Artem Moskalew
Jonas Christ
Maximilian Nolte

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Inhaltsverzeichnis

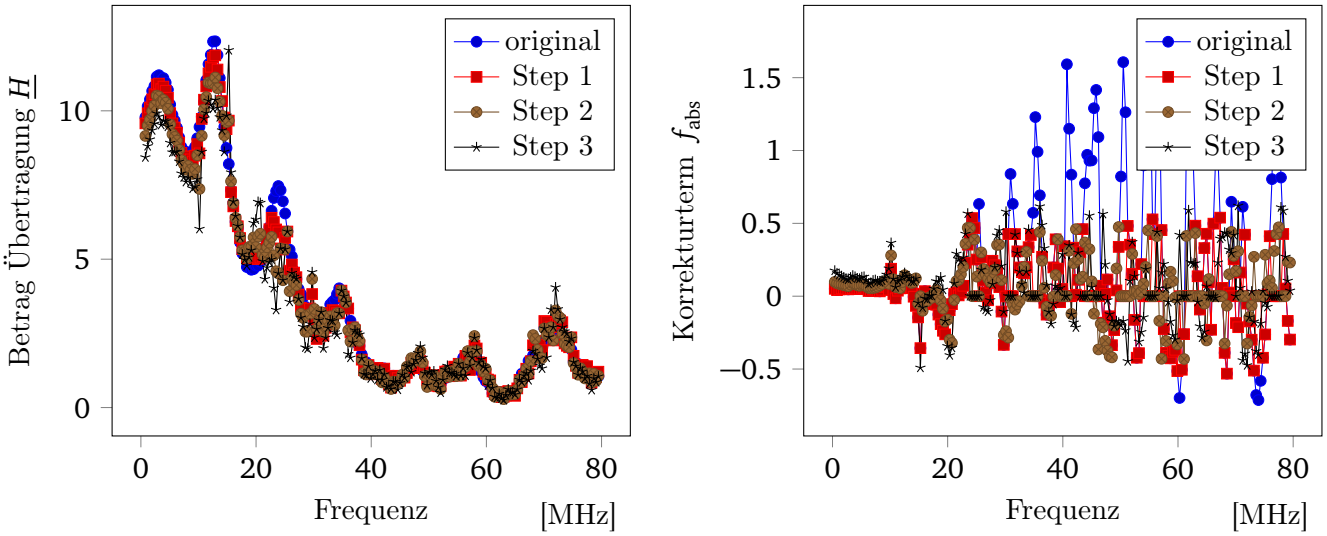
1 Test

1.1 —- firstName —-

Diese Zeile teste das paket *nameref* mit Test und hier mit einer PHANTOMSECTION via ??—

Dies ist ein Versuch, ein Frequenzspektrum zu plotten und Marker an relevanten Punkten zu setzen: Hiermit soll die Refernz in Sub-figures versucht werden: Dies hier soll zu 6.1 dem Hauptbild referenzieren. Dies hier soll zu 6.2a dem Betragsspektrum referenzieren.
Dies hier soll zu 6.2b dem Phasen referenzieren.

Abbildung 1.1: Entwicklung von Übertragungsfunktion und Korrekturterm bei Beschränkung von f_{abs}



(a) – name —

(b) — name –

2 Abstract

— dieses chapter führt eine kurze Vorstellung unseres Projektseminars aus. Beinhaltet einen kurz-Überblick über wichtigste Ergebnisse und Erkenntnisse, sollte (m. E.) wenig auf den Ausblick eingehen —

3 Einführung

Mit den Barrier Bucket (BB) RF Systemen können am neu entstehenden Synchrotron SIS100 oder im Experimentier Speicherring (ESR) am GSI Helmholtzzentrum viele longitudinale Manipulationen am Teilchenstrahl vorgenommen werden. Der dazu notwendige Spannungspuls hat die Form wie in Abb. (3.1) dargestellt. Wenn die Wiederholfrequenz des Spannungspulses gleich der Umlauffrequenz ist, so wird im Phasenraum eine stationäre Potential Barriere erstellt. Wenn die Wiederholfrequenz nicht gleich der Umlauffrequenz ist verschiebt sich die Potential Barriere im Phasenraum und es entstehen Bunches mit unterschiedlicher Länge.
Der Anspruch an diese Systeme liegt darin eine hohe Qualität des Impulses am Gap der Kavität zu erzeugen, damit sogenannte Microbunches unerwünscht entstehen, deshalb müssen die Nachschwinger nach dem Einzelsinus kleiner als 2,5% von $\hat{U}_{BB}$ sein.

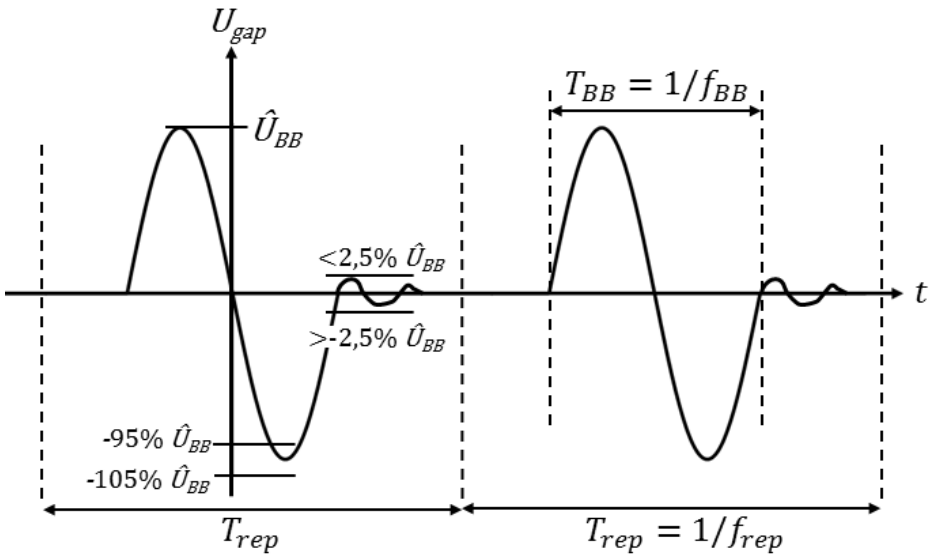


Abbildung 3.1: Ausgangssignal

Dabei benutzen wir im Folgenden f_{rep} für die Wiederholfrequenz des Spannungssignals und die Barrier Bucket Frequenz f_{BB} bestimmt die Breite der Potential Barriere. Für unsere Messungen haben wir $f_{rep} = 900\text{kHz}$ und $f_{BB} = 5\text{MHz}$ verwendet.

3.1 Versuchsaufbau und Modell

Der Prototyp des ESR BB besteht aus einem Funktionsgenerator (Keysight 3600A series 2-channel AWG), einem Verstärker (AR1000A225) und der Breitband Ringkern Kavität. Der Versuchsaufbau ist in Abb. (3.2) gezeigt.
Dabei kann angenommen werden, dass sich das System bis $\hat{U}_{BB} = 550V$ annähernd linear verhält und durch die Übertragungsfunktion $\underline{H}$ beschrieben werden kann. Eine mathematische Modellierung ist ebenfalls in Abb. (3.2) gegeben, bei der zur linearen Übertragungsfunktion $\underline{H}$ noch eine nichtlineare Kennlinie enthalten ist. Die Größen werden im nächsten Abschnitt erklärt.

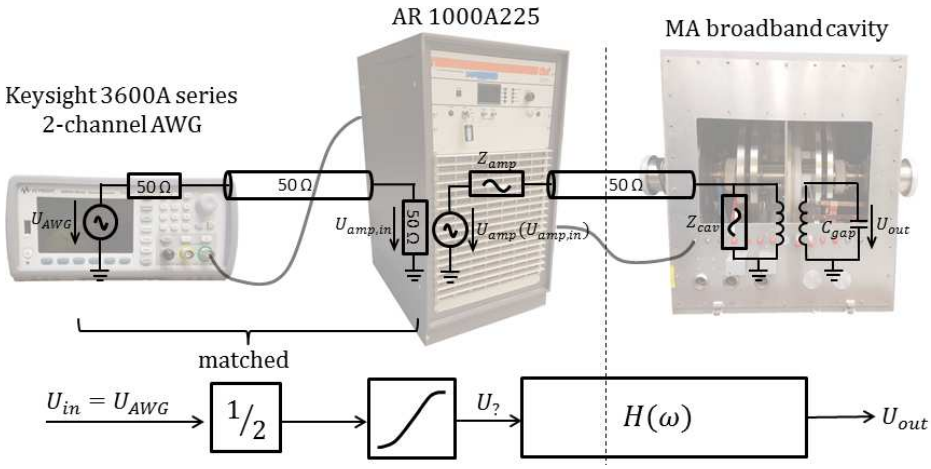


Abbildung 3.2: Versuchsaufbau und Modell

3.1.1 Hammerstein Modell

In Abb. (3.3) sind alle von uns verwendeten Größen dargestellt. U_{in} ist die Eingangsspannung vom Funktionsgenerator. Der erste Block in Abb. (3.3) stellt die nichtlineare Modellierung dar. Die Spannung $U_?$ ist eine rein virtuelle Größe und kann wie in 3.1 über die Koeffizienten a_n berechnet werden. Bei der Potenzreihe hatte $N = 3$ in der MATLAB Implementierung schon gute Ergebnisse geliefert und wurde von uns auch weiterhin so verwendet. Die nichtlinearen Kennlinie im Folgenden nur noch als K bezeichnet wird als Look-Up Tabelle in unserem Programm hinterlegt. Die Spannung U_{out} ist die Gap Spannung in der Kavität.

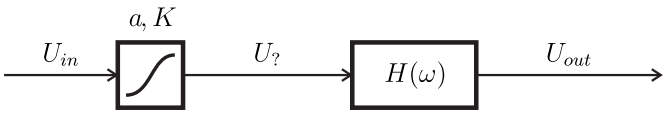


Abbildung 3.3: Hammerstein Modell

$$U_?(t) = \sum_{n=1}^{N} a_n \left[U_{in}(t)\right]^n \qquad U_{out}(t) = \mathscr{F}^{-1}\left\{\underline{H}(\omega) \cdot \underline{U}_{in}(\omega)\right\} \tag{3.1}$$

3.2 Motivation

Dieses Modell wurde bis auf die Berechnung von K schon erfolgreich in Python implementiert, dabei sei auf das Projektseminar [2] verwiesen. Deshalb lag unsere Motivation darin dieses Modell in Python zu vervollständigen, um dann eine möglichen Optimierungsansatz aufzustellen.

3.3 Aufgabenstellung

Im Rahmen unseres Projektseminar sollte eine iterative Optimierung der Vorverzerrung auf Basis einer Hammersteinmodellierung in Python implementiert werden. Dabei sollte das Tool die nichtlineare Kennlinie K und die Übertragungsfunktion $\underline{H}$ wechselseitig optimieren, wobei eine optimale Gewichtung der Parameter zur Optimierung noch nicht erreicht werden musste.

4 Vorgehen

Was war gegeben? Auf Basis von Matlab weiter entwickelt und warum wir dieses Design gewählt haben —- In diesem Kapitel soll die Übernahme des Matlab-Codes, die Dokumentation (samt Konventionen und Dokumentationskonzept) und die Gerätekommunikation dargestellt werden Ggf. ist es sinnvoll, Teile davon in eigene Chapter auszulagern oder den Titel anzupassen—

4.1 —- Start und bereits vorhanden —-

— in dieser section wird auf die anfangs gegebene Code-Struktur, die vorgegebenen Methoden und die parallelen Strukturen zwischen matlab und python eingegangen und ggf. die projektarbeit von Denys und Armin in Hinblick auf konkretere Aspekte referenziert — — Ziel des Abschnitts: Leser hat (gute) Vorstellung, mit welchen Voraussetzungen, welchen Daten- und Programmstrukturen wir gestartet sind und auf welchen wir aufbauen—

4.2 — Konventionen / Dokumentation —-

— in dieser section wird unser Konzept der Dokumentation in Code wie auch in Dokumenten erläutert und ggf. auf die Sammlung von Handbüchern eingegangen — — es könnte sinnvoll sein, diese section nach der folgenden erst zu setzen —

4.3 Gerätekommunikation

Um Messergebnisse sinnvoll für die Verarbeitung in Python aufzunehmen, ist eine effiziente Kommunikation zwischen den Messgeräten und dem angeschlossenen Computer essentiell. Besonders relevant ist hier die zeitliche Abstimmung zwischen Computer und Gerät sowie die bestmögliche Nutzung der Genauigkeiten der verwendeten Geräte. Hierbei wurden die zu Beginn vorliegenden Implementierungen für die Gerätekommunikation erweitert, wobei vor allem

1. die Optimierung der Laufzeit beim Schreiben und Lesen von Gerätewerten sowie
2. die Anpassung der Darstellung des Oszilloskops an die gemessenen Daten für eine höhere Genauigkeit

Ziele der Anpassungen waren.

Die Fernsteuerung von Messgeräten, der `Remote`-Modus, wird standardmäßig durch das Visa-Protokoll und im vorliegenden Fall durch die Implementierung von National Instruments, NIVisa, geregelt. Die Einbindung in Python wird durch die Erweiterung PyVisa ermöglicht. Übertragen werden Befehle in Form standardisierter Befehlsstrukturen, den SCPI (Standard Commands for Programmable Instruments). Dem inneren Aufbau von Messgeräten zugrunde liegt der IEE-488 Standard, sodass unabhängig vom konkreten Gerät oder der Art der Verbindung eine Reihe von Befehlen existiert, die gemäß IEE-488.2 gebräuchliche Standards bieten [5, S. 224 ff.]. Weiterhin bieten die Geräte spezifische Befehle zur Manipulation der Einstellungen, die in großer Analogie zur direkten Benutzeroberfläche formuliert sind. Diese werden nach Kontext gegliedert in sogenannte Subsysteme oder Command Groups.

(a) Knotenliste

$P_\#$	x	y
1	$\cos(72°)$	$\sin(72°)$
2	1	0
3	$\cos(72°)$	$-\sin(72°)$
4	$-\cos(36°)$	$-\sin(36°)$
5	$-\cos(36°)$	$\sin(36°)$
6	0	0

Tabelle 4.1: Laufzeit unterschiedlicher Befehle am Keysight 33622A

Laufzeitoptimierung

Im vorliegenden Messaufbau ist es notwendig, beim Konfigurieren des AWG alle Einstellungen zurückzusetzen und anschließend neu zu setzen. Dabei werden Befehle nach dem First-Come-First-Serve Prinzip aus dem internen Speicher abgearbeitet. Dies führt aufgrund begrenzter Speicherkapazität und Verarbeitungsgeschwindigkeit bei einer Reihe von Befehlen zur Notwendigkeit, das Senden von Befehlen im Programm mit dem Arbeitsstatus des Gerätes zu synchronisieren. Einen Eindruck der zeitlichen Größenordnung bietet Tab. (4.1).

In der zu Beginn vorliegenden Implementierung lagen pauschale Wartezeiten nach einigen Befehlen sowohl in der Ansteuerung des AWG als auch des Oszilloskops vor.

Die Wartezeiten für das AWG wurde letztendlich mit ———————————————— ———— ———————————————— angepasst. Die Laufzeit der Routine zum Schreiben eines Arbiträrsignals in der speziellen hier geforderten Form wurde damit von etwa 25 s auf ungefähr — reduziert.
Erfolglos blieben Versuche mit einigen anderen Standardbefehlen. `*WAI` und `*BUSY?` sind keine für das AWG definierten Befehle. `*OPC?` zwingt Python zum Warten auf Beenden aller Befehle im AWG und führt zu einem `TimeoutError`, da die Ausführung zu lange für die internen Routinen von Python braucht, vergleiche hierfür [7, S. 9]. Weiterhin wurde die pauschale Wartezeit an der im Datenblatt [6, S. 21] für die Verbindung via USB angegebenen, gemessenen Ladedauer eines Arbiträrsignals orientiert. Dieser Richtwert von 1.25 s stellte sich jedoch als zu gering für den hier vorliegenden Fall heraus.

Es wurde festgestellt, das die Verarbeitungsgeschwindigkeit des Oszilloskops bei den hier benötigten Abfragen und Befehlen kein Hindernis darstellt und die Ausführung der Befehle in Python ohne Zeitverzögerung möglich ist. Durch diese Feststellung konnte die Laufzeit je Lese-Aufruf bereits um ungefähr 15 s reduziert werden. Bei den zur Sicherheit implementierten Warte-Befehlen handelt es sich um Status-Abfragen ähnlich den oben für das AWG erläuterten.

Letztlich sei erwähnt, dass die Abfrage der Fehler im AWG mittels `SYSTem:ERRor?` möglich ist und durch das dabei erfolgte Löschen der Fehler aus dem internen Speicher nicht nur die Abfrage sondern auch das Rücksetzen des Fehlerstatus ermöglicht. Dies ist insbesondere für das Debugging vorteilhaft, wenn Befehle ausprobiert werden und die gegebenenfalls auftretenden Fehler gehandhabt werden müssen [5, S. 454]. Dies vermeidet das unter Umständen mehrmalige Trennen und Wiederherstellen der Verbindung zum AWG sowie das Löschen der Fehler an der Benutzeroberfläche.

Anpassung der Auflösung am Oszilloskop

Um die Auflösung des Oszilloskops ausnutzen zu können, ist die Anpassung der horizontalen wie auch vertikalen Auflösung an das zu messende Signal notwendig.

4.3.1 — offene Punkte—

– hier könnten punkte wie etwa die Kommunikation mit dem neuen Oszi oder die evtl. Möglichkeit, Geräte künftig eher als Klassen zu implrementieren, in denen die Visa-Commands gehandhabt werden —

5 Code-Design

—- In diesem Kapitel wird auf das neue Code-Design (Refactoring), das Konzept und die Motivation im Kontext eines Mess-Konzepts eingegangen ggf. auch auf Versionenverfolgung Git?—

5.1 —- Motivation / Begründung —-

— hier kann auf die Sinnhaftigkeit unseres Code-Designs (rückwärts gedacht ;)) eingegangen werden: Welche Ideen finden sich nachher wieder? Warum also wurde die folgende Umsetzung gewählt? Ggf. auch kurzes Eingehen auf Entwicklung des Designs hier??? Keine Ausführlichen Erklärungen, eher allgemeine Ideen und Aspekte des Software-Engineering ausführen. Konkrete Punkte nachfolgend erst —

5.2 — Aspekte des Desings —-

— in dieser section werden die einzelnen Punkte weiter ausgeführt. Aus Gründen der Dokument-Hierarchie erscheint es sinnvoller, die einzelnen Punkte in sub-sections zu setzen. Hier also nur eine Kurze Zusammenfassung? —

5.2.1 — Struktur – ?

— hier kann ein kleiner Überblick über die Programmstruktur gegeben werden, also die Aufteilung nach Ordnern (routinen und Funktionalität) und Nomenklatur??? Unbedingt auf Redundanz mit Motivation prüfen!!! Insb. Einführung von Klassen und Helpers-System erläutern —

5.2.2 — TDD und Mock-System —

— Ausführen des Test-Driven-Developments und des Mock-Systems. Wie wurde implementiert?—

5.2.3 Offene Punkte

— hier ist m.E. eine Ausführung noch offener Aspekte angebracht, etwa: Typprüfung bei Übergaben ausweiten (! und an RF-Tool Konventionen anpassen?), Test für falsche übergabe-Parameter noch als Möglichkeit nennen, Erweiterbarkeit des Mock-Systems erklären —

6 Iterative Optimierung des Hammerstein-Modells

—- In diesem Kapitel werden die Aspekte der durchgeführten Optimierungs-Algorithmen erläutert — Ziel der Optimierung von Übertragungsfunktion $\underline{H}$ und Kennlinie K mit ihren Parametern a ist die Minimierung des Fehlers zwischen idealem und gemessenem Ausgangssignal, $U_{out,\mathrm{id}}$ und $U_{out,\mathrm{meas}}$. Die Minimierung des relativen Fehlers ist also gegeben durch

$$\min f(t) = \min\left(\frac{U_{out,\mathrm{meas}} - U_{out,\mathrm{id}}}{U_{out,\mathrm{id}}}\right) = \min\left(\frac{U_{out,\mathrm{meas}}}{U_{out,\mathrm{id}}} - 1\right). \tag{6.1}$$

Für das verwendete Hammerstein-Modell liegt die in [?] vorgeschlagene getrennte, iterative Optimierung von $\underline{H}$ und K nahe. Die Auswertung der Qualität des Einzelsinus erfolgt dabei durch das RF-Tool von — Zitat RF-Tool — mit Entwicklungsstand vom — — unter Verwendung des als `QGesamt1` geführten Qualitätswerts [1].

6.1 Optimierung der linearen Übertragungsfunktion H

— evaluate-Aufruf, Schleife, Speicher? Laufzeit? . insbesondere gemessene Daten, ohne jedwede Anpassung /Limitierung der Faktoren,kann auf die Einbindung des neuen Qualitäts-Tools eingegangen werden —

Die Optimierung von $\underline{H}(\omega)$ beruht auf der Annahme, dass sich (6.1) auf die Betragsspektren des berechneten und des gemessenen Ausgangssignals, $\underline{U}_{out,\mathrm{id}}(\omega)$ und $\underline{U}_{out,\mathrm{meas}}(\omega)$ fortsetzen lässt mit

$$f_{\mathrm{abs}}(\omega) := \frac{\mathrm{abs}\left(\underline{U}_{out,\mathrm{meas}}(\omega)\right)}{\mathrm{abs}\left(\underline{U}_{out,\mathrm{id}}(\omega)\right)} - 1\,. \tag{6.2}$$

Ist im Betragsspektrum des gemessenen Signals eine Frequenz mit halbem Betrag verglichen mit dem idealen Signal vertreten, wird dies entsprechend der Linearität der Übertragunsfunktion dahingehend gedeutet, dass die Verstärkung von $\underline{H}$ bei dieser Frequenz um einen Faktor 2 zu gering ist. Iterativ mit einer Schrittweite σ_H ausgeführt, folgt für den i-ten Schritt

$$\mathrm{abs}\left(\underline{H}^{i+1}\right) = \mathrm{abs}\left(\underline{H}^{i}\right) \cdot \left(1 - \sigma_H^i \, f_{\mathrm{abs}}^i\right) \tag{6.3}$$

für $\sigma_H^i \in [0,1]$ und $\underline{U}_{out,\mathrm{meas}}^i$ in f_{abs}^i als gemessenem Ausgangssignal für das mit $\underline{H}^i$ berechnete Eingangssignal [2]. Würde allerdings (6.3) mit komplexen Zahlen und nicht allein den Beträgen ausgeführt, würde auch die Phase der -1 beachtet und folglich die durch σ_H skalierte komplexe Zahl wesentlich verändert. Also muss für das Phasenspektrum eine andere Optimierung erfolgen. Eine Möglichkeit hierfür wäre die simple Anpassung der Phase $\arg\left(\underline{H}\right) = \varphi_H$ mit

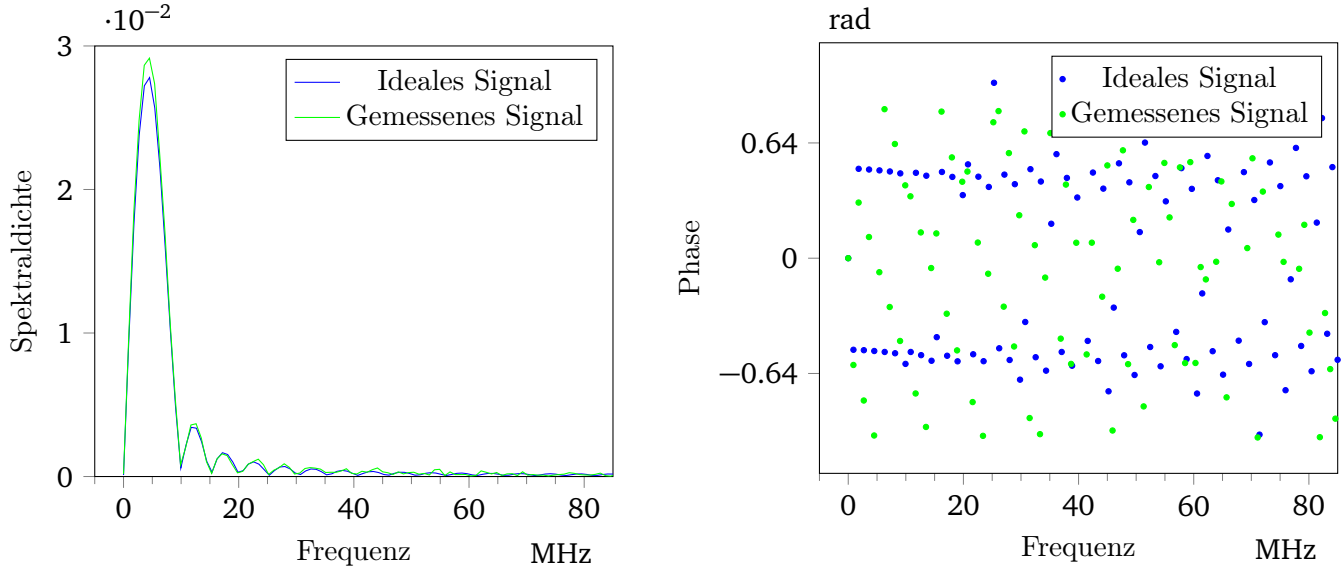
$$\varphi_H^{i+1} = \varphi_H^i - \sigma_\varphi^i \left(\arg\left(\underline{U}_{out,\mathrm{meas}}\right) - \arg\left(\underline{U}_{out,\mathrm{id}}\right) \right) \tag{6.4}$$

[1] Hierauf beziehen sich alle weiteren Angaben zur Qualität des Signals. Eine intensivere Befassung mit dem Tool hat nicht stattgefunden.

[2] Nachfolgend wird aus Gründen der Übersichtlichkeit f_{abs} statt dem länglichen Bruch genutzt

mit $\sigma_{\varphi}^{i} \in [0,1]$. Diese Anpassung der Phase wurde jedoch nur kurzen Tests unterzogen und anschließend nicht weiter verfolgt. Es hat sich die Signalform des Ausgangssignals unproportional stärker verändert, als dies nur im Falle der Betrags-Anpassung der Fall war. Vermutlich liegt dies an dem aus dem Ausgangssignal gewonnenen Phasengang, der in wesentlich größerem Maße vom idealen Phasengang abweicht als im Betragsspektrum. In Abb. (6.1) sind Betrag und Phase der durch FFT erhaltenen Spektren für gemessenes und ideales Ausgangssignal vor Durchführung einer Optimierung dargestellt. Insbesondere illustriert Abb. (6.2b) die bei gemessenem Signal auftretende Streuung der Phase.

Abbildung 6.1: Spektrum des Einzelsinus-Signals, berechnet und gemessen mit je 109 Punkten



(a) Betragsspektren

(b) Phasenspektren

Eine Aufstellung der rein auf (6.3) beruhenden Anpassung der Übertragungsfunktion über mehrere Iterationsschritte findet sich in — Abb —. Neben den für kontinuierliche Funktionen problemlos definierbaren iterativen Zuweisungen ergeben sich in Messung und diskreter Ausführung jedoch Fehlerquellen. Problematisch sind insbesondere solche, die in (6.3) durch das Betragsverhältnis der Ausgangssignale verstärkt werden. Unterscheiden sich die Spektren hier um einen großen Faktor, resultiert dies in einer großen Anpassung der Übertragungsfunktion für die betreffende Frequenz. Dies ist folglich insbesondere bei kleinen Beträgen der Spektren problematisch, wenn Ungenauigkeiten und Störeinflüsse betrachtet werden.
Besondere Störeinflüsse ergeben sich also durch

- Rauschen: Weißes Rauschen macht sich in allen Frequenzen bemerkbar mit kritischem Einfluss bei geringer Spektraldichte des Signals.
- Diskretisierungsfehler: Die FFT bedingt eine begrenzte Auflösung, in den Spektren von $\underline{H}$ und den gemessenen Signalen und liegt insbesondere im Allgemeinen an unterschiedlichen Frequenzen und mit unterschiedlich vielen Punkten vor.
- Interpolationsfehler: Die (hier lineare) Interpolation der Spektren zur Auswertung von f_{abs} an den Frequenzen von $\underline{H}$ kann insbesondere den Einfluss oben genannter Punkte verstärken.

Ignorieren kleiner Beträge im Spektrum

Um Rauscheinflüsse und Probleme durch Nulldurchgänge zu dämpfen, wurde einer erster intuitiver Ansatz vorgenommen: Bei den Betragsspektren der in f_{abs} eingehenden Signale, des gemessenen und idea-

lisierten Spannungssignals, wurden alle Anteile, die verglichen mit dem Maximalwert des betreffenden Spektrums besonders klein sind, auf einen vorgegebenen Wert, im Folgenden Default-Wert genannt, gesetzt. Dies führt an den betroffenen Frequenzen zu $f_{\text{abs}} = 0$ und damit keiner Änderung von $\underline{H}$. Dies bedeutet also, dass alle Einträge des Betragsspektrums von $U_{out,\text{ideal}}$ mit weniger als zum Beispiel $5\,‰$ der maximalen Amplitude auf den Default-Wert gesetzt werden. Insbesondere werden auch die Einträge an den Frequenzen zurückgesetzt, die im Spektrum von $U_{out,\text{meas}}$ klein gegen das zugehörige Maximum sind.
Zu beachten bei letzterem Punkt ist die notwendige Rundung, wenn die Einträge der FFT an unterschiedlichen Frequenzen vorliegen.

Mit Beschränkung auf $5\,‰$ und dem globalen Minimum beider Spektren als Default-Wert ergeben sich die angepassten Betragsspektren wie in —- ABB — zu sehen, der Übersichtlichkeit halber mit kleinem vertikalen Ausschnitt. Mit diesem Schritt ergibt sich über drei Iterationen ein Verlauf des Korrekturterms f_{abs} und der Übertragungsfunktion für eine Schrittweite $\sigma_H = 1/2$ wie in —- ABB — dargestellt.

Ignorieren großer Korrektur-Terme

Ein zweiter, sehr grober Ansatz liegt in der Beschränkung von f_{abs} auf Werte unterhalb einer vorgegebenen Schwelle. Zugrunde liegt die Annahme, dass die gerade an Nulldurchgängen des Spektrums sowie bei vielen hohen Frequenzen auftretenden großen Werte durch die in obiger Aufzählung genannten Fehlerquellen entstehen. Hier bedeutet dies insbesondere, dass die Diskretisierung die Nulldurchgänge nicht korrekt darstellen kann. Die Interpolation auf Frequenzen von $\underline{H}$ ist dann aufgrund der großen Sprünge von Werten in direkter Umgebung der problematischen Frequenzen mit großer Ungenauickgeit behaftet. Dies kann zu den beschriebenen, großen Korrektur-Termen in f_{abs} führen.

Listing 6.1: Pseudocode zur Veranschaulichung der Anpassung des Korrekturterms

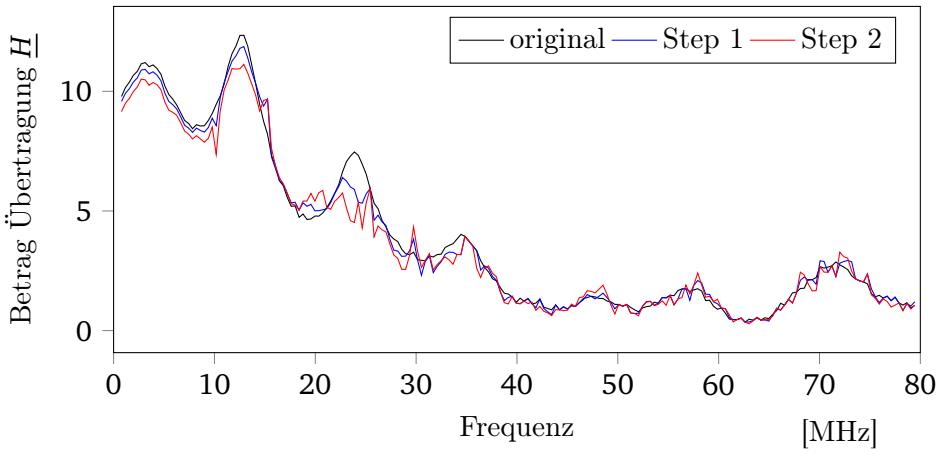
```
rms_orig = root_mean_square( f_abs )
f_abs_to_use = f_abs[ where( abs(f_abs) >= 0.02 * rms_orig ]
rms_mod = root_mean_square( f_abs_to_use )
idx_to_clear = f_abs[ where( abs(f_abs) >= rms_mod ]
f_abs[ ix_to_clear ] = 0
```

Vereinfacht bedeutet der verfolgte Ansatz, ausnehmend große Werte von f_{abs} als unrealistisch abzutun. Eine Pseudo-Implementierung findet sich in 6.1, um die nachfolgende Erläuterung zu illustrieren. In der vorgenommenen Implementierung wurde f_{abs} an den ausgewählten Frequenzen auf 0 gesetzt. Als Grenze genutzt wurde ein modifizierter Effektivwert, nachfolgend mit RMS (Root Mean Square) bezeichnet. Der reine RMS von f_{abs} unterliegt der Problematik, eine unproportional große Gewichtung von kleinen Einträgen zu enthalten.
Idealerweise enthält f_{abs} mit jeder Iteration kleinere Einträge als zuvor. Es würden also bei Nutzung des reinen RMS unter Umständen mit zunehmender Schrittzahl zunehmend mehr Werte in f_{abs} ignoriert - was der Optimierung entsprechende Grenzen setzt. In Kombination mit den im vorigen Abschnitt erläuterten Anpassungen wäre die Problematik unumgänglich, da Frequenzen, die explizit nicht bei der Anpassung berücksichtig werden sollen, den reinen RMS-Wert beeinflussen. Folglich muss der RMS modifiziert werden. Hier wurden zur Berechnung des modifizierten RMS nur die Werte einbezogen, die mehr als beliebig gewählte 2% des reinen RMS betragen. Es handelt sich also bei der vorgenommenen Anpassung um eine sehr grobe und größtenteils willkürliche Wahl der Parameter, die zu Zwecken der Illustration jedoch brauchbare Ergebnisse liefert.

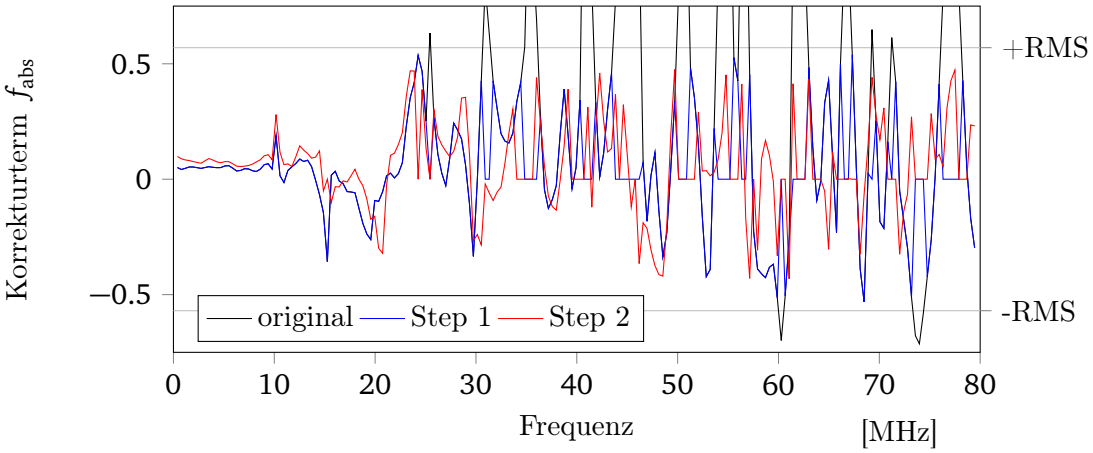
In Abb. (6.3) ist die Entwicklung von Übertragungsfunktion und f_{abs} über mehrere Iterationen aufgetragen. Der Einfluss des RMS-Cutters macht sich dabei verglichen mit —- ABB oben, rein iteriert —

bemerkbar, es treten weniger starke Änderungen auf. Es zeigt sich, dass die vorgenommene Anpassung keinen großartigen Einfluss auf die Qualität des Signals hat. Dies ist insofern beachtenswert, als dass die Übertragungsfunktion auch an einigen Stellen mit massiver Verstärkung stark angepasst wird, vergleiche hierzu Abb. (6.4a) bei etwa 25 MHz. Die Qualität des Signals bewegt sich zwischen einem Wert von —-—- und —- —- und zeigt vor allem rauschbedingte Schwankungen.

Abbildung 6.3: Entwicklung von Übertragungsfunktion und Korrekturterm bei Beschränkung von f_{abs} mit angepasstem RMS-Wert und Schrittweite $\sigma_H = \frac{1}{2}$



(a) Entwicklung des Betrags der Übertragungsfunktion über mehrere Iterationen und im Anfangszustand



(b) Entwicklung des Korrekturterms in angepasster Form über mehrere Iterationen und in initialer, nicht angepasster Form - RMS aus Step 1 zum Vergleich

6.2 Optimierung der nichtlinearen Kennlinie K

Der Unterschied zur Optimierung von $\underline{H}$ ist, dass diese Optimierung im Zeitbereich statt findet. Deshalb kann 6.1 zu

$$\Delta U_? (t) = U_{?,\text{meas}} (t) - U_{?,\text{ideal}} (t) \qquad U_{?,\text{meas}} (t) = \mathscr{F}^{-1} \left\{ \underline{H}^{-1} (\omega) \cdot \underline{U}_{out,\text{meas}} (\omega) \right\} \tag{6.5}$$

geändert werden. Bei den Funktionen $U_{?,\text{meas}}(t)$ und $U_{?,\text{ideal}}(t)$ handelt es sich um Polynome gleichen Grades deshalb lässt sich die Differenz ebenfalls als ein Polynom mit Grad N darstellen

$$\Delta U_?(t) = \sum_{n=1}^{N} \tilde{a}_n \left[U_{in}(t)\right]^n \tag{6.6}$$

Die Berechnung der Koeffizienten $\tilde{a}_n$ stellt ebenso ein lineares Optimierungsproblem dar wie schon die Berechnung der Koeffizienten a_n in -Gleichung- 3.1 siehe —Paper Kerstin—. Dabei werden M Samples von $\Delta U_{?,i} = \Delta U_?(i \cdot \Delta t)$ mit zugehörigen Samples des Eingangssignals $U_{in,i} = U_{in}(i \cdot \Delta t)$ verglichen. Mit der Potenzreihe aus -Gleichung- 6.5 ergibt sich folgendes Gleichungssystem

$$\begin{pmatrix} U_{in,1} & U_{in,1}^2 & \dots & U_{in,1}^N \\ U_{in,2} & U_{in,2}^2 & \dots & U_{in,2}^N \\ \vdots & \vdots & \ddots & \vdots \\ U_{in,M} & U_{in,M}^2 & \dots & U_{in,M}^N \end{pmatrix} \cdot \begin{pmatrix} \tilde{a}_1 \\ \tilde{a}_2 \\ \vdots \\ \tilde{a}_N \end{pmatrix} = \begin{pmatrix} \Delta U_{?,1} \\ \Delta U_{?,2} \\ \vdots \\ \Delta U_{?,M} \end{pmatrix} \tag{6.7}$$

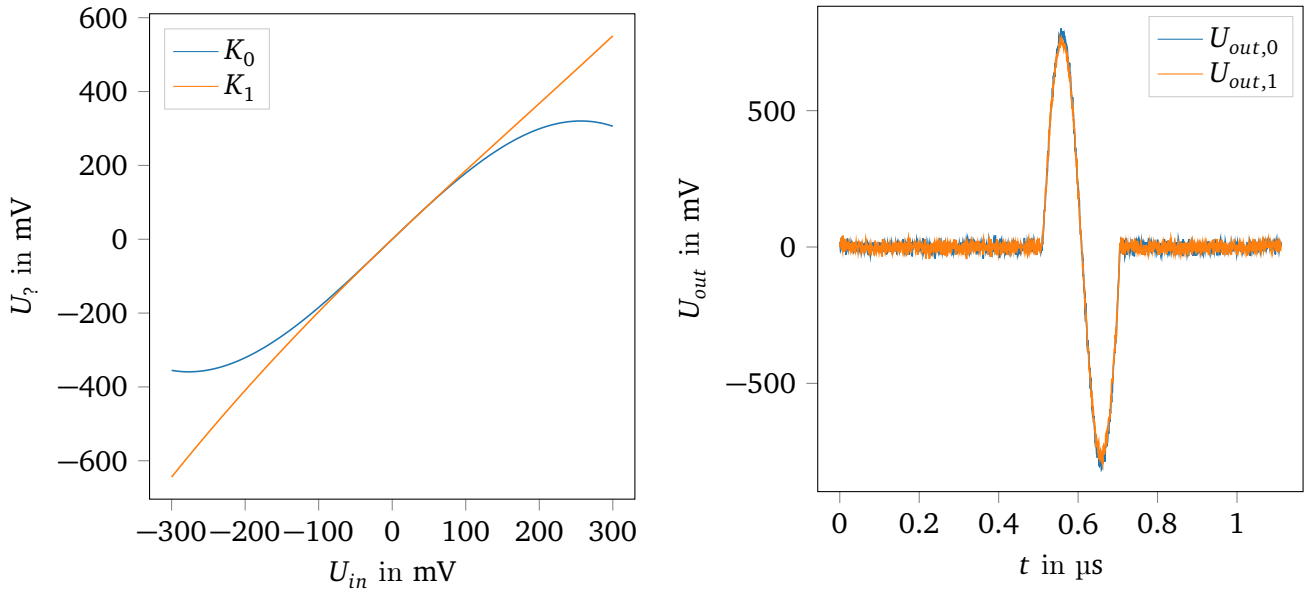
Dieses Gleichungssystem ist mit normalerweise $M > N$ überbestimmt und wird mit der Methode der kleinsten Quadrate gelöst. Die Koeffizienten $\tilde{a}_n$ werden nun wie folgt zur Anpassung der Koeffizienten a_n verwendet

$$a_n^{i+1} = a_n^i + \sigma_a^i \tilde{a}_n^i \tag{6.8}$$

Für die Schrittweite gilt $\sigma_a^i \in [0, 1]$.

Erste Ergebnisse

Für die Berechnung der ersten Kennlinie K_0 wurde das ideal Ausgangssignal $U_{out,\text{ideal}}$ über $\underline{H}^{-1}$ zurückgerechnet und als Eingangssignal verwendet $U_{in,\text{ideal}} = U_{?,\text{ideal}}$. Dabei wurde $V_{PP} = 600\,\text{mV}$ gesetzt, um K_0 in einen größeren Bereich berechnen zu können.
Wenn man jetzt $U_{?,\text{ideal}}$ mit $V_{PP} = 600\,\text{mV}$ über K_0 zurückrechnet, um das erste nichtlinear vorverzerrtes Eingangsignal zu erhalten, so stellt man fest, dass die Grenzen, in denen K_0 invertiert werden kann, zu klein sind. Wenn also $U_{?,\text{ideal}}$ über die Grenzen von K_0 geht, so wäre ein möglicher Ansatz V_{PP} auf den maximal von K_0 zulässigen Wert zu setzen.
Als andere Möglichkeit die Kennlinie anzupassen könnte man V_{PP} von $U_{?,\text{ideal}}$ verkleinern siehe Abb. (??)



(a) Kennlinie

(b) Ausgangsspannung

Abbildung 6.5: Anpassung von K

In Abb. (6.5a) ist K_0 die initial Kennlinie und K_1 ist die Kennlinie nach der ersten Anpassung. Dabei wurde $U_{?,\text{ideal}}$ so berechnet, dass für $U_{out,\text{ideal}}$ gilt $V_{PP} = 1.5\,\text{V}$

-Probleme mit der Kennlinie-

-Hier wird die Problematik mit der Amplitude nochmal aufgegriffen und unsere Lösungsansätze mit Plots verdeutlicht-

7 Fazit

—- In diesem Kapitel wird eine kurze Evaluierung vorgenommen. Welche Aspekte der Problemstellung wurden erfüllt (welche nicht), welche Hindernisse genommen? Einordnung der eigenen Arbeit in Kontext der in der Einleitung geführten Rahmenbedingung? Welche Erkenntnisse sind besonders erwähnenswert? Hier können Erfahrungen mit den gedachten Vorteilen (siehe 5.1) des Codes oder eine Bewertung der Sinnhaftigkeit der Optimierung nochmals geführt werden.—

8 Ausblick

—- In diesem Kapitel wird auf offene Fragen / neue Probleme / Anstöße für weitere Arbeiten eingegangen. Dabei sollte es um eher inhaltliche Aspekte gehen (u. U. wenig to dos für Code-Design) gegebenenfalls darf hier bei vielem auf die Erfahrungen aus den vorigen Kapiteln verwiesen werden und damit einen Übersichts-Charakter haben (erleichtert nachfolgenden Projekten die Arbeit) —

8.1 —- Optimierung —-

— in dieser section werden die offenen Fragen / Anregungen in Bezug auf den Optimierungs-algorithmus dargelegt, etwa:
- Iterations-Reihenfolge mit Messungen: abwechselnd K / H optimieren und dann messen oder zwischen zwei Messungen beide optimieren? Oder erst das eine mit mehreren Messungen optimieren und dann das andere ?
- sinnhaftigkeit / Grenzen des genutzten Algorithmus (Idee Jens) erfragen?
- Problem / Frage nach Phasen-Optimierung des H-Optimierers
- Umgang mit Rauschen, Null-Durchgängen o. ä. im H-Optimierer
- Möglichkeiten der Optimierung: Durchlaufen lassen mit direkter Anpassung an steigende Amplituden?
- (offene Punkte K-Optimierung?)

8.2 — Gerätekomm—

— in dieser Section werden weitere Punkte der Geräte-Komm aufgegriffen, etwa - die (geringe) Auflösung des AWG im Kontext der Optimierung (ggf. in 8.1 besser?) , die Einbindung des neuen Oszis oder die Idee der Klassen-Implementierung

Literaturverzeichnis

[1] Leslie Lamport, LaTeX: a document preparation system, Addison Wesley, Massachusetts, 2nd edition, 1994.

[2] Denys Bast, Armin Galetzka, Projektseminar Beschleunigertechnik, 2017.

[3] Jens Harzheim et al., Input Signal Generation For Barrier Bucket RF Systems At GSI, — year —-

[4] Kerstin Gross et al., Test Setup For Automated Barrier Bucket Signal Generation, In Proceedings of IPAC2017 in Copenhagen, Dänemark. pp. 3948 - 3950 2017.

[5] Keysight Technologies, Keysight Trueform Series Operating and Service Guide, 2015.

[6] Keysight Technologies, 33600A Series Trueform Waveform Generators - Data Sheet, 2014.

[7] C. Tröser, Application Note: Top Ten SCPI Programming Tips for Signal Generators, Rohde & Schwarz, 2013.

9 — Anhang —