**Fachbericht**



**«DJ» EMI Filter für Netzteil**

Pro2E - Team 5

**Auftraggeber:** Luca Dalessandro

**Dozierende:** Anita Gertiser

Pascal Buchschacher

Peter Niklaus

Sebastian Gaulocher

Richard Gut

**Projektteam:** Marina Taborda, Projektleiterin

Michel Alt, Stv. Projektleiter

Frank Imhof

Luca Krummenacher

Richard Britt

Fady Hanna

Windisch, 06.06.2019

**Abstract**

*Alle elektronischen Geräte benötigen heute Netzschaltteile, um die Netzspannung in die gewünschte Gerätespannung zu transformieren. Die verwendeten Bauteile in den Schaltnetzteilen erzeugen dabei leistungsgebundene Störungen.*

*Damit diese Störungen (Differential Mode und Common Mode) auf der Netzseite keine negativen Auswirkungen haben, müssen die Netzschaltteile die normativen Anforderungen erfüllen und dementsprechend dimensioniert werden.*

*Diese Normen werden von den europäischen Normengremien festgelegt und sind ständig in Bearbeitung.*

*Das Ziel dieser Arbeit ist die Entwicklung eines bedienerfreundlichen Programms, das in der Lage ist, die Einfügungsverluste des Filters zu berechnen, sowie graphisch darzustellen. Das Programm soll ausserdem aufzeigen, wie sich bestimmte Parameter, auf die zwei Störungsarten im ganzen Frequenzspektrum auswirken. Diese Informationen sollten dem Auftraggeber das Dimensionieren von Filtern vereinfachen.*

Problemstellung

Ziel/Anforderung

Methodik

Hauptresultate

Konklusion

Inhaltsverzeichnis

[1 Einleitung 4](#_Toc9597746)

[2 Theoretische Grundlagen 5](#_Toc9597747)

[2.1 EMI Filter 5](#_Toc9597748)

[2.2 Störungsarten 5](#_Toc9597749)

[2.3 Definition Einfügedämpfung «Insertion loss» 6](#_Toc9597750)

[2.4 Parasitäre Parameter 6](#_Toc9597751)

[2.5 MVC 8](#_Toc9597752)

[3 Software 8](#_Toc9597753)

[3.1 Betriebssystem/Lizenzierung: 8](#_Toc9597754)

[3.2 Strukturierung 9](#_Toc9597755)

[3.2.1 Klassendiagram 9](#_Toc9597756)

[3.3 Programmablauf **Error! Bookmark not defined.**](#_Toc9597757)

[3.3.1 Programmstart 10](#_Toc9597758)

[3.3.2 Laden und Speichern von Daten 10](#_Toc9597759)

[3.3.3 Eingabe der Daten 10](#_Toc9597760)

[3.4 Implementierung der Berechnungen in JAVA 11](#_Toc9597761)

[3.5 GUI 11](#_Toc9597762)

[3.5.1 Architektur 11](#_Toc9597763)

[4 Elektrotechnik 12](#_Toc9597764)

[4.1 Vereinfachung der CM-Schaltung 12](#_Toc9597765)

[4.2 Vereinfachung der DM-Schaltung 14](#_Toc9597766)

[4.3 Vorgehensweise Berechnung Einfügungsverluste 17](#_Toc9597767)

[4.4 S-Parameter 18](#_Toc9597768)

[4.5 CM-Kurven Realisierung mit Matlab 20](#_Toc9597769)

[4.6 DM-Kurven Realisierung mit Matlab **Error! Bookmark not defined.**](#_Toc9597770)

[5 Validierung 21](#_Toc9597771)

[5.1 Berechnungen 21](#_Toc9597772)

[5.2 Software 21](#_Toc9597773)

[5.2.1 Eingabe 21](#_Toc9597774)

[5.2.2 Benutzerfreundlichkeit 21](#_Toc9597775)

[6 Schlusswort 22](#_Toc9597776)

[Quellenverzeicnis 23](#_Toc9597777)

[Anhang 24](#_Toc9597778)

# Einleitung

In separatem word doc

# Theoretische Grundlagen

## EMI Filter

Nahezu jedes elektrische Gerät besitzt ein Schaltnetzteil, um die Netzspannung auf die benötigte Spannung zu regeln. Betrachtet man die Eingangsspannung ohne Netzfilter, wird man auf dem ganzen Frequenzspektrum, d.h. von Netzfrequenz bis zu mehreren MHz Störungen feststellen. Die Aufgabe vom EMI (elektromagnetische Interferenzen) Filter ist es, diese Störungen zu filtern, so dass keine anderen Geräte gestört werden. Damit dieses eingesetzt werden darf, muss jedes Schaltnetzteil sich an bestimmte Normen im Bereich EMV halten.

Ein EMI Filter für einphasige Geräte besteht nur aus wenigen Bauteilen. Zwei X-Kondensatoren, zwei Y-Kondensatoren, einer Drossel mit zwei Windungen, welche um einen Ferit Ring gewickelt sind und einem Widerstand. Diese Schaltung kann sehr kompakt verbaut werden, was in folgendem Filter von Schaffner sichtbar wird.

Abbildung 1:Schaltung des FN 2020 Filters (Schaffner), sowie der Filter selbst

Die stromkompensierte Drossel (L) ist in der Lage, Gleichtaktstörungen (CM) zu filtern. Diese Störungen treten gleichzeitig auf beiden Leitungen auf.

Die Y-Kondensatoren, welche gegen Erde geschaltet sind, sind ebenfalls dazu da, um CM-Störungen zu filtern. Diese müssen jedoch eine sehr hohe Überspannungsfestigkeit besitzen, um beispielsweise bei einem Blitzschlag keinen Kurzschluss im Gehäuse zu verursachen.

Störungen zwischen den Zuleitungen, so genannte Gegentaktstörungen (DM), werden mithilfe der X-Kondensatoren gedämpft. [1]

## Störungsarten

Die existierenden EMV Normen gelten für das Gesamtrauschen. Doch in der Praxis wird einfachheitshalber die Gesamtstörung in Gleichtaktrauschen, Comon Mode (CM) und Gegentaktrauschen, Differential Mode (DM) gesprochen. TODO Erklären unterschied und Ursache CM / DM MOde

Abbildung 2: Stromzirkulation der Störungen im DM- und CM-Mode

## Definition Einfügedämpfung «Insertion loss»

Die Leistung eines EMI Filters wird mit den Einfügedämpfung in Abhängigkeit der Frequenz bestimmt. Diese Funktion lautet:

: Lastspannung gemessen ohne Filter mit einer Last von 50 Ω

: Lastspannung gemessen mit EMI Filter

Abbildung 3: Lastspannung ohne Filter und Lastspannung mit EMI Filter

## Parasitäre Parameter

In der Realität verhalten sich die Bauteile eines EMI-Filters leider nicht genau so, wie wir es idealerweise annehmen. Jedes Bauteil hat aufgrund der physikalischen Gegebenheiten eine Ersatzschaltung mit den parasitären Parametern. Diese Ersatzschaltungen sehen wie folgt aus:

A

Abbildung 4: Ersatzschaltung für einen realen Kondensator - eine reale Spule

Betrachtet man diese Ersatzschaltbilder, sieht man eigentlich einen Serie- und Parallelschwingkreis. Weit unterhalb der Resonanzfrequenz haben die parasitären Parameter jedoch keinen grossen Einfluss. Im Resonanzfall haben wir aber einen Serie- oder Parallelschwingkreis. Danach wirkt die Spule nicht mehr als Induktivität und der Kondensator nicht mehr als Kapazität. Die Resonanzfrequenzen liegen dabei meist bei mehreren MHz. Doch bei unserer Anwendung wollen wir auch über diesem Frequenzbereich unsere Schaltung berechnen. Der Widerstand hat auch ein Ersatzschema, dieses wurde aber in dieser Schaltung nicht berücksichtigt, da dieses keinen grossen Einfluss auf die Gesamtschaltung hat.

Deshalb hat man für das Grundschema des Filters in Abbildung 5, die parasitären Parameter gemäss folgenden Ersatzschemas in Abbildung 6 für die CM-Schaltung. Interessant dabei ist der 1MΩ Widerstand, welcher parallel zu CX2 liegt. Dieser wird aus Sicherheitsgründen benötigt, um den CX2 Kondensator zu entladen. In der Funktion des Filters hat er jedoch keinen Einfluss, weshalb er nicht in unsere Simulationen und Berechnungen einfliesst.



Abbildung 5: CM-Ersatzschaltung ohne parasitäre Parameter



Abbildung 6: CM-Ersatzschaltung mit parasitären Parametern

Für die DM-Ersatzschaltung in Abbildung 7 hat man ebenfalls mit parasitären Parametern ergänzt. Der 1MΩ Widerstand fällt genau wie bei der CM-Schaltung weg. Diese Schaltung ist in der Abbildung 8 ersichtlich.



Abbildung 7: DM-Ersatzschaltung ohne parasitäre Parameter



Abbildung 8: DM-Ersatzschaltung mit parasitären Parametern

# Software

*In diesem Kapitel wird das Softwaretool ”DJ” vorgestellt. Das Programm ist in der Sprache Java als Desktop-Applikation für die Betriebssysteme MacOS und Windows entwickelt worden. Um den Inhalt möglichst von der Darstellung trennen zu können, wird die Library JavaFX verwendet.*

*Das Programm sagt das Frequenzverhalten und die Einfügungsverluste von CM und DM des EMI-Filters vor. Die Auswirkungen, die die parasitären Parameter auf die Einfügungsverluste des Filters haben, werden im Programm ersichtlich dargestellt. Das Programm zeigt die beiden Filtervarianten hinsichtlich der Leistung. NOPE*

*Auf Wunsch des Arbeitgebers ähnelt die Bedienoberfläche einem Mischpult mit Schiebereglern.*

## Strukturierung



Die oberste Instanz bezüglich des Layouts der Applikation ist die Klasse View. Als erbende Klasse der Klasse Borderpane beinhaltet sie die Bereiche Top, Center und Bottom, die die in Abbildung NR gezeigten, rot umrahmten, Elementen enthalten.

1: CustomMenuBar

Die CustomMenuBar enthält die Menus File, Sweep, CISPR und Help. Alle MenuItems oder ToggleGroups, die in diesen Menus enthalten sind, sind mit einem eigenen anonymen Actionlistener ausgestattet um auf Mausclicks entsprechend reagieren zu können.

**File**

Save: Speichern der Schaltungsdaten. -> controller.writeCircuitToFile(file);

Load: Laden von Schaltungsdaten. -> controller.readFile(file)

**Sweep**

Hier kann eingestellt werden, wie die Schieberegler die Werte der parasitären Parameter verändern:

Linear +-30%

Logarithmic (10^-3 to 10^3)

**CISPR**

Hier kann aus drei verschiedenen Werten für die Widerstände (wie heissen sie?) R1/R2 gewählt werden.

**Help**

Tooltips sind bei guter Kenntnis des Tools abwählbar. Die Informationen über das Tool dessen Herausgeber sind ebenfalls aufrufbar über die Menübar.

2. Graphische Darstellung der Werte und Schemadarstellung:

Hier werden die berechnenden Werte der Sliderwerte ausgegeben in zwei Graphen(CM und DM- Graphen). Mit Hilfe der Tabs kann auf die Schemas der Schaltungen gewechselt werden und sieht sofort welche Werte man verändert.

3. Slider:

Hier sind die veränderbaren Parameterwerte, welche mit einem Doppelklick selber in ein Textfile geschrieben werden können. Die Sliderwerte sind einstellbar zwischen +- 30 Prozent. Auch sind die Slider so implementiert, dass bei Berührung eines Sliders dessen Auswirkung graphisch direkt ersichtlich ist.

## Struktur

Die Applikation ist im klassischen MVC (Model-View-Controller) und Observable/Observer Entwurfsmuster, gemäss Abbildung 7, implementiert.

Model View Controller (MVC) ist eine Architektur zur Unterteilung einer Software in die drei Komponenten; Datenmodell (Model), Präsentation (View) und Programmsteuerung (Controller). Die MVC-Architektur bildet die grundlegende Organisation und Interaktion zwischen Komponenten einer Applikation. Sie ermöglicht eine flexible Applikation, die Änderungen und Erweiterungen zulässt sowie die Wiederverwendung einzelner Komponenten ermöglicht. Das Model erhält Daten via Controller von der View und arbeitet mit diesen. Es bleibt dabei von der View und vom Controller unabhängig. Änderungen von Daten werden mittels Observer [gibt Datenänderungen an von diesen Datenabhängige Strukturen weiter] der View zurückgegeben. Die View stellt die Daten dar und ist die Schnittstelle zur Interaktion zwischen Benutzer und Applikation. Sie kennt das Modell, dessen Daten sie darstellt, ist aber nicht für die Verarbeitung dieser Daten zuständig. Die View ist vom Controller unabhängig. Bei Dateneingabe durch den Benutzer werden diese mittels Observer an den Controller weitergegeben. Der Controller verwaltet das Model und die View. Er wertet Datenänderungen von Model und View aus und leitet diese an den jeweils anderen weiter.



## View

### Topdown Beschreibung

In diesem Abschnitt wird beschrieben, wie die Bedienoberfläche aufgebaut ist.

Die oberste Instanz bezüglich des Layouts der Applikation ist die Klasse **View**. Als erbende Klasse der Klasse Borderpane beinhaltet sie die Bereiche Top, Center und Bottom, die die in Abbildung NR gezeigten, rot umrahmten, Elementen enthalten. Diese werden nun genauer beschrieben.

**TOP**

Im Top Bereich befindet sich die **CustomMenuBar**. Diese enthält die Menus File, Sweep, CISPR und Help.

**File:** Beinhaltet menuItemSaveAs und menuItemLoad.

**Sweep:** Beinhaltet die RadioMenuItems menuItemLin und menuItemLog

**CISPR**: Beinhaltet die RadioMenuItems menuItemRefR50\_50, menuItemRefR100\_01 und menuItemRefR01\_100.

Die RadioMenuItems von Sweep und CISPR sind als ToggleGroup groupiert, damit jeweils nur ein RadioMenuItem pro Gruppe angewählt werden kann.

**Help:** Beinhaltet das Menu menuHelp und das Menuitem menuItemAbout.

**CENTER**

Der mittlere Bereich der Applikation dient der Darstellung sowohl der Einfügungsverluste wie auch der Schemadiagramme. Um Platz zu sparen und das Design schlicht zu halten, wurde dieser Bereich als TabPane gestaltet. Durch anwählen des entsprechenden Reiters wird das Entsprechende Element (GraphPane oder die SchemaPane) angezeigt und das andere in den Hintergrund gerückt.

**GraphPane**

Die GraphPane beinhaltet 2 Instanzen der Klasse GraphBox: graphBoxCM und graphBoxDM. Diese werden durch eine entsprechende Layout-Hülle (HBox «graphContainer») horizontal nebeneinander dargestellt und vergrössern sich automatisch in der Breite bei vergrössern des Applikationsfensters.

**SchemaPane**

Genau gleich aufgebaut wie die Graphpane, nur enthält sie 2 Elemente der Klasse SchemaBox.

**GraphBox**

Die GraphBox beinhaltet eine Instanz der Klasse XYChartPane, die es erlaubt, Graphen logarithmisch darzustellen und Kurven mit unterschiedlichen Farben und Formen darzustellen.

**SchemaBox**

Die SchemaBox beinhaltet ein Label und eine Instanz der Klasse ImageView für die Darstellung des entsprechenden Schemas.

**BOTTOM**

In diesem Bereich werden alle Komponenten der Schaltung auf möglichst schlichte Art und Weise dargestellt. Sie besteht aus einer Instanz der Klasse CircuitPane (extends HBox), die in sich drin alle Komponenten der Schaltung als Instanzen der Klasse CompBox enthält und diese horizontal auslegt.

**CompBox**

Die CompBox (extends VBox) ist eines der zentralsten Elemente der Bedienoberfläche, da sie als Hauptschnittstelle zwischen Nutzenden und Applikation fungiert. Deshalb wurde grosser Wert darauf gelegt, das Element einfach und klar zu gestalten.

Die CompBox beinhaltet 2 Elemente fürs Layout und ordnet diese vertikal, also untereinander an: compBoxHeader und compBoxBody. Diese sind in der Abbildung NR rot umrahmt dargestellt.

CompBoxHeader beinhaltet in der Mitte ein Label für den Namen der Komponente und einen JFXHamburger auf der rechten Seite, mit dessen die Darstellung des unteren Bereichs verändert werden kann.

Beim Aufstarten des Programms befindet sich im compBoxBody eine HBox, die alle parasitären Parameter in der Slidervariante (SubCompSliderBox) horizontal auflistet. Mit Click auf den JFXHamburger wird diese HBox nun mit einer anderen HBox ausgetauscht, die die parasitären Parameter nun in der Textfeldvariante (SubCompSettingsBox) horizontal auflistet.

Auf diese Art und Weise bleibt das Layout der Applikation schlicht, ohne dabei an Funktionalität zu verlieren.

Was ist die View? -> Borderpane

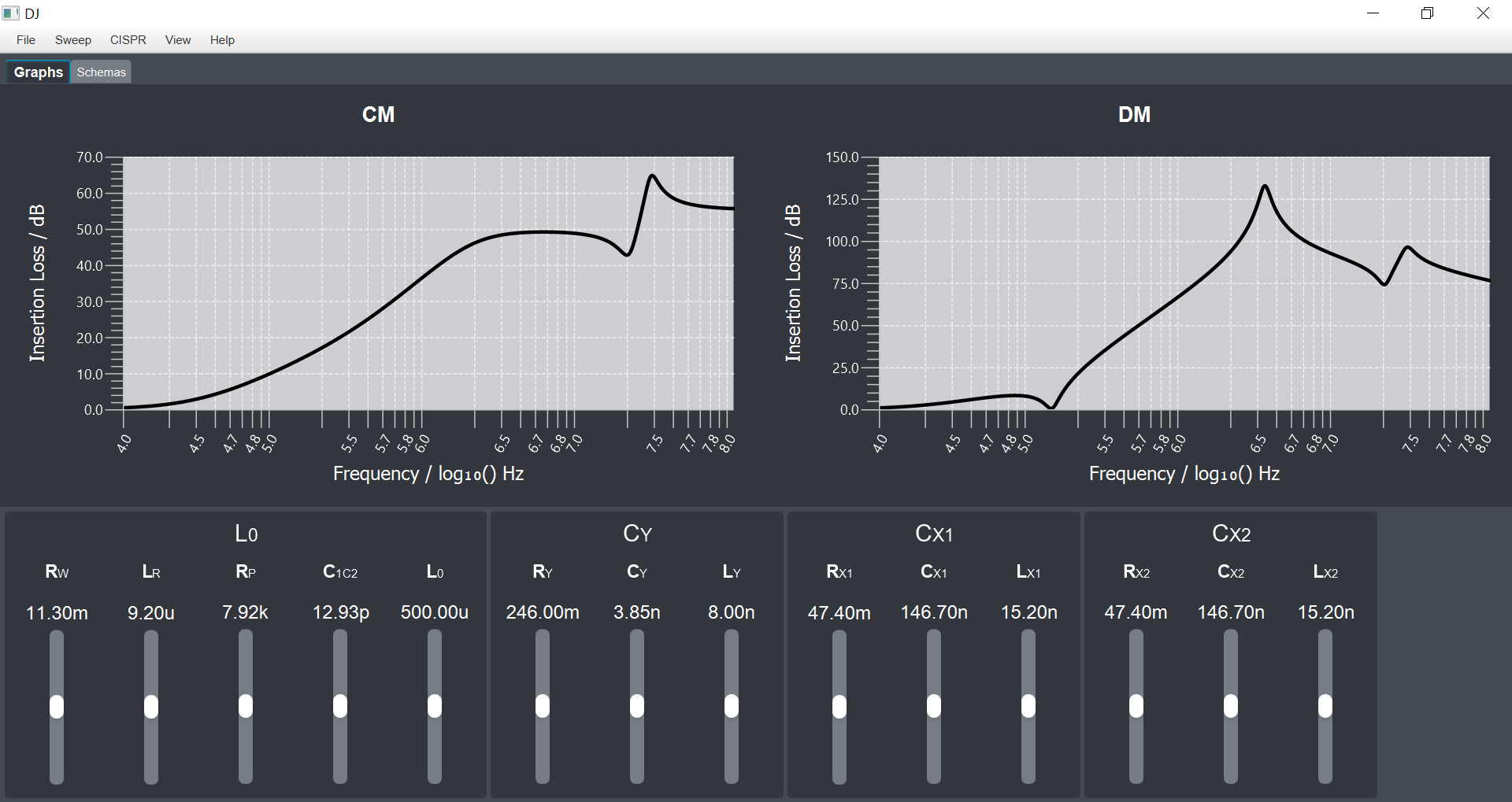
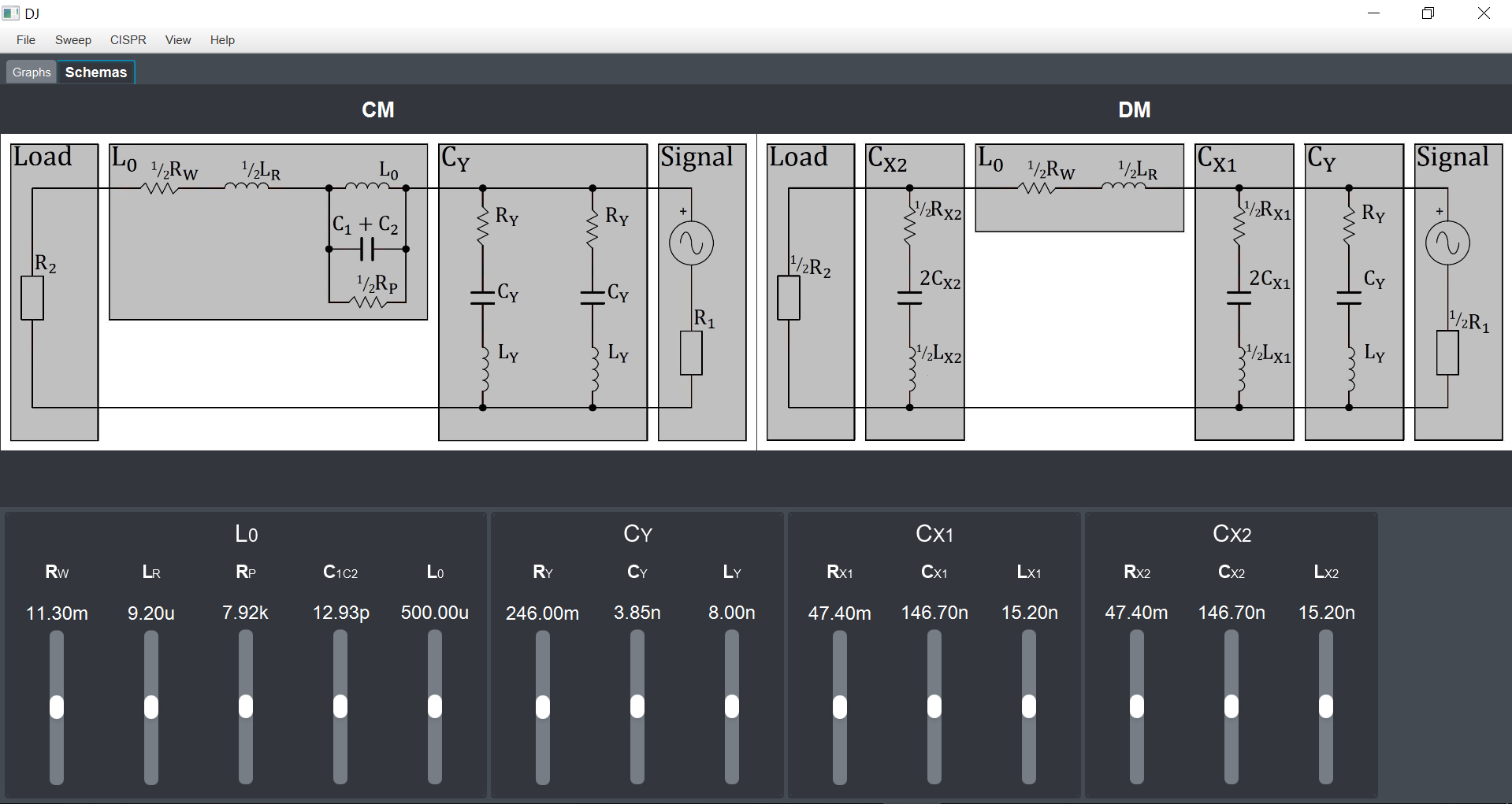
*Die View beinhaltet eine CustomMenuBar(Nummer 1 Abbildung), eine TabPane(Nummer 2 Abbildung) und eine ScrollPane(Nummer 3 Abbildung).*

*Die CustomMenuBar beinhaltet folgende Menus:*

File: Hier kann die Schaltung als Textfile abgespeichert werden. Um die Daten in einem lesbaren Textfile zu laden wurde die Textform JSON verwendet. Dies hat auch den Zweck die Daten einfacher wieder zurück ins Programm zu laden.

*Sweep: Hier kann die Funktion der Slider von linear auf logarithmisch umgeschalten werden. Dies wird mit der Klasse …….*

*CISPR: Hier kann die Dimensionierung der Referenzwiderstände geändert werden.*

In der TabPane befinden sich sowohl die GraphPane zur Darstellung der Diagramme wie auch die Schemapane zur Darstellung der vereinfachten Schaltungen.



In der ScrollPane befindet sich eine Liste von CompBox Objekten. Diese bilden die Schnittstelle für die Dimensionierung der Schaltung.

BILD VON COMPBOX OBJEKTEN

Jede CompBox stellt einen Komponenten der Schaltung dar und beinhaltet in sich drin eine VBox, wo die parasitären Parameter

Die CompBox beinhaltet in sich drin ein Label für die Darstellung des Komponentennamen und eine HBox zur Darstellung aller Im Anfangszustand zeigt die CompBox das Label mit dem Namen der Komponente und dessen. Dies sind im Normal Slider zu Setting gewechselt werden durch einen Klick auf den Hamburger.

## Controller

Der Controller hat die Aufgabe, die Benutzereingaben in der View zu prüfen und anhand dieser das Model zu beeinflussen.

Ebenfalls hat der Controller die Klassen schreibeDatei und ladeDatei. Diese werden gebraucht, um die Sliderwerte zu speichern und zu laden.

## Model

Das Model führt die nötigen Berechnungen durch. Dabei greift es auf folgende Klassen zu.

TwoPort:

In dieser abstrakten Klasse wird eine AMatrix erstellt. Dabei erbt sie von den Klassen SerieImpedance und ParallelImpedance um die AMatrix mit einer Parallelschaltung oder einer Serieschaltung zu erzeugen. Diese wird dann der Klasse Component weitergegeben.

Component:

Diese abstrakte Klasse erbt von den abstrakten Klassen IdealComponent und RealComponent die Idealen Komponente und die realen Komponente und kann damit alle Bauteile optimal erhalten. Die Klasse RealComponent erbt dabei noch von den Klassen RealInductor, RealResistor und RealCapacitor.

Circuit:

In dieser Klasse wird die Liste der TwoPort eingelesen und verrechnet.

In der View befinden sich sowohl die GraphPane zur Darstellung der Diagramme wie auch die Schemapane zur Darstellung der vereinfachten Schaltungen und die Klasse CustomMenuBar welche die Funktionen hat, welche in Kapitel……. beschrieben sind.

Um das Schema darzustellen, greift die Klasse Schemapane auf SchemaBox zu, in welcher sich das Schema als Bild befindet.

Das Diagramm wird in der Klasse GraphBox gezeichnet mit den Funktionen drawMainLine, drawSweepLines und createLineData.

## Implementierung der Berechnungen in JAVA

## Betriebssystem/Lizenzierung:

## Help: Tooltips an und ausschhalten und die About Seite lesen.3.2.2 Klassendiagram

Abbildung 7: Klassendiagr

## Programmablauf

## 3.8.1 Eingabe der Daten

*Beim Programmstart werden vorgegeben Parameterwerte automatisch eingelesen und dargestellt. Durch Verändern der parasitären Parameter wird das Neuzeichnen der Kurven ausgelöst. Zum Auslesen von einzelnen Werten kann man mit der Maus über die Diagramme fahren. Dabei werden die Mauswerte direkt neben der Maus angezeigt.*



Abbildung 9: Anpassung der Parameter

*Bei Doppelklick auf einen Parameter wechselt die Darstellung in seinem Fenster.  
Nun können die ursprünglichen Werte der parasitären Parameter via Textfelder geändert werden. Bei erneutem Doppelklick geht die Ansicht wieder zurück zu den Schiebereglern.*

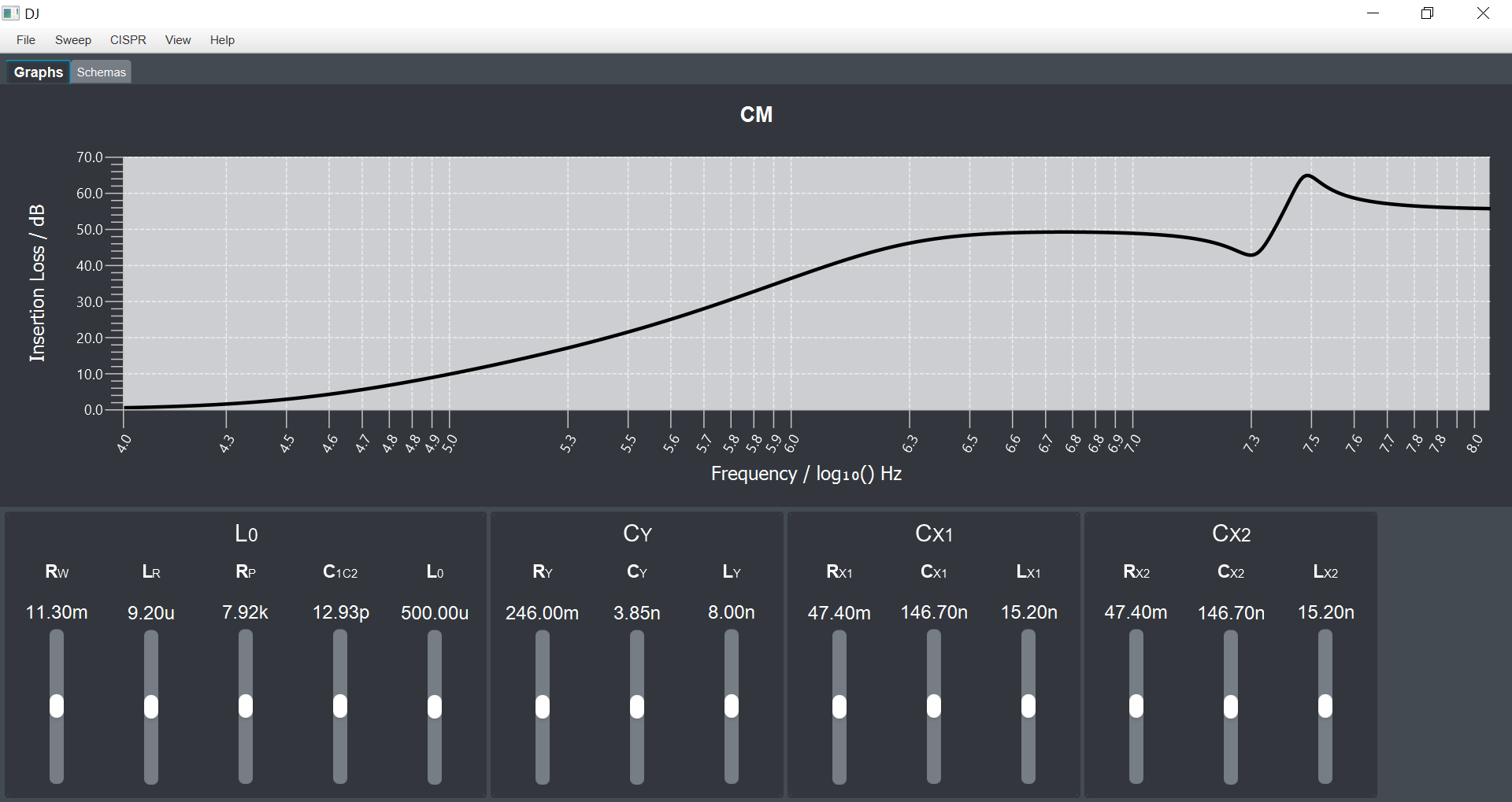


Abbildung 10: Kurvenansicht

*Bei Doppelklick auf ein Diagramm, übernimmt dieses die ganze Breite des Fensters und kann so genauer angesehen werden, wie man in Abbildung 10 sehen kann. Bei erneutem Doppelklick wird der Vorgang rückgängig gemacht, so dass wieder beide Diagramme sichtbar sind.*



Abbildung 11: Parameterauswirkung

*Wenn Nutzende den Mauszeiger über einen Schieberegler bewegen, wird im Diagramm gezeigt, welche Auswirkungen dieser Schieberegler bei voller Auslenkung (±30%) auf die Kurve hätte. Somit können die unterschiedlichen Auswirkungen aller parasitären Parameter schnell eingesehen werden, ohne dass die Nutzenden die Schieberegler einzeln verschieben müssen, was man in Abbildung 11 sehen kann.*

# Elektrotechnik

Für die Berechnungen sind die Schaltungen mit den parasitären Elementen von Interesse, welche bereits in den theoretischen Grundlagen der Grundschaltung ergänzt wurden. Danach wurden die beiden Schaltungen gemäss den Regeln der Elektrotechnik vereinfacht, so dass Zweitore gebildet werden können. Daraus berechnen sich die S-Parameter. Mit diesen kann schlussendlich die Einfügedämpfung berechnet werden. Nachfolgend werden alle Schritte erklärt.

## Vereinfachung der CM-Schaltung



Abbildung 11: CM Grundschaltung

Für die Vereinfachung der Schaltung gehen wir von der Grundschaltung in Abbildung 11: CM Grundschaltung aus. Die CM-Störungen auch symmetrische Störungen genannt, fliessen in beiden Leitern in die gleiche Richtung. Das heisst die beiden Leiter liegen auf demselben Potential, somit haben die Bauteile zwischen den beiden Leitern (CX1, CX2, RX1, RX2, LX1, LX2, C4, C3) keinen Einfluss. Nun können die beiden Leiter zusammengefasst werden, da beide gegenüber Masse symmetrisch sind. Das Zusammenfassen der beiden Leiter entspricht einer Parallelschaltung der beiden Leiter. Deshalb ergeben sich folgende Faktoren für die vereinfachte Schaltung: Der parasitäre Widerstand Rw und die parasitäre Spule Lr halbieren ihre Werte, genau wie der Widerstand Rp1. Durch die Parallelschaltung der Kondensatoren C1 und C2 addieren sich bekanntlich die beiden Kapazitäten zu einer Gesamtkapazität C1 + C2. Die beiden Spulen L0 und L1 sind auf einem Ferritkern aufgewickelt. Da beide Spulen den gleichen Wicklungssinn besitzen, verdoppelt sich die Induktivität L0. Durch das Zusammenfassen der beiden Spulen halbiert sich ihr Wert wieder, woraus der ursprüngliche Wert der Spule resultiert. Die beiden Kondensatoren CY1 und CY2 mit ihren parasitären Elementen bleiben unbeeinflusst, da sie bereits gegen Masse geschalten sind. Nach diesen Vereinfachungen ergibt sich die Schaltung in Abbildung 12: Vereinfachte CM Schaltung.



Abbildung 12: Vereinfachte CM Schaltung

Die Validierung der Umformungen wurde mit zwei Simulationen in «MPLAB» bewerkstelligt. Im folgenden Diagramm (Abbildung 13: Vergleich Grundschaltung / Vereinfachte Schaltung) liegen die beiden Ergebnisse der Simulationen übereinander. Darin ist sichtbar, dass die Vereinfachung zum selben Ergebnis führt wie die Grundschaltung. Die beiden Schaltungen dazu sind im Anhang hinterlegt.



Abbildung 13: Vergleich Grundschaltung / Vereinfachte Schaltung

## Vereinfachung der DM-Schaltung



Abbildung 14:Grundschaltung DM

Die Grundschaltung der DM Schaltung ist in der Abbildung 14:Grundschaltung DM sichtbar. Die DM-Störungen, auch asymmetrische Störungen genannt, fliessen in einem Leiter hin und im anderen Leiter wieder zurück. Daher lassen sich die Bauelemente nicht so leicht vernachlässigen wie in der CM Schaltung. Der Ringkern mit den beiden Spulen L0 und L5 gekoppelt sind, hat in der DM-Schaltung keinen Einfluss, da sich die beiden Induktivitäten aufgrund des entgegengesetzten Feldes aufheben. Dadurch heben sich auch die parasitären Elemente (C1, C2, Rp, Rp1) auf. Es bleiben nur noch die die parasitären Parameter Rw und Lr der Spulen übrig. Damit wir wie auch schon in der CM Schaltung die beiden Leiter zusammenfassen können, müssen wir die Bauteile zwischen den beiden Leitern verdoppeln. Dafür werden die Widerstände und Spulen halbiert und die Kondensatoren verdoppelt. Dies betrifft auch den Innen- und den Lastwiderstand. Dabei wird die Funktion der Schaltung nicht verändert. Die 1:1 Trafos werden dabei vernachlässigt, da diese als ideal angenommen werden und im Filter nicht als Bauteil vorkommen. Die Kondensatoren C3 und C4 werden weggelassen, da sie einen vernachlässigbaren Einfluss auf die Schaltung haben. Dies wurde vom Auftraggeber akzeptiert. In Abbildung 15: DM Schaltung angepasst sieht man die angepasste Schaltung.



Abbildung 15: DM Schaltung angepasst

Bei der blau eingezeichneten Achse wurde die Schaltung aufgetrennt und gegen Masse gelegt. Dies ist möglich, da die gleichen Störungen in Hin- und Rückleiter fliessen und man diese halbieren kann. Die parasitären Bauteile der Spule (Lr und Rw) werden zusammengefasst, das heisst sie werden beide halbiert. Einer der beiden Y-Kondensatoren fällt weg. Dabei spielt es keine Rolle welcher, da die beiden immer gleichgross sind. Deshalb muss man annehmen, dass die parasitären Werte davon gleichgross sind. In der Software kann man deshalb auch nur einen Y-Kondensatorwert verstellen. Die vereinfachte Schaltung, welche zur Berechnung verwendet wurde, ist in Abbildung 16: Vereinfachte DM Schaltung sichtbar.



Abbildung 16: Vereinfachte DM Schaltung

Die Validierung der Umformungen wurde mit zwei Simulationen in «MPLAB» bewerkstelligt. Im folgenden Diagramm (Abbildung 17: Vergleich Simulationen) liegen die beiden Ergebnisse der Simulationen übereinander. Darin ist sichtbar, dass die Vereinfachung zum selben Ergebnis führt wie die Grundschaltung. Die beiden Schaltungen dazu sind im Anhang hinterlegt.



Abbildung 17: Vergleich Simulationen

## Vorgehensweise Berechnung Einfügungsverluste

Wir befassen uns einfachheitshalber mit der CM Schaltung, um das Prinzip der Berechnungen zu erklären. Bei der DM Schaltung wird wiederum genau die gleiche Vorgehensweise angewendet. Jedoch werden die Zweitore anders aufgeteilt. In Abbildung 18: Schaltung CM mit Einteilung in Zweitore wurde die CM Schaltung in Zweitore unterteilt und beschriftet.

Z1

Z2

Z3



Abbildung 18: Schaltung CM mit Einteilung in Zweitore

Zuerst wird die Gesamtimpedanz (Z1), sowie (Z2) und (Z3) bestimmt.

Wobei ist und später die Impedanzen für ein Frequenzspektrum von 50Hz bis 30MHz berechnet werden.

Nachdem die Impedanzen in Zweitore zusammengefasst wurden, kann man diese in Kettenmatrizen umwandeln. Dies funktioniert für die unterschiedlichen Zweitore folgendermassen:





Wir berechnen nun die Kettenmatrix (A1) für die Längsimpedanz (Z1) sowie (A2) für die Querimpedanz (Z2) und (A3) für (Z3).

Für den S21-Parameter benötigen wir die Gesamtmatrix, diese berechnet man mit Hilfe der Kettenschaltung, in dem man die drei Matrizen miteinander multipliziert. Dabei ist es wichtig, dass zuerst A1 mit A2 multipliziert wird und danach mit A3. [2]

## S-Parameter

Sobald man die Kettenmatrix für ein Zwei-Tor berechnet hat, lassen sich daraus die S-Parameter berechnen. Deshalb ist es der einfachste Weg, die Einfügungsverluste mit Hilfe dieser Parameter zu bestimmen. Im folgenden Zweitor sind die Wellengrössen zur Berechnung der S-Parameter angegeben.



Wobei und die Amplituden der einfliessenden Wellen sind.

Die Grössen und sind hingegen die Amplituden der reflektierten Wellen.

Abbildung 19: Zwei Tor mit Wellengrössen

Daraus lässt sich die Streumatrix S wie folgt berechnen:

Die einzelnen Parameter haben dabei folgende Bedeutung [3]:

: Eingangsreflexionsfaktor

: Rückwärtstransmissionsfaktor

: Vorwärtstransmissionsfaktor

: Ausgangsreflexionsfaktor

Wir benötigen den Rückwärtstransmissionsfaktor. Dieser benötigt die Bezugswiderstände (R1 und R2) , welche bei uns am Eingange, sowie am der Last 50 Ω betragen. Daraus berechnen wir den Streuparameter folgendermassen:

: Wert der Gesamtmatrix an bestimmter Position.

: Eingangswiderstand

: Lastwiderstand

Damit wird schlussendlich die Einfügungsdämpfung a in dB berechnet:

## Realisierung mit Matlab

Bevor die Berechnungen in Java implementiert wurden, sind die Einfügungsverluste in Matlab berechnet worden. Dies wurde gemäss der Vorgehensweise in 4.3 bewerkstelligt. [4] [5]



Abbildung 20: Vergleich Berechnung / Simulation

Abbildung 21: Insertion loss Kurve aus MATLAB

Der Vergleich der beiden Kurven zeigt, dass die Berechnungen mit der Simulation übereinstimmten. Der Matlab Code für die Berechnungen ist im Anhang vorhanden.

# Validierung

## Berechnungen

// TODO was kommt hier? Vergleich Matlab Java?

## Software

### Eingabe

### Benutzerfreundlichkeit

# Schlusswort

*Dem Auftraggeber steht nach erfolgreichem Projektabschluss die Software ”emj-dj“ voll funktionsfähig zur Verfügung. Alle Pflichtziele sowie ein Wunschziel, das Speichern und Laden der Eingabewerte, konnten gemäss Pflichtenheft umgesetzt werden. Die CM und DM Graphen wurden auf Matlab richtig berechnet und ins Java Programm implementiert. Dabei kann man durch das Verstellen der Bauteilwerte dessen Einfluss im Diagramm beobachten und vorhersagen können. Auch können eingegebene Werte abgespeichert werden und für die spätere Verwendung wieder ins Programm geladen werden.*

*//Die Problematik*

*Die Problematiken konnten alle bewältigt werden und das Programm wurde fristgerecht geliefert. In weiteren Arbeitsprozessen könnten weitere Wunschziele umgesetzt werden.*

*Ebenfalls kann man das Tool noch beliebig optimieren*

*// Beispiele was man noch erweitern könnte.*

Was wurde im Projekt erreicht? Was wurde übertroffen?

Wo besteht Verbesserungsbedarf? Was (und warum) konnte nicht realisiert werden

Optimierungs/Weiterentwicklungsmöglichkeiten

# Quellenverzeicnis

|  |  |
| --- | --- |
| [1] | Schaffner, «Single-Stage Filters FN 2020,» Schaffner Group, Luterbach, 2018. |
| [2] | P. Niklaus, «Zweitore (2-Tore),» FHNW, Windisch, 2018. |
| [3] | A. Schaum, «Zweitore - Zusammenfassung,» Lehrstuhl für Regelungstechnik, Universität zu Kiel, 2017. |
| [4] | A. Maier und S. Rupp, «Hochfrequenztechnik, Teil 2 - Anwendungen,» DHBW, Stuttgart, 2016. |
| [5] | B. Mößlang, «EMC Filter Insertion Loss Simulation,» OMICRON Lab, Reutlingen, 2016. |

# Anhang

//TODO

Schaltungssimulationen

Matlabcode CM / DM