卒 業 論 文

機械学習を用いた
音声の伝達特性からの声道断面積の自動抽出

北海道科学大学 工学部
情報工学科
2-15-3-033
指導教員　松﨑 博季
2019 年 (平成 31 年)2 月

目 次

第1章　緒言

音声の音韻性（声道伝達特性）は音源フィルタ理論 [1] より発話時の声道形状により一意的に特徴付けられる。一方、音声から声道形状を求めることは同じ音声波形であっても声道形状が異なる場合があり得るため、一意的に決まらず容易ではない。近年、機械学習により多量のデータを学習させることで、これまで得ることが困難であったデータを得ることが可能になった。そこで、この技術を用いることで音声から声道形状を求めることができれば、声道形状の制御を行う仕組みが求められる。この仕組みを用いて人間と同じ発話機構を有する発話ロボットなどの開発が可能となる。これらのことから本研究では、機械学習を用い、音声情報としての声道伝達特性から声道形状を得ることを試みた結果について報告する。3次元の声道断面積関数を声道形状として用いると情報量が大きく、機械学習による計算コストが高くなり、学習結果を得るまでに多大な時間がかかる。概ね 3Hz の母音であれば、1 次元の近似で声道伝達特性を求められることから、最初の試みとして本報告では、情報量が小さく機械学習による計算コストの低い、1 次元の声道断面積関数を声道形状として扱う。また、学習に必要な多量の声道断面積関数を生成する方法についても述べる。

第2章　音声生成機構

2.1　声道断面積関数および声道伝達特性、音源フィルタ理論について

音声の生成過程は、声門側の片端を閉鎖し、口唇側となるもう一方の端が開いた音響管でモデル化される。声門で発生した音（音源）は声道内で声道形状によって決定される共鳴特性（伝達特性）により変化し、口唇から空気中に放射される。以上より本章では、音声の生成過程を線形モデルで表現した音源フィルタ理論、および声道断面積関数と声道伝達特性について述べる。

2.1.1　音源フィルタ理論

人間は、喉頭における発声（音源）と声道における調音（フィルタ）とが独立であると仮定できる。音源が声道フィルタに入力されると共鳴特性が反映される。その結果出力されるものが音声であると考えることができる。このような線型システムによるモデル化を音源フィルタ理論 [1] と いう。もし、声道形状が一定ならば、声道フィルタは線形時不変となり、入力信号 $x(t)$、インパルス応答 $s(t)$、出力信号 $h(t)$ としたとき以下の式で表せる。ここで、t は時間を表す。

$$h(t) = s(t) * x(t) \tag{2.1}$$

ここで*は、たたみ込み演算を表す。上の式を周波数領域において表現すると、音源のスペクトル $X(\omega)$、声道フィルタのスペクトル $H(\omega)$、声道の伝達関数 $T(\omega)$ と口唇からの放射特性 $R(\omega)$ としたとき以下の式で表せる。ここで ω は周波数を表す。

$$H(\omega) = [T(\omega)R(\omega)]X(\omega) \tag{2.2}$$

2.1.2　声道断面積関数

声門から口唇に向かって伝搬する軸（声道中心線）と中心線に垂直な面と声道の輪郭との交線により声道の断面積形状を近似できる。声道断面積関数とは、この断面積を声道から口

唇に至る中心線上の距離を関数として表したものである [1]。また、声道断面積関数は声道の伝達特性を求める重要なパラメータである。

2.1.3　声道伝達特性

声道伝達特性とは、音源が声道フィルタに入力されると反映される共鳴特性である [1]。その結果出力されるものが音声であると考えることができるため、音声を構成する要素とも言える。

第3章　声道形状を考慮した多量の声道断面積関数の自動作成について

3.1　まえがき

機械学習には、大量のデータが必要である。しかし、声道断面積関数を求める際に、MRI等を用いて多数の人から得るのは困難である。そこで、声道形状を考慮してランダムに声道断面積関数を求めることができれば、容易に多量のデータを得ることができる。そこで本章では、声道形状を考慮した制約条件よりランダムに求めた多量の声道断面積関数の自動作成について示す。

3.2　声道断面積関数のセクション数について

本手法では、成人男性の声道長17cm [2] を0.5cmごとに区切った34セクションを求める。セクション数に関して、神山の論文 [3] より、母音を区別するために必要な3つのホルマントと呼ばれる伝達特性内の特徴を求めるのに影響しないセクション数は20セクション以上あればよいとされている。

3.2.1　声道形状の決定要因

声道形状は、唇、口蓋、咽頭壁、舌、喉頭蓋、披裂軟骨、声帯などの発声器官の位置によって決まる [1]。このことから、本研究では、調音位置として声門、歯列、喉頭、咽頭、軽口蓋、口腔、口唇の7つの位置を求め、声道の形状を求める。

3.3　声道断面積関数の導出方法

導出手順として、まず声道断面積関数の計算のため、文献 [6] の図69などの断面積の模式図を参考に7つの調音位置と声道断面積関数を求める。調音位置以外のセクションの声道断面積関数はスプライン関数 [4,5] を用いて、先に求めた7つの調音位置が滑らかにつながるよう求める。

3.3.1　調音位置の導出方法

声門および口唇の位置を固定し、喉頭、咽頭、軟口蓋、口腔、歯列の位置を乱数により、1〜2 セクション程前後させ 7 セクションの調音位置を求める。また、歯列は出口から 1cm の位置で固定している。以下にそれぞれの位置の導出方法を示す。乱数は 0.0〜1.0 の間の値である。

$$
\begin{aligned}
glottis(\text{声門}) &= 0 && (3.1)\\
larynx\,(\text{喉頭}) &= \frac{1}{seclng} + \frac{1}{seclng * \text{乱数}} && (3.2)\\
velum\,(\text{軟口蓋}) &= \frac{\text{セクション数}}{2} - 1 && (3.3)\\
pharynx\,(\text{咽頭}) &= larynx + \frac{(velum - larynx - 1) * \text{乱数}}{2} && (3.4)\\
teeth\,(\text{歯列}) &= \text{セクション数} - 1 - \frac{1}{seclng} && (3.5)\\
oral\,(\text{口腔}) &= velum + \frac{(teeth - velum - 2) * \text{乱数}}{2} && (3.6)\\
lips\,(\text{口唇}) &= \text{セクション数} - 1 && (3.7)
\end{aligned}
$$

上式より、喉頭 (larynx) を例に seclng=0.5 とした場合、larynx = 2 + 2*乱数 となる。よって喉頭のセクションは 2〜3 という結果が求められる。その他 6 つの調音位置についても同様に求められる。

3.3.2　調音位置における声道断面積

ここでは、先に求めた 7 つのセクションの声道断面積を求める。声道断面積の出力範囲は神山の文献 [3] より 0.45029cm^2〜12.67086cm^2 とする。また、声門の最大値は同文献 [3] より 1.23364cm^2 とする。

3.3.3　多量の声道断面積関数の求めるプログラム

本章 2.1〜2.4 を元に作成したプログラムを用いて多量の声道断面積関数を求める。付録 A にプログラムの詳細を記述する。図 3.1 に本プログラムより求めた声道断面積関数の例を示す。

以上より、声門より 14cm の位置での声道断面積はいずれも絞られていることが分かる。

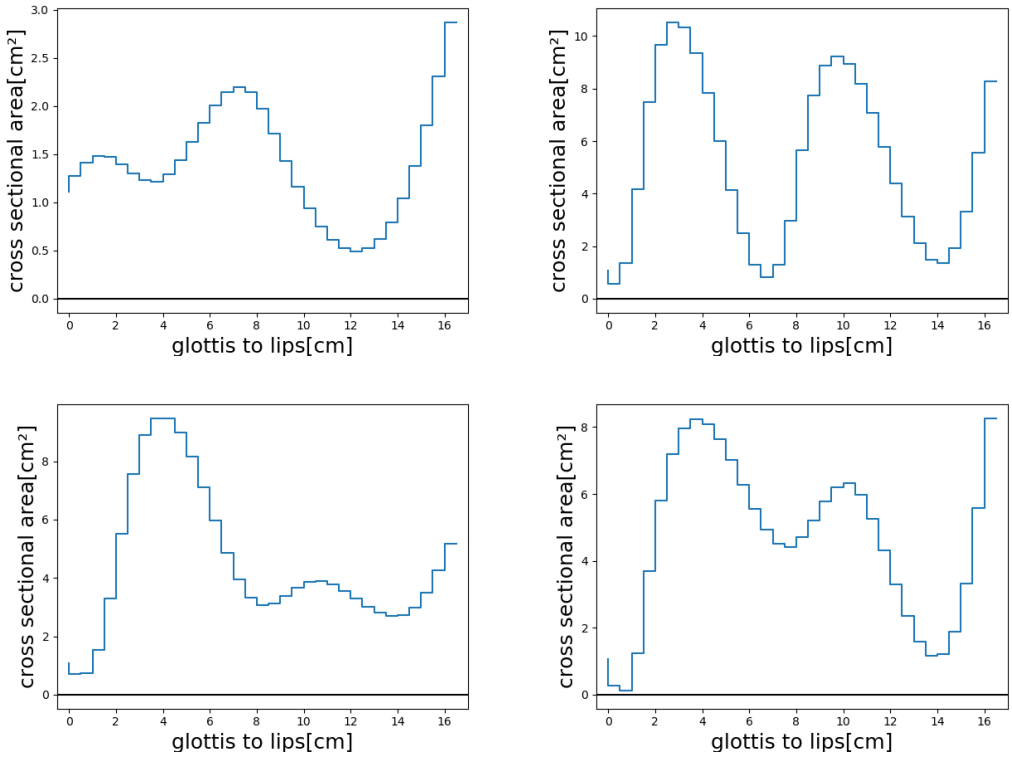

図 3.1: 声道断面積関数の例

第4章　電気等価回路モデルを用いた声道断面積関数からの声道伝達特性の導出方法

4.1　まえがき

本章では、文献 [3] の電気的等価回路モデルを参考にして声道断面積関数から声道伝達特性を求める。

4.2　声道伝達特性の導出方法

第 3 章より求めた声道断面積関数から電気的等価回路モデル [3] を用いて声道伝達特性を導出する方法を以下に示す。

4.2.1　電気的等価回路モデル

文献 [1] の電気的等価回路より、声道伝達特性を求めるには、声道断面積関数は階段関数により近似し、等しい長さ l で分割（本研究では 34 分割）する。その時、1 区間を伝送線路による等価回路として扱った場合、全区間を縦続接続音響管として扱うことができる。このような処理により、声道断面積関数から声道伝達特性を求めることができる。

図 4.1 に上記の手順で求めた声道伝達特性の例を示す。

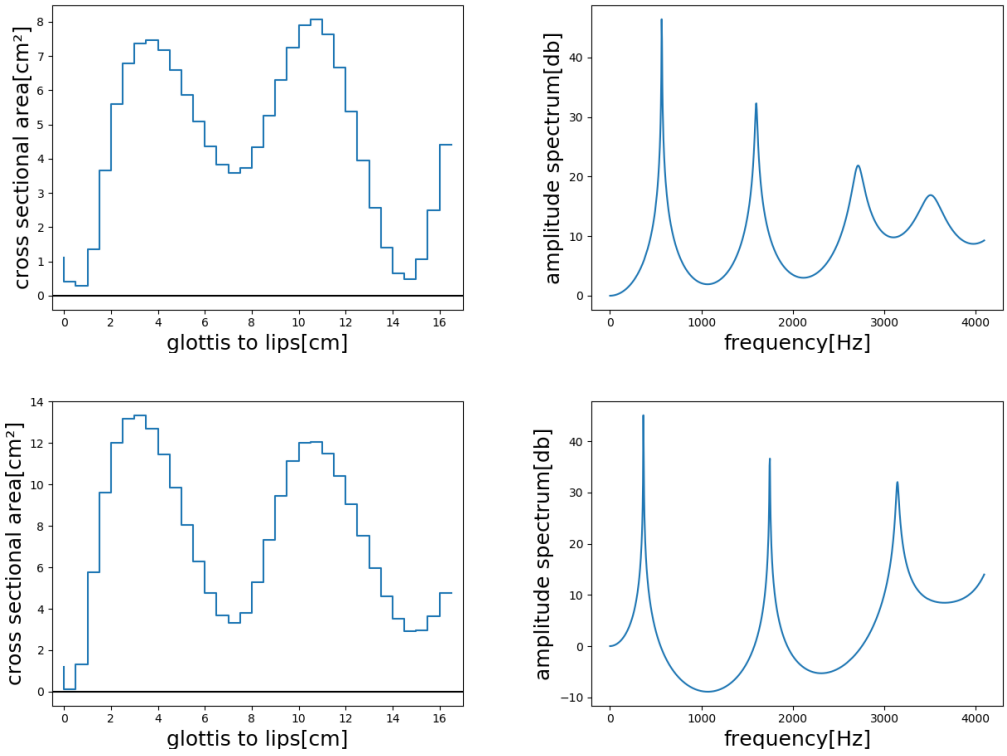

図 4.1: 声道断面積関数から求めた声道伝達特性の例（左図：声道断面積関数、右図：声道伝達特性

第5章　機械学習による声道断面積の出力方法および結果と考察

5.1　まえがき

本研究では、4 章を元に作成した声道伝達特性データを入力とし、3 層ニュートラルネットワークを用いて声道断面積関数の出力を試みる。

5.2　ニューラルネットワークの設定

本研究で用いるニューラルネットワークの設定を本項に示す。

5.2.1　ニューラルネットワークの概略図

ニューラルネットワークの構造を図 5.1 に示す。

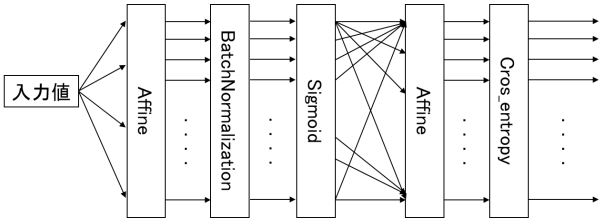

図 5.1: 学習に用いるニューラルネットワークの構造

上図は 3 層ニューラルネットワークを示し、1 層目を Affine、Batch Normalization とし、2 層目を Sigmoid、3 層目を Affine、Cross entropy とした図である。Affine は入力値をそのまま出力する処理であり、Cross entropy は損失関数と呼ばれるものであり、ニューラルネットワークの精度を測定するものである。その他の処理についての詳細は次項以降に示す。

5.2.2　教師あり学習

ニューラルネットによる出力を試みるにあたり、ニューラルネットワークの学習手法の一種である教師あり学習を用いた手法を提案する。ニューラルネットワークとは、機械学習の一種であり、適切な重みパラメータをデータから自動で学習するものである。教師あり学習とは、入力データと教師データ（入力データに対しラベル付けしたデータ）を対としたデータセットを学習させる手法である。本手法では、入力データに声道伝達特性、教師データに声道断面積関数を用いて学習を試みる。

5.2.3　ノード数、学習係数、学習回数およびバッチサイズ

ノード数は入力層、中間層、出力層の順に 1024、50、34 とする。学習係数は 0.001 とし、学習回数は 10000 回とする。バッチサイズを 100 とする。学習係数とは、1 回の学習でどれだけパラメータを更新するかを定めるものである。バッチとは、あるデータセットをひとまとめにして処理する単位である。バッチサイズを入力データ数より小さくすることで学習速度を高めることができる [7]。

5.2.4　活性化関数

活性化関数として式 (5.1) で表される Sigmoid 関数を用い、出力層では恒等関数を用いる。活性化関数とは、入力信号の総和を出力信号に変換する関数のことを表す。ニューラルネットワークでは活性化関数に非線形関数を用いる必要がある [7] ため、非線形関数の一種であり古くからニューラルネットワークの学習に用いられてきた Sigmoid 関数を使用する。恒等関数とは、入力信号をそのまま出力信号とする関数である。

$$h(x) = \frac{1}{1 + exp(-x)} \tag{5.1}$$

5.2.5　重みの初期値

重みの初期値として Xavier らの論文 [8] で推奨される Xvaier の初期値を用いる。重みの初期値とは、ニューラルネットワークの学習の成否が分かれるものとされており [7]、活性化関数として Sigomid 関数を用いる場合は、Xvaier の初期値が推奨されている [7]。Xvaier の初期値は、前層のノード数を n とした場合、$\frac{1}{\sqrt{n}}$ の標準誤差をもつ分布を使うことで、各層のアクティベーション分布（活性化関数の出力）の広がりを同程度にすることができる。この結果、Sigmoid 関数の表現力を制限することなく学習を効率的に進めることが期待できる。

5.2.6　パラメータ更新

重みパラメータの更新には Adam [9] という手法を用いる。この手法は、他のパラメータ更新手法である Momenturm の最適化の更新経路 [7] と AdamGrad [10] の学習係数の衰退という利点を併せ持つ手法であり、ハイパーパラメータの偏りを補正するという特徴を持つ。

5.2.7　Batch Normalization

学習を効率的に進めることを目的に、以下の利点が期待できる Batch Normaliztion [11] を用いる。

- 学習を速く進めることができる（学習係数を大きくできる）
- 初期値にそれほど依存しない
- 過学習を抑制する

過学習とは、あるデータセットにだけ過度に対応した汎化能力が失われている状態のことを言う。Batch Normalization は重みの初期値を強制的にアクティベーション分布を同程度にするよう調節する手法である。具体的には、学習を行う際のミニバッチを単位としてミニバッチごとに、データの分布の平均が 0、分散が 1 になるように正規化する。

5.3　実験結果

下記に、学習後のニュートラルネットワークの認識精度と自動抽出より得た声道断面積を図 5.2、図 5.3 に示す。

図 5.2 はニュートラルネットワークの学習精度の推移を示しており、学習後の精度は約 35 %前後となっている。この学習により得た重みパラメータを用いて出力した声道断面積関数が図 5.3 である。出力された声道断面積関数とテストデータとして使用した声道断面積関数の平均誤差は 140cm^2 であり、最小誤差は 32cm^2、最大誤差は 266cm^2 であった。また、出力された声道断面積関数には負数が含まれていた。

5.4　考察

実験より以下、3 点の問題があることがわかった。ニュートラルネットワークの学習精度が低い点と出力した声道断面積関数の誤差が大きい点および負数が含まれる点。本章では、これらの問題の考察を示す。

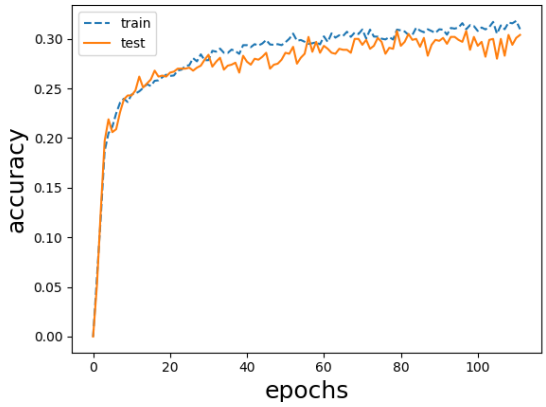

図 5.2: ニュートラルネットワークの精度の推移（実線：テストデータの認識精度、破線：訓練データの認識精度）

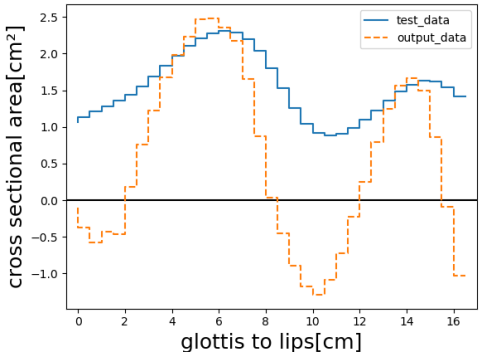

図 5.3: テストデータの声道断面積データと出力された声道断面積データ（実線：テストデータとして用いた声道断面積関数、破線：学習より得た声道断面積関数）

5.4.1　学習精度が低い原因

学習精度が低い原因の 1 つとして以下の要素が考えられる

- 訓練データに過度に対応している（オーバーフィッティング）
- 訓練データに過度に対応していない（アンダーフィッティング）

上記が起こる原因として以下のようなことがある。

オーバーフィッティング	アンダーフィッティング
訓練データ数が少ない	訓練データ数が多い
特徴量の種類が多い	特徴量の種類が少ない

以上より、オーバーフィッティングによる訓練データ数の過小が原因と仮定し、追加実験を行った。追加実験は訓練データおよびテストデータ数のみ変更する。実験内容は訓練データ数を 35000、テストデータを 15000 に変更しそれ以外は同条件にて学習を行った。図 5.4 に追加実験の学習結果を示す。

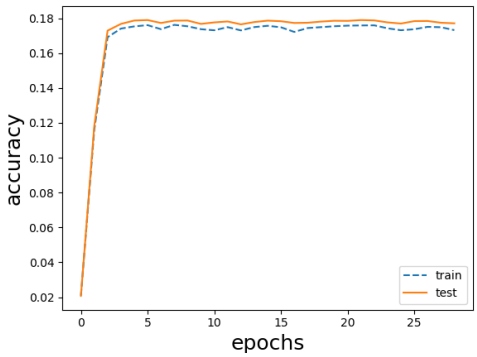

図 5.4: 追加実験結果：ニュートラルネットワークの精度の推移（実線：テストデータの認識精度、破線：訓練データの認識精度）

以上の結果から、データ数を増やした結果、学習精度が低下することが分かる。これらから、訓練データ数およびテストデータ数は当初の設定で適当であり、特徴量の種類が学習精度低下の要因ではないかと考える。

5.4.2　出力された声道断面積関数の誤差が大きい原因

出力結果にニューラルネットワークの学習後に得た重みパラメータを用いていることから出力された声道断面積関数とテストデータの声道断面積関数との誤差が大きい原因としてニューラルネットワークの学習精度が低いことが関与していると考えられる。

5.4.3　出力された声道断面積関数に負数が含まれる原因

出力された声道断面積関数に負数が含まれる原因として、出力結果の誤差が大きい原因と同様、ニューラルネットワークの学習精度に関係があると考える。

第6章　結論

本研究では、3層ニューラルネットワークを用いて伝達特性データから声道断面積関数データの出力を行った。ニュートラルネットワークの学習精度は出力に十分なほど上がらず、適切な声道断面積関数データの出力には至らなかった。原因としては、ニューラルネットワークの認識精度が低いため、出力した結果の精度も低いものなったと考える。本研究ではニュートラルネットワークを用いたが、機械学習では新たな技術の開発が進んでいる。今後はそれらの技術が出力に利用できないか実験と考察を行いたい。

参考文献

[1] 正木信夫, 元木邦俊, 松崎博季, 北村達也, 音声生成の計算モデルと可視化, コロナ社, 2010

[2] Goldstein, U.G. An articulatory model for the vocal tracts of growing children. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, pp.184–197, June. 1980.

[3] 神山直久, 音声生成過程における音響特徴量抽出と実体的声道モデルに関する研究, 北海道大学博士論文, March. 1993.

[4] PolynomialSplineFunction(Commons Math 2.2 API), https://commons.apache.org/proper/commons-math/javadocs/api-2.2/org/apache/commons/math/analysis/polynomials/PolynomialSplineFunction.html

[5] SplineInterpolator(Commons Math 3.0 API), https://commons.apache.org/proper/commons-math/javadocs/api-3.0/org/apache/commons/math3/analysis/interpolation/SplineInterpolator.html

[6] 千葉勉, 梶山正登, 母音-その性質と構造-, 岩波書店, 2003

[7] 斎藤泰毅, ゼロから作る Deep Learning − Python で学ぶディープラーニングの理論と実装, O'REILLY, 2016

[8] Xvaier Glorot, Yoshua Bengio, Understanding the difficulty of training deep feedforward neural nerworks, AISTATS2010, 2010

[9] Diederik Kingma, Jimmy Ba, Adam:A Method for Stochastic Optimization, arXiv:1414.6980, December. 2014

[10] Jhon Duchi, Elad Hazan, Yoram Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Journal of Machine Learning Reasearch,

pp.2121–2159. July. 2011

[11] Sergey Ioffe, Christian Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, arXiv:1502.03167, February. 2015

付 録A　声道断面積関数プログラム

声道断面積関数を求めるプログラムを以下に示す。

```
1  package jp.ac.hus.mhlab;
2
3  import java.util.Random;
4  import static java.lang.Math.*;
5  import org.apache.commons.math3.analysis.interpolation.
       SplineInterpolator;
6  import org.apache.commons.math3.exception.DimensionMismatchException;
7  import org.apache.commons.math3.exception.NonMonotonicSequenceException;
8  import org.apache.commons.math3.exception.NumberIsTooSmallException;
9  import org.apache.commons.math3.exception.OutOfRangeException;;
10 import org.apache.commons.math3.analysis.polynomials.
       PolynomialSplineFunction;
11 import java.io.File;
12 import java.io.FileWriter;
13 import java.io.PrintWriter;
14 import java.io.BufferedWriter;
15 import java.io.IOException;
16 /* 各部位の断面積 [cm2]（千葉＆梶山）
17       gl la P v o t lip
18   /a/ 1.18 1.18 7.45 1.15 3.70 11.30 5.65 8.40
19   /i/ 1.18 1.55 7.94 6.81 7.10 0.80 1.76 4.55
20   /u/ 1.18 1.55 6.59 6.59 2.00 4.75 0.75 1.65
21   /e/ 1.18 1.18 6.59 6.61 3.10 1.90 3.20 5.00
22   /o/ 1.18 1.18 7.30 1.35 2.95 10.65 2.00 1.40
23 */
24
25
26 public class AreaFunctionMaker {
27   private final static double minArea = 0.45029; // 最小面積 [cm2]。神山
         さんのD 論から
28   private final static double maxArea = 12.67086; // 最大面積 [cm2]。神
         山さんのD 論から
29   private final static double maxInitArea = 1.23364; // 入力面の最大面
         積 [cm2]。神山さんのD 論から
30   private int nsec; //セクション数
31   private double length; // 軸長
32   private double[] area; // 面積
33   private double[] dist; // 入力からの距離
```

```
34      private double seclng;
35      private int iglottis;
36      private int ilarynx;
37      private int ipharynx;
38      private int ivelum;
39      private int ioral;
40      private int iteeth;
41      private int ilips;
42      private int dataset = 10000;
43
44      public AreaFunctionMaker() {
45        this(34, 17.0); // 1セクション0.5cmとすると17cmで34セクション
46      }
47
48      public AreaFunctionMaker(int nsec, double length) {
49        this.nsec = nsec;
50        this.length = length;
51        area = new double[nsec];
52        dist = new double[nsec];
53
54        seclng = length / nsec;
55
56        for(int i=0; i<nsec; i++) {
57          dist[i] = seclng * i;
58          //System.out.println("deit[" + i + "]=" + dist[i]);
59        }
60
61        setSection();
62      }
63
64      private void setSection(){
65        iglottis = 0;
66        Random rand = new Random();
67        ilarynx = (int) (1.0 / seclng) + (int) (1.0 / seclng *random());
68        //ipharynx = (int) (length / 3.33333 / seclng) - 1;
69        ivelum = nsec / 2 - 1;
70        //ipharynx = ilarynx + (int)((ivelum - ilarynx - 1)*random()) + 1;
71        ipharynx = ilarynx + (int)((ivelum - ilarynx - 1)*random())/2;
72        //ioral = (int) (length / 1.41421356 / seclng) - 1;
73        iteeth = nsec - 1 - (int) (1.0 / seclng); // 歯は出口から1
              cmの位置で固定
74        //ioral = ivelum + (int)((iteeth - ivelum - 2)*random()) + 1;
75        ioral = ivelum + (int)((iteeth - ivelum - 2)*random())/2;
76        ilips = nsec - 1;
77      }
78
79      public void calculate(){
80        boolean flag = true;
```

```
81        int count=0;
82        while(count<dataset){
83          double[] x = new double[7];
84          double[] y = new double[7];
85              x[0]=dist[iglottis];
86              x[1]=dist[ilarynx];
87              x[2]=dist[ipharynx];
88              x[3]=dist[ivelum];
89              x[4]=dist[ioral];
90              x[5]=dist[iteeth];
91              x[6]=dist[ilips];
92
93              y[0]=1.0 + 0.2*random();
94              y[1]=1.2 + (1.55 - 1.2) * random();
95              y[2]=1.15 + (7.94 - 1.15) * random();
96              y[3]=2.0 + (7.1 - 2.0) * random();
97              y[4]=0.8 + (11.3 - 0.8) * random();
98              y[5]=0.75 + (5.65 - 0.75) * random();
99              y[6]=1.4 + (8.4 - 1.4) * random();
100             //for(int i=0; i< x.length; i++){
101             // System.out.print("x[" + i + "]=" + x[i]);
102             // System.out.println(", y[" + i + "]=" + y[i]);
103             //}
104
105             SplineInterpolator si = new SplineInterpolator();
106             try {
107               PolynomialSplineFunction psf = si.interpolate(x, y);
108               for (int i=0; i<area.length; i++){
109                 area[i] = psf.value(dist[i]);
110             if(area[i]<0.0) {
111               flag = false;
112               break;
113             }
114                 //System.out.println(dist[i] + " " + area[i]);
115                 //System.out.println(i + " " + area[i]);
116               }
117             } catch (OutOfRangeException ore) {
118           //ore.printStackTrace();
119           flag = false;
120             } catch (Exception ex) {
121           flag = false;
122           //ex.printStackTrace();
123             }
124
125         if(flag) {
126             String dataname = "" + count;
127             try{
128                 String dir = System.getProperty("dir","C:\\Users\\Masataka
```

```
                    \\Documents\\NetBeansProjects\\Peaks\\area");
129                 File file = new File(dir,dataname);
130                 FileWriter fw = new FileWriter(file);
131                 BufferedWriter bw = new BufferedWriter(fw);
132                 PrintWriter pw = new PrintWriter(bw);
133
134                 pw.format("%f",17.0);
135                 pw.format("\n");
136                 pw.format("%d", area.length);
137                 pw.format("\n");
138                 for (int i=0; i<area.length; i++){
139                     System.out.println(area[i]);
140                     pw.format(" %5f ",area[i]);
141                     pw.format("\n");
142                 }
143                 bw.close();
144           }
145             catch(IOException ex){
146                 System.out.println(ex);
147             }
148           System.out.println(count);
149           count++;
150         }
151
152         flag=true;
153         setSection();
154       }
155     }
156
157     public void show(){
158       System.out.println("iglottis: " + iglottis);
159       System.out.println("ilarynx: " + ilarynx);
160       System.out.println("ipharynx: " + ipharynx);
161       System.out.println("ivelum: " + ivelum);
162       System.out.println("ioral: " + ioral);
163       System.out.println("iteeth: " + iteeth);
164       System.out.println("ilips: " + ilips);
165     }
166
167     public static void main(String args[]) {
168       AreaFunctionMaker af = new AreaFunctionMaker();
169       //af.show();
170       af.calculate();
171     }
172   }
```

付 録B　声道伝達特性のプログラム

声道伝達特性を求めるプログラムを以下に示す。

```
/*********************************************************

        peaks : 声道伝達関数計算プログラム

   CC peaks.C calf.C radz.C news.C -lcomplex -o peaks

   pea.com でコンパイルできる。

   ヘッダーファイルの例は ex.hed。


   (C)opyright NAOHISA KAMIYAMA DEC. 1991

 *********************************************************/
package jp.ac.hus.mhlab;
import java.io.IOException;
import java.io.File;
import java.io.FileReader;
import java.io.PrintWriter;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileWriter;
import static java.lang.Math.*;
import org.apache.commons.math3.transform.*;
import org.apache.commons.math3.complex.*;
import org.apache.commons.math3.exception.*;

import java.util.Scanner;

public class Peaks {
  private WallImpedance ZW, ZWN;
  private double freq, frest, finalf, dit, Na1, Ndit;
  private int cal_tag, Ef_tag, nasal_tag;
  private int nareav, icntlr, Nport, Nsec;
  private String areav, arean, filekp, fileV, fileN;
  private double amp_1, amp_2, amp_3, fin_1, fin_2, fin_3;
  private double vleng, vlg, Vvleng, Nvlg;
  private double[] radin, radout, sfact, Vradin, Vradout, Vsfact;
```

```
39      private double[] fdat = new double[4000]; //*double → int
40      private double[] abdat = new double[4000]; //*double → int
41      private double[] Nabdat = new double[4000]; //*double → int
42      private double[] Sabdat = new double[4000]; //*double → int
43      //private Complex gam, fimp, Vgam, Vfimp;
44      private int ntimes;
45      private Complex hvtra, Nhvtra;
46
47      private double temper = 25.0;
48
49      public Peaks(File file, int a) throws IOException{
50        //try {
51        FileReader fr = new FileReader(file);
52        BufferedReader br = new BufferedReader(fr);
53        String data;
54        while((data = br.readLine()) != null) {
55          String[] tmp = data.split("[ ]+");
56          for(int i=0; i<tmp.length; i++){
57            //System.out.println(tmp[i]);
58            switch(tmp[i]) {
59            case "FinalFreq":
60              finalf = Double.parseDouble(tmp[++i]);
61              break;
62            case "CalSpeed":
63              cal_tag = Integer.parseInt(tmp[++i]);
64              break;
65            case "AreaName":
66              areav = tmp[++i];
67              //bufferedWriter.write("1");
68              //bufferedWriter.close();
69              break;
70            case "Ef_tag":
71              Ef_tag = Integer.parseInt(tmp[++i]);
72              break;
73            case "LossMode":
74              icntlr = Integer.parseInt(tmp[++i]);
75              break;
76            case "ZwallParam":
77              ZW = new WallImpedance(Double.parseDouble(tmp[++i]),
78                                     Double.parseDouble(tmp[++i]),
79                                     Double.parseDouble(tmp[++i]),
80                                     Double.parseDouble(tmp[++i]),
81                                     Double.parseDouble(tmp[++i]));
82              break;
83            case "Thickness":
84              dit = Double.parseDouble(tmp[++i]);
85              break;
86            case "NasalFile":
```

```
87              arean = tmp[++i];
88              break;
89            case "ConnecSec":
90              Nport = Integer.parseInt(tmp[++i]);
91              break;
92            case "ConnecArea":
93              Na1 = Double.parseDouble(tmp[++i]);
94              break;
95            case "ZwnPara":
96              ZWN = new WallImpedance(Double.parseDouble(tmp[++i]),
97                                      Double.parseDouble(tmp[++i]),
98                                      Double.parseDouble(tmp[++i]),
99                                      Double.parseDouble(tmp[++i]),
100                                     Double.parseDouble(tmp[++i]));
101             break;
102           case "ThickN":
103             Ndit = Double.parseDouble(tmp[++i]);
104             break;
105           case "Ofile":
106             filekp = tmp[++i];
107             break;
108           }
109         }
110       }
111       fr.close();
112       //} catch (IOException e) {
113       // e.printStackTrace();
114       //}
115       /*String[] tmp = new String[3];
116       finalf = 4096; //FinalFreq
117       cal_tag = 4; //CalSpeed
118       areav = "0"; //AreaName
119       tmp[0] = areav;
120       Ef_tag = 0; //Ef_tag
121       icntlr = 0; //LossMode
122       ZW = new WallImpedance(3700.0, 1800.0, 2900.0, 2.2, 410.0); //
              ZwallParam
123       dit = 0.0; //Thickness
124       arean = "no"; //NasalFile
125       tmp[1] = arean;
126       Nport = 18; //ConncSec
127       Na1 = 0.4; //ConnecArea
128       ZWN = new WallImpedance(3700.0, 1800.0, 2900.0, 2.2, 410.0); //
              ZwnPara
129       Ndit = 2.0; //ThickN
130       filekp = "ddd"; //Ofile
131       tmp[2] = filekp;
132   */
```

```
133       if(cal_tag==4){ // added by Matsuzaki
134         frest = 4.0; // added by Matsuzaki
135         finalf+=1.0;
136       } else { // added by Matsuzaki
137         frest= 20.0;
138         if(finalf<=0.0) finalf=8000.0;
139         if(cal_tag<=0) cal_tag=2;
140         finalf+=40.0;
141         CALTAG(cal_tag);
142       } // added by Matsuzaki
143
144       if((dit<0.0)||(icntlr !=2)) dit=0.0;
145       ZW.setGw(ZW.getGw()*1000.0);
146
147       ////////////////////////////////////////
148       if(icntlr==3){ // 裏技
149         System.out.println(">Wall Impedance Parameter");
150         System.out.println(" [R],[L],[K], [800, 2.1, 84500] ===> ");
151         Scanner scan = new Scanner(System.in);
152         //cin>> ZW.R1; if(ZW.R1<=0.0){ ZW.R1=800.;ZW.Lo=2.1 ;ZW.Gw
                =84500.;}
153         // else { cin >> ZW.Lo; cin >> ZW.Gw;}
154         ZW.setR1(scan.nextDouble());
155         if(ZW.getR1()<=0.0){ ZW.setR1(800.0); ZW.setLo(2.1); ZW.setGw
                (84500.0);}
156         else {
157           ZW.setLo(scan.nextDouble());
158           ZW.setGw(scan.nextDouble());
159         }
160       }
161       ////////////////////////////////////////
162       readAreaFunctionData();
163       if(!arean.equals("no")) {
164         readNasalAreaFunctionData();
165         nasal_tag = 1;
166       }
167       else
168         nasal_tag = 0;
169
170       /***** 出力ファイル *****/
171       fileV = filekp + ".v";
172       fileN = filekp + ".n";
173
174        FileWriter fw = new FileWriter(file);
175       BufferedWriter bw = new BufferedWriter(fw);
176       PrintWriter pw =new PrintWriter(bw);
177       pw.format("FinalFreq");
178       pw.format(" 4096");
```

```
179       pw.format("\n");
180       pw.format("CalSpeed");
181       pw.format(" %s", cal_tag);
182       pw.format("\n");
183       pw.format("AreaName");
184       pw.format(" %s", ++a);
185       pw.format("\n");
186       pw.format("Ef_tag");
187       pw.format(" %s", Ef_tag);
188       pw.format("\n");
189       pw.format("LossMode");
190       pw.format(" %s", icntlr);
191       pw.format("\n");
192       pw.format("ZwallParam");
193       pw.format(" 3700.0 1800.0 2900.0 2.2 410.0");
194       pw.format("\n");
195       pw.format("Thickness");
196       pw.format(" %s", dit);
197       pw.format("\n");
198       pw.format("NasalFile");
199       pw.format(" no");
200       pw.format("\n");
201       pw.format("ConnecSec");
202       pw.format(" %s", Nport);
203       pw.format("\n");
204       pw.format("ConnecArea");
205       pw.format(" %s", Na1);
206       pw.format("\n");
207       pw.format("ZwnPara");
208       pw.format(" 3700.0 1800.0 2900.0 2.2 410.0");
209       pw.format("\n");
210       pw.format("ThickN");
211       pw.format(" %s", Ndit);
212       pw.format("\n");
213       pw.format("Ofile");
214       pw.format(" %s", a);
215       pw.format("\n");
216
217       bw.close();
218     }
219
220     public void showHelp() {
221       System.out.println(" o..... P E A K S (C) N.KAMIYAMA 1991 .....o%n
              ");
222
223       System.out.println("******* Hedder Format *******");
224       System.out.println("FianlFreq [freq]");
225       System.out.println("Calculate Speed [1~3]");
```

```
226       System.out.println("FileName [Area-curcum format file]");
227       System.out.println("Ef Mode [1/0]");
228       System.out.println("LossMode [0:Lossless 1:Air 2:Zwall]");
229       System.out.println("Zwall Parameter [R1][R2][R3][Lo][Gw(xe3)]");
230       System.out.println("WallThickness [dit]");
231       System.out.println("NasalName [name]");
232       System.out.println("ConnectSec [int]");
233       System.out.println("ConnectArea [Area]");
234       System.out.println("Zwn Param (if Loss mode = 2) [R1][R2][R3][Lo][
              Gw(xe3)]");
235       System.out.println("WallThickness [dit]");
236       System.out.println("Outputfile [name]");
237       System.out.println("*****************************");
238     }
239
240     public void showStatus(){
241       System.out.println(" * ************** STATUS *************** *");
242       System.out.printf("\tInput File : %s(%dsec.)%n", areav, nareav);
243       System.out.printf("\tOutput Fie : %s%n", filekp);
244       System.out.printf("\tLoss Mode : %d%n", icntlr);
245       if(icntlr == 2) {
246         System.out.printf("\t Zw R1:%5.1f R2:%5.1f R3:%5.1f%n", ZW.getR1
              (), ZW.getR2(), ZW.getR3());
247         System.out.printf("\t Lo:%5.1f Gw:%f %n",ZW.getLo(), ZW.getGw
              ());
248         System.out.printf("\t Thickness : %f%n",dit);
249       }
250       System.out.printf("\tElongation : %d%n",Ef_tag);
251       if(!arean.equals("no")){
252         System.out.printf("\tNasal File : %s(%dsec.)%n", arean, Nsec);
253         System.out.printf("\t Connect Sec : %d%n", Nport);
254         System.out.printf("\t Connect Area: %5.2f%n", Na1);
255         if(icntlr == 2){
256           System.out.printf("\t Zwn R1:%5.1f R2:%5.1f R3:%5.1f%n", ZWN.
                getR1(), ZWN.getR2(), ZWN.getR3());
257           System.out.printf("\t Lo:%5.1f Gw:%f %n", ZWN.getLo(), ZWN.
                getGw());
258           System.out.printf("\t Thickness : %f%n", Ndit);
259         }
260       }
261       else System.out.println("\tNasal File : <None>");
262       System.out.println(" * ************************************
              *");
263     }
264
265     private void CALTAG(int n) {
266       switch (n) {
267       case 1:
```

```
268         amp_1= 8.0; fin_1=78.0;
269         amp_2=14.0; fin_2=48.0;
270         amp_3=18.0; fin_3=16.0;
271         break;
272       case 2:
273         amp_1= 5.0; fin_1=30.0;
274         amp_2=10.0; fin_2=20.0;
275         amp_3=16.0; fin_3=10.0;
276         break;
277       case 3:
278         amp_1= -10.0; fin_1=16.0;
279         amp_2= 3.0; fin_2=8.0;
280         amp_3=10.0; fin_3=4.0;
281         break;
282       }
283     }
284
285     //* ----------------- READ AREA FUNCTION DATA
            -------------------------*/
286     public void readAreaFunctionData(){
287         readAreaFunctionData(areav);
288     }
289     public void readAreaFunctionData(String filename){
290       String dir = System.getProperty("dir","C:\\Users\\Masataka\\
              Documents\\NetBeansProjects\\Peaks\\area");
291       File file = new File(dir,filename);
292       if(!file.exists()) {
293         System.out.println("ファイルが存在しません。");
294         System.exit(1);
295       }
296
297       try {
298         FileReader fileReader = new FileReader(file);
299         BufferedReader bufferedReader = new BufferedReader(fileReader);
300         String data;
301         //while((data = bufferedReader.readLine()) != null) {
302         // System.out.println(data);
303         //}
304         data = bufferedReader.readLine();
305         vleng = Double.parseDouble(data);
306         data = bufferedReader.readLine();
307         nareav = Integer.parseInt(data);
308
309         radin = new double[nareav];
310         radout = new double[nareav];
311         sfact = new double[nareav];
312
313         double[] area = new double[nareav];
```

```
314         double[] circ = new double[nareav];
315         for(int i=0; i<nareav; i++){
316           data = bufferedReader.readLine();
317           String[] tmp = data.split("[ ]+");
318           //for(int j=0; j<tmp.length; j++){
319           // System.out.println("i=" + i + "tmp[" + j + "]=" + tmp[j]);
320           //}
321           area[i] = Double.parseDouble(tmp[1]);
322           //circ[i] = Double.parseDouble(tmp[2]);
323           //System.out.println(i + " " + area[i] + " " + circ[i]);
324         }
325         fileReader.close();
326
327         for(int i=0; i<nareav; i++) {
328           radin[i] = sqrt(area[i]/PI);
329           radout[i] = radin[i] + dit;
330         }
331
332         if(Ef_tag == 1)
333           for(int i=0; i<nareav; i++) sfact[i]= circ[i]/(2.0*sqrt(PI*
                  area[i]));
334         else
335           for(int i=0; i<nareav; i++) sfact[i]=1.0;
336
337         /*=========================================*/
338         revar(radin); revar(radout);
339         /*=========================================*/
340         vlg=vleng/nareav;
341         System.out.println("vlg=" + vlg);
342       } catch (IOException e){
343         e.printStackTrace();
344         System.exit(1);
345       }
346     }
347
348     /*---------------------- Nasal Part --------------------------*/
349     public void readNasalAreaFunctionData(){
350       readNasalAreaFunctionData(arean);
351     }
352     public void readNasalAreaFunctionData(String filename){
353       File file = new File(filename);
354       if(!file.exists()) {
355         System.out.println("ファイルが存在しません。");
356         System.exit(1);
357       }
358
359       try {
360         FileReader fileReader = new FileReader(file);
```

```
361         BufferedReader bufferedReader = new BufferedReader(fileReader);
362         String data;
363         //while((data = bufferedReader.readLine()) != null) {
364         // System.out.println(data);
365         //}
366         data = bufferedReader.readLine();
367         Vvleng = Double.parseDouble(data);
368         data = bufferedReader.readLine();
369         Nsec = Integer.parseInt(data);
370
371         Vradin = new double[Nsec];
372         Vradout = new double[Nsec];
373         Vsfact = new double[Nsec];
374
375         double[] area = new double[Nsec];
376         double[] circ = new double[Nsec];
377         for(int i=0; i<Nsec; i++){
378           data = bufferedReader.readLine();
379           String[] tmp = data.split("[ ]+");
380           //for(int j=0; j<tmp.length; j++){
381           // System.out.println("i=" + i + "tmp[" + j + "]=" + tmp[j]);
382           //}
383           area[i] = Double.parseDouble(tmp[1]);
384           //circ[i] = Double.parseDouble(tmp[2]);
385           //System.out.println(i + " " + area[i] + " " + circ[i]);
386         }
387         fileReader.close();
388
389         area[0]=Na1; /* まだArea[cm2]. */
390         Nport = nareav - Nport;
391
392         for(int i=0; i<Nsec; i++){
393           Vradin[i]=sqrt(area[i]/PI);
394           Vradout[i]= Vradin[i]+ Ndit;
395         }
396         if(Ef_tag == 1)
397           for(int i=0; i<Nsec; i++) Vsfact[i]= circ[i]/(2.0*sqrt(PI*area
                  [i]));
398         else
399           for(int i=0; i<Nsec; i++) Vsfact[i]=1.0;
400
401         /*-------------------------------------*/
402         revar(Vradin); revar(Vradout);
403         /*-------------------------------------*/
404
405         System.out.printf("Vrad %f%n", Vradin[Nsec-1]);
406
407         Nvlg=Vvleng/Nsec;
```

```
408       } catch (IOException e){
409         e.printStackTrace();
410         System.exit(1);
411       }
412     }
413
414     /************* revar.c ****************/
415     private void revar(double[] data)
416     {
417       int N = data.length;
418       double[] tmp = new double[N];
419
420       for(int i=0 ; i<N ; i++) tmp[i]=data[i];
421       for(int i=0 ; i<N ; i++) data[i]=tmp[N-i-1];
422     }
423
424     public void calculate(){
425       System.out.println("\t!!!!! NOW CALCULATING !!!!!");
426
427       freq=frest;
428       ntimes=0;
429       hvtra = null; Nhvtra = null;
430
431       while(freq<finalf){
432   // ----------------------------------------
433         if(nasal_tag ==1) {
434           VocalAndNose(hvtra, Nhvtra); /* NASAL VERSION */
435           Nabdat[ntimes]= (20.0*log10(Nhvtra.abs()));
436           abdat[ntimes] = (20.0*log10(hvtra.abs()));
437           Sabdat[ntimes]= (20.0*log10(Nhvtra.abs() + hvtra.abs()));
438         }
439         else {
440           //Vocal(hvtra); /* VOCAL VERSION */
441           Vocal(); /* VOCAL VERSION */
442           Sabdat[ntimes]= (20.0*log10(hvtra.abs()));
443         }
444   // ----------------------------------------
445
446         fdat[ntimes]= freq; //*double → int
447
448         // if(((ntimes)%40)==0) printf("%4.0f %n",freq);
449         if(cal_tag==4){ // added by Matsuzaki
450           freq += 4.0; // added by Matsuzaki
451         } else { // added by Matsuzaki
452           if(Sabdat[ntimes]< amp_1) freq+=fin_1;
453           else{ if(Sabdat[ntimes]< amp_2) freq+=fin_2;
454             else{ if(Sabdat[ntimes]< amp_3) freq+=fin_3;
455               else freq+=2.0;}}
```

```
      }
      ntimes+=1;
    }
  }

  public void saveFile(){
    /* ファイルへの書き込み */
    try{
      if(nasal_tag ==1) {
        String dir = System.getProperty("dir","C:\\Users\\Masataka\\Documents\\NetBeansProjects\\Peaks\\peaks");
        File file_V = new File(dir,fileV);
        PrintWriter pwV = new PrintWriter(file_V);
        pwV.format("%d%n", ntimes);
        for(int i=0; i<ntimes ; i++) pwV.format("%f%n", abdat[i]);
        pwV.format("%n");
        pwV.close();

        File file_N = new File(dir,fileN);
        PrintWriter pwN = new PrintWriter(file_N);
        pwN.format("%d%n", ntimes);
        for(int i=0; i<ntimes ; i++) pwN.format("%f%n", Nabdat[i]);
        pwN.format("%n");
        pwN.close();
        System.out.println("File: " + fileV);
        System.out.println("File: " + fileN);
      }

     String dir = System.getProperty("dir","C:\\Users\\Masataka\\Documents\\NetBeansProjects\\Peaks\\peaks");
      File file = new File(dir,fileV);
      PrintWriter pwkp = new PrintWriter(file);
      //pwkp.format("%d%n", ntimes);
      for(int i=0; i<ntimes; i++){
        //fdat[i]=fdat[i] * 0.001;
        fdat[i]=fdat[i];
        pwkp.format("%f%n", Sabdat[i]);
      }
      //pwkp.format("%n");
      pwkp.close();
      System.out.println("File: "+ filekp + " is made !");
    } catch (IOException ex){
      ex.printStackTrace();
    }
  }

///*-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-*/
```

```
///* KEEP IN THE LOG FILE "Log.Peaks"
///*-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-*/

//
//fp=fopen("Log.Peaks","a");
//
// fprintf(fp," * ************** STATUS *************** *%n");
// fprintf(fp,"\tInput File : %s%n",areav);
// fprintf(fp,"\tOutput Fie : %s%n",filekp);
// fprintf(fp,"\tLoss Mode : %d%n",icntlr);
// if(icntlr == 2)
// {
// fprintf(fp,"\t Zw R1:%5.1f R2:%5.1f R3:%5.1f%n",ZW.R1, ZW.R2, ZW.R3);
// fprintf(fp,"\t Lo:%5.1f Gw:%f %n",ZW.Lo, ZW.Gw);
// fprintf(fp,"\t Thickness : %f%n",dit);
// }
// fprintf(fp,"\tElongation : %d%n",Ef_tag);
// if(strcmp(arean,"no") != 0)
// {
// fprintf(fp,"\tNasal File : %s%n",arean);
// fprintf(fp,"\t Connect Sec : %d%n",Nport);
// fprintf(fp,"\t Connect Area: %5.2f%n",Na1);
// if(icntlr == 2)
// {
// fprintf(fp,"\t Zw R1:%5.1f R2:%5.1f R3:%5.1f%n",ZWN.R1, ZWN.R2, ZWN.R3);
// fprintf(fp,"\t Lo:%5.1f Gw:%f %n",ZWN.Lo, ZWN.Gw);
// fprintf(fp,"\t Thickness : %f%n",Ndit);
// }
// }
// else fprintf(fp,"\tNasal File : <None> %n%n");
// fclose(fp);
//
//
    //-------------- End of main. -------------------------

  private void VocalAndNose(Complex hvtra, Complex Nhvtra){
    double area1=radin[0]*radin[0]*PI;
    double Varea1=Vradin[0]*Vradin[0]*PI;
    Complex[] gam = new Complex[nareav];
    Complex[] fimp = new Complex[nareav];
    Complex[] Vgam = new Complex[Nsec];
    Complex[] Vfimp = new Complex[Nsec];
    Complex Vzl, zl;
    Complex Al, Bl, Cl, Dl;
    Complex Av, Bv, Cv, Dv;
    Complex An, Bn, Cn, Dn;
```

```
546       Complex KKnose, KKlips, FC, FD;
547
548       if(icntlr <= 1){
549         newsec01(radin, nareav, gam, fimp, sfact);
550         newsec01(Vradin, Nsec, Vgam, Vfimp, Vsfact);
551       }
552       else{
553         newsec23(radin, radout, nareav, gam, fimp, sfact);
554         newsec23(Vradin, Vradout, Nsec, Vgam, Vfimp, Vsfact);
555       }
556       // ------------------------------------------------
557       RadiationZ radimpV = new RadiationZ(Varea1, freq, temper);
558       radimpV.calculate();
559       Vzl = Complex.valueOf(radimpV.getRadr(), radimpV.getRadi());
560       //radimp.setArea(area1);
561       RadiationZ radimp = new RadiationZ(area1, freq, temper);
562       radimp.calculate();
563       zl = Complex.valueOf(radimp.getRadr(), radimp.getRadi());
564
565       CalcF calfl = new CalcF(gam, fimp, 0, Nport-1, vlg);
566       calfl.calculate();
567       Al = calfl.getA();
568       Bl = calfl.getB();
569       Cl = calfl.getC();
570       Dl = calfl.getD();
571       CalcF calfv = new CalcF(gam, fimp, Nport, nareav-1, vlg);
572       calfv.calculate();
573       Av = calfv.getA();
574       Bv = calfv.getB();
575       Cv = calfv.getC();
576       Dv = calfv.getD();
577       CalcF calfn = new CalcF(Vgam, Vfimp, 0, Nsec-1, Nvlg);
578       calfn.calculate();
579       An = calfn.getA();
580       Bn = calfn.getB();
581       Cn = calfn.getC();
582       Dn = calfn.getD();
583
584
585       KKnose = ((Cn.multiply(Vzl)).add(Dn)).divide((An.multiply(Vzl)).add
              (Bn));
586       KKlips = ((Cl.multiply(zl)).add(Dl)).divide((Al.multiply(zl)).add(
              Bl));
587
588         /* ----------------------------------
589              H(w)=Hv(w)+sum(n=0,to nareav-1) Hn(w)
590              ---------------------------------- */
591
```

```
592       FC = (Al.multiply(Cv)).add(Dv.multiply((KKnose.multiply(Al)).add(Cl
              )));
593       FD = (Bl.multiply(Cv)).add(Dv.multiply((KKnose.multiply(Bl)).add(Dl
              )));
594       hvtra = Complex.valueOf(1.0).divide((FC.multiply(zl)).add(FD));
595
596       FC = (An.multiply(Cv)).add(Dv.multiply((KKlips.multiply(An)).add(Cn
              )));
597       FD = (Bn.multiply(Cv)).add(Dv.multiply((KKlips.multiply(Bn)).add(Dn
              )));
598       Nhvtra = Complex.valueOf(1.0).divide((FC.multiply(zl)).add(FD));
599     }
600
601     //private void Vocal(Complex hvtra) {
602     private void Vocal() {
603       double area1=radin[0]*radin[0]*PI;
604       Complex[] gam = new Complex[nareav];
605       Complex[] fimp = new Complex[nareav];
606       Complex Vzl, zl;
607       Complex Cv, Dv;
608
609       if(icntlr <=1)
610         newsec01(radin, nareav, gam, fimp, sfact);
611       else
612         newsec23(radin, radout, nareav, gam, fimp, sfact);
613
614       RadiationZ radimp = new RadiationZ(area1, freq, temper);
615       radimp.calculate();
616       zl = Complex.valueOf(radimp.getRadr(), radimp.getRadi());
617
618       CalcF calf = new CalcF(gam, fimp, 0, nareav-1, vlg);
619       calf.calculate();
620       Cv = calf.getC();
621       Dv = calf.getD();
622       /* ----------------------------------
623          H(w)=Hv(w)+sum(n=0,to nareav-1) Hn(w)
624          ---------------------------------- */
625       hvtra = Complex.valueOf(1.0).divide((Cv.multiply(zl)).add(Dv));
626     }
627
628
629     private void newsec01(double[] radin, int nsec, Complex[] gamma,
              Complex[] z, double[] sfact){
630       double Ai,Li,Gi,Ri,Ci,ome,ome2,S;
631       Complex Z2, gam;
632       double mu=1.86e-4;
633       double lambda=5.5e-5;
634       double cp=0.24;
```

```
635       double eta=1.4;
636     // const double c0=34359.0;
637     // const double rho=1.2046e-3;
638
639       double c0=3.3145e4+60.7*temper;
640       double rho=1.293e-3/(1.0+0.00367*temper);
641
642       ome=PI*2.0*freq;
643       ome2=ome*ome;
644
645       for(int i=0; i<nsec ;i++){
646         Ai = radin[i] * radin[i] * PI;
647         S = 2.0 * PI * radin[i] * sfact[i];
648
649         if(Ai < 1.0e-4){
650           System.out.println("Too small ! -> A[" + i + "]" + Ai);
651           gamma[i] = Complex.valueOf(1.0e-10);
652           z[i] = Complex.valueOf(1.0e10);
653         }
654         else{
655           switch (icntlr){
656           case 0:
657             z[i] = Complex.valueOf(rho*c0/Ai,0.0);
658             gamma[i] = Complex.valueOf(0.0,ome/c0);
659             break;
660
661           case 1:
662             Li = rho/Ai;
663             Ci = Ai/(rho*c0*c0);
664             Ri = (S/(Ai*Ai))*sqrt(ome*rho*mu/2.0);
665             Gi = (S*(eta-1.0)/(rho*c0*c0))*sqrt(lambda*ome/(2.0*cp*rho
                      ));
666             gam = Complex.valueOf(Ri,ome*Li).multiply(Complex.valueOf(Gi,
                      ome*Ci));
667             gamma[i] = gam.sqrt();
668             Z2 = Complex.valueOf(Ri,ome*Li).divide(Complex.valueOf(Gi,ome*
                      Ci));
669             z[i] = Z2.sqrt();
670             break;
671           default:
672             System.out.println("icntlr is not 0 nor 1...");
673             break;
674           } //<-switch
675         } //<-else
676       } //<-for
677     } // ----------------------- End of newsec01 -------
678
679   ///**********************************
```

```
680   // newsec23 calculates Zi, ri in the vocal tract
681   // from various parameters (R1, G, etc...)
682   // ******************************************/
683   //
684     private void newsec23(double[] radin, double[] radout, int nsec,
          Complex[] gamma, Complex[] z, double[] sfact){
685       double Ai,Li,Gi,Ri,Ci,ome,ome2,Lw,S,dit,xx,yy1,yy,bunbo,bunb2;
686       Complex Yw, Z2, gam;
687       double mu=1.86e-4;
688       double lambda=5.5e-5;
689       double cp=0.24;
690       double eta=1.4;
691
692       double c0 = 3.3145e4 + 60.7*temper;
693       double rho = 1.293e-3 / (1.0+0.00367*temper);
694
695       ome=PI*2.0*freq;
696       ome2=ome*ome;
697
698       for(int i=0; i<nsec ;i++){
699         dit = (radout[i]) - (radin[i]);
700         Ai = radin[i] * (radin[i]) * PI;
701         S = 2.0 * PI * (radin[i]) * (sfact[i]);
702         Lw = ZW.getLo();
703
704         if(Ai < 1.0e-4){
705           System.out.println("Too small ! -> A[" + i + "]" + Ai);
706           gamma[i] = Complex.valueOf(1.0e-10);
707           z[i] = Complex.valueOf(1.0e10);
708         }
709         else{
710           switch (icntlr){
711           case 2:
712             /******* Wall impedance *******/
713             if(dit==0.0)
714               System.out.println("Thickness =0 ! Something is wrong
                      ...");
715             bunbo = (ZW.getGw() * ZW.getGw()) + ome2 * ZW.getR1() * ZW.
                    getR1();
716             bunb2 = ZW.getR3() * ZW.getR3() + ome2 * Lw * Lw;
717             xx=((ZW.getGw() * ZW.getGw() * ZW.getR1() / bunbo) + ZW.
                    getR2()) / dit + ome2 * Lw * Lw * ZW.getR3() / bunb2;
718             yy1 = (-1.0) * ome * ZW.getGw() * ZW.getR1() * ZW.getR1();
719             yy = yy1 / bunbo / dit + ome * Lw * ZW.getR3() * ZW.getR3()
                    / bunb2;
720
721             Yw = Complex.valueOf(S,0.0).divide(Complex.valueOf(xx,yy));
722             Li= rho / Ai;
```

```
723             Ci = Ai / (rho * c0 * c0);
724             Ri=(S / (Ai * Ai)) * sqrt(ome * rho * mu / 2.0);
725             Gi=(S * (eta - 1.0)/(rho * c0 * c0)) * sqrt(lambda * ome /
                    (2.0 * cp * rho));
726             gam = Complex.valueOf(Ri, ome * Li).multiply(Complex.valueOf(
                    Gi, ome * Ci).add(Yw));
727             gamma[i] = gam.sqrt();
728             Z2 = Complex.valueOf(Ri, ome * Li).divide(Complex.valueOf(Gi,
                     ome * Ci).add(Yw));
729             z[i] = Z2.sqrt();
730             break;
731
732           case 3:
733             xx=ZW.getR1(); yy = ome * ZW.getLo() - ZW.getGw() / ome;
734             Yw=Complex.valueOf(S,0.0).divide(Complex.valueOf(xx, yy));
735             Li = rho / Ai;
736             Ci = Ai / (rho * c0 * c0);
737             Ri = (S / (Ai * Ai)) * sqrt(ome * rho * mu / 2.0);
738             Gi = (S * (eta - 1.0) / (rho * c0 * c0)) * sqrt(lambda * ome
                     / (2.0 * cp * rho));
739             gam = Complex.valueOf(Ri, ome * Li).multiply(Complex.valueOf(
                    Gi, ome * Ci).add(Yw));
740             gamma[i] = gam.sqrt();
741             Z2 = Complex.valueOf(Ri, ome * Li).divide(Complex.valueOf(Gi,
                     ome * Ci).add(Yw));
742             z[i] = Z2.sqrt();
743             break;
744
745           default:
746             System.out.println(" icntlr is not 2 nor 3 ...");
747             break;
748           } //<-switch
749         } //<-else
750       } //<-for
751     } // ----------------------- End of newsec23 -------
752
753     public static void main(String[] args){
754         int dataset = 10000;
755         for(int i=0; i<dataset; i++){
756             System.out.println(args);
757           File file = new File(args[0]);
758           if(!file.exists()) {
759             System.out.println("ファイルが存在しません。");
760             return;
761           }
762
763
764           try {
```

```
765             Peaks peaks = new Peaks(file, i);
766         if (args.length == 2)
767           if(args[1].equals("-h")) peaks.showHelp();
768         peaks.showStatus();
769         peaks.calculate();
770         peaks.saveFile();
771         } catch (IOException e) {
772           e.printStackTrace();
773         }
774       }
775   }
776 }
```

以下は声道伝達特性を求めるプログラムの補足プログラム（CalcF.java)

```
1
2  package jp.ac.hus.mhlab;
3
4  import java.io.*;
5  import static java.lang.Math.*;
6  import org.apache.commons.math3.transform.*;
7  import org.apache.commons.math3.complex.*;
8  import org.apache.commons.math3.exception.*;
9
10 public class CalcF {
11   private Complex A, B, C, D;
12   private Complex[] gamma, fimp;
13   private int nbgn, nend;
14   private double vlg;
15
16   public CalcF(Complex[] gamma, Complex[] fimp, int nbgn, int nend,
         double vlg) {
17     this.gamma = gamma;
18     this.fimp = fimp;
19     this.nbgn = nbgn;
20     this.nend = nend;
21     this.vlg = vlg;
22   }
23
24   public Complex getA(){
25     return A;
26   }
27   public Complex getB(){
28     return B;
29   }
30   public Complex getC(){
31     return C;
32   }
```

```
33      public Complex getD(){
34        return D;
35      }
36
37      public void calculate(){
38        Complex tanal,pk2;
39        Complex[] a = new Complex[nend+1];
40        Complex[] b = new Complex[nend+1];
41        Complex[] c = new Complex[nend+1];
42        Complex[] d = new Complex[nend+1];
43        Complex[] fa = new Complex[nend+1];
44        Complex[] fb = new Complex[nend+1];
45        Complex[] fc = new Complex[nend+1];
46        Complex[] fd = new Complex[nend+1];
47        double ei,er;
48        int ise,iid;
49
50        for(int i=nbgn; i<=nend; i++){
51          er = (gamma[i].multiply(vlg)).getReal();
52          ei = (gamma[i].multiply(vlg)).getImaginary();
53
54          tanal = Complex.valueOf(tanh(er)*cos(ei),sin(ei)).divide(Complex.
                valueOf(cos(ei),tanh(er)*sin(ei)));
55          //tanal=(std::complex<double>(tanh(er)*cos(ei),sin(ei))/std::
                complex<double>(cos(ei),tanh(er)*sin(ei)));
56          pk2=Complex.valueOf(1.0).divide(tanal.sqrt1z());
57          //pk2=1.0/sqrt(1.0-tanal*tanal);
58
59          a[i]=pk2;
60          b[i]=pk2.multiply(fimp[i].multiply(tanal));
61          c[i]=pk2.multiply(tanal.divide(fimp[i]));
62          d[i]=pk2; //Initialize...
63          //a[i]=pk2 ; b[i]=pk2*fimp[i]*tanal;
64          //c[i]=pk2*tanal/fimp[i] ; d[i]=pk2; //Initialize...
65
66      }
67
68        /*****-------------------------
69              Calculate Whole F_Matrix
70         -------------------------********/
71
72        fa[nbgn]=a[nbgn];
73        fb[nbgn]=b[nbgn];
74        fc[nbgn]=c[nbgn];
75        fd[nbgn]=d[nbgn];
76
77        for(ise=nbgn, iid = nbgn+1 ; iid<= nend ; ise++,iid++){
78          fa[iid]=(a[iid].multiply(fa[ise])).add(b[iid].multiply(fc[ise]));
```

```
79          fb[iid]=(a[iid].multiply(fb[ise])).add(b[iid].multiply(fd[ise]));
80          fc[iid]=(c[iid].multiply(fa[ise])).add(d[iid].multiply(fc[ise]));
81          fd[iid]=(c[iid].multiply(fb[ise])).add(d[iid].multiply(fd[ise]));
82         }
83
84        A = fa[nend];
85        B = fb[nend];
86        C = fc[nend];
87        D = fd[nend];
88      }
89    }
```

以下は声道伝達特性を求めるプログラムの補足プログラム（RadiationZ.java)

```
1
2   package jp.ac.hus.mhlab;
3   import static java.lang.Math.*;
4
5   public class RadiationZ {
6     private static final double SMALL = -1.0e36;
7     private double area, freq, temper, radr, radi;
8
9     public RadiationZ(double area, double freq, double temper){
10      this.area = area;
11      this.freq = freq;
12      this.temper = temper;
13      this.radr = 0.0;
14      this.radi = 0.0;
15    }
16
17    public double getRadr(){
18      return radr;
19    }
20
21    public double getRadi(){
22      return radi;
23    }
24
25    public void setArea(double area){
26      this.area = area;
27    }
28
29    public void calculate(){
30      double[] p = new double[10000];
31      double x, conv, d1, d2, df, d=0.0;
32      double xmyu, xreal, ximag, p1=0.0, p2=0.0, order=0.0, pp;
33      int m, norder;
34      final double C0 = 33145.0 + 60.7*temper; // cm
```

```
35      final double RHO = 1.293e-3 /(1.0 + 0.00367*temper); // cm
36      final int NPX = 41;
37
38      double rhead=8.0; // cm
39      x=sqrt(rhead*rhead-area/PI)/rhead; /* x: cos[theta0] */
40      conv = 0.5/(1.0-x); /*const=1/{4*sin(theta0/2)*sin(theta0/2)}*/
41
42      /* Generate p[ ] */
43      p[1] =1.0 ; p[2] =1.0 ; p[3] =x;
44      for(int i=3 ; i <= NPX; i++){
45        d1 = x*p[i] ;
46        d2 = d1-p[i-1];
47        df = (double)i - 1.0;
48        p[i+1] = d1+d2 - d2/df ;
49      }
50
51      xmyu=2.0*PI*freq*rhead/C0;
52
53      /* myu = 2*pai*a/lamda
54                 = 2*pai*a*freq./c
55                 a: = rhead
56      */
57      for(m=0, xreal=0.0, ximag=0.0; m<=26 ; m++ ) {
58        if(m == 0) {
59          p1= - snbes(1,xmyu);
60          p2= sjbes(1,xmyu);
61        }
62        else {
63          p1=(double)m*snbes(m-1,xmyu)-((double)m + 1.0)*snbes(m+1,xmyu
              );
64          p2=((double)m +1.0)*sjbes(m+1,xmyu)-(double)m*sjbes(m-1,xmyu);
65        }
66        if(abs(p1)>1.0e16 || abs(p2)>1.0e16) break;
67        norder=(int)(log10(max(abs(p1),abs(p2))));
68        order=pow(10.0,(-norder));
69        p1=p1*order; p2=p2*order;
70
71        pp=p1*p1+p2*p2;
72        d=(p[m+1]-p[m+3])*(p[m+1]-p[m+3]);
73
74        /* P[m-1] --> p(m+1)
75           P[m+1] --> p(m+3) */
76
77        xreal =xreal+(2.0*(double)m +1.0)*d/pp*order*order;
78        ximag =ximag+d*(sjbes(m,xmyu)*p2-snbes(m,xmyu)*p1)/pp*order;
79      }
80
81      //System.out.printf("d p1 p2= %f, %f, %f %f %n", d,p1,p2,order) ;
```

```
82
83        radr =conv*xreal/(xmyu*xmyu);
84        radi =conv*ximag;
85        radr *= (conv*RHO*C0/(PI*rhead*rhead)); /* Radiation impedance */
86        radi *= (conv*RHO*C0/(PI*rhead*rhead)); /* Radiation impedance */
87      }
88      /* =-=-=-=-=-=- End of radimp -=-=-=-=-=-=-=-=-=-=-=- */
89
90      private double sjbes(int n, double x) {
91        double t1=0.0,t2,t3=0.0,dx,y,w,z,sj=0.0,dy,d55,xdum;
92        int lme=0,i,nm,k;
93
94            xdum=x - 3.0e4;
95            if( ((n-30000) > 0) ||(xdum >= 0.0)||(x < 0.0) || (n < 0)){
96              System.out.println("Bad Arg.1!");
97              return 0.0;
98            }
99
100           d55=1.0e-36;
101           dx=x;
102           xdum=x-7.0e-4;
103
104           if(xdum < 0.0){
105             w=1.0;
106             if(n == 0) return(w);
107             if(dx <= 1.0e-37) return(0.0);
108             dy =1.0e-37/dx;
109             t1=3.0 ; t2 =1.0;
110
111             for(i=1; i<=n ; i++) {
112               t3=t2*dx/t1;
113               t1=t1+2.0;
114               t2=t3;
115               if( abs(t3) <= t1*dy ) return(0.0);
116             }
117             return(t3);
118           }
119
120           if( (x-0.2) < 0.0 ) { y = dx*dx;
121             w = 1.0-y*(1.0-0.05*y)/6.0;}
122           else w = sin(dx)/dx;
123           if( n == 0) return(w);
124
125           if( (x-100.0) >= 0.0) lme=(int)(0.5*x+5.0);
126           else{ if((x-10.0) >= 0.0) lme=(int)(0.1*lme+10.0);
127             else{ if( (x-1.0) >= 0.0) lme=(int)(0.02*x+18.0);
128               else lme=5; }}
129
```

```
130           nm=max(n,(int)x)+ lme;
131           z =1.0/dx;
132           t3=0.0;
133           t2=1.0e-36;
134
135           for(i=1 ; i<=nm ; i++){
136             k=nm-i;
137             t1=(2.0*(double)k+3.0)*z*t2-t3;
138             if( n-k == 0) sj=t1;
139
140
141             if((abs(t1)-1.0e36) >= 0.0) {
142               t1=t1*d55;
143               t2=t2*d55;
144               if((n-k) >= 0) sj=sj*d55;}
145
146             t3=t2; t2=t1;
147           }/* End of for*/
148
149           if(abs(w) <= 0.01){
150             w=w/dx-cos(dx)/dx;
151             return (sj/t3*w);
152           }
153           return (w/t1*sj);
154      }
155     /* ===== End of sjbes ====================
156        ### snbes ###
157     */
158
159      private double snbes(int n,double x) {
160        double dx,z,qn0,qn1,qn2=0.0,w,fdum;
161        int ndum,i,m;
162
163        ndum=n-30000;
164        if((ndum >= 0)||(x <= 0.0)||(n < 0)){
165          System.out.println("Bad Arg.2!");
166          return(0.0);
167        }
168
169        dx=x;
170        z=1.0/dx;
171
172            if( (x-7.0e-4) <= 0.0) {
173              if(n == 0) return (-z);
174              m=(int)(30.0/log10(z));
175          if((n-m+1) > 0) return(SMALL);
176
177              for(i=1,qn0=z,qn1=1.0; i<=n; i++){
```

```
178               qn2=qn0*qn1*z;
179               qn1+=2.0;
180               qn0=qn2;
181             }
182             return (-qn2);
183           }
184
185       qn0= -z*cos(dx);
186       if(n == 0) return(qn0);
187       qn1=z*(qn0-sin(dx));
188
189       ndum=n-1;
190       if( ndum < 0) return (qn0);
191       if( ndum == 0) return(qn1);
192
193       for(i=2; i <=n ; i++){
194         fdum=abs(qn1)-1.0e36;
195         if(fdum >= 0.0) {
196           w=qn1*1.0e-10;
197           qn2=(2.0*(double)i-1.0)*z*w-qn0*1.0e-10;
198           if(abs(qn2) >= 1.0e36){
199             System.out.println("Bad ..3");
200             return(SMALL);
201           }
202           qn2*=1.0e10;
203         }
204         else qn2=(2.0*(double)i-1.0)*z*qn1-qn0;
205
206         qn0=qn1; qn1=qn2;
207       }
208       return(qn2);
209     }
210   }
```

以下は声道伝達特性を求めるプログラムの補足プログラム（WallImpedance.java)

```
1
2   package jp.ac.hus.mhlab;
3
4   public class WallImpedance {
5     private double R1, R2, R3, Lo, Gw;
6
7     public WallImpedance(double R1, double R2, double R3, double Lo,
          double Gw) {
8       this.R1 = R1;
9       this.R2 = R2;
10      this.R3 = R3;
11      this.Lo = Lo;
```

```
12        this.Gw = Gw;
13      }
14
15      public double getR1(){
16        return R1;
17      }
18      public double getR2(){
19        return R2;
20      }
21      public double getR3(){
22        return R3;
23      }
24      public double getLo(){
25        return Lo;
26      }
27      public double getGw(){
28        return Gw;
29      }
30      public void setR1(double R1){
31        this.R1 = R1;
32      }
33      public void setLo(double Lo){
34        this.Lo = Lo;
35      }
36      public void setGw(double Gw){
37        this.Gw = Gw;
38      }
39    }
```

付 録C　ニューラルネットワークのプログラム

ニューラルネットワークのプログラムを以下に示す。

```
# coding: utf-8
import sys, os
sys.path.append(os.pardir)
import matplotlib.pyplot as plt
import numpy as np
import pickle
from dataset import load_data #要変更　データ取得
from two_layer_net import TwoLayerNet
from common.optimizer import Adam

# データの読み込み
(x_train, t_train), (x_test, t_test) = load_data()

network = TwoLayerNet(input_size=1024, hidden_size=50, output_size=34)
optimizer = Adam(lr=0.001)

iters_num = 10000
train_size = x_train.shape[0]
test_size = x_test.shape[0]
batch_size = 100

train_loss_list = []
test_loss_list = []
train_acc_list = []
test_acc_list = []

iter_per_epoch = max(train_size / batch_size, 1)

for i in range(iters_num):
    batch_mask = np.random.choice(train_size, batch_size)
    x_batch = x_train[batch_mask]
    t_batch = t_train[batch_mask]

    batch_mask_for_test = np.random.choice(test_size, batch_size)
    x_t_batch = x_test[batch_mask_for_test]
    t_t_batch = t_test[batch_mask_for_test]
```

```
37
38      # 勾配
39      grad = network.gradient(x_batch, t_batch)
40      optimizer.update(network.params, grad)
41
42      loss = network.loss(x_batch, t_batch)
43      loss_test = network.loss(x_t_batch, t_t_batch)
44
45      train_loss_list.append(loss)
46      test_loss_list.append(loss_test)
47
48      if i % iter_per_epoch == 0:
49          train_acc = network.accuracy(x_train, t_train)
50          test_acc = network.accuracy(x_test, t_test)
51          train_acc_list.append(train_acc)
52          test_acc_list.append(test_acc)
53
54  with open('C:￥￥Users￥￥Masataka￥￥Desktop￥￥akemi￥￥grad_sample.pkl
        ',mode='wb') as f:
55      pickle.dump(grad, f)
56
57  plt.figure()
58  plt.xlabel('iterations')
59  plt.ylabel('loss')
60  plt.plot(train_loss_list, label = 'train', linestyle = 'dashed')
61  plt.plot(test_loss_list, label = 'test', linestyle = 'solid')
62  plt.legend()
63
64  plt.figure()
65  plt.xlabel('epochs', fontsize = 18)
66  plt.ylabel('accuracy', fontsize = 18)
67  plt.plot(train_acc_list, label = 'train', linestyle = 'dashed')
68  plt.plot(test_acc_list, label = 'test', linestyle = 'solid')
69  plt.legend()
70  plt.show()
```

以下はニューラルネットワークの補足プログラム（dataset.py）

```
1   import glob
2   import os
3   import numpy as np
4   import random
5
6   def load_data():
7       train_peaks = []
8       for filepath in glob.glob('C:￥￥
            Users￥￥Masataka￥￥Documents￥￥NetBeansProjects￥￥Peaks￥￥peaks￥￥
            *.v'):
```

```
9           with open(filepath, 'r') as f:
10              l = [s.strip() for s in f.readlines()]
11              l_f = [float(s) for s in l]
12              train_peaks.append(l_f)
13
14      random.seed(10)
15      t_p = np.array(random.sample(train_peaks, 7000))
16
17      train_area = []
18      for filepath in glob.glob('C:￥￥
            Users￥￥Masataka￥￥Documents￥￥NetBeansProjects￥￥Peaks￥￥area￥￥
            *'):
19          with open(filepath, 'r') as f:
20              l = [s.strip() for s in f.readlines()]
21              l_f = [float(s) for s in l]
22              del l_f[0:2]
23              train_area.append(l_f)
24
25      random.seed(10)
26      t_a = np.array(random.sample(train_area, 7000))
27
28      test_peaks = []
29      for filepath in glob.glob('C:￥￥
            Users￥￥Masataka￥￥Documents￥￥NetBeansProjects￥￥Peaks￥￥peaks￥￥
            *.v'):
30          with open(filepath, 'r') as f:
31              l = [s.strip() for s in f.readlines()]
32              l_f = [float(s) for s in l]
33              test_peaks.append(l_f)
34
35      random.seed(1)
36      ts_p = np.array(random.sample(test_peaks, 3000))
37
38      test_area = []
39      for filepath in glob.glob('C:￥￥
            Users￥￥Masataka￥￥Documents￥￥NetBeansProjects￥￥Peaks￥￥area￥￥
            *'):
40          with open(filepath, 'r') as f:
41              l = [s.strip() for s in f.readlines()]
42              l_f = [float(s) for s in l]
43              del l_f[0:2]
44              test_area.append(l_f)
45
46      random.seed(1)
47      ts_a = np.array(random.sample(test_area, 3000))
48
49      return (t_p, t_a),(ts_p, ts_a)
```

以下はニューラルネットワークの補足プログラム（two layer net.py）

```
# coding: utf-8
import sys, os
sys.path.append(os.pardir) # 親ディレクトリのファイルをインポートするための設定
import numpy as np
from common.layers import *
from common.gradient import numerical_gradient
from collections import OrderedDict

class TwoLayerNet:

    def __init__(self, input_size, hidden_size, output_size, weight_init_std = 0.01):
        # 重みの初期化
        self.params = {}
        self.params['W1'] = np.random.randn(input_size, hidden_size) / np.sqrt(input_size)
        self.params['b1'] = np.zeros(hidden_size)
        self.params['W2'] = np.random.randn(hidden_size, output_size) / np.sqrt(hidden_size)
        self.params['b2'] = np.zeros(output_size)

        #レイヤの生成
        self.layers = OrderedDict()
        self.layers['Affine1'] = Affine(self.params['W1'], self.params['b1'])
        self.layers['BatchNormalization'] = BatchNormalization(1, 0)
        self.layers['Sigmoid1'] = Sigmoid()
        self.layers['Affine2'] = Affine(self.params['W2'], self.params['b2'])

        self.lastLayer = IdentityWithLoss()

    def predict(self, x):
        for layer in self.layers.values():
            x = layer.forward(x)

        return x

    # x:入力データ, t:教師データ
    def loss(self, x, t):
        y = self.predict(x)
        return self.lastLayer.forward(y, t)

    def accuracy(self, x, t):
        y = self.predict(x)
        y = np.argmax(y, axis=1)
```

```
42          if t.ndim != 1 : t = np.argmax(t, axis=1)
43
44          accuracy = np.sum(y == t) / float(x.shape[0])
45          return accuracy
46
47      # x:入力データ, t:教師データ
48      def numerical_gradient(self, x, t):
49          loss_W = lambda W: self.loss(x, t)
50
51          grads = {}
52          grads['W1'] = numerical_gradient(loss_W, self.params['W1'])
53          grads['b1'] = numerical_gradient(loss_W, self.params['b1'])
54          grads['W2'] = numerical_gradient(loss_W, self.params['W2'])
55          grads['b2'] = numerical_gradient(loss_W, self.params['b2'])
56
57          return grads
58
59      def gradient(self, x, t):
60          # forward
61          self.loss(x, t)
62
63          # backward
64          dout = 1
65          dout = self.lastLayer.backward(dout)
66
67          layers = list(self.layers.values())
68          layers.reverse()
69          for layer in layers:
70              dout = layer.backward(dout)
71
72          # 設定
73          grads = {}
74          grads['W1'], grads['b1'] = self.layers['Affine1'].dW, self.
                layers['Affine1'].db
75          grads['W2'], grads['b2'] = self.layers['Affine2'].dW, self.
                layers['Affine2'].db
76
77          return grads
```

以下はニューラルネットワークの補足プログラム（optimizer.py）

```
1   # coding: utf-8
2   import numpy as np
3
4   class SGD:
5
6       """確率的勾配降下法(Stochastic Gradient Descent)"""
7
```

```
8       def __init__(self, lr=0.01):
9           self.lr = lr
10
11      def update(self, params, grads):
12          for key in params.keys():
13              params[key] -= self.lr * grads[key]
14
15
16  class Momentum:
17
18      """Momentum SGD"""
19
20      def __init__(self, lr=0.01, momentum=0.9):
21          self.lr = lr
22          self.momentum = momentum
23          self.v = None
24
25      def update(self, params, grads):
26          if self.v is None:
27              self.v = {}
28              for key, val in params.items():
29                  self.v[key] = np.zeros_like(val)
30
31          for key in params.keys():
32              self.v[key] = self.momentum*self.v[key] - self.lr*grads[key
                    ]
33              params[key] += self.v[key]
34
35
36  class Nesterov:
37
38      """Nesterov's Accelerated Gradient (http://arxiv.org/abs
            /1212.0901)"""
39
40      def __init__(self, lr=0.01, momentum=0.9):
41          self.lr = lr
42          self.momentum = momentum
43          self.v = None
44
45      def update(self, params, grads):
46          if self.v is None:
47              self.v = {}
48              for key, val in params.items():
49                  self.v[key] = np.zeros_like(val)
50
51          for key in params.keys():
52              self.v[key] *= self.momentum
53              self.v[key] -= self.lr * grads[key]
```

```
54              params[key] += self.momentum * self.momentum * self.v[key]
55              params[key] -= (1 + self.momentum) * self.lr * grads[key]
56
57
58  class AdaGrad:
59
60      """AdaGrad"""
61
62      def __init__(self, lr=0.01):
63          self.lr = lr
64          self.h = None
65
66      def update(self, params, grads):
67          if self.h is None:
68              self.h = {}
69              for key, val in params.items():
70                  self.h[key] = np.zeros_like(val)
71
72          for key in params.keys():
73              self.h[key] += grads[key] * grads[key]
74              params[key] -= self.lr * grads[key] / (np.sqrt(self.h[key])
                    + 1e-7)
75
76
77  class RMSprop:
78
79      """RMSprop"""
80
81      def __init__(self, lr=0.01, decay_rate = 0.99):
82          self.lr = lr
83          self.decay_rate = decay_rate
84          self.h = None
85
86      def update(self, params, grads):
87          if self.h is None:
88              self.h = {}
89              for key, val in params.items():
90                  self.h[key] = np.zeros_like(val)
91
92          for key in params.keys():
93              self.h[key] *= self.decay_rate
94              self.h[key] += (1 - self.decay_rate) * grads[key] * grads[
                    key]
95              params[key] -= self.lr * grads[key] / (np.sqrt(self.h[key])
                    + 1e-7)
96
97
98  class Adam:
```

```
99
100         """Adam (http://arxiv.org/abs/1412.6980v8)"""
101
102         def __init__(self, lr=0.001, beta1=0.9, beta2=0.999):
103             self.lr = lr
104             self.beta1 = beta1
105             self.beta2 = beta2
106             self.iter = 0
107             self.m = None
108             self.v = None
109
110         def update(self, params, grads):
111             if self.m is None:
112                 self.m, self.v = {}, {}
113                 for key, val in params.items():
114                     self.m[key] = np.zeros_like(val)
115                     self.v[key] = np.zeros_like(val)
116
117             self.iter += 1
118             lr_t = self.lr * np.sqrt(1.0 - self.beta2**self.iter) / (1.0 -
                     self.beta1**self.iter)
119
120             for key in params.keys():
121                 #self.m[key] = self.beta1*self.m[key] + (1-self.beta1)*
                        grads[key]
122                 #self.v[key] = self.beta2*self.v[key] + (1-self.beta2)*(
                        grads[key]**2)
123                 self.m[key] += (1 - self.beta1) * (grads[key] - self.m[key
                        ])
124                 self.v[key] += (1 - self.beta2) * (grads[key]**2 - self.v[
                        key])
125
126                 params[key] -= lr_t * self.m[key] / (np.sqrt(self.v[key]) +
                         1e-7)
127
128                 #unbias_m += (1 - self.beta1) * (grads[key] - self.m[key])
                        # correct bias
129                 #unbisa_b += (1 - self.beta2) * (grads[key]*grads[key] -
                        self.v[key]) # correct bias
130                 #params[key] += self.lr * unbias_m / (np.sqrt(unbisa_b) + 1
                        e-7)
```

謝 辞

本研究を進めるにあたり，終始，御指導，御鞭撻をいただいた北海道科学大学工学部 情報工学科　音声処理分野, 松﨑教授に心より感謝致します.

本研究の遂行にあたり様々な御助言，御協力をいただいた北海道科学大学工学部 情報工学科　松﨑ゼミの皆さんに深く感謝致します.

最後に本研究の遂行にあたり著者を常に支援し応援してくれた両親，並びに親族のみなさまに心より感謝申しあげます.