電磁石水冷導体の発熱の状態推定

STATE ESTIMATION ON HEATING OF WATER-COOLED CONDUCTOR IN MAGNET

尾崎俊幸#
Toshiyuki Ozaki#
Carpio AI

Abstract

The heat in the conductor of the coil of magnet is cooled by water. In PF-AR, the bus-bar is installed between the magnet and the power supply. The bus-bar is also water-cooled, but has no interlock against water shortage. This paper proposes a new interlock by AI technology. The temperature is measured by a thermistor and the voltage sent to the input of ADC module in the building on the ground. The long cable is under the power supply operation noise. The Kalman filter treatment reduces the noise. The trend of the heating is estimated by the state-space model.

1. はじめに

電磁石を保護するために、水量インターロックと温度インターロックがある。流量は、出口に流量スイッチがあるので総量のインターロックである。4 極電磁石においては、左右に分岐するので、片方に偏れば、保護できない。ただし、温度スイッチは、左右に各 1 個ある。

施設のポンプからの母管で、セル毎に給水されるが、B-QF-SXF-B-QD-SXD と分配するに当たり、TRISTAN AR や MR では、各電磁石の必要流量を確保するため、テフロンに穴を開けたオリフィスで調整している。

AR の水冷ブスバーでは、必要水量はすくないので、1 mm ϕ の穴で調整している。水量インターロックは、適切な市販品はないので設けていない。43 カ所ある。穴が小さいので、危険度は一番大きいので、今回の研究が始まった。

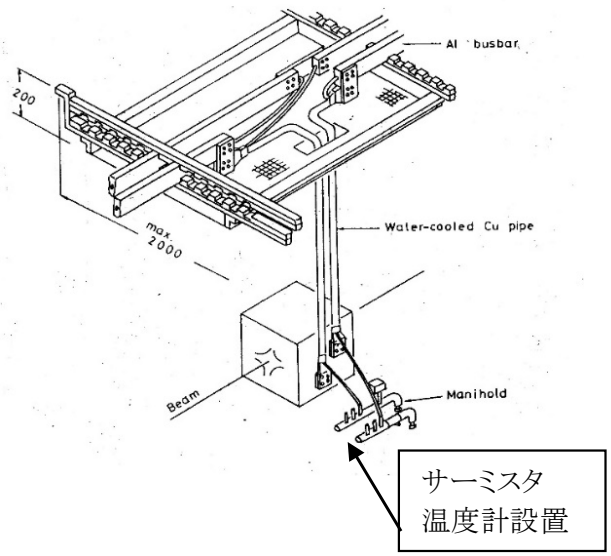

Figure 1: The magnet system in PF-AR is shown. The bus-bar is installed to feed the current. The sensor is set on the surface of the stainless pipe.

carpio.ai.185@gmail.com

オリフィスの穴は経年変化し、分配の割合は変わるだろう。アナログ信号で管理すれば、危険の程度が把握できる。しかし、記録用電子機器は、放射線に弱く、厳重なシールドを施して置くことになる。これは厄介である。電子機器は高価であり台数に限りがある。

ここでは、サーミスタを設置し、長いケーブルを使い、制御室で監視する方法の提案である。サーミスタは、半導体であるが、構造が簡単であり、その寿命は長いと予想できる。

2. 温度計測

2.1 温度計測法

4 極電磁石本体の下に、冷却母管があり、ゴムホースで 4 極電磁石のコイルに冷却水を送っている。その出口側ステンレスパイプにサーミスタを付けた。これを図1に示す。サーミスタ間の電圧を、制御室まで 30~40 m の長いケーブルを使い、制御室で AD 変換する。

電源運転中は、このケーブルに電源ノイズが乗る。この信号を AD 変換したデータは、数値の変動幅は大きい。これをカルマンフィルタで除去する。

読み込まれたデータが大きく離れた場合、欠損値として処理する。2 つ以上のセンサーが同時に高温を出したら、インターロックをするなどの対策が考えられる。

2.2 カルマンフィルタ推定

状態方程式を

$$x_{n+1} = x_n + \eta_n$$

とする。xは状態、ηをノイズとする。観測方程式を、

$$y_n = x_n + \varepsilon_n$$

とする。ここで、yは観測値、εをノイズとする。

最初の推定値をx_1とし、そのノイズ分散値をp_1と仮定する。最初のカルマンフィルタ・ゲインは、

$$K_1 = \frac{p_1}{p_1 + r}$$

である。ここで、rはノイズεの分散である。

$$x_{21} = x_1 + K_1(y_1 - x_1\ )$$
$$p_{21} = p_1\ \ (1 - K_1\ )$$

次には

$$x_2 = x_{21}$$
$$p_2 = p_{21} + q$$

となる。ここで、qはノイズηの分散値である。これを繰り返し、x_3, x_4, x_5 ...を得ていく。

2.3 マイコンへの実装

ランダムウォーク・モデルは、ほぼ一定値の周りで、データがランダム変化する状態に適する。これは、短時間では温度変化がない本研究に当てはめられる。ランダムウォークは時点が進むに連れて分散が無限に大きくなる非定常時系列である。従って、短時間だけ使用する。あまりの短時間では、仮定した初期値が収束しない。適切な判断が必要である。

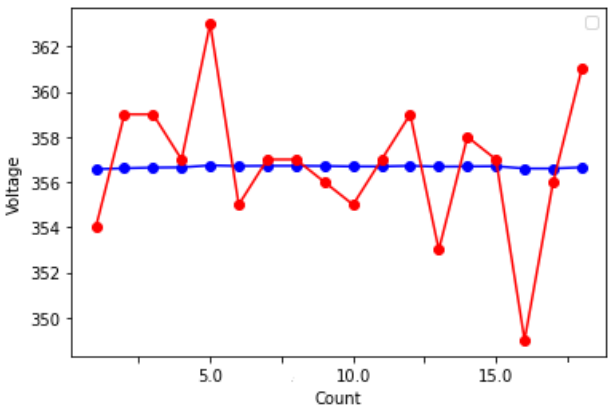

Figure 2: The red line shows input data to AD convertor. The blue line shows data after the Kalman filter treatments.

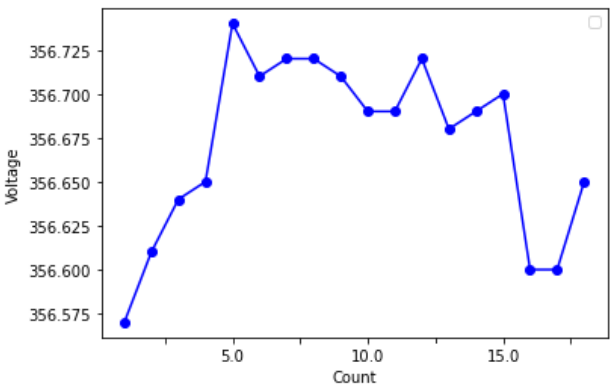

Figure 3: The blue line is data after the Kalman filer treatments. The Y-axis is enlarged compared to Fig. 2.

開発したシステムは、温度センサーとしてサーミスタを用いている。副制御室の計測モジュールの Arduino UNO からマグネット傍らまでのケーブルは、従来インターロックで使用しているシールド付きの撚線対であり、その長さは 30~40 m ある。片端はサーミスタに、他方は Arduino UNO の ADC 端子に接続されている。運転中には電源ノイズが入る。このノイズは、電源のスペクトルに関係しているが、データを読み込むタイミングが非同期であるから、白色化するだろう。同期した場合、大きな電圧になるが、欠測値として処理することにした。

文献[1]によれば、Arduino 言語のベースは C++であり、それを表面に出さないような仕組みである。したがって、外部取り合いのプログラミングは、Arduino 言語で書き、カルマンフィルタの演算は C++で書いている。本研究では、C++で書いた自作関数 Kalman を導入した。これは、ランダムウォーク・モデルである。

10bit の AD 変換器に入った電圧データを、図 2 の赤線で示す。Arduino UNO の AD 変換は 10 bit でしかないので量子化されているのが見えている。カルマンフィルタ処理した結果を青線で示す。図 2 では、あまりに直線に近いので、図 3 において、縦軸の精度を上げた。

3. 劣化推定

3.1 状態空間モデル

時系列y_nが 2 次の AR（自己回帰モデルで表現できるとする。つまり、

$$y_n = a_1\ y_{n-1} + a_2\ y_{n-2} + v_n$$

ここで、$v_n \sim N(0, \sigma^2) v_n \sim N(0, \sigma^2)$ である。

これを、カルマンフィルタの状態空間モデルで表現する[2]。

$$x_n = F_n\ x_{n-1} + G v_n \quad \text{（システムモデル）}$$
$$y_n = H_n\ x_n + R w_n \quad \text{（観測モデル）}$$

ここで、

$$x_n = \begin{bmatrix} y_n \\ y_{n-1} \end{bmatrix} \quad F = \begin{bmatrix} a_1 & a_2 \\ 1 & 0 \end{bmatrix}$$
$$G = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad H = [1 \quad 0]$$
$$Q = \sigma^2 \qquad R = 0$$

局所的に直線で変化するモデル、つまりトレンドがある場合は以下の式になる。

$$y_n = 2y_{n-1} - y_{n-2} + v_n$$

つまり、$a_1 = -2 \quad a_2 = 1$ にするので、未知変数が減る。ノイズv_nの分散σ^2は未知数θである。正規分布とする。

状態推定においては、推定値だけでなく、その分散も求める必要がある。観測値y_nの生成モデルと状態x_nの時系列モデルを

$$y_n \sim h(y_n|x_n)$$
$$x_n \sim f(x_n|x_{n-1})$$

とする。このhとfは条件付き確率密度関数である。上式での～は、左辺の確率変数が、その分布に従う事を表す。

条件付き平均　$\hat{x}_n = E(x_n|y)$
条件付き分散　$\hat{\rho}_n^2 = Var(x_n|y)$

を求める問題として、扱われる。

3.2　ベイズ統計

ベイズの定理は

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

であるが、このBの部分を「データ」に変更する。
ベイズ推定において、事後確率$p(x|y)$からxの推定値を求めるのであるが、事後確率を最大とするxをもって最適推定解とする考え方で、観測データは、最も起こりやすい事が起こった結果であるとする。

観測値$Y_j\,(j = 1,\ldots,n)$が得られた時、状態推定の問題は、Y_jのもとで状態x_nの条件付き分布$p(x_n|Y_j)$を求める問題となる。

$$p(x_n|y_n) = \frac{p(y_n|x_n)p(x_n|Y_{n-1})}{p(y_n|Y_{n-1})}$$

観測値y_nの確率分布は、過去の状態$x_1,\ldots,x_{n-1}$と観測値$y_1,\ldots,y_{n-1}$には依存しない。これはマルコフ性と呼ばれ、カルマンフィルタによって効率的に計算できる理由である。

未知数θを持つ時系列モデルの尤度は、長さNの時系列$Y_N = \{y_1, y_2, \ldots, y_N\}$が与えられる時、その同時確率密度関数を用いて

$$f_n\ (Y_n|\theta) = f_{n-1}(Y_{n-1}|\theta)p_n(y_n|Y_{n-1},\theta)$$

このような分解を繰り返し適用して、最終的には

$$\prod_{n=1}^{N} p_n\,(y_n|Y_{n-1},\theta)$$

である。この確率分布を正規分布として、尤度関数を定義する。

$$L(\theta,\sigma) = \prod_{j=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x_j-\theta)^2}{2\sigma}}$$

この式の対数を取り、偏微分して、解析的にも解ける。

観測結果は、最も起こりやすいことが起きたと考える（最尤原理）。

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$
$$= \prod_{j=1}^{N} p(X_j = x_j) \quad = \prod_{j=1}^{N} f(x_j|\theta)$$

右辺をθの関数とみて

$$L(\theta) = \prod_{j=1}^{N} f(x_j|\theta)$$

と置く。これを尤度関数と呼ぶ。ただし、$p_1(y_1|Y_n,\theta) = f_1(y_1|\theta)$である。$f$も正規分布とする。

2次のトレンドモデルは、各時刻で、水平成分と傾き成分に分けて表示する事ができる。

3.3　データ蓄積

上位を Raspberry、下位を Arduino とするシステムが普及している[3]。

Arduino は OS が動いていない。1つの処理を実行する「シングルタスク」で動作する。他方、Raspberry は OS があり、外部との通信など幅広い動作ができる。しかし、Raspberry での精密な時間制御は、Linux の仕組み上、ほぼ不可能である。また、入出力はデジタル信号しかないので、アナログ信号は外部からデジタルに変えて入れる。だから、アナログ信号の処理を Arduino に任せて、ユーザーインターフェイスを Raspberry に担当する。両者の長所・短所を利用するような連携動作が最適である。

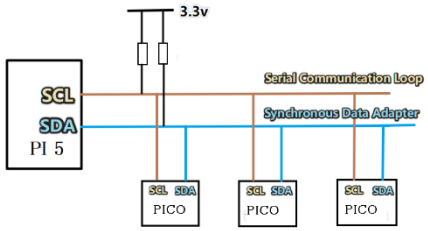

Figure 4: Raspberry PI5 as I2C master and Pico as slave are adopted.

Raspberry の電源は 3.3 V で、Arduino の電源は 5 V であり、互いの通信の間に双方向電圧レベル変換モジュールを入れる必要がある。ここでは、3.3 V で動作する PICO を用いて、SLAVE として多数並べる。

PICO は科学計測に適している[4]。Arduino IDE を立ち上げて、tools の中から、board として、“Raspberry Pi Pico”を選択する。Arduino 言語でプログラミングができる。

両者の接続は、文献[5]の Chapter 16 Arduino and Raspberry Pi に詳しい。図 4 は I2C の場合である。

3.4　数値計算

文献[6]では、電流変動をトレンドモデルで調べた。そこでは、R 言語でプログラミングした。Raspberry Pi では、Python が使えるので、これで開発している[7]。

より安定性を増すために以下の改造をした。文献[8]に従い、共分散行列の更新式を、Joseph 形式

$$P_k = (I - K_k H_k)P_k^-(I - K_k H_k)^T + K_k R_k K_k^T$$

にした。これは Pに関して対称性を有し、数値計算上の問題が発生しにくい。

4.　あとがき

本論文での議論で、後半は開発の途上である。この

技術は、AI の代表的技術である自動車の無人運転の技術に類似している。これは、センサーからの信号をカルマンフィルタでノイズを落とし、主コンピューターで、自動車の状態を推定し、最適制御する[9]。本論文で述べる内容と共通点が多い。本論文は加速器開発の種々の分野に応用できそうな一例だと認識している。

参考文献

[1] Michael Margolis, Brian Jepson & Nicholas Robert Weldin, “Arduino Cookbook Recipes to Begin, Expand, and Enhance Your Projects”, O’REILLY, April 2020.

[2] 北川源四郎, “Rによる時系列モデリング入門”, 岩波書店, 2020 年.

[3] Volker Ziemann, “A Hands-On Course in Sensors Using the Arduino and Raspberry Pi”, CRC Press, 2018.

[4] 田口海詩 他, ”ラズパイから始まる科学計測“, トランジスタ技術 Special, CQ 出版, 2025 年夏.

[5] Simon Monk, ‘Raspberry Pi Cookbook’, O’REILLY, June 2016.

[6] 尾崎俊幸, “カルマンフィルターによる励磁電流安定度の評価”, Proceedings of the 13th Annual Meeting of Particle Accelerator Society of Japan, 12271229, 2016.

[7] 馬場真哉, “Python ではじめる時系列分析入門”, 講談社, 2024 年.

[8] 廣川類, “カルマンフィルタ入門”, インターフェース, 2025 年 2 月号.

[9] 綱島均, 橋本雅文, 菅沼直樹, “カルマンフィルタの基礎と実装―自動運転・移動ロボット・鉄道への実践まで-”, 科学情報出版株式会社, 2021 年.