Estimating Ambient Temperature using Internal Sensors and Thermal Modelling in Mobile Phones

Isak Evaldsson

Henrik Paulcén

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6206
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Abstract

This thesis studies the possibility of estimating the ambient temperature around a mobile phone using its internal temperature sensors.

The thesis aims to develop and evaluate ambient temperature models that use internal temperature sensors within the device and, from a manufacturer's perspective, investigate if the models for ambient temperature can be enhanced by tailoring them to the internals of specific devices. The chosen approach was to use thermal circuits and, based on them, derive state-space models. The parameters of these models were then estimated using collected data. A linear polynomial approach was also evaluated but gave worse estimations than the state-space approaches. The best-performing model, one of the state-space models, has an estimation accuracy of within ±1–3°C and an average error of below 1.5°C when evaluated on a broad collection of testing scenarios.

Once parameter estimations have been performed, all the models have a low processor resource utilisation, which is ideal for an on-device implementation. The thermal models should theoretically be generalisable and could be used on other mobile phone models with similar internal layouts with only the need for new parameter estimations. The thermal models' principles could theoretically also be able to be used on almost all embedded devices with similar internal temperature sensors.

Acknowledgements

We want to thank Rickard Möller and Jimmy Olsson at Sony for their support and the opportunity to write this thesis in collaboration with Sony. We would also like to thank all the talented people at SW1 for all their invaluable help and warm welcome.

We would like to thank Karl-Erik Årzén at LTH for being a great supervisor and giving us valuable advice and feedback throughout the project.

And finally, we would like to give a big thanks to the coffee machine to the left and a !thanks to the annoying rotary doors at Sony's office!

Contents

1

Introduction

High temperature is an unwanted artefact of mobile phones caused by heat generated from multiple sources such as the CPU, battery, charging, and camera. The temperature constrains how a device can operate efficiently over extended periods of time without overheating the components or harming the user, especially in warmer climates. With the increased power of modern mobile phone processors and advanced cellular modems, thermal challenges will continue to be a problem in future smartphone design. Therefore, it is in all mobile phone manufacturers' interest to improve their strategies for thermal mitigation.

The mitigation of thermal issues is done in both hardware and software. Android applies multiple strategies to solve this in software, such as energy-aware scheduling and a thermal engine allowing vendors to control the hardware based on the phone's temperature [Google, 2022; Parab and Mangaonkar, 2018].

To optimise the software mitigation strategies further, it would be beneficial to know the ambient temperature of the air surrounding the device, for example, to compensate for warmer climates. An ambient temperature parameter in the device's thermal management system would allow for more precise control, and it could, for example, allow the system to rely on the cooling effect of the surrounding air and thereby be more or less aggressive when controlling the processor frequency to maximise performance without overheating the device.

The ambient temperature could be estimated using thermal sensors scattered around the device's internals. But this is not as straightforward as reading the raw sensor data since the internal thermometers get affected by ”spill heat” from different surrounding components. This means a model needs to be created to consider these factors when combining all the sensor data.

This modelling could be done using different approaches, and this thesis aims to investigate how the problem could be solved using thermal circuit heat modelling and automatic control methods. The thesis is performed in collaboration with Sony to improve their thermal mitigation strategies. Due to this collaboration, one of their devices will be used as the main testing platform, however, the models and methods can be applied to other mobile phones or devices with a similar sensor setup.

1.1 Related work

There has been some previous research on ambient temperature modelling for mobile phones. Chau has shown how the battery temperature sensor could be used to estimate the ambient temperature with a linear one-variable equation. However, the model is only built to handle when the mobile phone is idle, i.e. when only running the operating system with no apps active, and does not compensate for when the battery temperature is influenced by heat generated from internal sources such as the processor. The testing environment is also a normal room with air conditioning, lacking the accuracy of a proper lab environment [Chau, 2019].

He et al. improve upon this model by replacing the linear equation with a quadratic regression method called Support vector machine (SVM) and adding the current draw from the battery as an additional parameter compensating for internal heat. Their testing shows an average error of 1.25°C, but when applying load, it is closer to 2-3°C [He et al., 2020].

There is also some other research similar to this thesis that applies the concept of thermal circuits for thermal modelling. Ishii and Nakashima derive transfer functions for the surface temperature of the device and, in the process, also estimate the ambient temperature. But due to the focus on surface temperature, the performance of the ambient estimations is not shown [Ishii and Nakashima, 2017].

Li et al. derive a fixed point iteration algorithm based on thermal circuits to achieve an accuracy of 0.7°C. However, it is worth mentioning that the device heats a maximum of 7°C compared to idle when full CPU load is applied [Li et al., 2020], which later will be shown to be significantly less than the device used in this thesis.

The approach in this thesis differs from other related work in a few ways. Firstly, no previous work investigates how state-space models derived from thermal circuits could be used to predict ambient temperature. Secondly, other work focuses on using as few sensors as possible and allowing any type of sensors to be used, while this thesis looks at the problem from a device manufacturer perspective, i.e., which specific sensors does the device contain and which are the most interesting ones to use when estimating the ambient temperature.

1.2 Objectives

The thesis aims to develop and evaluate ambient temperature models that use internal temperature sensors within the device and, from a manufacturer's perspective, investigate if the models for ambient temperature can be enhanced by tailoring them to the internals of specific devices. The different models will be simulated, implemented, and evaluated in a laboratory environment. The goal of the project can be summarised into the following research questions.

1. How can the ambient temperature of a mobile phone be modelled using existing internal temperature sensors?

2. Which models can be used, how accurate are they, and how efficiently can they be implemented on a mobile phone?
3. How do the number of sensors and their placement within the device affect the accuracy of the models?

1.3 Limitations

To make the project feasible for the scope of a master thesis, the following limitations had to be made.

- Due to time constraints, the thermal modelling and evaluation have only been done on a single mobile phone model. Testing the thermal model on multiple platforms would be of interest to further investigate its generality. Investigating how a mobile phone case would affect the model's performance would also be interesting, but it has also not been considered due to time constraints.
- Radiation from the sun shining on the phone could make the estimation incorrect due to it adding external heat. But this scenario is not considered due to the lack of the proper testing equipment.
- In this thesis, the evaluation is only done based on data collected in a laboratory environment. This limitation was set due to practical reasons and time constraints. But further evaluating the models with field testing could provide additional information on the model's usability.

1.4 Outline

Except for the introduction, the report is divided into five additional chapters, briefly summarised below.

- *Chapter 2* introduces the theory behind thermal modelling and other relevant concepts for this thesis.
- *Chapter 3* describes the equipment and methods used to collect the data required to design and evaluate the models. It also describes the parameter estimation and evaluation procedure.
- *Chapter 4* describes the iterative process of model development by describing which models were chosen, why, and how the ambient temperature was derived from them.
- *Chapter 5* evaluates the models designed in the previous chapter, showing how the different models perform in different scenarios by providing plots

and tables with different metrics and discussing the results and the models in general.

- *Chapter 6* summarises the discussions in the previous chapter and ties them into the thesis's research questions and overall goal. It also provides a discussion on future works.

1.5 Contribution statement

During the creation of the thesis, there has been an equal contribution from both authors. The authors collaborated on data collection and model development, contributing equal work and key ideas. Both authors have worked together on the actual model evaluation implementations. However, Paulcén has mostly focused on the non-state-space implementations in Python, and Evaldsson has focused on the state-space Matlab implementations.

Regarding the writing, both authors have been involved in all chapters and in creating a report skeleton, including chapters, sections and subsections. The writing of specific sections or subsections has been divided among the authors to speed up the process. However, the authors have reviewed each other's sections and have had constant dialogue regarding all sections and their contents.

2 Theory

This chapter introduces the necessary theory required to model the ambient temperature. It will give the reader a brief overview of the physics behind heat and the theory required to build models, both thermal specific such as thermal circuits and general concepts, such as state-space and grey box modelling.

2.1 The physics of heat transfer

To model a system, some fundamental knowledge about the domain is required. This chapter, therefore, begins with a short introduction to the physics of heat and heat transfer.

Borgnakke and Sonntag define the thermodynamical definition of heat as ”the form of energy that is transferred across the boundary of a system at a given temperature to another system” [Borgnakke and Sonntag, 2013, p. 98]. If there exists a temperature differential between the two systems, then the system with the higher temperature will transfer heat energy, denoted by Q, to the other system until they achieve equal temperature [Borgnakke and Sonntag, 2013]. The relationship between heat and temperature can be compared with how a difference in voltage drives current in electronics.

The heat transfer described above can occur both within and between mediums. The different ways of heat transfer are called conduction, convection and radiation. Conduction is heat transfer within a fluid, meaning both liquids and gases, or solid medium, while convection is heat transfer between a solid and fluid, and radiation is heat transfer carried out without an intervening medium using electromagnetic waves [Incropera et al., 2007].

In the case of a mobile phone, conduction occurs within the phone internals, while convection occurs between the mobile phone's surface and the ambient air surrounding it. There can also exist radiation between the phone and heat sources such as the sun. However, as stated within the limitations in Section 1.3, this aspect will not be investigated in this thesis.

Conduction

The conductive heat transfer between two points with temperatures T_1 and T_2 and the distances L can be described using Fourier's Law,

$$q = k\frac{T_2 - T_1}{L} = k\frac{\Delta T}{L}, \tag{2.1}$$

where q (alternatively $\dot{Q}$ or $\frac{dQ}{dt}$) is the heat flux (W/m^2), i.e., the amount of heat energy transferred per unit of time and area, and k is a material-specific constant called thermal conductivity which uses the unit $W/(m \cdot K)$ [Incropera et al., 2007].

Convection

Convective heat transfer can be described with an equation similar to the one for conduction called Newton's law of cooling,

$$q = h(T_s - T_\infty). \tag{2.2}$$

In this formula, we have the surface temperature T_s of the solid material, the fluid temperature T_∞ and the convection heat transfer coefficient h expressed in $W/(m^2 \cdot K)$ [Incropera et al., 2007].

2.2 Thermal modelling

When modelling thermal behaviour, a common approach is to use a lumped-element model, meaning that the thermal system is simplified to a system of a finite amount of discrete nodes representing the different materials in the system [Incropera et al., 2007]. This method is similar to node analysis in electronics, and those similarities will continue in this section, where the concept of thermal circuits is explained.

Thermal circuits

Continuing the analogies between electronics and heat, using so-called thermal circuits is a common approach to modelling heat transfer. Sidebotham describes thermal circuits as circuits where the traditional current sources, capacitors and resistors are replaced with thermal heat sources, thermal capacitors and thermal resistors. Thermal capacitors model thermal energy stored with a material, and thermal resistors model heat change resistance and how easily heat can be transferred between materials. These thermal capacitances and resistances in the thermal circuits are an actual measurement of the thermal properties of the combined materials between the nodes. The measurement of the thermal capacitors and resistors can vary significantly if the materials are changed or altered in size. These thermal capacitors and resistors can be calculated if the physical materials and size are known [Sidebotham, 2015]. Otherwise, if the materials are unknown, they can be calculated through different estimation approaches. There is also a thermal circuit equivalent of batteries, providing a source of heat for the system instead of voltages. An example of a simple thermal circuit with a resistor (R), capacitor (C) and thermal source (T_0) can be seen in Figure 2.1.

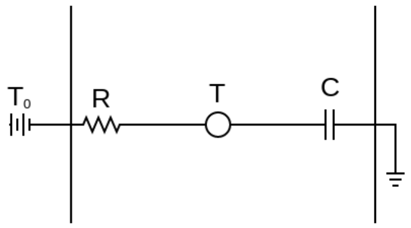


Figure 2.1 Thermal circuit example with a resistor, capacitor and temperature source.

The heat flux transferred over a resistor with the thermal resistance R between nodes with temperature T_1 and T_2 can be expressed with Equation 2.3 [Sidebotham, 2015]. The thermal resistors can be used to both model convection and conduction, and the formula closely resembles the laws defined in Equation 2.1 and 2.2.

$$q_R = \frac{T_2 - T_1}{R} \tag{2.3}$$

For a thermal capacitor in a system with the reference temperature T_{ref} and the capacitance C connected to a node with temperature T, the following heat flux is generated [Sidebotham, 2015],

$$q_c = C\frac{d(T - T_{ref})}{dt}. \tag{2.4}$$

For thermal circuits, most laws of regular electrical circuits apply, parallel and serial resistances can, for example, be combined into a single equivalent resistor.

$$\frac{1}{R_{parallel}} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots \tag{2.5}$$

$$R_{serial} = R_1 + R_2 + R_3 + \dots \tag{2.6}$$

Another important law that applies similarly to electrical circuits is Kirchhoff's current law, which states the sum of the incoming and outgoing current in a node is always zero. In the case of thermal circuits, the law can be rewritten using heat flux instead of currents, this law makes sure that each node within the circuit upholds the first law of thermodynamics, i.e., the principle of conservation of energy [Sidebotham, 2015; Incropera et al., 2007]. For a generic node with N incoming and outgoing arcs having the heat flux q_x, we get the following,

$$\sum_N q_i = 0. \tag{2.7}$$

The laws defined in Equation 2.3, 2.4, and 2.7 can be used to set up balance equations for each node within the circuit. The balance equations for the simple thermal circuit in Figure 2.1 becomes

$$q_R + q_C = 0. \tag{2.8}$$

The above-described basic building blocks can be used to setup up arbitrary large thermal circuits, allowing the modelling of highly complex thermal behaviour.

2.3 State-space modeling

A common way to model a physical system within the automatic control theory is to use a so-called state-space model. As the name implies, the system is defined by having a vector of physical variables called its state $x(t)$, and the external input signals $u(t)$. The current state and the input signal determine the system's output $y(t)$, which in the linear case can be expressed with the matrices C and D,

$$y(t) = Cx(t) + Du(t). \tag{2.9}$$

While the state change is expressed with the derivative $\dot{x}$, which depends on the current state and input signals, which similarly can be expressed using the matrices A and B in the linear case,

$$\dot{x}(t) = Ax(t) + Bu(t). \tag{2.10}$$

Combining equations 2.9 and 2.10 gives the equation system definition of the linear state-space model [Ljung and Glad, 2004].

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{2.11}$$

One important property to consider when building state-space models is the stability of the model. A state-space system is asymptotically stable if it, for any arbitrary initial state, given that no external input $u(t)$ is applied, eventually reaches the equilibrium $x = 0$. The asymptotic stability of a system can be determined by looking at the eigenvalues of the matrix A, if their real part is negative, the system is stable [Daleh et al., 2011]. An unstable system does not need to reach a point of equilibrium, meaning that the state could potentially grow towards infinity.

2.4 Methods for parameter estimation

If the parameters of a thermal system, such as resistances and capacitances, are unknown and the materials composition of these are also unknown, these could be estimated instead. Ljung and Glad define a general approach for the function $\hat{y}(t|\theta)$ modelling the system $y(t)$ using the parameter vector θ. First, they define the prediction error ε,

$$\varepsilon(t,\theta) = y(t) - \hat{y}(t|\theta). \tag{2.12}$$

The prediction error can then be used to sum the error V_N for a time series of input and output signals of length N,

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon(t,\theta). \tag{2.13}$$

The optimal parameter estimation $\hat{\theta}_N$ is then achieved by minimising the V_N [Ljung and Glad, 2004],

$$\hat{\theta}_N = \arg\min_{\theta} V_N(\theta). \tag{2.14}$$

Plenty of numerical methods exist to minimise a function, often called the objective function. A common approach is to use the derivatives, the Jacobian matrix, to be more precise, to minimise them using Newton's method. When they are hard to find or computationally heavy, a Quasi-Newton method can be used, which approximates the derivatives [Bonnans et al., 2006].

2.5 Grey-box modeling

In model development, there are two main methods: white box modelling, where the model is purely based on theory and thereby tailored to the specific system, and black box modelling, where you have a generic model structure where data is used to fit it to the specific system. Grey box modelling combines the two methods, allowing the model structure to be based on theoretical knowledge while the unknown aspects, such as parameters, are estimated using numerical methods [Bohlin, 2006]. Many modelling tools, such as Matlab, support grey box modelling [Mathworks, 2023a].

When building models from data, it is important to consider the identifiability of the system. A system $\hat{y}$ with the parameters θ_* is identifiable if there are no other parameters θ that result in the same predictions when supplied the same input data. If the system is unidentifiable, then multiple parameter sets for that system result in the same output, potentially causing trouble for an estimator by introducing unwanted properties in objective functions, such as saddle points. A system can either be unidentifiable because of its mathematical properties or due to the chosen set of input signals not providing enough inputs to be able to expose a difference in output between parameterisations [Ljung and Glad, 2004; Guillaume et al., 2019].

3 Methods

Data collection is necessary for multiple purposes to build the thermal models, such as gaining insights regarding the system behaviour, parameter estimation and model evaluation. This chapter describes how the test environment was set up, how the data was collected in different scenarios, how Python and Matlab's grey-box methods were used for the parameter estimation and how the models with their estimated parameters were evaluated.

3.1 Equipment and testing environment

To create a controllable environment where temperature data can be collected at specific ambient temperatures, a Vötsch VT 4004 heating chamber was used. The chamber allows testing in an extensive temperature range (–40°C and +130°C) and it can be programmed to change the temperature in specific sequences, allowing for complex test scenarios.

Several temperature probes were placed free-floating in the middle of the chamber to get accurate ambient temperature measurements within the heating chamber. Each temperature probe is calibrated using a pre-calibrated thermometer, specifically a G1700 from Greisinger, with a verified accuracy of 0.1 degrees Celsius. The temperature probes are all connected to a multiplexer from which the data can be imported to a computer. The connected computer can control the multiplexer by setting, for instance, the sample rate and calibrations. As a part of processing the multiplexer data, the average temperature value of all the probes is used to determine the ambient temperature. The testing environment, including the heating chamber and multiplexer with probes, can be seen in Figure 3.1.

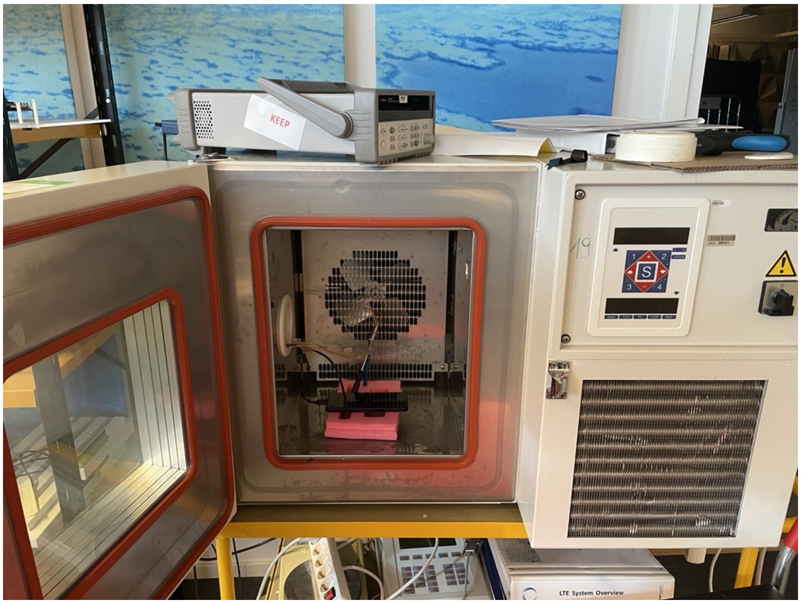


Figure 3.1 Image of the laboratory set up for collecting data, the test device is placed within the heating chamber together with the temperature probes, and the multiplexer is placed on top of the chamber.

A Sony Xperia 5 IV is used as the main testing platform. It is representative of a modern flagship smartphone, released in 2022 and equipped with a Qualcomm Snapdragon 8 gen 1 SoC [Sony Electronics, 2022]. An SoC or System-on-Chip is a single chip integrating the processor, GPU and RAM. The mobile phone is placed on a foam platform within the heating chamber to isolate it from the metal side panels of the heating chamber, eliminating sources of conduction. A computer is connected to the mobile phone to gather real-time temperature data from the mobile phone and store it. The computer can also control the mobile phone by generating computation loads or disabling settings such as USB charging, WiFi, etc., allowing the creation of specific test scenarios.

3.2 Data collection and test scenarios

The data collected from the mobile phone consists of thermal readings from about 50 sensors. The data was collected from the mobile phone during different testing scenarios with varying complexity. The tests were designed to evaluate different aspects of the phone's thermal behaviours by applying ambient temperature changes or different computational loads. The testing scenarios are explained in the following subsections.

Test scenarios

Scenario 1 - Idle state. Static ambient temperature of 20°C for 30 minutes with minimal computation load on the mobile phone with no apps actively running, often called the idle state.

Scenario 2. Scenario 2 consists of five sub-scenarios all performed with a static ambient temperature. Each sub-scenario tests a certain kind of load on the mobile phone.

1. 20°C ambient temperature with 10 minutes 100% CPU load on all cores for 10 minutes.
2. 20°C ambient temperature with charging for 50 minutes.
3. 20°C ambient temperature while using the camera filming at 1080p resolution with 60 Hz frame rate for 10 minutes and then 10 minutes idle followed by filming again but at 4k resolution with 120 Hz frame rate for 10 minutes.
4. 20°C ambient temperature while streaming YouTube at a resolution of 4k over modem for 20 minutes.
5. 50°C ambient temperature with 10 minutes 100% CPU load on all cores for 15 minutes.

Scenario 3. Test with mobile phone in idle state and temperature change. It starts at a normalised temperature of -10°C ambient temperature for 30 minutes and then increases to 50°C in ambient temperature. The mobile phone will not have any computation loads, and the total length of the scenario is 80 minutes.

Scenario 4 - Verification. Tests temperature change and different CPU loads. It starts at a normalised temperature and then either increases or decreases to another temperature point. The mobile phone will have CPU loads at different timestamps. This test is our main verification test and has the following sequence of changes:

- Starts with -10°C in ambient temperature and idle state for the mobile phone in 35min.
- CPU gets 100% load on all cores for 25 minutes.
- Ambient temperature is increased to 10°C , and the mobile phone idles for 60 minutes.
- CPU gets a load of 50% for 20 minutes.
- Idles for 150 minutes.
- CPU load of 50% for 20 minutes.

- Idles for 25 minutes.
- Ambient temperature increase to 30 °C and the mobile phone idle for 60 minutes.
- CPU load of 100% for 15 minutes.
- CPU load of 50% for 20 minutes.
- The mobile phone is in idle while the ambient temperature increases to 50°C and continues to idle for 50 minutes.
- CPU gets a load of 100% for 20 minutes.
- Idles for 75 minutes.

Scenario 5 - Training. This scenario's main purpose is to be used as training data to estimate variables for the models. This means it needs to be quite extensive scenarios, which go through almost all the possible scenarios. In other words, the longer and more complex the scenario is, the better the estimation is. The training scenario consists of the following sequence:

- Starts at a normalised ambient temperature of -10°C.
- Idle for 20 minutes.
- 100% CPU load for 10 minutes.
- Change of ambient temperature to 10°C.
- Idle for 35 minutes.
- 50% CPU load for 10 minutes.
- Idle for 75 minutes.
- 100% CPU load for 10 minutes.
- Idle for 15 minutes.
- Change of ambient temperature to 30°C.
- Idle for 30 minutes.
- 100% CPU load for 10 minutes.
- 50% CPU load for 10 minutes.
- Change of ambient temperature to 50°C.

- Idle for 30 minutes.
- 100% CPU load for 10 minutes.
- Idle for 35 minutes.
- Filming at 1080p resolution with 60Hz refresh rate for 10 minutes.
- Idle for 15 minutes.
- Filming at 4k resolution with 120Hz refresh rate for 10 minutes.
- Change of ambient temperature to 20°C.
- Filming at 1080p resolution with 60Hz refresh rate for 10 minutes.
- Idle for 15 minutes.
- Filming at 4k resolution with 120Hz refresh rate for 10 minutes.
- Idle for 15 minutes.

3.3 Parameter estimation

The parameter estimation process differs between the state-space models, where the parameters occur non-linearly, and the non-state-space linear equations and is therefore described under two different subsections. The first one describes how the linear equations were fitted to data using minimisation methods in Python, and the second one how the grey-box methods within Matlab were used to estimate the state-space models.

Estimation of linear equations

For the parameter estimations of the linear equations, Python was used. The process closely resembles the method described in Section 2.4 and is described below.

1. Define the function for the model with Python and the NumPy library.
2. Define the sum of squared errors V_N (Equation 2.13) over the training data sequence, which is described in Section 3.2.
3. Perform parameter estimation by minimising Equation 2.14 using the minimisation method in the SciPy library, which uses the BFGS algorithm[SciPy Authors, 2023a], a kind of Qausi-Newton method. The initial guess for all parameters was set to 1.

Grey-box estimation of state-space models

For the state-space models, Matlab's grey-box modelling tools were used. To define the state-space models as shown in Equation 2.11, the `idgrey` method was used, which requires a function to be supplied defining the matrices A, B, C and D based on a set of user-defined parameters [Mathworks, 2023c]. The exact structure of the state-space models, i.e. the number of states, inputs, parameters and matrices, varies among the presented models and is therefore described for each model in Section 4.2, where each model has there own subsection *Parameter estimation*.

Once the models are defined, their parameters need to be estimated, which was done using the Matlab method `greyest` [Mathworks, 2023b]. As described in more detail in Section 4.2, the parameters of models consist of a combination of capacitances, resistances and, in some models, weights defining their matrices. To initialise the estimation process, some initial values parameter values need to be set, these initial values are based on qualified guesses and testing, which gives the best results. Except for an initial model, the parameter estimator needs to be provided with training data as well, the training data set is defined in 3.2 together with some other test scenarios used for evaluation.

3.4 Model evaluation metrics

For the model evaluation, a few different metrics to compute the average error were used to make our results comparable to as many other studies as possible. The following error metrics were chosen [Ljung and Glad, 2004; Jha et al., 2023], where y is the true value and $\hat{y}$ is prediction:

- Root mean square error:

$$RMSE(y,\hat{y}) = \sqrt{\frac{\sum_{i=1}^{N}(y_i-\hat{y}_i)^2}{N}} \tag{3.1}$$

- Mean absolute error:

$$MAE(y,\hat{y}) = \frac{\sum_{i=1}^{N}|y_i-\hat{y}_i|}{N} \tag{3.2}$$

- Mean bias error (equivalent to V_N in equation 2.13):

$$MBE(y,\hat{y}) = \frac{\sum_{i=1}^{N}(y_i-\hat{y}_i)}{N} \tag{3.3}$$

To further evaluate the usability of the predictions made by the models, confidence intervals at 50%, 90% and 98% for the error of an individual prediction were computed. The sample size of one when computing the confidence intervals was chosen since it represents the range error that can be expected for each individual prediction.

Assuming that the errors $x_i = y_i - \hat{y}_i$ are normally distributed, the mean μ and the standard deviation σ are defined like this,

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N} \tag{3.4}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N-1}}. \tag{3.5}$$

Once μ and σ are known, the confidence interval at the given percentage $1-\alpha$ becomes

$$\mu \pm \lambda_{\alpha/2} \cdot \sigma, \tag{3.6}$$

where λ_α corresponds to the value of the standardised normal distribution where $P(X > \lambda_\alpha) = \alpha$ [Blom et al., 2017]. The actual implementations used SciPy's `stats.norm` and Matlab's `fitdist` methods for computing standard deviations and normal distributions [SciPy Authors, 2023b; Mathworks, 2023d].

4

Modelling

This chapter shows the model development process by describing which models were chosen and why, their mathematical definitions and how they could be improved. For each model, it mentions the specific parameter configuration used when estimating the parameters. The chapter focuses mainly on model design and will only briefly mention performance, the full performance evaluation of all the models is presented in more detail in Chapter 5.

4.1 The system

To model the system, it is necessary to have an understanding of how the system behaves. About 50 temperature sensors can be read in real-time, but most are either inside the SoC or located close to it on the mainboard. These sensors are heavily affected by heat from the SoC, meaning they are less useful. The more valuable sensors are oriented further from the SoC, like the flashlight, battery, display and USB port thermal sensors. To illustrate this, the sensor temperatures when applying a processor load are shown in Figure 4.1. To make the figure more readable, only two main-board sensors are shown. To understand the models presented in this chapter, the reader will need a basic understanding of the phone's hardware layout, as shown in Figure 4.2.

4.2 Thermal circuit approach

The primary focus of this thesis was modelling using thermal circuits. In this section, the thermal circuits and the models derived from them are described in the same order as they were designed, presenting the iterative processes of model improvement.

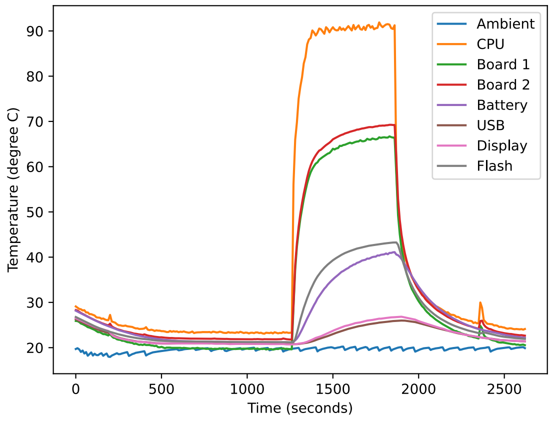


Figure 4.1 Raw sensor temperature readings when applying a processor load in Scenario 2.1. Note that when the load is applied, the CPU temperature reaches about 90°C. Board 1 and Board 2 are two sensors on the main PCB board close to the CPU.

Thermal circuit for Model 1

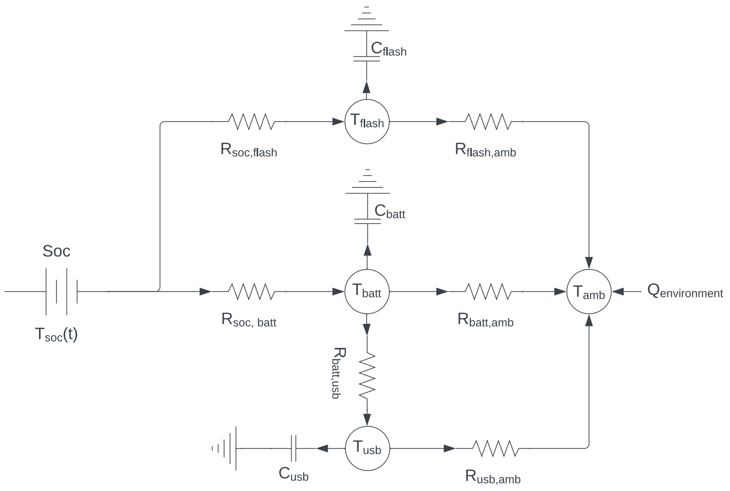


Figure 4.3 Model 1 thermal circuit of the mobile phone, where the T nodes are readings from the corresponding temperature sensor, except for T_{amb}, which is the estimated ambient temperature. SoC is the CPU temperature.

The thermal circuit for Model 1, which can be seen in Figure 4.3, uses three thermal sensors as reference nodes: the sensor in the camera flashlight, a sensor within the

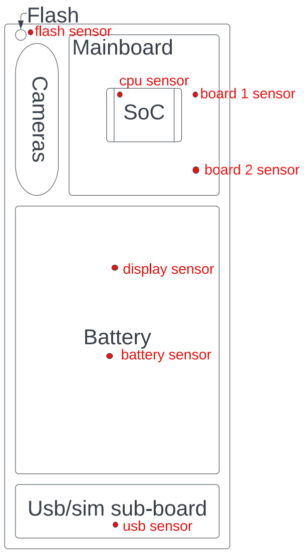


Figure 4.2 Simplified hardware layout schematic of the mobile phone, the sensors plotted in Figure 4.1 is marked in red.

battery and a sensor in the USB contact. They were selected based on their position relative to the heat source, see Figure 4.2, and how much they were affected by it, where the less affected sensors were more appealing since they have a stronger correlation with the ambient temperature. The circuit has a single heat source, namely the SoC or, more precisely, the CPU within the SoC. It has arcs going to the battery and flashlight nodes since they are directly connected to the main board housing the SoC. The USB sensor is located far from the SoC, with the battery acting as a thermal buffer in between, which is modelled by having an arc between the battery and the USB node. Arcs go from each thermal sensor node to an ambient temperature node representing how the device interacts with the environment.

Based on the thermal circuit for Model 1, the following balance equations can be defined, describing the relationship between the node's temperatures. Note that in this and all subsequent equations, the temperatures are functions of time, the time $T_x(s)$ is not written out to keep the equations brief.

$$\begin{cases} -\frac{T_{soc}-T_{flash}}{R_{soc,flash}} + \frac{T_{flash}-T_{amb}}{R_{flash,amb}} + C_{flash}\frac{d[T_{flash}-T_{amb}]}{dt} = 0 \\ -\frac{T_{soc}-T_{batt}}{R_{soc,batt}} + \frac{T_{batt}-T_{amb}}{R_{batt,amb}} + \frac{T_{usb}-T_{batt}}{R_{usb,batt}} + C_{batt}\frac{d[T_{batt}-T_{amb}]}{dt} = 0 \\ -\frac{T_{batt}-T_{usb}}{R_{batt,usb}} + \frac{T_{usb}-T_{amb}}{R_{usb,amb}} + C_{usb}\frac{d[T_{usb}-T_{amb}]}{dt} = 0 \end{cases} \tag{4.1}$$

From the thermal circuit and balance equations defined above, the ambient temperature can be derived in multiple ways. The different derivations done in this thesis are shown in the sections below. To make it easier to understand the results in Chapter 5, the titles of the sections are named the same as within the tables.

Model 1 Kirchhoff

The simplest way to achieve the ambient temperature is to apply Kirchhoff's current law to the ambient node, combining all the arcs resulting in Equation 4.2, where the ambient temperature is based on the adjacent nodes' temperatures and the resistance between them and the ambient node, modelling the behaviour of the system once the transients caused by capacitors have stabilised. Since the heat stored within the capacitors is ignored, the processor temperature driving heat flow towards the nodes can be ignored, only the temperatures of the nodes are of interest.

$$0 = \frac{T_{flash} - T_{amb}}{R_{flash,amb}} + \frac{T_{batt} - T_{amb}}{R_{batt,amb}} + \frac{T_{usb} - T_{amb}}{R_{usb,amb}} + q_e \tag{4.2}$$

By replacing all the resistances with their corresponding conductances

$$G_f = \frac{1}{R_{flash,amb}}, G_b = \frac{1}{R_{batt,amb}}, G_u = \frac{1}{R_{usb,amb}}. \tag{4.3}$$

Equation 4.2 can be solved for T_{amb}. resulting in Equation 4.4,

$$(G_f + G_b + G_u) \cdot T_{amb} = G_f \cdot T_{flash} + G_u \cdot T_{usb} + G_b \cdot T_{batt} + q_e \tag{4.4}$$

where q_e is the heat flux contributed to the ambient air from other sources than the mobile phone. This heat flux is impossible to measure for the mobile phone itself and can therefore be seen as the estimation error.

Parameter estimation. The parameter estimation for *Model 1 Kirchhoff* is done through the Python library SciPy, which is described in Section 3.3. All the parameters were initially set to one, and the final parameters after estimation can be seen in Table 4.1.

As mentioned earlier, the variable q_e represents heat flow from the environment around the mobile phone, which cannot be measured in the general case. Setting this value to zero would assume the temperature of the surrounding air only to be influenced by the mobile phone, which is incorrect and would potentially yield bad results. Through experimentation, the value of 1 was chosen since it worked well in the heating chamber, for better accuracy in a more complex real-world scenario, a more sophisticated estimation technique would have to be used.

Model 1 Kirchhoff	$R_{batt,amb}$	$R_{usb,amb}$	$R_{flash,amb}$
Parameters	$2.28 \cdot 10^5$	0.01	$2.32 \cdot 10^5$

Table 4.1 Estimated parameters for *Model 1 Kirchhoff*.

The estimated value of the parameters suggests that the heat transfer between USB and ambient is a factor 10^5 times greater than for the other two nodes, which is physically unrealistic. The estimator itself has no intuition for physics and does only find the parameters yielding the best result, meaning there are some inaccuracies within the model causing these parameters. The hard to accurately estimate q_e might be one of the causes for the weird parameters.

Model 1 State-space

Model 1 Kirchhoff takes only the thermal resistances into account, however, the nodes have thermal capacitors embedded in them. Hence, the model could be improved by also considering the capacitances. By re-writing the balance equations in 4.1 using the term $\Delta T_x = T_x - T_{amb}$ we achieve a system of first-order differential equations.

$$\begin{cases} C_{flash}\dot{\Delta T_{flash}} = -\frac{\Delta T_{flash}}{R_{f,amb}} + \frac{T_{soc}-T_{flash}}{R_{soc,f}} \\ C_{batt}\Delta\dot{T}_{batt} = -\frac{\Delta T_{batt}}{R_{batt,amb}} - \frac{T_{usb}-T_{batt}}{R_{usb,batt}} + \frac{T_{soc}-T_b}{R_{soc,batt}} \\ C_{usb}\Delta\dot{T}_{usb} = -\frac{\Delta T_{usb}}{R_{usb,amb}} + \frac{T_{batt}-T_{usb}}{R_{batt,usb}} \end{cases} \tag{4.5}$$

This makes it possible to express the circuit as a continuous-time state-space model

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{4.6}$$

where the output $y(t)$ corresponds to the ambient temperature, the states $x(t)$ to the temperature deltas, and the input $u(t)$ to the sensor value readings.

$$x(t) = \begin{bmatrix} \Delta T_{flash} \\ \Delta T_{batt} \\ \Delta T_{usb} \end{bmatrix}, \quad u(t) = \begin{bmatrix} T_{flash} \\ T_{batt} \\ T_{usb} \\ T_{soc} \end{bmatrix} \tag{4.7}$$

The matrices A and B can be obtained directly from Equation 4.5.

$$A = \begin{bmatrix} -\frac{1}{C_{flash}R_{flash,amb}} & 0 & 0 \\ 0 & -\frac{1}{C_{batt}R_{batt,amb}} & 0 \\ 0 & 0 & -\frac{1}{C_{usb}R_{usb,amb}} \end{bmatrix} \tag{4.8}$$

$$B = \begin{bmatrix} -\frac{1}{R_{soc,flash}C_{flash}} & 0 & 0 & \frac{1}{R_{soc,f}C_{flash}} \\ 0 & \frac{1}{C_{batt}}\left(\frac{1}{R_{usb,batt}} - \frac{1}{R_{soc,batt}}\right) & -\frac{1}{R_{usb,batt}C_{batt}} & \frac{1}{R_{soc,batt}C_{batt}} \\ 0 & \frac{1}{R_{batt,usb}C_{usb}} & -\frac{1}{R_{batt,usb}C_{usb}} & 0 \end{bmatrix} \tag{4.9}$$

The C matrix defines how the ambient temperature depends on the states, which is the temperature delta between each sensor and the ambient. Since the sensor values are known, you could easily compute the estimation for each node $T_{amb} = \Delta T_{node} + T_{node}$, but they need to be combined into a singular estimation. One way to do it is to replace the three arcs in Figure 4.3 with a singular arch with the temperature T_x, producing the same amount of heat flux as the individual arcs combined.

$$\frac{T_x - T_{amb}}{R_{parrallel}} = \frac{\Delta T_{flash}}{R_{f,amb}} + \frac{\Delta T_{batt}}{R_{batt,amb}} + \frac{\Delta T_{usb}}{R_{usb,amb}} \tag{4.10}$$

From this, a single ambient temperature estimation can be derived. The parallel resistances $R_{parrallel}$ can be computed using the law of parallel resistances in Equation 2.5. However, there is still one unknown, the temperature T_x, which could be estimated as a linear combination of the sensor temperatures with weights w_1, w_2 and w_3, resulting in the matrices C and D.

$$C = -\frac{1}{\frac{1}{R_{flash,amb}} + \frac{1}{R_{batt,amb}} + \frac{1}{R_{usb,amb}}} \begin{bmatrix} \frac{1}{R_{flash,amb}} & \frac{1}{R_{batt,amb}} & \frac{1}{R_{usb,amb}} \end{bmatrix} \tag{4.11}$$

$$D = \begin{bmatrix} w_1 & w_2 & w_3 & 0 \end{bmatrix} \tag{4.12}$$

Parameter estimation. When using grey-box estimation on *Model 1 State-space*, the supplied function to `idgrey` produces the matrices A, B, C and D defined above using the parameters described in Table 4.2. The model has three states, and the input consists of three temperature sensors and the average temperature from the processor temperature probes within the SoC. All the parameters were set to free, and the capacitances and resistances were restricted to above zero, the requirement of positive values is done both for physical correctness and having the eigenvalues of A negative, thereby keeping the system stable. The weights w_x were held between $[0,1]$ since they represent a weighted average of the three temperature sensors.

Model 1 State-space single input

Another way to formulate the state-space model is to use the sensor temperatures instead of the deltas as the state vector and have the heat source as the only input.

$$x(t) = \begin{bmatrix} T_{flash} \\ T_{batt} \\ T_{usb} \end{bmatrix}, \quad u(t) = \begin{bmatrix} T_{soc} \end{bmatrix} \tag{4.13}$$

This requires some modifications of the A and B matrices. To achieve an equation system containing $\dot{x}$ similar to Equation 4.5, the capacitance expressions must be rewritten only to contain the node temperature T_f, T_b or T_u, which, according to Equation 2.4 can be achieved by assuming a reference temperature of zero for the

Parameter	Free	Min/Max	Initial Values	Estimated Values
C_{flash}	Yes	$[0,\infty]$	1	4.45
C_{batt}	Yes	$[0,\infty]$	1	0.47
C_{usb}	Yes	$[0,\infty]$	1	0.20
$R_{flash,amb}$	Yes	$[0,\infty]$	1	4.05
$R_{batt,amb}$	Yes	$[0,\infty]$	1	0.67
$R_{usb,amb}$	Yes	$[0,\infty]$	1	0.01
$R_{soc,flash}$	Yes	$[0,\infty]$	1	0.44
$R_{soc,batt}$	Yes	$[0,\infty]$	1	0.53
$R_{batt,usb}$	Yes	$[0,\infty]$	1	1.60
w_1	Yes	$[0,1]$	0.33	0
w_2	Yes	$[0,1]$	0.33	0
w_3	Yes	$[0,1]$	0.33	0.96

Table 4.2 The grey-box parameter setup for *Model 1 State-space*. The *Free* column indicates if the parameters were free, i.e. if the grey-box estimator was allowed to change it or not.

thermal system. With this assumption, the following matrices are obtained.

$$A = \begin{bmatrix} -\frac{1}{R_{soc,flash}C_{flash}} & 0 & 0 \\ 0 & \frac{1}{C_{batt}}\left(-\frac{1}{R_{soc,batt}} + \frac{1}{R_{usb,batt}}\right) & -\frac{1}{R_{usb,batt}C_{batt}} \\ 0 & \frac{1}{R_{batt,usb}C_{usb}} & -\frac{1}{R_{batt,usb}C_{usb}} \end{bmatrix} \tag{4.14}$$

$$B = \begin{bmatrix} \frac{1}{R_{soc,flash}C_{flash}} \\ \frac{1}{R_{soc,batt}C_{batt}} \\ 0 \end{bmatrix} \tag{4.15}$$

Since the state differs from the other state-space representation, the C and D matrices must also change. Because the state is equal to the node temperatures, Kirchhoff's laws can be used similarly to Equation 4.4 with the same definition of conductances as in Equation 4.3.

$$C = \frac{1}{G_f + G_b + G_u}\begin{bmatrix} G_f & G_u & G_b \end{bmatrix} \tag{4.16}$$

$$D = [0] \tag{4.17}$$

Parameter estimation. For the grey-box estimation of *Model 1 State-space single input*, the supplied function to `idgrey` produces the matrices A, B, C and D defined above using the parameters described in Table 4.3. The model has three states, and the input T_{soc} is the average temperature from the processor temperature probes within the SoC. It is worth noting that the parameter C_{flash} was constrained to 1 since it is always multiplied with $R_{soc,flash}$ when used in A and B, meaning that it's unidentifiable. The capacitances and resistances were restricted to above zero, the requirement of positive values is done both for physical correctness and to keep the system stable.

Parameter	Free	Min/Max	Initial Value	Estimated Values
C_{flash}	no	$[0,\infty]$	1	1
C_{batt}	yes	$[0,\infty]$	1	199.9
C_{usb}	yes	$[0,\infty]$	1	5172
$R_{flash,amb}$	yes	$[0,\infty]$	1	223.9
$R_{batt,amb}$	yes	$[0,\infty]$	1	156.6
$R_{usb,amb}$	yes	$[0,\infty]$	1	272.3
$R_{soc,flash}$	yes	$[0,\infty]$	1	$1.1 \cdot 10^4$
$R_{soc,batt}$	yes	$[0,\infty]$	1	2.59
$R_{batt,usb}$	yes	$[0,\infty]$	1	5176

Table 4.3 The grey-box parameter setup for *Model 1 State-space single input*.

Thermal circuit for Model 2

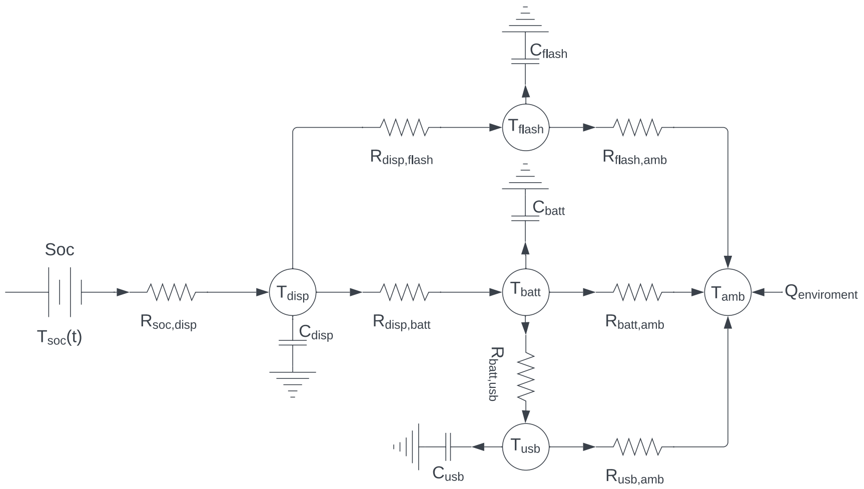
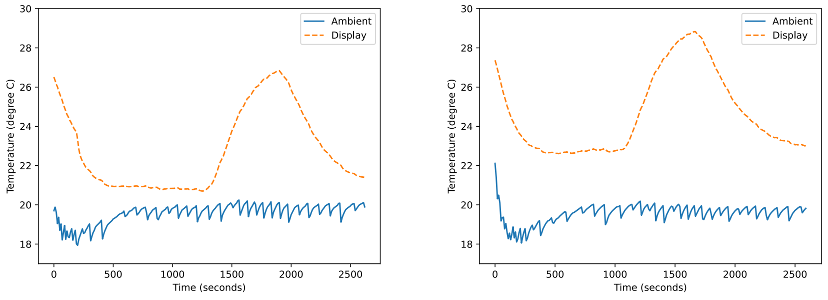


Figure 4.4 Model 2 thermal circuit of the mobile phone. The T nodes are readings from the corresponding temperature sensor, except for T_{amb}, which is the estimated ambient temperature. SoC is the CPU temperature. Compared with Model 1, a state has been added for the display.

The thermal circuit for Model 2, which can be seen in Figure 4.4, is the next iterative step from Model 1 in Figure 4.3. It was developed to suppress the small spikes which occur because the SoC increases its temperature quickly when put under load, as an example, see Figure 5.2 in Chapter 5. To solve this issue, it was desired to find a sensor that changes with the SoC temperature but less aggressively, thereby acting as a thermal buffer between the SoC and the other nodes. The thermal sensor behind the display node was suitable since it represents how the heat travels through the phone. The display covers the whole mobile phone and is one of the main avenues of heat transfer due to its metal plating. The display itself does not generate any significant heat, as shown in Figure 4.5, showing the sensor value when the display is on at 100% brightness compared to when it is off. Changing the circuit does not change the model that is purely based on Krichhoff's law, therefor there is no Model 2 equivalent to *Model 1 Krichhoff*.



(a) Display off **(b)** Display on

Figure 4.5 Thermal readings from the display temperature sensor when the display is off respectively on, and CPU gets 100% load on all cores for 10 minutes. The plots show only a difference of about 1–2°C.

These changes led to a new set of balance equations similar to Model 1, the difference is that T_{soc} and $R_{soc,x}$ have been replaced with the display node equivalents.

$$\begin{cases} -\frac{T_{disp}-T_{flash}}{R_{disp,flash}}+\frac{T_f-T_{amb}}{R_{flash,amb}}+C_{flash}\frac{d[T_{flash}-T_{amb}]}{dt}=0 \\ -\frac{T_{disp}-T_{batt}}{R_{disp,batt}}+\frac{T_{batt}-T_{amb}}{R_{batt,amb}}+\frac{T_{usb}-T_{batt}}{R_{usb,batt}}+C_{batt}\frac{d[T_{batt}-T_{amb}]}{dt}=0 \\ -\frac{T_{batt}-T_{usb}}{R_{batt,usb}}+\frac{T_{usb}-T_{amb}}{R_{usb,amb}}+C_{usb}\frac{d[T_{usb}-T_{amb}]}{dt}=0 \end{cases} \tag{4.18}$$

As for Model 1, the ambient temperature can be derived in multiple ways, which are shown in the sections below.

Model 2 State-space

Using the same procedure as for Model 1, a similar state-space representation can be derived. Similar to the balance equation, the difference between Model 1 is the heat source and its resistances.

$$x(t) = \begin{bmatrix} \Delta T_{flash} \\ \Delta T_{batt} \\ \Delta T_{usb} \end{bmatrix}, \quad u(t) = \begin{bmatrix} T_{flash} \\ T_{batt} \\ T_{usb} \\ T_{disp} \end{bmatrix} \tag{4.19}$$

$$A = \begin{bmatrix} -\frac{1}{C_{flash}R_{flash,amb}} & 0 & 0 \\ 0 & -\frac{1}{C_{batt}R_{batt,amb}} & 0 \\ 0 & 0 & -\frac{1}{C_{usb}R_{usb,amb}} \end{bmatrix} \tag{4.20}$$

$$B = \begin{bmatrix} -\frac{1}{R_{disp,flash}C_{flash}} & 0 & 0 & \frac{1}{R_{disp,flash}C_{flash}} \\ 0 & \frac{1}{C_{batt}}\left(\frac{1}{R_{usb,batt}} - \frac{1}{R_{disp,batt}}\right) & -\frac{1}{R_{usb,batt}C_{batt}} & \frac{1}{R_{disp,batt}C_{batt}} \\ 0 & \frac{1}{R_{batt,usb}C_{usb}} & -\frac{1}{R_{batt,usb}C_{usb}} & 0 \end{bmatrix} \tag{4.21}$$

$$C = -\frac{1}{\frac{1}{R_{flash,amb}} + \frac{1}{R_{batt,amb}} + \frac{1}{usb,amb}} \begin{bmatrix} \frac{1}{R_{flash,amb}} & \frac{1}{R_{batt,amb}} & \frac{1}{R_{usb,amb}} \end{bmatrix} \tag{4.22}$$

$$D = \begin{bmatrix} w_1 & w_2 & w_3 & 0 \end{bmatrix} \tag{4.23}$$

Parameter estimation. When using grey-box estimation on *Model 2 State-space*, the supplied function to `idgrey` produces the matrices A, B, C and D defined above using the parameters described in Table 4.4. The model has three states, and the input consists of three temperature sensors and the display temperature sensor. All the parameters were set to free, and the capacitances and resistances were restricted to above zero, the requirement of positive values is done both for physical correctness and having the eigenvalues of A negative, thereby keeping the system stable. The weights w_x were kept between $[0,1]$ since they represent a weighted average of the three temperature sensors.

Model 2 State-space single input

Using the same procedure as for Model 1, a similar single input state-space representation can be derived. Similar to the other state-space model, the difference between model 1 is the changed heat source and its resistances.

$$x(t) = \begin{bmatrix} T_{flash} \\ T_{batt} \\ T_{usb} \end{bmatrix}, \quad u(t) = \begin{bmatrix} T_{disp} \end{bmatrix} \tag{4.24}$$

Parameter	Free	Min/Max	Initial Values	Estimated Values
C_{flash}	Yes	$[0,\infty]$	1	14.14
C_{batt}	Yes	$[0,\infty]$	1	37.72
C_{usb}	Yes	$[0,\infty]$	1	0.08
$R_{flash,amb}$	Yes	$[0,\infty]$	1	11.57
$R_{batt,amb}$	Yes	$[0,\infty]$	1	36.74
$R_{usb,amb}$	Yes	$[0,\infty]$	1	0.18
$R_{disp,flash}$	Yes	$[0,\infty]$	1	9.59
$R_{disp,batt}$	Yes	$[0,\infty]$	1	3.82
$R_{batt,usb}$	Yes	$[0,\infty]$	1	11.71
w_1	Yes	$[0,1]$	0.33	0.42
w_2	Yes	$[0,1]$	0.33	0
w_3	Yes	$[0,1]$	0.33	0.94

Table 4.4 The grey-box parameter setup for *Model 2 State-space*.

$$A = \begin{bmatrix} -\frac{1}{R_{disp,flash}C_{flash}} & 0 & 0 \\ 0 & \frac{1}{C_{batt}}\left(-\frac{1}{R_{disp,batt}} + \frac{1}{R_{usb,batt}}\right) & -\frac{1}{R_{batt,usb}} \\ 0 & \frac{1}{R_{batt,usb}C_{usb}} & -\frac{1}{R_{batt,usb}C_{usb}} \end{bmatrix} \tag{4.25}$$

$$B = \begin{bmatrix} \frac{1}{R_{disp,flash}C_{flash}} \\ \frac{1}{R_{disp,batt}C_{batt}} \\ 0 \end{bmatrix} \tag{4.26}$$

$$C = \frac{1}{G_f + G_b + G_u}\begin{bmatrix} G_f & G_u & G_b \end{bmatrix} \tag{4.27}$$

$$D = [0] \tag{4.28}$$

Parameter estimation. The grey-box estimation of *Model 2 State-space single input* uses almost the same matrices A, B, C and D and parameters for `idgrey` as the model 1 counterpart. There is also a change in input, which is the value of the display sensors (T_{disp}) instead of the CPU temperature (T_{SoC}).

Model 2 state-space random C & D

Based on observations when developing the other models, it could be seen that *Model 2 state-space* model performed quite well when letting both the C and D matrix be filled up with free parameters. Due to its performance, it was added as an additional model called *Model 2 state-space random C & D*, where $x(t)$, $u(t)$, A and B are the same as for *Model 2 state-space*, but C and D are replaced with the following matrices,

$$C = \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} \tag{4.29}$$

$$D = \begin{bmatrix} w_4 & w_5 & w_6 & w_7 \end{bmatrix}. \tag{4.30}$$

Parameter	Free	Min/Max	Initial Value	Estimated Values
C_{flash}	no	$[0,\infty]$	1	1
C_{batt}	yes	$[0,\infty]$	1	0.13
C_{usb}	yes	$[0,\infty]$	1	3.26
$R_{flash,amb}$	yes	$[0,\infty]$	1	2.21
$R_{batt,amb}$	yes	$[0,\infty]$	1	0.10
$R_{usb,amb}$	yes	$[0,\infty]$	1	1.05
$R_{disp,flash}$	yes	$[0,\infty]$	1	0.19
$R_{disp,batt}$	yes	$[0,\infty]$	1	2.33
$R_{batt,usb}$	yes	$[0,\infty]$	1	2.26

Table 4.5 The grey-box parameter setup for *Model 2 State-space single input*.

Parameter estimation. When using grey-box estimation on *Model 2 State-space random C & D*, the supplied function to `idgrey` produces the matrices C and D defined above as well as the matrices A and B from *Model 2 State-space*. Since this model is largely based on *Model 2 State-space*, it has almost the same parameter configuration but with two exceptions: there are four extra weights, and the range constraints of the weights are removed. The full parameter setup can be seen in Table 4.6.

Parameter	Free	Min/Max	Initial Value	Estimated Value
C_{flash}	yes	$[0,\infty]$	1	4.16
C_{batt}	yes	$[0,\infty]$	1	1.98
C_{usb}	yes	$[0,\infty]$	1	1.82
$R_{flash,amb}$	yes	$[0,\infty]$	1	3.96
$R_{batt,amb}$	yes	$[0,\infty]$	1	10.34
$R_{usb,amb}$	yes	$[0,\infty]$	1	11.05
$R_{disp,flash}$	yes	$[0,\infty]$	1	1.66
$R_{disp,batt}$	yes	$[0,\infty]$	1	30.09
$R_{batt,usb}$	yes	$[0,\infty]$	1	0.76
w_1	yes	$[-\infty,\infty]$	0.33	0.29
w_2	yes	$[-\infty,\infty]$	0.33	11.43
w_3	yes	$[-\infty,\infty]$	0.33	10.98
w_4	yes	$[-\infty,\infty]$	0.33	0.27
w_5	yes	$[-\infty,\infty]$	0.33	-1.68
w_6	yes	$[-\infty,\infty]$	0.33	-4.34
w_7	yes	$[-\infty,\infty]$	0.33	6.72

Table 4.6 The grey-box parameter setup for *Model 2 State-space random C & D*.

4.3 Linear approach

The simplest approach is a linear equation fitted to one thermal sensor's value and the ambient temperature. Chau applies this approach to the battery sensor [Chau, 2019]. In contrast to the thermal circuit-based models, this approach will be applied to both the battery, USB and flash sensors separately, resulting in the following linear equations,

$$T_{amb} = k \cdot T_{batt} + m \tag{4.31}$$

$$T_{amb} = k \cdot T_{usb} + m \tag{4.32}$$

$$T_{amb} = k \cdot T_{flash} + m. \tag{4.33}$$

To improve the linear approach, all three sensors are combined into a single expression, as seen in the equation below.

$$T_{amb} = c_0 \cdot T_{usb} + c_1 \cdot T_{batt} + c_2 \cdot T_{flash} + m \tag{4.34}$$

Similarly to *Model 1 Kirchhoff*, the parameter estimation for the linear models is done through the Python library SciPy, which is described in Section 3.3. All the parameters were initially set to one. The final parameters for the single-sensor linear equations can be seen in Table 4.7, and the parameter for the combined linear model can be seen in Table 4.8. Adding additional sensors to the equation, such as T_{soc} or T_{disp}, to Equation 4.34 did not improve the results, it instead made the results significantly worse in many of the test scenarios. Only the USB, battery and flash sensors were kept to give the linear options the best chance at performing.

Model linear	k	m
Battery linear	0.86	-0.82
USB linear	0.96	-0.88
Flash linear	0.84	-0.99

Table 4.7 Estimated parameters for each linear model on the form $Y = k \cdot X + m$.

Model Linear combined	c_0	c_1	c_2	m
Parameters	-1.36	1.52	0.80	-0.38

Table 4.8 Estimated parameters for combined linear equation.

5

Model evaluation and discussion

This chapter presents and discusses the result when evaluating the models in different scenarios, both in the verification scenario showing the general performance of the models and in other scenarios evaluating more specific aspects. The chapter visualises the accuracy of the models using plots and presents the evaluation metrics defined in Chapter 3 in tables, where the best results for each table are marked in bold font. It also discusses how the models compare to related works, their limitations, identifiability and the models' computational resource utilisation and ease of implementation.

5.1 General Performance

This section shows how different models perform when evaluated on the verification scenario (scenario 4), a test which aims to provide an idea of the general performance both at different temperatures and processor loads. The performance of the thermal circuit-based models can be seen in Figure 5.1, 5.2, 5.3, 5.4, 5.5 and 5.6. The same scenario was also used to evaluate the linear non-thermal circuit-based approaches, see Figure 5.7 and 5.8. In addition to the model estimation and ambient temperature, the plots contain the processor temperature to illustrate the device's internal heat generation. The light blue areas within the graphs illustrate a 98% rolling window confidence interval for the prediction error, the reason behind these intervals is to show how the consistency of prediction error changes during the different parts of the tests. The confidence was created by applying a rolling window of size 20 samples over the prediction error, for each window, the mean and standard deviation were computed, allowing the construction of a confidence interval under the assumption of the error being normally distributed using the formulas presented in Section 3.4. When plotting the confidence intervals, they were aligned to the time stamp at the centre of each window, leading to the first and last

10 time stamps not having a confidence interval. Therefore, they were padded with the first and last confidence interval values.

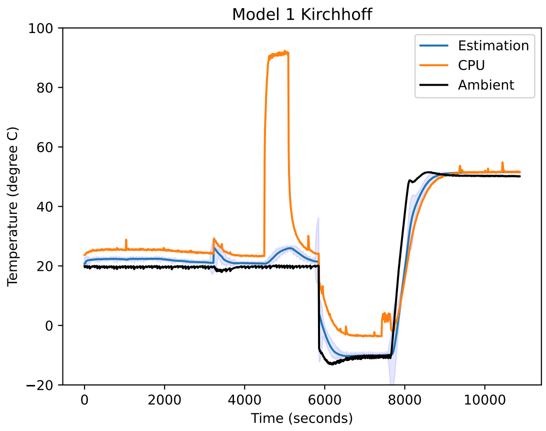


Figure 5.1 Model 1 Kirchhoff ambient temperature estimation for the verification scenario. The closer the estimation line is to the ambient line, the better the estimation is. The graph's light blue area is the prediction error's rolling window confidence interval.

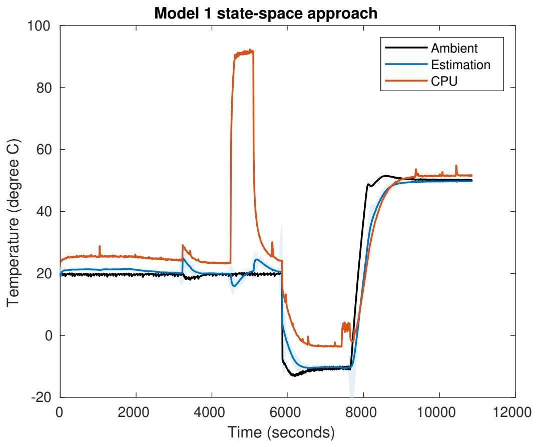


Figure 5.2 Same scenario as Figure 5.1, but with *Model 1 state-space* approach. It works reasonably well but still has performance issues when CPU load is applied, but not as much as in 5.1.

As mentioned in the introduction, the graphs above clearly show that estimating the ambient temperature is not as trivial as just making a linear approximation of

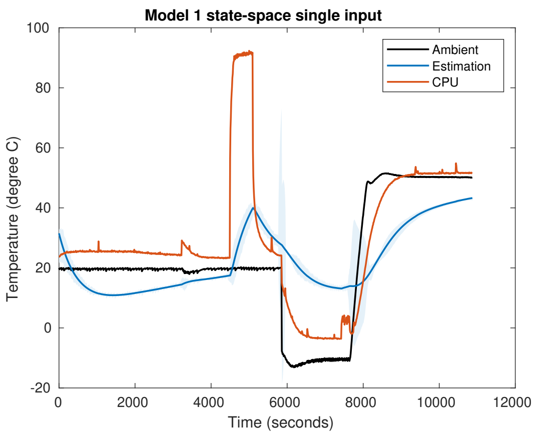


Figure 5.3 Same scenario as Figure 5.1, but for *Model 1 State-space single input*, which clearly do not work very well.

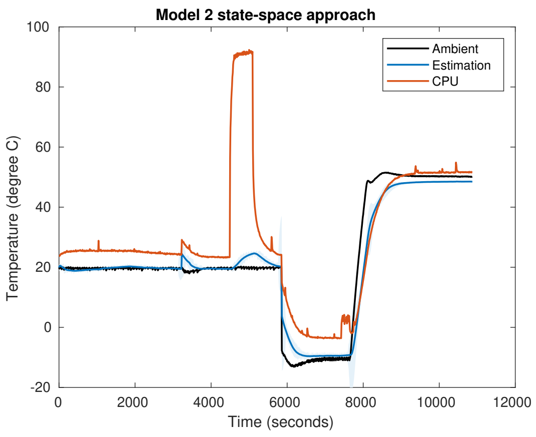


Figure 5.4 Same scenario as Figure 5.1, but for *Model 2 state-space* instead. It seems to perform similarly to its Model 1 counterpart but is still not able to fully compensate for the CPU load.

one single sensor value. Comparing Figure 5.7 with the other ones shows that the linear equations cannot handle CPU loads and swift changes in ambient temperatures. When observing the linear equation combining the three sensors in Figure 5.8, the performance seems on par with the best state-space model. To give a better understanding of how the performance differs between them, it is necessary to inspect the evaluation error metrics defined in Section 3.4, which can be seen in Table 5.1 and as well as the confidence interval for individual predictions at different percentages in Table 5.2. The confidence intervals shown within the tables differ

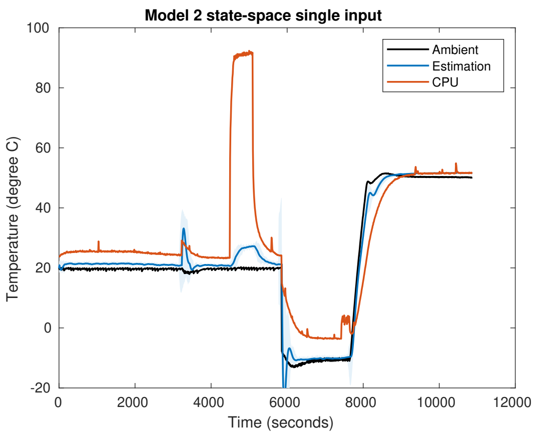


Figure 5.5 Same scenario as Figure 5.1, but for *Model 2 state-space single input* instead. This approach is a massive improvement compared to its Model 1 counterpart.

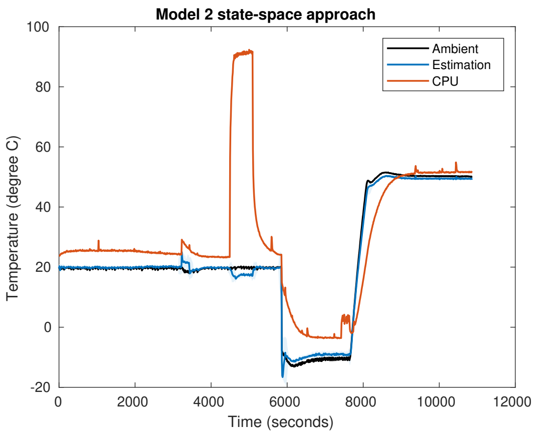


Figure 5.6 Same scenario as Figure 5.1, but for *Model 2 state-space with random C and D* instead. This approach for estimating ambient temperature will be shown to perform the best. But it still has some slight overcompensation when applying CPU load.

from the coloured area within the graphs since they are computed over the whole scenario, giving a sense of the models' overall error range. When looking through the tables, focus primarily on RMSE and MAE, the MBE can be deceiving since a test with very high positive and negative errors can yield a small MBE.

The error metrics and confidence intervals show decent results for all state-space models except *Model 1 state-space single input*. The model that performs best is *Model 2 state-space with random C & D*, which shows an average error below 1.5°C and accuracy of within ±1–3°C, which can be seen in Table 5.2, however, it

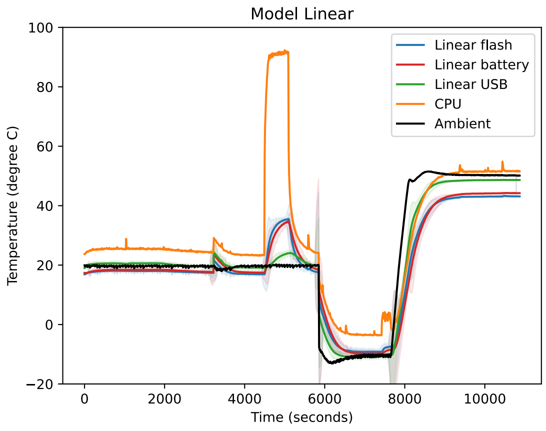


Figure 5.7 Same scenario as Figure 5.1, but for the linear equations $k \cdot T_x + m$. Compared to the previous plots, these equations do not work very well.

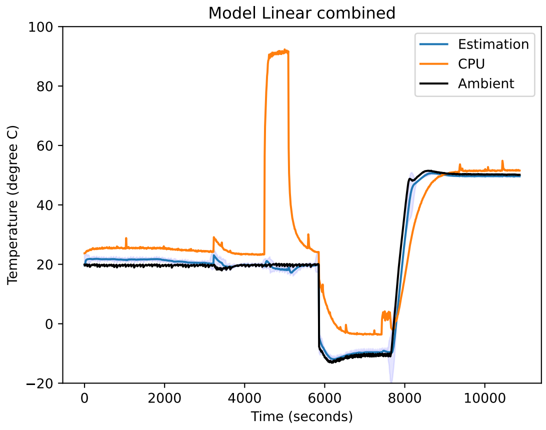


Figure 5.8 Same scenario as Figure 5.1, but for the linear polynomial combing multiple sensors. This is a clear improvement over the sensors in Figure 5.7 and it looks to be on par with *Model 2 State-space random C & D*.

is worth noting that in 90% of the cases, it is closer to ±1–2°C. This level of accuracy is good enough to approximate how warm or cold the environment around the mobile phone is. This approximation could be useful for optimising the sustained performance of the mobile phone, especially in more extreme weather, such as very warm or cold climates. As mentioned in the introduction, having an ambient-aware thermal mitigation policy would allow these optimisations, and the shown result

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	4.55	3.00	-1.01
Model 1 state-space	4.43	2.42	0.27
Model 1 state-space single input	16.07	13.02	**0.00**
Model 2 state-space	4.64	2.58	0.69
Model 2 state-space single input	3.21	2.19	-1.16
Model 2 state-space random C & D	**1.30**	**0.92**	0.11
Linear battery	8.61	5.52	2.37
Linear USB	4.62	2.46	0.79
Linear Flash	8.16	5.64	2.56
Linear combined	2.27	1.44	-0.07

Table 5.1 Error metrics defined in Section 3.4 for both the non-thermal circuit-based and thermal circuit-based approaches for the verification scenario 4. The best result for each error metric is marked with bold font.

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-4.015, 1.978)	(-8.326, 6.289)	(-11.354, 9.317)
Model 1 state-space	(-2.72, 3.25)	(-7.02, 7.55)	(-10.04, 10.57)
Model 1 state-space single input	(-10.84, 10.84)	(-26.44, 26.44)	(-37.40, 37.40)
Model 2 state-space	(-2.41, 3.78)	(-6.86, 8.24)	(-9.99, 11.36)
Model 2 state-space single input	(-3.18, 0.86)	(-6.08, 3.76)	(-8.13, 5.81)
Model 2 state-space random C & D	**(-0.76, 0.99)**	**(-2.02, 2.25)**	**(-2.90, 3.13)**
Linear battery	(-3.20, 7.96)	(-11.24, 16.00)	(-16.88, 21.64)
Linear USB	(-2.28, 3.86)	(-6.70, 8.28)	(-9.80, 11.38)
Linear flash	(-2.66, 7.79)	(-10.18, 15.32)	(-15.46, 20.60)
Linear combined	(-1.61, 1.46)	(-3.82, 3.67)	(-5.37, 5.22)

Table 5.2 Confidence interval for each model over the verification scenario 4 at different percentages. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

would allow for different thermal mitigation policies with a granularity of a few degrees Celsius.

As concluded earlier, using a single sensor is not enough to achieve good and reliable ambient temperature estimations, however, *Linear combined* is a massive improvement compared to the other linear approaches, and even it even slightly outperforms the second-best state-space model *Model 2 state-space single input*. The next section will go into more detail, investigating how they differ in performance in more specific situations.

5.2 Evaluation on specific scenarios

This section contains more specific scenarios and use-case-based scenarios, such as camera and YouTube, highlighting how the models perform in those specific cases compared to the more general case presented in the previous section.

Idle test

The first test scenario of interest is scenario 1, i.e. which tests the models when the mobile phone is idling at a static ambient temperature of 20°C. Since it is the simplest of the defined scenarios, it provides a best-case scenario for the models. Table 5.3 and 5.4 show the error metrics and confidence intervals. Worth noting is the linear models perform relatively well compared to the state-space ones in this scenario. All models except the flash and battery-based ones are within 1°C in RMSE.

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	0.80	0.76	-0.76
Model 1 state-space	0.30	0.25	0.17
Model 1 state-space single input	**0.19**	**0.16**	**0.00**
Model 2 state-space	0.20	**0.16**	0.01
Model 2 state-space single input	0.76	0.70	-0.70
Model 2 state-space random C & D	0.30	0.26	0.21
Linear battery	2.60	2.59	2.59
Linear USB	0.96	0.93	0.93
Linear flash	3.10	3.09	3.09
Linear combined	0.35	0.30	0.25

Table 5.3 Error metrics for test when idling in 20°C , see scenario 1. The best result for each error metric is marked with bold font.

Applying high CPU load

By applying a high CPU load, we can get some insight into how the models perform when exposed to internal heat generation, this is done by evaluating Scenario 2.1, i.e. when applying 100 % CPU load for 10 minutes at a static ambient temperature of 20°C. The error metrics and confidence intervals are shown in Table 5.5 and

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-0.93, -0.60)	(-1.17, -0.36)	(-1.33, -0.19)
Model 1 state-space	(0.00, 0.34)	(-0.24, 0.58)	(-0.41, 0.75)
Model 1 state-space single input	**(-0.13, 0.13)**	**(-0.32, 0.32)**	**(-0.45, 0.45)**
Model 2 state-space	(-0.13, 0.14)	(-0.32, 0.34)	(-0.45, 0.47)
Model 2 state-space single input	(-0.90, -0.49)	(-1.19, -0.20)	(-1.40, 0.00)
Model 2 state-space random C & D	(0.06, 0.36)	(-0.15, 0.57)	(-0.30, 0.72)
Linear battery	(2.43, 2.75)	(2.19, 2.98)	(2.03, 3.15)
Linear USB	(0.76, 1.10)	(0.53, 1.33)	(0.36, 1.50)
Linear flash	(2.93, 3.25)	(2.70, 3.48)	(2.54, 3.64)
Linear combined	(0.08, 0.41)	(-0.15, 0.65)	(-0.32, 0.82)

Table 5.4 Confidence interval for test when idling in 20°C , see scenario 1. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

5.6. Similar to the verification scenario, we can see a similar performance between *Linear combined* and *Model 2 state-space random C & D*.

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	3.53	3.02	-3.02
Model 1 state-space	2.26	1.68	-0.98
Model 1 state-space single input	4.72	3.72	**0.00**
Model 2 state-space	1.92	1.51	-0.52
Model 2 state-space single input	3.60	2.82	-2.80
Model 2 state-space random C & D	1.36	0.85	0.43
Linear battery	6.18	4.28	-2.80
Linear USB	2.16	1.58	-1.25
Linear flash	6.69	4.58	-2.32
Linear combined	**1.19**	**0.82**	0.07

Table 5.5 Error metrics for test in 20°C ambient temperature with CPU load on all cores for 10 minutes, see Scenario 2.1. The best result for each error metric is marked with bold font.

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-4.263, -1.794)	(-6.039, -0.018)	(-7.286, 1.229)
Model 1 state-space	(-2.36, 0.40)	(-4.34, 2.38)	(-5.73, 3.77)
Model 1 state-space single input	(-3.19, 3.19)	(-7.77, 7.77)	(-10.99, 10.99)
Model 2 state-space	(-1.77, 0.73)	(-3.57, 2.53)	(-4.83, 3.79)
Model 2 state-space single input	(-4.33, -1.27)	(-6.52, 0.92)	(-8.07, 2.47)
Model 2 state-space random C & D	**(-0.44, 1.30)**	**(-1.69, 2.55)**	(-2.57, 3.43)
Linear battery	(-6.51, 0.92)	(-11.87, 6.27)	(-15.62, 10.02)
Linear USB	(-2.44, -0.06)	(-4.15, 1.64)	(-5.35, 2.84)
Linear flash	(-6.55, 1.91)	(-12.64, 8.00)	(-16.91, 12.27)
Linear combined	(-0.72, 0.88)	(-1.88, 2.03)	**(-2.69, 2.85)**

Table 5.6 Confidence interval for test in 20°C ambient temperature with CPU load on all cores for 10 minutes, see scenario 2.1. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

Performance during temperature changes

This scenario tests heavy ambient temperature changes from -10°C to 50°C , showing how well the models will follow the ambient temperature changes. The error metrics and confidence intervals are shown in Table 5.7 and 5.8. In this test scenario, we see how *Linear combined* performs significantly worse than all state-space models except *Model 1 state-space single input.*

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	7.969	5.773	-0.826
Model 1 state-space	6.11	3.42	2.08
Model 1 state-space single input	13.43	11.18	**0.00**
Model 2 state-space	6.13	4.26	2.20
Model 2 state-space single input	2.46	1.75	0.03
Model 2 state-space random C & D	**1.38**	**1.24**	0.19
Linear battery	7.79	6.67	4.07
Linear USB	8.24	5.37	1.06
Linear flash	10.08	8.42	4.14
Linear combined	12.22	8.55	-0.73

Table 5.7 Error metrics for test when ambient temperature changes from -10°C to 50°C, see scenario 3. The best result for each error metric is marked with bold font.

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-6.172, 4.521)	(-13.864, 12.212)	(-19.266, 17.614)
Model 1 state-space	(-1.80, 5.96)	(-7.38, 11.54)	(-11.30, 15.46)
Model 1 state-space single input	(-9.07, 9.07)	(-22.12, 22.12)	(-31.28, 31.28)
Model 2 state-space	(-1.67, 6.06)	(-7.23, 11.62)	(-11.13, 15.52)
Model 2 state-space single input	(-1.63, 1.69)	(-4.03, 4.08)	(-5.71, 5.76)
Model 2 state-space random C & D	**(-0.73, 1.11)**	**(-2.05, 2.44)**	**(-2.98, 3.37)**
Linear battery	(-0.41, 8.55)	(-6.86, 15.00)	(-11.39, 19.53)
Linear USB	(-4.44, 6.57)	(-12.38, 14.51)	(-17.95, 20.08)
Linear flash	(-2.06, 10.34)	(-10.98, 19.27)	(-17.25, 25.54)
Linear combined	(-8.96, 7.49)	(-20.80, 19.32)	(-29.11, 27.64)

Table 5.8 Confidence intervals for test when ambient temperature changes from -10°C to 50°C, see scenario 3. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

Camera tests

Tests were conducted when filming at different resolutions to see how the models perform when the camera is active. Table 5.9 and 5.10 show the error metrics and confidence intervals. In this scenario, *Linear combined* outperforms the state-space in the two first metrics.

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	4.675	4.452	-4.452
Model 1 state-space	2.34	1.91	-1.78
Model 1 state-space single input	2.97	2.39	**0.00**
Model 2 state-space	2.16	1.74	-0.76
Model 2 state-space single input	5.90	5.49	-5.45
Model 2 state-space random C & D	2.88	2.60	-2.51
Linear battery	7.44	6.12	-6.07
Linear USB	2.95	2.61	-2.61
Linear flash	8.72	7.16	-6.93
Linear combined	**1.61**	**1.43**	-1.32

Table 5.9 Error metrics for tests when filming at different resolutions, scenario 2.3. The best result for each error metric is marked with bold font.

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-5.415, -3.489)	(-6.800, -2.104)	(-7.773, -1.131)
Model 1 state-space	(-2.80, -0.76)	(-4.27, 0.70)	(-5.30, 1.73)
Model 1 state-space single input	(-2.00, 2.00)	(-4.89, 4.89)	(-6.91, 6.91)
Model 2 state-space	(**-2.12, 0.61**)	(-4.08, 2.57)	(-5.46, 3.95)
Model 2 state-space single input	(-6.98, -3.92)	(-9.18, -1.72)	(-10.72, -0.18)
Model 2 state-space random C & D	(-3.46, -1.56)	(-4.83, -0.19)	(-5.79, 0.78)
Linear battery	(-8.97, -3.18)	(-13.14, 0.98)	(-16.06, 3.91)
Linear USB	(-3.53, -1.68)	(-4.87, -0.35)	(-5.80, 0.58)
Linear flash	(-10.50, -3.37)	(-15.63, 1.76)	(-19.23, 5.36)
Linear combined	(-1.94, -0.70)	(**-2.83, 0.18**)	(**-3.46, 0.81**)

Table 5.10 Confidence intervals for tests when filming at different resolutions, scenario 2.3. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

YouTube over 4G

To see the modem's impact on the models and the varying processor load it introduces (see Figure 5.9), a test was conducted when watching YouTube over 4G at different resolutions. The error metrics and confidence intervals are shown in Table 5.11 and 5.12, here it is worth noting how *Model 1 state-space single input* state-space variations in difference to other test outperform the rest.

Model	**RMSE**	**MAE**	**MBE**
Model 1 Kirchhoff	3.359	3.339	-3.339
Model 1 state-space	1.52	1.46	-1.45
Model 1 state-space single input	**0.24**	**0.19**	**0.00**
Model 2 state-space	0.35	0.28	-0.04
Model 2 state-space single input	4.22	4.09	-4.01
Model 2 state-space random C & D	2.68	2.60	-2.50
Linear battery	1.89	1.81	-1.81
Linear USB	1.58	1.54	-1.54
Linear flash	1.85	1.77	-1.70
Linear combined	1.41	1.37	-1.36

Table 5.11 Error metrics for tests when watching YouTube at different resolutions over 4G, see scenario 2.4. The best result for each error metric is marked with bold font.

Model	**50%**	**90%**	**98%**
Model 1 Kirchhoff	(-3.590, -3.087)	(-3.952, -2.725)	(-4.206, -2.471)
Model 1 state-space	(-1.77, -1.12)	(-2.23, -0.66)	(-2.56, -0.33)
Model 1 state-space single input	**(-0.17, 0.17)**	**(-0.40, 0.40)**	**(-0.57, 0.57)**
Model 2 state-space	(-0.28, 0.19)	(-0.61, 0.53)	(-0.85, 0.77)
Model 2 state-space single input	(-4.90, -3.12)	(-6.19, -1.84)	(-7.09, -0.93)
Model 2 state-space random C & D	(-3.15, -1.85)	(-4.09, -0.91)	(-4.75, -0.26)
Linear battery	(-2.18, -1.43)	(-2.72, -0.89)	(-3.10, -0.51)
Linear USB	(-1.79, -1.30)	(-2.14, -0.95)	(-2.39, -0.70)
Linear flash	(-2.18, -1.22)	(-2.87, -0.54)	(-3.36, -0.05)
Linear combined	(-1.62, -1.09)	(-2.00, -0.71)	(-2.27, -0.45)

Table 5.12 Confidence intervals for tests when watching YouTube at different resolutions over 4G, see scenario 2.4. The best result for each confident interval is marked with bold font. The confidence interval is computed over the whole test scenario as described in Section 3.4.

Looking at the metrics shown above, the performance of the models varies between the different scenarios, but *Model 2 state-space with random C & D* performs among the best in most scenarios, making it the overall best model. The other state-space model based on the second thermal circuit also performs well. *Linear combined* often performs well, except for the test with larger temperature changes. However, overall, the results show that the thermal circuits approach leads to better results than linear, especially compared to the cases with linear equations that only use a single sensor.

When looking at the results, two question arises, why is *Model 2 state-space with random C & D* better than the other state-space variations more closely based on the thermal circuits? And why do the results from the YouTube and camera scenarios differ from the rest? Addressing the first question, this indicates the thermal circuits might be too simplified to fully address the thermal behaviour of the mobile phone, having more free parameters in *Model 2 state-space with random C & D* allows the model to better compensate for the inaccuracies of the model.

When looking at the YouTube scenario in Table 5.12, *Model 1 state-space single input* performs the best. One difference between the YouTube test is when looking at the CPU load in Figure 5.9, it is more varied than the other test cases while the other sensors stay stable. Our theory is that the additional input in the other state-space models, compared to the single input variation in this scenario, hinders the models from reacting to the load introduced by the CPU. For the camera tests, the performance varies quite a lot compared between the different models, making it difficult to explain the results, but it may be similar to the YouTube test caused by a more varied CPU temperature compared to the other tests.

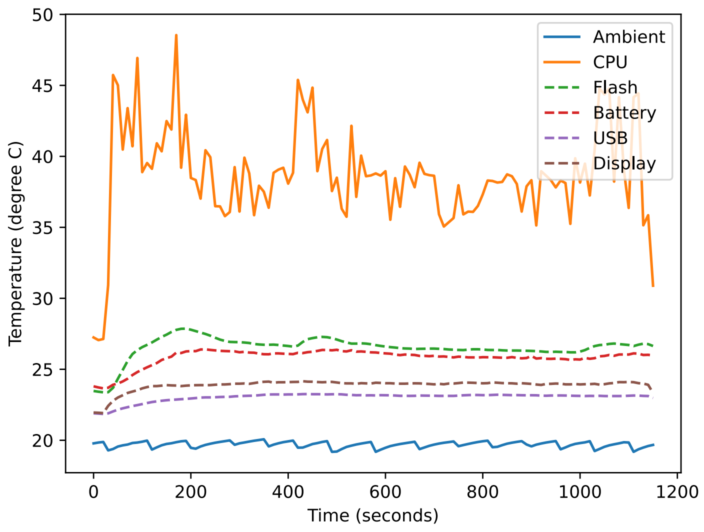


Figure 5.9 The sensor values during the YouTube test.

The worse test scenarios for the state-space models are the camera and YouTube ones, indicating further model development should consider the camera and modem temperature in their design.

As discussed in the modelling chapter, Model 2 outperforms Model 1 in most cases due to replacing the SoC with the display sensor since it should be more representative of how the heat from the SoC spreads within the mobile phone. However, when comparing the different state-space models, it can sometimes be seen how *Model 1 state-space* sometimes slightly outperforms *Model 2 state-space*. This indicates the hypothesis about adding an additional node might be wrong. However, adding the display node leads to a massive improvement for the single-input variations. The test clearly shows how certain scenarios fit one model better than another, but in most cases, when this happens, the *Model 2 state-space* is close to its Model 1 equivalent in performance. So even when the models show no benefit of adding the display node, the difference is so small that it does not harm the overall performance.

The results show the importance of choosing the best temperature probes within the phone since it gives a better chance for the model to perform better. Another point worth noting is that when choosing temperature sensors, it is better to choose sensors less affected by heat from heat sources than have additional worse-performing sensors. Since most of the sensors on the mainboard are heavily affected by the SoC and have similar temperature values to those on the SoC, they do not give the model additional information. This is also the reason why USB, battery and flash were chosen.

5.3 Performance compared to related works

As part of the introduction in Section 1.1, to give this thesis some context, a few relevant related works were described. To further elaborate on related works, this subsection will compare their results to the ones found within this thesis.

Looking at the battery-based approach by He et al., they achieved an average error of 1.25°C. The results in this thesis are comparable with an MAE of 0.90°C in the most general test, the verification scenario test. However, the approach in this thesis outperforms their results when including CPU loads in the test scenarios, where their error absolute error is closer to 2–3°C [He et al., 2020]. In contrast, in this thesis, the errors are below 1.5°C in all tests involving a CPU load.

Looking at the fixed point algorithm by Li et al., they show an MAE of 0.7°C . In most scenarios, the results in this thesis are similar or a bit worse, closer to 0.9°C. However, this thesis presents better results in YouTube or idle scenarios. It is worth noting that it is hard to make a fair comparison since the system in their report runs significantly cooler, while their CPU is fully loaded, the temperature increases by 7°C while the SoC in this thesis test system reaches over 90°C with a full load no mater which ambient temperature is set [Li et al., 2020].

5.4 Limitations of the models

The main limitation of the models is when the CPU is heavily loaded and gets extremely warm. Looking at the plots for the verification scenario, it can be seen how both *Model 2 state-space single input* and *Model 2 state-space* under-compensates for this additional heat (See Figure 5.4 and 5.5). While the overall state-space model *Model 2 state-space random C & D* handle it better but still slightly overcompensate for it (see Figure 5.6).

During extreme ambient temperature changes, it is almost impossible to have 100% accurate estimation since it takes time for the temperature material of the mobile phone to change and, thereby, the temperature sensors to register a change. This delay in temperature estimations can be observed for all models, however, for *Model 2 state-space* and *Model 2 state-space random C & D*, it is barely noticeable. Another limitation of the models is the amount of data used when training them. In this case, the more data from different scenarios, the better the parameter estimation will be. In this case, the heating chamber is also a limitation since it has difficulty keeping a stable temperature, resulting in slightly unstable lines for the ambient temperature within the plots, which can differ up to one degree Celsius.

All the temperature sensors also have their limitation in accuracy. In this case, the accuracy of the internal sensors is unknown, but they can limit how well the ambient temperature can be estimated. The available sensors of a system are also a limitation. In some cases, there might not be any good-performing sensors located far away from heat sources such as the CPU. However, in this thesis, that has not been a problem.

5.5 Identifiability analysis

To analyse the identifiability of the models, the definition in Section 2.5 must be considered. If another set of parameters providing the same output given the same set of input signals can be found, then the models clearly are unidentifiable. This was tested by generating a new data set from the estimated models and then re-estimating the model parameters using the same Matlab method described in Section 2.5 with the new data set. If the procedure found a different parameter set than the original, yielding a close to 100 % fit, then the model is unidentifiable. The new data set was generated by inputting the input data from the original training set to the models and combining their output with the input forming a new data set.

Performing the above-described procedure shows the non-single-input state-space models to be unidentifiable when using the parameter configurations in Chapter 4, achieving a fit of $98-99\%$ with another parameter set. However, when changing the models to have the capacitances fixed, the procedure could not find another parameter set achieving the same fit, which implies the models are identifiable. The single input variations could not be shown to be unidentifiable, which is reasonable since one of their capacitances was already fixed. Having to constrain the capacitance to achieve identifiability makes sense from a physical perspective as well, since the models are based purely on the input and output temperatures from a thermal system, it is difficult to know what kind of heat flow caused a temperature change. Comparing the formulas in Section 2.2, the same thermal flow can be caused by a system with high capacitance and low resistance or by a system with low capacitance and high resistance.

Making the capacitances fixed and thereby achieving identifiability would allow the estimator to find a unique optimal parameter set independent from the initial parameter guess and can potentially improve the estimation. When re-estimating and evaluating the models with the capacitance fixed, the performance difference was negligible, thereby, the original results with the free capacitances were kept. For the purpose of finding good ambient temperature estimations, the unidentifiability caused by the non-fixed capacitance does not seem to cause any practical problem, the estimator converges with the provided initial guesses anyway. However, if the models would be used for analysing the thermal properties of the phone, it would be important to consider since it is desirable to find a unique parameter set that is as physically accurate as possible. Because both the resistances and capacities can not be found in this approach, this would require capacitance parameters to be measured or estimated using some other methods.

5.6 Computational efficiency

When implementing a model on a computing device, such as a mobile phone, it is important to consider the computational cost, i.e. the processing power required to run the models. For the models presented in this thesis, the most resource-heavy part is the computational load on the CPU when estimating the parameter. However, this must only be done once for a specific mobile phone model. Once trained, all the models would theoretically be quite resource-efficient for on-device implementation. The most resource-heavy of the models, which is state-space-based ones, would require the CPU to perform four matrix multiplications with temperature sensor values for every sample, whereas the largest matrix has a size of 4x4, which is insignificant for a modern mobile phone CPU when sampling with a frequency of $1 - 10$ Hz. The CPU resource requirement is even less for the linear approaches, which only evaluate a simple polynomial. The low resource requirement allows the models to be implemented on the mobile phone without affecting its performance and heat generation in any significant way, making it a practically useful method for ambient estimation.

6 Conclusion

The results show that a mobile phone's ambient temperature can be estimated using existing internal temperature sensors with an average error below 1.5°C and accuracy of within ±1–3°C depending on the scenario. This was achieved by creating thermal circuits based on the internal temperature sensors and hardware layout. It is important to consider the sensors that are less affected by heat from SoC, compared to the ones on the mainboard close to the SoC, which give similar temperature values as SoC and therefore do not give the model any additional information. Numerical methods and grey-box estimation in Python and Matlab were used to calculate the parameters of the different thermal circuit-based models, where the state-space-based approaches yielded the best results.

Estimating ambient temperature can also be done using linear static (non-dynamic) models, using the different sensors as variables. The results show the polynomial combining the thermal sensors in the USB port, battery and flash performs well in many test scenarios, as long as the temperature does not change too quickly, but overall, the best state-space models still are slightly better.

Once trained, all the models would theoretically be very computationally efficient, for instance, the most computation-heavy of the model, which is state space approaches, consists of four matrix multiplications, whereas the largest one is a 4x4 matrix, which has an insignificant computation cost on a modern mobile phone.

The thermal models should theoretically be generalisable and be able to be used on other mobile phone models with only the need for new parameter estimations. The thermal models' principles should theoretically also be able to be used on almost all embedded devices with internal temperature sensors.

6.1 Future work

All the tests made in this thesis have been done in a laboratory environment. It would be useful to do field tests with an on-device implementation consisting of real use cases, allowing evaluation of the models' true accuracy and showing if the models are feasible to deploy on real devices.

As mentioned in the limitations section, see Section 1.3, the thesis has chosen not to investigate how much impact a phone case or screen cover would impact the models' estimation. This could be worth investigating since encasing the device in isolating materials such as glass and plastics may affect the thermal properties of the device, resulting in the models' parameters needing to be re-estimated. The radiant heat from the sun was also not included in the thesis, but it would be of interest to investigate if it has a significant impact on the accuracy of the estimation.

Another thing worth investigating is how much the human body would impact the estimations since the human body would act both as a heat source and a heat sink. This has been done in similar research papers such as in Chau [Chau, 2019], but this paper needed retraining of the model to compensate for the human heat generation. The models in this thesis might need to be revised where a node for human heat generation might need to be added. On the other hand, how that heat generation shall be approximated is far from obvious.

To evaluate the generality of the models, it would also be interesting to test the same pre-trained models on different mobile phone models and see if there are any significant accuracy losses. It might show that the model parameters need to be re-estimated for each mobile phone model, or in some cases, not work since the sensors might not exist or perform too poorly. These tests could also be expanded to test a completely different platform, such as a TV, video camera, or computer. The models might need to be revised to fit the layout of the temperature sensors of the specific device, but theoretically, similar models could be built for those devices using the same principles. As part of these efforts to maximise generality, it would be interesting to investigate the feasibility of an on-device parameter estimation scheme and an algorithm to systematically design and evaluate different circuit permutations.

Bibliography

Blom, G., J. Enger, G. Englund, J. Grandell, and L. Holst (2017). *Sannolikhet och statisikteori med tillämpningar*. 7th ed. Studentlitteratur AB, Lund. ISBN: 978-91-44-12356-1.

Bohlin, T. (2006). *Practical Grey-box Process Identification*. en. 1st ed. Advances in Industrial Control. Springer London. ISBN: 978-1-84628-402-1. DOI: 10.1007/1-84628-403-1. URL: http://link.springer.com/10.1007/1-84628-403-1 (visited on 2023-05-09).

Bonnans, J. F., J. C. Gilbert, C. Lemarechal, and C. A. Sagastizabal (2006). *Numerical Optimization*. en. 2nd edition. Universitext. Springer, Berlin, Heidelberg. ISBN: 978-3-540-35445-1. URL: https://link.springer.com/book/10.1007/978-3-540-35447-5 (visited on 2023-02-24).

Borgnakke, C. and R. E. Sonntag (2013). *Fundamentals of Thermodynamics*. en. 8 [edition]. Wiley, Hoboken, NJ. ISBN: 978-1-118-13199-2. URL: http://uowa.edu.iq/filestorage/file_1551543405.pdf (visited on 2023-10-02).

Chau, N. H. (2019). "Estimation of air temperature using smartphones in different contexts". *Journal of Information and Telecommunication* **3**:4. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/24751839.2019.1634869, pp. 494–507. ISSN: 2475-1839. DOI: 10.1080/24751839.2019.1634869. URL: https://doi.org/10.1080/24751839.2019.1634869 (visited on 2023-01-19).

Daleh, M., M. A. Daleh, and G. Verghouse (2011). *Lectures on Dynamic Systems and Control*. Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. URL: https://ocw.mit.edu/courses/6-241j-dynamic-systems-and-control-spring-2011/pages/readings/ (visited on 2023-05-17).

Google (2022). *Thermal Mitigation*. en. URL: https://source.android.com/docs/core/power/thermal-mitigation (visited on 2023-04-19).

Guillaume, J. H. A., J. D. Jakeman, S. Marsili-Libelli, M. Asher, P. Brunner, B. Croke, M. C. Hill, A. J. Jakeman, K. J. Keesman, S. Razavi, and J. D. Stigter (2019). “Introductory overview of identifiability analysis: A guide to evaluating whether you have the right type of data for your modeling purpose”. *Environmental Modelling & Software* **119**, pp. 418–432. ISSN: 1364-8152. DOI: `https://doi.org/10.1016/j.envsoft.2019.07.007`. URL: `https://www.sciencedirect.com/science/article/pii/S1364815218307278`.

He, L., Y. Lee, and K. G. Shin (2020). “Mobile Device Batteries as Thermometers”. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* **4**:1, 12:1–12:21. DOI: `10.1145/3381015`. URL: `https://doi.org/10.1145/3381015` (visited on 2023-01-20).

Incropera, F. P., D. P. DeWitt, T. L. Bergman, and A. S. Lavine (2007). *Fundamentals of heat and mass transfer*. en. 6. ed. Wiley, Hoboken, NJ. ISBN: 9780471457282.

Ishii, M. and Y. Nakashima (2017). “Development of algorithms for real-time estimation of smartphone surface temperature using embedded processor”. In: *2017 16th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, pp. 1088–1093. DOI: `10.1109/ITHERM.2017.7992609`.

Jha, A., M. S. Abirami, and V. Kumar (2023). *Predictive Model for Depression and Anxiety Using Machine Learning Algorithms*. Vol. 1719. Communications in Computer and Information Science. 1719. Pages: 147. Springer Nature Switzerland, Cham. ISBN: 978-3-031-27621-7. DOI: `10.1007/978-3-031-27622-4_11`.

Li, B., H. Liu, and R. Wang (2020). “Real-time Ambient Temperature Estimation of Mobile Electronic Devices”. In: *2020 26th International Workshop on Thermal Investigations of ICs and Systems (THERMINIC)*. ISSN: 2474-1523, pp. 249–254. DOI: `10.1109/THERMINIC49743.2020.9420537`.

Ljung, L. and T. Glad (2004). *Modellbygge och simulering*. swe. 2., [utvidgade och modifierade] uppl. Studentlitteratur, Lund. ISBN: 9789144024431.

Mathworks (2023a). *Grey-box model estimations*. URL: `https://www.mathworks.com/help/ident/grey-box-model-estimation.html`.

Mathworks (2023b). *Greyest - estimate ode parameters of linear grey-box model*. URL: `https://www.mathworks.com/help/ident/ref/greyest.html` (visited on 2023-03-16).

Mathworks (2023c). *Idgrey - linear ode (grey-box model) with identifiable parameters*. URL: `https://www.mathworks.com/help/ident/ref/idgrey.html` (visited on 2023-03-16).

Mathworks (2023d). *Normal distribution*. URL: `https://www.mathworks.com/help/stats/normal-distribution.html` (visited on 2023-06-01).

Parab, V. A. and N. Mangaonkar (2018). “Android Kernel with EAS for Better Power Management”. In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1–5. DOI: `10.1109/ICCONS.2018.8663075`.

SciPy Authors (2023a). *Scipy.optimize.minimize — SciPy v1.10.1 Manual.* URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html` (visited on 2023-03-16).

SciPy Authors (2023b). *Scipy.stats.norm — SciPy v1.10.1 Manual.* URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html` (visited on 2023-06-01).

Sidebotham, G. (2015). *Heat Transfer Modeling: An Inductive Approach.* 1st ed. 2015. Springer International Publishing, Cham. ISBN: 9783319145143.

Sony Electronics (2022). *Sony Compact design with 4K HDR 120fps video recording | Xperia 5 IV.* en. URL: `https://electronics.sony.com/mobile/smartphone/all/p/xqcq62b-gc` (visited on 2023-03-14).

Lund University **Department of Automatic Control** **Box 118** **SE-221 00 Lund Sweden**	*Document name* MASTER'S THESIS *Date of issue* June 2023 *Document Number* TFRT-6206
Author(s) Isak Evaldsson Henrik Paulcén	*Supervisor* Rickard Möller, Sony, Sweden Jimmy Olsson, Sony, Sweden Karl-Erik Årzén, Dept. of Automatic Control, Lund University, Sweden Bo Bernhardsson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle

Estimating Ambient Temperature using Internal Sensors and Thermal Modelling in Mobile Phones

Abstract

This thesis studies the possibility of estimating the ambient temperature around a mobile phone using its internal temperature sensors.

The thesis aims to develop and evaluate ambient temperature models that use internal temperature sensors within the device and, from a manufacturer's perspective, investigate if the models for ambient temperature can be enhanced by tailoring them to the internals of specific devices. The chosen approach was to use thermal circuits and, based on them, derive state-space models. The parameters of these models were then estimated using collected data. A linear polynomial approach was also evaluated but gave worse estimations than the state-space approaches. The best-performing model, one of the state-space models, has an estimation accuracy of within ±1–3°C and an average error of below 1.5°C when evaluated on a broad collection of testing scenarios.

Once parameter estimations have been performed, all the models have a low processor resource utilisation, which is ideal for an on-device implementation. The thermal models should theoretically be generalisable and could be used on other mobile phone models with similar internal layouts with only the need for new parameter estimations. The thermal models' principles could theoretically also be able to be used on almost all embedded devices with similar internal temperature sensors.

Keywords		
Classification system and/or index terms (if any)		
Supplementary bibliographical information		
ISSN and key title 0280-5316		*ISBN*
Language English	*Number of pages* 1-58	*Recipient's notes*
Security classification		

http://www.control.lth.se/publications/