Term Project Report
Team 10

Asude DUMAN	040220650
Ebrar GÜNDÜZ	040220748
Fatmagül ZAHTER	040210628
Semanur GÜLMEZ	040220619
Zeynep SÖYLEMEZ	040240797

January 21, 2025

Istanbul Technical University
Faculty of Electrical and Electronics Engineering

KON 307E - Programming Techniques in Control

Contents

List of Figures

INTRODUCTION

Extracorporeal Membrane Oxygenation (ECMO) is a medical device designed to support and restore the function of the heart and lungs in patients suffering from severe cardiac and/or respiratory conditions. By extracting blood, passing it through its internal mechanism, and ensuring gas exchange at desired levels, ECMO allows the heart and lungs to recover. It is primarily used as a temporary treatment method to provide cardiopulmonary support for patients experiencing acute and reversible cardiac and/or respiratory failure [1].

The objective of this project is to design and simulate an ECMO device using MATLAB and Simulink. As ECMO functions as an extension of the respiratory and circulatory systems, these systems were also implemented in the model. The lung mechanics and ECMO pump were modeled as electrical circuits, while other components were defined based on their governing equations.

Various simulations were conducted to represent different scenarios, including a healthy individual, an exercising individual, a patient with damaged lungs, and a COVID-19 patient. For simplicity, all scenarios assumed that the individual was unconscious and that lung inflation was maintained using a ventilator.

It was successfully demonstrated that a COVID-19 patient with critical blood partial pressure levels could be connected to the ECMO system. The simulation showed that, after the connection, the blood gas levels returned to nominal values, indicating that the ECMO system was effective in supporting and stabilizing the patient's condition.

A RESPIRATORY SYSTEM MODEL

A.1 Lung Mechanics Model

A.1.a

Model of electrical analogous of a lung is shown in Figure 1. Here, it is assumed that the patient is not breathing on its own, and the lungs are filled with air via the pressure provided from the ventilator, shown as $P_{vent}(t)$. It is controlled via a square wave reference signal with amplitude of 8 V, frequency of 0.2 Hz and duty cycle of 35%. The signal is also passed through a filter, $G_f(s) = \frac{1}{0.1s+1}$, to smoothen the wave. R_{tube} and R_{lung} represent the resistances of the tube of the ventilator and the lung respectively. C_{lung} is the compliance of the lungs.

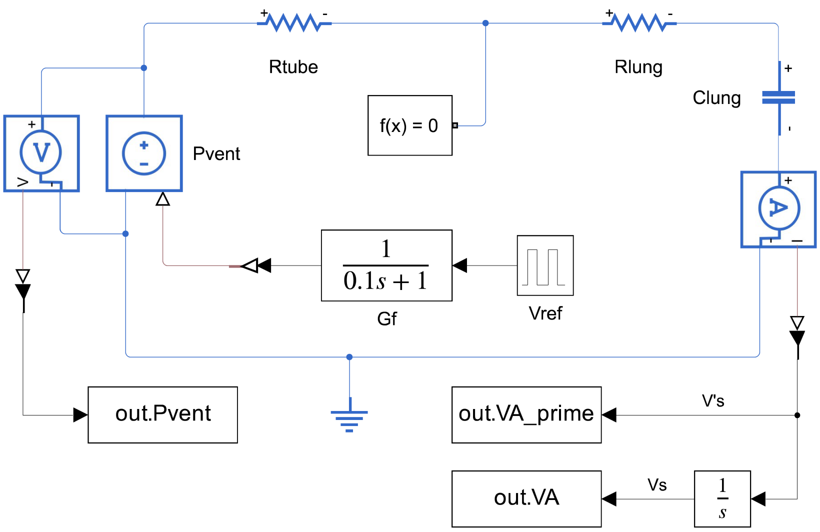

Figure 1: Electrical analogous of lung mechanics

A.1.b

Figure 2 shows the results of the simulation with the following values.

- $R_{lung} = 0.24\ cmH_2O \cdot s \cdot l^{-1}$
- $R_{tube} = 3.76\ cmH_2O \cdot s \cdot l^{-1}$
- $C_{lung} = 0.08\ l \cdot cmH_2O^{-1}$
- $|V_{ref}| = 8\ V$

Figure 2 illustrates the simulation results under a constant applied pressure of 8 cmH_2O. The provided pressure ($P_{\text{vent}}(t)$) leads to an increase in lung pressure, which in turn reduces the pressure difference across the respiratory system.

Initially, when the pressure difference is large, the alveolar airflow ($V'_A(t)$) is high. As the pressure difference diminishes over time, the alveolar airflow decreases correspondingly.

The tidal volume ($V_A(t)$) exhibits a growth phase when there is a positive alveolar airflow, signifying an influx of air into the lungs. Conversely, during phases of negative alveolar airflow, the tidal volume decreases, indicating air leaving the lungs.

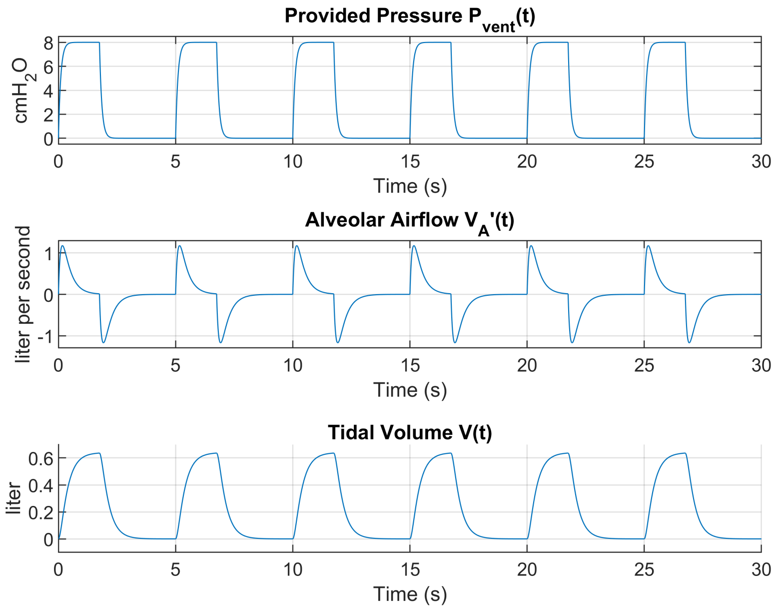

Figure 2: $P_{vent}(t),\ \dot{V}_A(t)$ and $V(t)$ with $|V_{ref}| = 8\ V$

A.1.c

To find the transfer function of the system, given $G(s) = \frac{V(s)}{P_{vent}(s)}$, using Kirchhoff's Voltage Law:

$$V(t) = R_{tube} \cdot I(t) + R_{lung} \cdot I(t) + \frac{1}{C_{lung}} \int I(t)\,dt + V_C(0)$$

Assuming $V_C(0) = 0$ and substituting $V(t) = P_{vent}(t),\ I(t) = V_A'(t)$ and $\int I(t)\,dt = V(t)$

$$P_{vent}(t) = R_{tube} \cdot V_A'(t) + R_{lung} \cdot V_A'(t) + \frac{1}{C_{lung}} V(t)$$

Taking the Laplace Transform and substituting $V_A'(s) = sV(s)$:

$$P_{vent}(s) = R_{tube} \cdot sV(s) + R_{lung} \cdot sV(s) + \frac{1}{C_{lung}} V(s)$$

Solving the equation:

$$\frac{V(s)}{P_{vent}(s)} = \frac{1}{s(R_{tube} + R_{lung}) + \frac{1}{C_{lung}}} = \frac{C_{lung}}{s(R_{tube} + R_{lung})C_{lung} + 1}$$

Substituting the values results in:

$$G(s) = \frac{V(s)}{P_{vent}(s)} = \frac{0.08}{0.32s + 1}$$

Open Loop function is found with cascading the filter $G_f(s)$ to $G(s)$:

$$G_{OL}(s) = G(s) \cdot G_f(s) = \frac{0.08}{0.32s + 1} \cdot \frac{1}{0.1s + 1} = \frac{0.08}{0.032s^2 + 0.42s + 1}$$

The poles of the $G_{OL}(s)$ are -10 and -3.125. There isn't any zero, and the gain K is 2.5.

A.1.d

It is desired to control the system in a closed loop with a PI Controller. The chosen controller is $F_{PI} = k\frac{s+3.125}{s}$, with desired settling time $t_s = 0.8\ s$ and damping ratio $\zeta = 0.93$. Open loop transfer function combined with the PI Controller is:

$$F_{PI}(s) \cdot G_{OL}(s) = k\frac{s+3.125}{s}\frac{1}{0.032s^2+0.42s+1} = k\frac{5}{2s^2+20s}$$

To find the locations of the desired poles in the Root Locus, calculate the natural frequency and the damping angle:

$$\omega_n = \frac{4}{\zeta \cdot t_s} = \frac{4}{0.93 \cdot 0.8} \approx 5.78$$

$$\theta = cos^{-1}(\zeta) = cos^{-1}(0.93) \approx 0.38$$

$$s = -\omega_n\, cos(\theta) \pm j\, \omega_n\, sin(\theta) = -5.0000 \pm 1.9761i$$

MATLAB plot of the root locus and the desired closed loop pole locations are shown in Figure 3. Since the desired poles lie on the root locus, they can be achieved by changing k. Using $rlocfind()$ reveals the gain at the desired location as $k = 11.562$.

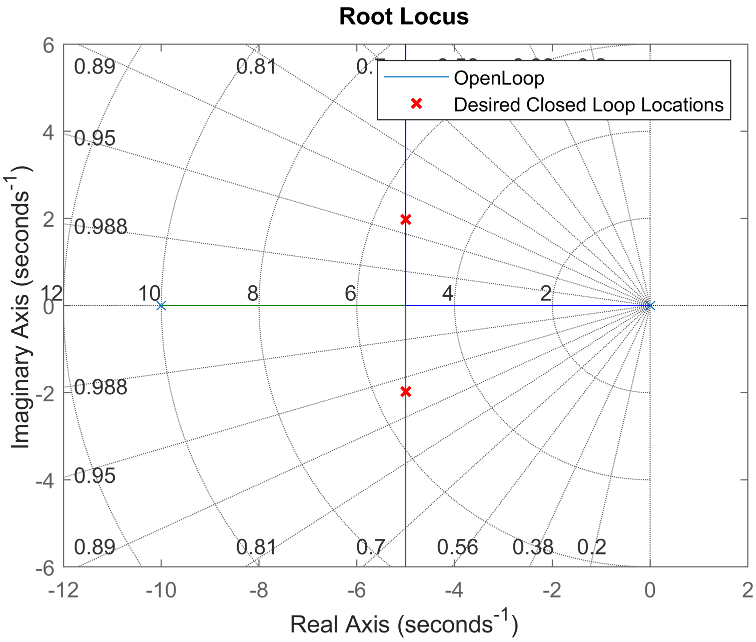

Figure 3: Root Locus of $F_{PI}(s) \cdot G_{OL}(s)$ along with desired closed pole locations

A.1.e

The final model of the lung mechanics is shown in Figure 4. Simulation results of the system with $V_{ref} = 0.45\ l$ is shown in Figure 5.

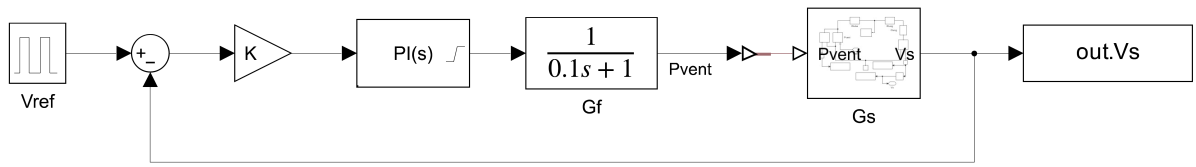

Figure 4: Simulation model for the lung mechanics

A closer look to the Figure of Tidal Volume, reveals that the $\zeta = 0.93$ is satisfied but $t_s = 0.8$ is not. The measured settling time value from the simulation is $t_s = 0.93$. Therefore the controller isn't able to control the tidal volume in the desired conditions. This mismatch in calculations is a result of estimation in t_s. In reality, t_s is calculated as:

$$t_s = -\frac{log(0.02 \cdot \sqrt{(1-\zeta^2)})}{\zeta \omega_n}$$

Which reveals that for $t_s = 0.8$ and $\zeta = 0.93$, ω_n must be 6.6034. But the corresponding closed loop pole locations does not lie on root locus, therefore they cannot be achieved with the given $F_{PI}(s)$

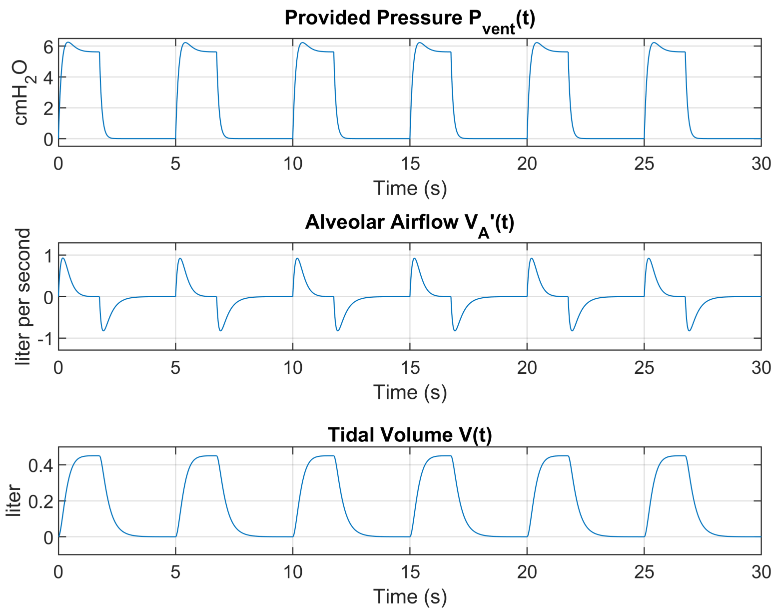

Figure 5: $P_{vent}(t)$, $\dot{V}_A(t)$ and $V(t)$ with $|V_{ref}| = 0.45\ L$

A.2 Dead Space and Alveolar Gas Exchange Model

A.2.a

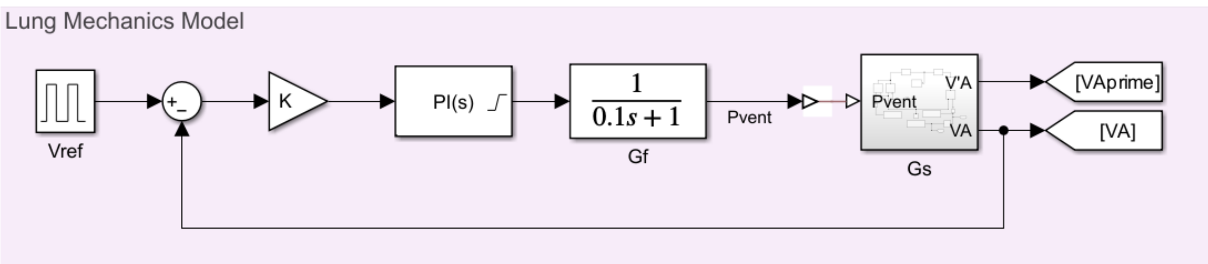

Figure 6: Final Lung Mechanics Model

Given equations are implemented to Simulink or MATLAB, the results are as follows, the codes for the MATLAB Function blocks are in the Appendix.

- Eq. 2.1 and 2.2

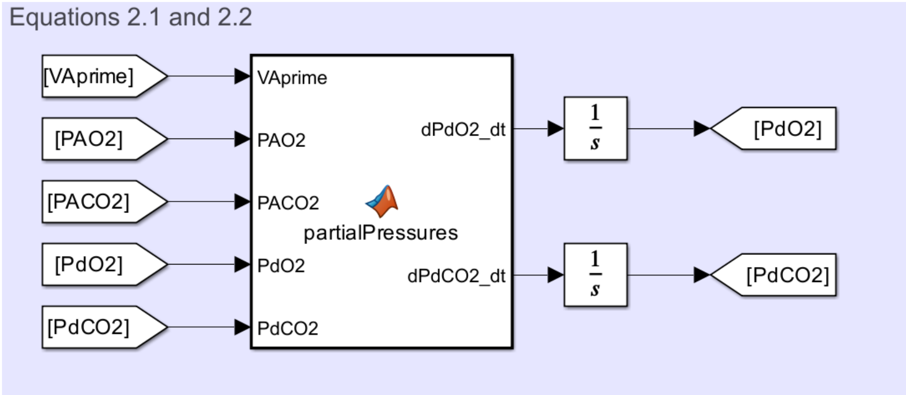

Figure 7: Simulink implementation of Eq 2.1 and 2.2

- Eq. 2.3 has been directly implemented in MATLAB since it calculates a constant value.
- Eq. 2.4 and 2.5

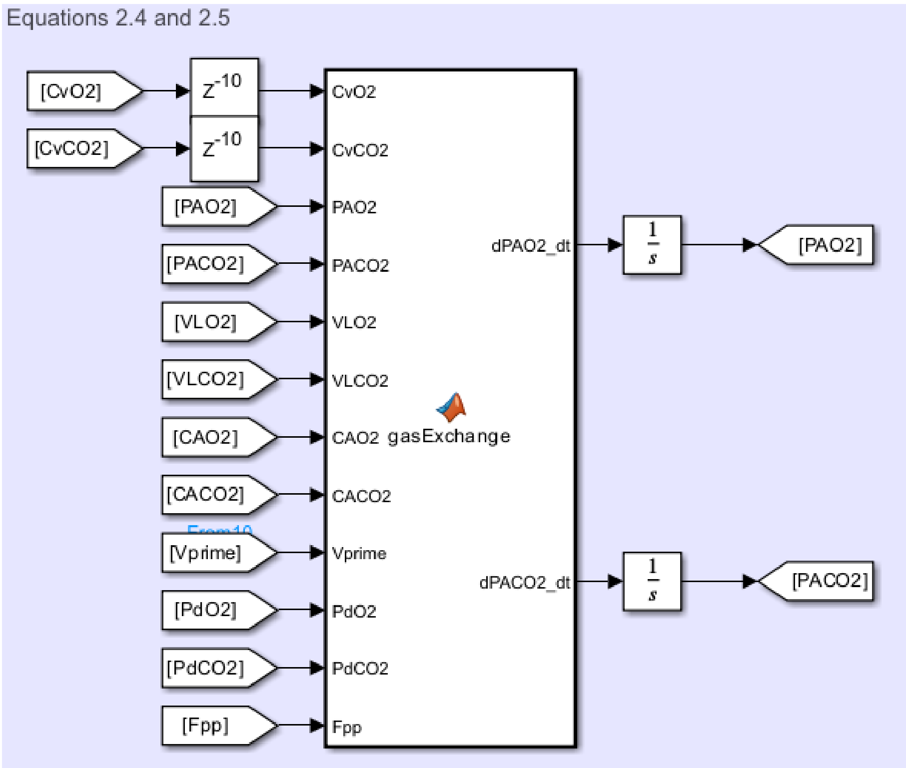

Figure 8: Simulink implementation of Eq. 2.4 and 2.5

- Eq. 2.6

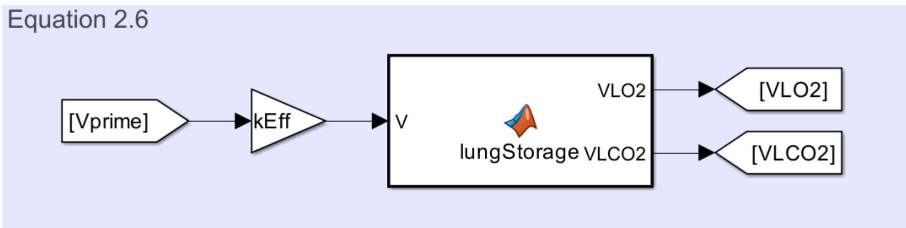

Figure 9: Simulink implementation of Eq 2.6

- Eq 2.7

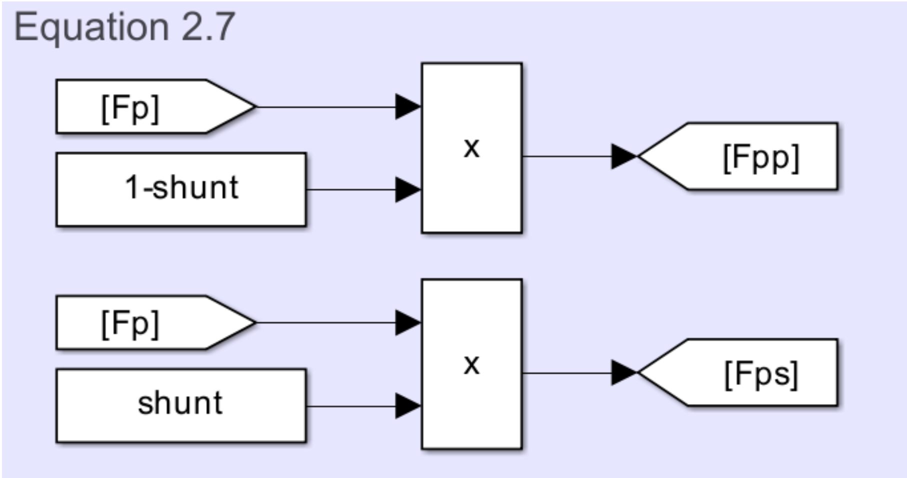

Figure 10: Simulink implementation of Eq 2.7

- Eq 2.8 and 2.9

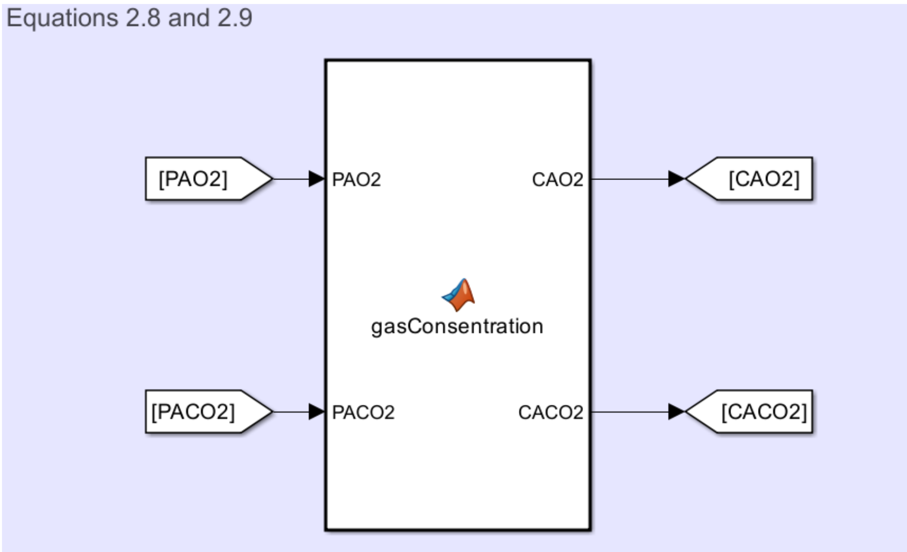

Figure 11: Simulink implementation of Eq 2.8 and 2.9

- Eq 2.10

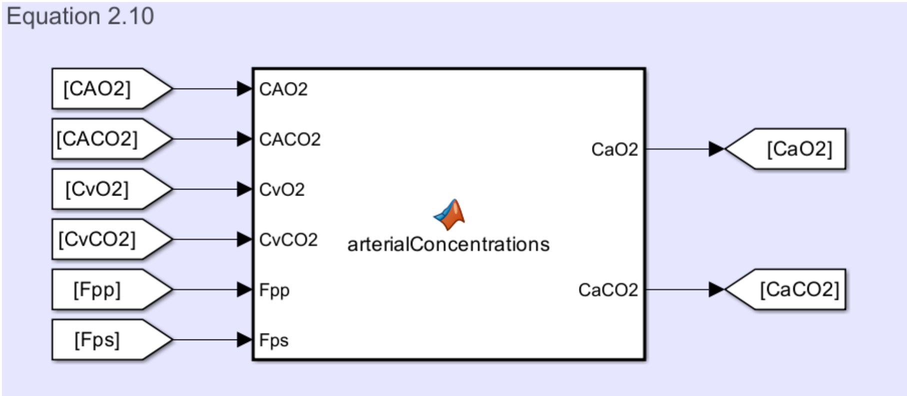

Figure 12: Simulink implementation of Eq. 2.10

- Eq. 2.11

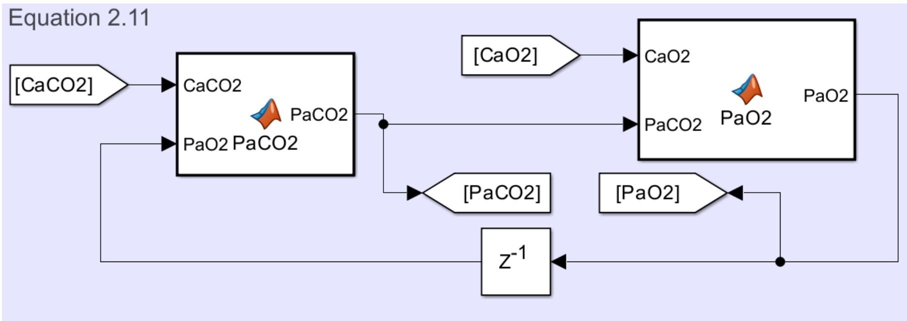

Figure 13: Simulink implementation of Eq. 11

- Given that $V'(t) = k_{eff}\ V'_A$

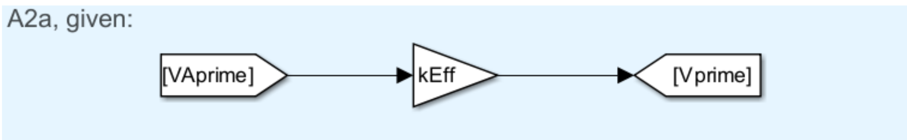

Figure 14: Simulink implementation of $V'(t) = k_{eff}\ V'_A$

A.2.b

Given that, $F_p(t) = 0.075\ l$ and considering a healthy subject's parameters, simulating the system for 5 minutes and plotting the last 60 second to view the arterial oxygen and carbon dioxide results in Figure 15. For a healthy subject, arterial pressures of O_2 and CO_2 are around 93 $mmHg$ and 40 $mmHg$, respectively.

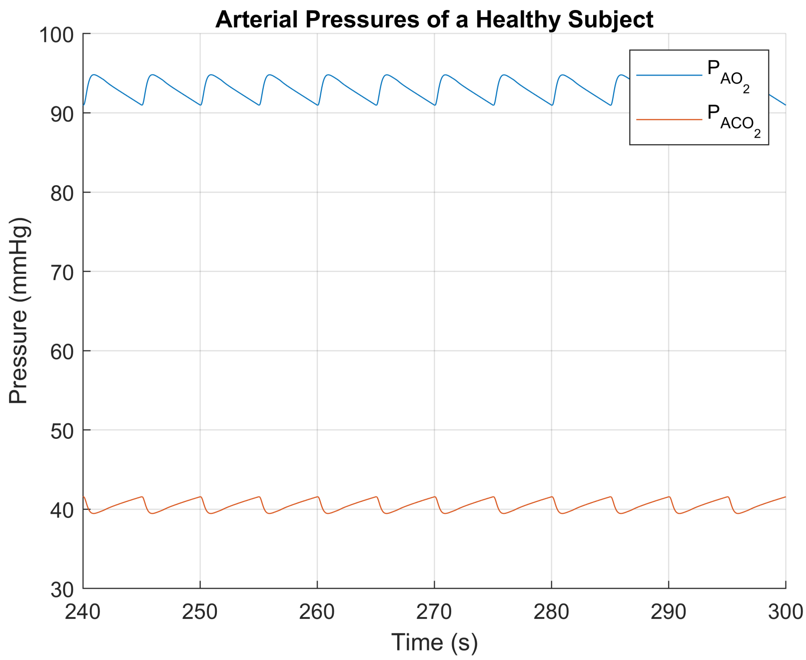

Figure 15: Arterial pressures of a healthy subject

A.2.c

The blood flows are represented as periodic functions given as,

$$F_p(t) = \begin{cases} 1 + 460 \sin\left(\frac{\pi}{0.25}t\right)^{1.55}, & 0 \le t \le 0.25, \\ 1, & 0.25 < t \le 0.84. \end{cases}$$

$$F_s(t) = \begin{cases} 1 + 670 \sin\left(\frac{\pi}{0.19}t\right)^{2}, & 0 \le t \le 0.19, \\ 1, & 0.19 < t \le 0.84. \end{cases}$$

Implementing these equations into Simulink and plotting the results are shown in Figure 16 and 17

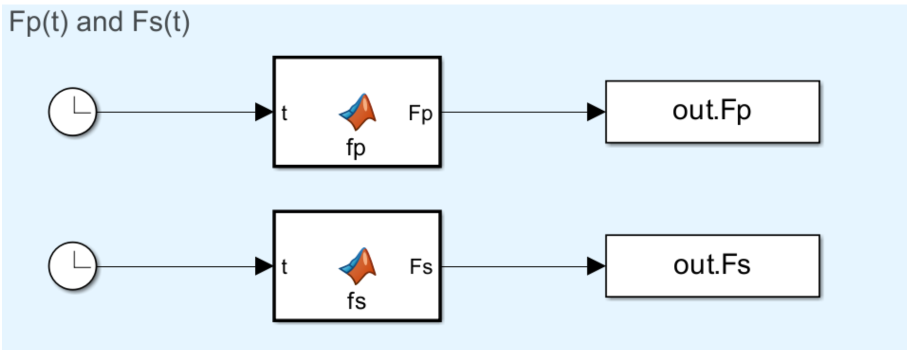

Figure 16: Simulink Implementation of $F_p(t)$ and $F_s(t)$

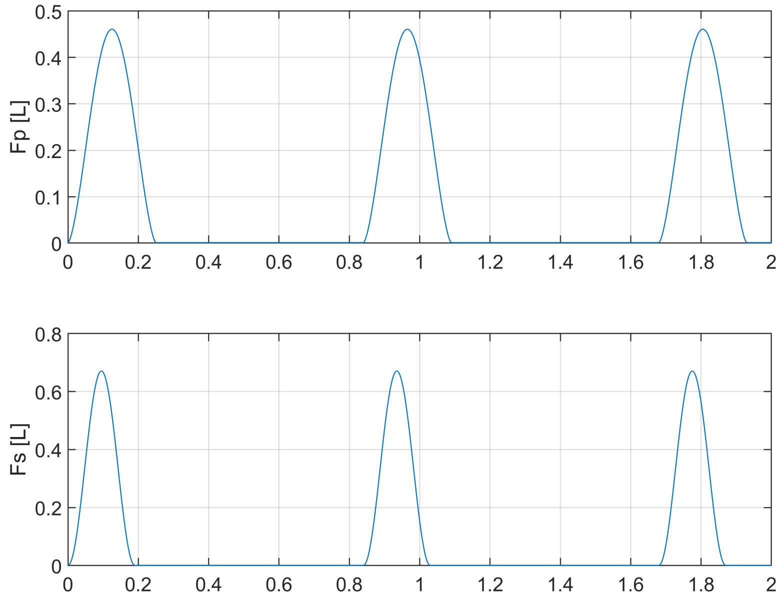

Figure 17: Periodic functions of $F_p(t)$ and $F_s(t)$

A.2.d

Calculating the integrals of $F_p(t)$ and $F_s(t)$ over a period:
Blood volume pumped to the body per beat (Fp): 0.064081
Blood volume pumped to the body per beat (Fs): 0.064490

A.3 Tissue Gas Exchange Model

A.3.a

Figure 18 shows the Simulink implementation of the gas exchange model for the tissues, corresponding MATLAB code can be found in the Appendix.

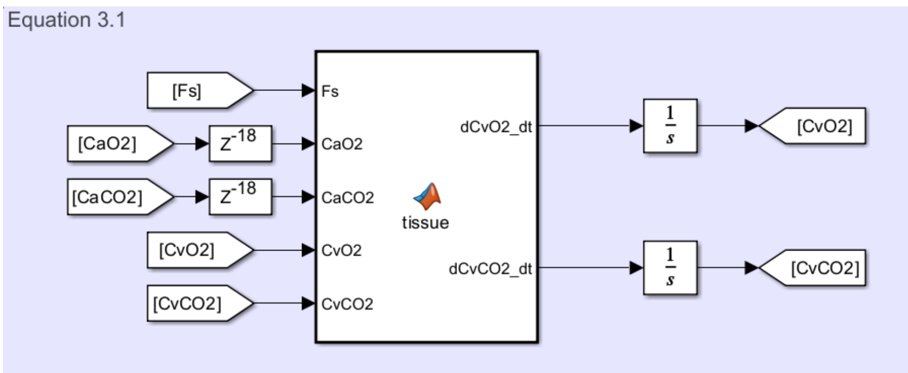

Figure 18: Simulink Implementation of Eq. 3.1

For a healthy subject, simulating the system for 5 minutes shown in Figure 19.

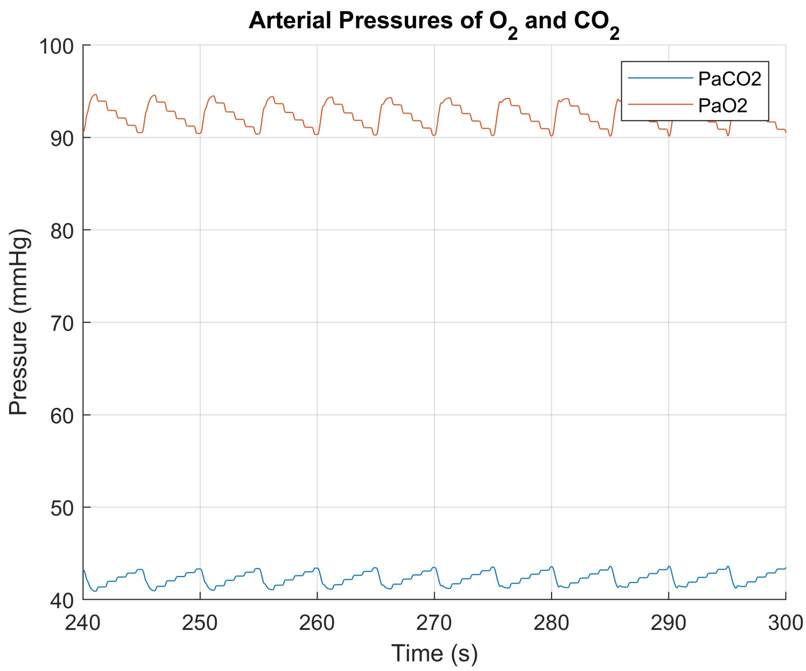

Figure 19: Arterial Pressures of O_2 and CO_2 with tissue exchange model implemented

For the same simulation, final values of the O_2 and CO_2 concentrations in the arterial and venous blood are measured as:

$$C_{aCO2} = 0.20$$

$$C_{aCO2} = 0.5$$

$$C_{vO2} = 0.155$$

$$C_{vCO2} = 0.53$$

A.3.b

To represent a healthy subject's exercise conditions, MR_{O_2} and MR_{CO_2} were increased by 50%. The simulation was then run for 20 minutes, and the results are shown in Figure 20. It can be observed that the partial pressure of O_2 has decreased from 93 $mmHg$ to around 65 $mmHg$; while the partial pressure of CO_2 increased from 45 $mmHg$ to around 68 $mmHg$. This represents a critical condition that is not expected to occur in a healthy subject. If this condition persists, it could harm the body in various ways. In a healthy subject, when such a situation arises, the body compensates by increasing the heart rate, the breathing frequency, and the tidal volume.

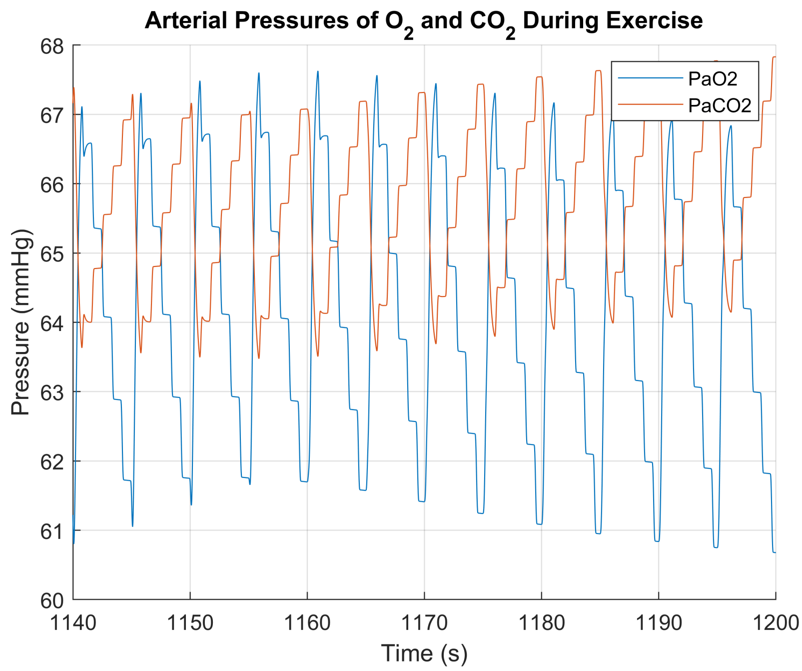

Figure 20: P_{aO_2} and P_{aCO_2} during exercise

A.3.c

To compensate the exercise, increase the tidal volume from 0.4 l to 0.65 l. The simulation results are shown in Figure 21. It can be observed that the pressures have returned to their normal ranges.

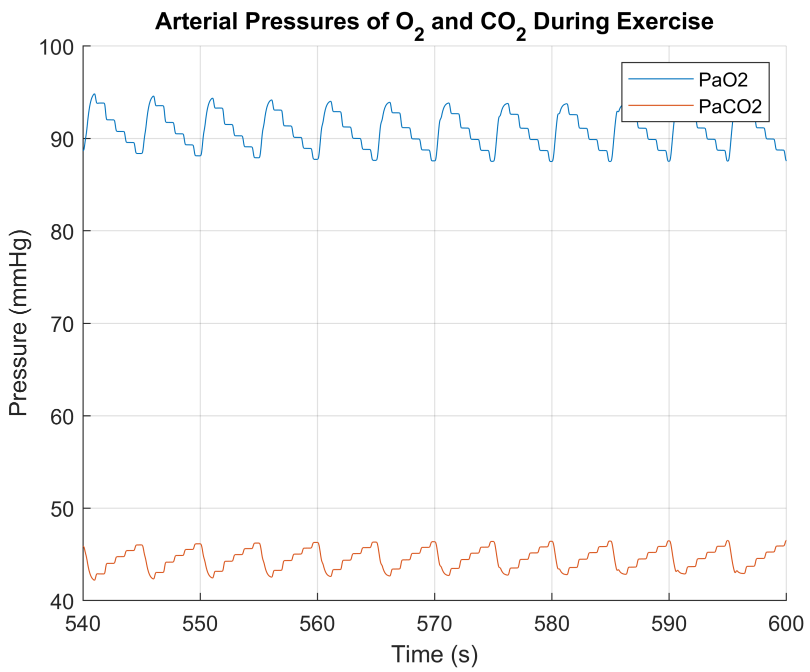

Figure 21: P_{aO_2} and P_{aCO_2} during exercise with increased tidal volume

A.3.d

For a diseased lung, some parameters are set as $k_{eff} = 0.6$ and $shunt = 16.5\%$. Simulating the system for an otherwise healthy person for 20 minutes results in Figure 22.

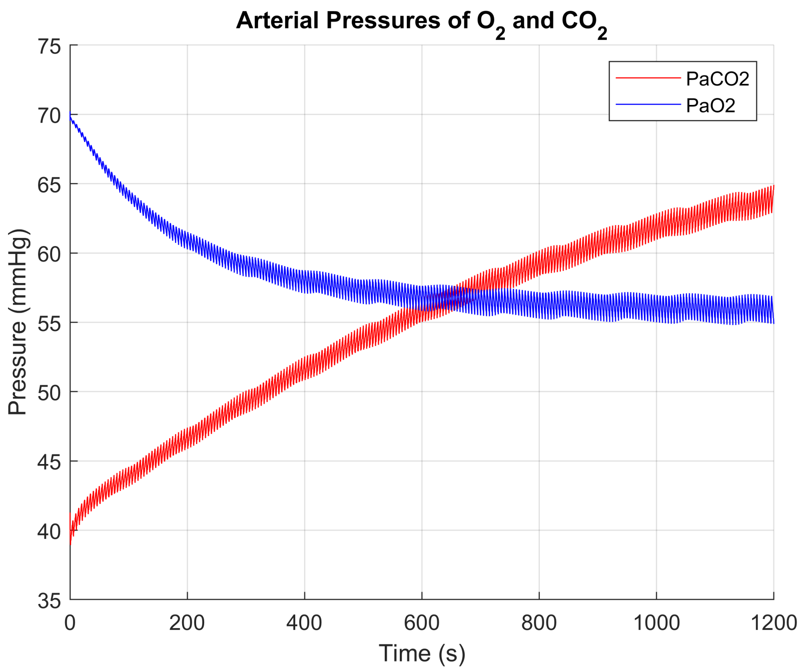

Figure 22: P_{aO_2} and P_{aCO_2} with a diseased lung

A.3.e

The subject is in need of supplementary oxygen, to achieve this, change the percentage of oxygen in the inhaled air to 70%, which was 21% before. Also, increase the tidal volume back to 0.65 l so that the increased oxygen shows its effect faster. Run the simulation for another 30 minutes results in Figure 23.

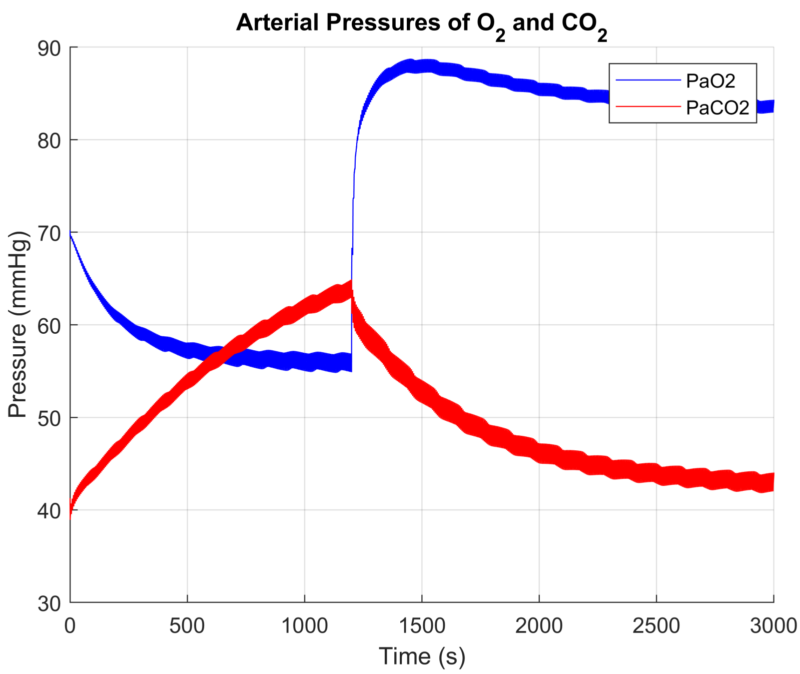

Figure 23: P_{aO2} and P_{aCO_2} with a diseased lung and 70% oxygen in the air

A.3.f

When the subject is a COVID-19 patient, its arterial pressures under normal circumstances are shown in Figure 24. In this case, arterial pressure of O_2 is low (below 60 mmHg). This situation is critical for the patient and it is in need of supplementary oxygen. Thus, to the same patient, when pure oxygen is supplied, its arterial pressures are shown in Figure 25. Even with the supplemented oxygen, the patient is still in critical condition, and the lungs are not capable of transmitting the oxygen to the blood. In such a case, a device called ECMO is used as an artificial lung, to supply the blood with oxygen.

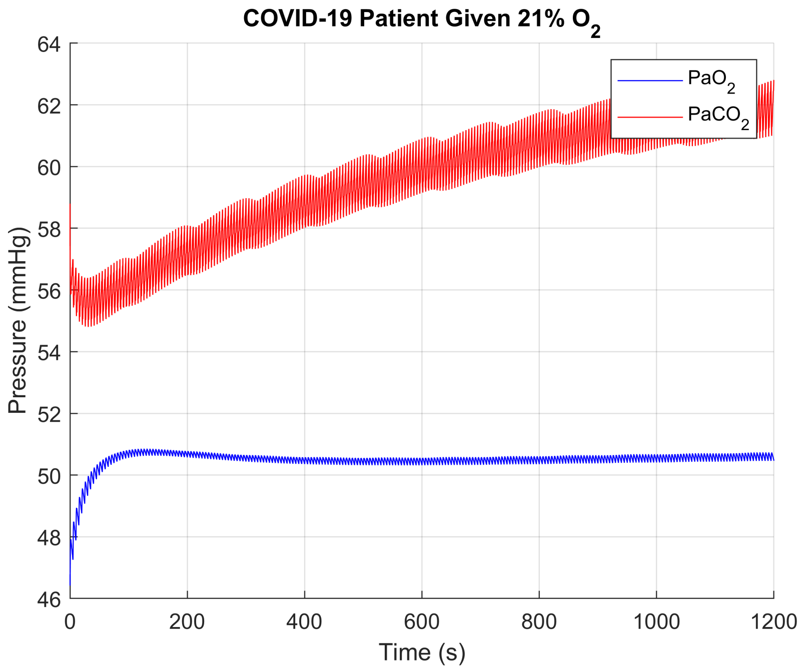

Figure 24: P_{aO_2} and P_{aCO_2} values of a COVID-19 patient when $F_{IO_2} = 0.21$

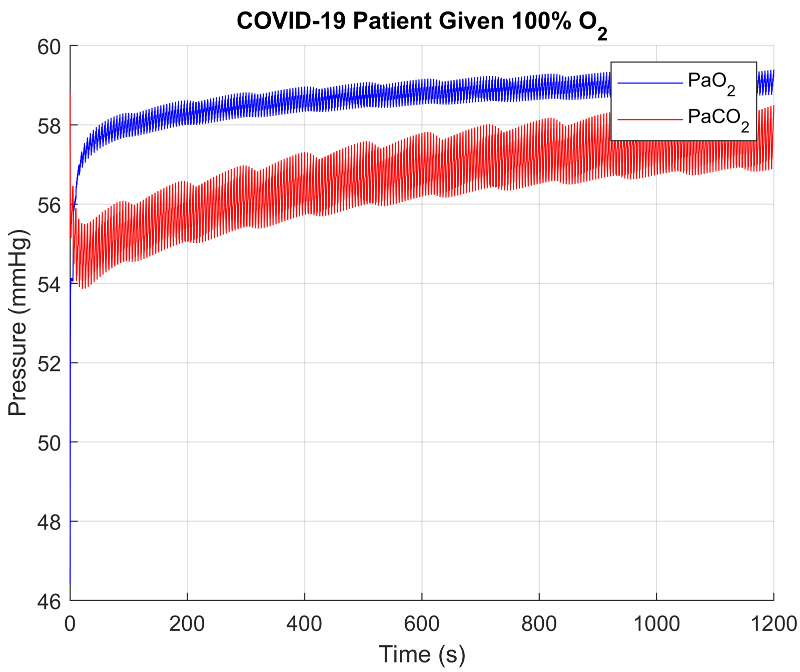

Figure 25: P_{aO_2} and P_{aCO_2} values of a COVID-19 patient when $F_{IO_2} = 1.00$

B EXTRA CORPOREAL MEMBRANE OXYGENATOR MODEL

B.1 Pump Model

B.1.a

Electrical model of the pump has been modeled in Simulink using Simscape library and is shown in Figure 26. The parameters are $R = 10\Omega$, $L = 10\ mH$, $k_{em} = 0.192\ Vs/rd$, $J_L = 8 \cdot 10^{-4}\ kg\,m^2$ and $N_g = 0.5$. This system was simulated with the maximum input of $V_{pump} = 48V$, the output angular velocity, ω_{out} is shown in Figure 27. As expected, at maximum input voltage, the output angular velocity also reaches its maximum value of 500 rad/s.

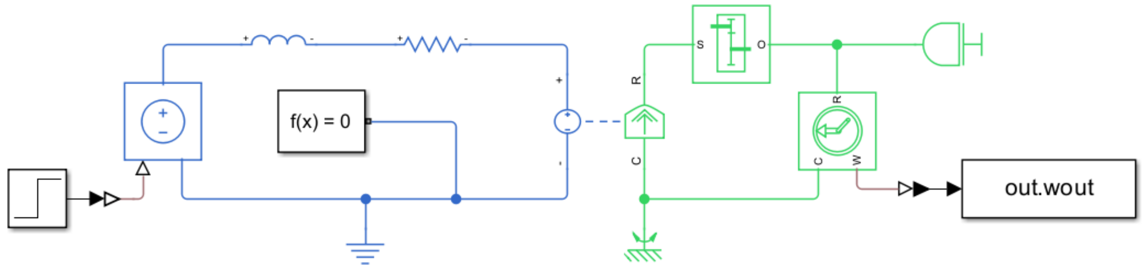

Figure 26: Electrical model of the pump

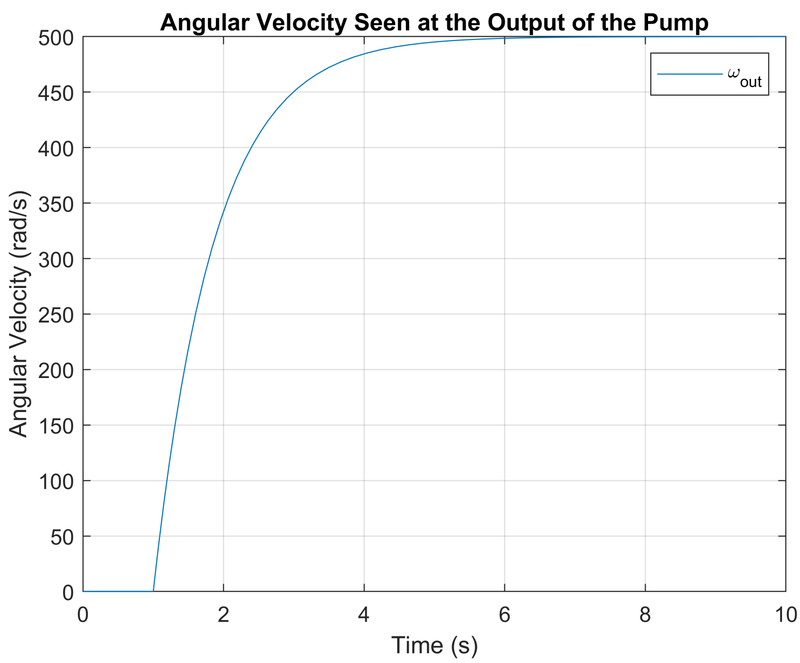

Figure 27: Angular velocity seen at the output of the pump

B.1.b

The relation between the shunt and the blood flow is modelled in Figure 28. Corresponding function codes can be found in the Appendix.

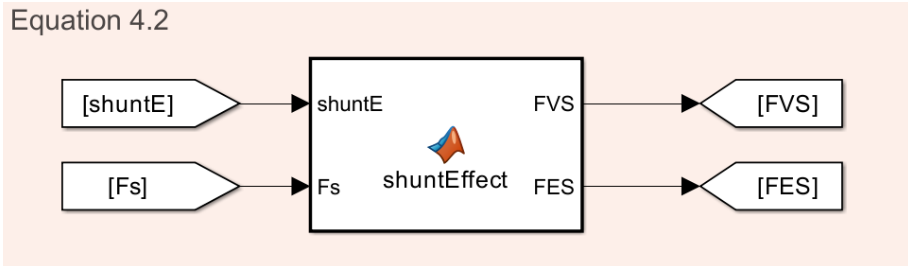

Figure 28: Simulink implementation of equation 4.2

The shunt of the ECMO is determined by the output angular velocity of the pump and defined to be between 0 and 1. Since ω_{out} is between 0 and 500, it can be defined that:

$$shuntEcmo = \frac{|\omega_{out}|}{500}$$

To control the shunt, a PI controller has been added to the previous circuit and is shown in Figure 29. With a reference of 0.5, the simulation result is shown in Figure 30.

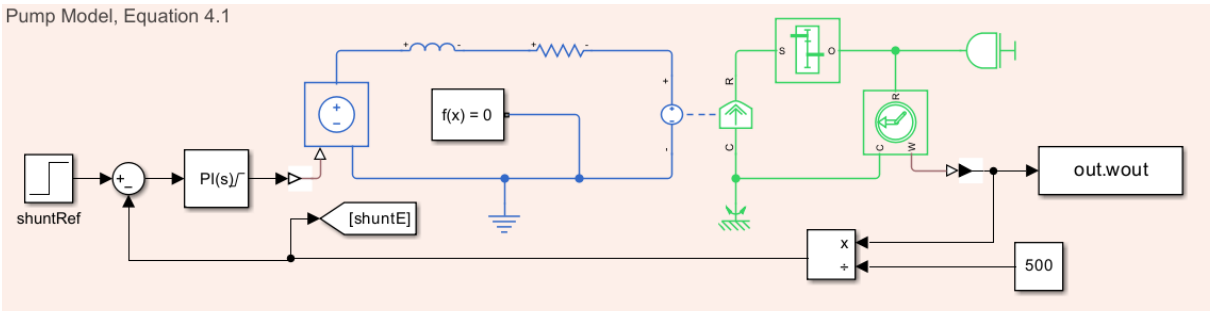

Figure 29: Pump model with PI controller

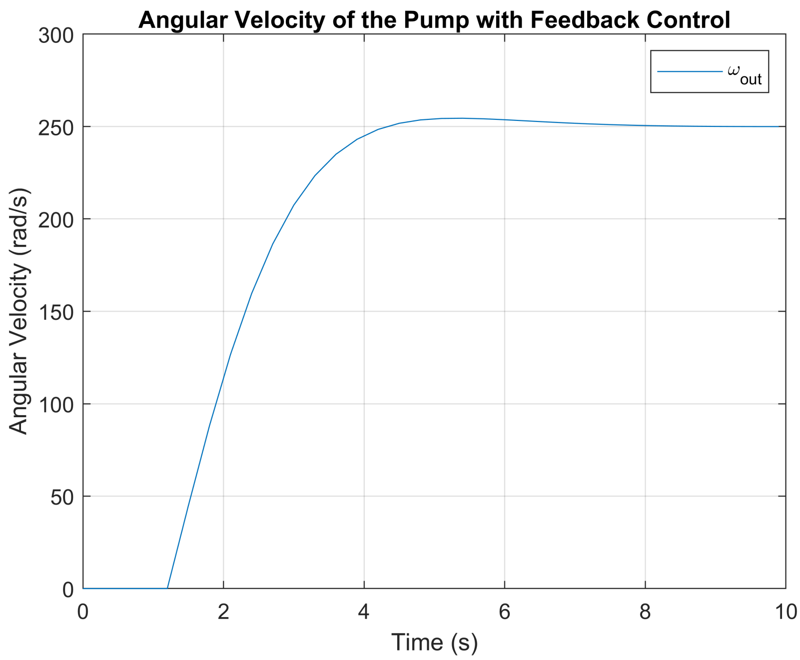

Figure 30: Pump output with PI Controller

B.2 Oxygenator Model

B.2.a

ECMO is supposed to behave like the lungs, so that until the lungs of the patient are able to work properly, ECMO can take the place of the patient's lungs. With lung-like equations, oxygenator can be modelled. Simulink implementations of the oxygenator are shown in the Figure 31 and 32. MATLAB Codes can be viewed at the Appendix.

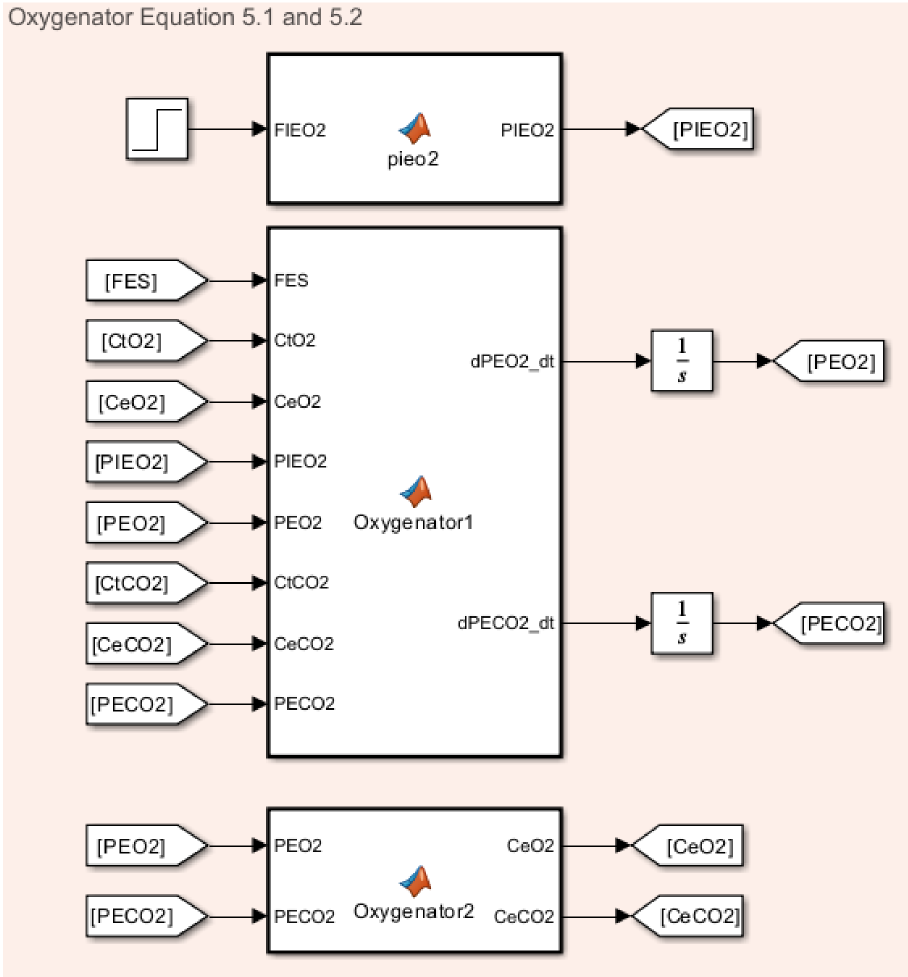

Figure 31: Simulink implementation of oxygenator defining equations

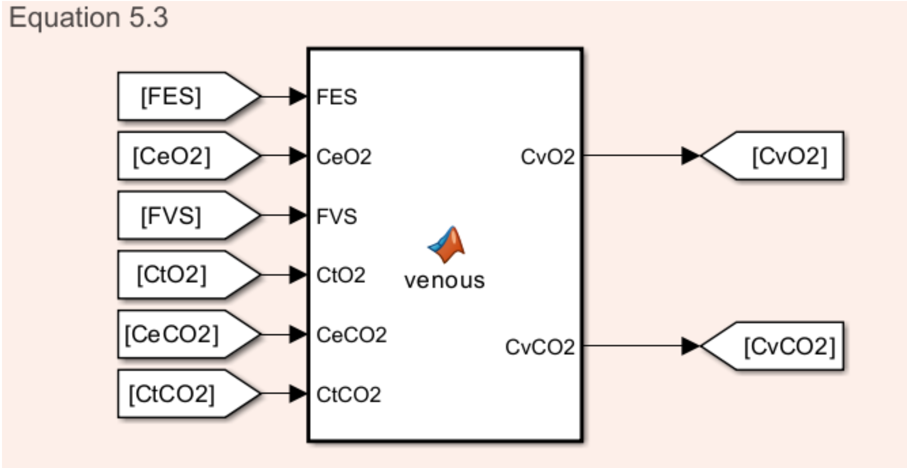

Figure 32: Simulink implementation of venous outputs with ECMO equations

B.2.b

The complete model of ECMO is now ready. Simulating the device with a COVID-19 patient, with the given ECMO parameters, and also with patient being supplied with air that has 50% oxygen, along with reduces tidal volume of 0.2 l via the ventilator, the resulting arterial pressures are shown in Figure 33.

$$V'_{ECMO} = 0.08$$

$$F_{IEO_2} = 90\%$$

$$shuntRef = 70\%$$

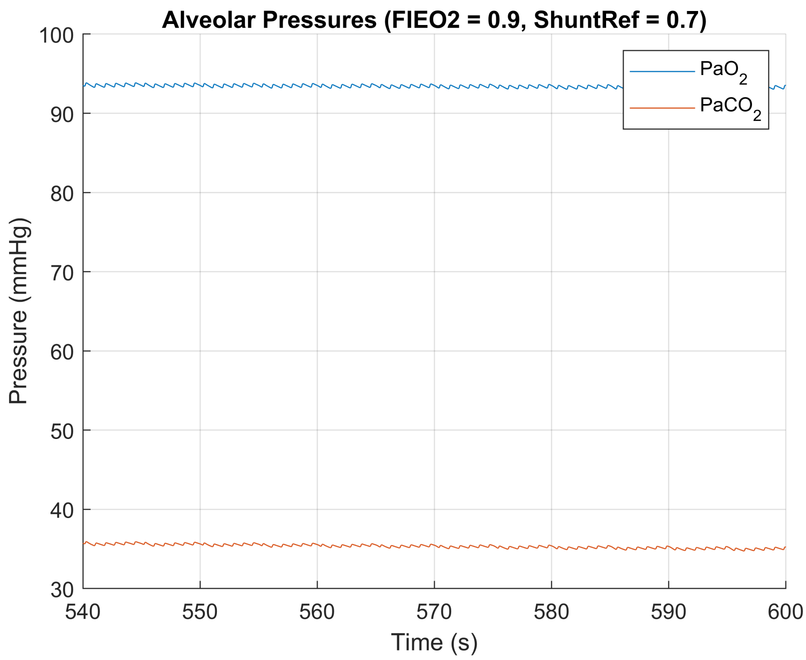

Figure 33: P_{aO_2} and P_{aCO_2} of a COVID-19 patient connected to ECMO

In the current situation, ECMO is working as intended with $P_{aO_2} \geq 80\ mmHg$ and $P_{aCO_2} \leq 40\ mmHg$.

B.2.c

To see the effects of F_{IEO_2} and $shuntRef$, one of them kept constant while the other is changed. Figure 34 shows the effect of $shuntRef$ when F_{IEO_2} is kept constant at 0.9. Increasing the shuntRef results in more blood to be processed by the ECMO. ECMO increases the P_{aO_2} and reduces the P_{aCO_2}.

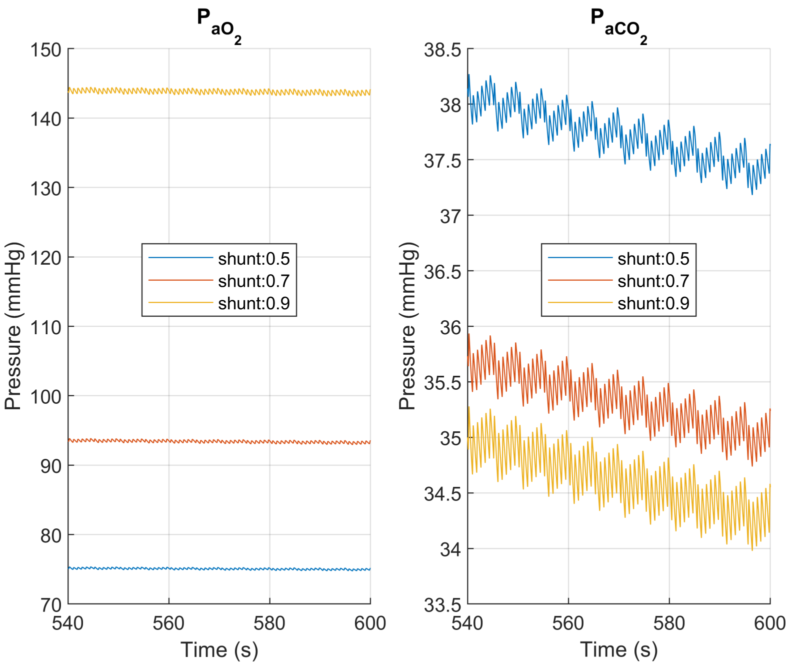

Figure 34: Effect of $shuntRef$ over the arterial pressures

Likewise, keeping $shuntRef$ constant at 0.7 and changing F_{IEO_2} results in the Figure 35. Increase in the F_{IEO_2} resulted in decrease of P_{aCO_2}. The level of decrease also reduced. Meanwhile, P_{aO_2} has increased, but in a similar manner, the level of increase is again reduced. So much that when F_{IO_2} is set to be 0.5 and 0.9, the output P_{aO_2} has almost remained the same.

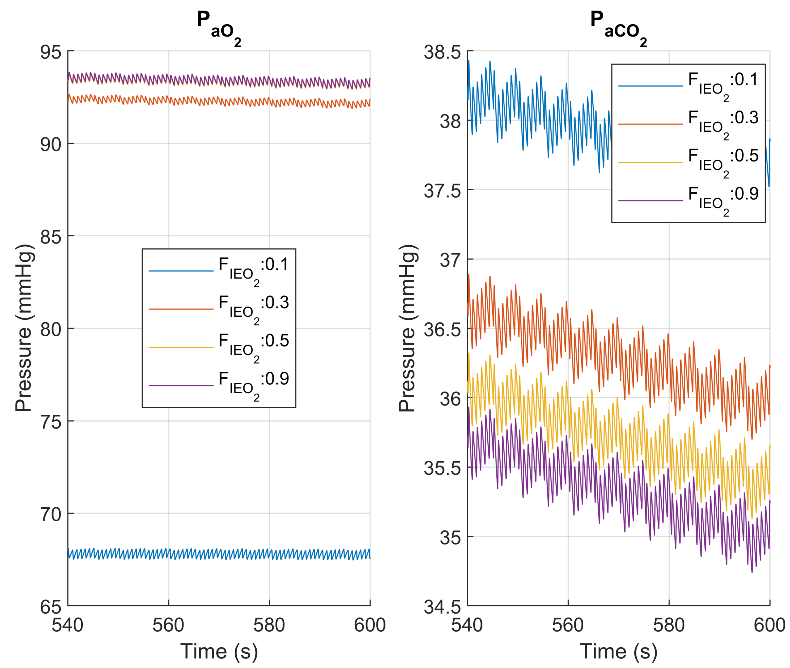

Figure 35: Effect of F_{IEO_2}over the arterial pressures

B.2.d

Designing a controller with trial and eventually resulted in the controller shown in Figure 37, which worked but the controller does not account for the fact of P_{aCO_2} as an input. Nevertheless, it still satisfies the conditions given as:

$$P_{aO2} = 80\ mmHg$$

$$P_{aCO2} \leq 40\ mmHg$$

Resulting arterial pressures are shown in Figure **??**. The PID constants for the F_{IEO_2} controller are: $K_P = 1$, $K_I = 0.1$ and $K_D = 0.2$. For the $shuntRef$ controller, the constants are $K_p = 10$, $K_I = 0.1$ and $K_D = 0.2$. Towards the steady state of the simulation, the control parameters have reached to the states shown in Figure 38.

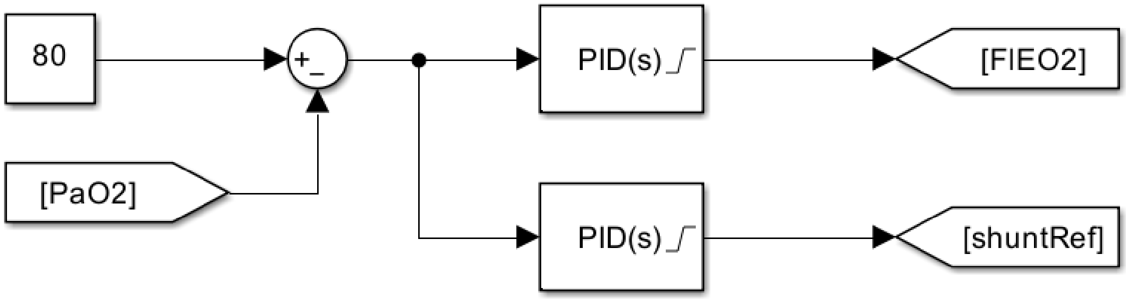

Figure 36: Controller for ECMO

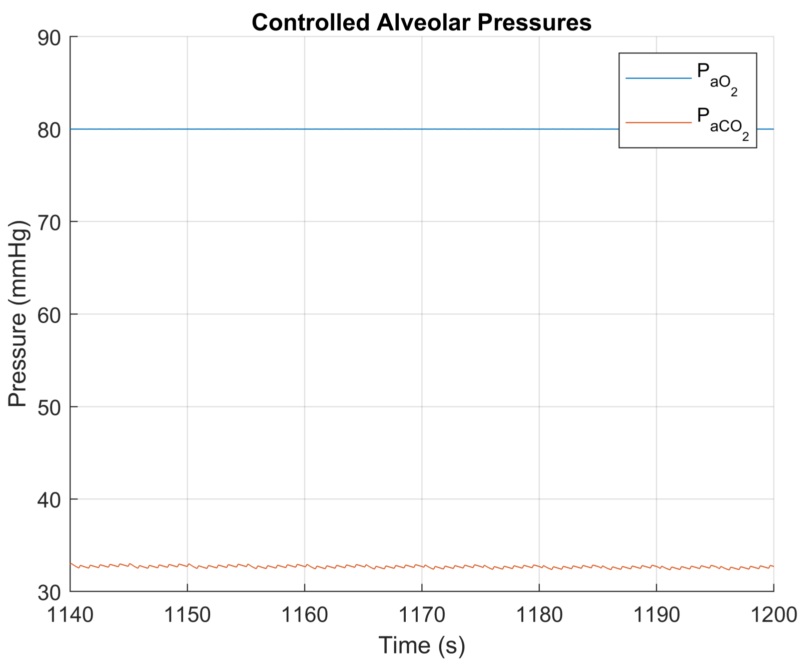

Figure 37: Arterial pressures controlled by PID controller

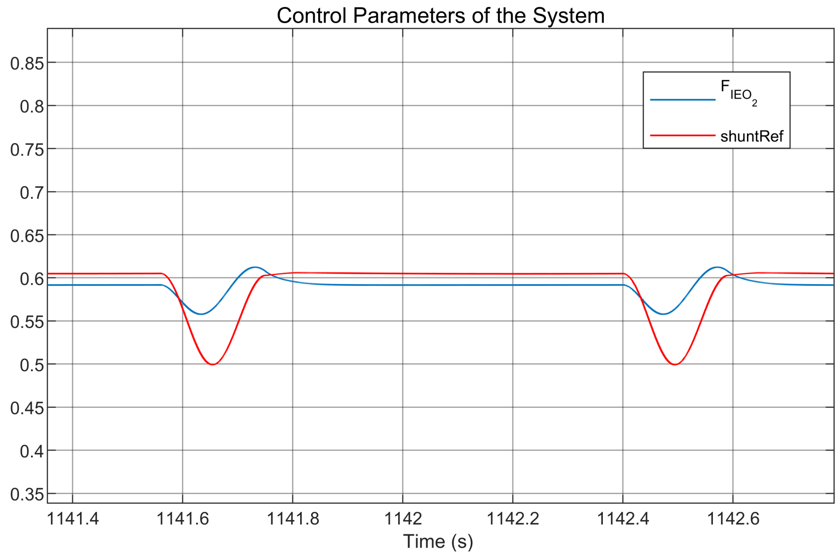

Figure 38: ECMO control variables at steady state with PID controller

B.2.e Comments on the Project

The term project proved to be a valuable learning experience for us as students, especially those who had not previously participated in control engineering or system modeling projects. While it was enjoyable overall, it was also challenging at times to troubleshoot errors and interpret the outcomes of various scenarios to evaluate the correctness of the simulation results. However, the difficulty we encountered contributed significantly to the project's educational value, as overcoming such challenges provided deeper insights and understanding.

One notable drawback, however, was the repetitive nature of the project. Initially, it resembled an engaging puzzle, but once the solution was identified, it became less stimulating and shifted towards repetitive practice without introducing new learning opportunities.

Additionally, the group project format tends to result in premature disengagement among team members. Often, one team member takes the lead and solves most of the challenges independently, finding the process both frustrating and rewarding. Once the solution is discovered, the project ceases to be a puzzle, as others can simply rely on the individual who has already determined the methods. This reduces opportunities for active participation and shared learning, as the focus shifts from exploration and discovery to mere repetition.

C APPENDIX

- Equations 2.1 and 2.2

```
function [dPdO2_dt, dPdCO2_dt] = partialPressures(VAprime, PAO2, PACO2, PdO2, PdCO2,
     PIO2, PICO2, VD)
    if VAprime > 0
        dPdO2_dt = VAprime * (PIO2 - PdO2) / VD;
        dPdCO2_dt = VAprime * (PICO2 - PdCO2) / VD;
    else
        dPdO2_dt = VAprime * (PdO2 - PAO2) / VD;
        dPdCO2_dt = VAprime * (PdCO2 - PACO2) / VD;
    end
end
```

- Equation 2.3

```
PIO2 = (Patm - Pws)*FIO2;   % mmHg
PICO2 = (Patm - Pws)*FICO2; % mmHg
```

- Equation 2.4 and 2.5

```
function [dPAO2_dt, dPACO2_dt] = gasExchange(CvO2, CvCO2, PAO2, PACO2, VLO2, VLCO2,
    CAO2, CACO2, Fpp, Vprime, PdO2, PdCO2)
    if Vprime > 0
        dPAO2_dt  = ( 863*Fpp*(CvO2-CAO2)   + Vprime*(PdO2-PAO2))    /VLO2;
        dPACO2_dt = ( 863*Fpp*(CvCO2-CACO2) + Vprime*(PdCO2-PACO2))  /VLCO2;
    else
        dPAO2_dt  = (863*Fpp*(CvO2-CAO2))   /VLO2;
        dPACO2_dt = (863*Fpp*(CvCO2-CACO2)) /VLCO2;
    end
end
```

- Equation 2.6

```
function [VLO2, VLCO2] = lungStorage(V, VL0)
    VLO2 = VL0 + V;
    VLCO2 = 1.28 * VLO2;
end
```

- Equation 2.8 and 2.9

```
function [CAO2, CACO2] = gasConsentration(K2, beta1, beta2, K1, alpha1, alpha2,
    CsatO2, CsatCO2, h1, h2, PAO2, PACO2)
    CAO2 = CsatO2 * ( ( PAO2 / K1 * ( 1 + beta1*PACO2 )/( 1 + alpha1*PACO2 ) ) ^(1/
        h1) ...
               / (1 + ( PAO2 / K1 * ( 1 + beta1*PACO2 )/( 1 + alpha1*PACO2 ) ) ^(1/
                   h1)));

    CACO2 = CsatCO2 * ( ( PACO2 / K2 * ( 1 + beta2*PAO2 )/( 1 + alpha2*PAO2 ) ) ^(1/
        h2) ...
               / (1 +   ( PACO2 / K2 * ( 1 + beta2*PAO2 )/( 1 + alpha2*PAO2 ) ) ^(1/
                   h2)));
end
```

- Equation 2.7

```
Fp = 0.075;                 % l
Fpp = (1-shunt)*Fp;         % l
```

- Equation 2.10

```
function [CaO2, CaCO2] = arterialConcentrations(CAO2, CACO2, CvO2, CvCO2, Fpp, Fps)
   CaO2 = (Fpp*CAO2 + Fps*CvO2)/(Fpp + Fps);
   CaCO2 = (Fpp*CACO2 + Fps*CvCO2)/(Fpp + Fps);
end
```

- Equation 2.11

```
function PaCO2  = PaCO2(CaCO2, PaO2, K2, beta2, alpha2, h2, CsatCO2)
    PaCO2 = K2 * (1 + alpha2*PaO2)  / (1 + beta2*PaO2)  * ( CaCO2/(CsatCO2 - CaCO2))
        ^h2;
end
```

```
function PaO2 = PaO2(CaO2, alpha1, beta1, K1, PaCO2, h1, CsatO2)
    PaO2 =  K1 * (1 + alpha1*PaCO2) / (1 + beta1*PaCO2) * ( CaO2 /(CsatO2 -  CaO2 ))
        ^h1;
end
```

- Periodic Functions of $F_p(t)$ and $F_s(t)$

```
function Fp = fp(t)
    u = mod(t, 0.84);
    if (0 < u) && (u < 0.25)
        Fp = (1+ 460*(sin(pi*u/0.25))^1.55) / 1000;
    else
        Fp = 1 / 1000;
    end
end
```

```
function Fs = fs(t)
    u = mod(t, 0.84);  % Sawtooth wave (0 to period)
    if (0 < u) && (u < 0.19)
        Fs = (1 + 670*(sin(pi*u/0.19))^2) / 1000;
    else
        Fs = 1/1000;
    end
end
```

- Equation 3.1

```
function [dCtO2_dt, dCtCO2_dt] = tissue(Fs, CaO2, CaCO2, CtO2, CtCO2, MRO2, MRCO2,
    VTO2, VTCO2)
    dCtO2_dt = (-MRO2 + Fs * (CaO2 - CtO2)) / VTO2;
    dCtCO2_dt = (MRCO2 + Fs * (CaCO2 - CtCO2)) /VTCO2;
end
```

- Equation 4.2

```
function [FVS, FES] = shuntEffect(shuntE, Fs)
    FVS = (1-shuntE)*Fs;
    FES = shuntE*Fs;
end
```

- Equation 5.1 and 5.2

```
function PIEO2 = pieo2(Pws, FIEO2, Patm)
    PIEO2 = (Patm - Pws)*FIEO2;
end
```

```
function [dPEO2_dt, dPECO2_dt] = Oxygenator1(FES, CtO2, CeO2, PIEO2, PEO2, CtCO2,
    CeCO2, PECO2, VprimeEcmo, VEO2, VECO2)
    dPEO2_dt = (863*FES*(CtO2 - CeO2) + VprimeEcmo*(PIEO2 - PEO2))/VEO2;
    dPECO2_dt = (863*FES*(CtCO2 - CeCO2) - VprimeEcmo*PECO2)/VECO2;
end
```

```
function [CeO2, CeCO2] = Oxygenator2(PEO2, PECO2, beta1, beta2, K1, K2, h1, h2,
    alpha1, alpha2, CsatO2, CsatCO2)
    CeO2 = CsatO2 * (PEO2/K1*(1+beta1*PECO2)/(1+alpha1*PECO2))^(1/h1) / (1 + (PEO2/
        K1*(1+beta1*PECO2)/(1+alpha1*PECO2))^(1/h1));
    CeCO2 = CsatCO2 * (PECO2/K2*(1+beta2*PEO2)/(1+alpha2*PEO2))^(1/h2) / (1 + (PECO2
        /K2*(1+beta2*PEO2)/(1+alpha2*PEO2))^(1/h2));
end
```

- Equation 5.3

```
function [CvO2, CvCO2] = venous(FES, CeO2, FVS, CtO2, CeCO2, CtCO2)
    CvO2 = (FES*CeO2 + FVS*CtO2) / (FES + FVS);
    CvCO2 = (FES*CeCO2 + FVS*CtCO2) / (FES + FVS);
end
```

D REFERENCES

[1] Republic of Turkey Ministry of Health. (2021). ECMO guide. https://hsgm.saglik.gov.tr/depo/birimler/bulasici-hastaliklar-ve-erken-uyari-db/Dokumanlar/Rehberler/20211109163159_ECMO_3.pdf.