrm{e}^{k(s-t_0)}\sin(\omega s)}{k^2}\right]_{s=t_0}^{t}$$

ergibt sich durch das Zusammenfassen der beiden gleichen Integrale auf einer Seite. Dividieren durch $\frac{k^2+\omega^2}{k^2}$ und das einsetzen der Integrationsgrenzen führt zu dem Ergebnis des Integrals

$$\int_{t_0}^{t} \mathrm{e}^{k(s-t_0)}\cos(\omega s)ds = \frac{\omega \mathrm{e}^{kt}\sin(\omega t) + k\mathrm{e}^{kt}\cos(\omega t)}{\mathrm{e}^{kt_0}(\omega^2+k^2)} - \frac{\omega\sin(\omega t_0) + k\cos(\omega t_0)}{\omega^2+k^2}.$$

Somit ist die Lösung für (3)

$$x(t) = \mathrm{e}^{-k(t-t_0)}\left(x_0 + \frac{\omega \mathrm{e}^{kt}\sin(\omega t) + k\mathrm{e}^{kt}\cos(\omega t)}{\mathrm{e}^{kt_0}(\omega^2+k^2)} - \frac{\omega\sin(\omega t_0) + k\cos(\omega t_0)}{\omega^2+k^2}\right),$$

durch das Ausmultiplizieren und Aufteilen in den gedämpften Anteil abhängig vom Startwert x_0 sowie sinus-förmige Schwingung, erfüllt

$$x(t) = \mathrm{e}^{-k(t-t_0)}\left(x_0 - \frac{\omega\sin(\omega t_0) + k\cos(\omega t_0)}{\omega^2+k^2}\right) + \frac{\omega\sin(\omega t) + k\cos(\omega t)}{\omega^2+k^2}. \tag{4}$$

die Differentialgleichung (2) im Zeitbereich.

Nach Berechnung der Differentialgleichung im Zeitbereich folgt nun die Betrachtung im Frequenzbereich. Hierbei wird die Lösung als Kosinus unbekannter Amplitude

$$x_{freq}(t) = \Re\left\{\underline{x}\mathrm{e}^{j\omega t}\right\}$$

angenommen, mit $\cos(\omega t) = \Re\left\{\mathrm{e}^{j\omega t}\right\}$ wobei $\underline{x}$ der Phasor ist. Die zeitliche Ableitung

$$x'_{freq}(t) = \Re\left\{j\omega\underline{x}\mathrm{e}^{j\omega t}\right\}$$

wird in (2) eingesetzt, dies liefert

$$\Re\left\{j\omega\underline{x}\mathrm{e}^{j\omega t}\right\} = -k\Re\left\{\underline{x}\mathrm{e}^{j\omega t}\right\} + \Re\left\{\mathrm{e}^{j\omega t}\right\}.$$

Umstellen und Zusammenfassen der Terme ergeben die Gleichung

$$\Re\left\{(j\omega\underline{x} + k\underline{x} - 1)\mathrm{e}^{j\omega t}\right\} = 0,$$

somit muss $j\omega\underline{x} + k\underline{x} - 1 = 0$ gelten. Nach $\underline{x}$ aufgelöst erhält man

$$\underline{x} = \frac{k - j\omega}{k^2+\omega^2}.$$

Zur Berechnung der Lösung

$$x_{freq}(t) = \frac{k\cos(\omega t) + \omega\sin(\omega t)}{k^2 + \omega^2}. \tag{5}$$

im Frequenzbereich (2) setzt man den berechneten Phasor $\underline{x}$ und die von der Eulerschen in die Polare Darstellung umgerechneten Werte $\mathrm{e}^{j\omega t} = \cos(\omega t) + j\sin(\omega t)$ in x_{freq} ein.

Zwar wird mit dem Ansatz der Lösung im Frequenzbereich das transiente Anfangsverhalten vernachlässigt, jedoch ist die Lösung, verglichen mit dem Ansatz zum Ermitteln des Zeitbereichs, deutlich leichter und weniger zeitaufwändig, da man nur einmal eine sehr einfache Ableitung berechnen muss.

Zusätzlich wird eine Reihenschaltung aus einer Spule und einem Widerstand betrachtet ()LR-Schaltung). Diese wird in LT-Spice gebaut und simuliert. Bei der gegebenen Datei `Beispielschaltung2.asc` handelt es sich um eine AC-Simulation, aus der das Verhalten der Schaltung in Abhängigkeit von verschiedenen Frequenzen, in diesem Fall von $f = 100\,\mathrm{Hz}$ bis $f = 1\,\mathrm{MHz}$, abgelesen werden kann.

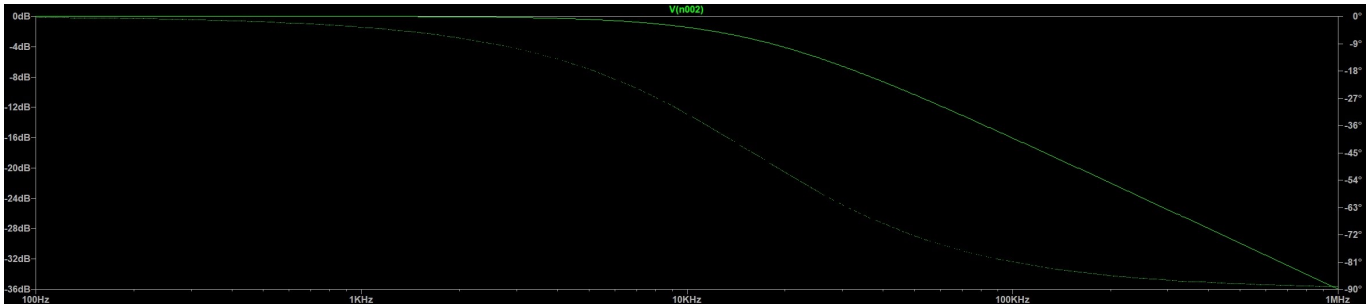


Abbildung 2.5: AC-Simulation aus LT-Spice

Die Schaltung lässt sich durch die Differentialgleichung

$$\frac{\mathrm{d}}{\mathrm{d}t}i(t) = -\frac{R}{L}i(t) + \frac{1}{L}u_i(t) \tag{6}$$

beschreiben wobei R der Widerstand, L die Induktivität, $i(t)$ der Strom und $u_i(t) = \cos(\omega t)$ die anregende Spannung ist. Die Lösung für diese Differentialgleichung im Zeitbereich lässt sich mit Hilfe von (4) ermitteln. Hierfür setzt man für $x(t) = i(t)$, $k = \frac{R}{L}$ und für $r(t) = \frac{1}{L}\cos(\omega t)$ ein. Erweitert man im selben Schritt auch noch die Brüche mit $\frac{L}{L}$ so folgt daraus als Lösung für (6)

$$i(t) = \mathrm{e}^{-\frac{R}{L}(t-t_0)}\left(x_0 - \frac{L\omega\sin(\omega t_0) + R\cos(\omega t_0)}{L^2\omega^2 + k^2}\right) + \frac{L\omega\sin(\omega t) + R\cos(\omega t)}{L^2\omega^2 + k^2}.$$

Analog ergibt sich die Lösung im Frequenzbereich durch einsetzen in (5) mit

$$i_{freq}(t) = \frac{R\cos(\omega t) + L\omega\sin(\omega t)}{R^2 + L^2\omega^2}.$$

Die Spannung über dem Widerstand lautet dann nach dem Ohm'schen Gesetz $U = R \cdot I$ und mit der Frequenz $f = \frac{\omega}{2\pi}$

$$u_R(t) = \frac{R\cos(2\pi ft) + 2\pi fL\sin(2\pi ft)}{R^2 + (2\pi fL)^2} R.$$

Das Schaubild für diese Funktion sieht man in Abbildung 2.6.

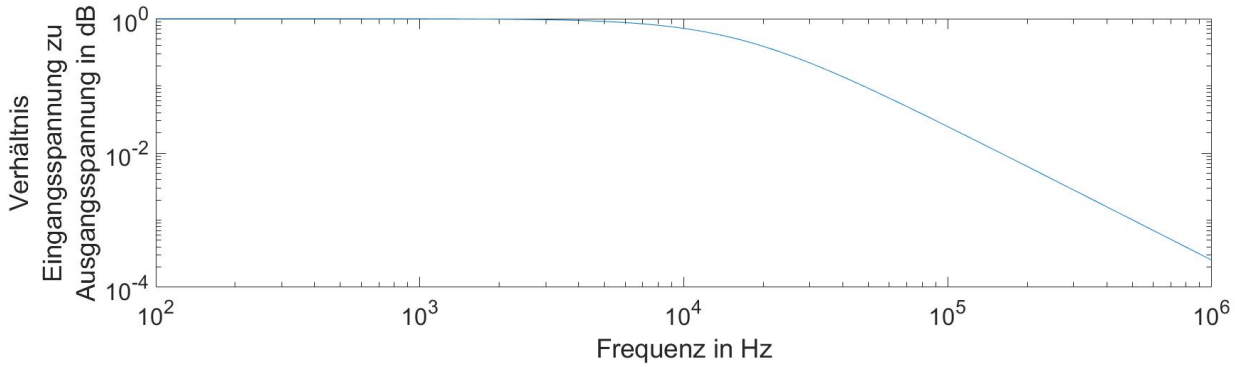


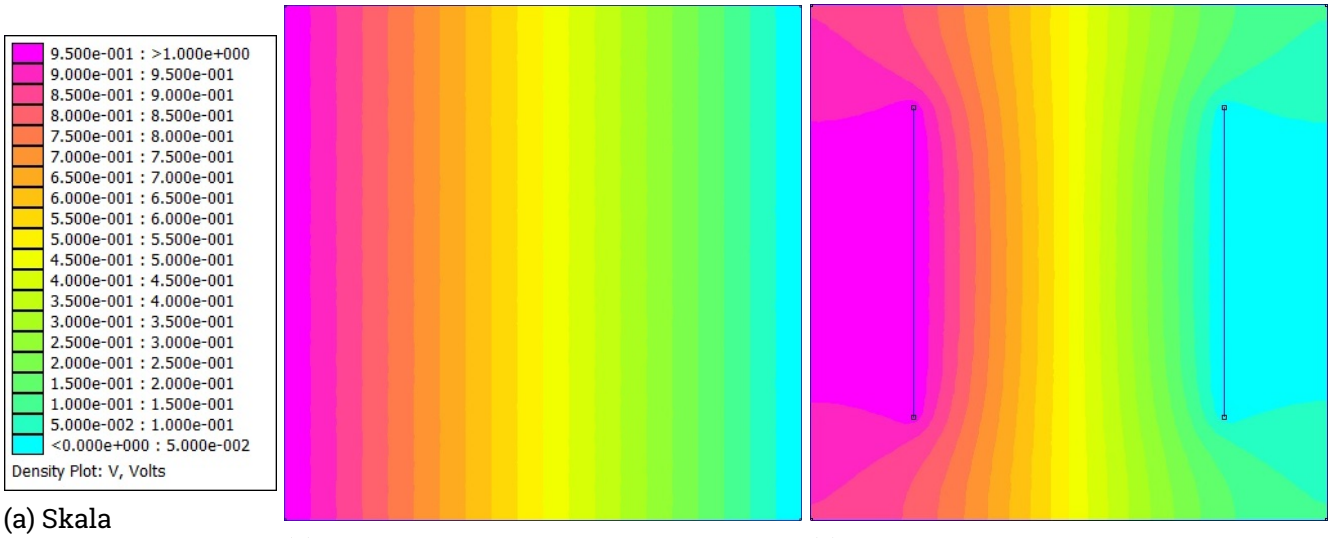
Abbildung 2.6: MATLAB-Plot der Spannung über dem Widerstand R

Anhand des Frequenzgangs fällt auf, dass es sich um eine Schaltung mit Tiefpass-Verhalten handelt, denn bei niedrigen Frequenzen bis ca. $10\,\mathrm{kHz}$ ist die Ausgangsspannung genauso groß wie die Eingangsspannung. Bei hohen Frequenzen nahe $1\,\mathrm{MHz}$ ist die Ausgangsspannung deutlich geringer als die Eingangsspannung.

2.3 Simulation homogener Plattenkondensator

In dieser Aufgabe wird ein harmonischer Plattenkondensator in dem Simulationsprogramm „FEMM “ nachgebaut. FEMM steht kostenfrei zur Verfügung und kann in Octave eingebunden werden.
Zunächst soll der Einfluss unterschiedlicher Randbedingungen auf das elektrische Feld eines Kondensators betrachtet werden.
Ein Kondensator ist eine Anordnung von zwei beliebigen Elektroden, die durch ein Dielektrikum von einander getrennt sind und die gleiche Ladung, aber mit unterschiedlicher Polarität in sich tragen, die Kapazität lässt sich mit $C = \epsilon_0 \epsilon_r \frac{A}{d}$ berechnen, wobei A die Fläche der Platten, d den Abstand der Platten, ϵ_0 die elektrische Feldkonstante und ϵ_r die relative Permittivität des Dielektrikums beschreibt. Außerdem kann die Formel $C = \frac{Q}{U}$ zur Berechnung der Kapazität benutzt werden, Q ist die Ladung in C und U die Spannung in V.

Zum Bau des Kondensators werden in FEMM vier Punkte gesetzt und diese zu Linien verbunden. Der Abstand der Punkte beträgt sowohl in x-, als auch in y-Richtung $30\,\mathrm{cm}$. Genau genommen wird ein Quader erstellt, da die Kondensatorplatten quadratisch sein sollen, hat der Quader eine Tiefe von $30\,\mathrm{cm}$, jedoch verfügt FEMM nur eine 2-Dimensionale graphische Darstellung.
Die linke und rechte Linie in Abbildung 2.7b stellen die beiden Kondensatorplatten dar, die obere und untere Linie dienen zur Berandung des elektrischen Felds. An der linken Platte des Kondensators liegt eine Spannung von $1\,\mathrm{V}$ an, an der rechten Seite eine Spannung von $0\,\mathrm{V}$. Des Weiteren wurde in Abbildung 2.7c ein Kondensator betrachtet, der nicht direkt, sondern durch einen äußeren Käfig berandet wird. Die inneren Linien markieren die beiden Kondensatorplatten, hier liegen die selben Spannungen wie im Vergleichsbild an.



(a) Skala

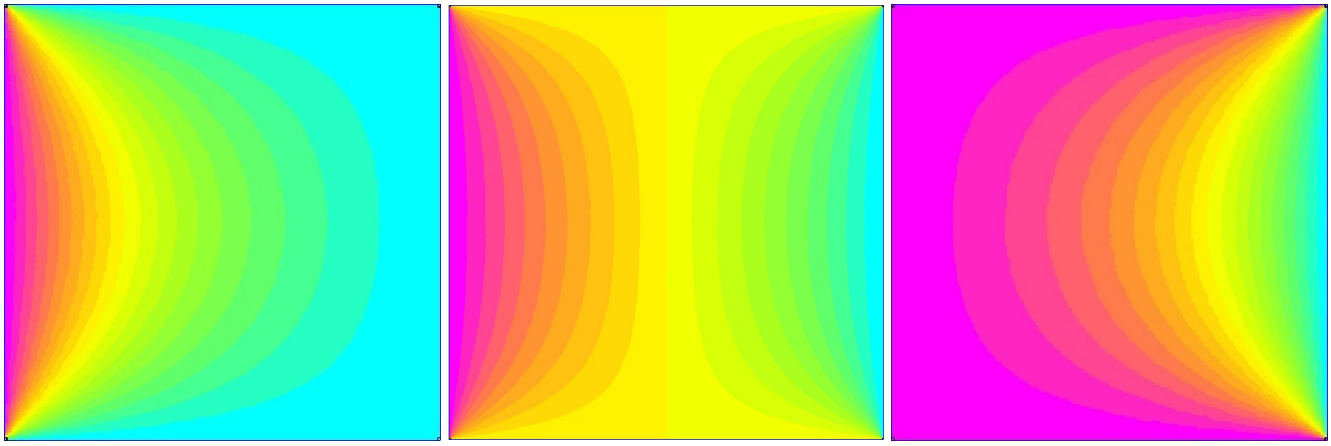
(b) Direkte Berandung

(c) Berandung durch Käfig

Abbildung 2.7: Aufbau und Feldlinien des Kondensators

Vergleich man nun den Plot 2.7b mit 2.7c, so wird eindeutig, dass die Feldlinien in Abbildung 2.7b senkrecht zu den beiden Flächen stehen. Berandet man den Kondensator durch einen Käfig, so werden auch die nicht senkrecht verlaufenden Feldlinien beachtet. In beiden Fällen entsteht ein homogenes Feld. Auch bei der Kapazität lassen sich unterschiedliche Werte ablesen. Während der Kondensator 2.7b eine Kapazität von $2{,}6562\,\mathrm{pC}$ hat, liegt die Kapazität von Kondensator 2.7c bei $3{,}9773\,\mathrm{pC}$

Zusätzlich soll verglichen werden, wie sich verschiedene Randbedingungen auf das entstehende elektrische Feld auswirken. Hierzu wurden dem Rand verschiedene Werte zugewiesen. Zunächst in Abbildung 2.8a eine Spannung von $0\,\mathrm{V}$, dann in Abbildung 2.8b $0{,}5\,\mathrm{V}$ und schließlich eine Spannung von $1\,\mathrm{V}$ in Abbildung 2.8c. Es gilt die Selbe Skala, die in 2.7a zu sehen ist.



(a) Rand mit 0V Spannung (b) Rand mit 0,5V Spannung (c) Rand mit 1V Spannung

Abbildung 2.8: Darstellung unterschiedlicher Randbedingungen

Da nun unterschiedliche Randbedingungen herrschen, entstehen auch Feldlinien zwischen den Kondensatorplatten und dem Rand. Es ergeben sich nicht nur unterschiedliche Feldlinien, sondern auch verschiedene Kapazitäten an den beiden Platten.

Spannung am Rand	Kapazität linke Kondensatorplatte	Kapazität rechte Kondensatorplatte
$0\,\mathrm{V}$	$22{,}1405\,\mathrm{pC}$	$-0{,}5861\,\mathrm{pC}$
$0{,}5\,\mathrm{V}$	$11{,}3633\,\mathrm{pC}$	$-11{,}3633\,\mathrm{pC}$
$1\,\mathrm{V}$	$0{,}5861\,\mathrm{pC}$	$-22{,}1405\,\mathrm{pC}$

Die Feldlinien werden offensichtlich durch die anliegende Spannung am Rand beeinflusst und damit auch die Kapazität des Kondensators.

FEMM lässt sich nicht nur mit Hilfe von Octave starten, sondern es besteht auch die Möglichkeit eine OctaveFEMM Routine zu schreiben und damit ein Modell zu erstellen, sowie eine Simulation dieses Modells zu starten. In dem gegebenen Fall, soll eine Routine geschrieben werden, die zwei Parameter a und h entgegen nimmt, wobei a die Kantenlänge und h der Abstand der Platten ist. Aus diesen Informationen soll ein Kondensator erstellt und die berechnete Kapazität C zurückgeliefert werden. Der Quellcode befindet sich im Anhang.

Zunächst wird in Zeile 7 bis 9 FEMM geöffnet und definiert um welche Art von Problem es sich handelt. Zusätzlich werden einige Einstellungen getroffen, wie die Einheit der Variablen, die Tiefe des Modells und die Rechengenauigkeit.
In Zeile 10 bis 13 werden die verschiedenen Properties erzeugt, also die Randbedingungen (falls gewünscht), die Kondensatorplatten und das Dielektrikum.
Durch den Befehl in Zeile 16 wird daraufhin ein Rechteck, mit Hilfe der an die Methode übergebenen Parameter, erstellt. Ein Blocklabel für das Dielektrikum wird in Zeile 19 erzeugt. Von Zeile 22 bis 38 werden nun die einzelnen Properties den richtigen Linien und Punkten zugewiesen.
Bevor die Simulation starten kann, muss das Projekt gespeichert werden. Danach wird die Analyse auf dieser Datei gestartet und die Lösung graphisch auf dem Bildschirm angezeigt.
Der zurückzuliefernde Parameter C ergibt sich nun aus dem Ergebnis der Ladung geteilt durch die Spannung. Diese sind in einem liegenden Vektor mit zwei Spalten gespeichert. Die Ladung an zweiter und die Spannung an erster Stelle.

3 Fazit

Dies ist das Fazit.

4 Anhang

Methode circuit_matrices

```
function [AC,AL,AR,AV,AI,C,L,R,V,I] = circuit_matrices(filename)

% PEMCE Uebung 2: Routine um Schaltunsmatrizen zu erzeugen:
%  [AC,AL,AR,AV,AI,C,L,R,V,I] = circuit_matrices(filename)
%
% Eingabe:
%  filename  - Dateiname der Netzliste
%
% Ausgaben:
%  AC,...,AI - reduzierte Inzidenzmatrizen fuer Kapazitaten, ..., Stromquellen
%  C,...,I   - Matrizen der Parameter (Kapazitaten, ..., Stromquellen)

%# lese Schaltung aus netzliste
cirstruct = prs_spice (filename);

%# initialisiere variablen
n = cirstruct.totextvar;
AC=AL=AR=AV=AI=sparse(n,0);
C=L=R = zeros(0,0);
V=I= zeros(0,1);

% Durchlaufe alle Elementtypen (R,L,C,...)
for i=1:size(cirstruct.LCR,2)
typ=cirstruct.LCR(i).parnames{1,1};
sz=size(cirstruct.LCR(i).pvmatrix,1);

% Durchlaufe alle Bauteile eines Typen
for j=1:size(cirstruct.LCR(i).pvmatrix,1)
val=cirstruct.LCR(i).pvmatrix(j,:);
pins=cirstruct.LCR(i).vnmatrix(j,:);

if typ == 'L'
matrix = AL;
diagonal = L;
endif
if typ == 'R'
matrix = AR;
diagonal = R;
endif
if typ == 'C'
matrix = AC;
diagonal = C;
endif
if typ == 'V'
```

```
45 matrix = AV;
46 diagonal = V;
47 endif
48 if typ == 'I'
49 matrix = AI;
50 diagonal = I;
51 endif
52
53 is_voltage_or_current = typ == 'V' || typ == 'I';
54
55 matrix = [matrix,sparse(n,1)]
56 %Diagonalmatrix Elemente einfuegen
57
58 if is_voltage_or_current == false
59 diagonal = [diagonal,zeros(sz,1)];
60
61 if typ == 'R'
62 val = 1/val;
63 endif
64
65 diagonal(j,j) = val;
66 endif
67
68 %Richtung des Stroms bestimmen
69 direction = pins(1) - pins(2);
70
71 %Strom fliesst von 1 nach 2
72 if direction < 0
73 %Nur hinzufuegen wenn es sich nicht um den Masseknoten handelt
74 if pins(1) != 0
75 matrix(pins(1),j)=1;
76 endif
77 matrix(pins(2),j)=-1;
78
79 if is_voltage_or_current == true
80 diagonal = [diagonal;val];
81 endif
82
83 %Strom fliesst von 2 nach 1
84 else
85 matrix(pins(1),j)=1;
86 %Nur hinzufuegen wenn es sich nicht um den Masseknoten handelt
87 if pins(2) != 0
88 matrix(pins(2),j)=-1;
89 endif
90
91 if is_voltage_or_current == true
92 diagonal = [diagonal;-val];
93 endif
94 endif
95
96 if typ == 'L'
97 AL = matrix;
98 L = diagonal;
99 endif
100 if typ == 'R'
101 AR = matrix;
102 R = diagonal;
```

```
endif
if typ == 'C'
AC = matrix;
C = diagonal;
endif
if typ == 'V'
AV = matrix;
V = diagonal;
endif
if typ == 'I'
AI = matrix;
I = diagonal;
endif

endfor
endfor
endfunction
```

Listing 4.1: Methode `circuit_matrices` in Octave

Methode calculate_matrices

```
function [M,K,r] = calculate_matrices(filename)

%Berechnet die Matrizen M,K und r
[AC,AL,AR,AV,AI,C,L,R,V,I] = circuit_matrices(filename);
bl = size(L,1);
bv = size(V,1);
n = size(AR,1);

M = [AC*C*AC' zeros(n,bl) zeros(n,bv);
zeros(bl,n) L zeros(bl,bv);
zeros(bv,n) zeros(bv,bl) zeros(bv,bv)];

K = [AR*R*AR' AL AV;
-AL' zeros(bl,bl) zeros(bl,bv);
-AV' zeros(bv,bl) zeros(bv,bv)];

r = [-AI*I; zeros(bl,1); -V];
endfunction
```

Listing 4.2: Methode `calculate_matrices` in Octave

Finales Skript zur Berechnen der Spannung an Rf (n=1)

```
t = [0:1e-8:1e-3];

% Bestimmung des Gleichungssystems:
[M,K,r] = calculate_matrices('koaxialKabel.net');

% r(5) (Spannungsquelle) wird wohl mit falschem Vorzeichen berechnet, diese Korrektur fuehrt
    nun zum richtigen Ergebnis
r(5) = -r(5);
```

```
8
9  %res = @(y, yd, t) (M*yd+K*y-r);
10 res =@(x,xdot,t)(M*xdot+K*x-r);
11
12 % Bestimmung der Anfangswerte:
13 U_q = 12;
14 L1 = 0.00025;
15 R1 = 0.001;
16 C1 = 0.0000001;
17 R2 = 1500;
18 Rf = 1500;
19
20 % x0:
21 % phi_1 = 12
22 % phi_2 = 12
23 % phi_3 = 0
24 % i_L   = 0
25 % i_V   = 0
26 x0 = zeros(size(r));
27 x0(1) = U_q;
28 x0(2) = U_q
29
30 % x0_Strich:
31 x0_Strich = zeros(size(r));
32
33 % Nummerische Berechnung des Gleichungssystems:
34 x = daspk(res,x0,x0_Strich,t);
35
36 % Erstellen des Plots:
37 plot(t,x(:,3:3));
38 xlabel('Zeit t in [s]', 'interpreter', 'tex')
39 ylabel('Spannung in [V]', 'interpreter', 'tex')
40 legend('\phi_3 entspricht Spannung an Rf')
```

Listing 4.3: `FinalesSkript` in Octave

Netzliste für n=10

```
1  Vs A1N1 0 12
2  Ro1 A1N1 A1N2 0.001
3  L1 A1N2 A1N3 0.00025
4  Ru1 0 A1N3 1500
5  C1 A1N3 0 0.0000001
6
7  Ro2 A1N3 A2N2 0.001
8  L2 A2N2 A2N3 0.00025
9  Ru2 0 A2N3 1500
10 C2 A2N3 0 0.0000001
11
12 Ro3 A2N3 A3N2 0.001
13 L3 A3N2 A3N3 0.00025
14 Ru3 0 A3N3 1500
15 C3 A3N3 0 0.0000001
16
17 Ro4 A3N3 A4N2 0.001
18 L4 A4N2 A4N3 0.00025
```

```
Ru4 0 A4N3 1500
C4 A4N3 0 0.0000001

Ro5 A4N3 A5N2 0.001
L5 A5N2 A5N3 0.00025
Ru5 0 A5N3 1500
C5 A5N3 0 0.0000001

Ro6 A5N3 A6N2 0.001
L6 A6N2 A6N3 0.00025
Ru6 0 A6N3 1500
C6 A6N3 0 0.0000001

Ro7 A6N3 A7N2 0.001
L7 A7N2 A7N3 0.00025
Ru7 0 A7N3 1500
C7 A7N3 0 0.0000001

Ro8 A7N3 A8N2 0.001
L8 A8N2 A8N3 0.00025
Ru8 0 A8N3 1500
C8 A8N3 0 0.0000001

Ro9 A8N3 A9N2 0.001
L9 A9N2 A9N3 0.00025
Ru9 0 A9N3 1500
C9 A9N3 0 0.0000001

Ro10 A9N3 A10N2 0.001
L10 A10N2 A10N3 0.00025
Ru10 0 A10N3 1500
C10 A10N3 0 0.0000001

Rf 0 A10N3 1500
```

Listing 4.4: Netzliste für Koaxialkabel mit 10 Segmenten in Octave

Finales Skript Vergleich n=1 und n=10

```
t = [0:1e-8:2e-3];

% Bestimmung des Gleichungssystems:
[M,K,r] = calculate_matrices('koaxialKabel.net');

%r(5) (Spannungsquelle) wird wohl mit falschem Vorzeichen berechnet
r(5) = -r(5);

%res = @(y, yd,t) (M*yd+K*y-r);
res =@(x,xdot,t)(M*xdot+K*x-r);

% Bestimmung der Anfangswerte:
U_q = 12;
L1 = 0.00025;
R1 = 0.001;
C1 = 0.0000001;
R2 = 1500;
```

```
18 Rf = 1500;
19
20 % x0:
21 % phi_1 = 12
22 % phi_2 = 12
23 % phi_3 = 0
24 % i_L   = 0
25 % i_V   = 0
26 x0 = zeros(size(r));
27 x0(1) = U_q;
28 x0(2) = U_q
29
30 % x0_Strich:
31 x0_Strich = zeros(size(r));
32
33 % Nummerische Berechnung des Gleichungssystems:
34 % x = dassl(res, transpose(x0), zeros(size(r)), t);
35 x1 = daspk(res,x0,x0_Strich,t);
36
37
38 %-----------------------------------------------------
39 % Bestimmung des Gleichungssystems mit 10 Segmenten:
40 %-----------------------------------------------------
41 [M10,K10,r10] = calculate_matrices('KK10.net');
42
43 %Spannungsquelle wird wohl mit falschem Vorzeichen berechnet
44 r10 = -r10;
45
46 %res = @(y, yd,t) (M*yd+K*y-r);
47 res10 =@(x,xdot,t)(M10*xdot+K10*x-r10);
48
49 % Bestimmung der Anfangswerte:
50 U_q = 12;
51 L1 = 0.00025;
52 R1 = 0.001;
53 C1 = 0.0000001;
54 R2 = 1500;
55 Rf = 1500;
56
57 % x0:
58 % phi_1 = 12
59 % phi_2 = 12
60 % phi_3 = 0
61 % i_L   = 0
62 % i_V   = 0
63 x0_10 = zeros(size(r10));
64 x0_10(1) = U_q;
65 x0_10(2) = U_q
66
67 % x0_Strich:
68 x0_10_Strich = zeros(size(r10))
69
70 % Nummerische Berechnung des Gleichungssystems:
71 % x = dassl(res, transpose(x0), zeros(size(r)), t);
72 x10 = daspk(res10,x0_10,x0_10_Strich,t);
73
74 %-----------------------------------------------------
75 % Erstellen des Plots:
```

```
plot(t,x1(:,3:3));
hold on;
plot(t,x10(:,21:21));
hold off;
xlabel('Zeit t in [s]', 'interpreter', 'tex')
ylabel('Spannung in [V]', 'interpreter', 'tex')
legend('Ausgangsspannung bei n=1', 'Ausgangsspannung bei n=10')
```

Listing 4.5: SkriptSegmentVergleich1vs10 in Octave

```

## Author: D. Schiller, S. Arnold, T. Lingenberg, C. Kramer
## Created: 2020-11-25

function C = femmbuilder (a, h)

  openfemm;
  newdocument(1);
  ei_probdef('centimeters','planar',1.E-8,a,30);
  ei_addmaterial('Vakuum',1,1,0);
  %ei_addboundprop('Randbedingung','Vs',0,0,0,0);
  ei_addconductorprop('Linker Rand',1,0,1);
  ei_addconductorprop('Rechter Rand',0,0,1);

  %Randpunkte fuer den Kondensator erstellen
  ei_drawrectangle(0,0,h,a);

  %Punkt fuer Materialproperty setzen
  ei_addblocklabel(h/2,a/2);

  %Linken und rechten Rand setzen
  ei_selectsegment(0,a/2);
  ei_setsegmentprop('<None>',0,1,0,0,'Linker Rand');
  ei_clearselected;
  ei_selectsegment(h,a/2);
  ei_setsegmentprop('<None>',0,1,0,0,'Rechter Rand');
  ei_clearselected;

  %Oberen und unteren Rand setzen
  ei_selectsegment(h/2,0);
  ei_selectsegment(h/2,a);
  ei_setsegmentprop('<None>',0,1,0,0,'<None>');
  ei_clearselected;

  %Material setzen
  ei_selectlabel(h/2,a/2);
  ei_setblockprop('Vakuum',1,0,0);
  ei_clearselected;

  %Speichere die Datei ab
  ei_saveas('Aufgabe3_b.FEE');

  %Lade die gepeicherte Datei und erzeuge Simulation
  ei_analyze(0);
  ei_loadsolution;

  %Liegenden Vektor erstellen, Ladung / Spannung ergibt Kapazit\"at C
  G = [0,0];
```

```
49   G = eo_getconductorproperties('Linker Rand');
50   C = G(1,2)/G(1,1);
51
52 endfunction
```

data/femmbuilder.m

```
1  x = 100:1:1000000;
2  R = 100;
3  L = 0.001;
4  t = 1
5  y = R.*(R.*cos(2.*pi.*x.*t)+2.*pi.*L.*x.*sin(2.*pi.*x.*t))./(R^2+(L.*2.*pi.*x).^2);
6
7  figure;
8  loglog(x,y);
9  set(gca,'fontsize',20)
10 xlabel 'Frequenz in Hz';
11 ylabel ({'Verhaeltnis', 'Eingangsspannung zu' 'Ausgangsspannung in dB'});
```

data/A32d.m

Abbildungsverzeichnis