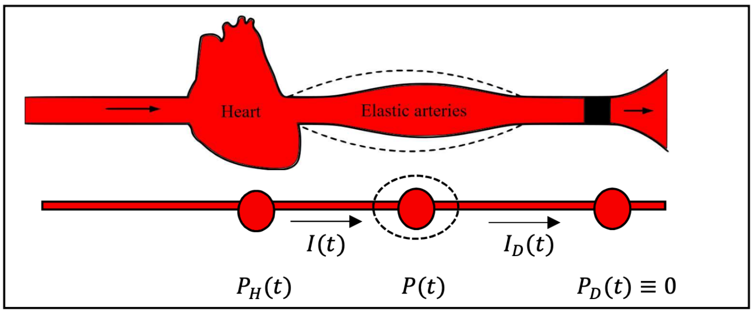
1 Windkessel Model

The *Windkessel* model of blood flow through an artery was defined by Otto Frank at the end of the nineteenth century. The Wikipedia page on this model contains an interesting illustration and an account of what the word *windkessel* means, and of how it is related to the model we are going to discuss. The model is over 120 years old, yet it is still a foundation for modeling work in human physiology. The idea behind the model is that an artery is an elastic reservoir that expands when the pressure inside it is high. In the basic model, the premise is that the volume of the vessel is proportional to the pressure in it.



The schematic in the Figure shows a simple Windkessel model with three nodes: one that represents the heart, one that represents an elastic artery, and one that represents a point downstream from the elastic artery. The pressures at the three nodes, representing the three locations in the arterial system, are $P_H(t)$, $P(t)$, and $P_D(t)$. By the basic premise of the Windkessel model, $P(t) = KV(t)$ for a positive constant K. Another premise of the Windkessel model is that the volumetric flow of blood between two locations is proportional to the difference in pressure between the two. If, as is represented in the schematic, $I(t)$ is the volumetric flow from the heart to the elastic artery, and $I_D(t)$ is the flow from the elastic artery to the downstream point, then

$$I(t) = \gamma\,[P_H(t) - P(t)] \qquad \text{and} \qquad I_D(t) = C\,[P(t) - P_D(t)]\,, \tag{1}$$

for appropriate positive constants γ and C. Conservation of mass then implies that

$$I(t) - I_D(t) = \frac{\mathrm{d}V}{\mathrm{d}t} = K\frac{\mathrm{d}P}{\mathrm{d}t}, \tag{2}$$

since as blood flows from one node to another there must be a corresponding change in the blood volume found at the nodes. If we now use the fact that $P_D(t) \equiv 0$, and we substitute the expressions for $I(t)$ and $I_D(t)$ in Eq. (1) into Eq. (2), we obtain

$$\gamma P_H(t) = K\frac{\mathrm{d}P}{\mathrm{d}t} + (C+\gamma)P. \tag{3}$$

1.1 First Question

Consider the ODE (3) for values of the parameters on the order of 1 (they don't have to all be 1; choose a few different values, but choose them around 1). Suppose that the pressure in the heart is periodic; in fact, assume that it's a constant plus a sine function: $P_H(t) = 1 + \sin(2\pi\omega t)$ Clearly, the magnitudes are not realistic, but let's make the period realistic. If we measure t in seconds, what is a range of realistic values of ω?

If $P_H(t)$ is periodic (which it should be, because it's the pressure in the heart) any solution of the ODE with any initial value will have a transient part –of the form $\varphi e^{-\frac{C+\gamma}{K}t}$– which decays to zero, and a periodic part. Because we are interested in persons who live past the transient time, we're concerned here only with the periodic parts of solutions.

Solve the ODE for one of those realistic values of ω, with the values $\gamma = C = 1, K = 2$, and graph $P_H(t)$ and $P(t)$ on the same axes. Discuss their periods and amplitudes and discuss the phase shift. Interpret all of these in terms of the application field and the model.

Solution. The parameter ω is usually reserved for *angular frequency* ($\omega = (2\pi)/t$, where t is the period measured in seconds). However, looking at our definition of $P_H(t)$, it is clear that we are here using ω to describe the *ordinary frequency*, which is measured in Hertz (or $1/t$). In this model, this frequency parameter refers to the heart rate, which depends on age and other factors. For our purposes, we choose a normal range for adults of 60 to 100 beats per minute (bpm); in seconds that translates to 1 to ~ 2 beats per second (bps). So, a realistic range is $1 \leq \omega \leq 2$.

Choosing $\omega = 1$, along with the values $\gamma = C = 1$ and $K = 2$, we now look for a numerical solution to Eq. (3) (the closed-form solution can be easily obtained using an integrating factor, but where's the fun in that? ☺). Now, for the initial value P_0 I did have to do some online search to find reasonable blood pressure values (I'm admittedly VERY ignorant on this stuff). I found that a normal heart rate is at or below 120mm Hg, where mm Hg refers to *millimeters of mercury* and $1\text{mm Hg} \approx 16\text{kPa}$. Since Pascal is a more known pressure unit for me, that's what I'll use; thus I will set $P_0 = 16000\text{Pa}$ and will let the code run until a minute of readings pass, i.e., until $t = 60$.

```
using namespace std;

int main(int argc, const char * argv[]) {

    //SET PARAMETERS
    const double gamma {1.0};
    const double omega {1.0};
    const double C {1.0};
    const double K {2.0};
    const double dt {0.1};

    vector <double> Ndata {};
    double Ph {};
    double P {16000.0};

    double t {0.0};

    for (int i {0}; i <= 100; i++){
        t = i * dt;
        Ndata.push_back(P);
        Ph = 1.0 + sin(2.0 * M_PI * omega * t);
        P = P + dt * ( gamma * Ph - ( (C + gamma) * P) )/K;
    }

   ofstream myfile ("num_data.csv");
    for (int i{0}; i <= 100; i++) {
        if (i != 100) {
            myfile << Ndata.at(i) << ",";
        } else {
            myfile << Ndata.at(i) << endl;
        }

    }

    return 0;
}
```

Numerical results are plotted on Fig. (1). Note that, even though we start off with a relatively high pressure of 16kPa, it quickly (after roughly ~ 10 seconds) decays to an oscillating amplitude of merely $\sim 0.42 - 0.58$ Pa. My (most likely very wrong) interpretation of this is that, at birth, the heart is pumping blood into the artery at a faster-than-usual rate, thus building up very high pressure on the walls of the artery. Why it only takes mere seconds for this high pressure to die out is completely beyond my very limited biology knowledge. As for the period of the sinusoidal part, we can tell that the heart rate is quite fast during that first minute. Intuitively, we may guess that if we let the code run for $t \to \text{adulthood}$ we should see longer wavelengths, since the heart does not beat as fast as an adult as during childhood.

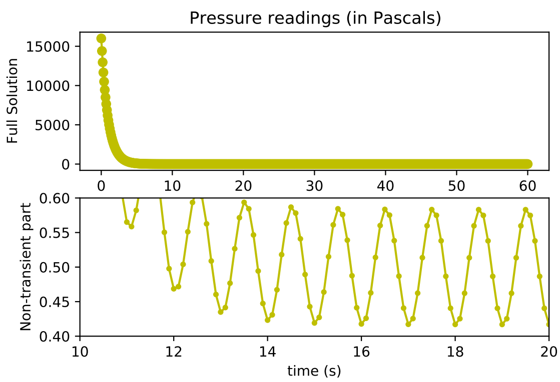


Figure 1: Above is the full numerical solution. At the bottom we show these results in $t \in [10, 20]$ for clarity. Note that the transient term dies out after ~ 13 seconds.

We now plot an interval of the non-transient part of the artery pressure readings ($P(t)$) along with the heart pressure readings ($P_H(t)$):

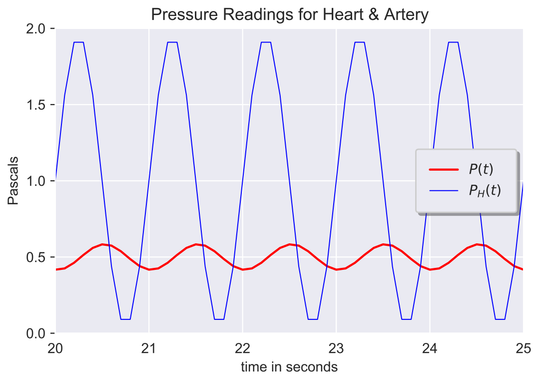


Figure 2: Pressure on the artery walls ($P(t)$) and the heart ($P_H(t)$), superimposed, for the interval $t \in [20, 25]$. The phase shift is due to the fact that the heart has to first pump the blood onto the artery, so there is a bit of a delay from the instant the heart pumps and the instant the blood passes through the artery.

The heart's pressure remains steadily oscillating throughout (even during the transient period, which was cut out from the graph for a clearer comparison). My interpretation (which, again, must be taken with a huge grain of salt!) of the heart's amplitude range from 0–$2\mathrm{Pa}$ is that, at the very instant the heart pumps blood to the artery, the pressure is momentarily relaxed, but then instantly increases again and the pumping cycle continues. □

1.2 Second Question

A more realistic model (in terms of pulse shape, not in terms of magnitude) of a pressure pulse in the heart might be that $P_H(t)$ has a period of 1 second, it rises linearly from 0 to 1 in the first 40 milliseconds, it decreases from 1 back to 0 linearly in the next 40 milliseconds, and then it's 0 for the next 920 milliseconds. Again, with this function $P_H(t)$, and with the values $\gamma = C = 1$, $K = 2$, solve the ODE for the steady state solution and graph $P_H(t)$ and $P(t)$ on the same axes.

Solution. It takes $t = 0.04$ seconds for P_H to go from 0 to 1 Pascals; this yields a line with slope 25. Similarly for the other two segments. Overall P_H is given by the piece-wise function

$$P_H(t) = \begin{cases} 25t & \text{if } 0 \leq t \leq 0.04; \\ 2 - 25t & \text{if } 0.04 < t \leq 0.08; \\ 0 & \text{if } 0.08 < t \leq 1. \end{cases} \tag{4}$$

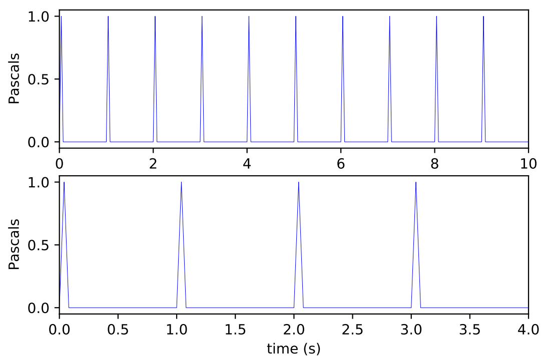


Figure 3: Graph of Eq. (4) on the whole interval $t \in [0, 10]$ (above), and on $t \in [0, 4]$ below, for clarity.

Now we use this $P_H(t)$ in our C++ code to evaluate $P(t)$:

```
using namespace std;


double Phfunc(double &time){
    vector <double> N {};
    vector <double> vPh {};
    double Ph {};

    for (int n {0}; n <= 59; n++){
    N.push_back(n);
    }

    for (int n : N){
        if ( (time >= (0.0 + double(n)) ) && (time <= (0.04 + double(n)) ) ) {
                Ph = 25.0 * (time - double(n));
            } else if ( (time > (0.04 + double(n)) && (time <= (0.08 + double(n)) ) ) ){
                Ph = 2.0 - (25.0 * (time - double(n)) );
            } else if (  (time > (0.08 + double(n)) && (time <= (1.0 + double(n))   ) ) ){
                Ph = 0.0;
            }
    }
    return Ph;
}



int main(int argc, const char * argv[]) {


    //SET PARAMETERS
    const double gamma {1.0};
    const double C {1.0};
    const double K {2.0};
    const double dt {0.01};


    vector <double> Ndata {};
    vector <double> N {};
    double P {16000.0};
    double t {};


    for (int i {0}; i <= 6000; i++){
        Ndata.push_back(P);
        t = i * dt;
        P = P + dt * ( gamma * Phfunc(t) - ( (C + gamma) * P) )/K;
    }

    ofstream myfile ("num_data2.csv");
     for (int i{0}; i <= 6000; i++) {
         if (i != 6000) {
             myfile << Ndata.at(i) << ",";
         } else {
             myfile << Ndata.at(i) << endl;
         }

     }

    return 0;
}
```

Then over to Python:

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import pandas as pd

font = {'family' : 'serif',
        'weight' : 'normal',
        'size'   : 14}

arr = np.arange(0,61)
```

```python

def Ph(t):
    for n in arr:
        if 0.0 + float(n) <= t <= 0.04 + float(n):
            return 25.0 * (t - float(n))
        elif 0.04 + float(n) < t <= 0.08 + float(n):
            return 2.0 - 25.0 * (t - float(n))
        elif 0.08 + float(n) < t <= 1.0 + float(n):
            return 0.0


t = np.linspace(0, 60,  num=6001)
vPh = np.vectorize(Ph)
y = vPh(t)


# Create the figure and and one axes
fig, ax1 = plt.subplots(1, 1)

# Create 2nd axes sharing the x-axis with ax1
ax2 = ax1.twinx()



#read data output from C++ code
Ndata = pd.read_csv("~/MyXCodeProjects/MathModeling_BloodPressure/num_data2.csv",
        header = None)
Ndata = Ndata.transpose()


# Add labels for the legend
function1 = ax1.plot(t, Ndata[0], 'r', label=r'$P(t)$')
function2 = ax2.plot(t, y, 'b', label=r'$P_H(t)$', linewidth=0.3)


# Create the legend by first fetching the labels from the functions
functions = function1 + function2
labels = [f.get_label() for f in functions]
plt.legend(functions, labels, fancybox=True, framealpha=1, borderpad=1, shadow=True,  loc=0)


ax1.set_xlabel('time in seconds')
ax1.set_ylabel('Pascals')
ax2.set_xlim(10, 60)
ax1.set_ylim(0, 0.05)
ax1.set_xlim(10, 60)
ax2.set_ylim(0, 0.05)
ax1.set_yticks([0, 0.02, 0.04])
ax2.set_yticks([])
ax1.set_xticks([10, 20, 40, 60])

# Add the title
plt.title('Pressure Readings for Heart & Artery')

#[Optional] #use seaborn rendering engine (imported as sns) to change the background
sns.set_style("darkgrid")

# Save the figure
# plt.savefig('Figures/BloodPressure_Per_prefinal.pdf')
plt.close()
```

And voilà! Here is the end result of our effort:

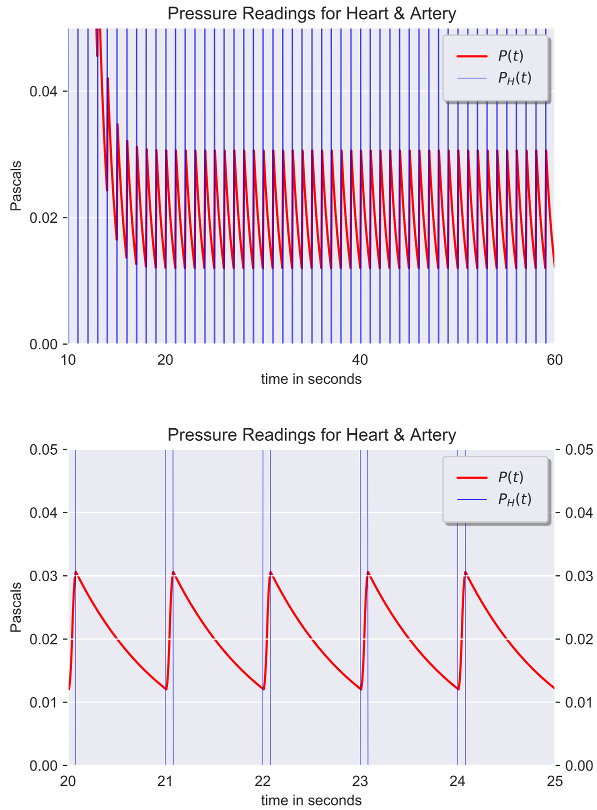


Figure 4: Graph of $P(t)$ and $P_H(t)$, with the latter coming from Eq. (4); for clarity we only show the results for $t \in [20, 25]$ at the bottom. We can see that in this scenario the pressure on the artery walls is much weaker than in our previous results. This behavior was to be expected, since in our new definition of $P_H(t)$ –as given by Eq. (4)– the heart's pressure is mostly zero with only sporadic spikes that last for a few milliseconds. $\square$