

修士論文

光ケーブル内に構成されたマッハツェンダ型光ファイバ振動センサへの振動伝達特性に関する研究

令和5年度

三重大学大学院 工学研究科 情報工学専攻

黒田 修平

## 目次

第1章 序論.....	3
1.1. 研究背景と目的.....	3
1.2. 提案手法.....	3
1.3. 本論文の構成.....	4
第2章 振動伝達特性.....	5
2.1. 様々な光ファイバ振動センサ.....	5
2.2. 振動伝達特性の把握の必要性.....	6
2.3. 振動伝達特性の定義.....	6
第3章 モデルパラメータ推定による振動伝達特性の測定.....	7
3.1. マッハツェンダ型光ファイバ振動センサのモデル.....	7
3.2. モデルのあてはめによる振動伝達特性の測定方法.....	7
3.3. モデルのあてはめによる振動伝達特性の測定アルゴリズム.....	8
第4章 モデルのあてはめによる実測実験.....	10
4.1. 実験系概要.....	10
4.2. ケーブル部.....	12
4.2.1. 局内ケーブル.....	12
4.2.2. 地中ケーブル.....	12
4.2.3. ドロップケーブル.....	12
4.3. 実験.....	13
4.4. モデルのあてはめによる振動伝達特性の測定.....	13
第5章 振動伝達特性の測定の簡易化の検討.....	15
5.1. モデルのあてはめの問題点.....	15
5.2. 振動センサのモデルの簡易化による振動伝達特性の測定方法.....	15
5.3. モデルの簡易化による振動伝達特性の測定アルゴリズム.....	17
5.4. モデルの簡易化による $V\omega v$ の推定.....	19
5.5. モデルの簡易化の問題点と改善策.....	19
第6章 まとめ.....	22
6.1. 結論.....	22
6.2. 今後の課題.....	22
参考文献.....	23
研究業績.....	24
謝辞.....	25
付録A 実験系のケーブル部について.....	26
付録B 実測実験に使用したプログラムについて.....	26
B.1 モデルのあてはめによる振動伝達特性測定プログラム.....	26
B.2 モデルの簡易化による振動伝達特性測定プログラム.....	31

## 図表目次

図 1.1	本論文の構成.....	4
図 2.1	マツハツェンダ型光ファイバ振動センサの基本的な構成図.....	5
図 2.2	スロット型地中ケーブルの断面図.....	6
図 3.1	マツハツェンダ型光ファイバ振動センサの構成図.....	7
図 3.2	モデルのあてはめによる振動伝達特性測定プログラムのフローチャート.....	9
図 4.1	実験系図.....	11
図 4.2	局内ケーブルの断面図.....	12
図 4.3	スロット型およびノンスロット型地中ケーブルの断面図.....	12
図 4.4	ドロップケーブルの断面図.....	13
図 4.5	モデルあてはめによる振動周波数における振動伝達特性.....	14
図 5.1	モデルあてはめ手法の 3 変数グラフ.....	15
図 5.2	局内ケーブルに 700Hz の振動を与えたときの FFT の結果.....	16
図 5.3	モデルの簡易化による振動伝達特性測定プログラムのフローチャート.....	18
図 5.4	2 手法による振動周波数における推定した $V_{\omega_v}$ のグラフ.....	19
図 5.5	モデルの簡易化手法による $V_{\omega_v}$ に対する $g(V_{\omega_v})$ の値のグラフ.....	20
図 5.6	モデルあてはめ手法による振動周波数における $g(V_{\omega_v})$ .....	20
図 5.7	モデルあてはめ手法としきい値を考えた場合のモデル簡易化手法の $V_{\omega_v}$ の比較.....	21

## 第1章 序論

### 1.1. 研究背景と目的

近年, FTTHをはじめとした光ファイバを用いた光通信が広く普及している. それに伴い, 光ファイバに異常が生じた際の通信障害の影響も増加する. そのため, 光ファイバの異常検知技術の向上が必要不可欠であり, 光ファイバを通信路としてのみならず, センサとして活用する技術が研究されている. その中でも, 光ファイバを用いた振動センシングに関する研究が盛んである[1,2]. 例えば, 地震による振動を検知する手法として, 地震計を使用する手法がある. しかし, 地震計を設置するにあたり人件費や設置コストが必要になる. 世界中には現在, 光ファイバケーブルが広く敷設されている. 光ファイバケーブル外部に与えられた振動が, 内部の光ファイバに伝わる際の特性が明確になれば, すでに敷設されている光ファイバケーブルを地震計として利用することができる可能性がある. このように, 地震や津波といった自然災害の検知などへの応用が考えられる. 振動センサとして屋外で使用される光ファイバはケーブル状に加工されたものである. したがって, 光ファイバケーブル外部に与えられる振動と内部の光ファイバに伝わる振動には, ケーブル種別により, その構造に起因した違いが発生すると考えられる. また, センシングケーブルの振動応答を計測する研究では, 光ファイバケーブルの応答は振動周波数に依存し, 振動周波数が異なると応答が異なることが示されている[3]. しかしながら, 光ファイバケーブル外部への振動が光ファイバの物理的な特性にどの程度影響を与えるのかは, 必ずしも明確にされていない.

そこで本研究では, ケーブル外部に与えられた振動が, 内部の光ファイバに伝達する際の特性について明確にすることを目的とする.

### 1.2. 提案手法

我々は簡易な振動センシング手法の一つとして, 光ケーブル内にマッハツェンダ型光ファイバ干渉計を構成する手法を提案している[4]. そこで本研究ではまず, 光ケーブル内に構成されたマッハツェンダ型光ファイバ振動センサについて, ケーブル外部に与えた振動の加速度の大きさと, 振動センサの位相変化との関係を振動伝達特性と定義する. 次に, マッハツェンダ型光ファイバ振動センサのモデル化を行い, そのモデルパラメータを推定することにより, 振動伝達特性を測定する手法を提案する. 提案手法を用いて光ファイバケーブルの振動伝達特性の測定を行い, ケーブル種別により振動伝達特性に違いがあることを示す.

### 1.3. 本論文の構成

本論文の構成を図 1.1 に示す.

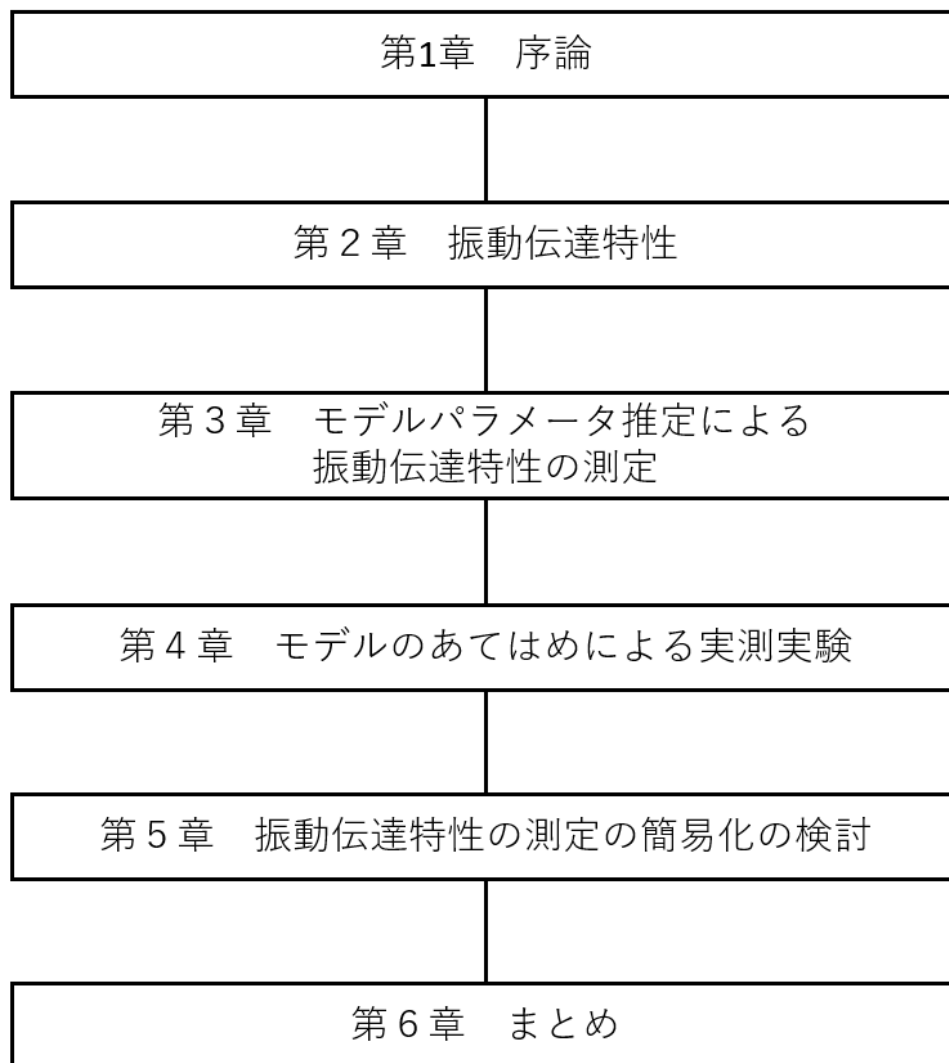


図 1.1 本論文の構成

第2章では、光ファイバ振動センサについての説明や、振動伝達特性把握の必要性について論じている。第3章では、モデルパラメータを推定することにより振動伝達特性の測定を行う手法を提案し、説明している。第4章では、実験系の説明をしたのち、モデルのあてはめによる振動伝達特性測定の実測実験の結果から、光ファイバケーブルの構造や材質の違いによる振動伝達特性の違いについて論じている。第5章では、振動伝達特性の測定をより簡易に行う手法について提案し、実測実験の結果から考察を行っている。第6章では本論文のまとめを示している。

## 第2章 振動伝達特性

### 2.1. 様々な光ファイバ振動センサ

光ファイバを用いた振動センシングには様々な手法があり、分散型光ファイバセンシングや、干渉計を用いた光ファイバセンシングなどがある[5,6]。例えば分散型音響センサ(DAS)は、分散型光ファイバセンシング手法の一つである。DASは数十キロメートルの光ファイバに沿って振動を検知することができるという利点がある。欠点として、DASシステムは高価であるため、設置及び保守にコストがかかることが挙げられる。光干渉を用いた光ファイバセンシングでは、マイケルソン干渉計や、サニャック干渉計、マッハツェンダ干渉計を用いて振動を検知する。利点として、干渉計は微小な位相変化を検知するため、高い感度と解像度を持つことが挙げられるが、環境ノイズに敏感であるという欠点がある。本研究では、簡易な振動検知手法として、光ケーブル内にマッハツェンダ型光ファイバ干渉計を構築する手法を提案する。

図 3.1 にマッハツェンダ型光ファイバ振動センサの基本的な構成図を示す。光源から射出された光はカップラによって分割され、センサファイバおよび参照ファイバに伝送される。マッハツェンダ型光ファイバ振動センサは、これらの光を再度カップラで結合することによって得られる干渉の状態を観測する干渉計である[5]。光ファイバに振動が与えられないときは、センサファイバと参照ファイバに光路差は存在しない。しかし、振動が与えられると、光ファイバの屈折率が変化することから、センサファイバと参照ファイバに位相差が生じ、干渉光も変化する。これにより振動の検知を行うことができる。

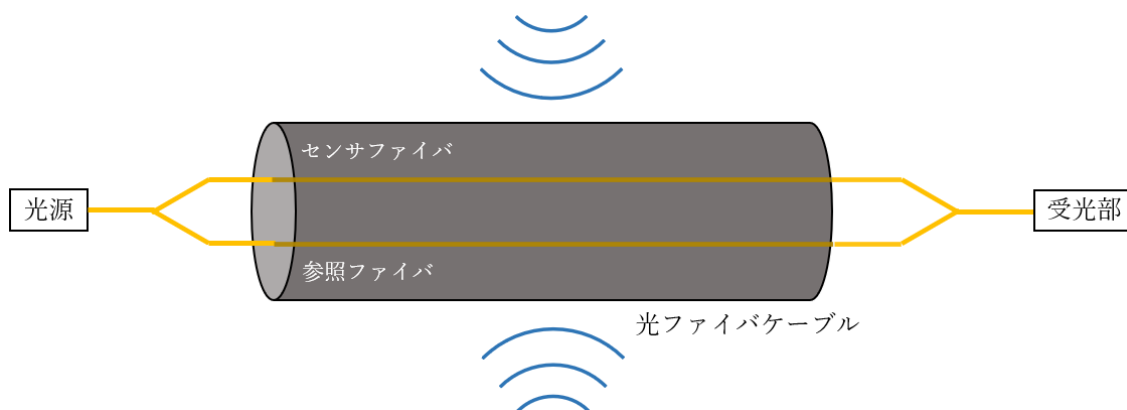


図 2.1 マッハツェンダ型光ファイバ振動センサの基本的な構成図

## 2.2. 振動伝達特性の把握の必要性

従来の光ファイバ振動センシングの研究では、 piezo素子に光ファイバを巻き付けたファイバストレッチャーを用いて、光ファイバに直接伸縮を与えている。しかしながら、実際に敷設された光ファイバはケーブル上に加工されているため、光ファイバケーブルに与えられた振動と、内部の光ファイバに伝わる振動は異なると考えられる。また、ケーブルの種別によってその構造が異なっている。図 2.2 にスロット型地中ケーブルの断面図を示す。スロット型地中ケーブルはスロットと呼ばれる溝に光ファイバが格納されており、振動がスロットを通じて伝達されるため、光ファイバケーブル外部に与えられた振動と内部の光ファイバに伝わる振動は異なると考えられる。以上の理由から、光ファイバケーブル外部に与えられた振動と、内部の光ファイバに伝わる振動の関係を計測することが必要であると考えられる。

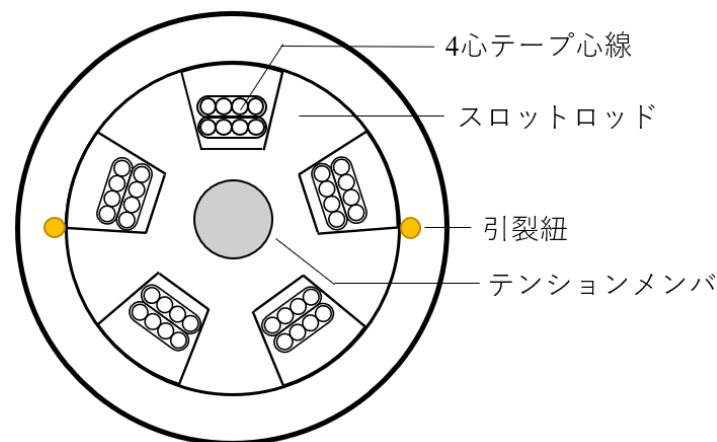


図 2.2 スロット型地中ケーブルの断面図

## 2.3. 振動伝達特性の定義

マツハツェンダ型光ファイバ振動センサのケーブル外部に振動を与えると、センサファイバおよび参照ファイバに位相差が生じる。本研究では、センサファイバおよび参照ファイバの位相変化を与えた振動の加速度の大きさを割った値を振動伝達特性と定義し、ケーブルの振動伝達特性を算出することを目的とする。

## 第3章 モデルパラメータ推定による振動伝達特性の測定

### 3.1. マッハツェンダ型光ファイバ振動センサのモデル

図 3.1 に加速度 $a(t) = A_{\omega_v} \cos \omega_v t$ の振動を与えたときのマッハツェンダ型光ファイバ振動センサの構成図を示す。ケーブル外部に加速度 $a(t)$ の振動を与えると、センサファイバおよび参照ファイバに位相差が生じ、光干渉が発生する。この光干渉を受光部で電気信号に変換することで観測する[7]。以下に、振動伝達特性の測定手法を提案する。

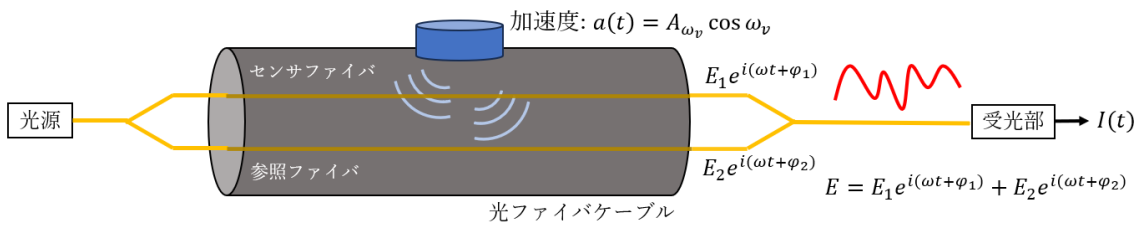


図 3.1 マッハツェンダ型光ファイバ振動センサの構成図

### 3.2. モデルのあてはめによる振動伝達特性の測定方法

センサファイバおよび参照ファイバにおける電界の複素振幅をそれぞれ $E_1 e^{i(\omega t + \phi_1)}$ ,  $E_2 e^{i(\omega t + \phi_2)}$ とすると、フォトダイオードにおける複素振幅 $E$ は以下のようにあわせる。

$$E = E_1 e^{i(\omega t + \phi_1)} + E_2 e^{i(\omega t + \phi_2)} \quad (3.1)$$

ここで、 $\phi_1, \phi_2$ はそれぞれセンサファイバおよび参照ファイバの位相、 $\omega$ は光の角周波数である。フォトダイオードで観測される光の強度 $I(t)$ は以下のように表すことができる。

$$I(t) = K|E|^2 = K\{|E_1|^2 + |E_2|^2 + 2|E_1||E_2|\cos(\phi_1 - \phi_2)\} \quad (3.2)$$

ここで、 $K$ は光検出器の光入力と電気出力間の係数である。センサファイバおよび参照ファイバの位相差 $\Delta\phi$ は以下のように表すことができる。

$$\Delta\phi = \phi_1 - \phi_2 = V_{\omega_v} \cos(\omega_v t) - \theta \quad (3.3)$$

ここで、 $\omega_v$ は与える振動の角周波数、 $V_{\omega_v}$ は角周波数 $\omega_v$ の振動による光ファイバ中の位相変化の大きさを表す係数、 $\theta$ は環境変化によるセンサファイバおよび参照ファイバの全長差や偏波状態の変化に起因した位相差である。マッハツェンダ型光ファイバ振動センサにおいて、光ファイバケーブル表面に加速度が $a(t) = A_{\omega_v} \cos \omega_v t$ （振幅： $A_{\omega_v}$ 、角周波数： $\omega_v = 2\pi f_v$ ）の振動を付加した際の光の強度 $I(t)$ は次式のように表せる。

$$I(t) = K[E_1^2 + E_2^2 + 2E_1 E_2 \cos\{V_{\omega_v} \cos(\omega_v t) - \theta\}] \quad (3.4)$$



DC ブロックフィルタ通過後の干渉光の強度 $I'(t)$ は次式のように表せる．

$$I'(t) = 2KE_1E_2 \cos\{V_{\omega_v} \cos(\omega_v t) - \theta\} \quad (3.5)$$

$KE_1^2$ および  $KE_2^2$ は各アームの光の強度を測定することで求めることができる． $I_1 = KE_1^2$ ， $I_2 = KE_2^2$ とにおいて，式(3.5)を変形することにより次式を得る．

$$I'(t) = 2\sqrt{I_1I_2} \cos\{V_{\omega_v} \cos(\omega_v t) - \theta\} \quad (3.6)$$

式(3.6)をモデル式として，得られた干渉波形との平均二乗誤差を算出し， $V_{\omega_v}$ を探索することで，光ファイバケーブルの表面に加速度の大きさが $A_{\omega_v}$ である振動を与えたときに生じる位相変化 $V_{\omega_v}$ の関係を算出することができる．様々な振動周波数について測定することにより，光ファイバケーブルの振動伝達特性を以下のように算出する．

$$c(f_{\omega_v}) = \frac{V_{\omega_v}}{A_{\omega_v}} \quad (3.7)$$

### 3.3. モデルのあてはめによる振動伝達特性の測定アルゴリズム

実験前に，センサファイバおよび参照ファイバそれぞれに光が通過した時の光の強度 $I_1, I_2$ を計測する．受光器で得られた干渉波と，加速度計で得られた加速度データをそれぞれAD変換器で量子化し，解析装置のメモリに格納する．メモリに格納したN点のデータを

$$x = [x_0, x_1, \dots, x_{N-1}] \quad (3.8)$$

$$y = [y_0, y_1, \dots, y_{N-1}] \quad (3.9)$$

とする． $y$ に離散フーリエ変換を行い得られたデータを

$$Y = [Y_0, Y_1, \dots, Y_{N-1}]^T \quad (3.10)$$

とすると， $Y_k$ は周波数 $f_k = (f_s/N) \times k$ における加速度の大きさとなる．ただし $f_s$ はAD変換器におけるサンプリング周波数である．次に， $X_k$ および $Y_k$ における振幅を求め，与えた振動の周波数成分における加速度の大きさ $A_{\omega_v}$ を求める．式(3.7)の $V_{\omega_v}$ を0.00001から0.00001ずつ加算， $\theta$ を $0^\circ$ から $10^\circ$ ずつ加算したものと， $x$ との平均二乗誤差を求め，最小の場合の $V_{\omega_v}$ と $\theta$ を算出する．求めた $A_{\omega_v}$ と，算出した $V_{\omega_v}$ を式(3.7)に代入することで，振動伝達特性を算出する．

図3.2にプログラムのフローチャートを示す．

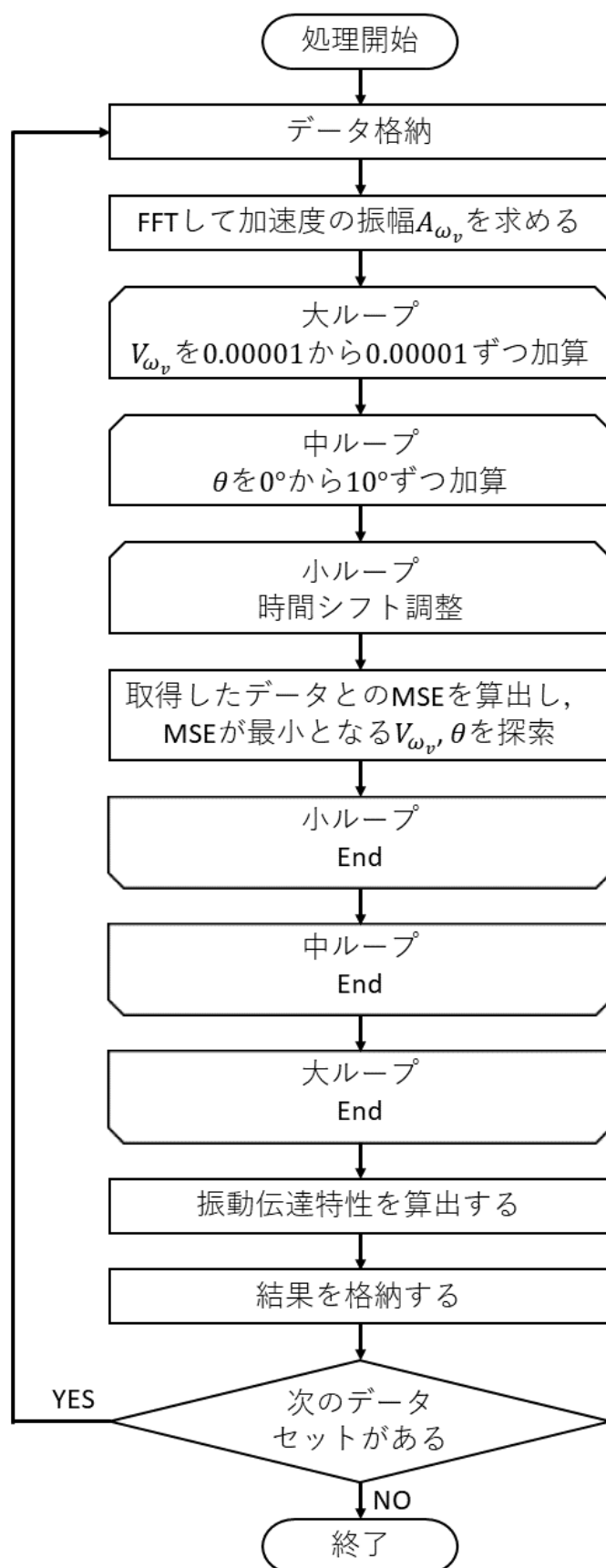


図 3.2 モデルのあてはめによる振動伝達特性測定プログラムのフローチャート

## 第4章 モデルのあてはめによる実測実験

### 4.1. 実験系概要

本研究を行うにあたって作成した実験系全体の構成図を図 4.1 に示す。本実験系では、光源から射出された光を  $1 \times 2$  カプラで分割し、光ファイバケーブル内にマッハツェンダ型光ファイバ干渉計を構成している。分割した光を再度  $2 \times 1$  カプラで結合し、干渉光を観測する。被測定部はケーブルであり、振動スピーカーを用いてケーブル部に振動を与えた。種別の異なるケーブルを用いて振動伝達特性の測定を行うことで振動伝達特性の比較を行った。

マッハツェンダ型光ファイバ振動センサの光源 (DFB-LD) には、中心波長 1551 nm、パワー 5 mW である FOLS-02 (澤木工房株式会社) を、出力 3.0 dBm になるように設定して使用した。センサファイバおよび参照ファイバは、ケーブル部の光ファイバに接続されている。ケーブル部は長さ 90 cm、40 心のスロット型およびノンスロット型地中ケーブル、局内ケーブル、ドロップケーブルである。ケーブル部には振動スピーカーを設置し、振動を与えた。任意波形発生器を PC に接続し、200~3000 Hz の正弦波を生成し、振動スピーカーを駆動した。任意波形発生器として Analog Discovery Pro (DIGILENT 社) [8] を用いた。Analog Discovery Pro は任意波形発生器の機能やオシロスコープ機能を持ち、付属する WaveForms 2015 というソフトを用いることで任意波形を作成したり、視覚的にデータを確認したりすることができる。ケーブル部に与えられた実際の振動は、加速度計である Model-1332B (昭和測器株式会社) を用いて計測した。マッハツェンダ型光ファイバ振動センサにおける干渉波は光検出器 (PD) を用いて検出し、PD で得られたデータの直流成分をカットするために DC ブロックフィルタを通過させたのち、干渉波形を測定しやすくするためにアンプに接続し、オシロスコープと PC で測定する。本研究では、PD として DET08CFC/M (THORLABS 社) を、DC ブロックフィルタとして EF500 (THORLABS 社) を、アンプとして T-01LNA (株式会社タートル工業) を使用した。

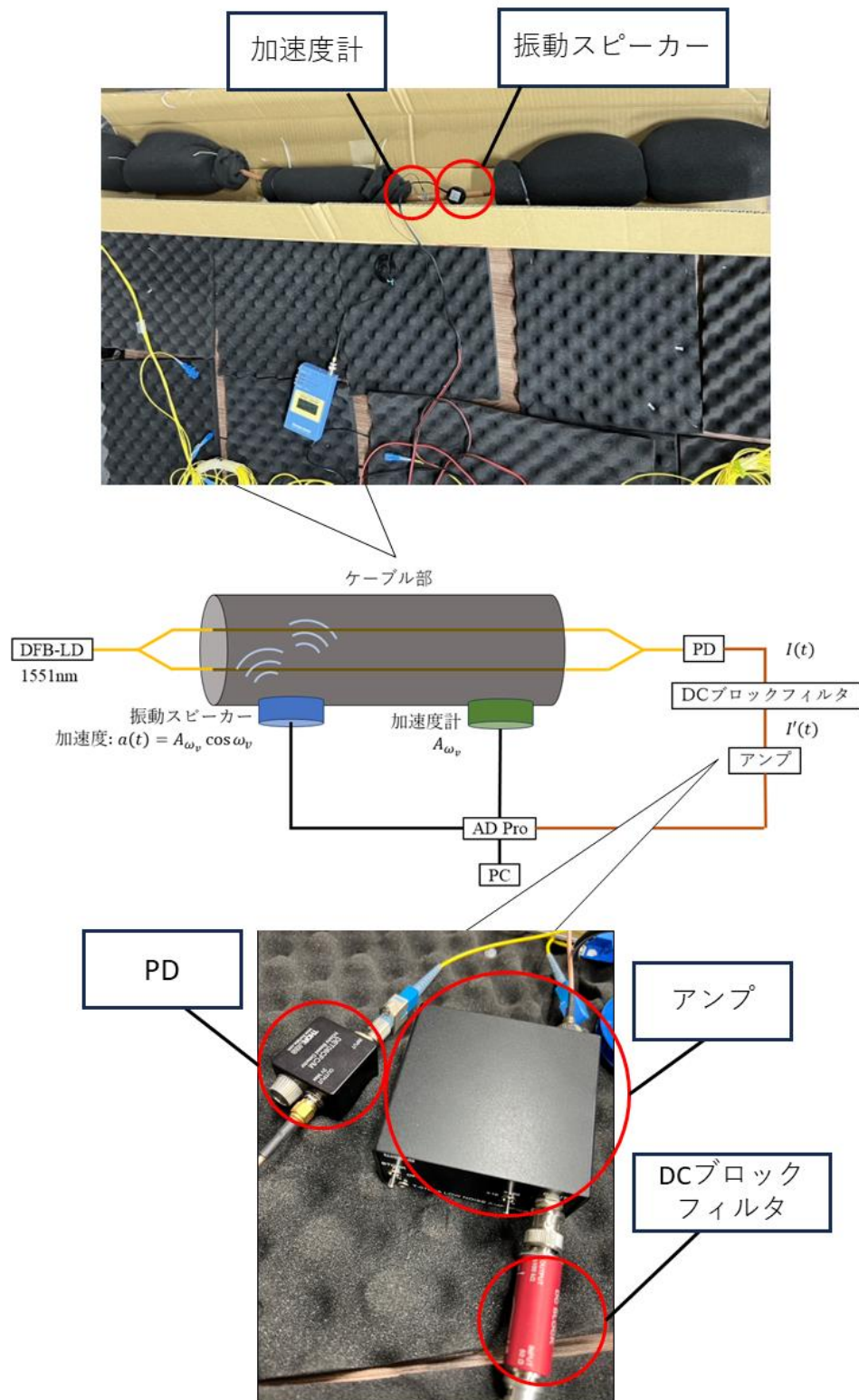


図 4.1 実験系図

## 4.2. ケーブル部

### 4.2.1. 局内ケーブル

局内ケーブルは、通信基地局内での中継光ファイバ間の接続に使用されるケーブルである[9]。今回の実験では、24 心、90cm の局内ケーブルを使用した。図 4.2 に局内ケーブルの断面図を示す。

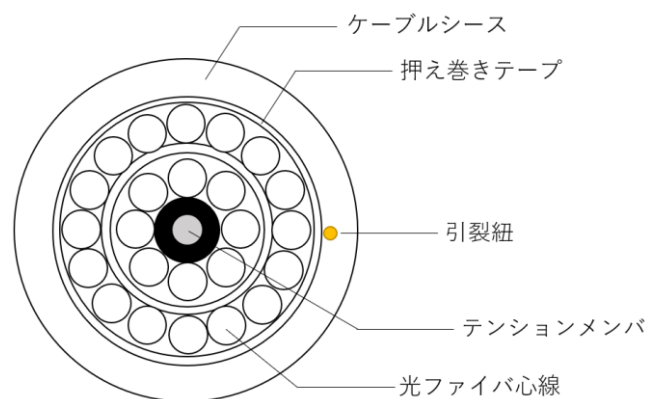


図 4.2 局内ケーブルの断面図

### 4.2.2. 地中ケーブル

地中ケーブルは、通信用に地下に敷設されているケーブルである。今回の実験では、40 心、90cm のスロット型地中ケーブルおよびノンスロット型地中ケーブルを使用した。図 4.3 にスロット型地中ケーブルおよびノンスロット型地中ケーブルの断面図を示す。

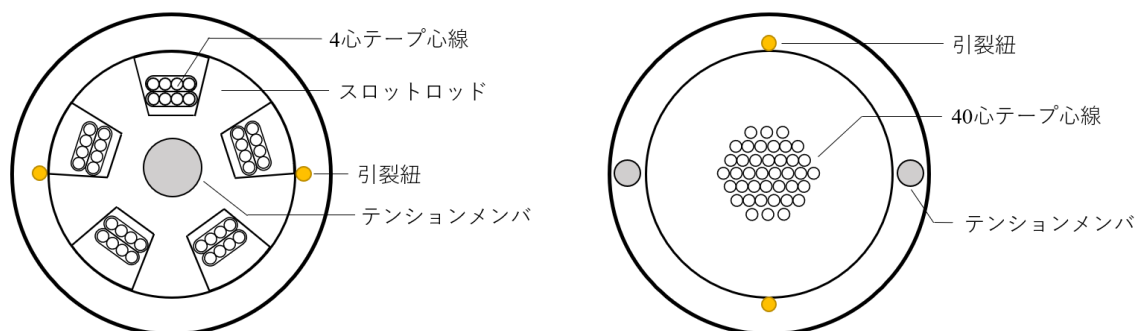


図 4.3 スロット型およびノンスロット型地中ケーブルの断面図

### 4.2.3. ドロップケーブル

ドロップケーブルは、FTTH における宅内への引き込み配線に用いられるケーブルである[10]。今回の実験では、8 心、200cm のドロップケーブルを使用した。図 4.4 にドロップ

ブケーブルの断面図を示す。

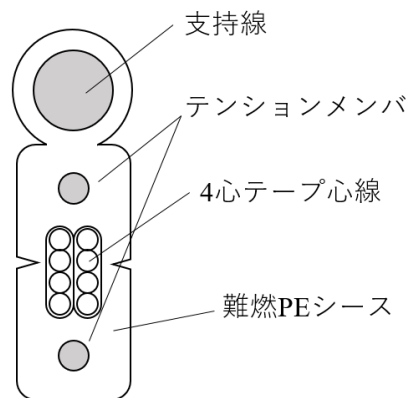


図 4.4 ドロップケーブルの断面図

### 4.3. 実験

実験では，Analog Discovery Pro の任意波形発生器機能と振動スピーカーを使用し，300～3000Hz の振動をケーブル部に与え，同機器のオシロスコープ機能を用いて，サンプリングレート 100kHz で各周波数におけるデータを取得した。

### 4.4. モデルのあてはめによる振動伝達特性の測定

モデルのあてはめ手法では，振動周波数ごとに 3 個のデータに対してモデルのあてはめを適用し，その中で平均二乗誤差が最も小さかった時の  $V_{\omega_p}$  を採用した。図 4.5 に，ケーブル部に 300Hz～3000Hz の振動を与えた場合の，モデルのあてはめ手法によって測定した振動伝達特性の測定結果を示す。ドロップケーブルは，光ファイバの周りが樹脂でおおわれた構造になっており，ケーブル内部の光ファイバに振動が伝わりやすく，結果として振動伝達特性もほかのケーブルを用いて測定した場合よりも大きくなっている。一方，ノンスロット型地中ケーブルを用いて測定を行った場合，材質が固いため振動が伝わりにくく，振動伝達特性も全体的に小さな値となっているが，共振点による特性の違いにより，500Hz, 2000Hz, 2800Hz の振動を与えた際の振動伝達特性が特に大きくなっている。このように，ケーブルの材質や構造の違いにより，振動伝達特性に違いが生じることが分かった。

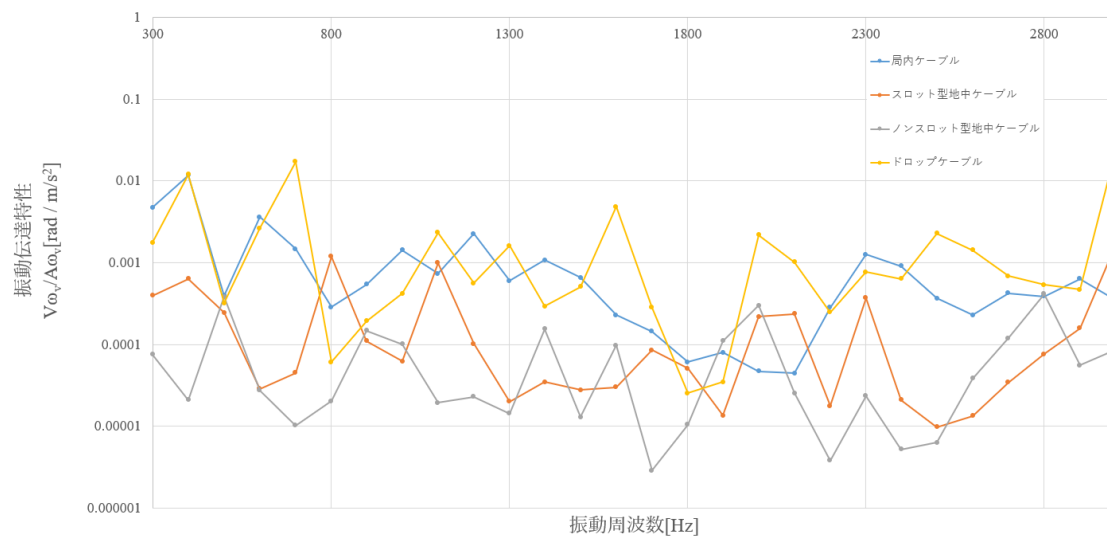


図 4.5 モデルあてはめによる振動周波数における振動伝達特性



## 第5章 振動伝達特性の測定の簡易化の検討

### 5.1. モデルのあてはめの問題点

モデルのあてはめによる測定手法の問題点として、3つのパラメータを動かしながら測定を行うため、計算量が多くなり時間がかかるということが挙げられる。そこで、最急降下法の適用に向けて、評価関数を確認した。図 5.1 に、局内ケーブルに 600Hz の振動を与えた際の干渉波にモデルのあてはめ手法を適用した場合の  $V_{\omega_v}, \theta$ 、二乗平均誤差の3変数グラフを示す。

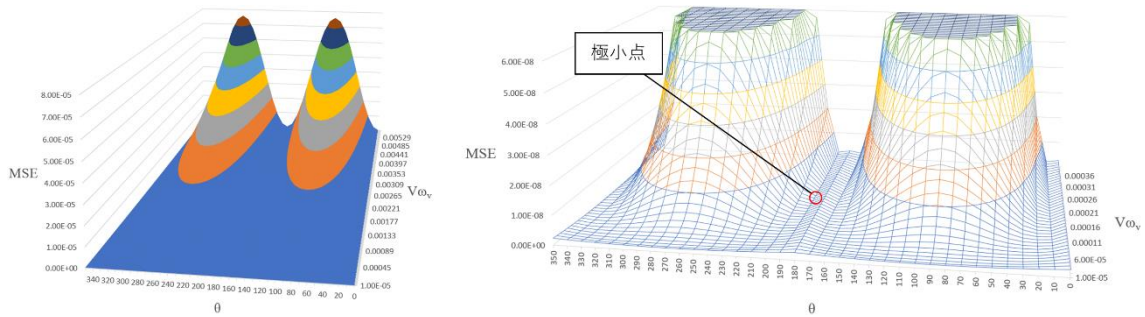


図 5.1 モデルあてはめ手法の3変数グラフ

このデータにおける平均二乗誤差の最小値は  $V_{\omega_v} = 0.00021, \theta = 190^\circ$  の時の  $3.93431454710623E-10$  である。図 5.1 からわかるように、平均二乗誤差の極小点が複数存在する。最急降下法を用いる場合、初期値によっては正しく極小値を算出することができないため、最急降下法の適用はできないと考える。

### 5.2. 振動センサのモデルの簡易化による振動伝達特性の測定方法

計算量が多く、時間がかかるという問題点を解消するために、マッハツェンダ型光ファイバ振動センサのモデルの簡易化を行うことで計算量の削減する手法を検討した。DC ブロックフィルタ通過後の干渉光の強度は、式(3.5)に加法定理を用いて、第一種ベッセル関数で展開することで次式のように表せる。

$$\begin{aligned}
 I'(t) = & 4KE_1E_2 \sin \theta J_1(V_{\omega_v}) \cos \omega_v t \\
 & - 4KE_1E_2 \cos \theta J_2(V_{\omega_v}) \cos 2\omega_v t \\
 & - 4KE_1E_2 \sin \theta J_3(V_{\omega_v}) \cos 3\omega_v t \\
 & + 4KE_1E_2 \cos \theta J_4(V_{\omega_v}) \cos 4\omega_v t \\
 & \dots
 \end{aligned} \tag{5.1}$$



ここで、 $J_n(x)$ は第一種ベッセル関数である．図 5.2 に，局内ケーブルに 700Hz の振動を与えた際の干渉光に離散フーリエ変換を施したグラフを示す．

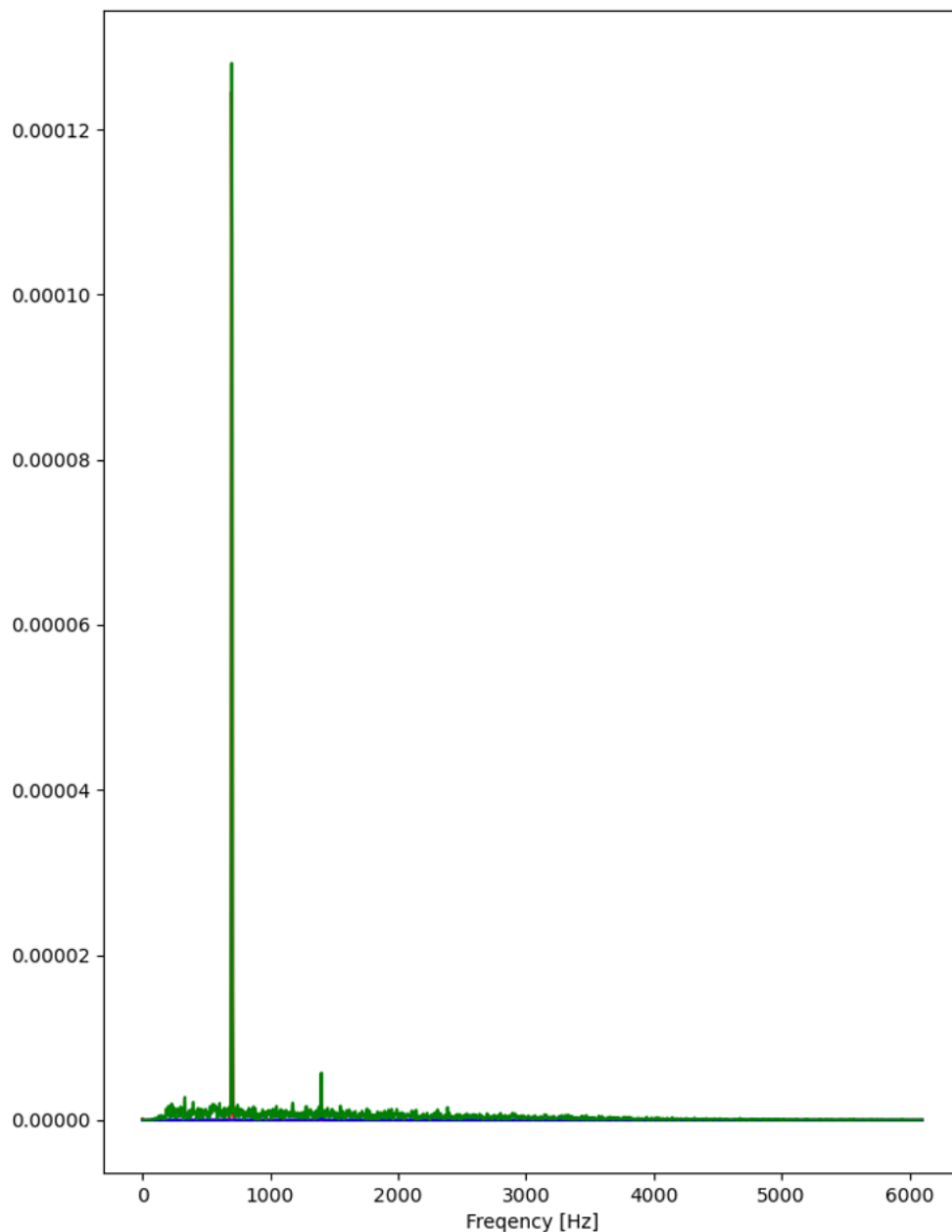


図 5.2 局内ケーブルに 700Hz の振動を与えたときの FFT の結果

このように，振動周波数と，その 2 倍の周波数以外の振幅がほとんど出ていないことに注目し，振動周波数とその 2 倍の周波数の振幅のみの成分を用いて近似することを考えた． $V_{\omega_v} \ll 1$ とすると， $J_3(V_{\omega_v}) = J_4(V_{\omega_v}) = \dots \approx 0$ であるとみなせるため，式(5.1)を近似することで次式を得る．

$$\begin{aligned}
 I'(t) \approx & 4KE_1E_2 \sin \theta J_1(V_{\omega_v}) \cos \omega_v t \\
 & - 4KE_1E_2 \cos \theta J_2(V_{\omega_v}) \cos 2\omega_v t
 \end{aligned} \tag{5.2}$$

$KE_1^2$ および  $KE_2^2$ は各アームの光の強度を測定することで求めることができる。また、 $4KE_1E_2 \sin \theta J_1(V_{\omega_v})$ および  $-4KE_1E_2 \cos \theta J_2(V_{\omega_v})$ は振動を与えたときの振動周波数とその2倍の周波数の振幅を測定することで求めることができる。一方  $\theta$ を直接求めることはできない。よって  $I_1 = KE_1^2$ ,  $I_2 = KE_2^2$ ,  $I'_{\omega_v} = 4KE_1E_2 \sin \theta J_1(V_{\omega_v})$ ,  $I'_{2\omega_v} = -4KE_1E_2 \cos \theta J_2(V_{\omega_v})$ とおき、式(5.2)を  $\cos^2 \theta + \sin^2 \theta = 1$ を用いて変形することにより次式を得る。

$$g(V_{\omega_v}) = \left( \frac{I'_{2\omega_v}}{4\sqrt{I_1 I_2} J_2(V_{\omega_v})} \right)^2 + \left( \frac{I'_{\omega_v}}{4\sqrt{I_1 I_2} J_1(V_{\omega_v})} \right)^2 = 1 \quad (5.3)$$

式(5.3)を満たす  $V_{\omega_v}$ を求めることで、光ファイバケーブルの表面に加速度の大きさが  $A_{\omega_v}$ である振動を与えたときに生じる位相変化  $V_{\omega_v}$ の関係を算出することができる。様々な振動周波数について測定することにより、光ファイバケーブルの振動伝達特性を式(3.7)と同様に算出することができる。

### 5.3. モデルの簡易化による振動伝達特性の測定アルゴリズム

実験前に、センサファイバおよび参照ファイバそれぞれに光が通過した時の光の強度  $I_1, I_2$ を計測する。受光器で得られた干渉波と、加速度計で得られた加速度データをそれぞれ AD 変換器で量子化し、解析装置のメモリに格納する。メモリに格納した  $N$  点のデータを

$$x = [x_0, x_1, \dots, x_{N-1}] \quad (5.4)$$

$$y = [y_0, y_1, \dots, y_{N-1}] \quad (5.5)$$

とする。  $x$ および  $y$ に離散フーリエ変換を行い得られたデータを

$$X = [X_0, X_1, \dots, X_{N-1}]^T \quad (5.6)$$

$$Y = [Y_0, Y_1, \dots, Y_{N-1}]^T \quad (5.7)$$

とすると、  $X_k$ は周波数  $f_k = (f_s/N) \times k$ における複素振幅となり、  $Y_k$ は周波数  $f_k = (f_s/N) \times k$ における加速度の大きさとなる。ただし  $f_s$ は AD 変換器におけるサンプリング周波数である。次に、  $X_k$ および  $Y_k$ における振幅を求め、与えた振動の周波数成分における振幅  $I'_{\omega_v}$ と、その2倍の周波数成分における振幅  $I'_{2\omega_v}$ 、および与えた振動の周波数成分における加速度の大きさ  $A_{\omega_v}$ を求める。事前に計測した  $I_1, I_2$ および求めた  $I'_{\omega_v}, I'_{2\omega_v}$ を式(5.3)に代入する。

ここで、  $V_{\omega_v}$ を 0.00001 から 0.00001 ずつ加算していき、式(5.3)を満たす  $V_{\omega_v}$ を探索する。

求めた  $A_{\omega_v}$ と、探索した  $V_{\omega_v}$ を式(3.7)に代入することで、振動伝達特性を算出する。

図 5.3 にプログラムのフローチャートを示す。

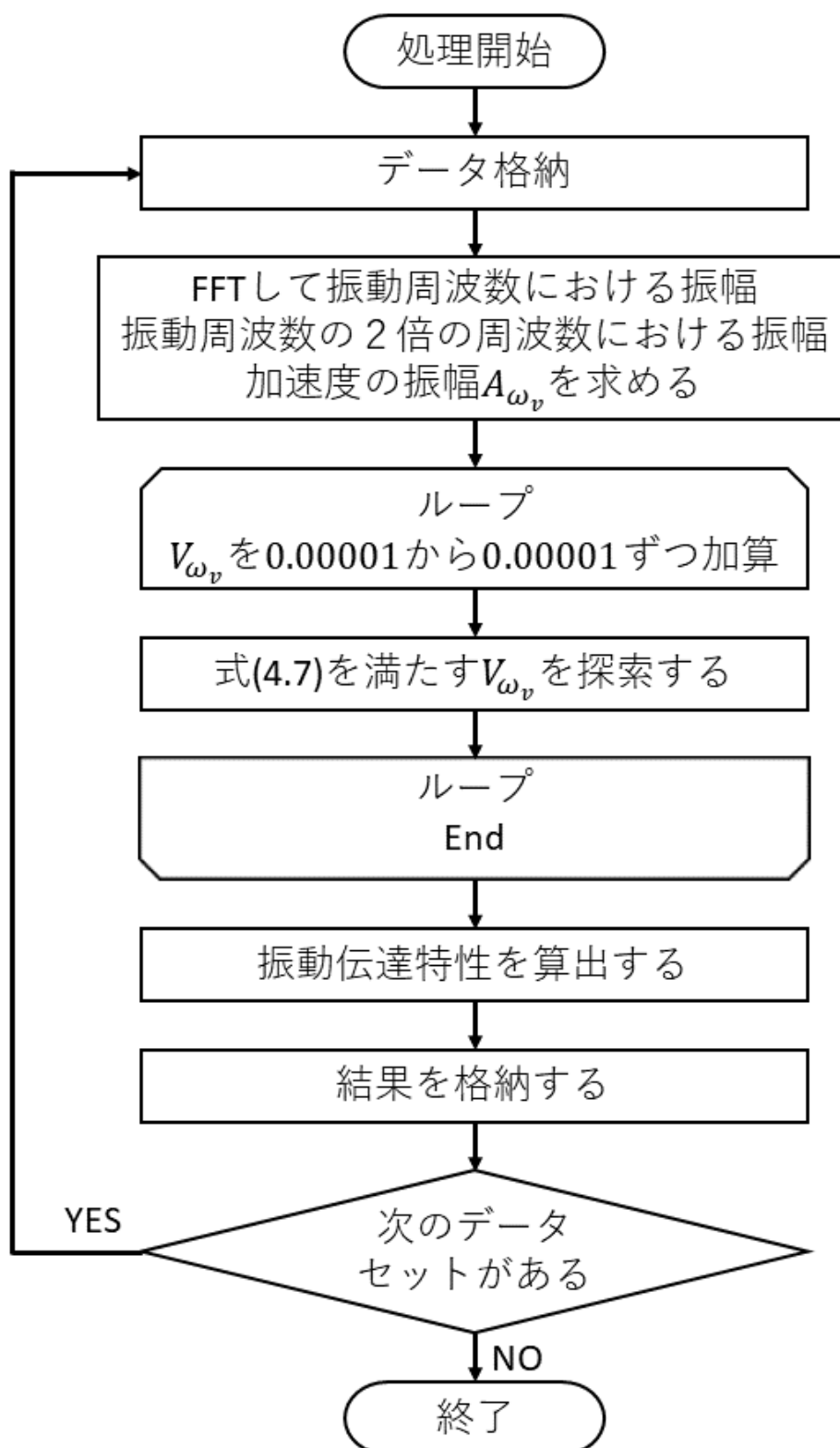


図 5.3 モデルの簡易化による振動伝達特性測定プログラムのフローチャート

## 5.4. モデルの簡易化による $V_{\omega_v}$ の推定

モデルの簡易化による手法では，振動周波数ごとに 200 個のデータに対してモデルの簡易化による手法を適用し，それぞれデータで推定した $V_{\omega_v}$ の平均値を $V_{\omega_v}$ として採用した．実験系はモデルのあてはめによる振動伝達特性を測定したのと同じ系である．図 5.4 に，局内ケーブルに 300～3000Hz の振動を与えた場合の，モデルの簡易化手法によって推定した $V_{\omega_v}$ の値と，モデルのあてはめ手法によって推定した $V_{\omega_v}$ の値を同時にプロットしたグラフを示す．

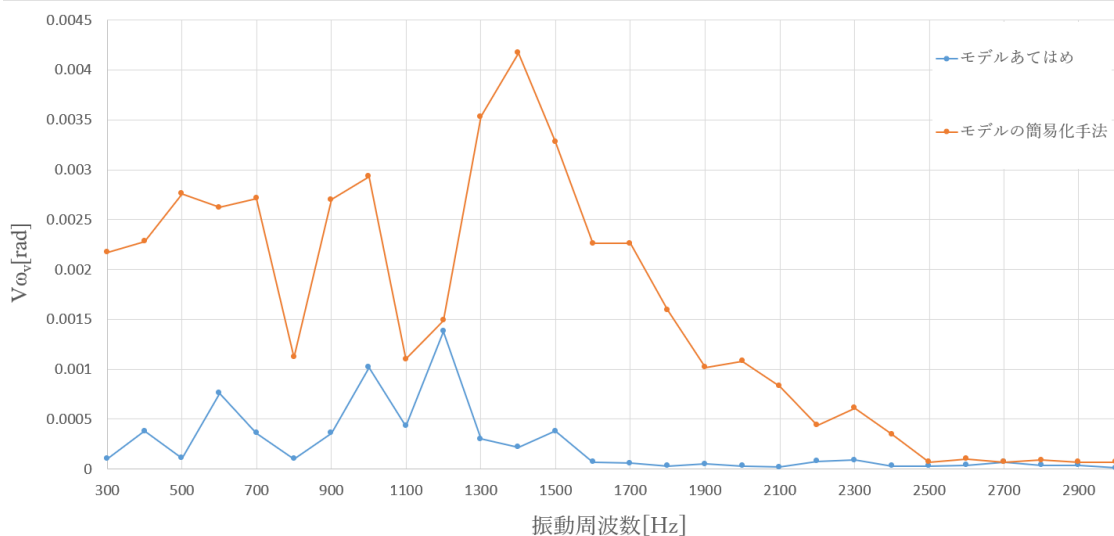


図 5.4 2 手法による振動周波数における推定した $V_{\omega_v}$ のグラフ

## 5.5. モデルの簡易化の問題点と改善策

モデルの簡易化手法では， $g(V_{\omega_v}) = 1$ となる $V_{\omega_v}$ を推定しているが，モデルのあてはめと比較すると $V_{\omega_v}$ が大きく異なることが分かった．原因の一つとして，モデル誤差があることが考えられる． $g(V_{\omega_v})$ の値は単調減少となっている．図 5.5 に，局内ケーブルに 600Hz の振動を与えた際の， $V_{\omega_v}$ に対する $g(V_{\omega_v})$ の値のグラフを示す．

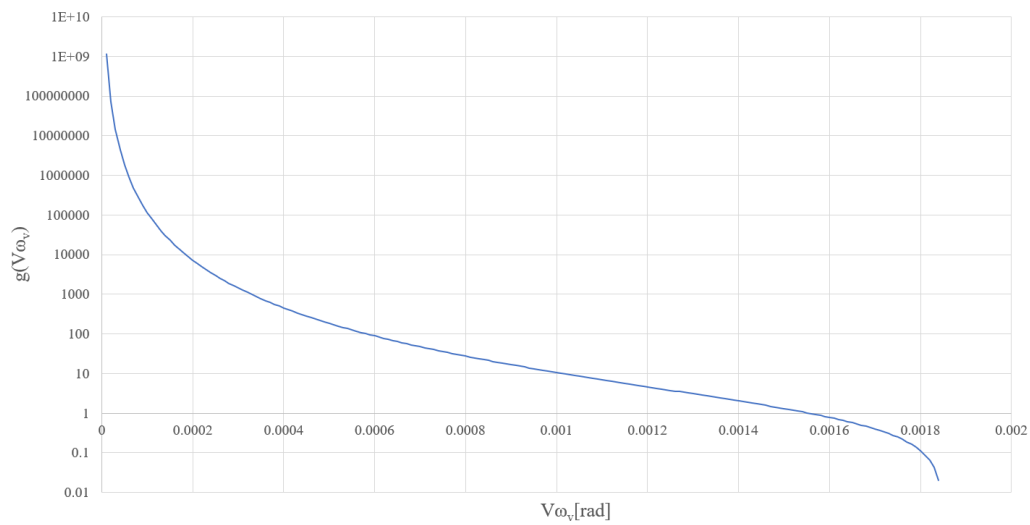


図 5.5 モデルの簡易化手法による  $V_{\omega_v}$  に対する  $g(V_{\omega_v})$  の値のグラフ

次に、モデルのあてはめ手法で算出した  $V_{\omega_v}$  に対して、 $g(V_{\omega_v})$  の値がどのようなになっているか調べた。図 5.6 に、モデルあてはめによる各周波数に対する  $g(V_{\omega_v})$  の値のグラフを示す。

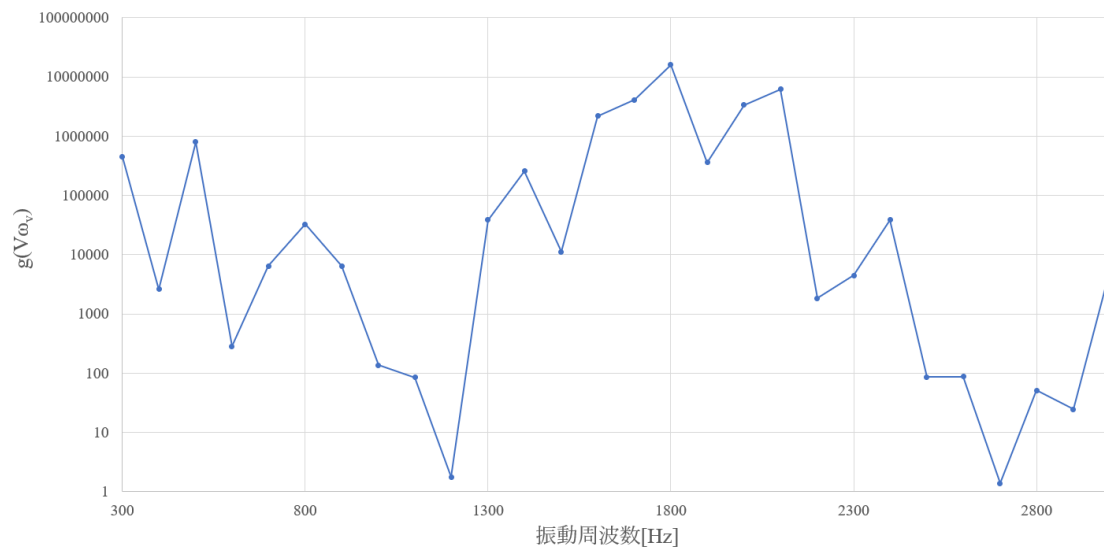


図 5.6 モデルあてはめ手法による振動周波数における  $g(V_{\omega_v})$

図 5.6 より、モデルあてはめの場合では、 $g(V_{\omega_v})$  が 0 より大きい  $V_{\omega_v}$  が推定されていることがわかる。そこで、モデルの簡易化手法において、 $g(V_{\omega_v})$  のしきい値を考えた。図 5.7 に、モデルあてはめ手法による振動周波数における  $V_{\omega_v}$  と、モデルあてはめ手法で用いたデータと同じデータに対してモデルの簡易化手法を適用し、 $g(V_{\omega_v})$  にしきい値を設定した場合の  $V_{\omega_v}$  を同時にプロットしたグラフを示す。

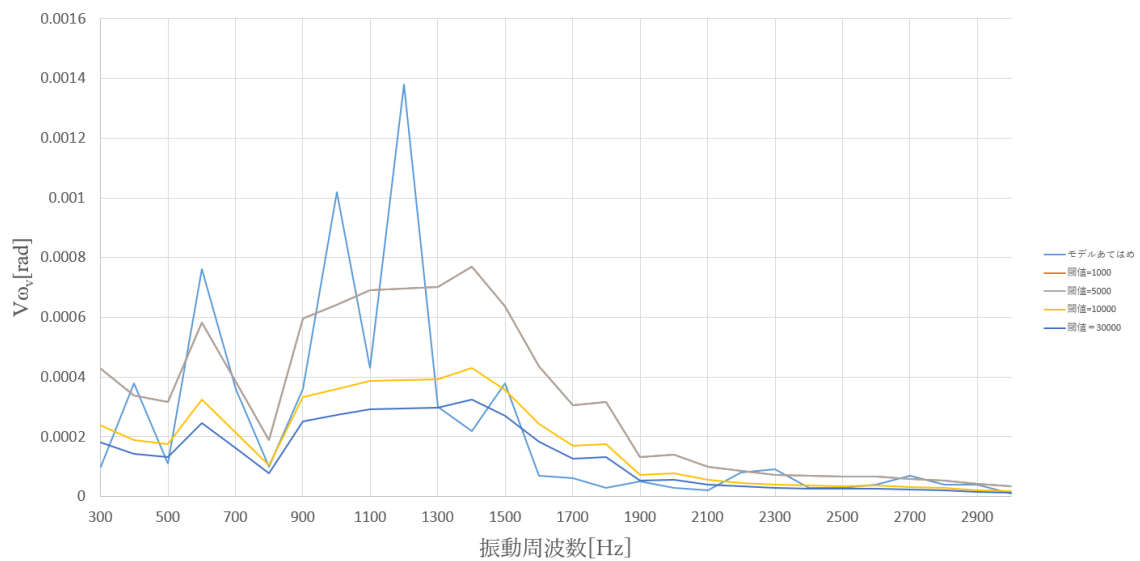


図 5.7 モデルあてはめ手法としきい値を考えた場合のモデル簡易化手法の $V_{\omega_p}$ の比較

2 手法のオーダーを合わせることはできたが、 $V_{\omega_p}$ の値は合わなかった。考えられる原因として、モデル誤差が考慮できていない可能性がある。モデルの簡易化の手法では、近似を用いて振動周波数とその2倍の周波数における光の強度にのみ着目しているが、それ以上の周波数の光の強度についても考慮する必要がある可能性がある。また、マッハツェンダ型光ファイバ振動センサの特徴として、干渉によって強度が大きく変わることがあげられ、強度の変化が影響している可能性がある。

また、モデルあてはめ手法に関しても、計算量が多く時間がかかる関係で3データに対してのモデルあてはめしかできておらず、データが少ないため、より多くのデータに対してモデルあてはめを行うことで、 $V_{\omega_p}$ が変わる可能性がある。

## 第6章 まとめ

### 6.1. 結論

本論文では、ケーブル外部に与えられた振動が、内部の光ファイバに伝達する特性の明確化を検討した。まず、センサファイバおよび参照ファイバの位相変化を加速度の大きさを割った値を振動伝達特性と定義し、モデルパラメータ推定による振動伝達特性の測定手法を提案した。モデルあてはめ手法による振動伝達特性の実測実験により、ケーブル種別により振動伝達特性に違いが生じることが分かった。次に、振動伝達特性の測定の簡易化を目的とし、モデルを簡易化することで振動伝達特性を測定する手法を提案した。モデルあてはめ手法とモデルの簡易化手法で $V_{\omega_v}$ が大きく異なるという問題点があったが、2手法の差を小さくするため、 $g(V_{\omega_v})$ のしきい値を考えた。

実験結果から、ケーブルの詳細な特性がまだわかっていないため、計測方法について見直す必要があることが分かった。

### 6.2. 今後の課題

本研究で提案した2つの手法では、 $V_{\omega_v}$ の値が異なることが分かった。原因としてモデルの簡易化手法においてはモデル誤差やマッハツェンダ型光ファイバ振動センサの特徴による干渉強度の問題、モデルあてはめ手法におけるデータの少なさがあげられる。干渉強度に関しては、よりノイズの入りにくい環境での実験検証による干渉強度の改善を検討する必要がある。モデルあてはめ手法についてはアルゴリズムを見直し、より多くのデータについてモデルあてはめを行い、 $V_{\omega_v}$ の値を算出することを今後の課題とする。

## 参考文献

- [1] Takeshi Nishimura, Kentaro Emoto, Hisashi Nakahara, Satoshi Miura, Mare Yamamoto, Shunsuke Sugimura, Ayumu Ishikawa, Tsunehisa Kimura, "Source location of volcanic earthquakes and subsurface characterization using fiber-optic cable and distributed acoustic sensing system." *Scientific reports* 11.1, 6319 (2021).
- [2] Glenn A. Wellbrock, Tiejun J. Xia, Ming-Fang Huang, Yuheng Chen, Milad Salemi, Yue-Kai Huang, Philip Ji, Ezra Ip, Ting Wang, "First Field Trial of Sensing Vehicle Speed, Density, and Road Conditions by Using Fiber Carrying High Speed Data" 2019 Optical Fiber Communications Conference and Exhibition (OFC), Th4C.7 (2019).
- [3] Xin Lu, Marcus Schukar, Katerina Krebber, "Characterizing vibration response of fiber cables for distributed acoustic sensing" *Optical Fiber Sensors*. Optica Publishing Group, W4.27 (2022).
- [4] Hiroki Fujita, Tetsuya Manabe, Atsushi Nakamura, Yusuke Koshikiya, "Vibration sensor configured in a cable using bidirectional Mach-Zehnder type optical fiber interferometer.", *Proceedings of ICETC2021*, P1-11 (2021).
- [5] Pablo D. Hernández, Jaime A. Ramírez, Marcelo A. Soto, "Improving Earthquake Detection in Fibre-Optic Distributed Acoustic Sensors Using Deep-Learning and Hybrid Datasets." *European Conference and Exhibition on Optical Communication (ECOC)*, Tu4A.2 (2022).
- [6] Grattan, K. T. V., T. Sun., "Fiber optic sensor technology: an overview." *Sensors and Actuators A: Physical* 82.1-3, 40-61 (2000)
- [7] 今井正明, 大塚喜弘, "光ファイバ干渉計測", *光学* 13(2), 153-162 (1984)
- [8] Analog Discovery Pro (DIGILENT)  
<<https://digilent.com/reference/test-and-measurement/analog-discovery-pro-3x50/start>>  
(2024 年 1 月参照)
- [9] 【参考】局内光ファイバの提供例 (NTT 西日本)  
<[https://www.ntt-west.co.jp/news/0108/010828b\\_1.html](https://www.ntt-west.co.jp/news/0108/010828b_1.html)>  
(2024 年 1 月参照)
- [10] ドロップケーブル (KEFR-SSD タイプ) (古河電工)  
<[https://www.furukawa.co.jp/telecom/product/opt\\_fibercable/drop.html](https://www.furukawa.co.jp/telecom/product/opt_fibercable/drop.html)>  
(2024 年 1 月参照)



## 研究業績

- [1] 黒田修平, 真鍋哲也, 中村篤志, 古敷谷優介, “光ケーブル内に構成されたマッハツェンダ型光ファイバセンサへの振動伝達特性の検討”, IEICE Conferences Archives. The Institute of Electronics, Information and Communication Engineers (2023)
- [2] Shuhei Kuroda, Tetsuya Manabe, “Vibration transmission characteristics of an optical fiber sensor using a Mach-Zehnder Interferometer configured in an optical cable”, International Symposium for Sustainability by Engineering at MIU (2023)

## 謝辞

本研究の遂行にあたり，多くの方々にお世話になりました。

指導教官として終始適切なご指導を賜りました，三重大学大学院工学研究科情報工学専攻教授 真鍋哲也先生に深謝いたします。

本論文の作成にあたり，副査として貴重なご助言を賜りました，同専攻教授 成瀬央先生，同専攻准教授 鈴木秀智先生に深く感謝申し上げます。

また，研究物品の手配など，研究室の庶務にかかわる面で大変お世話になりました事務の皆様に厚く感謝申し上げます。

情報通信システム研究室の皆様や友人には，本論文の執筆にあたり多くのご助言，激励をいただきました。ここに感謝の意を示します。

最後に，これまで学生生活を温かく見守り，本学終了まで惜しみない支援をしてくれた家族には，感謝の念に堪えません。本当にありがとうございました。

## 付録 A 実験系のケーブル部について

今回の実測実験では、24 心局内ケーブル、40 心スロット型およびノンスロット型地中ケーブル、および 8 心ドロップケーブルを使用している。その中でも、40 心スロット型地中ケーブルと 8 心ドロップケーブルについては、ケーブル内部に 4 心テープ心線が複数格納されており、光ファイバの選び方によっては構造およびファイバ間の距離の影響から位相差が異なり、振動伝達特性の測定結果にも差が生じると考えられる。今回の実測実験では、どちらのケーブルについても、同じ 4 心テープ心線の中の 2 本の光ファイバを使用し測定を行った。

## 付録 B 実測実験に使用したプログラムについて

今回の実測実験では、モデルのあてはめによる振動伝達特性測定プログラムおよびモデルの簡易化による振動伝達特性測定プログラムの 2 種類のプログラムを作成、使用した。これらの測定プログラムを紹介する。

### B.1 モデルのあてはめによる振動伝達特性測定プログラム

モデルのあてはめによる振動伝達特性測定プログラムを以下に示す。なお、実測実験では、モデルの簡易化による振動伝達特性測定を先に行い、後からモデルのあてはめによる振動伝達特性測定を行った関係から、プログラム中に信号生成やオシロスコープ設定といった項目や、オシロスコープからデータを取得する部分は記述していないため、後述するモデルの簡易化による振動伝達特性測定プログラムの項を参照のこと。

```
%matplotlib notebook
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import pearsonr
import scipy.signal as sg
import csv
import math
import statistics
import heapq
from collections import deque
import random
from scipy import signal
from scipy import fftpack
import pandas as pd
from matplotlib import pyplot as plt
import tqdm
import numpy as np

def calc_amp(data, fs):
    ''' フーリエ変換して振幅スペクトルを計算する関数
    ...
    N = len(data)
    window = signal.windows.hann(N) # ハニング窓
```

```

F = np.fft.fft(data * window)
freq = np.fft.fftfreq(N, d=1/fs) # 周波数スケール
F = F / (N / 2) # フーリエ変換の結果を正規化
F = F * (N / sum(window)) # 窓関数による振幅減少を補正する
Amp = np.abs(F) # 振幅スペクトル
return Amp, freq

def bandpass(x, samplerate, fp, fs, gpass, gstop):
    fn = samplerate / 2 # ナイキスト周波数
    wp = fp / fn # ナイキスト周波数で通過域端周波数を正規化
    ws = fs / fn # ナイキスト周波数で阻止域端周波数を正規化
    N, Wn = signal.buttord(wp, ws, gpass, gstop) # オーダーとバターワースの正規化周波数を計算
    b, a = signal.butter(N, Wn, "band") # フィルタ伝達関数の分子と分母を計算
    y = signal.filtfilt(b, a, x) # 信号に対してフィルタをかける
    return y

# アンプの倍率
mag = 31.6

# 経路の電流
i1 = 1.09620775351042
i2 = 1.09620775351042

# データ長
N = 32768

# サンプリングレート (Hz)
fs = 200 * 500

# サンプリングした時の時間軸
t = np.linspace(0, N/fs, N)

for i in tqdm.tqdm(range(3, 31)):

    freq = i * 100 # 振動周波数

    data_number = [] # データ番号格納用リスト
    app_data_list = [] # 近似データ格納用リスト
    V_list = [] # V 値格納用リスト
    theta_list = [] #  $\theta$  格納用リスト
    mse_min_list = [] # 平均二乗誤差確認リスト
    result_list = [] # 結果出力用リスト

    # 確認用データの取得
    check_file = str(freq) + 'Hz データ確認用.csv'

    with open(check_file, 'r', encoding = "utf-8") as f:
        reader = csv.reader(f)
        l = [row for row in reader]

        l_T = [list(x) for x in zip(*l)]

    for k in range(len(l_T[1])):
        l_T[1][k] = float(l_T[1][k])

    check = l_T[1]

    # V 値が最大値, 最小値, 中央値の時のデータ番号の算出
    for address in range(200):

```

```

if check[address] == max(check):
    max_address = address + 1
    original_max = float(max(check))
elif check[address] == min(check):
    min_address = address + 1
    original_min = float(min(check))
elif check[address] == statistics.median_low(check):
    median_address = address + 1
    original_median = float(statistics.median_low(check))

data_number.append(max_address)
data_number.append(min_address)
data_number.append(median_address)

app_data_list.append(original_max)
app_data_list.append(original_min)
app_data_list.append(original_median)

for j in range(3):
    filename = '元データ/' + str(freq) + 'Hz 元データ(' + str(data_number[j]) + ').csv'
    mse_min = 10000000000
    V = 0
    theta = 0
    offset = 0
    print(filename)

    # チェック用リスト
    check_list = []
    V_check_list = []
    theta_check_list = []
    mse_check_list = []

    # 簡易データチェック用リスト
    theta2_list = []
    mse2_min_list = []

    #
    with open(filename, 'r', encoding="utf-8") as f2:
        reader = csv.reader(f2)
        l = [row for row in reader]

        l_T = [list(x) for x in zip(*l)]

    for k in range(len(l_T[1])):
        l_T[0][k] = float(l_T[0][k]) / mag

    c1 = l_T[0]

    # バンドパスフィルタ
    fa = np.array([200, 3000]) # 通過域端周波数[Hz]
    fb = np.array([100, 6000]) # 阻止域端周波数[Hz]
    gpass = 3 # 通過域端最大損失[dB]
    gstop = 10 # 阻止域端最小損失[dB]

    c1 = bandpass(c1, fs, fa, fb, gpass, gstop)

    V_temp = 0.0

    while V_temp < app_data_list[0]:
        V_temp = V_temp + 0.00001

```

```

V_check_list.append(V_temp)

for o in range(36):

    V_temp = 0.0
    theta_temp = math.pi * o * 10 / 180
    theta_check_list.append(o * 10)

    mse_check_temp = []

    while V_temp < app_data_list[0]:

        V_temp = V_temp + 0.00001 # V の値の幅

        eq = 2 * math.sqrt(i1 * i2) * np.cos(V_temp * np.cos(2 * math.pi * freq *
t) - theta_temp)

        # DC 成分除去
        c1 = c1 - np.mean(c1)
        eq = eq - np.mean(eq)

        mse_min_check = 10000000

        # 平均二乗誤差
        for offset_temp in range(1000):

            c1_fix = c1[offset_temp * 10 : 12000 + (offset_temp * 10)]
            mse = np.mean((eq[:12000] - c1_fix[:12000]) ** 2)

            if mse_min >= mse:
                mse_min = mse

            theta = theta_temp
            V = V_temp
            offset1 = offset_temp

        # データチェック用処理
        if mse_min_check >= mse:
            mse_min_check = mse

        mse_check_temp.append(mse_min_check)

        if V_temp == app_data_list[j]:
            theta2_list.append(theta_temp * 180 / math.pi)
            mse2_min_list.append(mse_min_check)

    mse_check_list.append(mse_check_temp)

df = pd.DataFrame(data = mse_check_list, index = theta_check_list, columns =
V_check_list)
df.to_csv('フィッティング_' + str(freq) + 'Hz データ(' + str(j + 1) + ').csv')

print("モデルフィッティング手法の MSE の min は", mse_min, theta * 180 / math.pi, V)

if mse2_min_list != []:
    print("簡易化したモデルフィッティング手法の MSE の min は", min(mse2_min_list),
theta2_list[mse2_min_list.index(min(mse2_min_list))] * 180 / math.pi, app_data_list[j])

V_list.append(V)
theta_list.append(theta * 180 / math.pi)

```

```

mse_min_list.append(mse_min)

eq_fix1 = 2 * math.sqrt(i1 * i2) * np.cos(V * np.cos(2 * math.pi * freq * t) -
theta)
eq_fix1 = eq_fix1 - np.mean(eq_fix1)

if mse2_min_list != []:
    eq_fix2 = 2 * math.sqrt(i1 * i2) * np.cos(app_data_list[j] * np.cos(2 * math.pi
* freq * t) - theta2_list[mse2_min_list.index(min(mse2_min_list))])
    eq_fix2 = eq_fix2 - np.mean(eq_fix2)

c1_amp, freq_scale = calc_amp(c1, fs)
eq_fix_amp1, freq_scale = calc_amp(eq_fix1, fs)
if mse2_min_list != []:
    eq_fix_amp2, freq_scale = calc_amp(eq_fix2, fs)

fig = plt.figure(figsize=(8, 24))

ax1 = fig.add_subplot(2, 1, 1) # FFT データ比較用
ax2 = fig.add_subplot(2, 1, 2) # 元データ・フィッティングデータ比較用

ax1.clear()
ax1.plot(freq_scale[:2000], eq_fix_amp1[:2000], color="red")
if mse2_min_list != []:
    ax1.plot(freq_scale[:2000], eq_fix_amp2[:2000], color="blue")
ax1.plot(freq_scale[:2000], c1_amp[:2000], color="green")
ax1.set_xlabel("Frequency [Hz]")
ax1.set_title("FFT_data")

ax2.clear()
ax2.plot(t[: (9000//i)], eq_fix1[: (9000//i)], color="red")
if mse2_min_list != []:
    ax2.plot(t[: (9000//i)], eq_fix2[: (9000//i)], color="blue")
ax2.plot(t[: (9000//i)], c1[offset * 10 : (9000//i) + (offset * 10)], color="green")
ax2.set_xlabel("time[s]")
ax2.set_ylabel("Amplitude[V]")

fig.savefig('フィッティング_' + str(freq) + 'Hz データ(' + str(j + 1) + ').png')
plt.close(fig)

#計測データの V 値の標準偏差
V_std = np.std(V_list)
V_mean = np.mean(V_list)
print(f"フィッティングでの各 V 値は {V_list}")
print(f"フィッティングでの各 θ 値は {theta_list}")
print(f"近似での各 V 値は [{original_max}, {original_min}, {original_median}]")
print(f"各 mse 値は {mse_min_list}")

result_list = list(zip(data_number, V_list, theta_list, mse_min_list))
df2 = pd.DataFrame(result_list)
df2.to_csv("フィッティング_" + str(freq) + "Hz データ確認用.csv", header=False,
index=False)

```

## B.2 モデルの簡易化による振動伝達特性測定プログラム

モデルの簡易化による振動伝達特性測定プログラムを以下に示す。

```

from ctypes import *
import ctypes
import time
import datetime
from dwfconstants import *
import dwfconstants as constants
import sys
from IPython.display import display, clear_output
import numpy as np
import math
from collections import deque
import csv
from scipy import signal
import collections
import scipy.signal as sg
from os import sep
import pandas as pd
import copy
import tqdm

def getNearestValue(list, num):
    """
    概要: リストからある値に最も近い値を返却する関数
    @param list: データ配列
    @param num: 対象値
    @return 対象値に最も近い値
    """

    # リスト要素と対象値の差分を計算し最小値のインデックスを取得
    idx = np.abs(np.asarray(list) - num).argmin()
    return idx

def calc_amp(data, fs):
    ''' フーリエ変換して振幅スペクトルを計算する関数
    ...
    N = len(data)
    window = signal.windows.hann(N) # ハニング窓
    F = np.fft.fft(data * window)
    freq = np.fft.fftfreq(N, d=1/fs) # 周波数スケール
    F = F / (N / 2) # フーリエ変換の結果を正規化
    F = F * (N / sum(window)) # 窓関数による振幅減少を補正する
    Amp = np.abs(F) # 振幅スペクトル
    return Amp, freq

def bandpass(x, samplerate, fp, fs, gpass, gstop):
    fn = samplerate / 2 # ナイキスト周波数
    wp = fp / fn # ナイキスト周波数で通過域端周波数を正規化
    ws = fs / fn # ナイキスト周波数で阻止域端周波数を正規化
    N, Wn = signal.buttord(wp, ws, gpass, gstop) # オーダーとバターワースの正規化周波数を計算
    b, a = signal.butter(N, Wn, "band") # フィルタ伝達関数の分子と分母を計算
    y = signal.filtfilt(b, a, x) # 信号に対してフィルタをかける
    return y

chara_l = [] # 振動周波数における振動伝達特性のリスト
freq_l = [] # 振動周波数のリスト

```



```

start = time.perf_counter()

for freq_time in tqdm.tqdm(range(2, 31)):

    # csv データ出力フラグ 0:出力しない 1:出力する
    output_f = 1

    # SDK 使用するための設定
    if sys.platform.startswith("win"):
        dwf = ctypes.cdll.dwf
        constants_path = ":C" + sep + "Program Files (x86)" + sep + "Digilent" + sep +
        "WaveFormSDK" + sep + "samples" + sep + "py"
    elif sys.platform.startswith("darwin"):
        dwf = cdll.LoadLibrary("/Library/Frameworks/dwf.framework/dwf")
    else:
        dwf = cdll.LoadLibrary("libdwf.so")

    #チャンネル取得設定
    #-----変数宣言-----

    N = 32768 #データ長

    wave1 = c_int(0) # 信号生成に用いる変数

    #周波数
    freq_value = freq_time * 100 # 振動周波数
    freq_value2 = freq_value * 2 # 振動周波数の2倍の周波数

    #i1 と i2
    i1 = 1.096207753510206 # I_1 の値 (事前に計測した値)
    i2 = 1.096207753510206 # I_2 の値 (事前に計測した値)

    #増幅器の倍率
    mag = 31.6

    #サンプリングレート (Hz)
    fs = 200 * 500 # 周波数 * 500

    # 振動伝達特性
    chara = 0

    aw_list = [] # 加速度
    chara_tmp_list = [] # 振動伝達特性の一時ファイル

    #-----
    #print(version) #バージョン書いてるだけ
    version = create_string_buffer(16)
    dwf.FDwfGetVersion(version)
    print("Version: "+str(version.value))

    # enumerate connected devices (デバイスの列挙)
    cdevices = c_int()
    dwf.FDwfEnum(c_int(0), byref(cdevices))
    print("Number of Devices: "+str(cdevices.value))

    if cdevices.value == 0:
        print("no device detected")
        quit()

```

```

# デバイス有効化
print("Opening first device")
hdwf = c_int()
dwf.FDwfDeviceOpen(c_int(0), byref(hdwf))

if hdwf.value == hdwfNone.value:
    print("failed to open device")
    quit()

#-----信号生成-----
#dwf.FDwfAnalogOutCount(hdwf, byref(cChannel)) #Wavegen の Channel の数を数える

#Wavegen にて波形作成
#Channel1 の設定 (W1)
print("Configure and start first analog out channel")
dwf.FDwfAnalogOutEnableSet(hdwf, wave1, c_int(1))
#サイン波作成
dwf.FDwfAnalogOutFunctionSet(hdwf, wave1, funcSine)#波の形状
#dwf.FDwfAnalogOutFunctionSet(hdwf, wave1, funcNoise)#波の形状
dwf.FDwfAnalogOutFrequencySet(hdwf, wave1, c_double(freq_value))#周波数
dwf.FDwfAnalogOutNodeAmplitudeSet(hdwf, wave1, c_int(0), c_double(1))#振幅
dwf.FDwfAnalogOutNodeOffsetSet(hdwf, wave1, c_int(0), c_double(0))#オフセット
#dwf.FDwfAnalogOutNodeSymmetrySet(hdwf, wave1, c_int(0), c_double(50))

dwf.FDwfAnalogOutConfigure(hdwf, wave1, c_int(1)) #W1 で音を鳴らす

#-----オシロスコープ設定-----
#dwf.FDwfAnalogInChannelCount(hdwf, byref(cChannel)) #Scope の Channel の数を数える

print("Configure analog in")

sampling_rate = c_double(fs) # サンプリング周波数

dwf.FDwfAnalogInFrequencySet(hdwf, sampling_rate)
print("Set range for all channels")
dwf.FDwfAnalogInChannelEnableSet(hdwf, c_int(-1), c_int(1))
dwf.FDwfAnalogInChannelOffsetSet(hdwf, c_int(-1), c_double(-7.6))
dwf.FDwfAnalogInChannelRangeSet(hdwf, c_int(0), c_double(0.002))
dwf.FDwfAnalogInChannelRangeSet(hdwf, c_int(3), c_double(0.2))
dwf.FDwfAnalogInBufferSizeSet(hdwf, c_int(N))

print("Wait after first device opening the analog in offset to stabilize")
time.sleep(2)

print("Starting acquisition...")
dwf.FDwfAnalogInConfigure(hdwf, c_int(1), c_int(1)) # 取得開始

#-----データ処理部分-----

for count in range(200): # データ取得回数

    t = np.linspace(0, N/fs, N)

    sts = c_int()

```

```

rg1 = (c_double*N) ()
rg4 = (c_double*N) ()

c1 = []
c4 = []
chara_tmp = 0

while True:
    dwf.FDwfAnalogInStatus(hdwf, c_int(3), byref(sts))
    if sts.value == DwfStateDone.value : #Done 状態(sts.value = 2)になったら抜け出し
        break
    time.sleep(0.1)

#各チャンネルからデータ取得
dwf.FDwfAnalogInStatusData(hdwf, c_int(0), rg1, len(rg1)) # get channel 1 data
dwf.FDwfAnalogInStatusData(hdwf, c_int(3), rg4, len(rg4)) # get channel 4 data

#データ長取得
data_length = len(rg1)

#データを配列に格納 そのままだと操作しづらかったため(型が少し違う?)
for i in range(0, data_length):
    c1.append(rg1[i])
    c4.append(rg4[i])

# 元データ書き出し
if output_f == 1:
    origin_list = list(zip(c1, c4))
    df = pd.DataFrame(origin_list)
    df.to_csv('2024-01-24/' + str(freq_value) + 'Hz 元データ' + str(count+1) +
        ').csv', header=False, index=False)

#バンドパスフィルタ
fa = np.array([200, 3000]) # 通過域端周波数[Hz]
fb = np.array([100, 6000]) # 阻止域端周波数[Hz]
gpass = 3 # 通過域端最大損失[dB]
gstop = 10 # 阻止域端最小損失[dB]

c1 = bandpass(c1, fs, fa, fb, gpass, gstop)
c4 = bandpass(c4, fs, fa, fb, gpass, gstop)

#DC 成分除去
c1 = c1 - np.mean(c1)
c4 = c4 - np.mean(c4)

# 各データ FFT
c1_amp, freq_scale = calc_amp(c1, fs)
c4_amp, freq_scale = calc_amp(c4, fs)

# iwv と i2wv と aw の探索
idx1 = getNearestValue(freq_scale, freq_value)
idx2 = getNearestValue(freq_scale, freq_value2)
iwv = c1_amp[idx1] / mag
i2wv = c1_amp[idx2] / mag
aw = c4_amp[idx1]*10

aw_list.append(aw)

# 非線形方程式と振動伝達特性の計算
sigma1 = 0.0
sigma2 = 0.0

```

```

vwv = 0.00001
vwv_array = []

gvwv = []

while True:
    for i in range(100):
        sigma1 = sigma1 + pow(-1, i) / (math.factorial(i)*math.factorial(1+i)) *
        pow(vwv/2, 2*i)
        sigma2 = sigma2 + pow(-1, i) / (math.factorial(i)*math.factorial(2+i)) *
        pow(vwv/2, 2*i)

        j1 = (vwv/2)*sigma1
        j2 = pow(vwv/2, 2)*sigma2

        gvwv.append(pow(i2wv / (4*math.sqrt(i1*i2)*j2), 2) + pow(iwv /
        (4*math.sqrt(i1*i2)*j1), 2) - 1)

        sigma1 = 0.0
        sigma2 = 0.0
        vwv_array.append(vwv)
        vwv = vwv + 0.00001

    if gvwv[-1] < 0:
        gvwv = np.abs(gvwv)

        if gvwv[-1] > gvwv[-2]:
            chara_tmp = vwv_array[-2] / aw

        elif gvwv[-2] > gvwv[-1]:
            chara_tmp = vwv_array[-1] / aw

        break

    chara_tmp_list.append(chara_tmp)

chara = sum(chara_tmp_list) / len(chara_tmp_list)
freq_l.append(freq_value)
chara_l.append(chara)

test_list = list(zip(aw_list, chara_tmp_list))
df2 = pd.DataFrame(test_list)
df2.to_csv("2024-01-24/" + str(freq_value) + "Hz データ確認用.csv", header=False,
index=False)

print(str(freq_value) + "Hz done")
dwf.FDwfAnalogOutReset(hdwf, c_int(0))
dwf.FDwfDeviceCloseAll() #音停止&デバイスから切断

chara_list = list(zip(freq_l, chara_l))
df3 = pd.DataFrame(chara_list)
df3.to_csv("2024-01-24/振動伝達特性.csv", header=False, index=False)

print(time.perf_counter() - start)
print("finish program")

```