ME 121: Thermal Control of the Fish Tank

February 26, 2020

Gerald Recktenwald

gerry@pdx.edu

1 Overview

This document has three main purposes. First, we give an overview of how an energy balance can be used to relate the temperature changes in a volume of water to a heat input via an electrical resistance heater. The energy balance allows us to predict the requirements of a heater used to maintain temperature in the fish tank.

Second, we describe experiments to characterize the thermal performance of fish tank. Data from these experiments is used to set parameters in the control algorithm.

Finally, we provide a conceptual overview of the control algorithm, along with an outline of the Arduino code that controls both temperature and salinity of the fish tank.

1.1 Learning Objectives

After reading and studying these notes you should be able to

1. Describe and give a physical interpretation to each term in the thermal energy balance equation for the water in the fish tank.
2. Use the energy balance equation to predict the rate of temperature rise (°C/s) when a known voltage is applied to the heater.
3. Describe a procedure for estimating the smallest realistic value of the deadband for thermal control of the fish tank.
4. Describe experiments for measuring the characteristic constant, K of the fish tank.
5. Explain the physical significance of K.
6. Describe a procedure for using K and the temperature error to determine the duration of heat input.
7. Explain the role of salinity control deadtime in setting an upper bound on the duration of heat input.
8. Explain the pseudocode version of a thermal control algorithm for the fish tank.

2 Energy Storage in a Fluid

Consider a tank of water with an immersed electric resistance heater as depicted in Figure 1. Applying the First Law of Thermodynamics (the energy conservation principle) gives

$$\Delta E = E_{\text{in}} - E_{\text{out}} \tag{1}$$

where ΔE is the change in energy content of water during a specified time interval Δt, E_{in} is the energy added to the water during Δt, and E_{out} is the energy lost to the ambient during Δt.

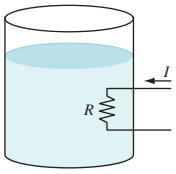


Figure 1: Electric resistance heater in a tank of water.

Since the tank is open to the atmosphere, the additional of energy will only cause an increase in temperature, not pressure. The change in energy of the tank is

$$\Delta E = mc_p \Delta T \tag{2}$$

where m is the mass, c_p is the specific heat at constant pressure, and ΔT is the change in temperature.

Thermal energy is stored in the water (mostly), but also in the structural and mechanical components – the tank walls, the pump body and impeller, the tubing, the body of the salinity sensor, etc. Therefore, the mass of material involved in energy storage is greater than the mass of the water. Furthermore, the temperature of all of the structural and mechanical components is not uniform, so defining an appropriate ΔT is problematic. Despite the difficulty in applying Equation (2) with precision, that equation is very helpful in our conceptual understanding of the energy balance in the system.

For our purposes, we assume that the energy storage in the structural components is small enough to neglect. If that assumption is true, the energy in the fish tank system is stored only in the circulating water. We can then use the volume of fluid in the system to compute the mass of fluid in the system. The volume $\mathcal{V}$ and mass m are related by the density, ρ

$$m = \rho \mathcal{V}. \tag{3}$$

2.1 Energy Input from the Heater

For the electric resistance heater shown in Figure 1, the energy input during Δt is

$$E_{\text{in}} = VI\Delta t \tag{4}$$

where V is the voltage across the resistor, I is the current through the resistor, and VI is the *rate* at which electrical power is dissipated.

Remember that the power dissipated as electrical current, I, flows through a resistor, R, is

$$P = VI = I^2R = \frac{V^2}{R}$$

where V is the voltage drop across the resistor.

2.2 Energy Balance in Rate Form

It is more convenient to consider the rate form of Equation (1), which is obtained by dividing through by Δt

$$\frac{\Delta E}{\Delta t} = \dot{E}_{\text{in}} - \dot{E}_{\text{out}} \tag{5}$$

where $\dot{E}_{\text{in}}$ and $\dot{E}_{\text{out}}$ are the *rates* of heat addition and heat loss, respectively. The units of ΔE, E_{in} and E_{out} are joule (energy). The units of $\Delta E/\Delta t$, $\dot{E}_{\text{in}}$ and $\dot{E}_{\text{out}}$ are watt (power).

Dividing Equation (2) by Δt we relate the rate of increase of energy in the water to the rate of change of temperature of the water

$$\frac{\Delta E}{\Delta t} = mc_p \frac{\Delta T}{\Delta t} \tag{6}$$

Substituting Equation (6) into Equation (5) and dividing through by mc_p gives

$$\frac{\Delta T}{\Delta t} = \frac{VI}{mc_p} - \frac{\dot{E}_{\text{out}}}{mc_p}. \tag{7}$$

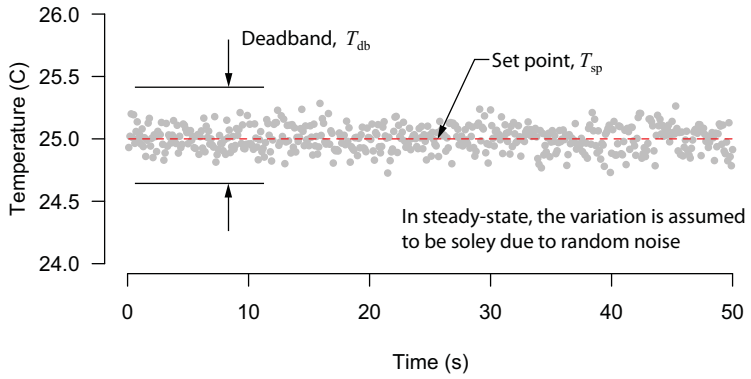


Figure 2: The deadband for thermal control should be larger than the magnitude of random temperature variation caused by noise.

Equation (7) is a model for how fast the temperature in the fish tank can rise when a constant electrical power input of VI is dissipated by the heater. Notice that the larger the tank (larger amount of water and hence larger m), the slower the temperature rises for a given power input. Also note that energy losses will result in a lower heating rate than what is predicted by the electrical power input alone.

A Note on Units: The units of c_p are usually written $\mathrm{J}/(\mathrm{kg\,K})$, but the temperature change is usually expressed in °C. For example, consider the increase in energy content when 2 kg of water has its temperature increase by 2 °C.

$$\Delta E = (2\,\mathrm{kg})\left(4180\,\frac{\mathrm{J}}{\mathrm{kg\,K}}\right)(2\,^{\circ}\mathrm{C}) = 16,720\;\frac{\mathrm{kg}}{}\frac{\mathrm{J}}{\mathrm{kg\;K}}\frac{^{\circ}\mathrm{C}}{} = 16,720\;\mathrm{J}$$

In this situation, the units of °C in the numerator cancel the units of K in the numerator. Why? The energy storage occurs because there is a *change* in temperature, and when temperature changes are involved, Δ°C and ΔK are interchangeable. For example, suppose the temperature of the water changes from 21 °C to 23 °C.

$$^{\circ}\mathrm{C\ units:}\quad T_2 - T_1 = 23\,^{\circ}\mathrm{C} - 21\,^{\circ}\mathrm{C} = 2\,^{\circ}\mathrm{C\ change}$$

$$\begin{aligned}\mathrm{K\ units:}\quad T_2 - T_1 &= (23 + 273.15)\,\mathrm{K} - (21 + 273.15)\,\mathrm{K}\\ &= (23 - 21) + (273.15 - 273.15)\,\mathrm{K} = 2\,\mathrm{K\ change}\end{aligned}$$

It does not hurt to work with temperature differences in K, but as long as only temperature *differences* matter, working with °C is also correct because the offset of 273.15 °C cancels.

3 Measurements to Determine System Characteristics

Two key parameters of the system control algorithm need to be measured. The first parameter is an estimate of the variability in the temperature signal while the system is operating in steady

state. That variability sets a lower bound on the size of the deadband for temperature control. In other words, the deadband has to be larger than the magnitude of the random variation in the temperature measurment. Otherwise, the system would respond to the noise, not the true deviation from the set point.

The second key parameter is the rate of temperature increase that is attained while the heater is turned on. If we know the heating rate, then we can estimate how long to turn on the heater in order to obtain a desired change in temperature. Although we can predict the heating rate from Equation (7), it is better to measure DT/Dt when we need a value of DT/Dt in the control code.

3.1 Measure Variability in T

Figure 2 is a conceptual representation of the temperature during steady-state operation of the fish tank. During steady state, the natural variability in the system causes the temperature to fluctuate about the nominal steady state temperature. Electrical noise in the breadboard and in the Arduino analog-to-digital (A/D) conversion circuit will also contribute to random variations in the temperature signal.

Given that there is noise in the temperature signal, we should not specify a deadband that is smaller than the magnitude of the noise. Use the following steps to directly measure the noise.

1. Write an Arduino program that only prints the temperature sensor reading (raw units are OK) to the Serial Monitor. Including time (in milliseconds or seconds) is OK. Just don't include any other formatting or messages.

2. Using tap water (or whatever water happens to be in your system), let the system run until the temperature appears to be constant

3. Extract a large number of temperature readings by copy/pasting from the Serial Monitor to a spreadsheet or text file

4. Use MATLAB to plot a histogram and evaluate σ (the standard deviation) of the data.

Recall that for salinity control, the deadband was set to $\pm 3\sigma$, where σ is the standard deviation in the output of the salinity sensor during steady-state operation. Since the temperature signal is less noisy than the salinity signal, it may be possible to use $\pm 2\sigma$ to estimate the deadband. However, you should keep the deadband for temperature control as a variable and adjust that deadband as necessary.

3.2 Measure the Maximum Rate of Heating

Figure 3 depicts an experiment where the heater is turned on for a period of time $\tau = t_{\text{stop}} - t_{\text{start}}$. The upper part of Figure 3 shows the power input versus time. The bottom part of Figure 3 is the temperature response: each gray dot represents a value of temperature measured with the thermistor. During heating the slope of the measured temperature versus time curve is

$$K = \frac{\Delta T}{\Delta t} \tag{8}$$

where K is a characteristic of the system. You can think of K as the heating rate at maximum power to the heater. The units of K are °C/s. The value of K in Equation (8) is determined by measuring DT/Dt, which will likely yield a different value than that predicted by Equation (7).

From Equation (7) we see that K increases when VI, the power input, increases. For a given power input, K decreases as mc_p increases. The rate of heat loss, $\dot{E}_{\text{out}}$, causes K to decrease, or at least to offset some of the contribution to K from the VI term in Equation (7).

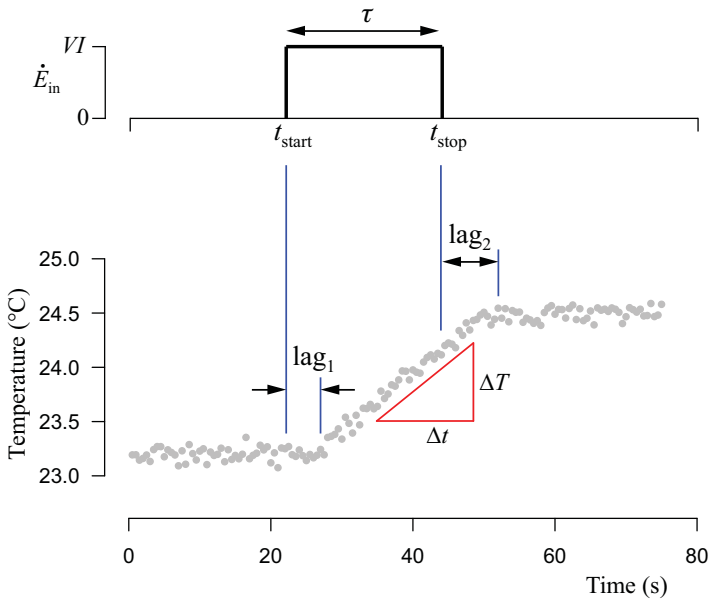


Figure 3: Power input and temperature response during experiment to measure system characteristics.

Although we could attempt to model each of the terms on the right hand side of Equation (7), it is more practical, efficient, and ultimately more accurate, to simply measure K from an experiment. We will assume that K is a constant even though the value of m may change from one session of running the fish tank to the next, and the heat loss to the ambient will vary with temperature of the pump, tubing, salinity sensor, as well as ambient temperature and air currents in the room.

3.3 Lags in Temperature Response

Figure 3 indicates two lags that may occur in response to changes in the heater on/off status. In both cases, thermal mass of the heater and the system contribute to a delay in the change in water temperature with time.

The first lag (lag_1) occurs immediately after the heater is turned on. The thermistor signal does not indicate an immediate rise in water temperature because the heat spreader and the water itself have thermal mass. Heat is transfered from the heat spreader to the water only after the heat dissipated by the resistor has caused a sufficient temperature increase in the heat spreader. The temperature of the pump, tubing, salinity sensors and fish tank walls also store some of the heat imparted by the heater, thereby delaying the apparent start of water heating.

The second lag (lag_2) occurs immediately after the heater is turned off. Once again, the thermal mass of the heat spreader is a contributing factor. Although the energy input to the heater ceases as soon as the power is turned off, the heat spreader is still warmer than the water, which allows the water temperature to rise even though there is no longer any energy addition to the system.

The size, and even the existence, of the lags will vary due to a number of factors such as proximity of the heater and thermistor in the fish tank, and the pump flow rate. Do not worry if your system characterization measurements do not reveal one or both lags. The key point is to measure K as the slope of temperature versus time while the heater is on. Being aware of the lags should help you interpret the measurements if the lags are evident in your data.

```
//  Arduino code snippet to turn the fish tank heater for a limited time.
//  Use this code to generate data to measure the K value for the tank

void loop() {
  int     heater_status;
  int     nave=30, reading;
  long    heater_start = 15000, heater_stop = 75000, now;
  float   T;

  now = millis() - start_time;

  if ( now >= heater_start & now <= heater_stop ) {
    digitalWrite( heater_power_pin, HIGH );
    heater_status = 1;
  } else {
    digitalWrite( heater_power_pin, LOW );
    heater_status = 0;
  }

  T = thermistor_reading_ave( thermistor_power_pin, thermistor_input_pin, nave);
  Serial.print(now);
  Serial.print("\t");   Serial.print(heater_status);
  Serial.print("\t");   Serial.println(T);
}
```

Listing 1: Code snippet to display data for measurement of temperature variability and maximum heating rate, K. This code is *not* used for controlling the temperature of the fish tank. The code is available on the class web site as `heaterCharacterizationSnippet.cpp`.

3.4 Measurement Code

Listing 1 shows the `loop` function from an Arduino code that can be used to measure the thermal system variability (§ 3.1) and the maximum heating rate K (§ 3.2).

4 Control Algorithm

From the measurements described in the preceding section, we obtain the deadband, ΔT_{db}, and the maximum heating rate K. In this section we describe the temperature control algorithm.

4.1 Thermal Control Strategy

Goal: Keep the temperature of the water within the deadband

Basic Algorithm:

measure T
if (T is less than the lower control limit) {
    turn on heater
    wait Δt_{heat}
    turn off heater
}

If Δt_{heat} is the time that we leave the heater on to correct a deviation from the temperature set point, how do we determine Δt_{heat}? And how would we "wait Δt_{heat}" without stopping the code from doing other work?

4.2 How Long to Turn on the Heater

At any instant of time we measure the temperature of the water T. The design specification is the setpoint temperature T_{sp}. If the measured temperatue is below the lower bound on the deadband, compute the error in the temperature as

$$T_{\text{err}} = T - T_{\text{sp}} \tag{9}$$

Estimate the time that the heater should be open by using T_{err} as ΔT in Equation (8), i.e.,

$$\Delta t_{\text{heat}} = \frac{-T_{\text{err}}}{K} \tag{10}$$

The minus sign is necessary because heat needs to be added when $T < T_{\text{sp}}$, i.e., when the error is negative, and Δt needs to be positive. The preceding considerations suggest the following pseudocode for the implementation of the temperature control

Basic Heat Addition Algorithm:

measure T
if (T is less than the lower control limit) {
    $\Delta t_{\text{heat}} = (T_{\text{sp}} - T)/K$
    turn on heater
    wait Δt_{heat}
    turn off heater
}

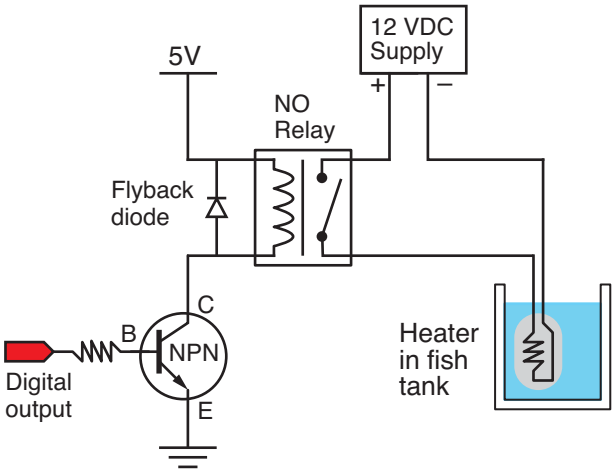


Figure 4: Circuit for controlling power to the heater.

4.3 Cascade Control Circuit for the Heater

Figure 4 is a schematic diagram of the heater control circuit. A SPST relay controls the current to the heater from a 12 V power supply. Actuation of the relay is controlled by an NPN transistor, with its base connected to a digital I/O pin on an Arduino. A flyback diode on the 5V side of the relay protects the 5V circuit. No flyback diode on the heater side of the relay is necessary because the heater is a pure resistive load. Unlike the solenoid valves in the salinity control circuit, the heater has no mechanism for storing energy while current is flowing through it.

4.4 Consideration of Salinity Control Deadtime

The thermal system is slower and more stable than the salinity control system. For example, the thermal system is unlikely to overshoot unless the heater is left on for a long time. On the other hand, the deadtime is crucial for the successful operation of the salinity control algorithm.

To help the salinity control algorithm remain stable and in good control, we should make sure that the duration of heat input is *not* greater than the salinity deadtime. With that modification, the *Basic Heat Addition Algorithm* listed above becomes the *New Heat Addition Algorithm*.

New Heat Addition Algorithm:

measure T
if (T is less than the lower deadband limit) {
  $\Delta t_{\text{heat}} = (T_{\text{sp}} - T)/K$
  If $\Delta t_{\text{heat}} >$ deadtime {
    $\Delta t_{\text{heat}} =$ deadtime
  }
  turn on heater
  wait Δt_{heat}
  turn off heater
}

Listing 2 is an outline of the code to control both temperature and salinity. Many details are missing. The logic in the incomplete code should enable you to write a working Arduino program.

```
//  File:  wait_for_deadtime_temp.ino
//
//  Structure of temperature and salinity control code to implement a deadtime
//  during which no salinity correction is made.
//
//  This code is incomplete and will not compile.

unsigned long last_salinity_update;  //  Time of last correction

void setup() {
  Serial.begin(9600);
  last_salinity_update = millis();   //  Initial value; updated later
}

void loop() {
  float LCL, UCL, salinity, LCLT;
  int deadtime = ... ;

  salinity = salinity_reading( ... );
  Tw = temperature_reading( ... );
  update_LCD( ... );

  // -- Check for temperature limit.  What about UCLT?
  if (  Tw < LCLT ) {
     // add heat:  See "New Heat Addition Algorithm" in the class notes
  }

  // -- Check for deadtime
  if ( ( millis() - last_salinity_update ) > deadtime ) {

    if ( salinity>UCL ) {
      //  add DI water:  several missing steps
      last_salinity_update = millis();
    }

    if ( salinity<LCL ) {
      //  add salty water:  several missing steps
      last_salinity_update = millis();
    }
  }
}
```

Listing 2: Outline of code to control salinity and temperature of the fish tank.

The pseudocode in the *New Heat Addition Algorithm* should substituted for the `if` block around the "`add heat`" comment statement in the middle of Listing 2.

4.5 LCD Display for Temperature and Salinity Control Status

Figure 5 shows a layout of the LCD panel to indicate the status of both the temperature and salinity control system. The first row contains the labels for three columns of information. LCL and UCL are the lower and upper control limits, respectively. SP is the set point. The second row (index = 1) indicates the control parameters for the salinity control system. The third row indicates the parameters of the temperature control system. The last row indicates the current values of salinity (S), temperature (T) and the heater status (H), which is either "on" or "off".

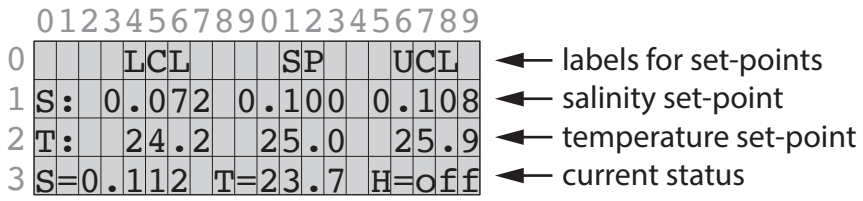
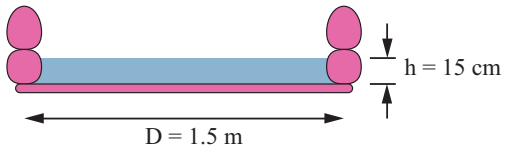


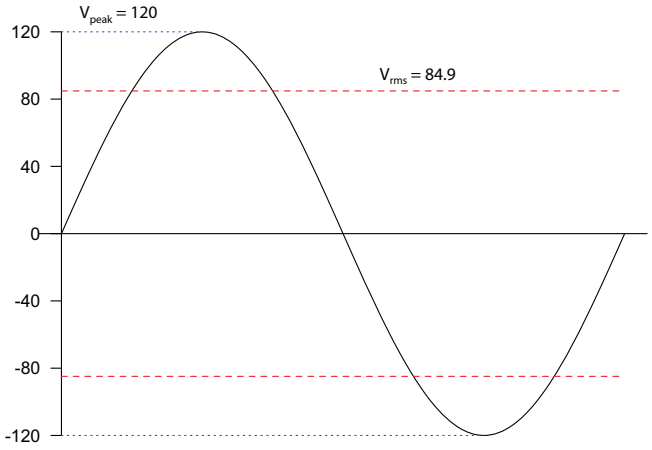
Figure 5: Layout of the 20×4 character display to show status of both salinity and temperature control.

5 In-Class Exercises

1. How long would it take to raise the temperature of a small, inflatable swimming pool 15 cm deep and 1.5 m in diameter by 1 °C using an electrical heater driven by a 12 VDC power supply capable of providing 10 A? What is the electrical resistance of the heater required?



2. Re-compute the temperature rise time, current requirement, and electrical resistance needed from Problem 1, if the a 120V AC power supply is used instead of 12 VDC power supply. For an AC voltage, the power is $P = V_{\text{rms}} I$ where $V_{\text{rms}} = V_{\text{peak}}/\sqrt{2}$.



3. If you have not yet done so, prepare for the system characterization experiments with the following steps.

   a. Create the relay circuit in Figure 4 to control power to the heater. By design, there is no flyback diode across the heater. Why?

   b. Write an LCD display function to show the heater temperature and indicate whether the heater power is on or off. Either use the display in Figure 5 or a simplified version that just shows the temperature and heater status.

c. Write a complete Arduino code to turn on the power to the heater for a specified time interval, and to print the time, heater status, and thermistor temperature to the Serial Monitor. Refer to Listing 1 for the main parts of this code.

4. With the preceding components and code in place and tested, perform the following experiment.

   a. Choose a set point temperature that is 1 °C to 3 °C above ambient.

   b. Let the system run for 5 minutes so that the components come into thermal equilibrium. Watch the temperature displayed on the LCD panel and Serial Monitor to determine when the system is in steady state.

   c. Run the code (like that in Listing 1) to turn the heater on for a fixed time. Copy the output from the Serial Monitor to an Excel spreadsheet.

   d. Plot the data and use other means (e.g., a line fit through the data) to determine the value of K from the plot of temperature versus time and Equation (8).

   e. Extract an estimate of the standard deviation of the temperature readings during a time period when the temperature is approximately constant, i.e., during the steady state operation before or after the heat is added.