Fachhochschule Aachen

Campus Jülich

Fachbereich: Medizintechnik und Technomathematik
Studiengang: Angewandte Mathematik und Informatik

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Entwicklung einer Softwarebibliothek für die Bestimmung des Wassergehalts von Pflanzenteilen mithilfe eines elektromagnetischen Hohlraumresonators

Bachelorarbeit von Lucas Gebhart
Matrikelnummer: 3279036

Jülich, den 14. September 2023

Angefertigt am

Institut für Bio- und Geowissenschaften,
Institutsbereich Pflanzenwissenschaften,
Forschungszentrum Jülich GmbH

Diese Arbeit wurde betreut von:

1. Prüfer: Prof. Dr.-Ing. Andreas Terstegge
2. Prüfer: Andreas Fischbach M.Sc.

Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und dabei keine anderen als die angegebenen Hilfsmittel benutzt habe. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher weder gesamt noch in Teilen einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Jülich, den 15.09.2023

.....
(Lucas Gebhart)

Abstract

Im Institut für Bio- und Geowissenschaften im Fachbereich der Pflanzenwissenschaften (IBG-2) wird an Pflanzen geforscht, um Konzepte für die Intensivierung und Nachhaltigkeit der Pflanzenproduktion zu entwickeln. Dort werden Hohlraumresonatoren zusammen mit Vektor-Netzwerk-Analysatoren (VNA) für die Messung des Wassergehalts von Pflanzenteilen verwendet. Bei den Resonatoren handelt es sich um ein Stück Metall mit einer großen Öffnung in der Mitte und zwei Antennen innerhalb. Diese Antennen könnten über Koaxialkabel an einen VNA angeschlossen werden. Der VNA ist in der Lage elektromagnetische Wellen für bestimmte Frequenzen zu erzeugen und auch wieder zu messen. Über die eine Antenne werden dann elektromagnetische Wellen in den Resonator freigegeben und die andere misst diese wieder. Der Resonator ist in der Lage einen Teil der elektrischen Energie der elektromagnetischen Wellen zu speichern und ein anderer Teil geht verloren. Diese Fähigkeit ist durch verschiedene Eigenschaften des Resonators bedingt. Wenn nun ein Objekt den Resonator betritt und auf das elektrische Feld innerhalb trifft, dann werden diese Eigenschaften verändert. Diese Veränderungen können in Form der gemessenen Werte der zweiten Antenne wahrgenommen werden. Für die hier verwendeten Resonatoren wird mit elektromagnetischen Wellen im Bereich der Mikrowellen gearbeitet. In diesem Bereich sind die zuvor erläuterten Veränderungen vor allem auf Wasser innerhalb des Objektes zurückzuführen, welches den Resonator betreten hat. Aus diesem Grund ist es möglich, mithilfe solcher Messwerte auf den Wassergehalt des Objektes zu schließen.

Im Institut werden bereits Resonatoren für eine solche Messung der Pflanzenteile verwendet. Zudem wurde ein neuer Messaufbau angefertigt, welcher eine Messung an einem Pflanzensamen durchführen soll. Um die Verwendung von Resonatoren zu vereinheitlichen und um ein Programm für die Messung eines Pflanzensamens bereit zu stellen, wurde im Rahmen dieser Arbeit eine Software-Bibliothek entwickelt. Diese ist in verschiedene Module aufgeteilt: Die Kommunikation mit dem VNA, die Durchführung einer Messung, die Auswertung der Messdaten und die visuelle Überprüfung der Messdaten. Für die Erfüllung der Ziele der Bibliothek wurden in der Entwicklung sowohl die objektorientierte, als auch die funktionale Programmierung genutzt. Dies sorgt dafür, dass die Anforderungen an die Bibliothek auf der Basis von Prinzipien guter Software erfüllt werden.

Die entwickelte Bibliothek ist in der Lage die Messung an einem Pflanzensamen durchzuführen und auszuwerten. An den Visualisierungen der Ergebnisse ist zu erkennen, dass

die Eigenschaften der dahinter liegenden Theorie von der Messung und der Auswertung erfasst werden. Zudem wurden Probleme des Messaufbaus aufgedeckt, welche die Genauigkeit der Endresultate beeinflussen. Im Anschluss an die Arbeit werden diese Probleme behoben, um eine genaue Approximationen des Wassergehalts des gemessenen Samen mithilfe der ausgewerteten Daten zu bestimmen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Im Institut für Bio- und Geowissenschaften im Fachbereich der Pflanzenwissenschaften (IBG-2) werden Konzepte entwickelt, um die Pflanzenproduktion für die Bioöknomie zu intensivieren und nachhaltiger zu gestalten [1]. Für die Forschung an solchen Konzepten werden viele verschiedene Ansätze verfolgt. Die Untersuchung des Phänotyps einer Pflanze während ihres Wachstums ist einer davon. Bei der Analyse dessen sind viele verschiedene Faktoren von Bedeutung. Der Wassergehalt ist einer davon. Dieser ist dabei sowohl bei der vollständigen Betrachtung einer bereits wachsenden Pflanze, als auch bei einem Pflanzensamen, welcher bald ausgesät wird, oder einem einzelnen Teil der Pflanze von Bedeutung. Die Bestimmung des Wassergehalts im Zuge der Pflanzenphänotypisierung liefert die meisten Informationen, wenn dieser Wert nicht nur einmalig bestimmt wird, sondern mehrfach über einen längeren Zeitraum, während dem die Pflanze wächst. Dies ist nur möglich, wenn die Methode zur Bestimmung der Wassermenge nicht invasiv durchgeführt wird. Das bedeutet, dass die Messung rein äußerlich am Pflanzenteil durchgeführt wird, sodass es nicht beschädigt wird. Um dies zu erreichen, werden am IBG-2 Messungen mit Hohlraumresonatoren durchgeführt.

Die verwendeten Hohlraumresonatoren sind von oben betrachtet rund und besitzen eine große Öffnung in der Mitte. Sie werden zusammen mit einem Vektor-Netzwerk-Analysator verwendet, welcher in der Lage ist elektromagnetische Signale zu erzeugen. Diese werden in den Resonator geleitet und innerhalb von diesem bildet sich dadurch ein elektromagnetisches Feld. Wenn nun ein Pflanzenteil einen solchen Resonator betritt, verändert es das Feld und die Veränderungen werden durch den VNA gemessen. Die Auswertung dieser Veränderungen lässt auf den Wassergehalt des Pflanzenteils schließen.

Innerhalb des Instituts werden Resonatoren in verschiedenen Größen für verschiedene Messobjekte verwendet. Abbildung 1.1 zeigt beispielsweise einen großen Resonator, welcher für die Messung des Wassergehalts von ganzen Pflanzen benutzt wird [2]. Für die Durchführung und Auswertung der Messungen wird kein einheitliches Programm benutzt, sondern die verwendeten Programme wurden speziell für den konkreten Messaufbau angefertigt. Sowohl die Durchführung als auch die Auswertung der Messungen überschneiden sich in einigen Punkten bei den verschiedenen Messaufbauten mit verschiedenen Resonatoren. Allerdings sind auch einige Teile dessen spezifisch von dem konkreten Messaufbau abhängig. Die Entwicklung eines universell verwendbaren Programms ist demnach nicht elementar. Allerdings würde die Verwendung eines solchen Programms die Nutzung der Resonatoren mehr vereinheitlichen und die Anwendung

von neuen vereinfachen.

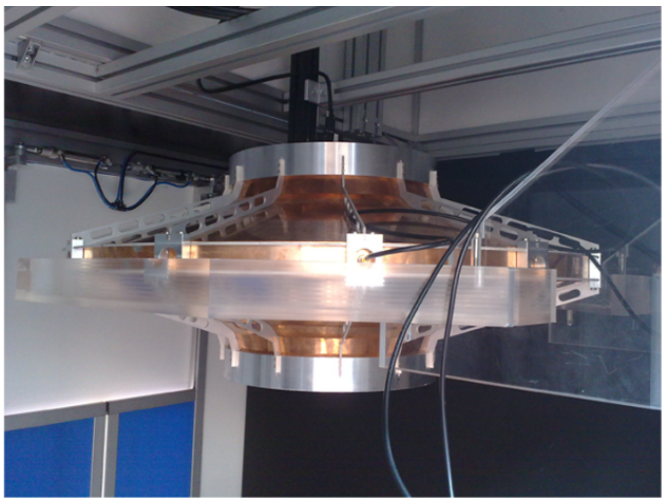


Abbildung 1.1: Resonator für die Messung von Pflanzen

Des weiteren wurde im IBG-2 ein Resonator entwickelt, welcher für die Messung des Wassergehalt eines Pflanzensamens verwendet werden kann. Es wurde in der Vergangenheit bereits erfolgreich überprüft, ob eine solche Messung an einem Pflanzensamen möglich ist.

Ziel der Arbeit ist die Entwicklung einer Softwarebibliothek für die Messung des Wassergehalts von Pflanzenteilen mittels eines Hohlraumresonators. Die Bibliothek soll primär die gesamte Durchführung und Auswertung der Messung eines Pflanzensamens enthalten. Dabei sollen die verwendeten Konzepte der Pflanzenphänotypisierierung für die Bestimmung des Wassergehalts eines Pflanzenteils konkret für einen Pflanzensamen umgesetzt werden [2]. Die zu entwickelnde Bibliothek soll dabei aber möglichst offen gehalten werden, um die Verwendung der Bibliothek auch für andere Resonatoren zu ermöglichen, da, wie zuvor erläutert, einige Schritte des Prozesses existieren, welche zwischen verschiedenen Messaufbauten identisch sind. Sie soll die Verwendung von Resonatoren in Bezug auf das zugehörige Programm vereinfachen.

In dieser Arbeit wird die Entwicklung einer solchen Bibliothek vorgestellt, indem zunächst der theoretische Hintergrund besprochen wird, auf dem die in der Bibliothek behandelte Thematik basiert. Darauf werden die verwendeten Konzepte besprochen, welche die Ziele der Bibliothek umsetzen sollen. Die folgende Umsetzung stellt dann die Implementation der Theorie auf Basis der zuvor besprochenen Konzepte dar. Zuletzt werden die Resultate in Form eines Testversuches vorgestellt und auf ihre Gültigkeit überprüft.

2 Theoretischer Hintergrund

2.1 Pflanzen als Dielektrikum

Um mithilfe eines Hohlraumresonators den Wassergehalt von Pflanzenteilen zu bestimmen, werden deren dielektrische Eigenschaften ausgenutzt [3]. Als ein Dielektrikum wird ein Material bezeichnet, welches Elektrizität kaum bis gar nicht leitet. Eine physikalische Eigenschaft von Dielektrika ist die Permittivität, welche in dem Kontext der Untersuchungen von besonderer Bedeutung ist. Sie ist zusammengesetzt aus der elektrischen Permittivität des Vakuums ε_0 und der relativen Permittivität des jeweiligen Materials ε_r. Diese ist messbar und bezieht sich auf die absolute elektrische Permittivität ε. Sie betrachtet, inwiefern ein Material die Energie eines externen elektrischen Feldes speichern kann [4]. Ein elektrisches Feld stellt eine Region um eine elektrische Ladung dar, welche eine Kraft auf andere Teilchen auswirkt [5, S.273-275].

Bei der relativen Permittivität ε_r eines Materials handelt es sich um eine komplexe Zahl. Der Realteil dieser Zahl ε_r' wird auch als dielektrische Konstante bezeichnet und repräsentiert die Fähigkeit, wie viel Energie von einem externen elektrischen Feld aufgenommen wird, während der Imaginärteil ε_r" darstellt, wie viel Energie in Form von thermischer Energie verloren geht. Beide Teile dieser Konstante werden durch verschiedene Faktoren beeinflusst.

Es gibt verschiedene dielektrische Mechanismen, welche einen Einfluss auf die Permittivität eines Materials haben, beispielsweise die Ionenleitfähigkeit oder die atomare Polarisation. Jede dieser Mechanismen besitzen eine Grenzfrequenz, ab der sie nicht mehr wirken. Stattdessen treten dann an ihrer Stelle andere Vorgänge in Erscheinung. Ein Resonator ist ein schwingfähiges System, welches speziell konzipiert wurde, um für einen bestimmten Frequenzbereich von Wellen einen Resonanzeffekt aufzuweisen. Der Resonanzeffekt beschreibt das verstärkte Mitschwingen eines schwingfähigen Systems, wenn die Frequenz der auf ihn einwirkenden Welle nahe der spezifischen Resonanzfrequenz des Systems liegt. Sie können mit verschiedenen Arten von Wellen wie beispielsweise Schallwellen oder optischen Wellen und in verschiedenen Frequenzbereichen verwendet werden. Je nach Resonator werden sie für unterschiedlichste Aufgaben verwendet. Die für diese Arbeitet relevanten Hohlraumresonatoren arbeiten mit elektromagnetischen Wellen im Bereich der Mikrowellen und erzeugen innerhalb von ihnen ein elektrisches

Feld. Wie zuvor erläutert ist der Einfluss von dielektrischen Mechanismen auf die Permittivität eines Materials abhängig von der Frequenz des Feldes. Abbildung 2.1 zeigt die verschiedenen Mechanismen mit ihren zugehörigen Frequenzen. Die x-Achse der Graphik stellt die Frequenz der elektromagnetischen Welle dar und die beiden Graphen zeigen jeweils die Veränderung des Real- und des Imaginärteils der relativen Permittivität. Innerhalb der Graphik sind verschiedene solcher Mechanismen auf der Höhe des jeweiligen Frequenzbereichs eingezeichnet, innerhalb dessen sie am stärksten wirken. Für den hier relevanten Bereich der Mikrowellen besitzt die Orientierungspolarisation den größten Einfluss [4].

In Abbildung 2.1 ist zu erkennen, dass die Einflussnahme der Orientierungspolarisation bereits in niedrigeren Frequenzbereichen beginnt und dann im Mikrowellenbereich weiterhin relevant bleibt. Diese funktioniert nach dem folgendem Prinzip: Es gibt Moleküle, bei denen die Schwerpunkte der jeweiligen positiven und negativen Ladungen voneinander abgegrenzt sind. Diese Moleküle werden Dipole genannt und ein Beispiel dafür sind Wassermoleküle. Wie diese Schwerpunkte orientiert sind ist zufällig. Wenn jedoch ein externes elektrisches Feld auf ein Dipol einwirkt, dann richten sich die Schwerpunkte nach dem elektrischen Feld aus. Dieser Vorgang wird als Orientierungspolarisation bezeichnet. Dies führt dazu, dass im dielektrischen Material, welches solche Dipole enthält, ein zusätzliches elektrisches Feld entsteht, welches dem externen entgegen gesetzt ist. Dieser Vorgang stellt eine Speicherung von Energie dar. Die durch das Ausrichten der Moleküle entstandene Drehung führt jedoch auch zu Reibung. Dies hat einen Energieverlust in Form von einer Abgabe von thermischer Energie zufolge. Somit hat der Vorgang der Orientierungspolarisation Einfluss sowohl auf den Real- als auch auf den Imaginärteil der dielektrischen Konstante [4].

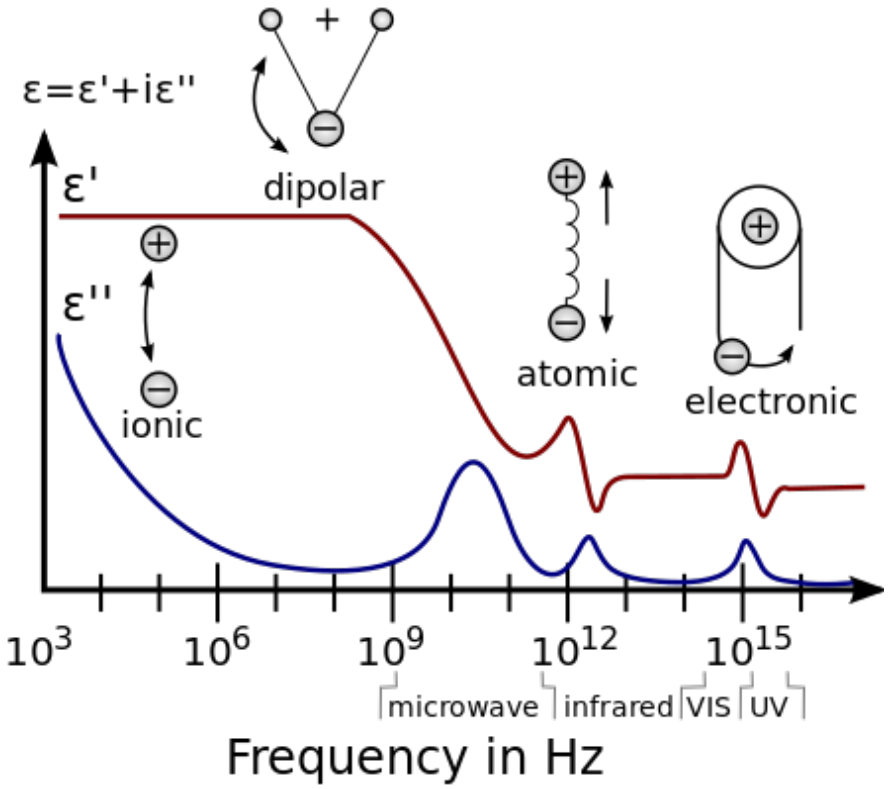


Abbildung 2.1: Einfluss von dielektrischen Mechanismen in Abhängigkeit der Frequenz [6]

Verschiedene Pflanzenteile stellen Dielektrika dar und somit sind diese Eigenschaften und Vorgänge auch bei ihnen vorhanden. Zudem bestehen sie zum Großteil aus Wasser. Da im gewählten Frequenzbereich des Resonators vor allem die Orientierungspolarisation eine bedeutende Rolle spielt und diese bei Wassermolekülen besonders stark ausgeprägt ist, kann man diese Eigenschaften nutzen, um auf den Wassergehalt der jeweiligen Teile zu schließen.

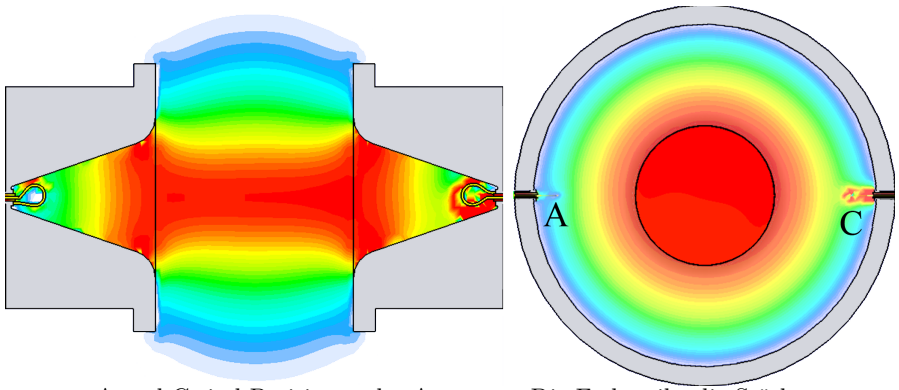
2.2 Vektor-Netzwerkanalysator

Ein Vektor-Netzwerkanalysator (VNA) wird verwendet, um die Messungen mit dem Resonator durchzuführen, welche benötigt werden, um im Anschluss den Wassergehalt des gemessenen Objektes zu bestimmen. Der VNA ist gleichzeitig für die Erzeugung eines Testsignals, als auch wieder für die Erfassung des modifizierten Testsignals zuständig. In dem betrachteten Anwendungsfall erzeugt er ein sinusförmiges Testsignal in Form einer elektromagnetischen Welle. Bei der Auswertung des wieder erhaltenen Signals wird das Übertragungsverhalten betrachtet. Während der Übertragung des Signals verändert

sich die Amplitude und es kommt zu einer Phasenverschiebung. Der VNA berechnet die Veränderungen in Form von sogenannten Streuparametern (S-Parameter). Es existieren die S-Parameter s_{11}, s_{12}, s_{21}, s_{22}. Diese behandeln die Reflexion und Transmission des ausgegeben Signals. Um das Signal auszusenden und wieder zu empfangen, werden zwei verschiedene Ports am Gerät verwendet. Im Kontext dieser Arbeit ist nur der s_{21}-Parameter von Bedeutung. Dieser beinhaltet Informationen über die Transmission des Signals vom Signalursprung zum erneuten Empfangen des Signals. Es wird demnach betrachtet, wie das Signal im Vergleich zur Aussendung wieder angekommen ist. Die S-Parameter werden in Abhängigkeit von der Frequenz des ausgesendeten Signals bestimmt [7].

2.3 Messaufbau

Hauptsächlich besteht der Messaufbau aus einem VNA, einem Hohlraumresonator und zwei Koaxialkabeln. Von oben betrachtet ist der Resonator rund mit einer größeren Öffnung in der Mitte. Im Querschnitt (Abb. 2.2) ist zu erkennen, dass die Öffnung noch in die Ränder des Resonators weiter verläuft. Zudem sind im Resonator zwei gegenüberliegende Antennen verbaut. Diese befinden sich in der Grafik bei A und C (Abb. 2.2). Diese sind über Stecker an der Außenseite des Resonators mit jeweils einem Koaxialkabel verbunden. Die beiden Koaxialkabel sind an verschiedene Ports des VNAs angeschlossen.



A und C sind Positionen der Antennen. Die Farbe gibt die Stärke des Feldes an von schwach (blau) bis stark (rot).

Abbildung 2.2: Vertikaler und horizontaler Querschnitt des Resonators [2]

Der VNA erzeugt ein Signal mit einer spezifischen Frequenz, welches dann über ein Koaxialkabel übertragen wird. Das Signal erreicht dann die Antenne im Inneren des Resonators, welche das Signal in den Innenraum des Resonators ausstrahlt. Dadurch entsteht innerhalb des Hohlraumresonators ein elektromagnetisches Feld. Ein solches Feld besteht aus einem elektrischen und einem magnetischen Feld. Die Form des Resonators wurde

speziell konzipiert, um eine bestimmte Verteilung der Felder zu erreichen. Dadurch bildet sich das Magnetfeld am inneren Rand des Resonators. Das elektrische Feld befindet sich hingegen in der Mitte des Resonators. Die spezifische Form sorgt dafür, dass sich das elektrische Feld auf horizontaler Ebene zwischen den Öffnungen homogen verteilt. Die zweite Antenne empfängt einen Teil der elektromagnetischen Wellen, welche durch den Resonator wandern. Das empfangene Signal wird über das angeschlossene Koaxialkabel an den VNA weitergeleitet, welcher es auswertet und das Transmissionsverhalten in der Form des s_{21}-Parameters bestimmt [2].

2.4 Abläufe innerhalb des Resonators

Ein Teil der elektrischen Energie, welche in Form der elektromagnetischen Wellen durch den Resonator traversieren, wird von dem Resonator gespeichert, während ein anderer Teil verloren geht. Wie viel Energie der Resonator abgibt und wie viel er speichert beeinflusst das Signal, welches die Antenne empfängt. Der Resonator besitzt zwei bestimmbare Kenngrößen, welche durch diesen Vorgang beeinflusst werden. Die erste ist die Resonanzfrequenz. Diese stellt eine bestimmte Frequenz dar, für die die Amplitude der zugehörigen Welle verstärkt wird. Desto näher die Frequenz der in den Resonator eintretenden elektromagnetischen Welle der Resonanzfrequenz ist, desto größer fällt die Amplitude dieser aus. Dieser Effekt sorgt für eine erhöhte Speicherung von Energie im Resonator. Die andere Kerngröße ist der Gütefaktor. Dieser stellt das Verhältnis zwischen der durch den Resonator gespeicherten Energie und der verlorenen Energie dar [3].

Inwiefern die elektromagnetischen Wellen den Resonator durchqueren und wie viel Energie gespeichert wird, steht demnach in Bezug zu dem Gütefaktor und der Resonanzfrequenz des Resonators. Wenn sich kein Objekt innerhalb des Resonators befindet stellt die Luft ein Dielektrikum dar, welche mit ihren dielektrischen Eigenschaften zudem die Energiespeicherung beeinflusst. Wenn nun ein zu untersuchendes Material das Innere des Resonators befüllt, wird dies verändert. Wie die Wellen dann den Resonator traversieren und wie viel Energie aufgenommen wird und verloren geht wird nun verändert. Diese Veränderungen sind abhängig von der relativen Permittivität des Materials [4]. Das Material speichert nämlich einen Teil der Energie des externen elektrischen Feldes. Dies ist bedingt durch die in 2.4 erläuterten Mechanismen. Dadurch wird die gesamte Menge an Energie, die innerhalb des Resonators gespeichert wird, verändert. Das spiegelt sich, wie zuvor erläutert, im durch die Antenne gemessenen Signal wider. Durch die Veränderungen werden sowohl die Resonanzfrequenz als auch der Gütefaktor des Resonators verändert [2]. Auf die veränderten Werte lässt sich durch das Empfangen des veränderten Signals schließen.

Da die Änderungen, wie in 2.1 erläutert, hauptsächlich durch die Wassermoleküle innerhalb des Pflanzenteils verursacht werden, lässt sich anhand dieser auf den Wassergehalt

des Objekts zurückschließen. Wie genau dies möglich ist, wird im nächsten Unterkapitel erläutert.

2.5 Verarbeitung der Messdaten

Die Messung mit einem Hohlraumresonator wird in einem Frequenzbereich durchgeführt, in dem sich die Resonanzfrequenz des Resonators befindet. Ein vollständiger Messvorgang, bei dem ein Material einmal komplett gemessen wird, besteht aus mehreren Einzelmessungen. Eine Einzelmessung wird dabei mehrmals für eine ausgewählte Anzahl an Messpunkten durchgeführt. Für jeden Messpunkt wird ein s_{21}-Wert gemessen. Die insgesamt bestimmte Anzahl von Messwerten ist demnach von der Anzahl der Einzelmessungen und der Messpunkte abhängig. Für n Einzelmessungen mit jeweils m Messpunkten beträgt die Anzahl von gemessen Werten $n \cdot m$. Im Zuge einer Einzelmessung wird für die Messpunkte mit verschiedenen Frequenzen gemessen. Für welche Frequenzwerte genau ein Messwert bestimmt wird, ist abhängig von der gewählten Anzahl an Messpunkten und dem Frequenzbereich. Wenn beispielsweise für eine Messung ein Frequenzbereich von 2.5 GHz bis 3 GHz ausgewählt wird und zehn Einzelmessungen mit jeweils sechs Messpunkten durchgeführt werden sollen, dann wird während einer Einzelmessung das Verhalten des Signals für die Frequenzen 2.5 GHz, 2.6 GHz, 2.7 GHz, 2.8 GHz, 2.9 GHz und 3 GHz bestimmt. Abbildung 2.3 zeigt den Aufbau eines einzelnen Messvorgangs für dieses Beispiel. Insgesamt werden 60 Messwerte bestimmt. Für jede der Frequenzen liegen, aufgrund der zehn Einzelmessungen, dann zehn s_{21}-Werte vor.

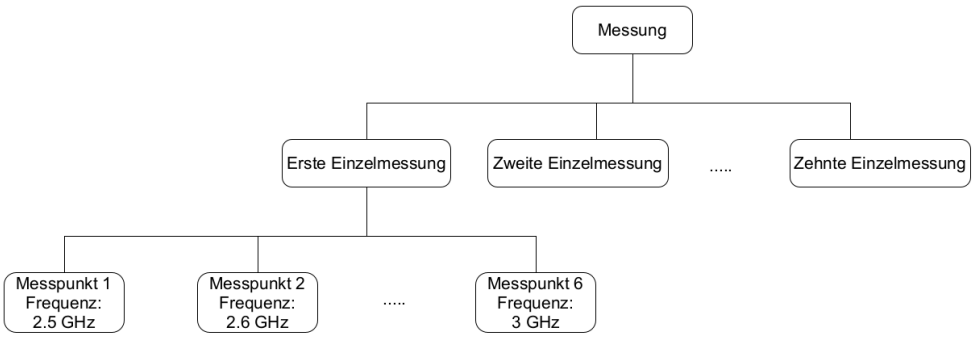


Abbildung 2.3: Aufbau einer einzelnen Messung an einem Beispiel mit zehn Einzelmessungen mit jeweils sechs Messpunkten

Die erhaltenen s_{21}-Werte sind komplexe Zahlen und beinhalten Informationen über das empfangene Signal. Für die hier relevante Betrachtung ist nur die Amplitude des ur-

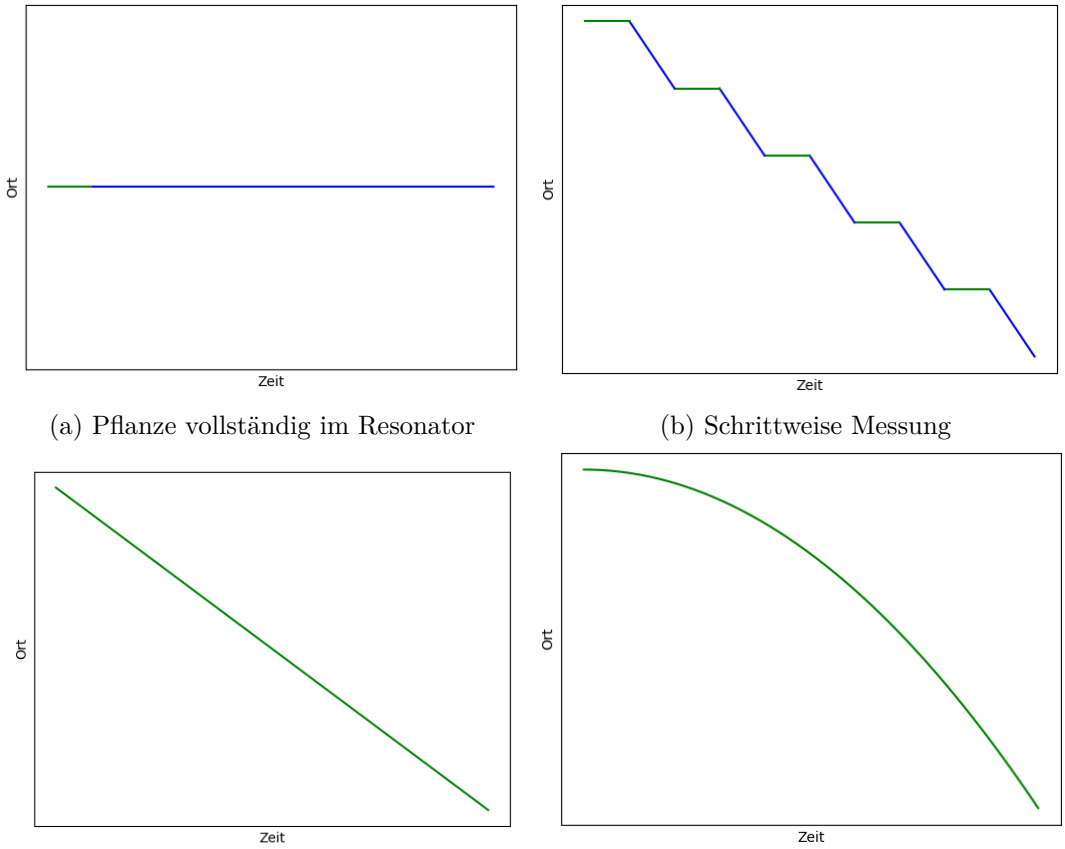
sprünglich empfangenen Signals von Bedeutung. Die logarithmische Amplitude lässt sich mit der folgenden Formel aus dem s_{21}-Parameter bestimmen [8]:

$$A = 20log_{10}(\sqrt{Realteil^2 + Imaginärteil^2}) \tag{2.1}$$

Es existieren mehrere Möglichkeiten eine solche Messung an einem Pflanzenteil durchzuführen. Diese sind abhängig von der Art des Pflanzenteils. Die einfachste ist eine Pflanze vollständig in einem Resonator zu platzieren, sodass nur eine Einzelmessung nötig ist, um die gesamte Pflanze zu erfassen (Abb. 2.4a) [8]. Bei Messungen an einer Pflanze mit einem bewegbaren Resonator, kann die Messung schrittweise durchgeführt werden (Abb. 2.4b). Das bedeutet, dass die Einzelmessungen jeweils an einer bestimmten Position an der Pflanze durchgeführt werden. Der Resonator wird dabei nach jeder Einzelmessung um eine vorgegebene Distanz vertikal bewegt und erst wenn die Bewegung abgeschlossen ist, wird die nächste Einzelmessung ausgeführt. Bei den entstandenen Messdaten ist dann bei der Auswertung bekannt, welche Einzelmessung an welcher Position an der Pflanze vollzogen wurde. Somit wurden die Signale für die verschiedenen Frequenzen während einer Einzelmessung an der selben Position bestimmt.

Der Resonator kann sich während einer Messung allerdings auch konstant bewegen, oder die Pflanze wird mit konstanter Geschwindigkeit durch den Resonator bewegt (Abb. 2.4c). Dadurch wird Zeit gespart, allerdings ist nicht direkt ersichtlich, welche der einzelnen Frequenzmessungen an welcher Position durchgeführt wurde. Die Messung erfolgt dann kontinuierlich. Des weiteren wurden die gemessenen Signale während einer Einzelmessung nicht an der gleichen Postion bestimmt, da der Resonator in konstanter Bewegung ist.

Der gleiche Fall liegt vor, wenn der Resonator nicht beweglich ist und das zu messende Objekt wie beispielsweise ein Pflanzensamen durch den Resonator fällt (Abb. 2.4d). In diesem Fall liegt zudem noch eine sich beschleunigende Bewegung vor, da das Objekt im freien Fall schneller wird. In beiden Fällen befinden sich die Messpunkte einer Einzelmessung an verschiedenen Positionen. Um dann die Messpunkte an einer gleichen Stelle zu erhalten, muss eine Neuberechnung der Daten vollzogen werden [9].



(a) Pflanze vollständig im Resonator

(b) Schrittweise Messung

(c) Konstante Bewegung

(d) Fall durch Resonator

grüne Abschnitte signalisieren Einzelmessungen, blaue Abschnitte keine Einzelmessungen

Abbildung 2.4: Messverhalten zwischen Ort und Zeit bei kontinuierlichen Messungen

Um die gemessenen Werte auswerten zu können, müssen sie in einer bestimmten Form vorliegen. In einem gewählten Intervall von Positionen an dem Messobjekt mit einer gleichbleibenden Schrittweite muss für jede dieser Positionen ein s_{21}-Wert für jede der betrachteten Frequenzen vorliegen. Wenn die Messung, wie zuvor erläutert, schrittweise vollzogen wird, liegen die Messdaten bereits in dieser Form vor. Wird die Messung allerdings kontinuierlich durchgeführt, befindet sich jeder einzelne Messpunkt der Einzelmessungen an verschiedenen Positionen an dem gemessen Objekt. Aus diesem Grund müssen die Daten in diesem Fall zuvor noch bearbeitet werden.

Zunächst müssen dabei für die jeweiligen Messpunkte die zugehörigen Positionen am Objekt vorliegen. Für eine konstante Bewegung des Resonators können diese mit externen Geräten zusätzlich gemessen werden oder mit der Formel für eine gleichförmige Bewegung $s = v \cdot t$ bestimmt werden. Im Falle des Falls durch den Resonator sind die

Positionen über die Formel des freien Falls $s = \frac{gt^2}{2}$ berechenbar. s ist der zurückgelegte Weg, v die Geschwindigkeit, g die Erdanziehungskraft und t stellt die Zeit dar. Wie zuvor erläutert gibt es in Abhängigkeit von der gewählten Messpunktanzahl und dem Frequenzbereich feste Frequenzwerte, für die in jeder einzelnen Messung ein s_{21}-Wert bestimmt wird. Demnach gibt es für eine einzelne Frequenz so viele s_{21}-Werte wie Einzelmessungen während einer Gesamtmessung durchgeführt wurden. Diese sind dabei mit der zugehörigen Position an dem Messobjekt verbunden. Sie können als Wertepaare betrachtet werden mit der Position als Eingabewert und dem s_{21}-Wert als Ausgabewert. Diese Paare werden für jede einzelne Frequenz benutzt, um für jeden einzigartigen Frequenzwert eine Funktion zu interpolieren. Dafür muss eine geeignete Interpolationsmethode gewählt werden. Mithilfe dieser Funktionen kann dann für ein gewähltes Intervall von Positionen mit fester Schrittweite ein Wert für jede der Messfrequenzen bestimmt werden. Die auf diese Weise bestimmten Werte liegen dann in der gleichen Form wie die einer inkrementell durchgeführten Messung vor [9].

Sei A_i die Amplitude der i-ten Mode, Q_i der Gütefaktor, f_{0i} die Resonanzfrequenz, φ_i die Phasenverschiebung, j die imaginäre Zahl $\sqrt{-1}$ und Γ_s sei der Einfluss des empfangenen Signales, welcher nicht auf das Feld innerhalb des Resonators zurückzuführen ist und somit Rauschen signalisiert [10]. Dann kann die Amplitude eines empfangenen Signals durch folgende Formel ausgedrückt werden [9]:

$$\Gamma(f) = \Gamma_s + \sum_{i}^{n} \frac{A_i e^{j\varphi_i}}{1 + 2jQ_i \frac{f - f_{0i}}{f_{0i}}} \tag{2.2}$$

Die Summe bezieht sich dabei auf unterschiedliche sogenannte Moden. Diese Schwingungsformen bezeichnen die Bedingungen, unter denen sich eine Welle während ihrer Ausbreitung in eine Welle verwandelt, welche sich nicht mehr weiter fortbewegt, sondern nur noch schwingt. Jede Mode hat ihre eigene Resonanzfrequenz, ihren eigenen Gütefaktor und ihre eigene Feldverteilung. Aufgrund ihrer endlichen Gütefaktorwerte sind die Moden über die Frequenz verteilt und beeinflussen sich gegenseitig im Gesamtspektrum. Es sei darauf hingewiesen, dass sich diese Arbeit für die spätere Implementierung nur auf die erste Mode des Resonators konzentriert, aber der Einfluss der zweiten Mode wird ebenfalls berücksichtigt. Eine tiefere Betrachtung dieses Themas ist für die betrachtete Thematik nicht von Bedeutung und kann bei Interesse in den Quellen [2] und [9] nachgelesen werden.

Die Formel besagt, dass die Amplitude des gemessenen Signals gleich der Summe der komplexen Amplituden der beteiligten Moden ist. Im Falle des s_{21}-Werts, bei dem $\Gamma(f)$ in logarithmischer Form betrachtet wird, lautet die Formel [9]:

$$S(f) = 20log_{10}(|\frac{\Gamma(f)}{\Gamma_0}|) \tag{2.3}$$

Wobei Γ_0 die Amplitude des ausgesendeten Signals darstellt.

Für jede Position liegen Messwerte für die ausgewählten Frequenzen vor. Mithilfe der Werte kann für jede Position eine optimale Lösung für die Parameter der Formel bestimmt werden. Mithilfe dieser berechneten Parameter für jede Position können die Daten weiter ausgewertet werden. Wie die Formel konkret für eine durchgeführte Gesamtmessung aussieht, ist von dieser abhängig.

Für die weiteren Berechnungen werden die bestimmten Parameter der ersten Mode verwendet. Mithilfe der zuvor bestimmten Parameter für jede Position können die folgenden Werte basierend auf den Beziehungen zwischen den Werten bestimmt werden [2]:

$$\Delta Q f_{rel}(z) = (\frac{f_0 Q_0}{Q_{0loaded}(z) f_{0loaded}(z)} - 1)\frac{f_0}{f_{0loaded}(z)} \tag{2.4}$$

$$\Delta f_{inv}^2(z) = f_{f0loaded}^{-2}(z) - f_0^{-2} \tag{2.5}$$

f_0 und Q_0 sind dabei die zugehörigen Werte für einen leeren Resonator und z stellt die Position an dem gemessen Objekt dar. Die Betrachtung der Integrale dieser Werte und dessen Zusammenhang zueinander stellen weitere benötigte Werte dar [2]:

$$Integral_{\Delta f_{inv}^2} = \frac{4\pi^2 L_0 E_0^2 \sqrt{\pi}\sigma}{U_0^2} \varepsilon_0 (\varepsilon'_{MO} - \varepsilon_{air}) V_{MO} \tag{2.6}$$

$$Integral_{\Delta Q f_{rel}} = \frac{2\pi R_0 E_0^2 f_0 \sqrt{\pi}\sigma}{U_0^2} \varepsilon_0 \frac{\varepsilon'_{MO}\varepsilon''_{MO}}{\varepsilon'_{MO} + \varepsilon''_{MO}} V_{MO} \tag{2.7}$$

$$\frac{Integral_{\Delta f_{inv}^2}}{Integral_{\Delta Q f_{rel}}} = \frac{1}{Q_0 f_0^2}(\varepsilon'_{MO} - \varepsilon_{air})(\frac{1}{\varepsilon'_{MO}} + \frac{1}{\varepsilon''_{MO}}) \tag{2.8}$$

E_0, L_0 und U_0 beziehen sich auf die Theorie hinter dem Resonator und sind hier nicht weiter relevant. Bei Interesse kann mehr dazu in der Quelle [2] nachgelesen werden. V_{MO} ist das Volumen des gemessenen Objektes. ε'_{MO} und ε''_{MO} stellen dabei den Real- und Imaginärteil der relativen Permittivität des zu untersuchenden Objektes dar. Die Werte stehen also im direkten Zusammenhang mit der Dielektrizitätskonstante des gemessenen Objektes und lassen somit auf den Wassergehalt des Objektes schließen . Dies ist über verschiedene Arten möglich. Beispielsweise kann $Integral_{\Delta f_{inv}^2}$ zusammen mit dem Volumen des gemessenen Objektes verwendet werden. Der Wert wird dann für mehrere Testobjekte bestimmt, welche im Anschluss getrocknet werden. Zudem wird sowohl vor als auch nach der Trocknung das Gewicht der Testobjekte bestimmt. Danach werden sie erneut gemessen. Mithilfe dieses Vergleichs und einer Miteinberechnung des Volumens der Objekte kann eine Kalibrierung erstellt werden, welche aus dem Integralwert von einem neuen Messobjekt den zugehörigen Wassergehalt approximiert. Solch eine Kalibrierung kann nach diesem Verfahren auch durch die Division der beiden Integrale erstellt werden. Für diesen Fall wäre das Volumen des Objektes nicht benötigt, da es durch die Division aus der Formel gekürzt wird.

3 Konzept

3.1 Anforderungen

Die zu entwickelnde Bibliothek soll in erster Linie die Durchführung einer Messung an einem Pflanzensamen ermöglichen. Der Aufbau enthält einen über einen seriellen Anschluss ansteuerbaren Mechanismus für die Fixierung von Samen oberhalb des Resonators. Mithilfe dessen kann der Samen dann fallen gelassen werden. Dies ist auch über Knöpfe am Mechanismus selbst steuerbar. Nach der Aktivierung kann der Samen per Hand platziert werden und dann automatisch fallen gelassen werden. Der Resonator ist bei diesem Aufbau mit zwei Antennen ausgestattet, welche über Koaxialkabel mit dem verwendeten VNA verbunden sind. Die Bibliothek soll für diesen spezifischen Messaufbau den kompletten Messvorgang und die Auswertung der entstandenen Daten ermöglichen. Zudem soll die Bibliothek aber auch modular aufgebaut sein, um sie später in Verbindung mit anderen Resonatoren und dessen variierenden Aufbauten verwenden zu können. Diese beiden Anforderungen stellen die Hauptanforderungen an das System, durch welche weitere induziert werden.

Der Fall-Mechanismus des Messaufbaus muss über die Bibliothek steuerbar sein. Wie zuvor erwähnt muss die Fixierung des Samens händisch erfolgen. Es muss dabei garantiert sein, dass dem Benutzer genug Zeit zur Verfügung steht, um den Samen zu befestigen, bevor er fallen gelassen wird. Des Weiteren muss beachtet werden, dass die Messung erst gestartet wird, wenn der Fall des Samens begonnen hat.

Es muss eine Kommunikation innerhalb der Bibliothek mit dem VNA erfolgen. Die Messung muss nämlich durch die Software gestartet werden und die gemessenen Daten müssen abgerufen werden. Zudem sollte die Kommunikation ohne große Verzögerung erfolgen, damit die Messung direkt, nachdem der Fall des Samens begonnen hat, gestartet wird. Des weiteren müssen Parameter für die Messung gesetzt werden können wie beispielsweise der Frequenzbereich, die Anzahl der Einzelmessungen oder die Art der Messung.

Die Auswertung der Messdaten muss in einzelnen trennbaren Schritten erfolgen. Zunächst muss die Bibliothek die komplette Auswertung der Messdaten enthalten. Die einzelnen Auswertungsschritte müssen unabhängig erfolgen, damit sie auch für andere

Resonatoren verwendet werden können. Die Auswertungsprozeduren können sich nämlich für verschiedene Resonatoren unterscheiden. Manche Teile der Prozedur verhalten sich anders oder fallen vollständig weg. Daher muss die Implementation der einzelnen Schritte unabhängig voneinander sein, damit sie universal einsetzbar sind. Die einzelnen Schritte müssen auf der Basis einer anderen Prozedur verwendet werden können, wenn beispielsweise ein Schritt bei einem anderen Aufbau anders ausfällt. Die Implementation eines neuen Schritts muss daher auch einfach in die Bibliothek einzufügen sein.

Zudem muss die Bibliothek die Funktionalität bieten die Messdaten speziell für den Messaufbau für die Messung eines Pflanzensamens auszuwerten. Dabei ist eine konkrete Implementierung der in 2.5 beschriebenen Prozedur benötigt. Dies beinhaltet die Berechnung der zugehörigen Postionen der Messdaten unter Beachtung des freien Falls des Samens. Zudem muss die konkrete Formel für die Amplitude des gemessenen Signals implementiert werden. Die für diesen Messaufbau spezifischen Schritte müssen mit den Implementierungen der allgemeinen Schritte kompatibel sein.

Es muss möglich sein, die während der Verarbeitung der Messdaten entstandenen Daten zu speichern und zu visualisieren um diese zu überprüfen und im Nachhinein zu betrachten. Dies ist wichtig um bei der Implementation von neuen Auswertungsschritten diese einfach auf Korrektheit zu überprüfen.

3.2 Einteilung in Module

Um die zu entwickelnde Bibliothek sinnvoll zu strukturieren, sollte sie in einzelne Module eingeteilt werden. Diese Einteilung soll gewährleisten, dass die Anforderungen an die Software erfüllt werden. Die Anforderungen selbst geben an, welche Module in der zu entwickelnden Bibliothek vorhanden sein müssen und worauf dabei zu achten ist. Aus den zuvor erläuterten Anforderungen ergeben sich folgende Module:

- Die Verwaltung der Verbindung und der Kommunikation mit dem VNA
- Die Durchführung des Messprozesses spezifisch für die Messung eines Pflanzensamens
- Der Auswertungsprozess der Messdaten für diese Messung mit der Implementation aller Schritte aus 2.5
- Die optionale Verwaltung der während der Auswertung angefallenen Daten mit der Möglichkeit zur Speicherung und Visualisierung dieser Daten

Für die Entwicklung der Struktur der Bibliothek auf der Basis dieser Anforderungen hilft die Einhaltung von anerkannten Designprinzipien, welche für die zu entwickelnde Software passend sind:

- Prinzip der losen Kopplung: Kopplung bezeichnet die Anzahl der Beziehungen zwischen den einzelnen Modulen eines Programms. Diese sollte möglichst gering ausfallen. Das vereinfacht die Struktur und eine spätere Anpassung der Module [11, S.130-132]. Dies ist hilfreich, da somit die späteren Erweiterungen der Bibliothek für andere Resonatoren einfacher hinzuzufügen sind. Zudem ist auch der Austausch von ganzen Modulen einfacher implementierbar, wenn beispielsweise für einen anderen Resonator ein anderer VNA verwendet werden muss.
- Prinzip der hohen Kohäsion: Dieses Prinzip bezieht sich auf das Innere eines Moduls. Kohäsion betrifft die Abhängigkeiten und Zusammenhänge zwischen den einzelnen Elementen eines Moduls. Diese sollte möglichst hoch ausfallen, damit die Module unabhängiger voneinander sind [11, S.133-134] . Die Beachtung dieses Prinzips erleichtert ebenfalls den Austausch eines Moduls für die Nutzung mit weiteren Resonatoren. Eine hohe Kohäsion und damit auch eine geringe Kopplung bietet sich zudem an, da die Aufgabenbereiche der zuvor aufgestellten Module gut voneinander abgrenzbar sind. Es hilft sie untereinander unabhängig zu gestalten und ermöglicht beispielsweise eine Auswertung von Messdaten ohne dass dabei die eigentliche Messung von der Bibliothek durchgeführt wurde.
- Separation-of-Concerns-Prinzip: Verschiedene Aufgabenbereiche sollen auch in dem Aufbau der Software voneinander getrennt sein. Die einzelnen Module sollen unabhängig sein und feste Aufgabenbereiche besitzen [11, S.137-139] . Die zuvor definierten Module der zu entwickelnden Bibliothek sollen voneinander abgegrenzt sein, um erneut die Modularität zu erhöhen. Dies verbessert die Fähigkeit der Software sie für verschiedene Aufgaben zu nutzen und nicht nur für die vollständige Durchführung einer Messung mit anschließender Auswertung.
- Prinzip des Entwurfs für Veränderung: Bei der Strukturierung eines Programms sollen vorhersehbare Veränderungen mit eingeplant werden, sodass ihre schließliche Implementation einfach ausfällt [11, S.135-137] . Eine Betrachtung der Anforderungen legt sehr nahe, dass die Beachtung dieses Prinzips sehr von Vorteil ist. Da die Bibliothek eventuell in Zukunft noch für andere Resonatoren benutzt werden soll, müssen die daraus resultierenden Erweiterungen einfach zu implementieren sein. Das bedeutet erneut, dass die Module voneinander unabhängig sein sollen und vor allem, dass die einzelnen Schritte während der Auswertung der Daten offen implementiert werden, sodass sie einfach mit neuen Auswertungsarten zusammen verwendet werden können.

Die Beachtung dieser Prinzipien für die zuvor erwähnten Module ermöglicht die Entwicklung einer Bibliothek, welches die zugehörigen Anforderungen erfüllt.

3.3 Programmierparadigmen

Programmierparadigmen stellen eine Sammlung von konzeptuellen Mustern dar, welche den Designprozess und auch die entstehende Struktur eines Programms bestimmen [12]. Zu den bekanntesten zählen die objektorientierte Programmierung und die funktionale Programmierung. Die objektorientierte bezieht sich auf Objekte, welche Daten und Methoden beinhalten und mittels dieser Methoden die Daten manipulieren. Funktionale Programmierung basiert hingegen auf Funktionen, welche zustandslos sind und zu einer Eingabe immer nur eine feste Ausgabe liefern, ohne dabei Nebeneffekte zu bewirken [13]. Diese beiden Paradigmen sind in ihrer Funktionsweise sehr verschieden und bieten bestimme Vorteile. Es existieren spezifische Programmiersprachen, welche nur eine der beiden Paradigmen unterstützen. Allerdings existieren auch Multiparadigmensprachen, welche die Verwendung von mehreren Paradigmen ermöglichen und unter anderem sowohl die objektorientierte, als auch die funktionale Programmierung unterstützen [14]. Zur Entwicklung der Bibliothek wird Python verwendet, welches eine solche Sprache darstellt. Aus diesem Grund ist es sinnvoll beide Paradigmen für die Entwicklung der Bibliothek zu betrachten, da beide für bestimmte Komponenten von Vorteil sind.

3.3.1 Objektorientierte Programmierung

Das Grundkonstrukt der objektorientierten Programmierung stellt das Objekt dar. Es ist ein Element, welches sowohl Daten speichert, als auch die zugehörigen Operationen zur Manipulation dieser liefert. Dies stellt das Konzept der Kapselung dar [15, S.282]. Des Weiteren wird das Konzept der Vererbung genutzt, um die Definition von neuen Objekten von bereits existierenden abzuleiten. Dies vermeidet Redundanz und sorgt für eine einfachere Erweiterbarkeit [15, S.292]. Der Polymorphismus der objektorientierten Programmierung bewirkt, dass Objekte Instanzen der von ihnen abgeleiteten Klassen aufnehmen können. Somit wird eine einzelne Schnittstelle für verschiedene Objekte bereitgestellt [13]. Die Objekte besitzen demnach einen inneren Zustand, welcher durch die eigenen Methoden verändert werden kann. Sie sind durch die Konzepte des Polymorphismus und der Vererbung somit vielseitig einsetzbar. Die objektorientierte Programmierung bietet sich für Teile eines Programms an, welche gut durch ein Objekt modelliert werden können und wo die Daten und deren Nutzung sehr nahe aneinander liegt. Des Weiteren ermöglicht sie eine verbesserte Modularität und eine einfache Erweiterung des Programms durch das Hinzufügen von neuen Unterklassen, welche in einem bereits existierenden Kontext benutzt werden können [16].

Die objektorientierte Programmierung bietet sich in der zu entwickelnden Bibliothek vor allem für die Kommunikation mit dem VNA und für die Speicherung und Visualisierung der während der Auswertung entstandenen Daten an. Eine Kommunikation mit dem VNA setzt eine Verbindung zu diesem voraus, welche optimal in einer Klasse verwaltet werden kann. Diese Verbindung stellt nämlich einen Zustand dar, welcher sich

verändern kann. Die durchzuführenden Operationen, wie beispielsweise die Festlegung der Messparameter oder der Start einer Messung, können sehr passend in eine Klasse mit der Verbindung gekapselt werden, da sie diese für die Durchführung benötigen. Welche Operationen der Nutzer der Bibliothek an dem VNA durchführen kann, wird somit durch die Klasse bestimmt und es wird festgelegt, in welchem Rahmen der VNA im Kontext der Bibliothek verwendet werden soll. Wenn die Implementation der Klasse offen gehalten wird, kann sie für unterschiedliche VNAs genutzt werden und in speziellen Fällen erweitert oder als Basisklasse verwendet werden.

Auch für die Überprüfung der während der Auswertung angefallenen Daten bietet sich eine eigene Klasse an. Die einzelnen Daten fallen zu verschiedenen Zeitpunkten an. Eine einzelne Speicherung der jeweiligen Daten wäre umständlich. Die Verwendung einer Klasse vereinfacht dies, durch die Speicherung der Daten an einem zentralen Ort mit einem einzigen Aufruf. Eine einzelne externe Speicherung dieser Klasse würde genügen, um alle angefallenen Daten auf einmal zu speichern. Die Verwaltung all dieser Daten in einer Klasse ermöglicht auch die freie Betrachtung und Überprüfung dieser. Des weiteren können die Visualisierungen der Daten als Methoden in der Klasse gespeichert werden. Auf diese Weise ist die Visualisierung für den Nutzer einfach durchzuführen und es ist möglich frei zu entscheiden, welcher Teil der Daten betrachtet werden soll. Die Erzeugung einer Instanz dieser Klasse kann von der Bibliothek nach der Auswertung automatisch durchgeführt werden und somit entsteht für den Nutzer nur ein geringer Mehraufwand, wenn er neben dem Ergebnis auch noch die einzelnen Zwischenschritte der Auswertung genauer betrachten möchte. Allerdings ist die Klasse von der spezifischen Auswertung der Messdaten abhängig und somit wird für jeden Messaufbau eine neue Klasse benötigt, wenn sich die Auswertungsschritte unterscheiden.

Das Paradigma der objektorientierten Programmierung bietet sich somit besonders zur Überprüfung der während der Auswertung angefallenen Daten und Hardwaresteuerung an.

3.3.2 Funktionale Programmierung

Das zentrale Element der funktionalen Programmierung sind Funktionen [17]. Dabei werden veränderbare Objekte und die Veränderung von Zuständen vermieden. Ein funktionales Programm ist somit eine Sammlung von Funktionen, welche sich gegenseitig nicht beeinflussen [18, S.8-9]. Das zu lösende Problem wird dabei in eine Menge von elementaren Teilproblemen aufgeteilt, welche dann von einzelnen Funktionen gelöst werden. Innerhalb des Programms existiert kein interner Zustand, die Funktionen erhalten lediglich Eingaben und liefern Ausgaben und sie werden dabei in einer gewählten Reihenfolge ausgeführt [14]. Dies findet sowohl auf der Ebene der Teilprobleme als auch auf der Ebene des Gesamtproblems statt. Die Eingabe fließt praktisch durch eine Menge von Funktionen. Veränderungen sind nur im Rückgabewert einer Funktion sichtbar [14].

Die Funktionen ähneln dabei stark mathematischen Funktionen, da sie für die gleiche Eingabe immer die selbe Ausgabe liefern [17]. Ein zentrales Konzept der funktionalen Programmierung sind „pure functions“. Es handelt sich dabei um Funktionen, die keinerlei Nebeneffekte erzeugen, sondern nur eine Rückgabe. Die Verwendung von auschließlich diesen Funktionen ermöglicht den Wechsel der Reihenfolge der einzelnen Schritte eines Programms, da keine Nebeneffekte und weiterführende Abhängigkeiten bestehen. Dies liefert eine Möglichkeit zur Optimierung eines Programms. Zudem sind diese Funktionen konzeptuell simpler und somit auch leichter auf Korrektheit zu überprüfen [18, S.23]. Des Weiteren werden sogenannte „high-order functions“ verwendet. Diese stellen Funktionen dar, welche als Argumente andere Funktionen erhalten oder eine Funktion als Rückgabewert liefern [18, S.24]. Zudem werden in der reinen funktionalen Programmierung Rekursionen anstelle von Schleifen benutzt, um die Überprüfung und Speicherung von Laufvariablen zu vermeiden [18, S.29]. All diese Konzepte sorgen für eine verbesserte Modularisierung des entstandenen Programms. Die Aufteilung des Problems in viele einzelne Funktionen liefert Funktionen, welche auf verschiedene Weisen miteinander kombinierbar sind und auch in anderen Kontexten wiederverwendet werden können [17].

Die Verwendung der funktionalen Programmierung bietet sich vor allem für die Auswertung der Messdaten an. Diese stellt nämlich einen Prozess dar, welcher bei verschiedenen Resonatoren an unterschiedlichen Stellen entweder gleich, ähnlich oder unterschiedlich ausfällt. Die Aufteilung der Auswertung in einzelne Teilprobleme ermöglicht die einfache Wiederverwendung von Schritten, welche sich bei unterschiedlichen Resonatoren überschneiden. Der Auswertungsprozess besteht zudem aus mehreren Teilschritten, welche eine Zerlegung des Gesamtprozesses nahelegt und somit auch einfach durchzuführen ist. Die Abwesenheit von Zuständen innerhalb der funktionalen Programmierung sorgt dafür, dass die Funktionen beliebig miteinander verwendet werden können, sodass, je nach Art der auszuwertenden Daten, einzelne Schritte einfach hinzugefügt oder ausgelassen werden können. Dies ermöglicht die Verwendung der Bibliothek in verschiedenen Szenarien. Die Verwendbarkeit und Erweiterbarkeit der Bibliothek wird durch die Anwendung der funktionalen Programmierung für die Auswertung gesteigert, da die Schritte vollständig unabhängig voneinander implementiert werden. Somit können auch neue Schritte einfach als neue Funktion hinzugefügt werden.

Die Verwendung der funktionalen Programmierung bei der Entwicklung der Bibliothek liefert somit einige Vorteile, welche dazu beitragen, die Anforderungen an diese zu erfüllen.

4 Umsetzung

Die hier behandelte Umsetzung bezieht sich hauptsächlich auf den Messaufbau für die Messung eines Pflanzensamens. Die zu entwickelnde Bibliothek soll den Messvorgang mit diesem Aufbau vollständig umfassen, daher wird in den folgenden Abschnitten viel Bezug auf diesen genommen. Wie in den vorherigen Kapiteln erläutert, soll die Bibliothek aber auch allgemein für andere Aufbauten verwendbar sein, daher wird auch darauf Rücksicht genommen. Die Bibliothek wurde in Python entwickelt.

4.1 Messaufbau für Pflanzensamen

Abbildung 4.1 zeigt den Messaufbau für einen Pflanzensamen. Dieser enthält wie bereits in 2.3 erläutertet einen VNA, den Resonator und zwei Koaxialkabel. Zusätzlich besitzt dieser Messaufbau eine über eine serielle Schnittstelle ansteuerbare Art von Ansauger, um einen Samen über dem Resonator platzieren zu können. Sobald er angeschaltet wird entsteht an der Spitze ein Unterdruck. Wenn dann ein Samen an die Öffnung gehalten wird, wird dieser dadurch angesogen und hängt dann über dem Resonator. Wird er wieder abgeschaltet fällt der Samen durch den Resonator. Das Gerät wird über einen USB-Anschluss an einen Computer angeschlossen, welcher in diesem Fall als serielle Schnittstelle benutzt wird. Über diesen Anschluss können Befehle an das Gerät gesendet werden, welche das Ansaugen entweder starten oder stoppen.

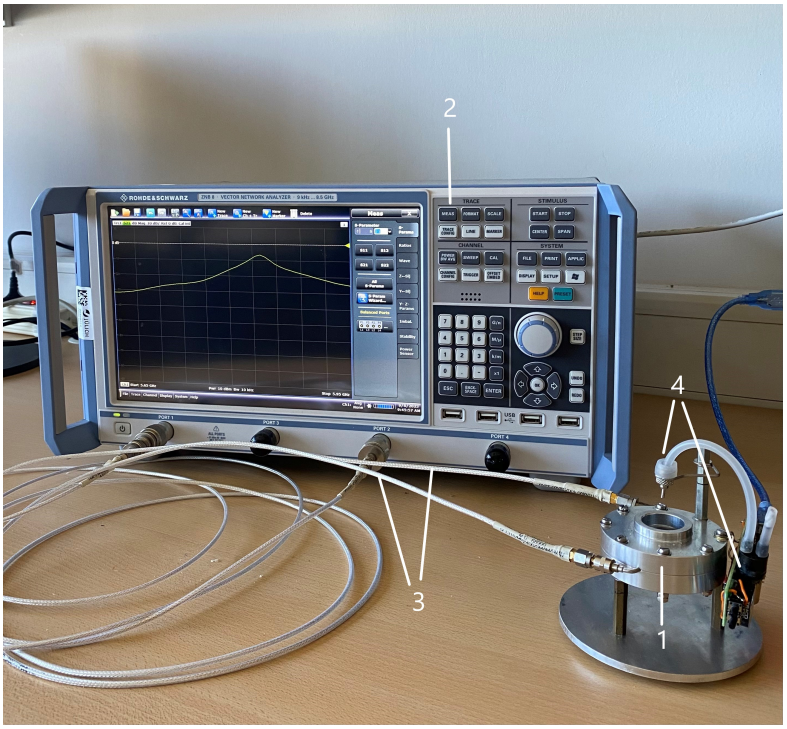


Abbildung 4.1: Messaufbau für Messung an einem Pflanzensamen
1: Resonator, 2: VNA, 3: Koaxialkabel, 4: Ansauger

Um eine Messung mit dem Aufbau durchzuführen muss der VNA über einen Computer kontrollierbar sein. Der hier verwendete VNA von Rhode & Schwarz ist über eine LAN-Schnittstelle ansteuerbar [19]. Wenn sich der VNA in einem Netzwerk befindet, dann kann er über seine IP-Adresse von anderen Computern im selben Netzwerk angesprochen werden. Auf diese Weise können Befehle an ihn gesendet werden. Für die Entwicklung der Bibliothek wurde der zugehörige Computer über eine Netzwerkschnittstelle mithilfe eines LAN-Kabels mit dem VNA verbunden.

Somit ist für diesen Messaufbau sowohl der VNA, als auch der Ansauger über einen Computer steuerbar. Dadurch kann die gesamte Messung über die Steuerung des Computers vollzogen werden.

4.2 Kommunikation mit dem VNA

4.2.1 Das RsInstrument-Paket

RsInstrument ist ein Python-Paket für die remote-Steuerung von Rhode & Schwarz Messinstrumenten [20]. Mithilfe dieses Pakets kann der VNA über ein Python-Programm

gesteuert werden, solange er sich im gleichen Netzwerk wie der ausführende Computer befindet. Um im Code eine Verbindung mit dem Gerät aufzubauen, muss lediglich eine Instanz der Klasse RsInstrument erzeugt werden. Dieser muss die zugehörige IP-Adresse übergeben werden und die Verbindung wird hergestellt. Über zwei verschiedene Methoden können dann Befehle an den VNA gesendet werden, um zum einen Aktionen durchzuführen und zum anderen Daten anzufordern. Intern verwendet das Paket für die Kommunikation VISA(Virtual instrument software architecture). Diese stellt eine Programmierschnittstelle für die Kommunikation zwischen Computern und Messgeräten dar [21].

Das Paket ist allgemein für verschiedene Rhode & Schwarz Messgeräte verwendbar. Die Definition der gerätespezifischen Befehle ist in den zugehörigen Handbüchern zu finden (für den hier verwendeten VNA siehe [22]). Somit ermöglicht die Verwendung dieses Pakets eine einfache und kompakte Steuerung des VNAs.

4.2.2 Implementation innerhalb der Bibliothek

Die Verwaltung der Verbindung zu dem VNA wird innerhalb der Bibliothek von einer eigenen Klasse geregelt. Dabei wird das RsInstrument-Objekt in eine Klasse gekapselt, um die Nutzung des VNAs bei der Verwendung der Bibliothek möglichst einfach zu gestalten. Wie im vorherigen Abschnitt erwähnt ist das für die Kommunikation mit dem VNA verwendete Paket nicht spezifisch für den hier verwendeten VNA, sondern generell für Messgeräte von Rhode & Schwarz konzipiert worden. Die Kapselung des Objektes in eine eigene Klasse für den VNA sorgt für die Verbergung des eigentlichen Objektes vor der Außenwelt. Auf diese Weise wird das RsInstrument-Objekt nur in der von der Bibliothek vorhergesehenen Weise verwendet. Des weiteren bietet sich die Verwendung einer eigenen Klasse an, da die Verbindung zu dem VNA ein bestehender Zustand ist, welcher benutzt und manipuliert werden kann (siehe Kapitel 3.3.1).

Die Klasse ist simpel gehalten und enthält nur die nötigsten Methoden (siehe Abbildung 4.2). Die privaten Methoden werden verwendet um den Verbindungsaufbau vorzubereiten und zu testen. Weitere Methoden behandeln die Aufgaben Befehle zu schreiben, Daten abzufragen oder sonstige kleinere Aufgaben wie das Neustarten der Verbindung. Die Methode *meassetup* wird genutzt, um eine Messung vorzubereiten. Innerhalb der Methode werden mehrere Befehle an den VNA gesendet. Dabei werden zum einen die spezifischen Parameter für die Messung (Anzahl von Einzelmessungen, Messpunkte pro Einzelmessung und Frequenzbereich) gesetzt. Zudem ist der VNA in der Lage eine Kalibrierung durchzuführen, um bei den Messdaten durch Rauschen aufgetretene Ungenauigkeiten zu eliminieren. Innerhalb der Methode wird diese Option, je nach übergebenen Wert, an- oder ausgeschaltet.

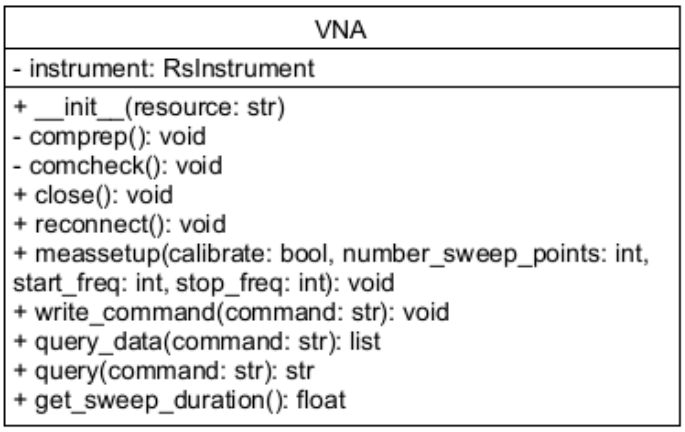


Abbildung 4.2: UML-Darstellung der VNA-Klasse

4.2.3 Messdurchführung

Die Durchführung einer Messung eines Pflanzensamens stellt einen simplen und in sich abgeschlossenen Ablauf dar. Es ist eine einzelne Routine, unabhängig von anderen Gegebenheiten. Aus diesem Grund wurde die Messdurchführung als Sammlung von Funktionen in einem einzelnen Modul implementiert (siebe Abbildung 4.3). Alle Funktionen, welche sich mit der Beschaffung von Messdaten befassen, befinden sich innerhalb der Bibliothek im selben Modul. Die Bestimmung der Messdaten ist dabei in zwei Routinen unterteilt: Die Durchführung des eigentlichen Messvorgangs und das Auslesen der Daten. Bei beiden handelt es sich um einfache Abläufe.

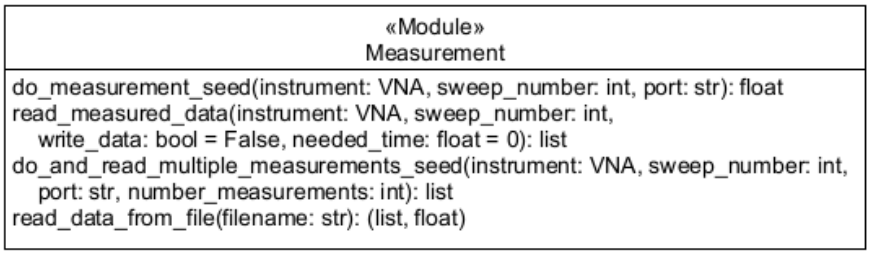


Abbildung 4.3: UML-Darstellung des Moduls für Messungen

Bevor die Messung durchgeführt werden kann, muss zunächst die Methode *meassetup* der Klasse VNA ausgeführt werden und der Samen muss über dem Resonator platziert werden. Nachdem diese Schritte erfolgt sind, kann die Routine gestartet werden. Die Durchführung stellt eine einzelne Funktion dar, welche unter anderem eine Instanz der VNA-Klasse übergeben bekommt. Zunächst wird mithilfe des Python-Packets *pySerial*

[23] der Befehl an den Ansauger gesendet, um den Samen fallen zu lassen. Direkt im Anschluss wird der aktuelle Zeitpunkt gespeichert und über die Instanz der VNA-Klasse der Befehl zum Starten der Messung gesendet. Dieser Aufruf blockiert den Prozess solange, bis die Messung abgeschlossen ist. Als Rückgabewert liefert die Funktion dann die Dauer der Messung, welche über eine einfache Substraktion des zuvor gespeicherten Zeitpunkts mit dem aktuellen bestimmt wird. Die Dauer ist bei den späteren Betrachtungen noch von Bedeutung.

Die von der Messung bestimmten Werte werden im Zwischenspeicher des VNA gespeichert. Das Abrufen dieser Daten wird in einer zweiten Routine abgearbeitet. Wie in Kapitel 2.5 erläutert stellen die gemessenen s_{21}-Werte einen komplexen Wert dar. Intern werden die Werte in einer Liste gespeichert, wobei zwei aufeinander folgende Werte einen Messwert repräsentieren. Die erste Zahl ist der Realteil und die zweite der Imaginärteil. Diese Liste wird in der Routine über eine Instanz der VNA-Klasse abgerufen. Das Format der Liste wird bei der Speicherung in Python entsprechend beachtet. Optional kann über einen Übergabeparameter der Funktion festgelegt werden, ob die gemessenen Daten lokal auf dem ausführendem System in Form einer Textdatei gespeichert werden soll. Als Rückgabewert liefert die Funktion eine Liste der Messwerte, in der ein Eintrag eine komplette komplexe Zahl darstellt.

Neben diesen zwei wichtigsten Routinen enthält das Modul noch zwei weitere. Zum einen eine, um Daten aus einer Datei einzulesen, welche von der zuvor erläuterten Funktion erstellt wurde. Dies ist vor allem nützlich, um nach Veränderungen der Verarbeitungsroutine dieselben Daten nochmal für einen Vergleich auszuwerten. Zum anderen existiert noch eine Funktion, welche beide Hauptroutinen verwendet um mehrere Messungen direkt hintereinander durchzuführen.

4.3 Implementation der Messauswertung

Wie zuvor in Kapitel 3.3.2 erläutert, bietet die funktionale Programmierung große Vorteile für die Implementation der Auswertung. Aus diesen Gründen wurde sie hier rein funktional implementiert. Die einzig nicht-funktional entwickelten Programmteile sind die vereinzelte Verwendung von Schleifen. Diese werden normalerweise in der funktionalen Programmierung durch Rekursionen ersetzt. Die Bibliothek wurde jedoch in Python entwickelt und in dieser Sprache findet intern keine Optimierung für rekursive Aufrufe statt [18, S.31]. Daher ist meistens die Verwendung von Schleifen in Python schneller als Rekursionen. Aus diesem Grund wurden, trotz des funktionalen Ansatzes, vereinzelt Schleifen genutzt. Alle im Programmcode verwendeten Schleifen hätten allerdings auch als Rekursion implementiert werden können.

Die gesamte Auswertung wurde in dem Modul *Evaluate* implementiert. Um einen funktionalen Aufbau wie in Kapitel 3.3.2 zu erstellen, wurde die in Kapitel 2.5 erläuterte Auswertung der Messdaten in die kleinstmöglichen Teilaufgaben aufgespalten. Für die Nutzung des Moduls im Kontext anderer Messaufbauten müssen alle Teilaufgaben für alle Schritte aus Kapitel 2.5 beachtet und implementiert werden. Abbildung 4.4 zeigt eine grobe Unterteilung in Teilaufgaben und wie sie miteinander verwendet werden. Einzelne dieser Teilaufgaben können in der Implementierung je nach Messaufbau unterschiedlich ausfallen. Dies wird bei der Betrachtung der jeweiligen Teilaufgabe näher beleuchtet.

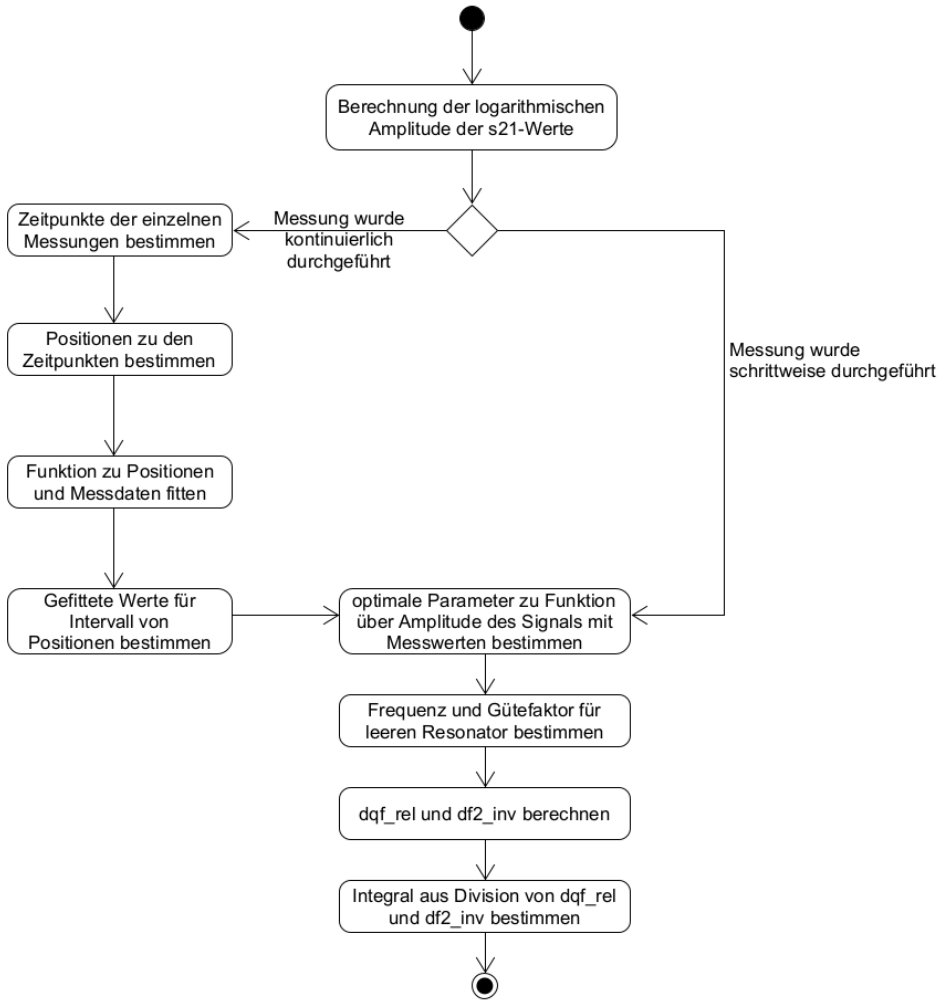


Abbildung 4.4: UML-Aktivitätsdiagramm für die Auswertung einer Messung

Die Teile der Abbildung 4.4 wurden dann in einer oder auch mehreren Funktionen implementiert, falls sie noch weiter aufteilbar sind. Sie funktionieren unabhängig voneinander, um auch bei anderen Implementationen einzelner Schritte noch miteinander kompatibel

zu sein. Die in dem Modul enthaltenen Funktionen sind in Abbildung 4.5 gezeigt.

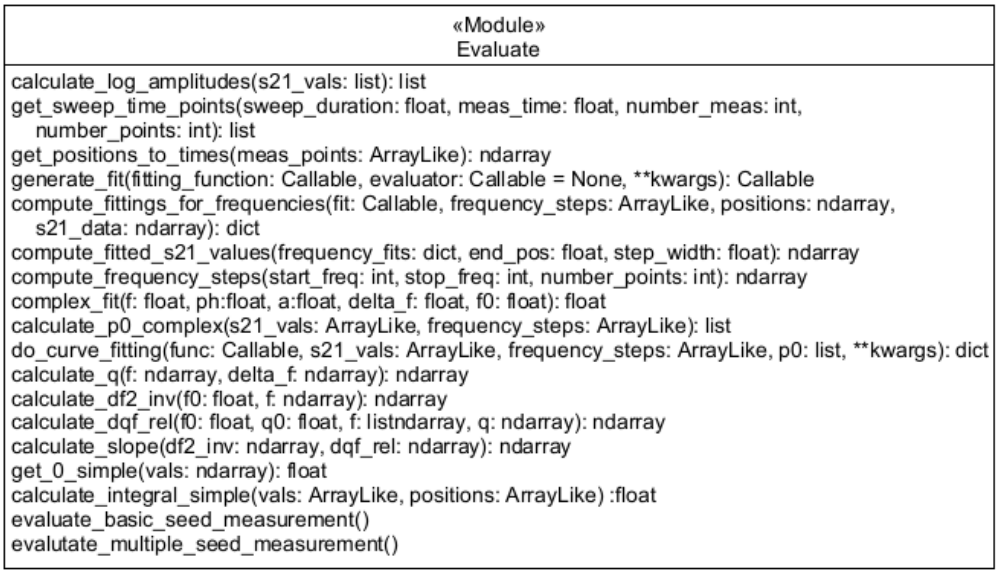


Abbildung 4.5: UML-Darstellung des Moduls für die Auswertung

In den folgenden Unterkapiteln wird die Umsetzung der einzelnen Schritte aus Abbildung 4.4 durch die Funktionen des Moduls erläutert. Dabei wird besonders auf die funktionale Konzipierung der Funktionen eingegangen, welche die flexible Wiederverwendung der Funktionen für unterschiedliche Messaufbauten ermöglicht. Um die Umsetzung der Theorie zu verdeutlichen, werden in den meisten Unterkapiteln Visualisierungen in Form von Graphen des zu dem Zeitpunkt aktuellen Zustands der Daten verwendet. Diese Graphen entsprechen der Auswertung einer echten Messung und sollen die Verarbeitung der Daten veranschaulichen. Die der Visualisierungen zu Grunde liegenden Messung wurde an einem Weizensamen durchgeführt. Der gewählte Frequenzbereich betrug 5.65 GHZ bis 5.95 GHz. Die Anzahl an Einzelmessungen belief sich auf 50 mit jeweils 10 Messpunkten. Das bedeutet, dass in jeder Einzelmessung für folgende Frequenzen jeweils gemessen wurde: 5.65 GHz, 5.68333 GHz, 5.71667 GHz, 5.75 GHz, 5.78333 GHz, 5.81667 GHz, 5.85 GHz, 5.88333 GHz, 5.91667 GHz und 5.95 GHz. Insgesamt liegen demnach 500 gemessene Werte vor.

4.3.1 Berechnung der logarithmischen Amplitude

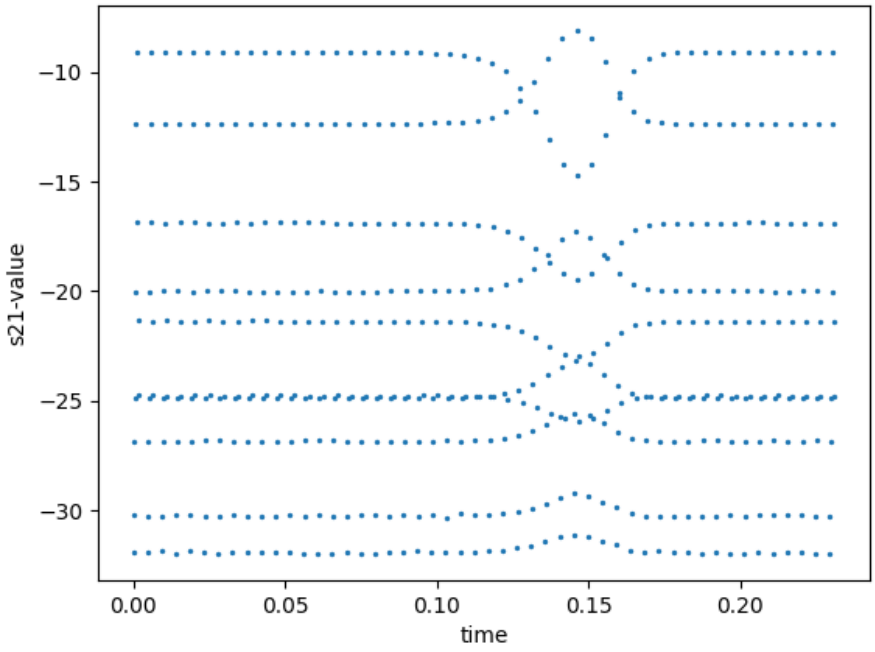
Die Berechnung der logarithmischen Amplitude der s_{21}-Werte ist unabhängig von dem Messaufbau durch die bereits in Kapitel 2.5 gezeigte Formel 2.1 gegeben. Die Umsetzung der Formel ist simpel und befindet sich in der Funktion *calculate_log_amplitudes*. Als Eingabe wird lediglich eine Liste von komplexen Werten erwartet. Die Funktion ist

demnach unabhängig vom Messaufbau anwendbar. Im weiteren Verlauf werden diese Werte als ausgewertete s_{21}-Werte bezeichnet.

4.3.2 Bestimmung der Zeitpunkte und Positionen

Die Bestimmung der Zeitpunkte und Positionen von den einzelnen Messungen findet nur im Falle einer kontinuierlich durchgeführten Messung Anwendung. Bei dieser Teilaufgabe bestimmt der spezifische Messaufbau die konkrete Implementation. Dessen Berechnungen unterscheiden sich nämlich je nach Messverhalten. Hier werden zunächst alle Messwerte gemeinsam betrachtet, unabhängig von der verwendeten Frequenz bei der Messung eines Wertes.

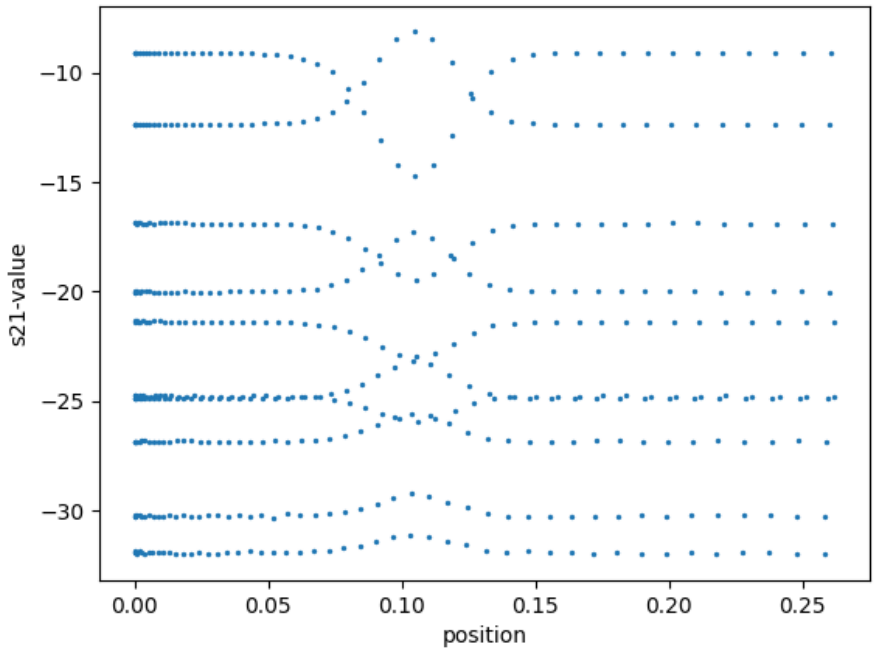
Die konkrete Implementierung für einen Pflanzensamen befindet sich in den Funktionen *get_sweep_time_points* und *get_positions_to_times*. Erstere leistet dabei die Vorarbeit für die Zweitere, indem sie anhand der insgesamt benötigten Messdauer, der Anzahl von Einzelmessungen, der Anzahl von Messpunkte pro Einzelmessung und der Dauer einer Einzelmessung den Messzeitpunkt jedes einzelnen Messwertes bestimmt. Abbildung 4.6 zeigt eine Visualisierung der Messwerte bezogen auf den durch *get_sweep_time_points* bestimmten Zeitpunkt, an dem sie gemessen wurden. Jeder Punkt stellt einen Messwert dar. Es ist zu erkennen, dass die Messwerte zunächst auf zehn verschiedenen Linien verlaufen. Dies ist der Fall, da im Zuge einer Einzelmessung in diesem Beispiel, wie zuvor erläutert, für zehn verschiedene Frequenzen gemessen wird. Die für einen Messpunkt verwendete Frequenz beeinflusst den gemessenen Wert. Der Resonator ist bis zu einer Zeit von ungefähr 120 Millisekunden leer und daher sind die Messwerte für die einzelnen Frequenzen fast konstant. Ab diesem Zeitpunkt betritt allerdings der fallende Samen den Resonator und die Messwerte verändern sich, wie in der Abbildung zu erkennen ist. Ab einem Zeitpunkt von ungefähr 170 Millisekunden ist der Samen nicht mehr innerhalb des Resonators und die gemessenen Werte sind wieder auf der gleichen Höhe wie zuvor. Mithilfe der Zeitpunkte und der Formel des freien Falls $s = \frac{gt^2}{2}$ berechnet *get_positions_to_times* dann die zugehörigen Positionen. In diesem Fall stellen die Positionen den Abstand des Samens zum Ort von dem er gefallen ist dar. Wenn im weiteren Verlauf des Kapitels von der Position eines Messwerts gesprochen wird, bezieht sich dies auf diese Definition.



Zeit in Sekunden

Abbildung 4.6: Gemessene Werte in Bezug zu deren Messzeitpunkt

Am Ende jeder dieser Funktionen liegt eine einfache Liste mit den Positionen für jeden Messwert vor, welche im Anschluss dann weiterverwendet wird. Abbildung 4.7 zeigt die graphische Darstellung der Messwerte bezogen auf deren Position. Es ist zu erkennen, dass der Abstand zwischen den Werten auf der x-Achse mit steigender Position zunimmt, da der Samen durch den freien Fall beschleunigt wird. Die weiteren Schritte können demnach auch für andere Aufbauten verwendet werden, wenn für diese lediglich die Positionen der Messpunkte in dieser Form vorliegen oder bestimmt werden.



Position in Meter

Abbildung 4.7: Gemessene Werte in Bezug zu deren relativen Positionen zur Ursprungsposition

4.3.3 Interpolationsfunktion zu Positionen und Messdaten bestimmen

Ab diesem Zeitpunkt wird differenziert zwischen den Messwerten einer Einzelmessung. Zusammen betrachtet werden die Werte, welche mit der selben Frequenz gemessen wurden. Im Falle einer kontinuierlichen Messung muss aus den gemessenen Daten für jede verwendete Frequenz eine Funktion interpoliert werden, um mit dieser Messdaten in der korrekten Form zu erzeugen (siehe Kapitel 2.5). Abbildung 4.8 zeigt beispielsweise eine solche Interpolation für eine Frequenz einer Messung im Vergleich zu den gemessenen Werten. Die interpolierte Funktion erhält als Eingabewert eine Position und liefert dazu einen Messwert zurück. Um aus den gegebenen Daten eine Interpolation zu erstellen gibt es viele verschiedene Möglichkeiten. Mehrere unterschiedliche Python-Pakete bieten verschiedene Implementierungen von mathematischen Methoden, um eine solche Funktion zu erstellen. Je nach Messaufbau und der Genauigkeit der gemessenen Daten werden eventuell andere Interpolationsmethoden benötigt. Demnach ist die verwendete Methode nicht einheitlich für verschiedene Szenarien, sondern kann variieren. Das bedeutet, um die Bibliothek möglichst offen zu halten, müssen die Methoden, welche die Interpolationen nutzen und erzeugen, in einer solchen Art gestaltet sein, dass sie mit den verschiedensten Interpolationsmethoden kompatibel sind. Dafür wurde diese Teilaufgabe in drei Funktionen aufgeteilt. Diese sind $generate\_fit$, $compute\_fittings\_for\_frequencies$ und $compute\_fitted\_s21\_values$.

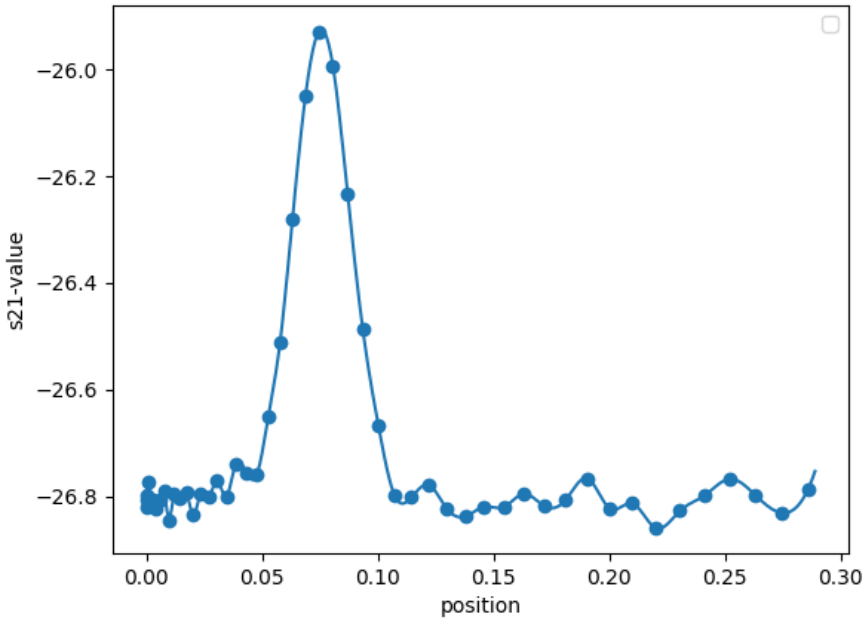


Abbildung 4.8: Interpolationskurve und Datenpunkte zu den Messwerten zu einer Frequenz

Aufbau und Funktionsweise der Funktionen

Die von verschiedenen Bibliotheken bereitgestellten Funktionen für die Interpolation einer Funktion fungieren nicht immer gleich. Sowohl die Rückgabewerte als auch die benötigten Paramter solcher Funktionen können sich unterscheiden. Um dies näher zu bringen, werden im folgenden zwei verschiedene Funktionen für eine Interpolation betrachtet.

Die Funktion *interp1d* aus der Python-Bibliothek scipy kann aus übergebenen x- und y-Werten eine Funktion interpolieren. Sie erhält als Übergabeparameter unter anderem die x- und y-Werte, auf denen die erstellte Interpolation basiert, und einen String, welcher die Art der Interpolation angibt. Als Rückgabe liefert sie die berechnete Interpolationsfunktion, welche direkt für jegliche x-Werte die zugehörigen y-Werte berechnet [24].

Eine weitere Funktion aus scipy ist *splrep*. Diese Funktion interpoliert einen B-Spline auf Basis der übergebenen Daten. Dieser stellt eine Funktion dar, welche stückweise aus Polynomen besteht. Die Ordnung des Splines gibt den Grad der Polynome an. Als Parameter erhält diese Funktion unter anderem ebenfalls die zugehörigen x- und y-Werte und die Ordnung des Splines. Als Rückgabe liefert sie die Koeffizienten des interpolierten B-Splines. Diese müssen noch in eine eigene Funktion weiterverarbeitet werden, um mit

dem Ergebnis von *splrep* einfach zu einem gegebenen x-Wert den zugehörigen y-Wert zu bestimmen [25].

Diese beiden Interpolationsmethoden sind ein Beispiel für das Problem dieser Teilaufgabe. Je nach der gewählten Interpolation unterscheiden sich die Übergabeparameter und die Art des Rückgabewertes. Um eine auswertbare Interpolationsfunktion zu erhalten, müssen im Falle von *splrep* noch weitere Schritte vollzogen werden, während $interp1d$ direkt als Rückgabewert eine solche Funktion liefert. Dies stellt ein Problem dar, da die Funktionen der Bibliothek generell gehalten werden müssen, sodass sie für verschiedene Szenarien einheitlich verwendbar sind. Somit muss dieser Schritt in der Ausführung unabhängig von der gewählten Interpolationsmethode sein. Um dies zu erreichen wurde die Funktion $generate\_fit$ erstellt.

Die Funktion bekommt folgende Parameter übergeben:

- $interp\_function$: Eine Interpolationsfunktion wie beispielsweise $interp1d$ oder *splrep*.
- *evaluator*: Dabei handelt es sich um einen optionalen Parameter. Wenn er nicht übergeben wird, enthält er keinen Wert. Er stellt eine Auswertungsfunktion dar. Beispielsweise für *splrep* wäre *evaluator* eine Funktion, welche aus dem Ergebnis der Interpolationsmethode eine Funktion erstellt, welche mit einem x-Wert aufgerufen werden kann und den zugehörigen y-Wert liefert. Hingegen für $interp1d$ ist dieser Parameter nicht benötigt, da das Ergebnis der Interpolation bereits eine solche Funktion ist.
- $^{**}kwargs$: Eine beliebig große Menge von Schlüsselwort-Parametern. Über diese können der Funktion alle von der Interpolationsfunktion benötigten Parameter übergeben werden.

Innerhalb der Funktion werden zwei Prinzipien der funktionalen Programmierung verwendet, um das Problem der Teilaufgabe zu lösen (siehe folgender Codeblock). Zunächst wird das Prinzip von sogenannten Closures benutzt. Dieses behandelt den Fall, dass bei der Generierung von einer Funktion innerhalb einer anderen Funktion der Geltungsbereich der äußeren Funktion in der inneren gespeichert wird. *create_interp* (Zeile 2) stellt die Funktion dar, welche in sich die übergebenen Parameter an *generate_interp* speichert, da sie sich innerhalb des gleichen Geltungsbereiches befindet. Diese Funktion wird von *generate_interp* als Rückgabe zurückgegeben (Zeile 8), um eine Funktion zu generieren, welche unabhängig von der Interpolationsmethode eine einheitlich aufrufbare Interpolation zu gegebenen x- und y-Werten erzeugen kann. Eine mit diesem Rückgabewert erzeugte Interpolation ist verwendbar, indem ihr ein x-Wert übergeben wird zu dem dann ein y-Wert berechnet wird. Damit *generate_interp* korrekt funktioniert, darf der Parameter *evaluator* nur übergeben werden, wenn die verwendete Interpolationsfunktion als Ergebnis eine nicht direkt verwendbare Funktion zurückgibt (Zeile 3-6). Für den Fall, dass sie als Ergebnis eine solche Funktion liefert, darf keine

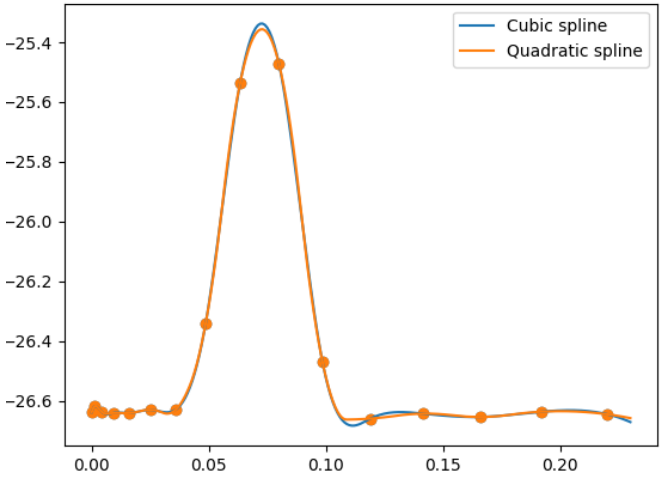
Auswertungsfunktion übergeben werden. Die spezifischen Parameter für die Interpolationsfunktion werden mittels $^{**}kwargs$ an die $generate\_fit$ übergeben und in $create\_fit$ dann dieser überreicht. Das zweite funktionale Prinzip ist die Nutzung von Funktionen als Übergabeparameter und Rückgabewert.

```
1 def generate_interp(interp_function: Callable, evaluator: Callable = None
    ↪ , **kwargs) -> Callable:
2         def create_interp(x: npt.ArrayLike, y: npt.ArrayLike) -> Callable
              ↪ :
3                 if evaluator:
4                         return evaluator(interp_function(x, y, **kwargs))
5                 else:
6                         return interp_function(x, y, **kwargs)
7
8         return create_interp
```

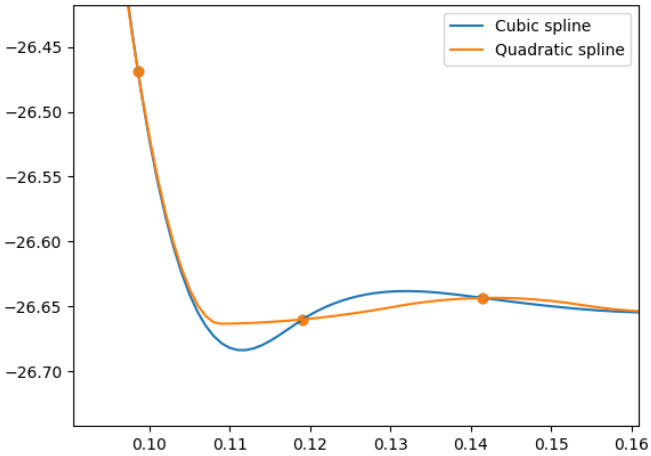
$compute\_fittings\_for\_frequencies$ nutzt dann die durch $generate\_fit$ erzeugte Funktion, um für jede gemessene Frequenz mit den zugehörigen Positionen und s_{21}-Werten eine Interpolation zu generieren. Diese Interpolationen werden zuletzt in der Funktion $compute\_fitted\_s21\_values$ verwendet, um für ein Intervall von Positionen die interpolierten s_{21}-Werte für jede Frequenz zu berechnen.

Verwendete Interpolationsmethode

Speziell für den Messaufbau zur Berechnung an einem Pflanzensamen wird die bereits erwähnte $interp1d$-Funktion genutzt. Diese bietet die Funktionalität für verschiedene Arten von Interpolationen. In diesem Fall wird die quadratische Spline-Interpolation verwendet. Dabei wird für die Bestimmung eines y-Wertes zu einem gegebenen x-Wert eine quadratische Funktion zwischen den drei dem gegebenen x-Werte am nächsten liegenden gemessenen Punkten interpoliert. Diese Funktion wird dann an der gegebenen Stelle ausgewertet [26]. Wenn eine geringe Anzahl von Messpunkten pro Einzelmessung gewählt wird, kann es zu dem Fall kommen, dass die Interpolation zum Überschwingen neigt, da für größer werdende Positionen die Abstände zwischen den Messwerten aufgrund der Beschleunigung des Samens während des freien Falls immer weiter zunehmen. Diese Abstände führen in manchen Fällen zum Überschwingen der interpolierten Funktion, sodass sie zwischen zwei Datenpunkten ausreißt. Ein Vergleich mehrerer Interpolationsmethoden hat gezeigt, dass ein quadratischer Spline auf diese Art von Fehler bei diesen Messdaten am wenigsten zu diesem Fehlverhalten neigt. In Abbildung 4.9 ist beispielsweise bei dem Vergleich zwischen einem quadratischen und kubischen Spline erkennbar, dass im quadratischen Fall das Überschwingen geringer ausfällt.



(a) Vollständiger Vergleich



(b) Vergrößerte Darstellung des Überschwingungsbereich

Abbildung 4.9: Vergleich einer kubischen und quadratischen Spline Interpolation

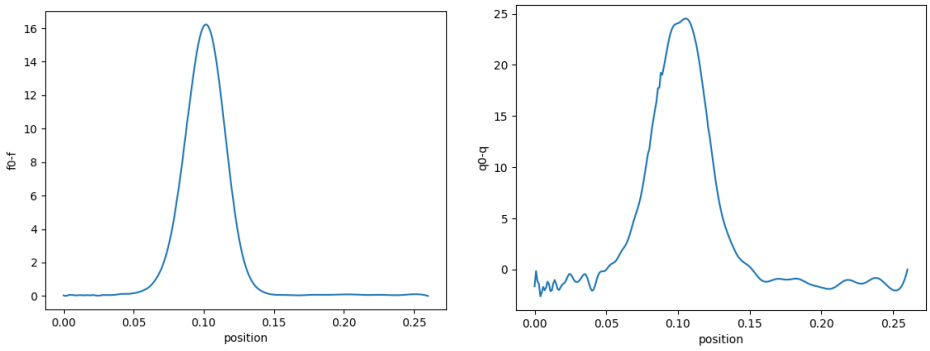
4.3.4 Bestimmung der Resonanzfrequenz und des Gütefaktors

Die zuletzt betrachteten Auswertungsschritte werden nur für kontinuierlich durchgeführte Messungen benötigt. Die ab jetzt folgenden Teilaufgaben sind wieder für beide Messarten von Bedeutung. Wie in Kapitel 2.5 genauer erläutert lässt sich die logarithmische Amplitude eines durch den VNA empfangenen Signals durch Formel 2.3 berechnen. Bei beiden Messarten liegen schließlich für ein Intervall von Positionen für jede Position mehrere ausgewertete s_{21}-Werte vor. Diese Daten werden bei einer kontinuierlichen Messung durch eine Interpolation bestimmt und bei einer inkrementellen liegen sie bereits nach der Messung in dieser Form vor. Wie viele Werte für eine Positionen vorliegen, ist

abhängig von der gewählten Anzahl an Messpunkten pro Einzelmessung.

Für jede dieser Positionen müssen die Variablen der Funktion 2.3 bestimmt werden. Für die weitere Auswertung sind vor allem die Resonanzfrequenz und der Gütefaktor von Bedeutung. Die Funktionswerte der Formel sind in Form der ausgewerteten s_{21}-Werte für jede Position gegeben, welche jeweils in Abhängigkeit von der verwendeten Frequenz gemessen bzw. bestimmt wurden. Somit erhält die Funktion als Eingabe eine Frequenz und liefert als Rückgabe einen ausgewerteten s_{21}-Wert, welcher wie in 2.5 erläutert die logarithmische Amplitude eines empfangenen Signals einer Messung darstellt. Die konkrete Form der Funktion ist von dem jeweiligen Messaufbau abhängig. Daher muss bei der Umsetzung innerhalb der Bibliothek erneut beachtet werden, dass die Implementation dieses Auswertungsschrittes unabhängig von dem jeweiligen Messaufbau ist.

Erfahrungen mit anderen Resonatoren haben in der Vergangenheit gezeigt, dass die graphische Darstellung der Resonanzfrequenz und des Gütefaktors gute Faktoren sind, um abzuschätzen, inwiefern die Auswertung korrekt funktioniert. Bei korrekter Durchführung sollten die Graphen wie in Abbildung 4.10 abgebildet aussehen. Die Grafik stellt die Entwicklung der Resonazfrequenz und des Gütefaktors im Vergleich zu den jeweiligen Werten für einen leeren Resonator dar. Es ist zu erkennen, dass der Samen diese Eigenschaften beeinflusst. Diese Werte werden im weiteren Verlauf genutzt, um die finalen Werte der Auswertung zu bestimmen. Der Graph des Gütefaktors ist dabei empfindlicher auf mögliche Ungenauigkeiten während der Messung oder der Auswertung. Anhand dieser Grafik ist gut zu erkennen, ab wann das zu messende Objekt den Resonator betreten hat und somit Einfluss auf das elektrische Feld ausgeübt hat.



(a) Resonanzfrequenz (b) Gütefaktor

Abbildung 4.10: Graphische Darstellung der Entwicklung der bestimmten Resonanzfrequenz und des Gütefaktors bei der Messung eines Pflanzensamens

Implementierung

Dieser Auswertungsschritt ist in der Funktion *do_curve_fitting* umgesetzt. Um für jede Position die optimalen Werte der Variablen zu bestimmen, wird die Ausgleichsrechnung mit der Methode der kleinsten Quadrate angewendet. Die Ausgleichsrechnung berechnet für eine Menge von Datenpunkten die optimalen Parameter einer dahinter liegenden Formel. Optimal bezieht sich auf die bestmögliche Anpassung der Funktion an die Daten. Um bestmöglich in diesem Fall zu konkretisieren, wird die Methode der kleinsten Quadrate verwendet. Sei y_i der i-te Messwert, f die zu bestimmende Funktion und x_i der Eingabewert, dann werden die Parameter der Funktion bei der Ausgleichsrechnung mit der Methode der kleinsten Quadrate bestimmt, sodass folgender Wert minimiert wird [27, S.129]:

$$\sum_{i=0}^{n}[y_i - f(x_i)]^2 \tag{4.1}$$

Für die Umsetzung wurde die Funktion *curve_fit* aus dem Python-Paket scipy verwendet, welche das zuvor erläuterte Verfahren implementiert [28]. Die zugehörige Funktion der entwickelten Bibliothek erhält als Übergabeparameter die hinter den Datenpunkten liegende Funktion, die Messwerte, die Frequenzen für die während der Einzelmessungen gemessen wurde und optional Initialwerte für die Variablen der Funktion. Des Weiteren können Schlüsselwortargumente übergeben werden, um gegebenenfalls weitere Parameter an *curve_fit* zu übergeben. Innerhalb der Funktion werden dann für jede der vorliegenden Postionen die optimalen Werte der Funktion für die Messwerte und zugehörigen Frequenzen bestimmt. *curve_fit* führt die Ausgleichsrechung durch und liefert als Rückgabewert die optimalen Werte der Variablen. Die Rückgabe von *do_curve_fitting* sind dann diese Werte für jede der Positionen. Um diesen Auswertungsschritt für einen beliebigen Messablauf durchzuführen, muss nur die Funktion für diesen Messaufbau in konkreter Form vorliegen.

Die Besonderheit der Ausgleichsrechnung ist, dass sie für eine besonders gut die Daten beschreibende Funktion sorgt indem sie nicht unbedingt exakt durch jeden einzelnen Punkt verläuft. [27, S.129-132]. Dies ist in Abbildung 4.11 erkennbar.

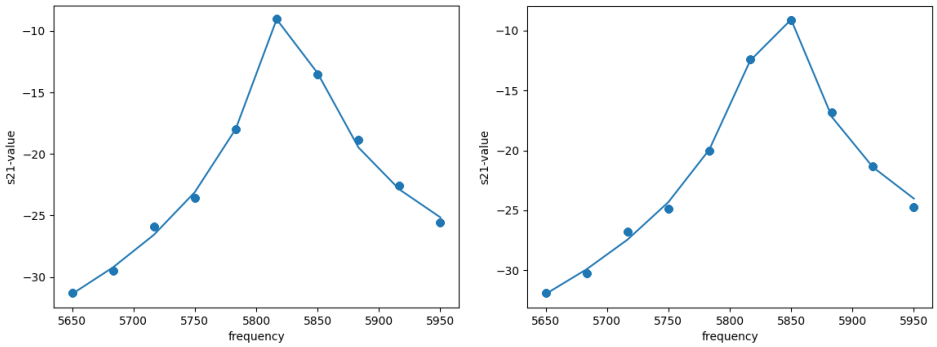


Abbildung 4.11: Zwei Beispiele für berechnete Kurven auf Basis der Formel 4.2 für verschiedene Positionen

Konkrete Umsetzung für Pflanzensamen

Für den Messaufbau zur Bestimmung an einem Pflanzensamen muss zunächst die Formel 2.3 konkretisiert werden. In diesem Fall wird primär die erste Mode betrachtet, jedoch wird der Einfluss der zweiten zusätzlich noch berücksichtigt. Sie sieht folgendermaßen aus:

$$S(f) = 20log_{10}\left(\left|\frac{A}{1+2jQ\frac{f-f_0}{f_0}} + \frac{0.025\exp^{j\varphi}}{1+2j5\frac{f-7500}{7500}}\right|\right) \tag{4.2}$$

Γ_0 ist in diesem Fall 1, da dies durch den verwendeten VNA gegeben ist. Die Störungen, welche durch Γ_s dargestellt werden, fallen für diesen Messaufbau sehr gering aus. Sie sind vernachlässigbar und daher wird Γ_s auf 0 gesetzt. Die zweite Mode wird nicht gemessen, sondern es wird lediglich der Einfluss dessen auf die erste mit einberechnet. Erfahrungen in der Verwendung mit anderen Resonatoren in der Vergangenheit haben gezeigt, dass für den verwendeten Frequenzbereich und bei der hauptsächlichen Betrachtung der ersten Mode es genügt nur den Einfluss der zweiten zu beachten. Um diesen mit einzubeziehen, werden die meisten Werte der zweiten Mode anhand des auf dem VNA sichtbaren Spektrums nach einer Messung approximiert. Um den Effekt dennoch gut zu erfassen, eignet es sich, eine Variable nicht festzulegen. Tests haben in der Vergangenheit gezeigt, dass die Phasenverschiebung dafür passend ist. Die Phasenverschiebung wurde in der ersten Mode nicht mitaufgenommen, da eine Verschiebung relativ zu etwas betrachtetem ist. Betrachtet wird hier die erste Mode und daher ist die Phasenverschiebung der zweiten Mode relativ zu der ersten.

Diese Formel wurde dann in einer eigenen Funktion implementiert und kann dann von *do_curve_fitting* benutzt werden. Auf diese Weise können speziell für diesen Messaufbau die optimalen Parameter der Formel 4.2 bestimmt werden.

4.3.5 Resonanzfrequenz und Gütefaktor für leeren Resonator bestimmen

Um die weiteren Werte der Verarbeitung zu berechnen, werden die Resonanzfrequenz und der Gütefaktor eines leeren Resonators benötigt. Durch den funktionalen Aufbau der Bibliothek sind die Funktionen unabhängig voneinander. Funktionen, die weitere Werte berechnen, bekommen diese Werte für einen leeren Resonator nur als Parameter übergeben. Daher können die Werte auf beliebige Art und Weise bestimmt werden. Eine Möglichkeit sie zu bestimmen ist, eine Messung und die anschließende Auswertung mit einem leeren Resonator vorzunehmen. Die durch die Ausgleichsrechnung berechnete Resonanzfrequenz und der Gütefaktor sind dann für alle Positionen fast gleich, es existieren lediglich kleine Abweichungen durch Messungenauigkeiten. Dann ist es möglich die Werte an einer beliebigen Position auszuwählen oder beispielsweise den Median über alle Werte zu bestimmen. Die ausgewählten Werte müssen dann gespeichert werden, um sie für eine richtige Messung zu verwenden.

Die Bestimmung dieser Werte ist auch ohne die Durchführung einer zusätzlichen Messung möglich. In diesem Fall muss der Resonator entweder am Anfang oder am Ende der Messung leer sein. Das ist beispielsweise der Fall, wenn die Messung bereits beginnt, bevor das zu untersuchende Material den Resonator betreten hat. Dabei kann entweder einer der Werte ausgewählt werden, welche gemessen wurden, bevor oder nachdem der Resonator mit einem Messobjekt befüllt war. Für den hier betrachteten Messaufbau werden die Werte der letzten Position verwendet.

4.3.6 Berechnung der finalen Werte

Nachdem für jede Position die Resonanzfrequenz und der Gütefaktor bestimmt wurden, können die final benötigten Werte ΔQf_{rel} und Δf^2_{inv} berechnet werden. Zur Bestimmung wurden die Formeln 2.4 und 2.5 jeweils in den Funktionen $calculate\_df2\_inv$ und $calculate\_dqf\_rel$ implementiert. Diese berechnen für eine übergebene Liste von Resonanzfrequenzen und Gütefaktoren für jeden der Werte der Liste den Ergebniswert. Die Formeln sind unabhängig vom Messaufbau gegeben, jedoch ist es auch in diesem Fall durch den funktionalen Aufbau der Bibliothek einfach sie anders zu bestimmen, da die Ergebnisse der vorherigen Auswertungsschritte unabhängig voneinander sind. Beispielsweise könnten für einen Spezialfall bis zu diesem Schritt in der Auswertung die Funktionen der Bibliothek verwendet werden und dann trotzdem für den Rest eine andere Auswertung durchgeführt werden.

4.3.7 Integration der finalen Werte

Um ein verwertbares Ergebnis zu erhalten muss ein Integral berechnet werden. Wie in Kapitel 2.5 erläutert, kann als Ergebnis entweder das Integral von Δf^2_{inv} oder die

Division aus diesem und dem Integral von ΔQf_{rel} dienen.

Die Durchführung einer Integration wurde in der Funktion *calculate_integral_simple* implementiert. Sie erhält die Positionswerte und die zugehörigen Funktionswerte. Dabei handelt es sich beispielsweise um Δf_{inv}^2. Konkret wurde innerhalb der Funktion die zusammengesetzte Mittelpunktsregel verwendet. Die einfache Mittelpunktsregel approximiert ein Integral folgendermaßen:

$$\int_a^b f(x)dx \approx f\Big(\frac{(a+b)}{2}\Big)(b-a) \tag{4.3}$$

Bei der zusammengesetzten Mittelpunktsregel wird das betrachtete Intervall in beliebig viele Teilintervalle aufgeteilt. Für jedes dieser Intervalle wird die einfache Mittelpunktsregel angewendet und die Ergebnisse werden zusammenaddiert.

In diesem Fall liegt keine Funktion, sondern eine Liste von Wertepaaren vor. Seien die Wertepaare (x_i,y_i) und n die Anzahl an Paaren, dann sieht die Implementation der Regel für diesen Fall wie folgt aus:

$$\sum_{i=0}^{n-1} \frac{y_i + y_{i+1}}{2}(x_{i+1} - x_i) \tag{4.4}$$

Mithilfe der Funktion *calculate_integral_simple* können dann die letzten Werte der Auswertung der Daten bestimmt werden.

4.3.8 Zusammenfassung der Auswertung

In den vorherigen Unterkapiteln wurde die Implementierung der verschiedenen Auswertungsschritte vorgestellt. Somit enthält die Bibliothek viele Funktionen, welche in unterschiedlichsten Szenarien bei der Verwendung verschiedener Resonatoren verwendet werden können. Zudem enthält sie die konkrete Implementierung aller benötigten Schritte, um die Messung an einem Pflanzensamen mit dem in dieser Arbeit beschriebenen Messaufbau auszuwerten. Damit ein Nutzer der Bibliothek, welcher mit diesem Aufbau eine Messung durchführt, die Verwendung jeder einzelnen Funktion für die Auswertung nicht selbst handhaben muss, wurde eine Funktion erstellt, welche alle Auswertungsschritte für diesen Aufbau zusammenfasst. Dabei handelt es sich um die Funktion *evaluate_basic_seed_measurement*. Diese erhält als Parameter alle für die Auswertung benötigten Werte. Zudem besitzt sie viele optionale Übergabeparameter, um die Auswertung gegebenenfalls anzupassen. Beispielsweise kann für die Interpolation eine andere Funktion mit zugehöriger Auswertungs-Funktion übergeben werden (siehe Kapitel 4.3.3) oder eine andere Formel für die Bestimmung der Resonanzfrequenzen und Gütefaktoren (siehe Kapitel 4.3.4). Als Rückgabewert liefert die Funktion die Integralwerte zu ΔQf_{rel}, Δf_{inv}^2 und dessen Division. Um die Auswertung zu debuggen, existiert

ein boolscher Übergabeparamter *debug*. Wenn dieser auf $True$ gesetzt wird, liefert die Funktion zudem eine Instanz der Debug-Klasse als Rückgabewert.

4.4 Debug-Klasse

Zur Überprüfung des Auswertungsprozesses wurde speziell für die Auswertung bei der Messung eines Pflanzensamens eine Debugging-Klasse erstellt. Der Aufbau der Klasse ist in Abbildung 4.12 dargestellt. Im Konstruktor werden alle Daten übergeben, welche während der einzelnen Auswertungsschritte entstehen. Die Klasse enthält demnach alle durch die Auswertung generierten Daten. Eine Instanz der Klasse ist nach der Auswertung einfach zu generieren, da die Übergabeparameter den Rückgabewerten der während der gesamten Auswertung verwendeten Funktion entsprechen.

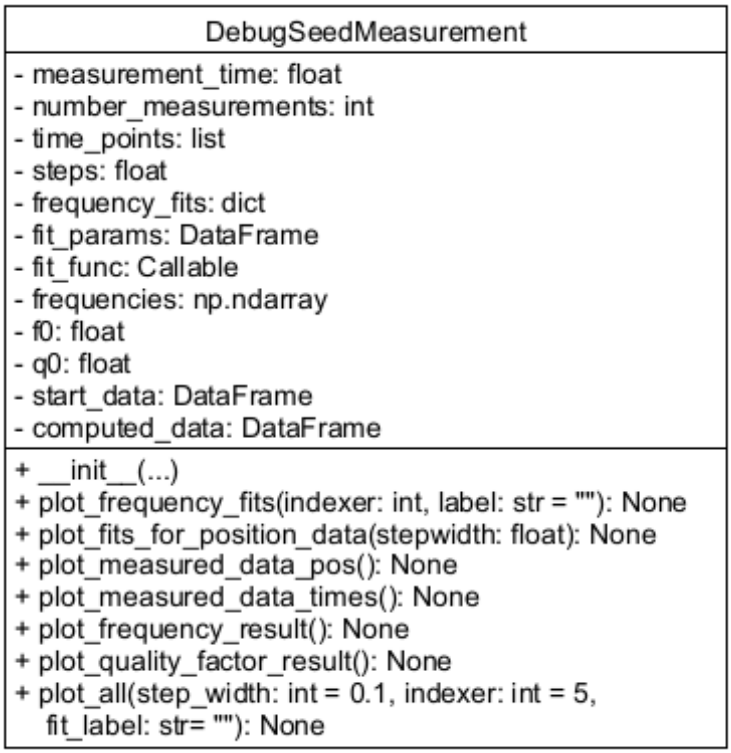


Abbildung 4.12: UML-Darstellung der DebugSeedMeasurement-Klasse

Die berechneten Variablen der Formel 4.2 werden gemeinsam mit den berechneten Werten ΔQf_{rel} und Δf_{inv}^2 und den zugehörigen Positionen in einer CSV-Datei abgespeichert. Jede Zeile der Datei entspricht dann den Werten zu einer Position. Diese Daten werden außerhalb der eigentlichen Bibliothek zusätzlich während des Debuggens abgespeichert,

da sie die Endresultate der Auswertung darstellen. Fehler während des Auswertungsprozesses spiegeln sich in diesen Daten wider und die gesonderte Speicherung vereinfacht eine genaue Betrachtung und ermöglicht einen Vergleich zwischen verschiedenen Messungen.

Um die einzelnen Auswertungsschritte auf Fehler und Ungewöhnlichkeiten zu überprüfen, ist die graphische Betrachtung der Ergebnisse der einzelnen Auswertungsschritte hilfreich. Damit diese Graphen leicht zu generieren sind, bietet die Klasse für jeden zentralen Schritt eine Methode für die Visualisierung dessen. Da alle Daten in der Klasse gespeichert sind, benötigen diese Methoden keine Übergabeparameter. Alle in den vorherigen Unterkapiteln gezeigten Graphen wurden mithilfe dieser Klasse generiert.

Diese Klasse stellt für die Auswertung der Messungen an einem Pflanzensamen eine einfache Möglichkeit dar, die Korrektheit der einzelnen Schritte zu validieren. Vor allem während der Entwicklung neuer und Anpassung alter Auswertungsschritte ist dies sehr hilfreich.

4.5 Verwendung der Bibliothek

Der folgende Codeblock zeigt, wie mithilfe der entwickelten Bibliothek eine Messung an einem Pflanzensamen mit dem in dieser Arbeit vorgestellten Messaufbau durchgeführt und ausgewertet werden kann. Zunächst müssen die benötigten Module und Funktionen der Module importiert werden (Zeile 1-3). Darauf wird eine Instanz der VNA-Klasse mit der zugehörigen IP-Adresse des Geräts initialisiert (Zeile 5). In den Zeilen sieben bis elf werden die für die Messung charakteristischen Werte festgelegt. Dabei handelt es sich um den Frequenzbereich, die Anzahl der Einzelmessungen und die der Messpunkte. Danach werden die Werte an den VNA übermittelt und die Messung wird durchgeführt (Zeile 13-15). Schließlich müssen die gemessenen Daten noch ausgewertet werden. Dies wird in Zeile 17 mit der in Kapitel 4.3.8 vorgestellten Funktion durchgeführt. Somit liegen nach wenigen Codezeilen die Endergebnisse der Messung vor.

```
1  from VNA import VNA
2  from Measurement import read_data_from_file
3  from Evaluate import evaluate_basic_seed_measurement
4
5  analyzer = VNA('TCPIP::192.168.0.100::INSTR')
6
7  start_freq = 5650
8  stop_freq = 5950
9  number_points = 10
10 number_measurements = 50
11
12 VNA_.meassetup(True, number_points_, start_freq_, stop_freq_)
```

```
13 meas_time = do_measurement_seed(analyzer, number_measurements, 'COM3')
14 data = read_measured_data(analyzer, number_measurements)
15
16 df2_inv, dqf_rel, division =  evaluate_basic_seed_measurement(data,
       ↪ analyzer.get_sweep_duration(), meas_time, number_measurements,
       ↪ number_points, start_freq, stop_freq, 0.001)
```

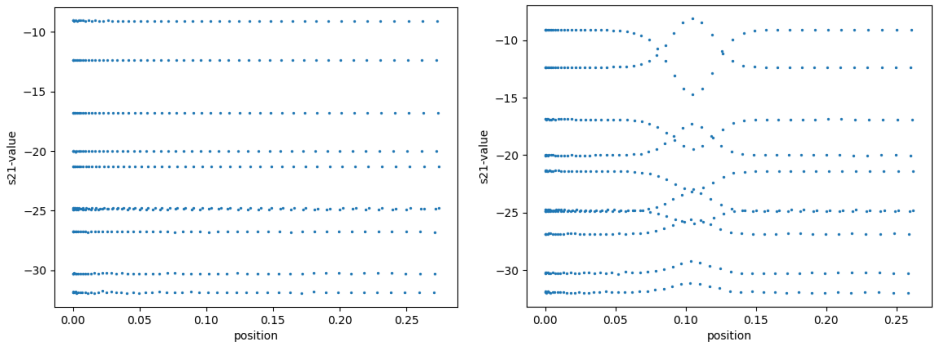
5 Resultate

Mit der zuvor in Kapitel 4 erläuterten Umsetzung ist es möglich eine vollständige Messung an einem Pflanzensamen durchzuführen und auszuwerten. Nach der Auswertung liegen drei Integralwerte vor, welche für die Approximation des Wassergehalts verwendet werden können (siehe Kapitel 2). Um zu überprüfen, ob diese auch den theoretischen Hintergrund korrekt darstellen, werden im Folgenden einige während der Auswertung anfallenden Zwischenergebnisse und die Endresultate in diesem Kontext betrachtet.

5.1 Betrachtung der Zwischenergebnisse

Die während der Auswertung angefallenen Zwischenergebnisse können grafisch betrachtet werden, um sicher zu stellen, dass der Auswertungsprozess und die Messung konzeptuell richtig funktionieren. Dabei hilft ein Vergleich der Zwischenergebnisse einer Messung mit einem leeren Resonator und einer Messung an einem Pflanzensamen.

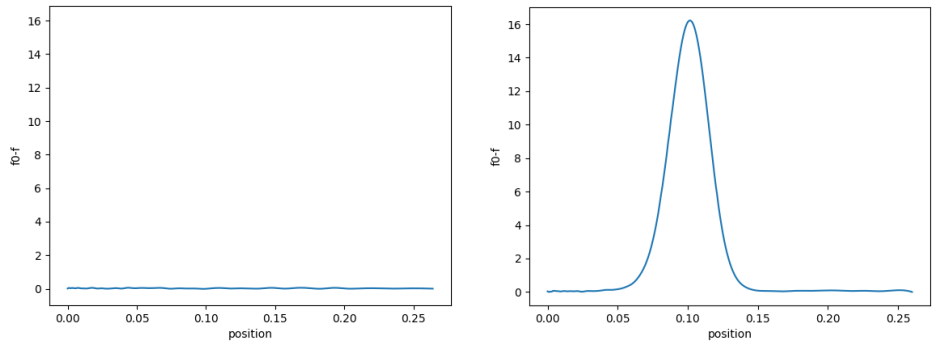
Die grafische Darstellung der gemessenen s_{21}-Werte sollte in Bezug zu den zugehörigen Zeitpunkten oder Positionen im Falle eines leeren Resonators konstant sein. Das bedeutet, dass für die gleiche Messfrequenz alle Messwerte auf einer Linie liegen müssten. Wenn hingegen ein Samen gemessen wird, dann müsste an den Zeitpunkten oder Postionen an denen er den Resonator betreten hat eine Veränderung des Graphen erkennbar sein. Abbildung 5.1 bestätigt dies für die Messung und Auswertung der s_{21}-Werte der entwickelten Bibliothek. Somit ist davon auszugehen, dass die Messung wie vorgesehen funktioniert.



(a) s_{21}-Werte für leeren Resonator (b) s_{21}-Werte für Samen-Messung

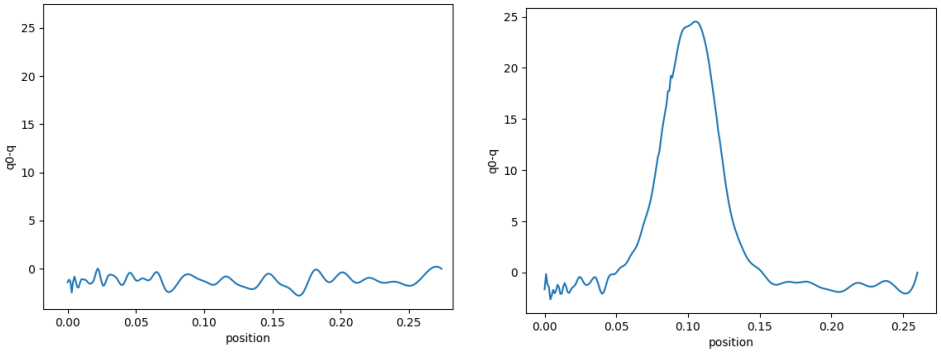
Abbildung 5.1: Vergleich der Messwerte für einen leeren Resonator und für die Messung eines Pflanzensamens bezogen auf die Positionen

Für die Überprüfung des Auswertungsprozesses hilft die Betrachtung der bestimmten Resonanzfrequenzen und Gütefaktoren (siehe Kapitel 4.3.4). Wie in Kapitel 2.4 erläutert, bleiben diese Werte bei der Messung eines leeren Resonators unverändert. Für den Fall der Messung eines Samens variieren diese Werte allerdings, wenn sich der Samen innerhalb des Resonators befindet. Die grafische Darstellung dieser Werte in Bezug auf die Positionen sollten demnach für einen leeren Resonator lediglich eine Gerade darstellen, während für die Messung eines Samens eine sichtbare Veränderung vorhanden sein müsste. Die Abbildungen 5.2 und 5.3 zeigen die Visualisierung der Resonanzfrequenzen und der Gütefaktoren für beide Fälle. Anhand der Grafik ist zu erkennen, dass dieser Fakt durch die implementierte Auswertung bestätigt wurde. Die Visualisierung des Gütefaktors sollte ähnlich aussehen, allerdings fällt auf, dass dieser für einen leeren Resonator eine gewisse Schwankung aufweißt. Diese Schwankungen fallen gering genug aus, um mit dem Wert weiter zu arbeiten, da die Veränderung durch den Samen trotzdem deutlich erkennbar ist.



(a) Resonanzfrequenz für leeren Resonator (b) Resonanzfrequenz für Samen-Messung

Abbildung 5.2: Vergleich der Entwicklungen der Resonanzfrequenz



(a) Gütefaktor für leeren Resonator (b) Gütefaktor für Samen-Messung

Abbildung 5.3: Vergleich der Entwicklungen des Gütefaktors

Aufgrund dieser Beobachtungen ist anzunehmen, dass die Messung und Auswertung auf der Basis des theoretischen Hintergrundes korrekt implementiert wurden. Ob die letztendlichen Ergebnisse der Auswertung allerdings für den gemessenen Samen eine realitätsnahe Approximation des Wassergehalts ermöglichen, muss noch überprüft werden.

5.2 Validierung der Endresultate

Die Messung eines Pflanzensamens löst die auf der Theorie basierenden Veränderungen der Eigenschaften des Resonators aus. Um zu überprüfen, inwiefern die Endresultate der Bibliothek tatsächlich den Wassergehalt des Samens widerspiegeln, wurde ein Testversuch durchgeführt. Zudem kann dieser später auch für die Erstellung der in 2.5 erläuterten Kalibrierung verwendet werden. Es wurden von drei verschiedenen Pflanzenarten insgesamt 16 Samen zunächst gewogen und dann mithilfe des Messaufbaus gemessen. Die auf diese Weise bestimmten Werte wurden abgespeichert. Danach wurden die Samen für fünf Tage in einem Trocknungsofen platziert, um deren Wassergehalt zu verringern. Im Anschluss fanden eine erneute Gewichtsmessung und Messungen mit der Bibliothek statt. Jeder Samen wurde jeweils vor und nach der Trocknung drei mal gemessen.

Während der mehrfachen Messungen eines einzelnen Samens wurde ein Problem ersichtlich. Die berechneten Werte für den selben Samen sind in gewissem Maße abhängig von der Rotation des Samens. Je nachdem wie der Samen durch den Resonator fällt, wird das Ergebnis der Auswertung verändert. Für symmetrische Objekte liefert die Bibliothek bis auf minimale Abweichungen einen konstanten Wert. Dies wurde mithilfe von unterschiedlichen Plastikbällen mit Durchmessern von 2,5 bis 6 mm getestet. Ein Pflanzensamen ist jedoch in den meisten Fällen nicht symmetrisch und dadurch fallen die gemessenen Werte in Abhängigkeit von der Rotation während des Falls unterschiedlich aus. Für den

aktuellen Aufbau ist es nicht möglich dafür zu sorgen, dass bei der mehrfachen Messung eines Samens dieser immer mit der exakt gleichen Rotation durch den Resonator fällt. Dies ist einer der Gründe für die Variation der Werte in den nachfolgenden Abbildungen.

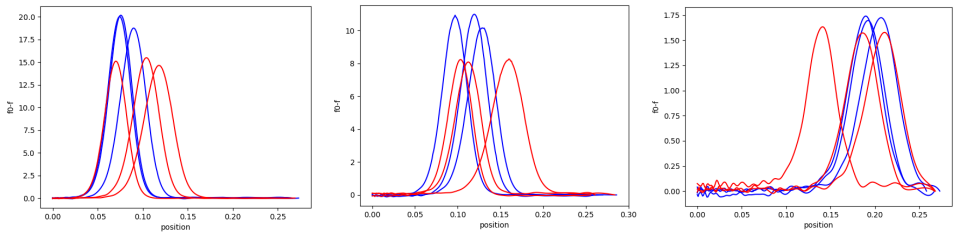
Die nachfolgende Tabelle 5.1 zeigt für drei repräsentative Beispiele die Entwicklung des Gewichts der Samen. Es ist zu erkennen, dass die Samen 1 und 2 relativ viel Wasser verloren haben. Das Gewicht des im Vergleich dazu deutlich leichteren Samen 3 wurde weniger stark reduziert. Durch die Trocknung der Samen sollten sich nun die Messwerte der Samen verändert haben. Der geringere Wasseranteil sollte sich auf die Eigenschaften des Resonators auswirken (siehe Kapitel 2.5).

Samen Nr.	Gewicht vT.	Gewicht nT.	Prozentuale Abnahme
1	0,04683g	0,04326g	7,62%
2	0,02852g	0,02629g	7,81%
3	0,00583g	0,00560g	3,95%

vT.: vor Trocknung, nT.: nach Trocknung
Nr. 1: Weizensamen, Nr. 2.: Buchweizensamen, Nr. 3: Rapssamen

Tabelle 5.1: Vergleich der Gewichte von drei Samen vor und nach der Trocknung

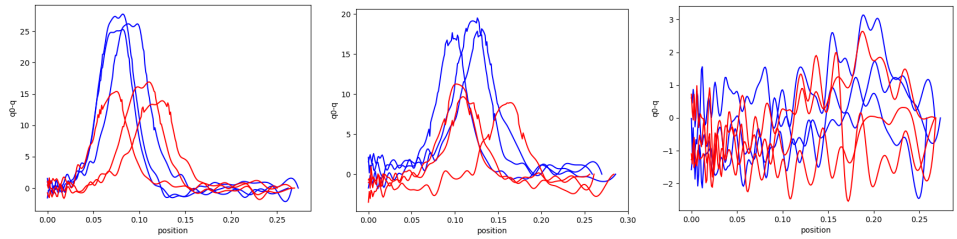
Die Abbildungen 5.4 und 5.5 zeigen die bestimmten Resonanzfrequenzen und Gütefaktoren für die jeweiligen drei Messungen der Samen vor und nach der Trocknung. Für Samen 1 und 2 ist für beide Eigenschaften ein deutlicher Unterschied zu erkennen. Die Kurven sind nach der Trocknung deutlich flacher als noch zuvor. Bei Samen 3 allerdings fallen die Unterschiede für die Resonanzfrequenz sehr gering aus. Es existiert eine Tendenz für eine etwas flachere Kurve für die Werte nach der Trocknung, allerdings ist der Unterschied nur sehr gering. Die Betrachtung des Gütefaktors liefert für Samen 3 keinen Mehrwert. Es ist keine eindeutige Veränderung der Kurve für die Einflussnahme eines Samens erkennbar, sowohl vor als auch nach der Trocknung. Samen 3 scheint zu klein und zu leicht zu sein, um einen Einfluss auf den Gütefaktor des Resonators zu nehmen.



(a) Samen 1 (b) Samen 2 (c) Samen 3

blau: Werte vor Trocknung, rot: Werte nach Trocknung

Abbildung 5.4: Vergleich der gemessenen Resonanzfrequenzen vor und nach der Trocknung



(a) Samen 1 (b) Samen 2 (c) Samen 3

blau: Werte vor Trocknung, rot: Werte nach Trocknung

Abbildung 5.5: Vergleich der gemessenen Gütefaktoren vor und nach der Trocknung

Bei der Betrachtung all dieser Graphen fällt auf, dass die Kurven nicht immer an der selben Stelle ihr Maximum erreichen. Sie wirken verschoben. Dies ist auf eine Verzögerung vor dem Starten der Messung zurück zu führen. Die für die Messung eines Samens in Kapitel 4 erläuterte Umsetzung basiert auf der Bestimmung der Positionen zu den zugehörigen Messpunkten. Für die Bestimmung werden die Zeitpunkte, an denen die Werte gemessen wurden, benötigt. Diese werden in Abhängigkeit von der Dauer der Messung bestimmt. Dabei kommt es zu folgendem Problem: Zunächst wird der Samen durch das Senden eines Befehls an den Ansauger fallen gelassen, darauf wird die aktuelle Zeit gestoppt und dann wird die Messung gestartet. Hier existieren zwei Ungenauigkeiten. Zum einen ist das Senden des Befehls an den Ansauger kein blockierender Aufruf und die benötigte Zeit, bis der Samen tatsächlich fällt, ist nicht konstant. Somit betritt der Samen für jede Messung zu unterschiedlichen Zeitpunkten den Resonator. Zum anderen wird Zeit benötigt, bis das Senden des Befehls für den Start der Messung den VNA erreicht und bis der ausführende Computer das Signal erhält, dass die Messung beendet ist. Diese Zeiten fallen für jede Messung unterschiedlich aus und sind in der durch die Bibliothek bestimmten Dauer der Messung enthalten. Somit ist die Zeit nicht vollständig repräsentativ für die Messdurchführung und die Bestimmung der Zeitpunkte

der einzelnen Messpunkte ist nur eine Approximation und enthält Fehler. Aus diesem Grund sind die Graphen von verschiedenen Messungen aus den Abbildungen 5.4 und 5.5 nicht immer auf derselben Stelle.

Die folgenden drei Tabellen 5.2, 5.3 und 5.4 zeigen die bestimmten Endresultate der Auswertung vor und nach der Trocknung für die drei Messungen jedes Samens. Für den dritten Samen fällt auf, dass $Integral_{\Delta Qf_{rel}}$ nicht nutzbar ist, da es beinah für alle Messungen 0 ist und somit signalisiert, dass sich für diesen Faktor im Vergleich zu einem leeren Resonator beinahe nichts verändert hat. Ansonsten fällt auf, dass die Werte im Gegensatz zu vor der Trocknung kleiner geworden sind. Allerdings liefert die genaue Betrachtung einige Diskrepanzen. Sowohl vor als auch nach der Trocknung ist zu erkennen, dass sich die bestimmten Werte der drei unterschiedlichen Durchläufe für einen Samen an verschiedenen Stellen sehr stark unterscheiden. Bei der Betrachtung von $Integral_{\Delta f_{inv}^2}$ (Tabelle 5.2) für Samen 2 sind beispielsweise die Werte vor der Trocknung alle recht ähnlich. Nach der Trocknung ist für zwei Messungen eine deutliche Abnahme des Wertes zu erkennen, während der dritte Wert größer ist, als der kleinste Wert vor der Trocknung. Diese Diskrepanzen zwischen den Messungen zum selben Zeitpunkt sind auf die zuvor erläuterten Probleme zurückzuführen. Die Rotation des Samens beeinflusst die gemessenen Werte und die unregelmäßigen Verzögerungen beeinflussen die Positionen, für die die Veränderungen der Eigenschaften des Resonators durch den Samen festgestellt wurden. Durch diese Veränderungen können visuell betrachtet die Graphen der Funktionen von ΔQf_{rel} und Δf_{inv}^2 gestreckt oder gestaucht werden. Dies sorgt für eine Veränderung des Integrals dieser Funktionen und somit auch der Endresultate. Durch diese Gründe treten die zuvor erläuterten Sonderfälle auch für die bestimmten Divisionen der Werte auf (Tabelle 5.4).

Samen Nr.	$Integral_{\Delta f_{inv}^2}$ vT.	$Integral_{\Delta f_{inv}^2}$ nT.
1	0,651	0,590
	0,663	0,459
	0,665	0,571
2	0,412	0,324
	0,413	0,284
	0,401	0,407
3	0,091	0,077
	0,086	0,082
	0,077	0,082

Tabelle 5.2: Vergleich von $Integral_{\Delta f_{inv}^2}$ vor und nach der Trocknung

Samen Nr.	$Integral_{\Delta Qf_{rel}}$ vT.	$Integral_{\Delta Qf_{rel}}$ nT.
1	0,069	0,047
	0,078	0,046
	0,094	0,048
2	0,068	0,017
	0,064	0,023
	0,045	0,008
3	-0,002	0,008
	-0,01	-0,009
	0,017	-0,015

Tabelle 5.3: Vergleich von $Integral_{\Delta Qf_{rel}}$ vor und nach der Trocknung

Samen Nr.	Division vT.	Division nT.
1	0,106	0,079
	0,117	0,101
	0,141	0,084
2	0,164	0,053
	0,155	0,081
	0,112	0,019
3	-0,024	0,1
	-0,117	-0,111
	0,221	-0,182

Tabelle 5.4: Vergleich von $\frac{Integral_{\Delta f_{inv}^2}}{Integral_{\Delta Qf_{rel}}}$ vor und nach der Trocknung

Die Trocknung der Samen ist in den Endresultaten der Auswertung zu erkennen, allerdings sorgen Probleme während der Messdurchführung für eine Ungenauigkeit dieser Werte. Dadurch muss weiter untersucht werden, wie die Messung konstanter gestaltet werden kann, um eine Approximation des Wassergehalts eines Pflanzensamens mithilfe einer Kalibrierung auf der Basis der bestimmten Werte zu erzeugen.

6 Fazit

Im Rahmen dieser Bachelorarbeit wurde eine Bibliothek entwickelt, welche in der Lage ist, die Messung eines Pflanzensamens mithilfe eines Hohlraumresonators durchzuführen und die gemessenen Daten auf Basis der dahinter liegenden Theorie auszuwerten. Um die Durchführung einer Messung an einem Pflanzensamen zu vereinfachen wurde eine Klasse für die Verwaltung der Verbindung mit einem VNA erstellt, welcher die eigentliche Messung durchführt. Es stehen fertige Funktionen in einem Modul für die Messausführung bereit, die den Prozess der Messung für den Anwender vereinfachen. Für die Überprüfung der während der Auswertung angefallenen Daten wurde eine Klasse entwickelt. Diese speichert die Daten und bietet die Möglichkeit sie auf Basis der Theorie auf vielfältige Weise zu visualisieren, um mögliche Fehler oder Besonderheiten während der Auswertung aufzudecken.

Die Bibliothek wurde zudem in Hinblick auf die Verwendung für andere Messaufbauten mit Resonatoren konzipiert, sodass sie für die Auswertung der Messdaten vorgefertigte Funktionen bereitstellt, welche zur Vereinfachung verwendet werden können. Um dieses Ziel zu erreichen, wurde die Auswertung nach dem Prinzip der funktionalen Programmierung entwickelt. Diese sorgt dafür, dass mehrere in sich abgeschlossene und unabhängige Routinen entwickelt wurden, welche für andere Messaufbauten auf simple Weise verwendet werden können. Auswertungsschritte, welche auf Basis der Theorie für verschiedene Messaufbauten Anwendung finden, wurden allgemein implementiert, sodass sie für die konkrete Nutzung einfach spezialisiert werden können. Dies macht die in der Bibliothek entwickelte Auswertungsroutine flexibel verwendbar für verschiedene Messaufbauten derselben Art.

Für die Messung eines Pflanzensamens sind diese Spezialisierungen enthalten. Es wurde überprüft und bestätigt, dass die Messungen und die ausgewerteten Daten den zugrunde liegenden Eigenschaften der Theorie entsprechen. Es konnten zudem auch Probleme der Messung und des Messaufbaus entdeckt werden, welche die Genauigkeit der finalen Werte der Auswertung negativ beeinträchtigen. Nichtsdestotrotz ist die Bibliothek in der Lage, den Einfluss eines Pflanzensamens auf ein sich innerhalb des Resonators befindendes elektrisches Feld festzustellen. Die Veränderungen der darauf basierenden spezifischen elektrischen Eigenschaften des Resonators können bestimmt werden, sodass mit diesen ein Rückschluss auf den Wassergehalt des Samens theoretisch möglich ist.

7 Ausblick

In der Zukunft wird die Bibliothek hinsichtlich der Messung eines Pflanzensamens weiter angepasst. Die konkrete Bestimmung des Wassergehalts eines Samens wird noch hinzugefügt. Durch die im Rahmen der Überprüfung der Resultate entdeckten Probleme der Messdurchführung muss überdacht werden, wie man auf Basis der bestimmten Werte während der Auswertung auf den Wassergehalt schließen kann. Dafür gibt es mehrere Möglichkeiten. Um an der Erstellung einer Kalibrierung auf der Basis der bestimmten Werte festzuhalten, muss zunächst das Problem der unterschiedlich langen Verzögerungen vor und nach der Messung gelöst werden. Die Genauigkeit der Endresultate muss erhöht werden. Wenn dies gegeben ist, können sie verwendet werden, um eine Kalibrierung für die Approximation des Wassergehalts zu erstellen. Des Weiteren wäre die Vermeidung dieser Probleme durch die Bestimmung anderer Endresultate möglich, welche nicht abhängig von der Berechnung eines Integrals sind. Dafür wäre beispielsweise die Verwendung des maximalen Werts der bestimmten Resonanzfrequenzen und Gütefaktoren möglich.

Zudem steht die Verwendung der Bibliothek für andere Messaufbauten aus. Es wird untersucht werden, inwiefern die Bibliothek tatsächlich nutzbar für andere Resonatoren ist. Die spezifischen Auswertungsschritte für diese Aufbauten können der Bibliothek hinzugefügt werden. Im optimalen Fall entsteht dadurch eine Bibliothek, welche für die Nutzung aller Resonatoren innerhalb des Instituts verwendet werden kann.

Literatur

[1] *Pflanzenwissenschaften (IBG-2)*. 10.09.2023. URL: https://www.fz-juelich.de/de/ibg/ibg-2.

[2] Viktor A. Sydoruk u. a. „Design and Characterization of Microwave Cavity Resonators for Noninvasive Monitoring of Plant Water Distribution“. In: *IEEE Transactions on Microwave Theory and Techniques* 64.9 (2016), S. 2894–2904. ISSN: 0018-9480. DOI: 10.1109/TMTT.2016.2594218.

[3] Rainer Herrmann u. a. „Feuchtemessung mit Mikrowellen- Resonatoren Materialfeuchtemessung mittels Time-Domain Reflectometry“. In: *teme* 64.JG (1997), S. 447–452. ISSN: 0171-8096. DOI: 10.1524/teme.1997.64.jg.447.

[4] balston. *Agilent Basics of Measuring the Dielectric Properties of Materials*. URL: https://academy.cba.mit.edu/classes/input_devices/meas.pdf.

[5] Christian Gerthsen und Helmut Vogel. *Physik: Ein Lehrbuch zum Gebrauch neben Vorlesungen ; mit 56 Tabellen und über 1150 Aufgaben*. 17., verb. und erw. Aufl. Springer-Lehrbuch. Berlin u. a.: Springer, 1993. ISBN: 3540566384.

[6] Wikipedia, the free encyclopedia. *Dielectric responses*. [zuletzt aufgerufen: 13.09.2023]. 2008. URL: https://en.wikipedia.org/wiki/File:Dielectric_responses.svg.

[7] Rhode & Schwarz. *Fundamentals of Vector Network Analysis Primer*. URL: https://www.rohde-schwarz.com/de/produkte/messtechnik/analyzers/netzwerkanalysatoren/fundamentals-of-vector-network-analysis-primer_253352.html.

[8] Marion I. Menzel u. a. „Non-invasive determination of plant biomass with microwave resonators“. In: *Plant, cell environment* 32.4 (2009), S. 368–379. DOI: 10.1111/j.1365-3040.2009.01931.x.

[9] Viktor Sydoruk. *Study of Spectra with Low-Quality Resonance Peaks*. 2019. URL: https://juser.fz-juelich.de/record/867694.

[10] V. N. Skresanov, V. V. Glamazdin und N. T. Cherpak. „The novel approach to coupled mode parameters recovery from microwave resonator amplitude-frequency response“. In: *2011 41st European Microwave Conference*. 2011, S. 826–829. DOI: 10.23919/EuMC.2011.6101922.

[11] Gernot Starke. *Effektive Softwarearchitekturen: Ein praktischer Leitfaden*. 9., überarbeitete Auflage. München: Carl Hanser Verlag GmbH & Co. KG, 2020. ISBN: 9783446465893.

[12] A. L. Ambler, M. M. Burnett und B. A. Zimmerman. „Operational versus definitional: a perspective on programming paradigms“. In: *Computer* 25.9 (1992), S. 28–43. ISSN: 0018-9162. DOI: 10.1109/2.156380.

[13] Dino Alic, Samir Omanovic und Vaidas Giedrimas. „Comparative analysis of functional and object-oriented programming“. In: *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2016, S. 667–672. ISBN: 978-953-233-086-1. DOI: 10.1109/MIPRO.2016.7522224.

[14] Python documentation. *Functional Programming HOWTO*. 18.07.2023. URL: https://docs.python.org/3/howto/functional.html.

[15] Maurizio Gabbrielli. *Programming Languages: Principles and Paradigms*. Undergraduate Topics in Computer Science. London: Springer London, 2010. ISBN: 9781848829145. DOI: 10.1007/978-1-84882-914-5.

[16] Bhanu Prasad Pokkunuri. „Object Oriented Programming“. In: *ACM SIGPLAN Notices* 24.11 (1989), S. 96–101. DOI: 10.1145/71605.71612. URL: https://doi.org/10.1145/71605.71612.

[17] J. Hughes. „Why Functional Programming Matters“. In: *Computer Journal* 32.2 (1989), S. 98–107.

[18] Steven F. Lott. *Functional Python programming: Discover the power of functional programming, generator functions, lazy evaluation, the built-in itertools library, and monads*. 2. Aufl. Birmingham: Packt Publishing, 2018. ISBN: 9781788621854. URL: https://www.wiso-net.de/document/PKEB__9781788621854408.

[19] gleissne. *Remote Operation of R & S Instruments Using Remote Desktop Connection via LAN*. URL: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/dl_application/00aps_undefined/RAC-0703-0029.pdf.

[20] *Welcome to the RsInstrument Python Step-by-step Guide — RsInstrument Python 1.54.0 documentation*. 3.07.2023. URL: https://rsinstrument.readthedocs.io/en/latest/StepByStepGuide.html.

[21] *What Is VISA?* 28.08.2023. URL: https://www.tek.com/en/support/faqs/what-visa.

[22] Rhode & Schwarz. *R&S ZNB/ZNBT Vector Network Analyzers User Manual*. Verfügbar unter: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/pdm/cl_manuals/user_manual/1173_9163_01/ZNB_ZNBT_UserManual_en_64.pdf.

[23] *pySerial — pySerial 3.4 documentation*. 17.08.2023. URL: https://pyserial.readthedocs.io/en/latest/pyserial.html.

[24] *scipy.interpolate.interp1d — SciPy v1.11.2 Manual*. 18.08.2023. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.interp1d.html#scipy.interpolate.interp1d.

[25] *scipy.interpolate.splrep — SciPy v1.11.2 Manual*. 18.08.2023. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.splrep.html#scipy.interpolate.splrep.

[26] Stuart Murphy. *Quadratic Spline Interpolation*. The Art of Polynomial Interpolation. 2022. URL: https://psu.pb.unizin.org/polynomialinterpretation/chapter/chapter-three-quadratic-spline-interpolation/.

[27] Jaan Kiusalaas. *Numerical methods in engineering with Python 3*. 3rd ed. Cambridge: Cambridge University Press, 2013. ISBN: 9781139613149. URL: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=1099957.

[28] *scipy.optimize.curve_fit — SciPy v1.11.2 Manual*. 18.08.2023. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html.