THE CAPILLARY-CENTRIC MODEL OF CARDIAC COUPLING-AS-THERMODYNAMICS

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By
Andrew James David Taylor

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Electrical and Computer Engineering

November 2015

Fargo, North Dakota
The Capillary-centric Model of Cardiac Coupling-as-thermodynamics

By

Andrew James David Taylor

The Supervisory Committee certifies that this disquisition complies with North Dakota State University’s regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dan L. Ewert
Chair

Jacob S. Glower

Mark J. Schroeder

Kyle J. Hackney

Approved:

11/5/15
Date

Scott C. Smith
Department Chair
ABSTRACT

Models of ventricular-arterial coupling (VAC) have historically described the heart as a function of its energetic interaction with the arterial system. However, these models either represent the dynamic, adaptive cardiovascular system (CVS) in isolation or sacrifice cardiac mechanics to use simplified, time-averaged values across the cardiac cycle. In this thesis a facsimile CVS is constructed that characterizes ventricular-arterial interactions with intact cardiac mechanics as a function of whole-body thermo-fluid homeostatic regulation. Simulation results indicate proportional-integral (PI) control of heart rate and arterial resistance is conditionally sufficient to maintain body temperature during square-wave exercise, but further elements may be required to mimic genuine physiological responses. These simulations of the primitive model lay the framework of capillary-centric VAC through the perspective of coupling-as-thermodynamics.
ACKNOWLEDGEMENTS

I would like to thank my classmates as well as the faculty and staff of the department for their continued support. I would like to especially thank my wonderful committee members Drs. Dan Ewert, Jake Glower, Kyle Hackney and Mark Schroeder for their insights, encouragements and criticisms. Their professionalism (and lack thereof!) has made graduate school a merry adventure.

I would also like to give my most heartfelt thanks to my friends and family—Amy, Ashley, Brian, Cassie, Charla, Dan, Dave, Dona, Doug, Jim, John, Liz, Lyle, Martin, Martin, Missy, Molly, Regina, Richard, Sylvia, Tim and Toby—to whom I owe my success and blame my character. Finally, above all: thank you, Mom!
TABLE OF CONTENTS

ABSTRACT ........................................................................................................ iii

ACKNOWLEDGEMENTS .................................................................................... iv

LIST OF TABLES ............................................................................................... vi

LIST OF FIGURES ............................................................................................. vii

LIST OF EQUATIONS ....................................................................................... viii

INTRODUCTION ................................................................................................. 1

BACKGROUND .................................................................................................... 2

METHODS .......................................................................................................... 7

RESULTS ............................................................................................................. 17

DISCUSSION ....................................................................................................... 32

CONCLUSION ..................................................................................................... 34

REFERENCES ..................................................................................................... 35

APPENDIX. MATLAB CODE ............................................................................. 41
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Initial conditions for the open-loop simulation.</td>
<td>13.</td>
</tr>
<tr>
<td>2.</td>
<td>Percent differences in settling time for selected intervals of a quintet.</td>
<td>20.</td>
</tr>
<tr>
<td>3.</td>
<td>Percent differences in settling time for selected intervals of a quartet.</td>
<td>22.</td>
</tr>
<tr>
<td>4.</td>
<td>Percent differences in settling time for selected intervals of a triplet.</td>
<td>25.</td>
</tr>
<tr>
<td>5.</td>
<td>Sweat production, final temperature and correlation coefficient between these variables for each trial of the second data set.</td>
<td>31.</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Expanded windkessel model for the “block of humanity.”</td>
<td>7</td>
</tr>
<tr>
<td>2.</td>
<td>Normalized time-varying elastance of the left ventricle.</td>
<td>9</td>
</tr>
<tr>
<td>3.</td>
<td>Pressure, flow and volume in the left ventricle.</td>
<td>14</td>
</tr>
<tr>
<td>4.</td>
<td>Program flow chart.</td>
<td>15</td>
</tr>
<tr>
<td>5.</td>
<td>PI controller used in the program.</td>
<td>16</td>
</tr>
<tr>
<td>6.</td>
<td>Settling time as a function of proportional control of heart rate.</td>
<td>19</td>
</tr>
<tr>
<td>7.</td>
<td>Settling time as a function of integral control of heart rate.</td>
<td>23</td>
</tr>
<tr>
<td>8.</td>
<td>Settling time as a function of integral control of arterial resistance.</td>
<td>26</td>
</tr>
<tr>
<td>9.</td>
<td>Time courses for HR $K_p = 10$, HR $K_i = 1$ and RA $K_p = 10$.</td>
<td>28</td>
</tr>
<tr>
<td>10.</td>
<td>Time courses for HR $K_p = 10$, HR $K_i = 10$ and RA $K_p = 50$.</td>
<td>29</td>
</tr>
<tr>
<td>11.</td>
<td>Temperature comparison between reported data and model.</td>
<td>30</td>
</tr>
</tbody>
</table>
# LIST OF EQUATIONS

<table>
<thead>
<tr>
<th>Equation</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Net filtration pressure.</td>
<td>9</td>
</tr>
<tr>
<td>2. Mean capillary pressure.</td>
<td>9</td>
</tr>
<tr>
<td>3. Arterial-venous oxygen difference.</td>
<td>10</td>
</tr>
<tr>
<td>4. Sweat filtration flow rate.</td>
<td>11</td>
</tr>
<tr>
<td>5. Rate of vaporization of latent heat.</td>
<td>11</td>
</tr>
<tr>
<td>6. Interbeat temperature change.</td>
<td>11</td>
</tr>
<tr>
<td>7. End of beat body temperature.</td>
<td>11</td>
</tr>
<tr>
<td>8. End of beat change in heart rate.</td>
<td>11</td>
</tr>
<tr>
<td>9. End of beat change in arterial resistance.</td>
<td>11</td>
</tr>
</tbody>
</table>
INTRODUCTION

Ventricular-arterial coupling (VAC) describes the heart as a function of its interaction with the arterial system and is quintessential to exploring the behavior of the cardiovascular system (CVS). Current models provide significant insight into many important aspects of the CVS such as energetic expenditure, ventricular loading sequences and pulse-wave velocities in the proximal aorta, all of which play an integral role in discovering or describing many CVS dysfunctions. Ultimately, treatments to these dysfunctions depend on understanding the role VAC plays in the CVS.

However, current models represent the dynamic, adaptive CVS in isolation and fail to describe coupling in terms of system level interactions—namely regulating temperature, balancing fluids at the capillary and supplying adequate metaboloids to the tissues [10], [25], [45]. By contrast, models of integrated physiology can illustrate these three functions well even over the course of simulation-years, but their effectiveness comes at the expense of cardiac mechanic transparency [1], [22], [29], [30], [35]. Nonetheless, a balance between usefulness and complexity may be achieved by applying a proportional-integral (PI) controller to emulate lumped body parameters and processes [12], [17], [41], [47].

In this thesis a facsimile CVS is constructed that characterizes ventricular-arterial interactions with intact cardiac mechanics as a function of whole-body thermo-fluid homeostatic regulation, shifting the source-load coupling paradigm of the left ventricle and arteries to the capillary and tissues. The model is then used to explore the question of whether PI control of heart rate and arterial resistance is sufficient to maintain body temperature during square-wave exercise under the assumptions that fluid balance is maintained at the capillary level and that there is adequate supply of metaboloids to the tissues. These simulations of the primitive model lay the framework of capillary-centric VAC through the perspective of coupling-as-thermodynamics.
BACKGROUND

Ventricular-arterial coupling

Traditionally, investigations of ventricular-arterial coupling have worked towards characterizing the interplay between the left ventricle of the heart and the arterial system [10], [25], [45]. This coupling is most often described as a source-load problem, where the heart is modeled as a power supplier and the arterial system as a power dissipater, allowing for the application of familiar engineering tools to a complex biological structure. The CVS has historically been analyzed in the pressure-volume (PV) domain, enabling indices for cardiac efficiency and energetics to be classified by analogy to a heat engine. PV domain analysis was born from experiments under A. C. Guyton and his colleagues in the 1950s that demonstrated cardiac output was a function of right atrial pressure [14], [15], [16]. Dissection of the CVS continued both literally and metaphorically for another thirty years as the advent of personal computing gave rise to complicated mathematical models.

These models led directly to the foundation of ventricular-arterial coupling. Building on a concept published in 1980 by Piene that described right ventricular mechanical properties, Sunagawa et al. derived an equation that bridged the dimensional gap left open by Piene [33], [45]. In Piene’s model, the ventricle was characterized by its PV relationship while the pulmonary arterial system was defined by its impedance. Conversely, the work of Sunagawa et al. focused on the left ventricle and was predicated on treating both ventricle and arterial system as elastic elements. This formulation led directly to the creation of effective arterial elastance, the first “coupling index” used in describing VAC [46].

Effective arterial elastance, or $E_A$, has remained a popular method for describing coupling because of its intuitiveness—its units are directly translatable to indices of energy utilization and is commonly paired with the end-systolic elastance $E_{ES}$ [5], [6], [7], [10], [11], [25], [26], [37], [40]. By using the Jacobian expression of efficiency it was predicted that at
$E_A/E_{ES} = 1$ maximum stroke work had been reached, whereas at $E_A/E_{ES} = 0.5$ maximum efficiency would be achieved [48]. However, at any given time it is more likely that the cardiovascular system is controlled somewhere between these two extremes, and it has been shown that this coupling ratio $E_A/E_{ES}$ has a normal range between 0.62 and 0.82 [11]. Furthermore, this same work reveals that both stroke work and efficiency are maintained at optimal values across a wide range of coupling ratios.

Although useful as a non-invasive measurement, these findings showed $E_A$ and its couple $E_{ES}$ have several limitations to their usefulness as diagnostic tools. Additionally, the accuracy of $E_A$ is highly susceptible to errors in the pulsatile load characteristics of the ejected blood, since in its most common form ($R/T$) the index is averaged over a whole beat. This is very apparent in studies in which steady beats are not observed, such as with a venal caval occlusion where the pulsatile component of cardiac output changes with respect to time. Furthermore, heart rate has a strong influence on $E_A$ and in those studies using paced specimens $E_A$ is not preserved [40], [47], [52].

To address these issues, research turned towards the pressure-flow (PQ) domain have become increasingly more common. The two primary failings of PV analysis that PQ assessments attempt to rectify are its disregard for the loading sequence that takes place during a cardiac cycle and the PV-domain emphasis on energetic “optimization” [2], [3], [10], [23], [31], [42], [46], [44], [49], [50]. Time-domain analysis has been an important step forward in the characterization of cardiac coupling.

While PV analyses wash out inter- and intrabeat mechanics, PQ analyses can discern not only the ventricular loading sequence but also arterial load, pulse wave velocity, aortic and arterial stiffness and others [9], [10], [28]. Essentially, it brings cardiac coupling into the demesne of time-domain reflectometry (TDR), a technique for analyzing electrical transmission lines that is commonly used for characterizing commercial coaxial cable lines. Incorporating TDR into measures of the cardiovascular system offers extensive insight into its organization, and can even be used to calibrate therapies for pacing and CVS rehabilitation.
For example, in the 60 years that separate ages 25 and 85, the pulse wave velocity down the aorta doubles, so using time-domain analysis can improve quality of life in patients using cardiac resynchronization therapy by tuning the pulse wave generated from the assist device such that its reflecting waves return at a more opportune time, i.e. after ejection.

This same idea can be applied to coupling: by measuring the generated and reflected waves, an index of workload and efficiency can be produced that characterizes how well the left ventricle ejects blood into the proximal aorta. As conduction speeds increase, reflected waves will return to the aortic root more and more quickly, which corresponds to less volume ejected due to higher afterload. Although this phenomenon can be seen in the PV domain, its cause is much more apparent in the PQ domain. Furthermore, there is evidence that heart failure in the presence of elevated afterload is actually a symptom caused by the impact these early-returning wave reflections have on the loading sequence in the ventricle, which, as mentioned before, is neglected in the PV domain [8], [19], [20], [27], [51]. However, even as insightful as PQ domain analysis can be, as a system-level model it still fails to describe many facets of cardiovascular dysfunction such as those from the metabolic syndrome or depression, both of which can have significant implications for coupling.

**Models of integrated physiology**

Although VAC had been discussed only since the mid-1980s, a complete model of cardiovascular regulation was constructed well before in 1972 [17]. Published by Guyton et al. following a seven year effort, the model was the first to define coupling at the systems level, an innovation that became the prime antecedent to the field of integrative physiology [22]. Resembling an integrated circuit schematic, the aptly named model of “overall regulation of circulation” ramified the circulatory system into 18 different subdivisions that included both nervous and endocrine components. While each subdivision has a specific purpose, everything in the model is connected via feedback such that each physiological module is dependent, implicitly or explicitly, upon each and every other one.
Initially encapsulated as a system of equations, Guyton’s model was translated in 1983 to the digital domain by a team that included one of the original authors. Dubbed “Human”, this physiome program contained only the 150 variables from the original model, but since then has been transformed into a free, online computational analysis package termed Quantitative Circulatory Physiology (QCP) which has itself been succeeded by HumMod. QCP kept the core of Guyton’s original model, but expanded the integrated physiome to become one of the most complete since Guyton’s original production and its successor HumMod includes over 4000 variables [1], [22]. Like its predecessors, HumMod features all the feedback loops that governed the original architecture and has been used to simulate pathophysiology [22], [29].

While Guyton’s model and its descendants are incredibly thorough and complex, they are all founded on a framework that emphasizes simulation of chronic regulation. By using time-averaged values for inputs and outputs not only reduces the intricacy of the model to a manageable level but also allows the models to be computationally feasible over a simulation time-scale of years. Nonetheless, employing time-averaged variables has the deleterious side effect of washing out intrabeat mechanics; thus, just as with PV analysis, these models may omit critical information [35].

**Classic control theory**

The foundation of VAC is predicated upon the cardiovascular system’s response to stimulus. Feedback to the CVS is mediated through a multitude of methods ranging from the expression of NO₂ by red blood cells at the arteriole to neurohumoral activation of adrenergic receptors to the length of the sarcomere prior to ejection. Each of these examples are like an individual tuning knob on a vast physiological stereo mixer that contrives the body electric. However, the CVS is able to adapt to changes in each of these tuning knobs through its complex and adaptive feedback network.
The concept of feedback is an integral component of classic control theory, which compares the actual value of an output to its expected value and adjusts the system’s input(s) to compensate. For instance, weightlifting causes some of the muscle fibers to become damaged so the body compensates by rebuilding the muscle with even more fibers, increasing strength. Similarly, when the heart needs to pump harder due to elevated blood pressure, it too increases in size. However, while the hypertrophy in skeletal muscle is generally positive, hypertrophy of cardiac muscle is generally negative: the larger the ventricular wall is, the less efficient it is at pumping blood, thus requiring more energy to perform the same task. This can potentially lead to a positive-feedback loop, where the cardiac muscle can’t meet the pumping needs of the body and remodels itself—increasing in size—which in turn reduces its ability to pump blood. As predicted by control theory, this instability leads to the collapse of the system.
METHODS

Several assumptions were made to facilitate the development of the model in the interests of reducing it to a manageable scale. First, the human body was simplified to a “block of humanity,” which represents a lumped parameter model of the body with a lumped capillary, artery, vein and heart as shown in Figure 1. Additionally, this model assumes that skin temperature is equal to core temperature both at rest and during stress, i.e. the human body has an infinite conductance and no ability to redistribute blood flow between the core and the periphery, although this mechanism should be noted as being a significant determinant in the body’s ability to cool or warm itself [37]. It is also assumed that fluid losses are replaced on a time-scale that does not affect blood volume significantly.

Figure 1. Expanded windkessel model for the “block of humanity.” Four categories of heat pass through this control surface: $Q_L$, the latent heat of vaporization of sweat; $C$, convective heat, such as wind; $R$, heat due to radiation, such as incident sunlight; and $W$, the external work produced by the body upon another object.

In the block of humanity, $C_A$ and $R_A$ represent the arterial compliance and resistance respectively, and similarly $C_V$ and $R_V$ represent the venous compliance and resistance. $R_C$ is the capillary resistance. $AV$ and $MV$ denote the aortic and mitral valves, while the left ventricle is represented by a variable capacitor $E$ which produces the inverse elastance waveform [43].
The thick red line encapsulating the austere cardiovascular system denotes the control surface of the model—the block of humanity’s skin. Across this boundary four types of energy are catalogued: the latent heat $Q_L$, which is generated by the evaporation of sweat, the convective heat $C$, which occurs when moving air carries away heat from the skin’s surface, the radiative heat $R$, representing the skin’s absorption of radiation, and the external work $W$ produced by the body. Note, however, that simulation conditions reduce the variables $C$ and $R$ to zero.

The remainder of this section is broken down into the constituent components organized by anatomy and function. These sections are arranged as such: 1) left ventricle; 2) arterial system; 3) control system; 4) simulation parameters; 5) model validation. Figures and flow charts detailing the order of operation within the model can be found at the end of the section on pages 15 and 16.

**Left ventricle**

Initial conditions along with myocardial and arterial properties are fed into an ODE solver (Figure 4) that takes a normalized elastance waveform (Figure 2) and conforms it to a given $E_{\text{min}}$, $E_{\text{max}}$ and heart rate. This new elastance waveform is used in conjunction with given arterial properties to simultaneously calculate nine ventricular and arterial parameters over a single cardiac cycle: LV in- and out-flow, LV volume, LV pressure, filling pressure, aortic pressure, elastic pressure and a new diastolic elastance. An example elastance curve is shown in Figure 2, normalized with respect to time at 72 beats per minute.
Figure 2: Normalized time-varying elastance of the left ventricle. Minimum and maximum elastance are set at 0.097 and 2.3 mmHg / mL, respectively. The curve is generated by the getk function, shown in the Appendix.

**Arterial system**

The complete cardiac cycle generated by the ODE solver is shunted into an energetic tabulation routine that calculates AVO$_2$ as well as filtration rates through, into and out of the lumped capillary (Figure 4). Unless stated otherwise, the values in the following sub-section were taken from [18]; net filtration pressure is given as P$_{net}$ in equation (1).

\[
P_{net} = (\bar{P}_c - P_{tf}) + (\Pi_{tf} - \Pi_p)
\]  

(1)

\[
\bar{P}_c = \frac{P_{oa} + P_{cv}}{2}
\]

(2)
The mean capillary pressure $P_c$ was calculated from (2), where $P_{ca}$ is the capillary pressure at the arteriolar junction and $P_{cv}$ is the pressure on the venous end of the capillary; $P_r$ is the pressure of the interstitial fluid and is given a value of -3 mmHg; $\Pi_f$ is the osmotic pressure in the interstitial space and is given a value of 8 mmHg; the osmotic pressure from the plasma $\Pi_p$ due to albumin, fibrinogen and globulins is given a value of 28 mmHg. Note that positive terms indicate a contribution towards outward flow whereas negative terms contribute to the inward flow, thus the negative pressure of $P_r$ acts to abstract fluid from the capillary into the interstitial spaces.

The filtration pressure was then multiplied by the filtration coefficient of $6.67 \frac{ml}{min \cdot mmHg}$ and averaged, yielding the net volumetric filtration rate across the capillary wall. The lymphatic filtration rate is held constant at 1/30 mL/s.

$$AVO_2 = \frac{MQ + W_{ext}}{K_1 \cdot Q_c}$$

Arterial-venous oxygen differences were calculated from equation (3), which relates the energy used in metabolic processes to the energy delivered by blood. In the numerator, MQ is the metabolic rate of the body, which is roughly 100 W for a person consuming an average of 2000 kcal per day. $W_{ext}$ is the external work performed by the block of humanity on its environment. $K_1$ and $Q_c$ denote the energy equivalent of oxygen and the mean volumetric flow rate through the capillary, respectively.

**Control system**

If the net outward filtration flow rate $I_{filt}$ is less than the lymphatic filtration rate, the net outward filtration rate is taken to be zero. Otherwise, a constant, approximate lymphatic filtration rate (below) is subtracted from the net outward filtration rate. This new filtration flow rate $I_{sw}$ is assumed to be converted to sweat completely, allowing as there are no interstitial spaces between cells or tissues within the block of humanity.
\[ l_{SW} = I_T \mu \text{mL} \frac{1}{30 \text{s}} \]  

(4)

The rate of latent heat dissipated by the evaporation of sweat is calculated as shown in (5) using the density of blood per mL and the enthalpy of water, given as \(1.060 \times 10^{-3} \frac{\text{kg}}{\text{mL}}\) and \(2.257 \times 10^6 \frac{J}{kg}\) respectively.

\[ Q_L = I_{SW} \rho_{mL} \Delta h_{H_2O} \]  

(5)

The temperature change \(\Delta T_{CC}\) for a given beat is calculated as the difference in latent heat \(Q_L\) dissipated by the sweat rate to the heat generated by the current metabolic load \(MQ\) as shown in (6). The mass of the body \(m_p\) was given as 100 kg and the specific heat of the body \(c_{p-body}\) is given as \(3470 \frac{J}{kg \cdot ^\circ C}\). The change in body temperature is then added to the initial body temperature.

\[ \Delta T_{CC} = \frac{MQ - Q_L}{m_p \cdot c_{p-body}} \]  

(6)

\[ T_F = \Delta T_{CC} \cdot t_{cc} + T_i \]  

(7)

The relative difference between the nominal body temperature of 37 °C and the end-of-cycle body temperature \(T_F\) is the input error signal \(T_{Err}\) to a PI control system (Figure 5) that regulates heart rate and arterial resistance; note \(T_{Err}\) is directly proportional to heart rate and inversely proportional to arterial resistance, see skunkworks.m in the Appendix. These relationships are given in (8) and (9).

\[ \Delta HR = K_{p-HR} \cdot T_{Err} + K_{i-HR} \int T_{Err} \]  

(8)

\[ \Delta RA = K_{p-R_A} \cdot T_{Err} + K_{i-R_A} \int T_{Err} \]  

(9)

The proportional \((K_p)\) and integral \((K_i)\) coefficients were selected at the beginning of a given trial; the calculation of the respective change in heart rate and arterial resistance was the last
calculation made during a single cycle. The heart rate and arterial resistance values were then fed back into the initial conditions for the next cardiac cycle.

**Simulation parameters**

Two separate sets of simulations were run in this experiment. In the first, a “shotgun” simulation method using values of 0, 1, 10, 50 and 100 for HR $K_p$, HR $K_i$ and RA $K_p$ are run to create a map of output parameters; RA $K_i$ is set at a constant 0.1 for all trials. In the second, only 25 trials are generated with values of 0, 1, 10, 50 and 100 for HR $K_i$ and RA $K_p$ while HR $K_p$ is set at 10 and RA $K_i$ is maintained at 0.1. All trials are run for 3000 cardiac cycles.

Additionally, there exist some differences in the initial conditions between the two sets of trials. For the first set of trials, the external work performed by the block of humanity is set at 80 W but is increased to 228 W for the second and third sets. Mass also changes from 100 kg in the first set to 77 kg for the second. These changes are made to accommodate a comparison to those values found in [12], in which a 77 kg averaged participant exerts 228 W on an exercise ergometer.

**Model validation**

To validate the model, a baseline was established by running the simulation open-loop (with each control variable $K_p$ and $K_i$ set to zero, see Figure 5) for 3000 beats at 72 beats per minute (2500 s). The initial parameters for this simulation are detailed below in Table 1. These initial conditions resulted in steady beats with the following parameters: systolic aortic pressure 114.6 mmHg; diastolic aortic pressure 60.15 mmHg; peak LVP 114.8 mmHg; mean LV filling pressure 10.09 mmHg; stroke volume 63.78 mL; ejection fraction 55.56%; cardiac output 4.592 L/min; peak aortic flow 978.2 mL/s. Additionally, the simulated body reached a final body temperature of 37.88 °C with zero sweat produced. A representative beat is shown in Figure 3.
Table 1. Initial conditions for the open-loop simulation. LV refers to the left ventricle; LA refers to the left atrium. Note that these values are preserved for closed-loop simulations.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Initial</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LV $E_{min}$</td>
<td>0.097</td>
<td>mmHg/mL</td>
</tr>
<tr>
<td>LV $E_{max}$</td>
<td>2.300</td>
<td>mmHg/mL</td>
</tr>
<tr>
<td>Arterial resistance</td>
<td>0.928</td>
<td>PRU</td>
</tr>
<tr>
<td>Capillary resistance</td>
<td>0.072</td>
<td>PRU</td>
</tr>
<tr>
<td>Venous resistance</td>
<td>0.020</td>
<td>PRU</td>
</tr>
<tr>
<td>Arterial compliance</td>
<td>1.000</td>
<td>mL/mmHg</td>
</tr>
<tr>
<td>Venous compliance</td>
<td>30.00</td>
<td>mL/mmHg</td>
</tr>
<tr>
<td>Heart rate</td>
<td>72.00</td>
<td>BPM</td>
</tr>
<tr>
<td>Basal metabolic rate</td>
<td>100.0</td>
<td>W</td>
</tr>
<tr>
<td>External work</td>
<td>80.00</td>
<td>W</td>
</tr>
<tr>
<td>Efficiency</td>
<td>0.250</td>
<td>-</td>
</tr>
<tr>
<td>Mass</td>
<td>77.00</td>
<td>kg</td>
</tr>
<tr>
<td>Body temperature</td>
<td>37.30</td>
<td>°C</td>
</tr>
<tr>
<td>LV volume</td>
<td>120.0</td>
<td>mL</td>
</tr>
<tr>
<td>LV pressure</td>
<td>11.64</td>
<td>mmHg</td>
</tr>
<tr>
<td>LA pressure</td>
<td>10.00</td>
<td>mmHg</td>
</tr>
<tr>
<td>Aortic pressure</td>
<td>90.00</td>
<td>mmHg</td>
</tr>
<tr>
<td>Coronary flow</td>
<td>30.00</td>
<td>mL/s</td>
</tr>
<tr>
<td>Filtration pressure</td>
<td>0.300</td>
<td>mmHg</td>
</tr>
<tr>
<td>Sweat filtration rate</td>
<td>0.000</td>
<td>mL/s</td>
</tr>
<tr>
<td>Capillary pressure</td>
<td>25.00</td>
<td>mmHg</td>
</tr>
</tbody>
</table>
Figure 3: Pressure, flow and volume in the left ventricle. The plot represents a single beat at 72 BPM during an open-loop simulation of the model. LVV is the dotted-dashed line; LVP is the solid line; AoP is the dashed line; in- and outflow are shown as dotted lines. *Flow is scaled by 1/20. Additionally, magnitude of these two flows is shown here, but inflow occurs only during relaxation and outflow occurs only during ejection.
Figure 4: Program flow chart. The initial conditions for a beat are fed into an ODE solver, which produces a normalized elastance curve to compute pressures and flows in the left ventricle. These data are used to calculate metabolic changes and flows across the capillary wall as well as temperature changes during the beat. The temperature error between target and actual determines the magnitude of change the dual PI controller exerts on arterial resistance and heart rate.
Figure 5: PI controller used in the program. At the end of any given cardiac cycle, the total heat dissipated by the block of humanity is compared to the total heat produced and the difference ($T_{Err}$) is used to adjust heart rate and arterial resistance. The new $HR$ and $R_A$ are then used for the next iterative cardiac cycle.
RESULTS

The results of this experiment are split into two sub-sections, one for each set of simulation data as described in simulation parameters. In the first sub-section, the “shotgun” simulation method using values of 0, 1, 10, 50 and 100 for HR $K_p$, HR $K_i$ and RA $K_p$ are run to create a map of output parameters. In the second, only 25 trials are generated with values of 0, 1, 10, 50 and 100 for HR $K_i$ and RA $K_p$ while HR $K_p$ is set at 10. (As noted in the sub-sections below, the settling and rise time profiles were not significantly affected by changes in HR $K_p$ with one exception; thus, one value suffices to be representative.) However, note that in these trials the external workload of the block of humanity has increased to 228 W to match the profiles in [12].

Note that when settling or rise time is referenced, it is assumed to be the settling or rise time of the temperature in the block of humanity unless otherwise indicated. Additionally, “quintets” refer to the 5-trial sequence for the HR $K_p$ control variable in which the other two variables are held constant, e.g. “the quintet at HR $K_i = 0$ & RA $K_p = 10$” references the five trials where HR $K_i = 0$, RA $K_p$ = 10 and HR $K_p$ = 0, 1, 10, 50 and 100 consecutively. Similarly, triplets refer to 3-trial sequences for HR $K_i$ and quartets refer to 4-trial sequences of RA $K_p$.

The “Shotgun” Method of Parameter Estimation

In this sub-section, the “shotgun” simulation method using values of 0, 1, 10, 50 and 100 for HR $K_p$, HR $K_i$ and RA $K_p$ are analyzed to create a map of output parameters. Of these 125 trials, 75 corrected body temperature towards setpoint, but only 60 of these 75 trials reached the setpoint envelope of $37 \pm 0.75 \, ^\circ C$ within the simulation’s timeframe of 3000 cardiac cycles. The 65 trials that did not reach setpoint within the time allotted are not considered for the statistical settling time analysis.
**Relationship between settling time and proportional control of heart rate**

With some exceptions, increases in HR $K_p$ caused an increase in settling time; a 100-fold increase (from HR $K_p = 1$ to HR $K_p = 100$) averaged a 7.61% increase in settling time across all quintets. Among these data lies an interesting outlier at HR $K_i = 1$ & RA $K_p = 10$ where the 100-fold increase in HR $K_p$ results in a 50.58% increase in settling time, from 505 s to 762 s, a difference of 256 seconds. Without including this outlier, the per-quintet 100-fold mean increase in HR $K_p$ was 3.71%. Both of these figures are in contrast to a 5.23% increase in settling time across all quintets from HR $K_p = 0$ to HR $K_p = 100$. This trend can be most easily seen in Figure 6 below. Additionally, two quintets at HR $K_i = 10$ & RA $K_p = 1$ and at HR $K_i = 10$ & RA $K_p = 10$ actually showed a decrease in settling time with a 100-fold increase in HR $K_p$, which correspond to a 0.31% and 3.67% respective decrease in settling time.

Interestingly, analysis of 10-fold increases on intervals between both HR $K_p = 1$ to HR $K_p = 10$ and HR $K_p = 10$ to HR $K_p = 100$ yield much different results, especially between each pairing. Between the 1-10 interval, average settling time increased 2.43% contrary to an average settling time increase of 4.75% in the 10-100 interval. Excluding the same outlier at HR $K_i = 1$ & RA $K_p = 10$, average settling time increases for the 1-10 and 10-100 interval instead become 0.18% and 3.51%, respectively. These differences signify that the relationship of $\Delta T_s$ to changes in HR $K_p$ are not exponential. Relationships between incremental changes in HR $K_p$ and the commensurate percent change in settling time ($\Delta T_s$) are described in Table 2 below. The outlier trial at HR $K_i = 1$ & RA $K_p = 10$ is highlighted. Notably, this table shows how small changes in HR $K_p$ have little effect on settling time, but neither do large changes have as much an influence as do either HR $K_i$ or RA $K_p$ which can be seen in the following two sub-sections. In addition, the correlation between $\Delta T_s$ for each change in control variable and a straight line shows that $\Delta T_s$ represent a strong linear pattern ($|r| > 0.50$) for ten of the twelve sets. This indicates that changes in HR $K_p$ induce a linear change in settling time. Of the remaining two trial quintets, one set of $\Delta T_s$ correlates to a straight line moderately
(0.30 < |r| < 0.50) and the other weakly (|r| < 0.3); this last, the weakest, corresponds to the aforementioned outlier trial.

Figure 6. Settling time as a function of proportional control of heart rate. Mean values are represented by triangles. The dashed line represents the trend. Generally, settling time increased as a function of HR $K_p$; the average increase between the mean settling times taken at HR $K_p = 0$ and HR $K_p = 100$ was 5.23%. Two of the twelve total quintets had lower settling times at HR $K_p = 100$ than HR $K_p = 0$. 
Table 2. Percent differences in settling time for selected intervals of a quintet. For example, percent difference in settling time between HR $K_i = 1$ to HR $K_i = 10$ would be tabulated in the column “+$9$”. Magnitude of linear correlation ($|r|$) for each quintet is also included. Highlighted is the outlier at HR $K_i = 1$ and RA $K_p = 10$.

| HR $K_i$ | RA $K_p$ | $\Delta T_s$ (%) | $|r|$ |
|---------|----------|-------------------|-----|
| 0       | 0        | +1 0.05 0.33 2.01 2.27 4.87 0.47 4.32 4.82 0.94 |
| 1       | 0        | +10 0.09 0.64 4.68 4.17 9.74 0.55 4.04 9.64 0.90 |
| 10      | 0        | 0 0.00 0.06 0.13 -0.17 0.02 0.06 -0.04 0.02 0.46 |
| 0       | 1        | +1 0.05 0.53 2.00 2.27 4.87 0.47 4.32 4.81 0.96 |
| 1       | 1        | 1 0.10 0.98 4.20 5.17 10.66 0.87 4.69 10.54 0.97 |
| 10      | 1        | 1 0.01 0.03 0.00 -0.34 -0.31 0.02 -0.34 -0.31 0.78 |
| 0       | 10       | 10 0.10 1.19 4.04 2.85 8.28 1.09 7.01 8.17 0.84 |
| 1       | 10       | 10 0.20 27.42 8.95 8.69 50.88 27.16 18.41 50.58 0.23 |
| 10      | 10       | 10 -0.03 -0.29 -1.07 -2.37 -3.69 -0.26 -3.42 -3.67 0.96 |
| 0       | 50       | 50 0.04 -0.38 -0.94 5.05 3.66 -0.42 4.06 3.62 0.67 |
| 1       | 50       | 50 -0.31 -1.25 0.43 2.49 1.64 -0.95 2.93 1.95 0.84 |
| 10      | 50       | 10 0.00 0.08 0.40 0.70 1.19 0.08 1.11 1.19 0.97 |

**Relationship between settling time and integral control of heart rate**

Of the five trial variable settings for HR $K_i$, only three produced solutions that reached setpoint: 0, 1 and 10. Generally, increases in HR $K_i$ resulted in a decrease of settling time, which is somewhat contrary to the expectations of a control system. However, as shown in Figure 7, this trend is true for increases on the interval HR $K_i = 0$ to HR $K_i = 1$ as well as HR $K_i = 0$ to HR $K_i = 10$ but not for the interval HR $K_i = 1$ to HR $K_i = 10$.

Incrementing HR $K_i$ generally caused a change in settling time significantly larger than those produced by similar increments HR $K_p$. However, trial values of HR $K_i = 50$ and HR $K_i = 100$ shaped curves that were too slow to yield sweat and thus cool the block of humanity. Perhaps most interestingly in this data is the finding that there is a marked contrast in how HR $K_i$ affects $\Delta T_s$ that is strongly dependent on the value of RA $K_p$. Additionally, while increasing HR $K_i$ from 0 to 10 generally decreases settling time, on the interval from 1 to 10 there is a general increase in settling time. This can be seen most easily in Figure 7. Between the first and second values of HR $K_i$, there is a 25.97% decrease in mean settling time;
similarly, between HR $K_i = 0$ and HR $K_i = 10$ there is a 22.23% decrease in mean settling time. Conversely, the mean settling time increases by 4.92% between HR $K_i = 1$ and HR $K_i = 10$. Contrary to increments of HR $K_p$, the relationship between $\Delta T_s$ and HR $K_i$ is not generally linear, though in a few trials it is specifically linear. Notably, as RA $K_p$ increases there is a corresponding trend in HR $K_i$’s linear relationship to $\Delta T_s$. It is also interesting to note two trends expressed in Table 3. The first is that at HR $K_p = 50$ there is an “explosive” effect on incrementing HR $K_i$ from 0 to 1: on the interval RA $K_p = 0$ to RA $K_p = 50$, this corresponds to a 12.05%, 15.33%, 80.40% and 7.87% increase in percent change in $\Delta T_s$.

This last number points to the other trend that RA $K_p$ alters the behavior of incremental changes to HR $K_i$, a trend that can be easily seen at RA $K_p = 50$ when small changes in HR $K_i$ affect $\Delta T_s$ very little whereas larger increments have a correspondingly larger effect, which is in contrast to $\Delta T_s$ at smaller values of RA $K_p$. In both cases of RA $K_p = 10$ and RA $K_p = 50$, the response of the settling time is underdamped and the acceleration of RA $K_p$ is such that sweat begins to be produced rapidly following the onset of exercise. However, it is at the intersection HR $K_i = 10$ and RA $K_p = 50$ that the fastest and most stable system is produced, a system that has a fast enough heart rate response that RA does not bottom out, and conversely a fast enough arterial resistance response that HR does not reach dangerous levels.
Table 3. Percent differences in settling time for selected intervals of a quartet. For example, HR $K_i = 1$ to HR $K_i = 10$ would be tabulated in the column “+9”. Magnitude of linear correlation for each quartet is also included.

| HR $K_p$ | RA $K_p$ | $\Delta T_s$ (%) | $|r|$ |
|----------|----------|------------------|------|
|          |          | $+1$ | $+10$ | $+9$ |
| 0        | 0        | -20.87 | -16.18 | 5.93 | 0.17 |
| 1        | 0        | -20.84 | -16.22 | 5.84 | 0.17 |
| 10       | 0        | -20.79 | -16.57 | 5.32 | 0.16 |
| 50       | 0        | -23.02 | -22.12 | 0.74 | 0.03 |
| 100      | 0        | -17.20 | -20.06 | -3.46 | 0.16 |
| 0        | 1        | -22.11 | -18.15 | 5.08 | 0.14 |
| 1        | 1        | -22.07 | -18.19 | 4.97 | 0.14 |
| 10       | 1        | -21.76 | -18.56 | 4.09 | 0.12 |
| 50       | 1        | -25.11 | -25.24 | -0.11 | 0.00 |
| 100      | 1        | -17.81 | -22.19 | -5.33 | 0.24 |
| 0        | 10       | -69.24 | 1.38 | 229.58 | 0.50 |
| 1        | 10       | -69.21 | 1.25 | 228.82 | 0.50 |
| 10       | 10       | -61.27 | -0.09 | 157.93 | 0.50 |
| 50       | 10       | -146.55 | -5.27 | 57.30 | 0.48 |
| 100      | 10       | -57.14 | -9.83 | 110.37 | 0.45 |
| 0        | 50       | 0.25 | -62.64 | -62.73 | 0.86 |
| 1        | 50       | -0.10 | -62.65 | -62.62 | 0.87 |
| 10       | 50       | -0.63 | -62.46 | -62.23 | 0.87 |
| 50       | 50       | 0.74 | -162.84 | -164.81 | 0.86 |
| 100      | 50       | -1.71 | -63.53 | -62.90 | 0.88 |
Figure 7. Settling time as a function of integral control of heart rate. Values of HR $K_i = 50$ and $100$ are not shown as no trials with these values reached setpoint within the simulation timeframe. Mean values are represented by triangles. The dashed line represents the trend. Generally, settling time decreased as a function of HR $K_i$; the average decrease between the mean settling times taken at HR $K_i = 0$ and HR $K_i = 10$ was 22.23%.

**Relationship between settling time and proportional control of arterial resistance**

Of the five trial variable settings for RA $K_p$, only four produced solutions that reached setpoint: 0, 1, 10 and 50. Generally, increasing RA $K_p$ resulted in a decrease in settling time, which is keeping with the expectations of a control system. In those incidences where this convention does not hold, it is attributable to competing resources in the system, e.g. arterial resistance falls faster than heart rate increases, thus providing a net reduction in arterial pressure which in turn retards production of sweat and by extension cooling.

Incrementing RA $K_p$ generally caused a change in settling time significantly larger than those produced by similar increments of HR $K_p$. Much like with HR $K_i$, trial values of RA $K_p = 100$ shaped curves that were too slow to yield sweat and thus cool the block of humanity.
Additionally, the interplay between HR $K_i$ and RA $K_p$ is very apparent when looking at changes in settling time; these effects can be seen in Table 4 and Figure 8. In the plot specifically it is easy to observe that for every trial quintet RA $K_p$ decreases absolutely, but the five trials at $HR\ K_i = 1 \&\ RA\ K_p = 10$ reveal this trend is true globally but not explicitly true locally. This distinction can be seen more clearly in the table: excepting the five trials where $\Delta T_s$ was zero (or near zero) the only variations for which $\Delta T_s$ is positive were between RA $K_p = 10$ and RA $K_p = 50$ with HR $K_p = 0$ to 100 and HR $K_i = 1$. Notably, this trial is the outlier highlighted from the section *relationship between settling time and proportional control of heart rate* where these trials are the only ones in which HR $K_p$ plays a significant role in altering settling time.

In all other cases increases to RA $K_p$ result in reduced settling time, but at some point between RA $K_p = 10$ and RA $K_p = 50$ the critical damping point is reached, causing temperature response to transition from underdamped to overdamped behavior within the quartet. Due to this interaction between HR $K_i$ and RA $K_p$, there is an average increase in settling times between RA $K_p = 10$ and RA $K_p = 50$ of 7.09%. On all other intervals, however, average change in settling time for increments to RA $K_p$ remain negative at 1.27%, 41.50% and 53.87% for increments of 1, 10 and 50 RA $K_p$ respectively. Another facet of the interaction between HR $K_i$ and RA $K_p$ is the corresponding linearity of $\Delta T_s$ as each variable increases. When RA $K_p$ was held constant (Table 3) a clear progression from weak to strong linearity can be seen for each triplet. Conversely, Table 4 shows strong linearity when HR $K_i = 0$, very strong linearity when HR $K_i = 10$ but very weak linearity when HR $K_i = 1$ due to the aforementioned transition between under- and overdamped responses.
Table 4. Percent differences in settling time for selected intervals of a triplet. For example, RA $K_p = 1$ to RA $K_p = 10$ would be tabulated in the column “+9”. Magnitude of linear correlation for each triplet is also included.

| HR $K_p$ | HR $K_i$ | $\Delta T_s$ (%) | $|r|$ |
| --- | --- | --- | --- |
| | | +1 | +10 | +40 | +50 | +9 |
| 0 | 0 | 0.00% | -34.33% | -20.86% | -48.03% | -34.33% | 0.60 |
| 1 | 0 | 0.00% | -34.30% | -20.91% | -48.04% | -34.30% | 0.60 |
| 10 | 0 | 0.00% | -33.90% | -22.09% | -48.50% | -33.90% | 0.62 |
| 50 | 0 | -0.01% | -32.58% | -25.82% | -49.99% | -32.58% | 0.70 |
| 100 | 0 | 0.00% | -32.20% | -24.24% | -48.63% | -32.20% | 0.68 |
| 0 | 1 | -1.56% | -74.47% | 157.90% | -34.16% | -74.07% | 0.14 |
| 1 | 1 | -1.54% | -74.44% | 156.59% | -34.42% | -74.04% | 0.13 |
| 10 | 1 | -1.23% | -67.68% | 99.87% | -35.40% | -67.28% | 0.10 |
| 50 | 1 | -1.68% | -66.36% | 84.25% | -38.02% | -65.79% | 0.07 |
| 100 | 1 | -0.73% | -64.90% | 73.73% | -39.02% | -64.64% | 0.05 |
| 0 | 10 | -2.35% | -20.57% | -70.84% | -76.84% | -18.66% | 0.92 |
| 1 | 10 | -2.34% | -20.60% | -70.83% | -76.84% | -18.69% | 0.92 |
| 10 | 10 | -2.38% | -20.85% | -70.73% | -76.83% | -18.92% | 0.92 |
| 50 | 10 | -2.50% | -21.80% | -70.29% | -76.77% | -19.79% | 0.93 |
| 100 | 10 | -2.67% | -23.52% | -69.36% | -76.56% | -21.42% | 0.94 |
Figure 8. Settling time as a function of integral control of arterial resistance. Values of RA $K_p = 100$ are not shown since no trials with this value reached the temperature setpoint within the simulation timeframe. Mean values are represented by triangles. The dashed line represents the trend. Generally, settling time decreased as a function of RA $K_p$; the average decrease between the mean settling times taken at RA $K_p = 0$ and RA $K_p = 50$ was 53.87%. All quintets had lower settling times at RA $K_p = 50$ than at RA $K_p = 0$.

**Sweat production**

Using the sweat rates published by Godek et al. [13] for American football players, an upper bound of 0.81 mL/s can be established. Although the exercises producing the sweat rates from Godek’s experiment and this one differ significantly, Godek’s research highlights a reasonable upper bound for sweat rates in the present study.

Given these boundaries, the lowest sweat rate was at 0.20 mL/s and the highest at 0.43 mL/s, well under the maximum. There was a very weak correlation ($|r| = 0.08$) between the maximum sweat rate and settling time, due as much in part to many variable combinations being unable to swiftly and adequately induce a large enough pressure...
difference in the arterial vessel to begin countering the build-up of heat from exercise as it is from variable combinations introducing oscillations outside the settling envelope. A more fitting comparison between rise time and maximum sweat rate instead exhibits a very strong correlation ($|r| = 0.85$). Additionally, there was a strong correlation ($|r| = 0.78$) between the total amount of sweat produced (mL) and settling time. This last figure is not surprising, given that the production of heat within the block of humanity was constant at 320 W but simulation time was not, so for trials that shed heat more quickly through higher heart rates the total amount of sweat would be similarly reduced. More telling is the moderate correlation ($|r| = 0.61$) between the time-averaged sweat rate and settling time.

**Control variables**

Heart rate was bounded between 40 and 200, but in all 60 trials neither bound was reached. Minimum BPM did however go as low as 41.24, while maximum was 175.38. Likewise, arterial resistance was bounded between 0.1 and 4.0 peripheral resistance units (PRU), and in nearly every quintet where HR $K_i = 0$ or 1 the lower bound was reached. There exists one exception for the quintet at HR $K_i = 1$ and RA $K_p = 50$, when the minimum arterial resistance reached only 0.18.

**Fast settlers and risers**

Among all 60 trials, only two quintets had trials with settling times below 1000 seconds: the quintet at HR $K_i = 1$ & RA $K_p = 10$ and the quintet at HR $K_i = 10$ & RA $K_p = 50$. Representative samples from HR $K_p = 10$ are shown in Figure 9 and Figure 10. These two quintets are also evident as the 10 lowest data points in Figure 6. One quintet of particular note is that with HR $K_i = 50$ and RA $K_p = 50$. This quintet remains singular in this data set as one where sweat filtration was positive but body temperature increased. While the sweat produced was not nearly enough to counteract the build-up of heat within the body, it nonetheless remains the set of trials that most closely mimic actual physiological responses.
Figure 9. Time courses for HR $K_p = 10$, HR $K_i = 1$ and RA $K_p = 10$. Sweat filtration, arterial-venous oxygen difference, body temperature and the two control variables heart rate (dashed) and arterial resistance (solid) are shown. Settling time for this trial was 643 s and rise time was 278 s. Note that R\textsubscript{A} bottoms out for roughly 99 s until heart rate rises far enough to compensate. This trial is an excellent example of control variable combinations that met the success criteria but are not physiologically possible.
Figure 10. Time courses for \( \text{HR}_p = 10 \), \( \text{HR}_i = 10 \) and \( \text{RA}_p = 50 \). Sweat filtration, arterial-venous oxygen difference, body temperature and the two control variables heart rate (dashed) and arterial resistance (solid) are shown. Settling time for this trial was 486 s and rise time was 419 s, and the control variable combination is very close to reaching the critical damping point. Note that heart rate rises fast enough that \( R_A \) does not bottom out, which occurred in 35 trials. This trial is an excellent example of control variable combinations that met the success criteria but do not represent a real physiological response. However, it remains the fastest settling trial within this simulation data set.

**Increased workload and physiological considerations**

In this sub-section, the simulation set using values of 0, 1, 10, 50 and 100 for \( \text{HR}_i \) and \( \text{RA}_p \) with \( \text{HR}_p = 10 \) are analyzed to create a map of output parameters with a vastly increased external workload (228 W). Of these 25 trials, 16 corrected body temperature towards setpoint and all 16 trials reached the setpoint envelope of \( 37 \pm 0.75 \) °C within the simulation’s timeframe of 3000 heart beats. Note here that the 9 trials that did not reach setpoint had either \( \text{HR}_i = 50 \) or \( \text{HR}_i = 100 \), a similar pattern as those described in the previous sub-section; the trial \( \text{HR}_p = 50 \), \( \text{HR}_i = 50 \) and \( \text{RA}_p = 100 \) did reach the setpoint.
envelope, however. Of those 9 trials that did not reach setpoint, the trial with HR $K_i = 50$ and RA $K_p = 50$ yet again is the only one that had positive sweat filtration and, thus, cooling. This data is compared to [12] in Figure 11.

As shown by the experiment in [12], body temperature does not decrease during exercise of this magnitude (228 W, 60% $\dot{V}_{O_{2max}}$). This data is shown in Table 5, and it should be noted that the discrepancy in the final temperature $T_f$ can be explained by different simulation run times: in general, trials with HR $K_i = 50$ ran for an average of 245 seconds shorter than trials at HR $K_i = 100$ due to differences in heart rate.

![Body Temperature Graph](image)

Figure 11. Temperature comparison between reported data and model. Trial HR $K_p = 10$, HR $K_i = 50$ and RA $K_p = 50$ is plotted against interpolated data from [12]. Coefficient of determination $R^2 = 0.995$ between the model fit and reported data.
Table 5. Sweat production, final temperature and correlation coefficient between these variables for each trial of the second data set. In all trials, HR $K_p = 10$. Differences in sweat production between trials that converged to setpoint are due wholly or in part to differences in simulation run time; as noted, each trial was run for 3000 cardiac cycles, so simulation time decreases as average heart rate increases. $R$ denotes the correlation between a trial’s body temperature to that of the average participant in [12].

<table>
<thead>
<tr>
<th>HR $K_i$</th>
<th>RA $K_p$</th>
<th>Sweat (mL)</th>
<th>$T_f$ (°C)</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>391.36</td>
<td>36.98</td>
<td>-0.85</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>275.57</td>
<td>37.07</td>
<td>-0.43</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>256.99</td>
<td>37.20</td>
<td>-0.05</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>0.00</td>
<td>38.66</td>
<td>1.00</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>0.00</td>
<td>38.95</td>
<td>1.00</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>390.43</td>
<td>36.99</td>
<td>-0.85</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>283.50</td>
<td>36.97</td>
<td>-0.46</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>264.55</td>
<td>37.09</td>
<td>-0.05</td>
</tr>
<tr>
<td>50</td>
<td>1</td>
<td>0.00</td>
<td>38.66</td>
<td>1.00</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>0.00</td>
<td>38.95</td>
<td>1.00</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>389.41</td>
<td>37.00</td>
<td>-0.85</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>274.81</td>
<td>37.00</td>
<td>-0.56</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>249.74</td>
<td>37.00</td>
<td>-0.28</td>
</tr>
<tr>
<td>50</td>
<td>10</td>
<td>0.00</td>
<td>38.66</td>
<td>1.00</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>0.00</td>
<td>38.95</td>
<td>1.00</td>
</tr>
<tr>
<td>0</td>
<td>50</td>
<td>389.39</td>
<td>37.00</td>
<td>-0.86</td>
</tr>
<tr>
<td>1</td>
<td>50</td>
<td>236.74</td>
<td>37.01</td>
<td>-0.88</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>190.12</td>
<td>36.98</td>
<td>-0.86</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>42.89</td>
<td>38.27</td>
<td>1.00</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>0.00</td>
<td>38.95</td>
<td>1.00</td>
</tr>
<tr>
<td>0</td>
<td>100</td>
<td>388.00</td>
<td>37.01</td>
<td>-0.91</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>220.13</td>
<td>37.06</td>
<td>-0.97</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>177.15</td>
<td>37.04</td>
<td>-0.98</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>136.67</td>
<td>37.38</td>
<td>-0.72</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>0.00</td>
<td>38.95</td>
<td>1.00</td>
</tr>
</tbody>
</table>
DISCUSSION

Comparing only the temperature responses, the trial of HR $K_p = 10$, HR $K_i = 50$ and RA $K_p = 50$ most closely matches the data from [12], but two significant exceptions exist. First, while the block of humanity is powered by a time-changing elastance, i.e. the left ventricle, the properties $E_{\min}$ and $E_{\max}$ are constant. However, these two values are highly susceptible to inotropic effects and strongly correlated to both LV mass and LV end-diastolic volume (preload) [24], [41]. The range of $E_{\min}$ to $E_{\max}$ used in these simulations (detailed in simulation parameters) embodies normal cardiac function, but under stress—in this case, exercise—these values are no longer representative. In fact, [41] shows that at high values of preload, $E_{\max}$ may increase twenty-fold from the value utilized in these simulations. This incongruity results in a major shortfall of stroke volume—and by extension cardiac output—relative to that seen in the human body: with respect to the data in [12], there is a 55% and 59% difference between reported and simulated cardiac output and stroke volume.

Second, the time course of body temperature in the block of humanity can be seen to be concave up, indicating that given enough time body temperature would reach a maximum and then, albeit slowly, begin decreasing. This phenomenon is rooted in the fact that the block of humanity has been designed to value its temperature above all else—including survival. If we assume that there exists a physiological analog to this lumped parameter control system, then it follows from the comparison in Figure 11 that under some conditions maintaining thermo-fluid homeostasis is not in fact the primary goal of the human body. Else, the model’s output would more closely resemble that of physiology. Although thermoregulation is important, there still exists a range of heat storage the body deems acceptable in order to maximize its useful work; this is likely a mechanism in which useful may be defined as “increasing survival.” The premise of exchanging future hyperthermia for present work is quite evident from a comparison of the data shown in [12] to that of Table 5, where each living participant was able to leverage heat storage towards the performance of work whereas the
block of humanity instead leveraged its available resources towards decreasing core
temperature, even in the cases where it could not generate a large enough pressure difference
across the capillary to generate sweat.

A simple difference quotient is taken for the block of humanity to maintain body
temperature, expressed in equation (6). When the cooling exceeds the heat generated by the
block of humanity, temperature naturally decreases. All variables in these two equations are
held constant with the exception of $I_{SW}$, so body temperature depends solely on the pressure
difference at the capillary: the greater the net outward (tissue-ward) pressure, the greater
the flow and the greater the cooling. Effectively, the lumped parameter control system
emulates the hypothalamus but is constrained to a physiological setpoint described by the
coded conditions ($E_{max}$, BMR, control variable coefficients, etc.) and this setpoint can change
due to effectors ranging from ventilation or stress to minutia like the time of day or
psychosomatic beliefs [4], [21], [32]. Rather, the setpoint is more likely an emergent property
of a system attempting to minimize local entropy production, as posited by Ilya Prigogene in
his work on dissipative structures.

Schaible [38] posits that there exists a thermodynamic spectrum a living body falls
along at any given time. Along the abscissa, the negative direction indicates efficiency and
the positive direction indicates survival reserve. Similarly along the ordinate, the negative
direction indicates equilibrium while the positive indicates work. For the participants of [12],
evidence of fatigue suggests they are operating somewhere in the first quadrant: each
participant is exerting a fraction of their maximum available external work (60%) which
eventually leads to fatigue, i.e. the consumption of available resources outstrips the
generation (short-term) or accumulation (long-term) of resources. Conversely, the block of
humanity has innumerably fewer resources to consume; in fact, the only finite resources the
block of humanity can control are the setpoints of its two control variables HR and $R_A$—and,
maximum state evokes.
CONCLUSION

In this thesis a model CVS was built to emulate the thermodynamic responses observed at the capillary during exercise. A comprehensive evaluation was also performed to assess the contribution each control variable coefficient had in any given permutation to the model’s response. Furthermore, the results of these permutative trials were evaluated under conditions that could be compared to literature.

It was found that while the model’s responses could not replicate the responses in literature, the discrepancy was likely due to the assumptions made during the creation of the model that affected the target setpoint; in effect the model attempted to accomplish a task different from the living participants it was compared to. In the first, the model aimed exclusively to reduce body temperature through its control of heart rate and arterial resistance, whereas in the latter the participants aimed exclusively to maintain a constant external workload. However, while the primitive feedback loops in the model may not accurately portray *in vivo* performance of the human body, the simulation results highlight the potential of the capillary-centric, coupling-as-thermodynamics model to explore the emergent properties of the dynamic, adaptive and complex cardiovascular system.
REFERENCES


[29] A. Lerant et al., “Preventing and treating hypoxia: using a physiology simulator to
demonstrate the value of pre-oxygenation and the futility of hyperventilation,” Int J Med

pressure regulation: towards individualized models of whole-body physiology,” PLoS

[31] E. S. Myhre et al., "Optimal matching between canine left ventricle and afterload,” Am

complex regulation of human exercise performance,” Appl Physiol Nutr Metab, vol. 36,

[33] H. Piene, "Interaction between the Right Heart Ventricle and Its Arterial Load: A

[34] M. Reinig et al., “Left ventricular endocardial pacing: a transarterial approach,” Pacing

[35] K. Sagawa, “Critique of a large-scale organ system model: Guytonian cardiovascular

[36] K. E. Samuelson et al., "Time domain reflectometer impedance sensor method of use

[37] M. V. Savage and G. L. Brengelmann, "Control of skin blood flow in the neutral zone of
human body temperature regulation," J Appl Physiol, vol. 80 no. 4, pp. 1249-1257,
1996.

[38] N. S. Schaible, “Minimum entropy generation in the cardiovascular system,” M.S. thesis,
Elec. And Comp. Engr., North Dakota State University, Fargo, ND, 2011.


APPENDIX. MATLAB CODE

In this section, the code used to run the simulations is reported. **Note that the sections hemodynamics.m, odesolver.m and getk.m, marked with a superscript cross, are co-authored materials.** These functions, routines or methods were originally written by Andrew McNally, Matthew Korpela, Erin Lamke and Matthew Hudson of Iron Range Engineering in the unpublished work titled “Computational Model of a Left Ventricle: Showing the Effects of Inertia on Cardiac Dyssynchrony.” The last known revision of this work occurred Feb. 2012. The code has been revised such that the original functions are altered significantly, or it has been optimized in such a way that the original functions remain intact but are significantly improved over the original version.

**capinator.m**

```matlab
%%
% ----------------------------------------------- %
% Author:    Drew Taylor
% Date:      May 16, 2012
% Last Rev:  Sep 21, 2015
% Title:     capinator.m
% %
% ----------------------------------------------- %

clear all

logmode = 0;
if logmode == 1
    vdate = clock;
    if datenum(vdate(2)) < 10
        month = ['0' num2str(datenum(vdate(2)))];
    else
        month = num2str(datenum(vdate(2)));
    end
    if datenum(vdate(3)) < 10
        day = ['0' num2str(datenum(vdate(3)))];
    else
        day = num2str(datenum(vdate(3)));
    end
```

41
end

if datenum(vdate(4)) < 10
    hour = ['0' num2str(datenum(vdate(4)))];
else
    hour = num2str(datenum(vdate(4)));
end

if datenum(vdate(5)) < 10
    minute = ['0' num2str(datenum(vdate(5)))];
else
    minute = num2str(datenum(vdate(5)));
end

filecount1 = 1;
filecount2 = 1;
filedate = [num2str(datenum(vdate(1))) '. month '. day '.'];
filetime = [hour minute];
filename1 = [filedate filetime '.waveforms_' num2str(filecount1) '.txt'];
filename2 = [filedate filetime '.heatstuff_' num2str(filecount2) '.txt'];

% A = [vAoP vI1 vIi vIo vLAP vLVP vPE1 vQc vV1 vVi vVo];
waveformhead = ['AoP', 'I1', 'Ii', 'Io', 'LAP', 'LVP', 'PE1', 'Qc', 'V1', 'Vi', 'Vo'];
wfheadformat = '%9s %9s %9s %9s %9s %9s %9s %9s %9s %9s %9s\r\n';
wfformatSpec = '%3.4f %3.4f %3.4f %3.4f %3.4f %3.4f %3.4f %3.4f %3.4f %3.4f %3.4f\r\n';

% B = [vAVO2'; vBodTemp'; vIsweat'; vMQ'; vPca'; vQlat'; vRa'; vRaErr'; vTemperr'; vbpm'];
heatstuffhead = ['vAVO2', 'vBodTemp', 'vIsweat', 'vMQ', 'vPca', 'vQlat', 'vRa', 'vRaErr', 'vTemperr', 'vbpm'];
hsheadformat = '%10s %8s %8s %8s %8s %8s %8s %8s %8s %8s %8s\r\n';
hsformatSpec = '%10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f\r\n';

fileID1 = fopen(filename1,'w');
fprintf(fileID1,wfheadformat,'AoP', 'I1', 'Ii', 'Io', 'LAP', 'LVP', 'PE1', 'Qc', 'V1', 'Vi', 'Vo');
% fprintf(fileID,wfformatSpec,A);
% fclose(fileID1);

fileID2 = fopen(filename2,'w');
fprintf(fileID2,hsheadformat,'vAVO2', 'vBodTemp', 'vIsweat', 'vMQ', 'vPca', 'vQlat', 'vRa', 'vRaErr', 'vTemperr', 'vbpm');
% fprintf(fileID,hsformatSpec,B);
% fclose(fileID2);
end

val_run = false;

runspecHRKp = [0, 1, 10, 50, 100];
runspecHRKi = [0, 1, 10, 50, 100];
runspecRAKp = [0, 1, 10, 50, 100];
tic
for loopdex3 = 1:length(runspecRAKp)
for loopdex2 = 1:length(runspecHRKi)
for loopdex1 = 1:length(runspecHRKp)

clear PE1 I1 Ii Io LVP V1 Vi Vo LAP AoP e_1 Ea_RC Ea_EQ Ea_SG beatendtime;

%% Declare initial conditions
r1 = 0.0005;  %0.0005
elmin = 0.097;  %0.02
elmax = 2.3;  %5
m1 = 0.00045;  %0.00045
t1 = 0;  %0

% Arterial parameters
% Ra = 0.7*17*60/1000;  %1.5
Ra = 0.90933*17*60/1000;
Rc = 0.07067*17*60/1000;  %0.1
Rv = 0.02*17*60/1000;
Cv = 30;  %15
Ca = 1;

% bpm = beats per minute; MQ = metabolism (W);
% EWork = external power (J/s)
bpm = 72;  %de Cort
MQinit = 100;  % watts
Work = 512;  % watts
Effnc = 0.25;  % 25% efficiency
EWork = Work*Effnc;
heat_rad = 0;  % watts, heat due to radiation
heat_conv = 0;  % watts, heat due to convection

% K1 = energy equivalent of oxygen (J/mL O2)
K1 = 20;

% mass = N * s^2 / m (kg)
mass = 77;  % kg

age = 20;

bpmax = 220 - age;

% degrees C
BodTemp = 35.9;
BodTempTgt = 37;

% step = dt, cycle = secs to run
step = 0.001;
cycle = 2500;

%if truncating values, this sets the decimate resample rate at 1/resample,
%e.g. resample = 2 would halve the number of data points. must be integer.
resample = 1;

% defines total runtime samples length & index of end of first beat
% beats finds the number of beats in run time (assuming steady state BPM)
t_beat = 0:step:cycle;

43
pbeatdex = 0;
nbeatdex = ceil((60/bpm) / step + 1);
beats = cycle / (60 / bpm);

% E in mmHg / mL; V in mL; P in mmHg
volume = 120;
pressure = volume*e1min;

% Initialization parameters for the hemodynamics
init_hemo = ...
[pressure ... % 01 PE1
  0 ... % 02 I1 net flow into ventricle
  0 ... % 03 Ii inlet flow
  0 ... % 04 Io outlet flow
pressure ...
  05 LVP
volume ...
   06 V1 intial volume
  0 ... % 07 Vi integral Ii
  0 ... % 08 Vo integral Io
  10 ... % 09 LAP mmHg
  90 ... % 10 AoP mmHg
e1min ...
  11 e_1 diastolic elastance
bpm]; ...

% Initialization parameters for the thermodynamics and flows
init_therm = ...
[30 ...
  01 Qc mL/s coronary flow
  0.1 ...
  0.3 ...
  0 ... % 03 net filt pres mmHg
  0 ... % 04 sweat filtration mL/s
  25 ...
  0 ...
  BodTemp ...
  0.3 ...
  MQinit ...
  0.3 ...
  EWork]; ...

init_time = ...
[step ...
  2 first beat start index
pbeatdex ...
   % 4 beat number
nbeatdex ...
  3 first beat end index
1];

ctrl_bits = ...
[1 ...
  % Heart rate control bit; on = 1
  1];
  % Arterial resistance control bit; on = 1

art_props  = [Ra, Rc, Rv, Cv, Ca];
vent_props = [r1,e1min,e1max,m1,t1];

% Temperrdex = zeros(floor(beats),1);7
TempErrdex = BodTemp - BodTempTgt;
RaErrdex = 0;

%% Run X beats
for beatnum = 1:beats
    if val_run == true
if beatnum >= 0
    MQinit = 80;
end
end
endofbeat(1,beatnum) = nbeatdex;

if beatnum ~= beats+1
% Displays beat number on main screen
    disp(['beatnum = ' num2str(beatnum)]);
    % disp(['bpm = ' num2str(bpm)]);

% Generate elastance waveform using ode23 solver
[PE1, I1, Ii, Io, LVP, V1, Vi, Vo, LAP, AoP, e_1] = odesolver(init_hemo, init_time, vent_props, art_props, bpm);

% Solve for Ea
    ts=pbeatdex*step;
    te=nbeatdex*step;
    BigT = ts:step:te;
    ejecting = find(Io>0);
    Ea_RC = AoP(ejecting(end))/max(V1);
    Ea_EQ = (1 - Ca * (AoP(ejecting(1)) - AoP(ejecting(end))) ./ max(Vo)) * (Ra) ./ (te-ts)';
    Ea_SG = -0.127 + 1.023 * (Ra)./(te-ts) + 0.314 / Ca;
%
    disp(['Ea_RC = ' num2str(Ea_RC)]);
    disp(['Ea_EQ = ' num2str(Ea_EQ)]);
    disp(['Ea_SG = ' num2str(Ea_SG)]);
    fprintf(' 
');
%
    hemoform = [PE1, I1, Ii, Io, LVP, V1, Vi, Vo, LAP, AoP, e_1];
%
    % Compute flows and thermodynamic quantities
    [Qc, AVO2, filtnet, filtrate, lymphfilt, Pca, Ii, dAoP] ... = thermoflow(hemoform, init_therm, init_time, art_props);
    hemoform = [PE1, I1, Ii, Io, LVP, V1, Vi, Vo, LAP, AoP, e_1];
    % thermoform = [Qc, AVO2, filtnet, lymphfilt, Pca];
%
    if beatnum == 1
        formstep = 60/bpm/(length(PE1)-1);
        beatendtime = [1:beats]';
        beattime = [0:formstep:60/bpm]';
    else
        formstep = 60/bpm/(length(PE1));
        beattime = [formstep:formstep:60/bpm] + beatendtime(beatnum-1,1);
    end

    beatendtime(beatnum,1) = beattime(end);
%
    beattime = [0:formstep:60/bpm]';
    % beatendtime(beatnum) = beattime(end);
    % beattimex = [beatendtime(end)+formstep:formstep:60/bpm+beatendtime(end)'];

45
%% Send error signal to controller
MQE = MQinit + EWork;
heatgen = MQE + heat_conv + heat_rad;

%oscar - HRKp, boyd - HRKi, ike - RAKp
% Controller variables
HRKp = runspecHRKp(loopdex1);
HRKi = runspecHRKi(loopdex2);
RAKp = runspecRAKp(loopdex3);
RAKi = 0.1;

[bpm, delHR, Qlat, BodTemp, sweatfilt, Ra, RaErr, Isweat] =
skunkworks(filtrate, bpm, beatnum, heatgen, TempErrdex, BodTemp, art Props, ...
RaErrdex, BodTempTgt, step, mass, ctrl_bits, beatendtime, ...

bpmmax, HRKp, HRKi, RAKp, RAKi);
TempErrdex(beatnum,1) = BodTemp - BodTempTgt;
RaErrdex(beatnum,1) = RaErr;

art_props = [Ra, Rc, Rv, Cv, Ca];

%% Assign values to vectors
if beatnum == 1 || logmode == 1;
  vPE1 = decimate(PE1,resample);
  vI1 = decimate(I1,resample);
  vIi = decimate(Ii,resample);
  vIo = decimate(Io,resample);
  vLVP = decimate(LVP,resample);
  vV1 = decimate(V1,resample);
  vVi = decimate(Vi,resample);
  vVo = decimate(Vo,resample);
  vLAP = decimate(LAP,resample);
  vAoP = decimate(AoP,resample);
  ve_1 = decimate(e_1,resample);
  vQc = decimate(Qc,resample);
  vAVO2 = AVO2(end);
  vfiltnet = filtnet(end);
  vfiltrate = filtrate(end);
  vlymphfilt = lymphfilt(end);
  vsweatfilt = sweatfilt(end);
  vPca = Pca(end);
  vQlat = Qlat(end);
  vBodTemp = BodTemp(end);
  vTempErr = TempErrdex(end);
  vMQ = MQE(end);
  vEWork = EWork(end);
  vbpm = bpm(end);
  vdAoP = dAoP(end);
  vdelHR = delHR(end);
  vIsweat = Isweat(end);
  vRa = Ra(end);
  vRaErr = RaErr(end);
  vbeattime = decimate(beattime,resample);
vEa_RC = decimate(Ea_RC, resample);
vEa_EQ = decimate(Ea_EQ, resample);
vEa_SG = decimate(Ea_SG, resample);
else
vPE1 = [vPE1; decimate(PE1, resample)];
vI1 = [vI1; decimate(I1, resample)];
vIi = [vIi; decimate(Ii, resample)];
vIo = [vIo; decimate(Io, resample)];
vLVP = [vLVP; decimate(LVP, resample)];
vV1 = [vV1; decimate(V1, resample)];
vVi = [vVi; decimate(Vi, resample)];
vVo = [vVo; decimate(Vo, resample)];
vLAP = [vLAP; decimate(LAP, resample)];
vAoP = [vAoP; decimate(AoP, resample)];
ve_1 = [ve_1; decimate(e_1, resample)];
vQc = [vQc; decimate(Qc, resample)];
vbeattime = [vbeattime; decimate(beattime, resample)];

vAVO2 = [vAVO2; AVO2(end)];
vfiltnet = [vfiltnet; filtnet(end)];
filtrate = [filtrate; filtrate(end)];
vlymphfilt = [vlymphfilt; lymphfilt(end)];
vsweatfilt = [vsweatfilt; sweatfilt(end)];
vPca = [vPca; Pca(end)];
vQlat = [vQlat; Qlat(end)];
vBodTemp = [vBodTemp; BodTemp(end)];
vTempErr = [vTempErr; TempErrdex(end)];
vMQ = [vMQ; MQE(end)];
vEWork = [vEWork; EWork(end)];
vbpm = [vbpm; bpm(end)];
vAoP = [vAoP; aAoP(end)];
vdelHR = [vdelHR; delHR(end)];
vIsweat = [vIsweat; Isweat(end)];
vRa = [vRa; Ra(end)];
vRaErr = [vRaErr; RaErr(end)];
vEa_RC = [vEa_RC; Ea_RC(end)];
vEa_EQ = [vEa_EQ; Ea_EQ(end)];
vEa_SG = [vEa_SG; Ea_SG(end)];
end

%% Pull out end of beat values

% PE1 = getlast(PE1);
% I1 = getlast(I1);
% Ii = getlast(Ii);
% Io = getlast(Io);
% LVP = getlast(LVP);
% V1 = getlast(V1);
% Vi = getlast(Vi);
% Vo = getlast(Vo);
% LAP = getlast(LAP);
% AoP = getlast(AoP);
% e_1 = getlast(e_1);
% Qc = getlast(Qc);
% AVO2 = getlast(AVO2);
% filtnet = getlast(filtnet);
lymphfilt = getlast(lymphfilt);

sweatfilt = getlast(sweatfilt);
Pca = getlast(Pca);
Qlat = getlast(Qlat);
BodTemp = getlast(BodTemp);
Temperr = getlast(Temperr);
MQinit = getlast(MQdex);
EWork = getlast(EWork);
bpm = getlast(bpm);

PE1 = PE1(end);
I1 = I1(end);
Ii = Ii(end);
Io = Io(end);
LVP = LVP(end);
V1 = V1(end);
Vi = Vi(end);
Vo = Vo(end);
LAP = LAP(end);
AoP = AoP(end);
e_1 = e_1(end);
QC = QC(end);
AVO2 = AVO2(end);
filtnet = filtnet(end);
lymphfilt = lymphfilt(end);
sweatfilt = sweatfilt(end);
Pca = Pca(end);
Qlat = Qlat(end);
BodTemp = BodTemp(end);
TempErr = TempErrdex(end);
MQinit = MQdex(end);
EWork = EWork(end);
bpm = bpm(end);

%% Reinitialize arrays

beatrat = ceil(60/bpm * 100) / 100;
pbeatdex = nbeatdex;
nbeatdex = ceil(pbeatdex + (beatrat) / step);
beatmod = cycle / (60 / bpm);

if beatnum == cycle/2
    MQinit = 100;
end

init_hemo = ...

[PE1 ... % 01 PE1
I1 ... % 02 I1 net flow into ventricle
Ii ... % 03 Ii inlet flow
Io ... % 04 Io outlet flow
LVP ... % 05 LVP
V1 ... % 06 V1 initial volume
Vi ... % 07 Vi integral Ii
Vo ... % 08 Vo integral Io
LAP ... % 09 LAP mmHg
AoP ... % 10 AoP mmHg
e_1 ... % 11 e_1 diastolic elastance
bpm]; ... % 12 bpm 1 / s

% Initialization parameters for the thermodynamics and flows
init_therm = ...
[Qc ... % 01 Qc mL/s coronary flow
AVO2 ... % 02 AVO2 mL/100 mL
filtnet ... % 03 net filt pres mmHg
sweatfilt ... % 04 sweat filtration mL/s
Pca ... % 05 Pca mmHg
Qlat ... % 06 latent heat J / s
BodTemp ... % 07 body temp (degrees celsius)
TempErr ... % 08 temperature error
MQinit ... % 09 metabolism watts
EWork]; ... % 10 external work dot watts

init_time = ...
[step ... % 1 step size
pbeatdex ... % 2 first beat start index
nbeatdex ... % 3 first beat end index
beatnum]; % 4 beat number
else
break
end

if logmode == 1
A = [vAoP'; vI1'; vIi'; vIo'; vLAP'; vLVP'; vPE1'; vQc'; vV1'; vVi';
vVo';];
B = [vAVO2'; vBodTemp'; vIsweat'; vMQ'; vPca'; vQlat'; vRa'; vRaErr';
vTempErr'; vbpm'];

fileInfo = dir(filename1);
fileSize = fileInfo.bytes;
% if fileSize > 100000
% fclose(fileID1);
% filecount1 = filecount1 + 1;
% filename1 = [filedate filtime '.waveforms_'
num2str(filecount1) '.txt'];
% fileID1 = fopen(filename1,'w');
% end
fprintf(fileID1,wfformatSpec,A);

fileInfo = dir(filename2);
fileSize = fileInfo.bytes;
% if fileSize > 100000
% fclose(fileID2);
% filecount2 = filecount2 + 1;
% filename2 = [filedate filtime '.waveforms_'
num2str(filecount2) '.txt'];
% fileID2 = fopen(filename2,'w');
% end
fprintf(fileID2,hsformatSpec,B);
```matlab
clear vAoP vI vIi vIo vLAP vLVP vPE1 vQc vV1 vVi vVo vAVO2 vBodTemp
vIsweat vMQ vPca vQlat vRa vRaErr vTemperr vbpm
end
fclose all;

% fileID = fopen(fileID1,'a');
% fclose(fileID);
%
% fileID = fopen(fileID2,'a');
% fclose(fileID);
%
beatdex = 1:beats;

close all
[~, name] = system('hostname');
name = strtrim(name);
if strcmp('ilikeike',name) == 1
    dbxpath = 'C:\Users\Drew\Dropbox\Thesis\Figures';
    subpath = '\ike\';
elseif strcmp('phenomenaloscar',name) == 1
    dbxpath = 'I:\Dropbox\Thesis\Figures';
    subpath = '\oscar\';
elseif strcmp('savvyboyd',name) == 1
    dbxpath = 'R:\Dropbox\Thesis\Figures';
    subpath = '\boyd\';
elseif strcmp('gregariousfrank',name) == 1
    dbxpath = 'C:\Dropbox\Thesis\Figures';
    subpath = '\frank\';
end
if val_run == true
    mat_filename = [dbxpath,'\Ki-Ra','num2str(RAKi),subpath,num2str(beatnum),', 'beats GLF - HR Kp ','num2str(HRKp),' Ki ','num2str(HRKi),'- Ra Kp ','num2str(RAKp),' Ki ','num2str(RAKi),', 'VAL.mat'];
else
    mat_filename = [dbxpath,'\Ki-Ra','num2str(RAKi),subpath,num2str(beatnum),', 'beats GLF - HR Kp ','num2str(HRKp),' Ki ','num2str(HRKi),'- Ra Kp ','num2str(RAKp),' Ki ','num2str(RAKi),', '.mat'];
end
save(mat_filename);

% if logmode ~= 1;
%     t_tot = length(vAoP);
%     t_tot2 = interp1(vQc,0:step:beats);
% % plotting

h1 = figure(1);
```
subplot(2,2,1)
plot(beatendtime, vsweatfilt)
title('Sweat Filtration Rate')
xlabel('Time (s)')
ylabel('Filtration rate (mL / s)')
% figure
subplot(2,2,2)
% figure
plot(beatendtime, vAVO2, 'b');
title('AVO_2')
xlabel('Time (s)')
ylabel('Concentration (mL / 100 mL)')
% figure
subplot(2,2,3)
plot(beatendtime, vBodTemp)
title('Body Temperature')
xlabel('Time (s)')
ylabel('Temperature (deg C)')
% figure
subplot(2,2,4)
% this set of instructions plots Ra and BPM on one plot
x1 = beatendtime;
y1 = vbpm;
x2 = beatendtime;
y2 = vRa;

[hax, hL1, hL2] = plotyy(x1, y1, x2, y2);
set(hax(1), 'XColor', [.8 0 0], 'YColor', [.8 0 0])
set(hax(2), 'XColor', 'k', 'YColor', 'k')
set(hL1, 'Color', 'red')
set(hL2, 'Color', 'black')
title('Heart Rate & Arterial Resistance')
ylabel(hax(1), 'Heart Rate (BPM)')
ylabel(hax(2), 'Resistance (PRU)')
xlabel('Time (s)')

% plot EA
h2 = figure(2);%
plot(beatendtime, vEa_RC, 'r', beatendtime, vEa_EQ, 'k', beatendtime, vEa_SG, 'c')
title('Effective Arterial Elastances')
xlabel('Time (s)')
ylabel('Ea (mL / s)')
legend('E_A (P_E_S/SV)', 'E_A (Eqn)', 'E_A (Segers)')

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% figure
% plot(vbpm)
% ylabel('BPM')
% fprintf('
')
figure
plot(vbeattime, vAoP,'b'); hold on;
plot(vbeattime, vLVP,'g');
plot(vbeattime, vIo/10,'r');
plot(vbeattime, vIi/10,'k-');
plot(vbeattime, vV1,'k-.');
plot(beatendtime, vPca,'c');
plot(vbeattime, vLAP, 'm');hold off;

legend('AoP','LVP','Io','Ii','V1','Pca','LAP','Location','NorthEastOutside')
title('AoP / LVP')
xlabel('Time')
ylabel('Pressure (mmHg)')

% figure
% plot(vIi)
% xlabel('Time (ms)')
% ylabel('Ii ml/s')
% title('Cardiac Output')

% figure
% plot(vAoP)
% xlabel('Time (ms)')
% ylabel('Aop (mmHg)')
% title('Blood Pressure')

% figure
% plot(vV1,vLVP)
% xlabel('Volume (mL)')
% ylabel('Pressure (mmHg)')
% title('PV Loop')

end

pause on
if val_run == true
    fig_filename = [dbxpath, '\Ki-Ra ',num2str(RAKi),subpath,num2str(beatnum),' beats GLF - HR Kp ','num2str(HRKp), ' Ki ',num2str(HRKi),'- Ra Kp ',num2str(RAKp),' Ki ','num2str(RAKi)',' VAL.fig'];
    saveas(h1,fig_filename);
    fig_filename = [dbxpath, '\Ki-Ra ',num2str(RAKi),subpath,num2str(beatnum),' beats GLF - HR Kp ','num2str(HRKp), ' Ki ',num2str(HRKi),'- Ra Kp ',num2str(RAKp),' Ki ','num2str(RAKi)',' EA VAL.fig'];
    saveas(h2,fig_filename);
else
    fig_filename = [dbxpath, '\Ki-Ra ',num2str(RAKi),subpath,num2str(beatnum),' beats GLF - HR Kp ','num2str(HRKp), ' Ki ',num2str(HRKi),'- Ra Kp ',num2str(RAKp),' Ki ','num2str(RAKi)',' .fig'];
    saveas(h1,fig_filename);
    fig_filename = [dbxpath, '\Ki-Ra ',num2str(RAKi),subpath,num2str(beatnum),' beats GLF - HR Kp ','num2str(HRKp), ' Ki ',num2str(HRKi),'- Ra Kp ',num2str(RAKp),' Ki ','num2str(RAKi)',' .fig'];
    saveas(h2,fig_filename);
end
'num2str(HRKp),' Ki ',num2str(HRKi),' - Ra Kp ',num2str(RAKp),' Ki '
'num2str(RAKi),' EA.fig'];
    saveas(h2,fig_filename);
end

pause(3)
pause off

% fig_filename = ['D:\Dropbox\Thesis\Figures\Variable Kp-Ra, Ki-Ra 'num2str(RAKi),'\6000 beats GLF - HR Kp ',num2str(HRKp),' Ki '
um2str(HRKi),' - Ra Kp ',num2str(RAKp),' Ki ',num2str(RAKi),' EA.fig'];
% saveas(h2,fig_filename);
toc
fprintf(1,'%c',7);

end
end
end
toc
function [Qc, AVO2, filtnet, filtrate, lymphfilt, Pca, Ii, dAoP] = thermoflow(hemoform, init1, init2, art_props) ...

%% Initializations
Ii = hemoform(:,3);
Io = hemoform(:,4);
LAP = hemoform(:,9);
AoP = hemoform(:,10);

MQ = init1(:,9);
EWork = init1(:,10);

step = init2(:,1);

% [Qc, AVO2, filtnet, Isweat, Pca, Qlat, BodTemp, Temperr, MQ, EWork]

% Arterial parameters
Ra = art_props(:,1);
Rc = art_props(:,2);  % 0.1
Rv = art_props(:,3);  % 0.5
Cv = art_props(:,4);  % 15
Ca = art_props(:,5);

% pressures in mmHg
mPAC ref = 17.3;
Pflu = -3;
Piflu = 8;
Piplas = 28; % changed this

% energy equivalent of oxygen J/mL
K1 = 20;

% enthalpy of water J / kg
delh_h2o = 2257000;

% specific heat J / kg / degC
cp_body = 3470;

% density of blood = kg / m^3
rho_bl = 1060;
rho_blml = 1060 / 10^6;

% Calculate flows
dAoP = ezdiff(AoP, step);
% dAoP = [dAoP; 2*dAoP(end)-dAoP(end-1)]; % tack on last point
% dAoP = [dAoP(1); dAoP]; % tack on first point
% plot(dAoP, AoP)

% Qc is the volumetric flow through the capillary, mL/s
Qc = Io - Ca * dAoP;

Pca = (AoP - Qc*Ra);
Pcv = (Pca - Qc*Rc);
mPAC = (Pca + LAP) / 2;
mPAC = (Pca + Pcv) / 2;
filtnet = (mPAC - Pflu) + (Pflu - Piplas);
filtratev = filtnet * 6.67/60; % 6.67 mL / (min * mmHg) * (1 min / 60 sec)

filtratev(find(filtratev < 0)) = 0;
filtrate = mean(filtratev);
lymphfilt = 1/30 * zeros(length(filtrate));

% Isweat = zeros(length(lymphfilt),1);
% for j = 1:length(filtrate)
%     if lymphfilt(j,1) >= (1/60)
%         Isweat(j,1) = (filtrate(j,1) - (1/60));
%     end
% end
% Ii = Ii - lymphfilt;

% Qlat = 0;
AVO2 = (MQ + EWork) / (K1 * mean(Qc));
AVO2(find(AVO2 >= 0.2)) = 0.2;
% Awesome script to run arterial properties
function [bpm, delHR, Qlat, BodTempFinal, sweatfilt, Ra, delRa, Isweat] =
skunkworks(filtrate, bpm, beatnum, MQ, Temperr, BodTemp, art_props, ...
~, BodTempTgt, step, mass, ctrl_bits, ...)
beattendtime, bpmmax, HRKp, HRKi, RAKp, RAKi)

%% variable assignment
ctrl_BPM = ctrl_bits(:,1);
ctrl_Ra = ctrl_bits(:,2);

% Arterial parameters
Ra = art_props(:,1);
Rc = art_props(:,2);  %0.1
Rv = art_props(:,3);  %0.5
Cv = art_props(:,4);  %15
Ca = art_props(:,5);

% % time constants, seconds (Richard 2004, Yoshida 1994)
% tauAVO2 = 39;
% tauVO2on = 33.88;
% tauVO2off = 37.22;
% tauCOon = 29.43;
% tauCOoff = 44.28;

% enthalpy of water J / kg
delh_h2o = 2257000;

% specific heat J / kg / degC
cp_body = 3470;

% density of blood = kg / m^3
rho_bl = 1060;
% rho_bl = 0;
rho_blml = rho_bl / 10^6;

% Temperr
if beatnum == 1
  beatdt = beatendtime(beatnum,1);
else
  beatdt = beatendtime(beatnum,1) - beatendtime(beatnum-1,1);
end

% Calculate body temperature & its error
Isweat = ((MQ / rho_blml / delh_h2o) + (BodTemp - BodTempTgt) * cp_body * mass) / rho_blml / delh_h2o; % plus body heat

% Mdot - mdot_s* delh_h2o = mass * cp * dBodTemp
% Mdot - mass * cp *dT / delh_h2o = sweat_mass_flow
% if mean(Isweat) > mean(filtrate)*0.98
if mean(filtrate) < 1/30
    sweatfilt = 0;
else
    sweatfilt = filtrate - 1/30;
end

Qlat = sweatfilt * rho_blml * delh_h2o;

dBodTemp = (MQ - Qlat) / (mass * cp_body);

BodTempFinal = dBodTemp*beatdt + BodTemp;

dBodTempAvg = mean(dBodTemp);

if beatnum >= 3
    iHRErr = trapz([Temperr(beatnum-2:beatnum-1);dBodTempAvg]);
else
    iHRErr = trapz([Temperr(1:beatnum-1);dBodTempAvg]);
end

if ctrl_BPM == 1
    Kp1 = HRKp;
    Ki1 = HRKi;
else
    Kp1 = 0;
    Ki1 = 0;
end

delHR = Kp1 * dBodTempAvg + Ki1 * iHRErr;  
% trapz([iTemperr;Temperr]);
bpmtemp = bpm + delHR;

% if bpm > 200
%     bpm = 200;
% end

A = 40;
K = bpmmax;
B = 13;
v = 1;
Q = 1;
M = 0.5;

t = bpmtemp/bpmmax;
[dlogibpm, logibpm] = genlogfcn(A, K, B, v, Q, M, t, step);

bpm = bpm + delHR*dlogibpm;

% disp([{'delHR = ' num2str(delHR)}]);
% disp([{'bpm = ' num2str(bpm)}]);
% fprintf(' 

%% Why did the capacitor kiss the diode? Because it couldn't resistor.
% del1Ra = 0;
% dRa = -0.000;

if beatnum >= 3
    iRaErr = trapz([RaErr(beatnum-2:beatnum-1);dRa]);
else
    iRaErr = trapz([RaErr(1:beatnum-1);dRa]);
end

if beatnum >= 3
    iRaErr = trapz([Temperr(beatnum-2:beatnum-1);dBodTempAvg]);
else
    iRaErr = trapz([Temperr(1:beatnum-1);dBodTempAvg]);
end

% Temperr = mean(BodTemp) - 37.5;

if ctrl_Ra == 1
    Kp2 = RAKp;
    Ki2 = RAKi;
else
    Kp2 = 0;
    Ki2 = 0;
end

% Qstored = mass * cp_body * (trapz(dBodTemp) + BodTemp - 37);
% dRa = trapz(Qstored) - (Isweat - lymphfilt)

de1Ra = Kp2 * dBodTempAvg + Ki2 * iRaErr; % trapz([iTemperr;Temperr]);
Ra = Ra - de1Ra*dlogibpm;
% *dlogibpm/4;

if Ra < 0.1
    Ra = 0.1;
end

% disp([{'delRa = ' num2str(delRa)}]);
% disp([{'Ra = ' num2str(Ra)}]);
% fprintf(' 

end
%This is ejecting stacked model

function [dy] = hemodynamics(t, y, z)

% defining variables
pe1     = y(1);
I1      = y(2);
Ii      = y(3);
Io      = y(4);
LVP     = y(5);
V1      = y(6);
Vi      = y(7);
Vo      = y(8);
LAP     = y(9);
AoP     = y(10);
e_1     = y(11);

% resistance dyssynchrony for each section of the heart
r1      = z(1);

% elastance dyssynchrony for each section of the heart
elmin   = z(2);
elmax   = z(3);

% mass dyssynchrony for each section of the heart
m1      = z(4);

% timing dyssynchrony for each section of the heart
tl      = z(5);
bpm     = z(6);

% start and end of beat times from midboss
ts      = z(7);
te      = z(8);

% Defining constants
k1 = r1;        % This is really resistance 1
Ri = 0.005;     % orig 0.005
Ro = 0.01;      % valve resistance
mi = 0.0001;    %0.0002
mo = 0.0001;
Clvp = 0.0001;
m1 = m1;       % just for completeness

% Arterial parameters
Ra = z(9);
Rc = z(10);
Rv = z(11);
Cv = z(12);
Ca = z(13);

% Ra = 0.90933*17*60/1000;       %1.5
Ra = 0.90933*17*60/1000;       %1.5
Rc = 0.07067*17*60/1000;       %0.1
Rv = 0.02*17*60/1000;       %0.5
Cv = 30;       %15
Ca = 1;

% Attain time varying parameters
[el del] = getk(t+t1,elmin,elmax,bpm, ts, te);

%% Heart Chamber Differential Equations
dpel = el*(I1+pel*(1/(el)^2*del));

dI1 = (1/m1)*(LVP-pel-(k1*LVP)*I1);

Do = 20*(-(.15/(.15+exp(-6*Io)))+1); % diode equation

if (LVP > AoP) Do = 0.5; else Do = 1000000; % end

dIo = (1/mo)*(LVP-AoP-(Ro+Do)*Io);

Di = 20*(-(.15/(.15+exp(-6*Ii)))+1); % diode equation

dIi = (1/mi)*(LAP-LVP-(Ri+Di)*Ii);

Ilvp = Ii-Io-I1; % flow balance
dLVP = (1/Clvp)*Ilvp; % for the capacitor

dLAP=(Ii-((AoP-LAP)/(Ra + Rc + Rv)))/Cv;
dAoP=(Io-((AoP-LAP)/(Ra + Rc + Rv)))/Ca;
% disp(['Ra = ' num2str(Ra)]);

% dy = [dpel;dI1;dIi;dIo;dLVP;I1;Ii;Io;dLAP;dAoP;del];
end
function [PE1, I1, Ii, Io, LVP, V1, Vi, Vo, LAP, AoP, e_1] =  
odesolver(init1,init2,vent_props, art_props, bpm)  

PE1 = init1(:,1);  
I1 = init1(:,2);  
Ii = init1(:,3);  
Io = init1(:,4);  
LVP = init1(:,5);  
V1 = init1(:,6);  
Vi = init1(:,7);  
Vo = init1(:,8);  
LAP = init1(:,9);  
AoP = init1(:,10);  
e_1 = init1(:,11);  

r1 = vent_props(:,1);  
elmin = vent_props(:,2);  
elmax = vent_props(:,3);  
m1 = vent_props(:,4);  
t1 = vent_props(:,5);  

Ra = art_props(:,1);  
Rc = art_props(:,2);  
Rv = art_props(:,3);  
Cv = art_props(:,4);  
Ca = art_props(:,5);  

step = init2(:,1);  
pbeatdex = init2(:,2);  
nbeatdex = init2(:,3);  

%% beat length  
ts=pbeatdex*step;  
te=nbeatdex*step;  
t = ts:step:te;  

%% ODE Solver  
OPTIONS=odeset('MaxStep',1e-4);  

[a2, b2]=ode23s(@hemodynamics,t,[PE1 ... % 1 PE1  
I1 ... % 2 I1  
Ii ... % 3 Ii  
Io ... % 4 Io  
LVP ... % 5 LVP  
V1 ... % 6 V1  
Vi ... % 7 Vi  
Vo ... % 8 Vo  
LAP ... % 9 LAP  
AoP ... % 10 AoP  
e_1] ... % 11 e_1  
,OPTIONS, ...  
]  

[r1 ... % 1  
elmin ... % 2  
elmax ... % 3  

61
m1 ... % 4
t1 ... % 5
bpm ... % 6
ts ... % 7
te ... % 8
Ra ... % 9
Rc ... % 10
Rv ... % 11
Ca ... % 12
Cv); ... % 13

%% Output
PE1 = b2(:,1);
I1 = b2(:,2);
Ii = b2(:,3);
Io = b2(:,4);
LVP = b2(:,5);
V1 = b2(:,6);
V1 = b2(:,7);
Vo = b2(:,8);
LAP = b2(:,9);
AoP = b2(:,10);
e_1 = b2(:,11);
getk.m

function [k,dk] = getk(t2,Emin,Emax,bpm, ts,te)

t1 = t2 - ts;
a=1; %scales normal distribution to 1
b=.5*60/bpm; % centers the mean at 1/2 of the cycle
c=.13*b;% .23=50% duty cycle, .13= 1/3 duty cycle
%makes the spread of curve to 50% duty cycle
k=(Emax-Emin)*a*exp(-((t1-b).^2/(2*c.^2)))+Emin;
dk=(Emax-Emin)*a*exp(-((t1-b).^2/(2*c.^2))).*(-2*(t1-b)/(2*c.^2));

end
% -----------------------------------------------
% % Author:   Drew Taylor
% % Date:     Feb 09, 2015
% % Last Rev: Feb 09, 2015
% % Title:    genlogfcn.m
% %
% % Growth is never by mere chance; it is
% % the result of forces working together.
% %-----------------------------------------------
% % Richards' Curve

function [dy,y] = genlogfcn(A, K, B, v, Q, M, t, step)

% A is the lower asymptote (horizontal)
% K is the upper asymptote (horizontal)
% B is the growth rate; higher values increase max(dy)
% v shifts max(dy) along the abscissa
% Q changes the curviness of the sigmoid; higher values have lower max
% growth
% M shifts max(dy) along the abscissa

if length(t) >= 2
    dt = t(2) - t(1);
elseif (exist('step','var'))
    dt = step;
end

y = A + (K-A)./(1 + Q*exp(-B*(t-M)).^(1/v));

dy = B*Q*(K-A)*exp(-B*(t-M)).^(1/v)./(v*(1+Q*exp(-B*(t-M)).^(1/v)).^2)*dt;
refirugator.m

runspec = [0, 1, 10, 50, 100];
loop_num = 0;

for loopdex3 = 1:length(runspec)
  for loopdex2 = 1:length(runspec)
    for loopdex1 = 1:length(runspec)

      % for loopdex3 = 5
      % for loopdex2 = 2
      % for loopdex1 = 1

      % load matfiles
      clear HRKP ...
      HRKI ...
      RAKp ...
      RAKi ...
      beatendtime(end) ...
      absOS ...
      cpRiseTime_actl ...
      cpRiseTime_calc ...
      cpRiseTimeFull_calc ...
      cpSetlTime_actl ...
      cpSetlTime_calc ...
      cpsigma ...
      cpzeta ...
      cpf_damp ...
      cpw_damp ...
      cpf_natr ...
      cpw_natr ...
      sweat_rate_max ...
      sweat_total ...
      bpm_max ...
      bpm_final ...
      bpm_osc ...
      Ra_max ...
      Ra_final ...
      Ra_osc ...
      cpComment

      % poll hostname from computer
      [~, hostname] = system('hostname');
      hostname = strtrim(hostname);

      % set dropbox path to the figure root directory
      if strcmp('ilikeike',hostname) == 1
        dbxpath = 'C:\Users\Drew\Dropbox\Thesis\Figures\';
        subpath = '\ike\';
      elseif strcmp('PHENOMENALOSCART',hostname) == 1
        dbxpath = 'D:\Dropbox\Thesis\Figures\';
        subpath = '\oscar\';
      elseif strcmp('savvyboyd',hostname) == 1
        dbxpath = 'R:\Dropbox\Thesis\Figures\';
        subpath = '\boyd\';
      else
        dbxpath = 'D:\Dropbox\Thesis\Figures\';
        subpath = '';
% For debugging, this if statement creates a faux runspec to test 1 variable
runspec_debug = exist('runspec', 'var');
if runspec_debug == 0
    loopdex1 = 5;
    loopdex2 = 2;
    loopdex3 = 4;
    runspec = [0, 1, 10, 50, 100];
    loop_num = 0;
end

loop_num = loop_num+1;

strbeatnum = '3000';
strHRKp = num2str(runspec(loopdex1));
strHRKi = num2str(runspec(loopdex2));
strRAKp = num2str(runspec(loopdex3));
strRAKi = '0.1';

% sims are separated into folders with the structure Sim X-###-###-###
if length(strHRKp) < 3
    strHRKp_3dig = strHRKp;
    for h = 1:3-length(strHRKp)
        strHRKp_3dig = ['0',strHRKp_3dig];
    end
else
    strHRKp_3dig = strHRKp;
end

if length(strHRKi) < 3
    strHRKi_3dig = strHRKi;
    for i = 1:3-length(strHRKi)
        strHRKi_3dig = ['0',strHRKi_3dig];
    end
else
    strHRKi_3dig = strHRKi;
end

if length(strRAKp) < 3
    strRAKp_3dig = strRAKp;
    for g = 1:3-length(strRAKp)
        strRAKp_3dig = ['0',strRAKp_3dig];
    end
else
    strRAKp_3dig = strRAKp;
end

subfolder1 = [ 'Ki-Ra ', strRAKi, '\'];
subfolder2 = [ 'Sim X-', strHRKi_3dig, '-', strRAKp_3dig, '-', ...
              strRAKi, '\'];
filename_fig1 = [strbeatnum, ' beats GLF - HR Kp ', strHRKp, ' Ki ', ...
                 strHRKi, ' - Ra Kp ', strRAKp, ' Ki ', strRAKi, '.mat'];
mat_filepath = [dbxpath, subfolder1, subfolder2, filename_fig1];
load(mat_filepath, 'beatendtime', 'vsweatfilt', 'vAVO2', 'vBodTemp',...
    'vbpm', 'vRa', 'vEa_RC', 'vEa_SG', 'vEa_EQ')

%%

close all

% if logmode ~= 1;
%     t_tot = length(vAoP);
% t_tot2 = interp1(vQc,0:step:beats);

%% plotting

    h1 = figure(1);
    subplot(2,2,1)
    plot(beatendtime, vsweatfilt)
    title('Sweat Filtration Rate')
    xlabel('Time (s)')
    ylabel('Filtration rate (mL / s)')

    % figure
    subplot(2,2,2)
    % figure
    plot(beatendtime, vAVO2,'b');
    title('AVO_2')
    xlabel('Time (s)')
    ylabel('Concentration (mL / 100 mL)')

    % figure
    subplot(2,2,3)
    plot(beatendtime, vBodTemp)
    title('Body Temperature')
    xlabel('Time (s)')
    ylabel('Temperature (deg C)')

    % figure
    subplot(2,2,4)
% this set of instructions plots Ra and BPM on one plot
    x1 = beatendtime;
    y1 = vbpm;
    x2 = beatendtime;
    y2 = vRa;

    [hax,hL1,hL2] = plotyy(x1,y1,x2,y2);

    set(hax(1),'XColor',[.8 0 0],'YColor',[.8 0 0])
    set(hax(2),'XColor','k','YColor','k')
    set(hL1,'Color','red')
    set(hL2,'Color','black')

    title('Heart Rate & Arterial Resistance')
    ylabel(hax(1),'Heart Rate (BPM)')
    ylabel(hax(2),'Resistance (PRU)')
    xlabel('Time (s)')

% plot Ea
    h2 = figure(2);
    plot(beatendtime, vEa_RC, 'r', beatendtime, vEa_EQ, 'k', beatendtime, vEa_SG, 'c')
Effective Arterial Elastances

xlabel('Time (s)')
ylabel('Ea (mL / s)')
legend('E_A (P_E_S/SV)','E_A (Eqn)','E_A (Segers)')

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% figure
% plot(vbpm)
% ylabel('BPM')
% fprintf('
')

% figure
% plot(vbeattime, vAoP,'b'); hold on;
% plot(vbeattime, vLVP,'g');
% plot(vbeattime, vIo/10,'r');
% plot(vbeattime, vII/10,'k');
% plot(vbeattime, vV1,'k');
% plot(beatendtime, vPca,'c');
% plot(vbeattime, vLAP, 'm');hold off;
% legend('AoP','LVP','I1','V1','Pca','LAP','Location','NorthEastOutside')
% title('AoP / LVP')
% xlabel('Time')
% ylabel('Pressure (mmHg)')

% figure
% plot(vIi)
% xlabel('Time (ms)')
% ylabel('Ii ml/s')
% title('Cardiac Output')

% figure
% plot(vAoP)
% xlabel('Time (ms)')
% ylabel('Aop (mmHg)')
% title('Blood Pressure')

% figure
% plot(vV1,vLVP)
% xlabel('Volume (mL)')
% ylabel('Pressure (mmHg)')
% title('PV Loop')
% end
pause on

filename_fig1 = [strbeatnum, ' beats GLF - HR Kp ', strHRKp, ' Ki ', ...
strHRKi, ' - Ra Kp ', strRAKp, ' Ki ', strRAKi, '.fig'];

fig1_filename = [dbxpath, subfolder1, subfolder2, filename_fig1];
saveas(h1,fig1_filename);

filename_fig2 = [strbeatnum, ' beats GLF - HR Kp ', strHRKp, ' Ki ', ...
strHRKi, ' - Ra Kp ', strRAKp, ' Ki ', strRAKi, ' EA.fig'];

fig2_filename = [dbxpath, subfolder1, subfolder2, filename_fig2];
saveas(h2,fig2_filename);

pause(3)
pause off
der
end
der
end
```matlab
plotter.m

rayHRKp=cell2mat(csv_array(:,1));
rayHRKi=cell2mat(csv_array(:,2));
rayRAKp=cell2mat(csv_array(:,3));
rayRAKi=cell2mat(csv_array(:,4));
rayBeatEndTime=cell2mat(csv_array(:,5));
raySettlingTimeActl=cell2mat(csv_array(:,10));

xrayHRKp=zeros(length(rayHRKp),1);
xrayHRKi=zeros(length(rayHRKi),1);
xrayRAKp=zeros(length(rayRAKp),1);
%xrayRAKi=zeros(length(rayRAKi),1);
xrayBeatEndTime=zeros(length(rayBeatEndTime),1);
xraySettlingTimeActl=zeros(length(raySettlingTimeActl),1);

% axis([0 100 0 2800]);
% xtick = [0,1,10,50,100];
% xtick_label = ['0 ';'1 ';'10 ';'50 ';'100 '];
% set(gca','XTick',xtick,'XTicklabel',xtick_label,'xscale','log');

runspec = [0, 1, 10, 50, 100];
xrayMeanSettlingTime = zeros(length(runspec),1);

for k = 1:length(runspec)
    A = runspec(k);
    xrayHRKp(find(rayHRKp==runspec(length(runspec)\-k+1)),1) ...
    = length(runspec)\-k+1;
    xrayHRKi(find(rayHRKi==runspec(length(runspec)\-k+1)),1) ...
    = length(runspec)\-k+1;
    xrayRAKp(find(rayRAKp==runspec(length(runspec)\-k+1)),1) ...
    = length(runspec)\-k+1;
end

xset = [xrayHRKp xrayHRKi xrayRAKp rayBeatEndTime raySettlingTimeActl];
xrayHRKp_n0 = xrayHRKp;
xrayHRKp_n0(raySettlingTimeActl==0) = [];
xrayHRKi_n0 = xrayHRKi;
xrayHRKi_n0(raySettlingTimeActl==0) = [];
xrayRAKp_n0 = xrayRAKp;
xrayRAKp_n0(raySettlingTimeActl==0) = [];
raySettlingTimeActl_n0 = raySettlingTimeActl(raySettlingTimeActl~=0);

for j = 1:length(runspec)
    xrayMeanSettlingTimeHRKp_n0(j,1) = ...
    mean(raySettlingTimeActl_n0(find(xrayHRKp_n0==j)));
    xrayMeanSettlingTimeHRKi_n0(j,1) = ...
    mean(raySettlingTimeActl_n0(find(xrayHRKi_n0==j)));
    xrayMeanSettlingTimeRAKp_n0(j,1) = ...
    mean(raySettlingTimeActl_n0(find(xrayRAKp_n0==j)));
end

xrayMeanLength = 1:length(runspec);
```

70
figure(1)
plot(xrayHRKp_n0,raySettlingTimeActl_n0,'k.', ...
     xrayMeanLength,xrayMeanSettlingTimeHRKp_n0,'k^--')
title('Settling Time as a Function of HR K_p')
xlabel('HR K_p')
ylabel('Settling Time (s)')
axis([0.9 5.1 0 2800])
set(gca,'XTick',[1 2 3 4 5])
set(gca,'XTickLabel',[0 1 10 50 100])
set(groot, 'DefaultTextFontSmoothing', 'off');
set(gca, 'DefaultAxesFontSmoothing', 'off');

figure(2)
plot(xrayHRKi_n0,raySettlingTimeActl_n0,'k.', ...
     xrayMeanLength,xrayMeanSettlingTimeHRKi_n0,'k^--')
title('Settling Time as a Function of HR K_i')
xlabel('HR K_i')
ylabel('Settling Time (s)')
axis([0.9 5.1 0 2800])
set(gca,'XTick',[1 2 3 4 5])
set(gca,'XTickLabel',[0 1 10 50 100])
set(gca, 'DefaultTextFontSmoothing', 'off');
set(gca, 'DefaultAxesFontSmoothing', 'off');

figure(3)
plot(xrayRAKp_n0,raySettlingTimeActl_n0,'k.', ...
     xrayMeanLength,xrayMeanSettlingTimeRAKp_n0,'k^--')
title('Settling Time as a Function of RA K_p')
xlabel('RA K_p')
ylabel('Settling Time (s)')
axis([0.9 5.1 0 2800])
set(gca,'XTick',[1 2 3 4 5])
set(gca,'XTickLabel',[0 1 10 50 100])
set(gca, 'DefaultTextFontSmoothing', 'off');
set(gca, 'DefaultAxesFontSmoothing', 'off');
% This script has been built to load .mat files generated by capinator and % calculate several new variables: overshoot, settling time, rise time, % damping ratio, etc.

clear all

% These are the values of the control variables
runspec = [0, 1, 10, 50, 100];
loop_num = 0;

% % RAKp
% for loopdex3 = 1:length(runspec)
% % HRKi
% for loopdex2 = 1:length(runspec)
% % HRKp
% for loopdex1 = 1:length(runspec)

for loopdex3 = 1:length(runspec)
for loopdex2 = 1:length(runspec)
for loopdex1 = 3

% load matfiles
clear HRKp ...
   HRKi ...
   RAKp ...
   RAKi ...
   beatendtime(end) ...
   absOS ...
   cpRiseTime_actl ...
   cpRiseTime_calc ...
   cpRiseTimeFull_calc ...
   cpSetlTime_actl ...
   cpSetlTime_calc ...
   cpsigma ...
   cpzeta ...
   cpf_damp ...

%
cpw_damp ...
cpf_natr ...
cpw_natr ...
sweat_rate_max ...
sweat_total ...
bpm_max ...
bpm_final ...
bpm_osc ...
Ra_max ...
Ra_final ...
Ra_osc ...
cpComment

% poll hostname from computer
[~, hostname] = system('hostname');
hostname = strtrim(hostname);

% set dropbox path to the figure root directory
if strcmp('ilikeike',hostname) == 1
    dbxpath = 'C:\Users\Drew\Dropbox\Thesis\Figures\';
    subpath = '\ike\';
elseif strcmp('phenomenaloscar',hostname) == 1
    dbxpath = 'I:\Dropbox\Thesis\Figures\';
    subpath = '\oscar\';
elseif strcmp('savvyboyd',hostname) == 1
    dbxpath = 'R:\Dropbox\Thesis\Figures\';
    subpath = '\boyd\';
elseif strcmp('gregariousfrank',hostname) == 1
    dbxpath = 'C:\Dropbox\Thesis\Figures\';
    subpath = '\frank\';
end

% For debugging, this if statement creates a faux runspec to test 1
% variable
runspec_debug = exist('runspec', 'var');
if runspec_debug == 0
    loopdex1 = 3;
    loopdex2 = 2;
    loopdex3 = 4;
    runspec = [0, 1, 10, 50, 100];
    loop_num = 0;
end

strbeatnum = '3000';
strHRKp = num2str(runspecloopdex1));
strHRKi = num2str(runspecloopdex2));
strRAKp = num2str(runspecloopdex3));
strRAKi = '0.1';

% sims are separated into folders with the structure Sim X-###-####-##.
if length(strHRKp) < 3
    strHRKp_3dig = strHRKp;
    for h = 1:3-length(strHRKp)
        strHRKp_3dig = ['0',strHRKp_3dig];
    end
end
else
    strHRKp_3dig = strHRKp;
end

if length(strHRKi) < 3
    strHRKi_3dig = strHRKi;
    for i = 1:3-length(strHRKi)
        strHRKi_3dig = ['0',strHRKi_3dig];
    end
else
    strHRKi_3dig = strHRKi;
end

if length(strRAKp) < 3
    strRAKp_3dig = strRAKp;
    for g = 1:3-length(strRAKp)
        strRAKp_3dig = ['0',strRAKp_3dig];
    end
else
    strRAKp_3dig = strRAKp;
end

if temp_match == 0
    subfolder1 = ['Ki-Ra ', strRAKi, '];
else
    subfolder1 = ['Ki-Ra ', strRAKi, ' Temp Match\'];
end

% subfolder2 = ['Temp Match Over X-', strHRKi_3dig, '-', strRAKp_3dig, '-
%                   strRAKi, '];
subfolder2 = ['Sim X-', strHRKi_3dig, '-', strRAKp_3dig, '-
                   strRAKi, '];
filename = [strbeatnum, ' beats GLF - HR Kp ', strHRKp, ' Ki ', ...
                   strHRKi, ' - Ra Kp ', strRAKp, ' Ki ', strRAKi, '.mat'];

mat_filepath = [dbxpath, subfolder1, subfolder2, filename];

if exist(mat_filepath,'file') == 2
    loop_num = loop_num+1;
else
    continue
end

if temp_match == 1
    load(mat_filepath, 'vBodTemp', ... 'BodTempTgt', ...
             'beatendtime', ... 'beatnum', ...
             'HRKp', ... 'HRKi', ...
             'RAKp', ... 'RAKi', ...
             'step', ... 'vsweatfilt', ...
'vbpm', ...
'vRa');
else
    load(mat_filepath, 'vBodTemp', ...
'BodTempTgt', ...
'beatendtime', ...
'beatnum', ...
'HRKp', ...
'HRKi', ...
'RAKp', ...
'RAKi', ...
'step', ...
'vesweatfilt', ...
'vbpm', ...
'vRa', ...
'vtempestN', ...
'vBodTempFit');
end

%% Will it oscillate? Presented by Drewtec
vtempestNx=[0 300 600 900 1200 1500 1800 2300];
vtempestNy=[37.35 38.2 38.35 38.6 39.0 39.4 39.6 40.2];
ml = polyfit(vtempestNx,vtempestNy,1);
vtempestN=ml(1).*beatendtime+vtempestNy(1);

m2 = polyfit(beatendtime,vBodTemp,1);
vBodTempFit = m2(1).*beatendtime+m2(2);

% Set error band to 2% of BodTempTgt
err_ss = 0.02;
temp0 = vBodTemp(1);
err_min = BodTempTgt-abs(temp0 - BodTempTgt)*(err_ss);
err_max = BodTempTgt+abs(temp0 - BodTempTgt)*(err_ss);
trc_min = BodTempTgt-abs(temp0 - BodTempTgt)*(0.1); % unused since bias>sp
trc_max = BodTempTgt+abs(temp0 - BodTempTgt)*(0.1);

% calculate constants from sim
sweat_rate_max = max(vsweatfilt);
sweat_total = trapz(beatendtime,vsweatfilt);

% record bpm stats
bpm_max = max(vbpm);
bpm_min = min(vbpm);
bpm_final = vbpm(end);

% check if bpm oscillates
[bpmMinY,bpmMinX] = findpeaks(-vbpm);
bpmMinY = -bpmMinY;
if isempty(bpmMinX)
    bpm_osc = 'no';
elseif length(bpmMinX)<2
bpm_osc = 'no';
else
  bpm_osc = 'yes';
end

% record Ra stats
Ra_max = max(vRa);
Ra_min = min(vRa);
Ra_final = vRa(end);

% check if Ra oscillates
[Ra_minY, Ra_minX] = findpeaks(vRa);
if isempty(Ra_minX)
  Ra_osc = 'no';
elseif length(Ra_minX)<2
  Ra_osc = 'no';
else
  Ra_osc = 'yes';
end

% Test if body temp ever decreases during the trial
decdex = find(vBodTemp<vBodTemp(1));
test1 = isempty(decdex);

T_final = vBodTemp(end);

[tempMaxY, tempMaxX] = findpeaks(vBodTemp);
[tempMinY, tempMinX] = findpeaks(-vBodTemp);
tempMinY = -tempMinY;
if test1 == 0
  % Overshoot/undershoot test
  absOS = BodTempTgt - min(vBodTemp);
  if absOS > 0 % case: underdamped
    % find local extrema
    cpComment = 'underdamped';
    if length(tempMinY) >= 2
      pt1_mag = tempMinY(1);
      pt2_mag = tempMinY(2);
      period = (beatendtime(tempMinX(2)) - ...
        beatendtime(tempMinX(1)));
      cpsigma = log(pt2_mag/pt1_mag);
      cpzeta = (1+(2*pi/cpsigma)^2)^(-1/2);
      cpf_damp = 1/period;
      cpf_natr = cpf_damp*(1-cpzeta^2)^(-1/2);
      cpw_damp = 2*pi*cpf_damp;
      cpw_natr = 2*pi*cpf_natr;
\[
\text{cpSetlTime}\_\text{calc} = -\log(\text{err}\_ss)/(\text{cpzeta}\*\text{cpw}\_\text{natr}); \\
\text{cpRiseTime}\_\text{calc} = (2.23*\text{cpzeta}^2 - 0.078*\text{cpzeta} + 1.12)/\text{cpw}\_\text{natr}; \\
\text{cpRiseTimeFull}\_\text{calc} = (1/\text{cpw}\_\text{natr})*(1-\text{cpzeta}^2)^{(-1/2)}* \ldots \\
(\pi - \arctan(\sqrt{1-\text{cpzeta}^2}/\text{cpzeta})); \\
\]

% determine if the signal converges or oscillates "forever"
\[
\text{tempMinY}\_\text{err} = \text{zeros}(\text{length}(\text{tempMinY})-1,1); \\
\text{k} = 1:\text{length}(\text{tempMinY})-1 \\
\text{tempMinY}\_\text{err}(:,k) = \text{abs}(\text{tempMinY}(k)-\text{tempMinY}(k+1))/\text{tempMinY}(k); \\
\text{if } \text{tempMinY}\_\text{err}(k) < 0.02 \\
\quad \text{inf}_\text{osc} = 0; \\
\text{elseif } \text{tempMinY}\_\text{err}(k) > 0.02 \\
\quad \text{inf}_\text{osc} = 1; \\
\text{end} \\
\text{end} \\
\]

% find the last value outside error tolerance
\[
\text{if inf}_\text{osc} == 0 \\
\quad \text{setl}\_\text{ubound}\_\text{dex} = \text{find}(\text{vbodTemp}>\text{err}\_\text{max},1,'\text{last}')+1; \\
\quad \text{setl}\_\text{lbound}\_\text{dex} = \text{find}(\text{vbodTemp}<\text{err}\_\text{min},1,'\text{last}')+1; \\
\text{if } \text{setl}\_\text{ubound}\_\text{dex} > \text{length}(\text{beatendtime}) \\
\quad \text{setl}\_\text{ubound}\_\text{time} = \text{beatendtime}(\text{length}(\text{beatendtime})); \\
\text{else} \\
\quad \text{setl}\_\text{ubound}\_\text{time} = \text{beatendtime}(\text{setl}\_\text{ubound}\_\text{dex}); \\
\text{end} \\
\text{if } \text{setl}\_\text{lbound}\_\text{dex} > \text{length}(\text{beatendtime}) \\
\quad \text{setl}\_\text{lbound}\_\text{time} = \text{beatendtime}(\text{length}(\text{beatendtime})); \\
\text{else} \\
\quad \text{setl}\_\text{lbound}\_\text{time} = \text{beatendtime}(\text{setl}\_\text{lbound}\_\text{dex}); \\
\text{end} \\
\text{if isempty(setl}\_\text{ubound}\_\text{time}) && isempty(setl}\_\text{lbound}\_\text{time}) \\
\quad \text{cpSetlTime}\_\text{act}\_\text{l} = 0; \\
\text{elseif setl}\_\text{ubound}\_\text{time} >= setl\_\text{lbound}\_\text{time} \\
\quad \text{cpSetlTime}\_\text{act}\_\text{l} = \text{setl}\_\text{ubound}\_\text{time}; \\
\text{else} \\
\quad \text{cpSetlTime}\_\text{act}\_\text{l} = \text{setl}\_\text{lbound}\_\text{time}; \\
\text{end} \\
\text{else} \\
\quad \text{tempMinY}\_\text{osc}\_\text{within}\_\text{tol} = \text{true}; \\
\quad \text{if isempty(tempMinY)} == 0 \\
\quad \quad \text{if tempMinY(1) < err}\_\text{min} \\
\quad \quad \quad \text{tempMinY}\_\text{osc}\_\text{within}\_\text{tol} = \text{false}; \\
\quad \quad \quad \text{break} \\
\quad \quad \text{end} \\
\quad \text{end} \\
\text{if tempMinY}\_\text{osc}\_\text{within}\_\text{tol} == \text{true};
cpSetlTime_actl = beatendtime(find(vBodTemp<err_max,1,'first'));
else
    cpSetlTime_actl = 0;
    cpComment = [cpComment, '; osc. to inf.'];
end

cpRiseTime_actl = beatendtime(find(vBodTemp<trc_max,1,'first'));

else % if fewer than 1 full period are produced, this fork
    cpComment = [cpComment, '; too slow to calc pt1/pt2'];
    pt1_mag = 0;
    pt2_mag = 0;
    period = 0;
    cpRiseTime_actl = beatendtime(find(vBodTemp<trc_max,1,'first'));
    cpSetlTime_actl = beatendtime(find(vBodTemp<err_max,1,'first'));
    cpSetlTime_calc = 0;
    cpRiseTime_calc = 0;
    cpRiseTimeFull_calc = 0;
    cpsigma = 0;
    cpzeta = 0;
    cpf_damp = 0;
    cpf_natr = 0;
    cpw_damp = 0;
    cpw_natr = 0;
end

elseif absOS <= 0 % case: over/critical
    cpComment = 'overdamped';

    cpSetlTime_actl = beatendtime(find(vBodTemp<err_max,1,'first'));
    if isempty(cpSetlTime_actl)
        cpSetlTime_actl = 0;
    end

    cpRiseTime_actl = beatendtime(find(vBodTemp<trc_max,1,'first'));
    if isempty(cpRiseTime_actl)
        cpRiseTime_actl = 0;
    end

    cpSetlTime_calc = 0;
    cpRiseTime_calc = 0;
    cpRiseTimeFull_calc = 0;
    cpsigma = 0;
    cpzeta = 0;
    cpf_damp = 0;
    cpf_natr = 0;
    cpw_damp = 0;
    cpw_natr = 0;
end
% extremaX = sort([tempMaxX;tempMinX]);
% extremaY = sort([tempMaxY;tempMinY]);

if runspec_debug == 0;
    decimalpts = 2;
    xmin = 0;
    xmax = max(beatendtime);
    ymin = floor(min(vBodTemp)*10^(decimalpts)) / 10^decimalpts;
    ymax = ceil(max(vBodTemp)*10^(decimalpts-1)) / 10^(decimalpts-1)+10^-(decimalpts+10);
    close all
    hold on
    plot(beatendtime,vBodTemp)
    plot(beatendtime(tempMaxX),tempMaxY,'k*')
    plot(beatendtime(tempMinX),tempMinY,'k*')
    plot(beatendtime,linspace(err_max,err_max,length(beatendtime)),'r--')
    plot(beatendtime,linspace(err_min,err_min,length(beatendtime)),'r--')
    axis([xmin xmax ymin ymax])
    hold off
end

else % for this case, BodTemp never goes down
    absOS = BodTempTgt - min(vBodTemp);
    cpRiseTime_actl = 0;
    cpRiseTime_calc = 0;
    cpRiseTimeFull_calc = 0;
    cpSetlTime_actl = 0;
    cpSetlTime_calc = 0;
    cpsigma = 0;
    cpzeta = 0;
    cpf_damp = 0;
    cpw_damp = 0;
    cpf_natr = 0;
    cpw_natr = 0;
    cpComment = ['Body temperature never decreases within ', ...
                  num2str(beatnum),' beats.'];
    if runspec_debug == 0
        plot(beatendtime,vBodTemp)
    end
end

disp([strHRKp_3dig,'-',strHRKi_3dig,'-',strRAKp_3dig,'-',strRAKi,' (', ...
     num2str(loop_num),')'])

[r1, p1] = corrcoef(vtempestN,vBodTemp);
[r2, p2] = corrcoef(vtempestN,vBodTempFit);
R_real = r1(1,2);
R_fit = r2(1,2);

temp_match = 1;

if temp_match == 0
cp_array = [HRKp, ...
HRKi, ...
RAKp, ...
RAKi, ...
beatendtime(end), ...
absOS, ...
cpRiseTime_actl, ...
cpRiseTime_calc, ...
cpRiseTimeFull_calc, ...
cpSetlTime_actl, ...
cpSetlTime_calc, ...
cpsigma, ...
cpzeta, ...
cpf_damp, ...
cpw_damp, ...
cpf_natr, ...
cpw_natr, ...
sweat_rate_max, ...
sweat_total, ...
bpm_min, ...
bpm_max, ...
bpm_final, ...
{bpm_osc}, ...
Ra_min, ...
Ra_max, ...
Ra_final, ...
{Ra_osc}, ...
T_final, ...
{cpComment}];

else
    cp_array = [HRKp, ...
HRKi, ...
RAKp, ...
RAKi, ...
beatendtime(end), ...
cpRiseTime_actl, ...
cpRiseTime_calc, ...
sweat_rate_max, ...
sweat_total, ...
bpm_min, ...
bpm_max, ...
bpm_final, ...
{bpm_osc}, ...
Ra_min, ...
Ra_max, ...
Ra_final, ...
{Ra_osc}, ...
T_final, ...
R_real, ...
R_fit, ...
{cpComment}];
end

if loop_num == 1
    csv_array = cp_array;
else
    csv_array = [csv_array;cp_array];
end
%% these three end statements terminate the for loops
end
end
end

%% write the variables to a csv

% this array contains the labels for each column

if temp_match == 0
csv_headings = [{'HRKp'}, ...
{'HRKi'}, ...
{'RAKp'}, ...
{'RAKi'}, ...
{'beatendtime(end)'}, ...
{'absOS'}, ...
{'cpRiseTime_actl'}, ...
{'cpRiseTime_calc'}, ...
{'cpRiseTimeFull_calc'}, ...
{'cpSet1Time_actl'}, ...
{'cpSet1Time_calc'}, ...
{'cpsigma'}, ...
{'czeta'}, ...
{'cpf_damp'}, ...
{'cpw_damp'}, ...
{'cpf_natr'}, ...
{'cpw_natr'}, ...
{'sweat_rate_max'}, ...
{'sweat_total'}, ...
{'bpm_min'}, ...
{'bpm_max'}, ...
{'bpm_final'}, ...
{'bpm_osc'}, ...
{'Ra_min'}, ...
{'Ra_max'}, ...
{'Ra_final'}, ...
{'Ra_osc'}, ...
{'T_final'}, ...
{'cpComment'}];
else
csv_headings = [{'HRKp'}, ...
{'HRKi'}, ...
{'RAKp'}, ...
{'RAKi'}, ...
{'beatendtime(end)'}, ...
{'cpRiseTime_actl'}, ...
{'cpSet1Time_actl'}, ...
{'sweat_rate_max'}, ...
{'sweat_total'}, ...
{'bpm_min'}, ...
{'bpm_max'}, ...
{'bpm_final'}, ...
{'bpm_osc'}, ...
{'Ra_min'}, ...
end

81
{'Ra_max'}, ...
{'Ra_final'}, ...
{'Ra_osc'}, ...
{'T_final'}, ...
{'R real'}, ...
{'R fit'}, ...
{'cpComment'}];

end

csv_export = [csv_headings;csv_array];
xlswrite([dbxpath, subfolder1,'temp match HR Kp ', strHRKp, '.xlsx'],csv_export)
ezdiff.m

function y = ezdiff(func,step)

y = 1 / step * diff(func);

end