HART Study: Heart Rate Acquisition and Threshold-Based Ranges for Triathletes

Exercise Physiology Research Laboratory

Author: Jeremy Wadowski
Director: Dr. Christopher Cooper
Associate Director: Dr. Brett A. Dolezal

I. Introduction

The goal of any individualized exercise prescription is to enhance and maximize training efficacy and limit training unresponsiveness. Exercise intensity is arguably the most critical component of the exercise prescription model. In fact, it has been suggested that the method of exercise intensity prescription may underpin the inter-individual variation in cardiorespiratory VO2 response to exercise training (Wolpern et al. 2015). Previous research studies report wide variability in the individual VO2 response to training by using one of several cookie-cutter exercise intensity methods, including %HRmax %HHR or %VO2 max (Bouchard & Rankinen 2001, Skinner et al. 2000). This study will use a novel, empirically-based approach through use of indirect calorimetry and gas exchange by establishing metabolic (lactate) and ventilatory thresholds as lower and upper heart rate demarcations for optimal training zone prescriptions.

Data processing and graphing tasks were performed programmatically using Python. This project is available at: **https://github.com/Jswadowski/eeb-177-final-project**.

II. Purpose

To (i) remotely acquire heart rates of UCLA club-level triathletes during off-season training, (ii) develop heart rate training zones derived from baseline VO2 max, metabolic and ventilatory thresholds (via Oxycon Pro), and (iii) characterize the percentage of time spent in these zones.

III. Description

Eight participants (males=4, females=4) were recruited from the UCLA club-level triathlete team and underwent remote HR monitoring over 8-weeks. Participants donned a forearm-worn optical HR sensor device (Scosche Rhythm) paired to a smartphone that remotely collected, stored, and transferred HRs to a secure encrypted server in the Exercise Physiology Research Laboratory for subsequent analysis. The participants completed a series of three VO2 max assessments on a treadmill (baseline, 4-weeks, 8-weeks) where relative VO2 max was among one of several measurements gathered. Metabolic and ventilatory thresholds correspond to the first and second inflection points, respectively, when VCO2 was plotted against VO2. Subjects were instructed to maintain normal training and diet regimens throughout. Data over the 8-week period including HR, duration and self-reported intensity of exercise from all independent running and cycling training sessions was collected and analyzed.

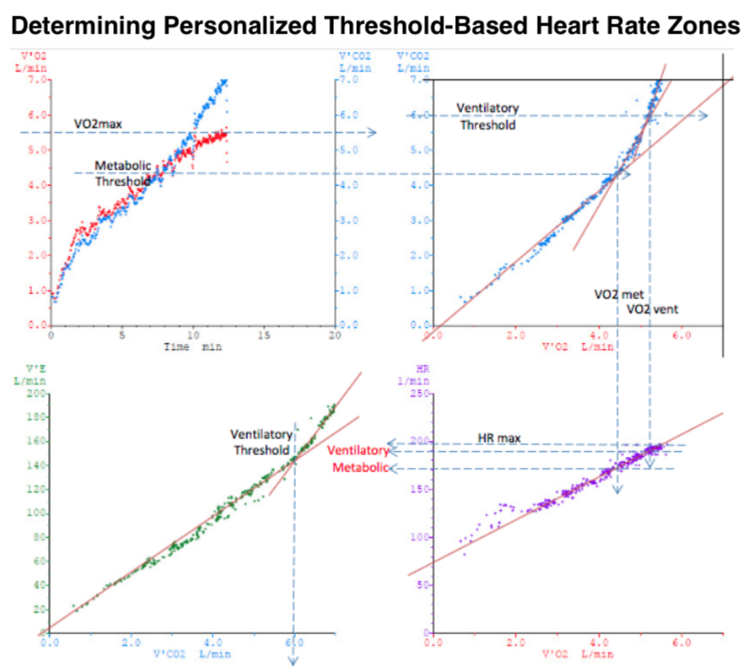


Figure 1: . Exemplary heart rate zone determination. Using this Cooper 4-plot, exact heart rate zones at MT and VT can be established.

IV. Methods

The first step in organizing the raw data after opening and reading in the data file was to make it presentable, i.e. more readable by removing the header line and any empty spaces via the "hr_data" function shown in figure 2. Secondly, the number of total workouts for the individual was determined by finding the length of the row in the function "row2Col". This function is also shown in figure 2. Within the row2Col function, an all-embracing array was created to hold arrays containing elements compromising each individual workout. A single element corresponds to a heart rate beats per minute measurement. The filename and predetermined heart rate range tuple were added to a dictionary as key-value pairs. A user input window for selecting the data file was added for convenience.

```
import csv

def hr_data(filename):
    lines = []
    data = open(filename, "r") #open the file
    all_records = csv.reader(data, delimiter=",")
    #read the file

    #Removing header line
    num_cols = len(next(all_records))
    print("num_cols: " + str(num_cols))
    #removing empty lines and empty spaces
    for i in data:
        i = i.rstrip()
        if i != '':
            points = i.split(",")


            lines.append(points)



    for row in all_records:  #loop over rows
                             #in the list all_records
        lines.append(row)

    return lines

def row2Col(arr):
    #Number of elements in a row
    #(i.e number of exercises)
    count = len(arr[0])
    print("count is: " + str(count))

    #colArr will hold the new list of columns
    colArr = []

    #Initialize the column array to a list of
    #'count' empty lists
    for i in range(count):
        empty = []
        colArr.append(empty)

    #We iterate through each exercise and then add the
    #element to the corresponding column
    for i in range(count):
        for row in arr:
            #Only add non empty element
            if (row[i]!=''):
                colArr[i].append(row[i])

    return colArr


thedict = {"Jay_data.csv": (189,200),
          "Jeremy_data.csv": (165,180),
          "Mila_data.csv": (167,180),
          "Rachel_data.csv": (178,188),
          "Nolan_data.csv": (172,187),
          "Nako_data.csv": (163,180),
          "Jewell_data.csv": (156,176),
          "Dillon_data.csv": (158,175)}

user_choice = input("Choose subject data: Jay_data.csv,
            Jeremy_data.csv, Mila_data.csv, Rachel_data.csv,
            Nolan_data.csv, Nako_data.csv,Jewell_data.csv,
            Dillon_data.csv")
selectfile = hr_data(user_choice)

colPoints = row2Col(selectfile)

print(colPoints)
```

Figure 2: Data formatting and setup functions.

The "calculate" function shown in figure 3 works by taking in two integer inputs: range_low and range_high. These values are the threshold window bounds. The function calculates the percentage of points within the threshold as well as the percentage of points above and below it. Python's easy type-checking was used as opposed to making three separate functions. A tuple containing the three values was returned and the value of interest was extracted upon function return.

To calculate the percentage, the total number of points was calculated by summing the length of every column. In addition, the range is checked for each point in the column to see which range the point falls into (within, above, or below). Each group has its own counter. Upon completion of both loops, the percentages are calculated via the following generic formula:

$$Percentage = \frac{\text{Number of Points Fitting Search Criteria}}{\text{Total Number of Points in Dataset}} * 100 \tag{1}$$

For consistency and documentation purposes, the return value (tuple) formatting was denoted in the comments.

```
#Calculates perc within, perc above, and perc below.
#Input: range_low and range_high for limits
#Output: a tuple in order of (Percent w/in threshold, Perc Above, Perc Below)
def calculate(range_low, range_high):
    count = 0
    count_below = 0
    count_above = 0
    total_points = 0
    for workout in colPoints:

        numPoints = len(workout)
        total_points += numPoints
        print("num points: ", numPoints)
        for current_val_str in workout:
            current_val = float(current_val_str)
            if ((current_val) > float(range_low) and (current_val) < float(range_high)):
                count = count + 1
            if( (current_val ) >= float(range_high)):
                count_above = count_above + 1
            if( (current_val) <= float(range_low) ):
                count_below = count_below + 1

    print("total pts is: " + str(total_points))
    percent = (count/total_points)*100
    percent_above = (count_above/total_points)*100
    percent_below = (count_below/total_points)*100
    return (percent, percent_above, percent_below)
```

Figure 3: Percentage calculator function. Inputs are two integer values for a threshold range and the output is a 3-tuple.

Given the hetergeneous nature of each subjects data file (where none were alike in number of rows nor workout duration), a time array had to be created for independent axis normalization. After finding the length of an individual workout column, that length was added to a length array; a counter was made to count the number of values in a given workout column of specified length. The code block below shows the creation of an array holding dummy values to represent time stamps. This differs based on the dataset being read so the array is created programmatically.

```python
#Creating time array

tinterval=2
tlen=[]
times=[]

for i in range(len(colPoints)):

    #finding the length of a workout column
    tlength=len(colPoints[i])
    #adding the length to an array
    tlen.append(tlength)

    #create an empty array (times) of arrays, the same amount as colPoints.

    times.append([])

#Initiate variable as 0. Want counter to start at 0. (n is our counter)
n=0

#tlen = [L1, L2, L3, ..., L8]
#Where each L value is the number of values in some workout column.
for length in tlen:
    for i in range(length):
        t=tinterval*i
        times[n].append(t)

    n = n + 1
```

Figure 4: Creation of axis spacing for plotting.

Lastly, to begin generating the graph, the time array and array containing the results of hr_data and row2Col were incorporated workout-by-workout to coordinate each dependent y-value to an independent x-value established created based on time axis normalization. Matplotlib.pyplot was used to manage graph specifications (title, labels, text sizes, legend, etc.)

```python
from pylab import *
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import scipy
from scipy import signal

fig = plt.figure()
for i in range(len(colPoints)):
    #times is array of arrays.  times[i] gives you an array consisting of
    #independent (x axis) points for some
    #workout column i

    #colPoints is an array of arrays. colPoints[i] gives you an array
    #containing the dependent (y axis)

    #data for i=0 (first workout), i=2 (second workout), etc.
    #if i=1, i'm looking at the second workout.
    plt.plot(times[i],colPoints[i],label="Workout "+str(i+1),linewidth=0.5)

plt.xlabel('Time(s)',fontsize=20)
plt.ylabel('Heart Rate',fontsize=20)
plt.title('Heart Rate Over Different Workouts',fontsize=30)
plt.rc('xtick',labelsize=7)
plt.rc('ytick',labelsize=7)
legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.5)   #legend placement


#Adding in the percent within threshold calculation
value = thedict[user_choice]
range_low = (value[0])
range_high = (value[1])

#Call calculate. Will return a tuple in order (percentage within, perc above, perc below)
calculated_perc_tuple = calculate(range_low, range_high)
calculated_perc = calculated_perc_tuple[0]
perc_above = calculated_perc_tuple[1]
perc_below = calculated_perc_tuple[2]
print("perc_above: " + str(perc_above))
print("perc_below: " + str(perc_below))

#Adjustable sig figs.
sig_figs = 2
perc_val = round(calculated_perc, sig_figs)
perc_str = "Percent within threshold: " + str(perc_val) + "%"

#Displaying the string 'Percent within threshold' and percentage
ax = fig.add_subplot(111)
ax.text(0.95, 0.02, perc_str, verticalalignment = 'bottom', horizontalalignment = 'right',
        transform=ax.transAxes, color='black', fontsize=13)

plt.axhline(y= range_low, c="black",linewidth=2)
plt.axhline(y= range_high, c="black",linewidth=2)


plt.show()
```

Figure 5: Function call to "calculate" in order to get percentage values. This code also plots the values, formats the axes, and displays custom text and variable percentage value.

The "calculate" function was called to generate percentage values. Since it returns a tuple, each individual (within threshold, above, below) value had to be extracted and stored in its own variable. To physically display the text Percent within threshold and result inside of the graph, matplotlib.patches was used.

V. Results

All of the graphs in this section show color-coded workouts as well as heart-rate thresholds. The lower bound is the heart rate at metabolic threshold, while the upper bound is heart rate at ventilatory threshold. These thresholds were calculated using the Cooper 4-plot method. This is shown in figure 1. The percentage of datapoints within each threshold is shown on each graph.

The tables below show VO2 max data for the test subjects. Data was collected from an even distribution of male and female subjects. There were four male subjects and four female subjects.

Baseline V02 Max	
Subject	VO2 max (ml/kg/min)
1(M)	57.3
2(M)	63.9
3(M)	61.1
4(M)	58.2
5(F)	39.8
6(F)	49.1
7(F)	57.5
8(F)	48.1

Table i: Baseline VO2 max for the eight subjects.

The baseline VO2 max data was collected at the beginning of the study.

4 -Week V02 Max	
Subject	VO2 max (ml/kg/min)
1(M)	-
2(M)	63.9
3(M)	58.0
4(M)	66.5
5(F)	39.1
6(F)	52.1
7(F)	61.0
8(F)	55.3

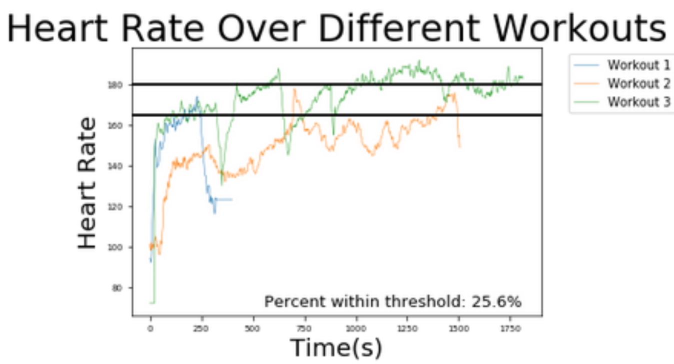
Table ii: 4-week VO2 max for the eight subjects.

Tables ii and iii show the VO2 max data at 4 weeks and at 8 weeks.

8 -Week V02 Max	
Subject	VO2 max (ml/kg/min)
1(M)	-
2(M)	63.9
3(M)	-
4(M)	66.3
5(F)	45.5
6(F)	48.9
7(F)	-
8(F)	53.2

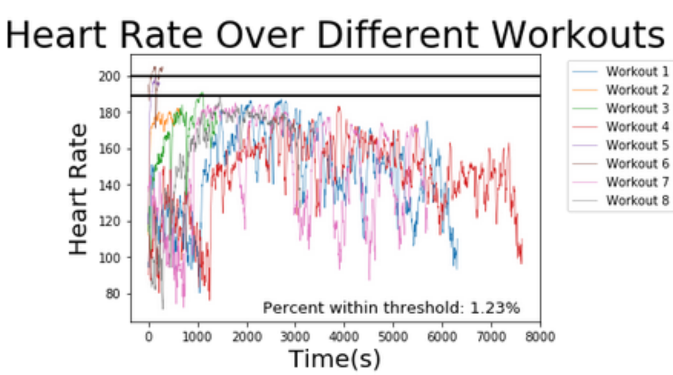
Table iii: 8-week VO2 max for the eight subjects.

The following four male subjects' results are shown in the below graphs. Subject 1's results are shown below in graph 1. This subject had 18.52% of his heart rate data points above the upper threshold line and 55.88% below. The bounded area (area within the thresholds) contained 25.6% of the data points.



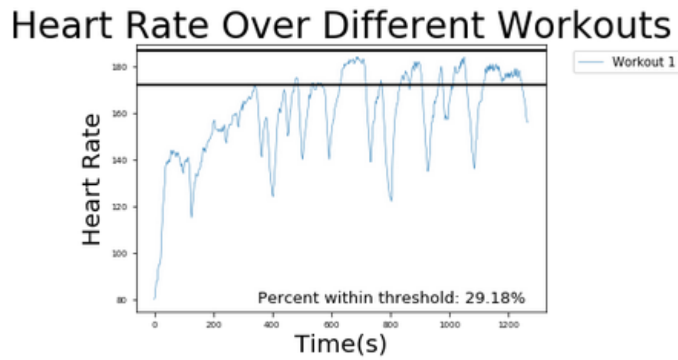
Graph 1: This graph shows the heart rate data for a male test subject (1M).

Subject 2's results are shown below in graph 2. This subject's heart rate samples were primarily below the lower threshold line, with this region encompassing 98.14% of the data points. The area between the thresholds encompassed 1.23% while only 0.62% of the data points exist above the upper threshold.



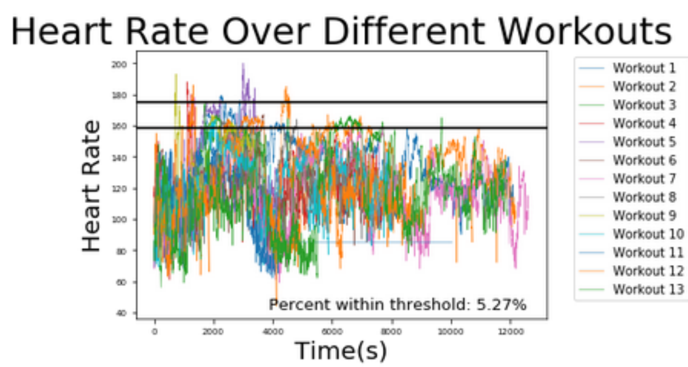
Graph 2: This graph shows the heart rate data for a male test subject (2M).

Subject 3's results are shown below in graph 3. Like subject 2, subject 3 showed a large skew toward the lower region with very little upper and middle region concentration. No data points were recorded above the upper threshold resulting in a 0% value. The area between thresholds contained 29.18% of the data points. The area below the lower threshold contained 70.82% of the data points.



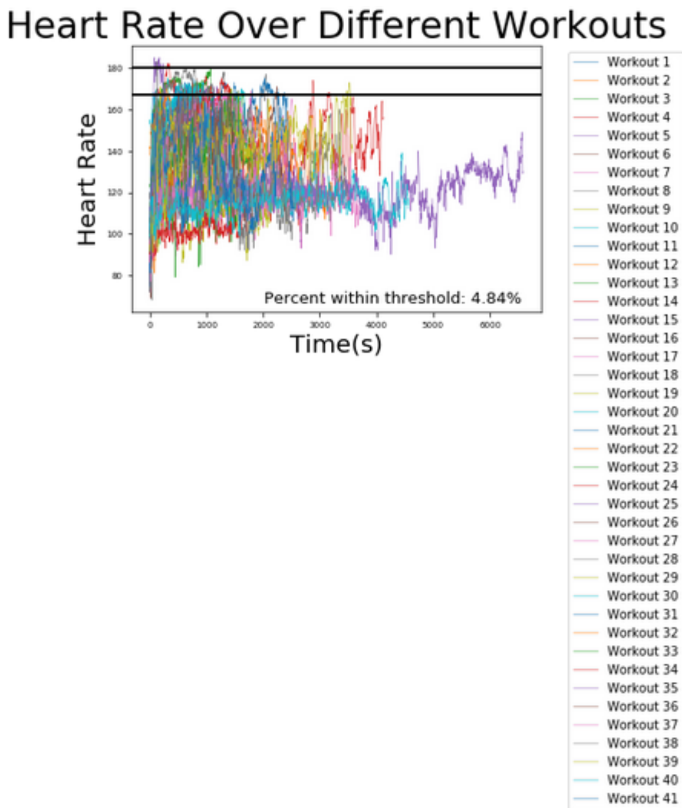
Graph 3: This graph shows the heart rate data for a male test subject (3M).

Subject 4's results are shown below in graph 4. This subject shows an even larger skew toward the lowest region, with the area below the bottom threshold encompassing 94.23% of the data points. The area between the thresholds contains 5.27% of the data points and the area above the upper limit contains 0.5%.



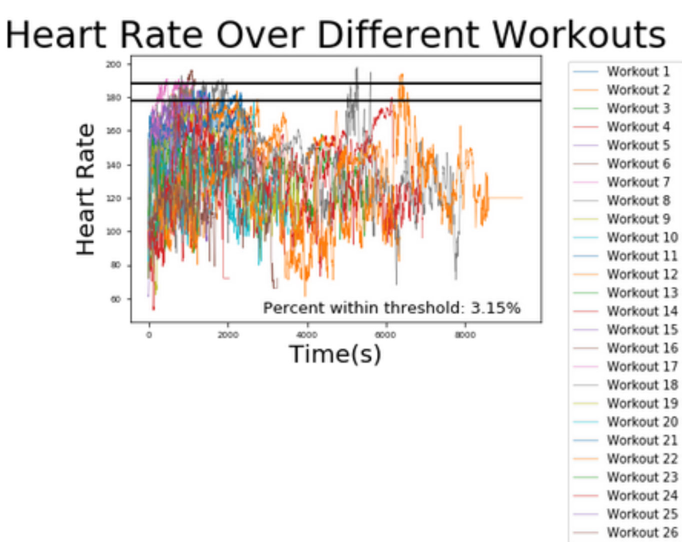
Graph 4: This graph shows the heart rate data for a male test subject (4M).

The following four female subjects' results are shown in the subsequent graphs. Subject 5's results are shown in graph 5. Like subject 3, the skew is shifted toward the lower region with relatively few data points in the bounded region and even less above the upper limit. Only 0.14% of the data points are above the upper threshold whereas 95.02% are contained in the area below the lower threshold. The region bounded by both limits contains the remaining 4.84%.



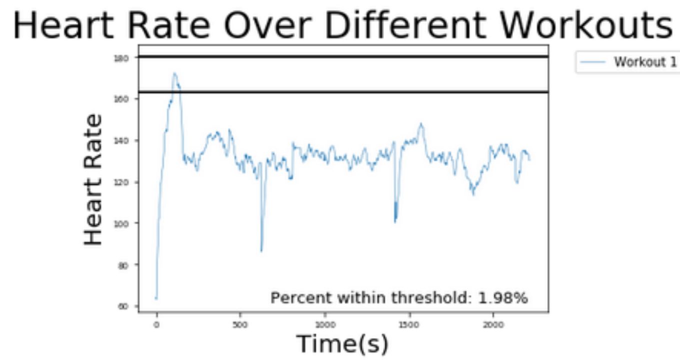
Graph 5: This graph shows the heart rate data for a female test subject (5F).

Subject 6's results are shown in graph 6. The region above the upper limit contained 0.73% of the values, the bounded area contained 3.15%, and the region below the lower limit contained 96.12%.



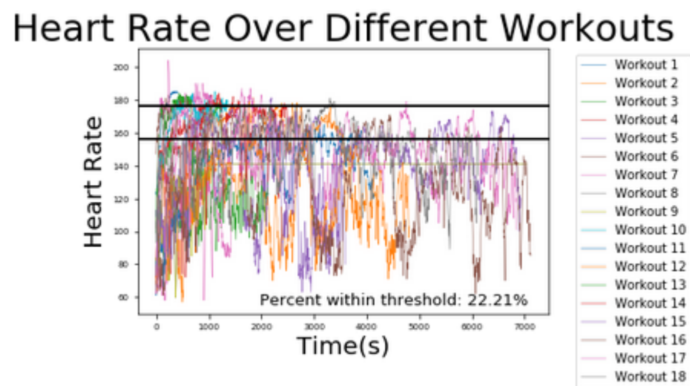
Graph 6: This graph shows the heart rate data for a female test subject (6F).

Subject 7's results are shown in graph 7. No data points (0%) were present above the upper limit line, while 98.02% are under the lower limit line. The bounded area between these limits contains the remaining 1.98% of the heart rate samples.



Graph 7: This graph shows the heart rate data for a female test subject (7F).

Subject 8's results are shown in graph 8. This subject's skew is shifted upwards, with 4.21% of the samples occurring above the upper limit, 22.1% in the bounded region, and 73.58% below the lower limit line.



Graph 8: This graph shows the heart rate data for a female test subject (8F).

VI. Conclusion

All participants VO2 max values were measured to be in the 99th percentile or above of the American College of Sports Medicines age- and gender-matched cardiorespiratory rankings. The usual method of training for these triathletes increased VO2 max over time but tended to plateau or slightly decrease during the latter half of the study. For all subjects, there was little time spent training above the metabolic threshold. Male subjects trained above the metabolic threshold more often than female subjects (14.76% average across all subjects; 20.23% average for men, 9.29% average for women).

Preliminary data in 6 triathletes suggests there may exist a moderate correlation between UCLA club-level triathletes VO2 max and the intensity at which these athletes train. Moreover, some triathletes with a higher VO2 max trended towards spending a greater percentage of time above their metabolic threshold and closer to their ventilatory threshold determined heart rate zones during off-season training.

An opportunity for further research would be to test whether subject threshold-ranges could be increased by providing various workout protocols (e.g. High Intensity Interval Training consisting of 60 second work, 60 second rest, or 3 minute work, 1 minute rest periods).

An area of potential improvement for this study would be to normalize the number of workouts and their duration across all subjects. In addition, it would be beneficial to encourage participants to return for their follow-up tests. A small monetary compensation could be provided as incentive.

VII. References

A. Primary References

Wolpern AE, Burgos DJ, Janot JM, Dalleck LC. Is a threshold-based model a superior method to the relative percent concept for establishing individual exercise intensity? a randomized controlled trial. BMC Sports Science, Medicine and Rehabilitation 7, 2015.

Bouchard C, Rankinen T. Individual differences in response to regular physical activity. Medicine and Science in Sports and Exercise 33: S446451, 2001.

Skinner JS, Wilmore KM, Krasnoff JB, Jasklski A, Jasklski A, Gagnon J, et al. Adaptation to a standardized training program and changes in fitness in a large, heterogeneous population: the HERITAGE Family Study. Medicine & Science in Sports & Exercise 32: 15761, 2000.

B. Supplementary References

Mann T, Lamberts RP, Lambert MI. Methods of Prescribing Relative Exercise Intensity: Physiological and Practical Considerations. Sports Medicine 43: 61325, 2013.

Scott CB, Littlefield ND, Chason JD, Bunker MP, Asselin EM. Differences in oxygen uptake but equivalent energy expenditure between a brief bout of cycling and running. Nutrition & Metabolism 3:1 , 2006.

Stoggl T, Sperlich B. Polarized training has greater impact on key endurance variables than threshold, high intensity, or high volume training. Frontiers in Physiology 2014