令和 元年度 卒業論文

OpenModelica のシミュレーションを用いたモータ特性表自動生成ツールの試作

指導教員　片山 徹郎 教授

宮崎大学 工学部 情報システム工学科

原田　海人

2020 年 1 月

目次

第 1 章

はじめに

近年、モータは、エアコン・洗濯機・掃除機などの家電製品をはじめ、自動車関係、医療関係など様々な分野に用いられており [1]、社会に必要不可欠な存在となっている。

～はじめに　流れ　案 1～

シミュレーションを行った場合、期待通りか確認する。

確認する際は、シミュレーション結果から目的のグラフや値を計算等して作成しなければならない。

今回試作したツールで、グラフや値を作成する手間を省くことで、モータ開発の効率化を図る。

本論文の構成は、以下の通りである。

第 2 章では、モータ特性表自動生成ツールを試作するために必要となる前提知識について説明する。

第 3 章では、試作したモータ特性表自動生成ツールの機能について説明する。

第 4 章では、モータ特性表自動生成ツールの実装について説明する。

第 5 章では、試作したモータ特性表自動生成ツールが正しく動作することを検証する。

第 6 章では、試作したモータ特性表自動生成ツールについて考察する。

第 7 章では、本論文のまとめと今度の課題を述べる。

第 2 章

研究の準備

本章では、本研究で必要となる前提知識を説明する。

2.1 Modelica 言語

Modelica 言語とは、微分代数方程式を用いた、複合領域のマルチドメインモデリングのために開発されたオブジェクト指向言語である [2]。その言語仕様は、非営利団体の Modelica Association が策定している。Modelica Association では、Modelica 言語による様々な物理領域のモデルライブラリを開発しており、数学、機械、電気、熱、流体、制御系、状態遷移機械などを含んだフリーの Modelica 標準ライブラリ (Modelica Standard Library : MSL) をリリースしている [2]。

2.2 OpenModelica

OpenModelica とは、Open Source Modelica Consortium (OSMC) が開発している Modelica コードのモデリング、シミュレーション、デバッグのための機能などを持つオープンソースプラットフォームである [3]。OpenModelica では、シミュレーション結果をグラフとして画面上に描画できる。また、シミュレーション結果は、以下の 3 つのファイル形式から保存することができる。

- mat ファイル

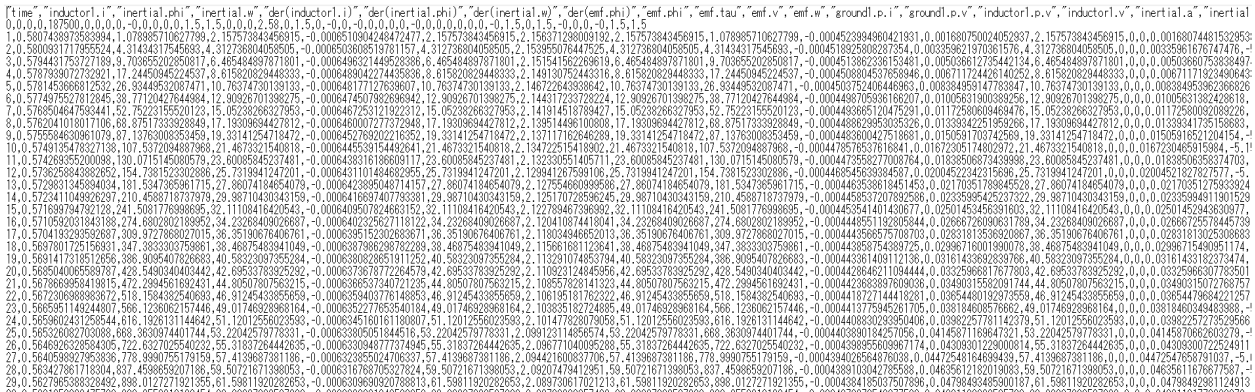

図 2.1: シミュレーション結果の csv ファイルの一部

- plt ファイル

- csv ファイル

今回試作するモータ特性表自動生成ツールでは、csv ファイルにのみ対応する。

OpenModelica から出力される csv ファイルの一部を、図 2.1 に示す。

図 2.1 に示す csv ファイルの 1 列目には、時間を表すデータが必ず入る。そして 1 列目の「time」を除いて 1 行目には、オブジェクト名を含んだ変数名が必ず入る。

OpenModelica から出力される csv ファイルのファイル名は、「(シミュレーションしたモデルの名前)_res.csv」で生成される。もし、シミュレーションしたモデルの名前が、「hoge」だった場合、csv ファイルのファイル名は「hoge_res.csv」となる。

2.3 ブラシ付き DC モータ

ブラシ付き DC モータとは、磁場の中にあるコイルに電流を流す事で発生するローレンツ力を回転方向に利用することで回すモータである [4]。シンプルな構造で、制御しやすく、汎用性が高く、模型用モータや自動車補機用モータなど世界で一番多く使われているモータである [5]。

2.4 対応するモデル

試作するモータ特性表自動生成ツールでは、以下 2 つの Modelica モデルのシミュレーション結果に対応する。

2.4.1 ブラシ付き DC モータの Modelica モデル

ブラシ付き DC モータの Modelica モデルとは、ブラシ付き DC モータの等価回路 [6] を Modelica 言語で表したモデルのことである。

ブラシ付き DC モータの等価回路を Modelica 言語で表すためには、電源部品、抵抗部品、インダクタ部品、起電力部品、慣性部品、接地部品が必要である。

また、電源部品、抵抗部品、インダクタ部品、起電力部品、慣性部品には、それぞれ以下のパラメータを設定しなければならない。

- 電源部品 ・・・ 電圧値 V
- 抵抗部品 ・・・ 抵抗値 Ω
- インダクタ部品 ・・・ インダクタンス値 H
- 起電力部品 ・・・ トルク定数 $\mathrm{N \cdot m/A}$
- 慣性部品 ・・・ 慣性モーメント $\mathrm{kg \cdot m^2}$

各部品で使用する MSL を表 2.1 に、ブラシ付き DC モータの等価回路を図 2.2 に、ブラシ付き DC モータの Modelica モデルの例を図 2.3 に、図 2.3 の Modelica コードを図 2.4 に、それぞれ示す。

表 2.1: MSL 対応表

部品名	使用する MSL
電源部品	Modelica.Electrical.Analog.Sources
抵抗部品	Modelica.Electrical.Analog.Basic
インダクタ部品	Modelica.Electrical.Analog.Basic
起電力部品	Modelica.Electrical.Analog.Basic
慣性部品	Modelica.Mechanics.Rotational.Components
接地部品	Modelica.Electrical.Analog.Basic

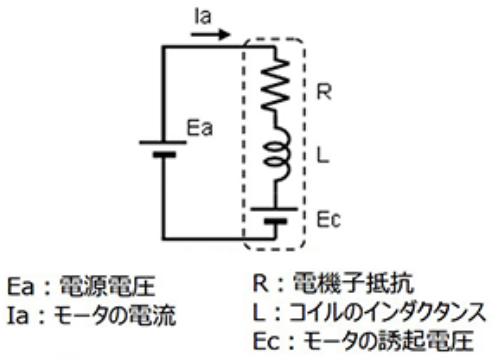

図 2.2: ブラシ付き DC モータの等価回路

2.4.2 ブラシ付き DC モータの Modelica モデルをサブシステムとするモデル

ブラシ付き DC モータの Modelica モデルをサブシステムとするモデルとは、2.4.1 節で説明したブラシ付き DC モータの Modelica モデルを 1 つのサブシステムとして扱い、他の部品と合わせたモデルのことである。

例として、ブラシ付き DC モータのサブシステムを用いた DC モータサーボのモデルを図 2.5 に、図 2.5 の Modelica コードを図 2.6 に、それぞれ示す。

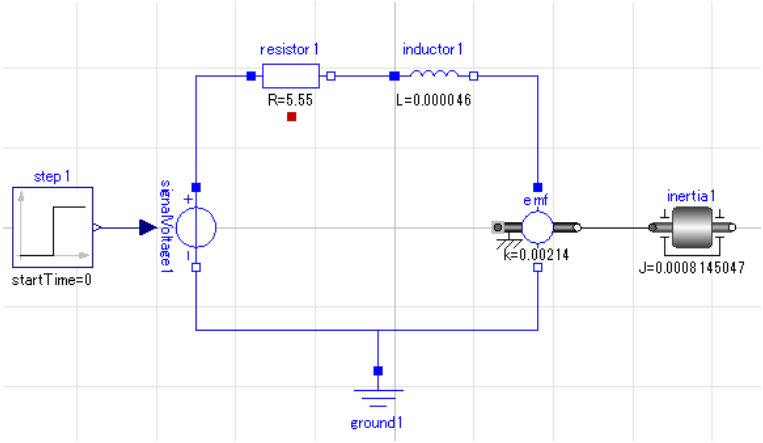

図 2.3: ブラシ付き DC モータの Modelica モデルの例

2.5 モータ特性表

モータ特性表とは、モータを選定する際に、参考にする資料である [7]。一般的に決まった形式はなく、企業によって書いている要素は異なるため、10 社のモータ特性表 [8, 9, 10, 11, 12, 13, 14, 15, 16, 17] に書かれている要素を集計した。その中でも出現回数が多く、ブラシ付き DC モータのシミュレーション結果から作成できる要素を、今回自動生成するモータ特性表の要素とした。

以下にモータ特性表の構成と要素を示す。

2.5.1 特性表

特性表を構成する 9 個の要素が表す内容について述べる。

電圧

電圧とは、シミュレーション時に、回路に印加された電圧値を表す。

単位は、V(ボルト) である。

始動電流

始動電流とは、モータの起動時に流れる電流値を表す。

```
model DCmotor
Modelica.Electrical.Analog.Basic.Resistor resistor1(R = 5.55, T = 283.15)  annotation(
    Placement(visible = true, transformation(origin = {-26, 38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Inductor inductor1(L = 0.000046)  annotation(
    Placement(visible = true, transformation(origin = {10, 38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Ground ground1 annotation(
    Placement(visible = true, transformation(origin = {-4, -46}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Components.Inertia inertia1(J = 0.0008145047, a(start = 0), phi(start = 0), w(start = 0))  annotation(
    Placement(visible = true, transformation(origin = {75, 0}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.EMF emf(k = 0.00214, useSupport = false)  annotation(
    Placement(visible = true, transformation(origin = {36, 0}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Sources.SignalVoltage signalVoltage1 annotation(
    Placement(visible = true, transformation(origin = {-50, 0}, extent = {{-10, 10}, {10, -10}}, rotation = -90)));
  Modelica.Blocks.Sources.Step step1(height = 2.4, startTime = 0)  annotation(
    Placement(visible = true, transformation(origin = {-86, 0}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(step1.y, signalVoltage1.v) annotation(
    Line(points = {{-75, 0}, {-62, 0}}, color = {0, 0, 127}));
  connect(emf.flange, inertia1.flange_a) annotation(
    Line(points = {{46, 0}, {65, 0}}));
  connect(signalVoltage1.p, resistor1.p) annotation(
    Line(points = {{-50, 10}, {-50, 38}, {-36, 38}}, color = {0, 0, 255}));
  connect(resistor1.n, inductor1.p) annotation(
    Line(points = {{-16, 38}, {0, 38}}, color = {0, 0, 255}));
  connect(inductor1.n, emf.p) annotation(
    Line(points = {{20, 38}, {36, 38}, {36, 10}}, color = {0, 0, 255}));
  connect(ground1.p, signalVoltage1.n) annotation(
    Line(points = {{-4, -36}, {-4, -26}, {-50, -26}, {-50, -10}}, color = {0, 0, 255}));
  connect(emf.n, signalVoltage1.n) annotation(
    Line(points = {{36, -10}, {36, -26}, {-50, -26}, {-50, -10}}, color = {0, 0, 255}));
  annotation(
    uses(Modelica(version = "3.2.3")));
end DCmotor;
```

図 2.4: 図 2.3 の Modelica コード

単位は、mA(ミリアンペア) である。

停動トルク

停動トルクとは、モータが出しうる最大トルクで、このトルク以上の負荷がかかれば、モータが停止する値を表す。

単位は、mNm(ミリニュートンメートル) である。

最大効率

効率とは、入力電力に対する機械出力の比を百分率［%］で表したものであり、最大効率は、その中で最大値を表す。

単位は、%(パーセント) である。

定格トルク

定格トルクとは、最大効率時のトルク値を表す。

単位は、mNM(ミリニュートンメートル) である。

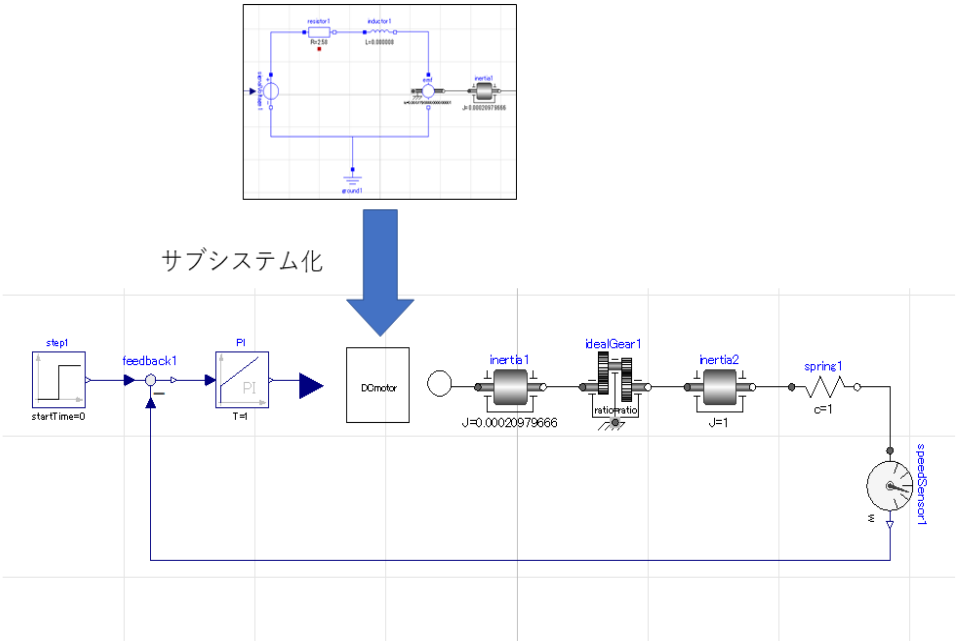

図 2.5: DC モータサーボのモデル

定格回転数

定格回転数とは、最大効率時の回転数値を表す。

単位は、rpm(アールピーエム) である。

定格電流

定格電流とは、最大効率時の電流値を表す。

単位は、mA(ミリアンペア) である。

定格出力

定格出力とは、最大効率時の出力値を表す。

単位は、W(ワット) である。

```
model submodel
  Modelica.Blocks.Sources.Step step1(height = 1.5)  annotation(
    Placement(visible = true, transformation(origin = {-70, 48}, extent = {{-4, -4}, {4, 4}}, rotation = 0)));
  Modelica.Blocks.Math.Feedback feedback1 annotation(
    Placement(visible = true, transformation(origin = {-56, 48}, extent = {{-4, -4}, {4, 4}}, rotation = 0)));
  Modelica.Blocks.Continuous.PI PI(T = 1)  annotation(
    Placement(visible = true, transformation(origin = {-42, 48}, extent = {{-4, -4}, {4, 4}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Components.IdealGear idealGear1 annotation(
    Placement(visible = true, transformation(origin = {13, 49}, extent = {{-5, -5}, {5, 5}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Components.Inertia inertia2(J = 1) annotation(
    Placement(visible = true, transformation(origin = {31, 49}, extent = {{-5, -5}, {5, 5}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Components.Spring spring1(c = 1)  annotation(
    Placement(visible = true, transformation(origin = {47, 49}, extent = {{-5, -5}, {5, 5}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Components.Inertia inertia3(J = 0.00020979666) annotation(
    Placement(visible = true, transformation(origin = {194, 14}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Mechanics.Rotational.Sensors.SpeedSensor speedSensor1 annotation(
    Placement(visible = true, transformation(origin = {57, 33}, extent = {{-5, -5}, {5, 5}}, rotation = -90)));
  pack_iner pack_iner1 annotation(
    Placement(visible = true, transformation(origin = {-16, 48}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(idealGear1.flange_a, pack_iner1.flange_b) annotation(
    Line(points = {{8, 49}, {8, 49.5}, {-6, 49.5}, {-6, 48}}));
  connect(idealGear1.flange_b, inertia2.flange_a) annotation(
    Line(points = {{18, 49}, {26, 49}}));
  connect(spring1.flange_b, speedSensor1.flange) annotation(
    Line(points = {{52, 49}, {57, 49}, {57, 38}}));
  connect(inertia2.flange_b, spring1.flange_a) annotation(
    Line(points = {{36, 49}, {42, 49}}));
  connect(PI.y, pack_iner1.u) annotation(
    Line(points = {{-38, 48}, {-28, 48}}, color = {0, 0, 127}));
  connect(speedSensor1.w, feedback1.u2) annotation(
    Line(points = {{57, 27}, {57, 22.5}, {-56, 22.5}, {-56, 45}}, color = {0, 0, 127}));
  connect(feedback1.y, PI.u) annotation(
    Line(points = {{-52.4, 48}, {-46.8, 48}}, color = {0, 0, 127}));
  connect(step1.y, feedback1.u1) annotation(
    Line(points = {{-65.6, 48}, {-59.2, 48}}, color = {0, 0, 127}));
  annotation(
    uses(Modelica(version = "3.2.3")));end submodel;
```

図 2.6: 図 2.5 の Modelica コード

最大回転数

最大回転数とは、回転数値の中で最大値を表す。

単位は、rpm(アールピーエム) である。

2.5.2 特性グラフ

今回試作したツールでは、以下の 4 つの特性グラフを作成する。

「トルク * 電流」グラフ

「トルク * 電流」グラフとは、横軸が「トルク mNm」、縦軸が「電流 mA」のグラフである。このグラフでは、トルクに対する電流の変化量を表している。

「トルク * 回転数」グラフ

「トルク * 電流」グラフとは、横軸が「トルク mNm」、縦軸が「回転数 rpm」のグラフである。このグラフでは、トルクに対する回転数の変化量を表している。

「トルク * 効率」グラフ

「トルク * 電流」グラフとは、横軸が「トルク mNm」、縦軸が「効率 %」のグラフである。このグラフでは、トルクに対する効率の変化量を表している。

「トルク * 出力」グラフ

「トルク * 電流」グラフとは、横軸が「トルク mNm」、縦軸が「出力 W」のグラフである。このグラフでは、トルクに対する出力の変化量を表している。

2.6 Python

Python は、1991 年にオランダ人のグイド・ヴァンロッサムというプログラマによって開発され、オープンソースで運営されている動的プログラミング言語である [18]。一括りに Python といってもその用途は様々で、組み込み開発や、Web アプリケーション、デスクトップアプリケーション、さらには人工知能開発、ビッグデータ解析などと多岐に渡る [19]。Python のプログラミング言語としての主な特徴は、少ないコードで簡潔にプログラムを書けること、専門的なライブラリが豊富にあることなどが挙げられる。

今回試作するモータ特性表自動生成ツールの開発言語に、Python を用いる。また、使用するライブラリを以下に示す。

- csv
- math
- matplotlib
- numpy

- decimal
- reportlab
- PIL
- pdf2image
- sys
- time
- os

第 3 章

機能

本章では、本研究で試作するモータ特性表自動生成ツールの機能について説明する。

Modelica 言語で作成したモータのモデルを、OpenModelica でシミュレーションした際に csv ファイルが出力される。今回試作するモータ特性表自動生成ツールは、OpenModelica が出力した csv ファイルを入力として読み込み、モータ特性表を出力として生成する。

3.1 モータ特性表生成

今回試作したモータ特性表自動生成ツールは、2.5 節で挙げた 9 個の要素を持つ特性表と、4 つの特性グラフを作成し、それらを 1 つの PDF ファイルにまとめ、モータ特性表として出力する。

3.2 ツールの実行

今回試作したツールを実行するためのコマンドを以下に示す。なお、このコマンドは、ツールの実行ファイルが存在するディレクトリで実行する必要がある。

第 1 引数には、入力とする csv ファイルのパスを含めたファイル名を指定する。

第 2 引数には、第 1 引数で指定した csv ファイルの中の、モータ特性表を自動生成したいモータのモデルに含まれる、慣性部品のオブジェクト名を指定する。

第 3 引数には、第 1 引数で指定した csv ファイルの中の、モータ特性表を自動生成したいモータのモデルに含まれる、電源部品のオブジェクト名を指定する。

コード 3.1: 実行コマンド

```
1 python characteristic.py 第 1引数 第 2引数 第 3引数
```

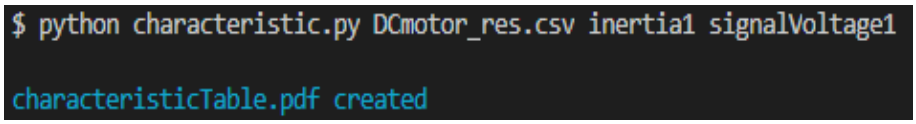

図 3.1: 実行コマンド例

第 2 引数と、第 3 引数に慣性部品と電源部品のオブジェクト名を指定する理由については、4.3 章で述べる。

図 2.1 のファイル名が、「DCmotor_res.csv」であり、慣性部品のオブジェクト名が、「inertia1」であり、電源部品のオブジェクト名が、「signalVoltage1」だった場合の実行コマンドを、図 3.1 に示す。

モータ特性表が自動生成できた場合は、図 3.1 に示すように、画面上に「characteristicTable.pdf created」と青色で表示する。

今回試作したツールでは、毎回モータ特性表を「characteristicTable.pdf」で保存するため、同じファイル名がある場合、上書き保存となる。

自動生成できなかった場合については、3.3 節で述べる。

図 3.1 のコマンドでツールを実行し、自動生成したモータ特性表を、図 3.2 に示す。

3.3 エラー表示

3.2 節で、試作したツールを実行するためのコマンドは、「python characteristic.py 第 1 引数 第 2 引数 第 3 引数」であると述べた。このコマンドを実行した際に、モータ特性表が自動生成できなかった場合は、原因が 3 つ考えられる。

1 つ目の原因は、引数の数が足りていない、もしくは多い場合である。
この場合は、エラー文として「ERROR : 引数の数が間違っています」を、赤色で画面上に表示する。この例を、図 3.3 に示す。

2 つ目の原因は、第 1 引数で指定した csv ファイルのファイル名が間違っている場合である。

この場合は、エラー文として「ERROR : 指定したファイル名が間違っています」を、赤色で画面上に表示する。この例を、図 3.4 に示す。

3 つ目の原因は、第 2 引数と第 3 引数のどちらか一方、もしくはその両方に間違いがあった場合である。

この場合は、エラー文として「ERROR : 指定したオブジェクト名が違います」を、赤色で画面上に表示する。この例を、図 3.5 に示す。

上記 3 つの原因にすべて該当するコマンドの場合は、第 1 引数から順に処理を行い、エラーがあればそこで処理を止めるため、図 3.3 のエラー文のみ表示する。

モータ特性表

電圧　V	2.4
始動電流　mA	431.9945756634572
停動トルク　mNm	0.9244683919197983
最大効率　%	99.89790026553919
定格トルク　mNm	0.0007704063898538723
定格回転数　rpm	10700.575772223889
定格電流　mA	0.3600029859130245
定格出力　W	0.0008632878742652782
最大回転数　rpm	10700.575772223889

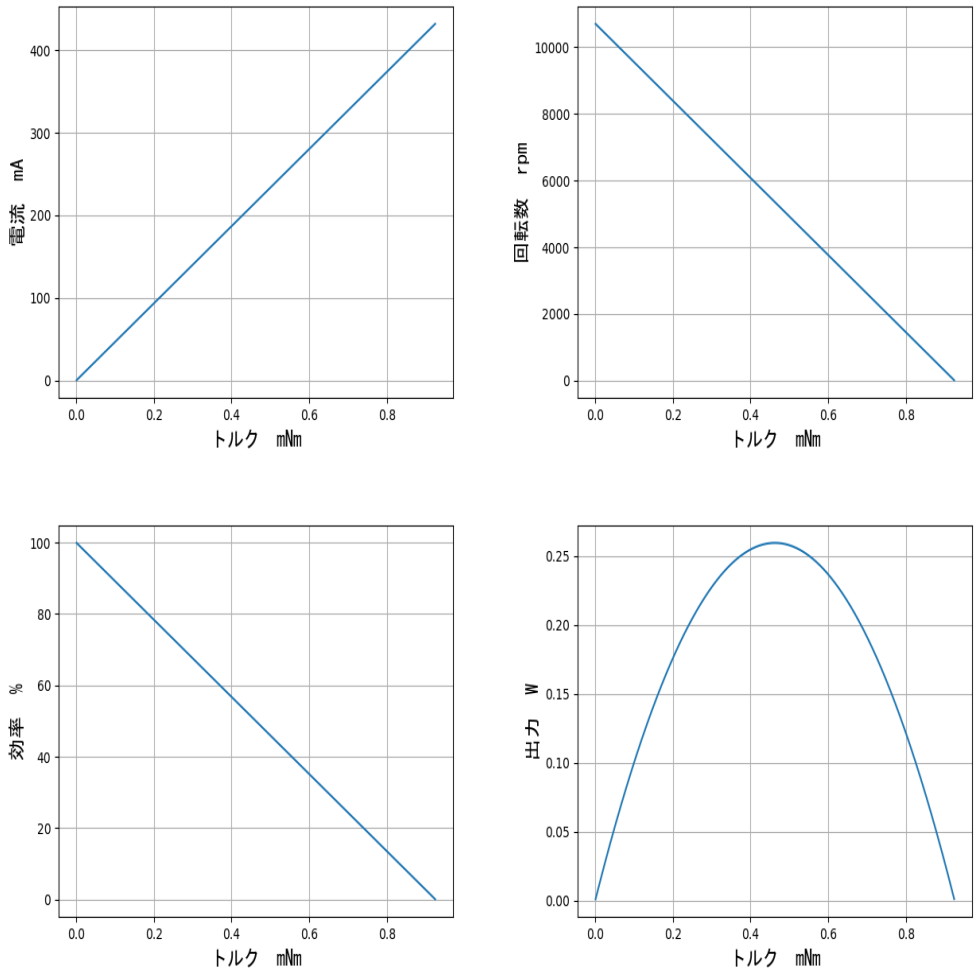

図 3.2: モータ特性表

```
$ python characteristic.py DCmotor_res.csv inertia1
ERROR : 引数の数が間違っています
```

図 3.3: 引数の数に誤りがあった場合のエラー分の例

```
$ python characteristic.py DCmoto_res.csv inertia1 signalVoltage1
ERROR : 指定したファイル名が間違っています
```

図 3.4: 第 1 引数に誤りがあった場合のエラー文の例

```
$ python characteristic.py DCmotor_res.csv inertia1 signalvoltage1
ERROR : 指定したオブジェクト名が間違っています
```

図 3.5: 第 2 引数、第 3 引数に誤りがあった場合のエラー文の例

第 4 章

実装

本章では、本研究で試作したモータ特性表自動生成ツールの実装について説明する。
モータ特性表自動生成機能の処理の流れを以下に示す。

1. 実行コマンドを取得する
2. 第 1 引数で指定された csv ファイルを読み込む
3. 第 2 引数、第 3 引数で指定したモジュール名が持つデータを、csv ファイルから取得する
4. モータ特性表の各要素を算出する
5. 特性表を作成する
6. 特性グラフを作成する
7. モータ特性表を生成する

以下、各処理について具体的に説明する。

4.1 実行コマンドの引数の取得

この処理では、Python スクリプトに渡されたコマンドライン引数のリストである sys.argv で引数を取得し、エラーの検出を行う。処理の流れを以下に示す。

1. sys.argv の要素数を取得する

2. 要素数が 4 以外であれば、図 3.3 のエラーを表示する

3. sys.argv の 1 番目のデータを、ファイル名を保存するために用意した変数に格納する

4.2 csv ファイルの読み込み

Python で実装するため、Python の標準ライブラリの csv モジュールをインポートし、csv ファイルを読み込む。以下の条件の時に、オブジェクト名が間違っていると判断する。

エラーは、以下の条件の 1 つにでも該当した場合、ファイル名が間違っていると判定し、エラー文を画面上に出力する。

- ファイルの拡張子が「.csv」ではない
- ファイルを開く際にエラーが出たとき

4.3 第 2 引数、第 3 引数で指定されたモジュールが持つデータを、csv ファイルから取得

4.2 節で読み込んだ csv ファイルから、以下のデータを取得する。

- 電流
- 電圧
- トルク
- 角速度

2.2 節で述べたように、OpenModelica から出力された csv ファイルの 1 行目には、各部品のモジュール名を含んだ変数名が記載されている。これを利用して、次の処理で必要なデータを取得する。

1. 取得したいデータを持つ変数名を検索する

2. 変数名のある場所の添字を取得する

3. 各データごとに用意した配列に、にある値を格納する

4.

5. あ

6. csv ファイルが読み終わるまでーーの処理を繰り返し行う

!!!取得したいデータを持つ変数名を探し、その変数名がある場所の添字を取得する。各データごとに用意した配列に、同じ添字の位置にある値を繰り返し処理で格納する方法でそれぞれの値を取得する。アルゴリズムっぽく

以下の条件の 1 つにでも該当した場合、指定したオブジェクト名が間違っていると判定し、エラー文を画面上に出力する。

- 電流値のある添え字を持つ変数が、初期値である「0」であった場合
- 電圧のある添え字を持つ変数が、初期値である「0」であった場合
- トルクのある添え字を持つ変数が、初期値である「0」であった場合
- 角速度のある添え字を持つ変数が、初期値である「0」であった場合

4.4 モータ特性表の各要素を算出する

4.3 節で取得したデータを用いて、3.1 節で挙げた各要素の値を求める。

4.4.1 電圧

今回試作したツールでは、電源部品の電圧値を電圧とする。

4.4.2 始動電流

モータが起動した後は、逆起電力を発生させるため、モータ・コイル部分にかかる電圧が下がり、電流値も下がる。
したがって、今回試作したツールでは、電流値の中で一番大きい値を始動電流とする。

4.4.3 停動トルク

したがって、今回試作したツールでは、トルク値の中で一番大きい値を停動トルクとする。

4.4.4 最大効率

効率は以下の式で算出する。

$$効率 = \frac{出力}{入力} * 100$$

$$出力 = 角速度 * トルク$$

$$入力 = 電圧 * 電流$$

今回試作したツールでは、効率値の中で一番大きい値を最大効率とする。

4.4.5 定格トルク

したがって、トルク値の配列の中で、最大効率のある効率値の配列の添字と同じ位置にある値が定格トルクとなる。

4.4.6 定格回転数

回転数は以下の式で算出できる。

$$回転数 = \frac{30 * 角速度}{\pi}$$

したがって、一度繰り返し処理で角速度を回転数に変換し、回転数値の配列の中で、最大効率のある効率値の配列の添字と同じ位置にある値が定格回転数となる。

4.4.7 定格電流

したがって、電流値の配列の中で、定格トルクのあるトルク値の配列の添字と同じ位置にある値が定格電流となる。

4.4.8 定格出力

定格出力は以下の式で算出できる。

$$\text{定格出力} = \text{定格トルク} * \text{定格回転数} * \frac{2\pi}{60}$$

4.4.9 最大回転数

今回試作したツールでは、回転数地の配列の中から一番大きい値を

4.5 特性表を作成

4.4 章で求めた各値と、3.1 章で挙げた各要素を、電圧から順に”,”で区切りつつ特性表生成配列に格納する。そして特性表生成配列を用いて csv ファイルを作成する。

4.6 特性グラフを作成

4.7 モータ特性表を生成

第 5 章

適用例

本章では、本研究で作成した

5.1 ブラシ付き DC モータの Modelica モデル

5.2 ブラシ付き DC モータの Modelica モデルをサブシステムとするモデル

第 6 章

考察

本論文では、モータ特性表自動生成ツールを試作した。

6.1 評価

6.1.1 評価方法

6.1.2 結果

本論文で試作したモータ特性表自動生成ツールは、

6.2 関連研究

関連研究について述べる。

6.3 ツールの問題点

以下に、今回作成したモータ特性表自動生成ツールの問題点を示す。

- 対応するモータのモデルは 1 種類しかない

  モータは～種類に分けることができ、今回は 1 つにしか対応していない。対応できる数を増やす必要がある。

第 7 章

おわりに

以下に、今後の課題を示す。

謝辞

参考文献

[1] 日本電産株式会社. 身の回りのモータ. https://www.nidec.com/jp/technology/scenes/. Accessed: 2020-1-22.

[2] 平野豊. Modelica によるモデルベースシステム開発入門. TechShare 社, 2017.

[3] Peter Fritzson, Peter Aronsson, Adrian Pop, Hakan Lundvall, Kaj Nystrom, Levon Saldamli, David Broman, and Anders Sandholm. Openmodelica-a free open-source environment for system modeling, simulation, and teaching. In *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, pp. 1588–1595. IEEE, 2006.

[4] 制御工学の基礎あれこれ. ブラシ付き dc モータの仕組み. http://arduinopid.web.fc2.com/K13.html. Accessed: 2020-1-31.

[5] 日本電産株式会社. ブラシ付き dc モータ. https://www.nidec.com/jp/technology/motor/glossary/000/0604/. Accessed: 2020-1-31.

[6] Tech Web motor. ブラシ付き dc モータの特性. https://micro.rohm.com/jp/techweb_motor/knowledge/basics/basics-03/209. Accessed: 2020-1-23.

[7] MekatoroNet. 仕様の見方. http://www.mekatoro.net/digianaecatalog/orien-sougou/book/orien-sougou-P0042.pdf. Accessed: 2020-1-30.

[8] 三菱電機エンジニアリング株式会社. 1．特長･標準仕様. http://www.mee.co.jp/sales/system-solution/machine/pdf/sales_fa_machine_kffk-mkb_kffk.pdf. Accessed: 2020-1-30.

[9] 住友重機械ギヤモータ株式会社. モータ特性表. https://cyclo.shi.co.jp/technical/pdf/tech032.pdf. Accessed: 2020-1-30.

[10] 株式会社ニッセイ. モータ特性表. https://www.nissei-gtr.co.jp/wp-content/uploads/2017/05/faq-0124-01.pdf. Accessed: 2020-1-30.

[11] 株式会社日立産機システム. 日立三相モータ. http://www.motor-shuri.com/new/images/hitachi.pdf. Accessed: 2020-1-30.

[12] MekatoroNet. モータ特性表. http://www.mekatoro.net/digianaecatalog/sumit-sougou/Book/sumit-sougou-P0615.pdf. Accessed: 2020-1-30.

[13] 東芝産業機器システム株式会社. 低圧三相かご形誘導電動機プレミアムゴールドモートル. https://ednjapan.com/edn/articles/1605/25/news042.html. Accessed: 2020-1-30.

[14] 相模マイクロ株式会社. 直流モータ特性表 dc motor characteristics. http://www.sagamimicro.co.jp/product/pdf_product/S-2230BA-24P0-384.pdf. Accessed: 2020-1-30.

[15] 三菱電機株式会社. 三菱モータ（総合カタログ）. http://dl.mitsubishielectric.co.jp/dl/fa/document/catalog/i_motor/l001040/l01040.pdf. Accessed: 2020-1-30.

[16] 富士電機株式会社. 富士低圧三相モータ. https://www.chienkung.com.tw/upload/files/%E5%AF%8C%E5%A3%ABIE3%E9%AB%98%E6%95%88%E8%83%BD%E9%A6%AC%E9%81%94.pdf. Accessed: 2020-1-30.

[17] スリーピース株式会社. モータ単体 ＊外形図・特性表＊. http://www.three-peace.com/cms/contents/data/3/4/CATALOG_4.pdf. Accessed: 2020-1-30.

[18] python Japan. Python とは. https://www.python.jp/pages/about.html. Accessed: 2020-1-31.

[19] Samurai Blog. Python とは？言語の特徴から学習法まで初心者向けにわかりやすく解説. https://www.sejuku.net/blog/7720.