week 4

November 22, 2022

Rule et al. Elements, this Document

This notebook follows the following Rule norms for Computatational Narratives:

1. narrative story
2. process documentation
3. cell divisions
4. modularized code
5. recorded dependencies
6. pipeline construction
7. shareability and explainability
8. a design to be read and explored

0.1 High Heart Rate as a Function of Phase, in Workout Regimen: a Directed Graph Analysis

Stefan Bund

University of Michigan, SIADS 521

bund@umich.edu

0.2 Abstract:

In a condition where an athlete's endurance and athletic output improved over time, factors such as distance, altitude and duration are studied, and composed into an ideal strategy for training. The question of what factors contribute most to improved athletic capability are identified in a quantified series of 75 workouts. As a conclusion, an experimental sequence is discovered, where the athlete executed a path of activities, during a workout. As these paths became more sophisticated, they contributed to a linear improvement in output, per workout, and thus contributed toward improving fitness. A Fruchterman-Reingold force-directed algorithm is used to analyze the efficiency by which heart rate could move into the highest state of productivity, at the start, middle and close of a multi-month training regime.

0.3 Introduction

With the arrival of self-quantified fitness devices, data scientists possess a wealth of new and enhanced means of studying athletic performance. Our study was facilitated through one individual's data collection across 75 workouts, where the strava workout app was used to collect data on navigation, speed, as well as data related to altitude and the stride of the runner.

The approach in this study prepped the strava data, and binned each workout into discreet time series. Within each subsampled workout, the heart rate of the athlete was used as an index to the intensity of the workout. Then, metrics in distance and speed were used to study the athlete's overall performance.

Heart rate was found to reach high rates between the start and middle of the overall training regimen. The data charted moving the athlete between exerting large amounts of energy to reach only moderate heart rates. This changed by the mid point of the regime.

At the middle of the regime, it was clear that the athlete could expend less energy to reach much higher heart rates, and establish new highs in his cardo-vascular fitness. By the close of the training regime, he could obtain much higher actual athletic output, traveling at much higher speeds while maintaining the same heart rate as he obtained at the start of the regime.

0.4 Applying Graph Analysis to Workout Studies

Graph visualization is used to model natural systems, in terms of forces needed to commute information, or other resources, between one site and another. Complex natural systems contain many nodes, or locii for energy, which is transferred to neighboring nodes.

In the case of the athlete we study, the time series of his data affords us opportunities to study each factor stored during his workouts. Thus, a multifactor time series expression of data is created through his sensor devices, which can map to graphs. Visualization of multi-node systems takes place when a weight, or force element shifts between nodes (in or out).[2] In a linear time series, the athlete simply shifts information forward, reaching new states of cardio vascular fitness.

Datum such as altitude as well as running form are captured, in several sequences. However, the interplay of the choice of workout, in terms of distance, as well as the resulting heart rate shift, is observable in the data. Hence, the choice of a graph to investigate changes in heart fitness opens up the application of the force directed graph to fitness studies.

The Directed Graph analysis enabled the researchers to chart a story of how the heart adapted to meet the needs of higher speeds. The graph supplies a set of weights, used to justify, or explain the arrival at subsequent hops in the graph. This approach to time series is helpful, in explaining why one state moves to the next. In this study, percentile tiers of heart rates were used as nodes in graphs, and the athlete's performance, or speed, was used as a weight. The athlete thus navigated increasing states of cardiovascular output, while performing with greater speed.[1]

0.5 Methodology

This study leveraged directed-force graph (DFG) analysis, applied to a long term set of time series, where athlete body statistics were available. Insights into how the heart adapted to greater athletic performance appear after the binning of workouts, and tiers of heart output were established within each workout.

DFG networks were then used to observe the heart's output as greater speeds were accomplished.

Rubrics:

1. heart rate and workout productivity
2. Force Directed Graphing
3. binning within subsampled time series

[2]:
```
!pip install networkx
```

```
Requirement already satisfied: networkx in /opt/conda/lib/python3.8/site-
packages (2.6.3)
```

0.6 Exploratory Studies

In a case where an athlete improved overall output per workout, the study posed these questions:

1. What factors accompanied improvement?
2. How does heart rate frame a story of improving athletic output?
3. How does the route chosen by the athlete factor into his/her improvement, in performance?

[8]:
```
import pandas as pd
import numpy as np
import datetime
import time
from scipy.stats import binned_statistic
import matplotlib.pyplot as plt
# from geopy import distance
import seaborn as sns
plt.rcParams['figure.figsize'] = [12, 4]
import networkx as nx
```

0.7 Data import

Strava data was transformed into epoch-based time series. Discreet workouts were dissected from the global data file using a means to separate rows, where a threshold 25 second delay in outputs separated workouts.

[9]:
```
sdf = pd.read_csv('assets/strava.csv')
sdf.rename(columns = {'timestamp':'t'}, inplace = True)
ts = sdf['t'].to_numpy()
arr = []
format = "%Y-%m-%d %H:%M:%S"
for i in ts:
    dto = datetime.datetime.strptime(i, format)
    tt = int(time.mktime(dto.timetuple()))
    arr.append(tt)
sdf['epoch'] = arr
sdf['epoch'] = sdf['epoch'].astype(int)
```

0.8 Prep Stage 1: Binning continuous intervals as distinct Workouts

Here, I establish the likely workouts as contiguous timestamp values. Where timestamp values contain near-contiguous values, a workout took place until the interval exceeds a maximum limit. Borders to bins are established, usable throughout the study. The edges for the bins are listed, as output.

[10]:
```
ll = sdf['epoch'].to_numpy()
be = []                                  #bin edges of discreet workouts in the␣
 ↪global datafile
for i in range(len(ll)):
    if (ll[i] - ll[i-1]) > 25:
        be.append(int(ll[i]))
sdf['ridebin'] = np.searchsorted(be, sdf['epoch'].values)
print(be)
```

```
[1562800680, 1562801097, 1562801502, 1562966831, 1563124828, 1563137107,
1563311440, 1563311809, 1563405426, 1563408718, 1563482874, 1563634600,
1563658117, 1563746305, 1563832037, 1563842947, 1564009723, 1564012585,
1564171804, 1564172828, 1564254690, 1564339726, 1564400841, 1564515107,
1564609817, 1564612035, 1564614223, 1564693930, 1564777238, 1564868976,
1564949016, 1564953279, 1565041511, 1565128180, 1565219966, 1565222537,
1565386120, 1565398029, 1565552804, 1565644973, 1565824756, 1566042636,
1566160897, 1566328707, 1566333177, 1566350154, 1566357856, 1566523479,
1566675377, 1567028488, 1567277143, 1568246980, 1568491720, 1568495688,
1568582062, 1568582120, 1568582308, 1568584910, 1568585412, 1568666839,
1569272151, 1569436057, 1569436576, 1569436964, 1569439045, 1569439205,
1569439611, 1569440326, 1569447725, 1569450034, 1569707472, 1569878734,
1569878834, 1569879075, 1570138203]
```

0.9 Prep Stage 2: Heart Rate Capacity Per Workout

Here, I chart the user's heart rate capacity. - Did a new max heart rate occur, over the 74~ workouts? - What distribution of heart rate quartiles was created, over the course of the workouts?

0.10 Filled Line Plots: How did Heart Rate Improve, during the Training Regimen?

What ranges of heart rates were accomplished? What is the range of heart rates?

Methodology

As a goal, the study outlines the progress the athlete made over time. Using bins in their ordered sequence, the high and low heart rate levels, from each workout is graphed as two series.

The effort is to understand, **did the athlete improve his/her performance, during the 75 workouts?**

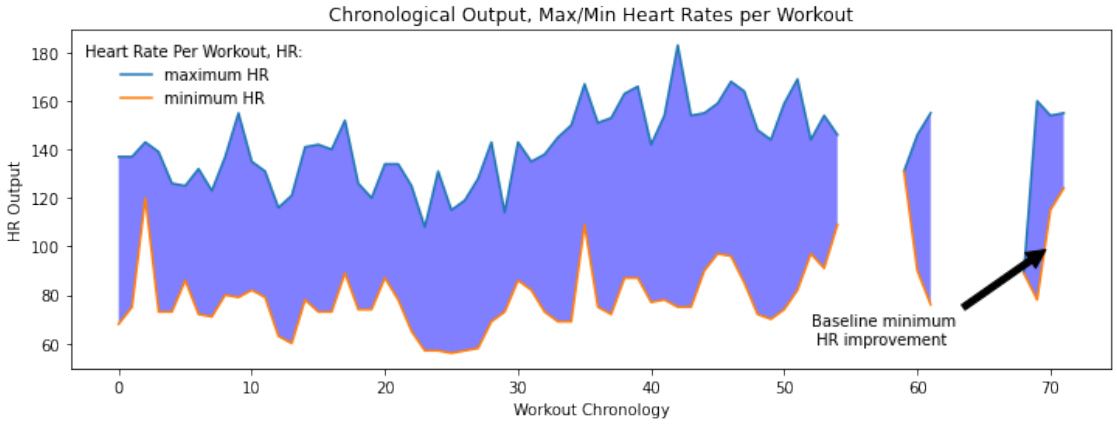
[11]:
```
hra = []                                 #heart rate array, of dicts: bin,␣
 ↪max, min
for i in range(74):                      #drop into each workout bin, dissect␣
 ↪heart rate
    x = sdf.loc[sdf['ridebin'] == i]
    mhr = x['heart_rate'].max()
    minhr =  x['heart_rate'].min()
    d = {"ridebin":i, "maxhr":mhr, "minhr":minhr}
```

```
    hra.append(d)
hrdf = pd.DataFrame(hra)
print("Historical MAX heart rate, {0}, then minimum heart rate: {1}".
 ↪format(hrdf.maxhr.max(), hrdf.minhr.min()))
```

```
Historical MAX heart rate, 183.0, then minimum heart rate: 56.0
```

[12]:
```
hrdf = pd.DataFrame(hra)
print("MAX heart rate, {0}, then minimum heart rate: {1}".format(hrdf.maxhr.
 ↪max(), hrdf.minhr.min()))
# print(hrdf.head(20))
#     ridebin  maxhr  minhr
# 0         0  137.0   68.0
def plotImprovementPath():
    x = hrdf['maxhr'].to_numpy()
    y = hrdf['minhr'].to_numpy()
    plt.plot(x, label="maximum HR")
    plt.plot(y, label="minimum HR")
    plt.legend(loc=0, frameon=False, title="Heart Rate Per Workout, HR:")
    plt.ylabel("HR Output")
    plt.xlabel("Workout Chronology")
    plt.title("Chronological Output, Max/Min Heart Rates per Workout")
    plt.gca().fill_between(hrdf.index,y,x,facecolor='blue', alpha=0.5)
    plt.annotate('Baseline minimum\n HR improvement', xy=(70, 100), xytext=(52,␣
 ↪60),
             arrowprops=dict(facecolor='black', shrink=0.05))
    plt.show()
    plt.close()
    return 0
hrdf.dropna()
plotImprovementPath()
```

```
MAX heart rate, 183.0, then minimum heart rate: 56.0
```



[12]: 0

As observed in the chronology above, the minimum heart rate improved gradually, while the maximum also improved. What is notable is how the maximum heart rate plateaued and did not return to its high, suggesting that **the athlete accomplished greater efficiency, during his workouts.**

This visualization helped to plot low and high heart rates, per workout. Clearly the athlete gradually delivered higher rates of cardio vascular fitness, then his heart rate located a lower high.

Given the limits of this visualization, the overall efficiency of the athlete would need more visualization, and justify the use of a graph to study how increased heart rate and workout performance collaborated.

In the following step I introduce the variable of distance, to enrich the study of heart rate, during workouts.

0.11 Multiline Plots: The Athlete's Accomplished Distance per Workout

Next, a study of the use of distance was done, to visualize how the athlete's choice of distance in his or her route impacted the rate of cardiovascular improvement. A chronology of distance accomplished is a means to express his or her improvement, during the experiment.

Some critical questions were asked:

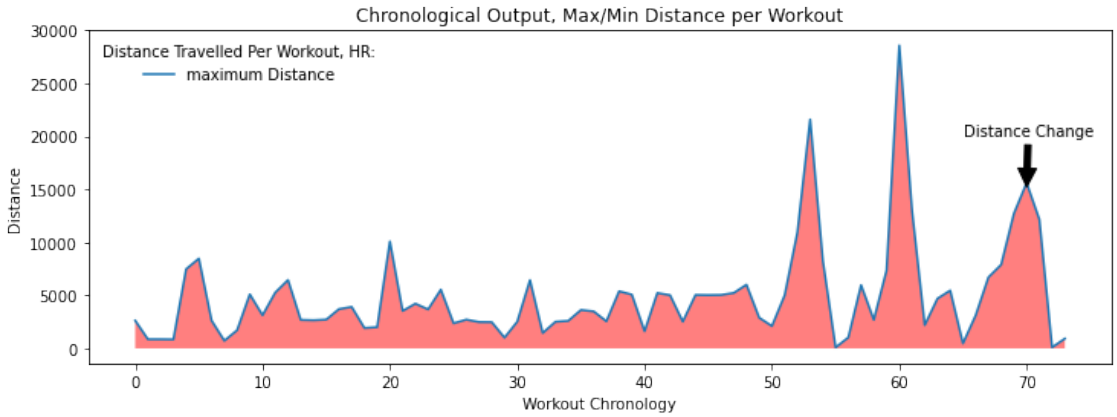
1. Did the distance travelled during the workout assist or contribute to a rise in heart rate?
2. Did the distance travelled accompany higher rates of cardio vascular output, and increased the rate of fitness?
3. What role does the chosen route, or the distance of the route play in improving cardio fitness?

[13]:
```
distdf = [] #heart rate array, of dicts: bin, max, min
for i in range(74):
    x = sdf.loc[sdf['ridebin'] == i]
    maxd = x['distance'].max()
    d = {"ridebin":i, "maxDistance":maxd}
    distdf.append(d)
ddf = pd.DataFrame(distdf)
print("MAX Distance throughout all workouts, {0}".format(ddf.maxDistance.max()))
```

```
MAX Distance throughout all workouts, 28577.43
```

[14]:
```
def distanceImprovementPath():
    x = ddf['maxDistance']
    plt.plot(x, label="maximum Distance")
    plt.legend(loc=0, frameon=False, title="Distance Travelled Per Workout, HR:
 ↪")
    plt.ylabel("Distance")
    plt.xlabel("Workout Chronology")
    plt.title("Chronological Output, Max/Min Distance per Workout")
```

```
    plt.gca().fill_between(hrdf.index,x,facecolor='red', alpha=0.5)
    plt.annotate('Distance Change', xy=(70, 15000), xytext=(65, 20000),
             arrowprops=dict(facecolor='black', shrink=0.05))
    plt.show()
    plt.close()
    return 0
ddf.dropna()
distanceImprovementPath()
```



[14]: 0

As visualized above, greater rates of athletic output occured over time, in the form of distance travelled during the workout.

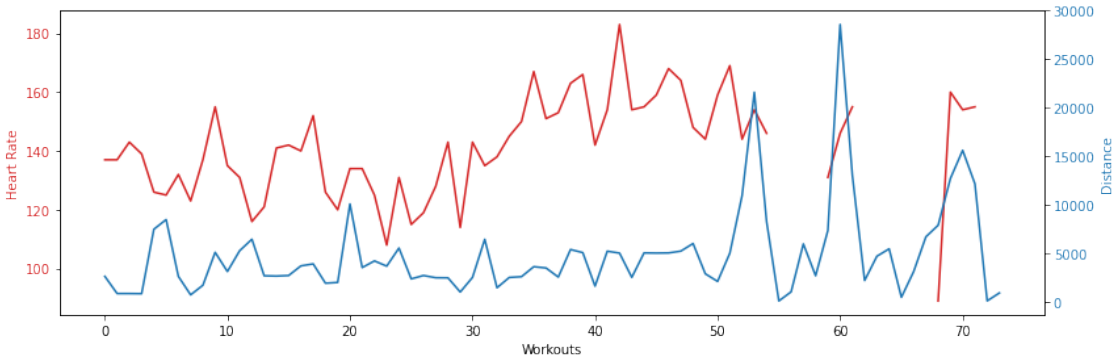
0.12 Twin Line Plots: Superimposing Heart Rate Output and Distance as a Composite Metric for Endurance

To study a potential association between rising heart rate capacity and distance, I charted heart rate and distance as a twin, dual axis chart.

Clearly, the athlete accomplished longer routes during an improved heart rate output. These two variables of workout duration, or distance, and heart rate bear a strong relationship.

[15]:
```
fig, ax1 = plt.subplots()
y = hrdf['maxhr']
x = hrdf['ridebin']
color = 'tab:red'
ax1.set_xlabel('Workouts')
ax1.set_ylabel('Heart Rate', color=color)
ax1.plot(x, y, color=color)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()
```

```
color = 'tab:blue'
yy = ddf['maxDistance']
xx = ddf['ridebin']
ax2.set_ylabel('Distance', color=color)
ax2.plot(xx,yy ,color=color)
ax2.tick_params(axis='y', labelcolor=color)
fig.tight_layout()
plt.show()
plt.close()
```



0.13 Studying Athlete Heart Rates, as a Distribution

In order to study the change in average heart rate per workout, I had to discover the distribution of heart rates. Since several sessions took place where very high heart rates occured, these outlier workouts had to be identified as important edge cases.

The occurence of high heart rate will be used, downstream, as a way to differentitate workouts, and apply distribution statistics across all binned workouts.

The chart below outlines the overall distribution of heart rates, in a non-normalized histogram, as a means to illustrate the probability distribution of heart output. I plan to use this distribution to help discover workouts that achieved these high productivity heart rates, as means to establish new fitness plateaus, or as expressions of the athlete's improving productivity.

Using a twin line plot, rates of cardiovascular fitness were imposed upon distance output. Clearly, the athlete accomplished more distance as higher heart rates were accomplished.

0.14 Calculating the caliber of each workout

how many high heart rate events transpired, per workout, which occupy the highest quartiles of the heart rate. I use a metric that counts the number of high heart rate events per workout, where

(number of HIIT heart rate events) / workout

thus: 1. within all heart rate events in the data set, locate all bin edges 2. rank all events with a quartile, or other bin value 3. iterate all binned workouts, count all values by binned weight, or

sum of binned values, where highest bin == most valued work out event 4. deliver a per workout metric, where high values equal better workouts, by total heart output

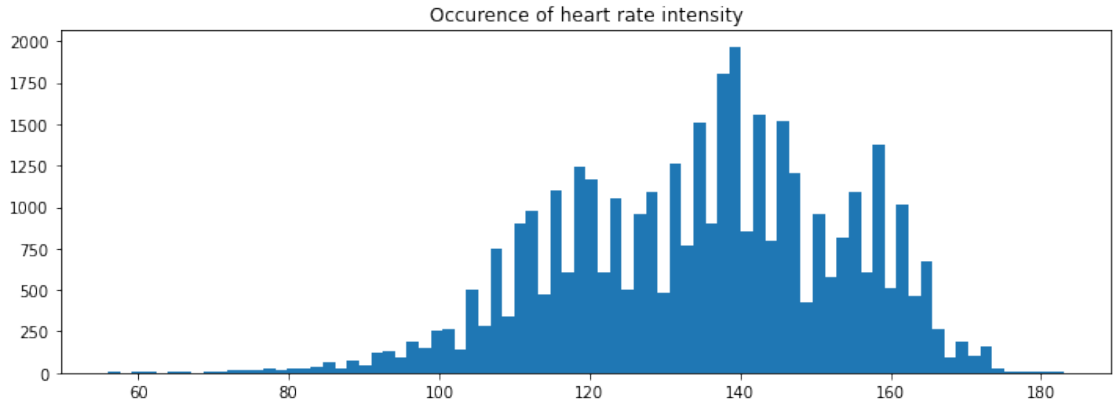
0.15 Leveraging Step Plots to Discover Facets of Highly Productive Workouts

Thus far, several computational artifacts have been established. 1. the chronology of heart rate levels, during the training narrative 2. heart rates, distributed, with outlier high and low levels 3. the occurence of the heart rate distribution over time, and in conjunction with distances travelled. 4. binned workouts, with discreet time series of heart rate and distance

We now possess some working pieces to help us study several perspectives on the workouts. 1. Did heart rate facilitate longer distances, or did the use of distance, by the athlete, factilitate creating higher heart productivity, ie a fitter heart? 2. Which specific workouts hold more insight into the role that distance plays, or that heart capacity plays, in building better routes?

[16]:
```
hre= sdf['heart_rate'].dropna().to_numpy()
bins = np.histogram_bin_edges(hre, bins='auto')
# print(bins, "( {0})".format(len(bins)))
```

[17]:
```
def exHRIntensity(hre):
    plt.hist(hre, bins='auto')
    plt.title("Occurence of heart rate intensity")
    plt.show()
    return 0
exHRIntensity(hre)
```



[17]:
```
0
```

[18]:
```
binL = []
for i in range(len(bins)-1):
    binL.append(i)
sdf['hrBin'] = pd.cut(sdf["heart_rate"], bins, labels = binL)
sdf['hrBin'] = pd.factorize(sdf['hrBin'])[0]
```

Step Chart: A Chronology of Workouts, By Maximum Heart Level

The twC dataframe thus captures 'total workout caliber,' with each binned workout, sequentially prepped for the next batch of studies.

Delivering a heart rate output metric table

for every workout, deliver a cumulative sum of all heart rate outputs, as a metric of each event's quality.

For the above chart, workouts are categorized by the maximum heart rates, each workout. This step chart is a chronology, not a distribution, and exhibits how maximum heart rates took place. These represent accomplishments by the athlete, to bring about elevated levels of fitness, and literally work harder.

The highlighted area represents the introduction of a new fitness device which captured stride and body animation data. This took place directly after these max heart rate events took place.

0.16 Scatterplots: How Distance Accompanied Changes in Heart Fitness

The series of binned workouts is thus delivered as a chronology. For every binned workout, there is a discreet time series of workout events.

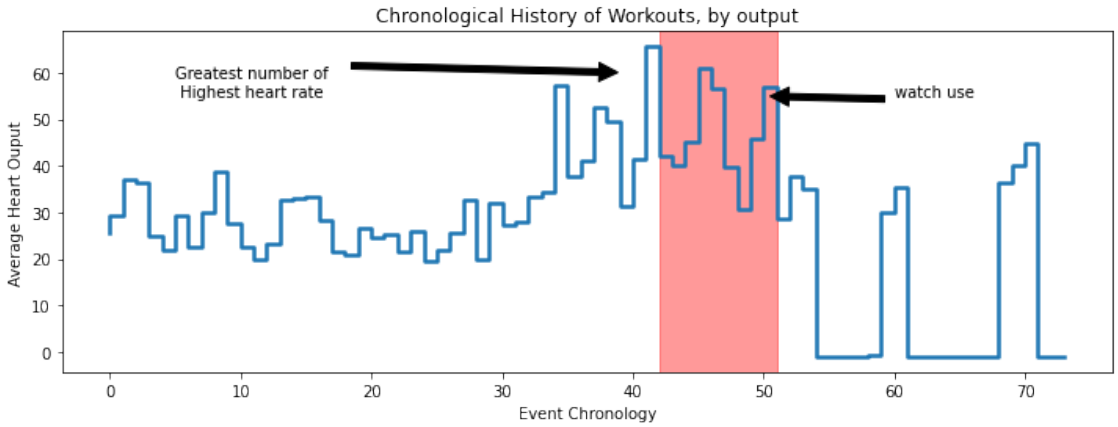
[19]:
```
sdf['hrBin'].dropna()
twc = []                                #total workout caliber, per workout as␣
 ↪expression of total heart rate output
for i in range(74):                     #drop into each workout
    x = sdf.loc[sdf['ridebin'] == i]  #get hrBin value and sum across the␣
 ↪entire workout
    s = x['hrBin'].mean()
    d = {"ridebin":i, "total workout quality":s}
    twc.append(d)
twC = pd.DataFrame(twc)
twC.head(5)
```

[19]:

	ridebin	total workout quality
0	0	25.595142
1	1	29.147059
2	2	37.129630
3	3	36.357143
4	4	25.075741

[20]:
```
def buildStep():
    fig, ax = plt.subplots()

    x = twC['ridebin'].to_numpy()
    y = twC["total workout quality"].to_numpy()
    ax.step(x, y, linewidth=2.5)
    plt.title("Chronological History of Workouts, by output")
    plt.xlabel("Event Chronology")
```

```
    plt.ylabel("Average Heart Ouput")
    plt.annotate('Greatest number of \n Highest heart rate', xy=(40, 60),␣
 ↪xytext=(5, 55),
                 arrowprops=dict(facecolor='black', shrink=0.05))
    plt.annotate('watch use', xy=(50, 55), xytext=(60, 55),
             arrowprops=dict(facecolor='black', shrink=0.05))
    plt.axvspan(42,51, color='red', alpha=0.4)
    plt.show()
    plt.close()
    return 0
buildStep()
```



[20]: 0

Each workout is ranked 1 through 74, in order in which they occur. The heart rates of each workout is also captured, ranking them in order.

For workouts that ranked highest for heart output, distance did not play a clear role. High heart rates could be accomplished with lower distance workouts. The left, veritical axis shows the workout, out of a total of 74 workouts. Clearly, the distance of workouts increased, as the maximum heart rate elevated.

The Impact of the Smartwatch, in Training

From the above chronology of workouts, by heartrate intensity, the shaded vertical highlights the workouts where a smart fitness watch was used. Clearly, high rates of productivity were exhibited during its use, then dropped off, after the watch was removed.

In the above scatter plot, the workouts with the highest heart output were ranked, by distance. In general, the highest ranked workouts, by heart output, also accompanied a longer distance travelled.

0.17 Analyzing the Top Workouts: Takeaways

Given the likely correlation between distances travelled and high heart rate, the data suggests that in order to accomplish a high heart rate and lift one's cardiovascular fitness, one should plan for a

longer distance route.

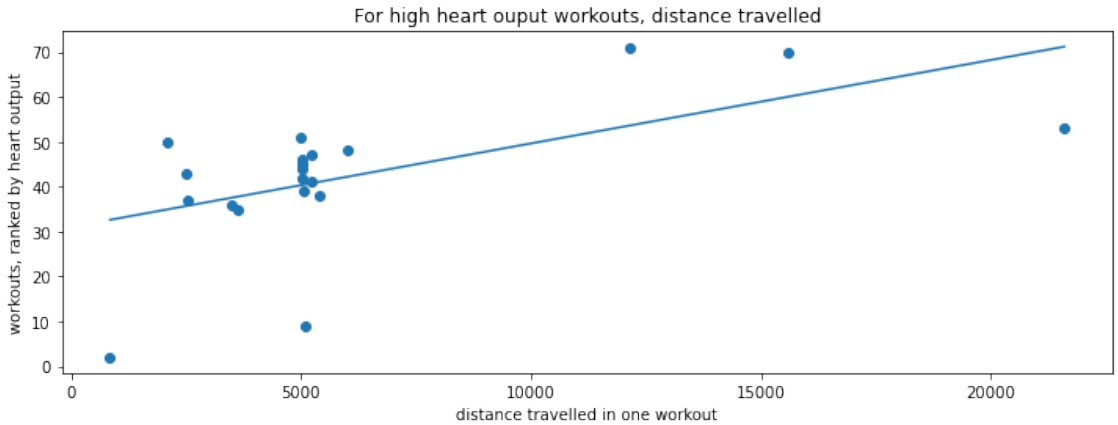
This notion is studied in the next few cells.

Distance Travelled is a Factor in Highly Productive Workouts

For the top scoring workouts, by most/high heart rate activities, are ranked by total distance. Does distance of your workout correlate to the heart rate output?

Generally, higher rates of distance does associate with higher rate of heart output.

[24]:
```
def genTopHeartRateDistance():
    lp = twC.sort_values(by="total workout quality", ascending=False).head(20)
    xa = lp['ridebin'].to_numpy()
    ll = []
    for i in range(len(xa)):     #drop into each, most-successful workout
        x = sdf.loc[sdf['ridebin'] == xa[i]]#.dropna()
        md = x['distance'].max()
        alt = x['altitude'].max() - x['altitude'].min()
        d = {"ridebin":xa[i], "distance":md}
        ll.append(d)
    dt = pd.DataFrame(ll)
    dtt = dt.sort_values(by="distance", ascending=False)
    plt.title("For high heart ouput workouts, distance travelled")
    z = np.polyfit(dtt['distance'], dtt['ridebin'], 1)
    p = np.poly1d(z)
    plt.plot(dtt['distance'], p(dtt['distance']))
    plt.scatter(dtt['distance'], dtt['ridebin'])
    plt.xlabel("distance travelled in one workout")
    plt.ylabel("workouts, ranked by heart output")
    plt.show();
    plt.close()
    return 0
sdf = sdf.drop(sdf.index[:3])
genTopHeartRateDistance()
```



[24]: 0

For Lower Productivity Workouts,Distance is not Determinative

In the lowest productivity workouts, the association with distance is not strong. In cases where overall workout productivity was the lowest, ie had the lowest heart rate maximums (and the least of them), the distance travelled was not constant. Data from the lowest output workouts shows that a person can experience moderate to low heart rate while exploring a multitude of workout distances.

Deficiencies in Linear, Two Factor Studies

Despite having the ability to combine bins of high heart rate and distance, our ability to correlate the causes of high heart rate workouts is marginal, at best. For every cluster of evidence supporting the role that distance plays, in boosting heart strength, there exist clusters where short distance accompanies strong heart output.

What our study needs is a sample of scenarios where heart rates were elevated, and how speed, or another variable representing athletic output, is present. I introduce the directed graph in the next phase of the study to help illustrate how a timed use of speed can elevate heart rate, and elevate the caliber of training.

However, it becomes visible that during the chronological training regimen, that **heart health, once plateaued, continues to decline despite increasing distances.**

[ ]:
```
def genLowHRDistance():
    lp2 = twC.sort_values(by="total workout quality", ascending=True).head(10)
    xa2 = lp2['ridebin'].to_numpy()
    ll2 = []
    for i in range(len(xa2)):      #drop into each, most-successful workout
        x = sdf.loc[sdf['ridebin'] == xa2[i]]#.dropna()
        md = x['distance'].max()
        d = {"ridebin":xa2[i], "distance":md}
        ll2.append(d)
    dt2 = pd.DataFrame(ll2)
    dt2 = dt2.sort_values(by="distance", ascending=False)
    plt.title("For low ouput workouts, distance travelled")

    z = np.polyfit(dt2['distance'], dt2['ridebin'], 1)
    p = np.poly1d(z)
    plt.plot(dt2['distance'], p(dt2['distance']))

    plt.scatter(dt2['distance'], dt2['ridebin'])
    plt.xlabel("distance travelled in one workout")
    plt.ylabel("workouts, ranked by heart output")
    plt.show();
    plt.close()
    return 0
```

```
sdf = sdf.drop(sdf.index[:3])
genLowHRDistance()
```

0.18 Workouts as Directed Graphs, and the Fitness Improvement Path

Directed graphs are typically used in causal inference studies. Though this study does not conclude causality, I introduce a data pipeline for doing so, in future studies.

I implemented a Force Directed Graph, using the NetworkX python visualization library, to pursue these questions: 1. how/why did fitness improve, overall, linear trend of workout productivity 2. what was the role of the watch? It appears that the watch was introduced after new heart rate plateaus were reached, not before 3. How did the combined strategies of altitude, distance, heart rate and duration work as a sequence, in creating more endurance, over time?

I inspected four of the top workouts, for the sequence of activities contributing toward fitness-building workouts. A sample of the top workouts was collected, where a workout in the middle of the training phase (42/75), at the start (2/75), then a the end (71/75).

[26]:
```
def prepGraph(x):
    x = sdf.loc[sdf['ridebin'] == x]
    q70 = x['heart_rate'].quantile(q=0.70)
    q75 = x['heart_rate'].quantile(q=0.75)
    q80 = x['heart_rate'].quantile(q=0.80)
    q85 = x['heart_rate'].quantile(q=0.85)
    q90 = x['heart_rate'].quantile(q=0.90)
    q95 = x['heart_rate'].quantile(q=0.95)
    q99= x['heart_rate'].quantile(q=0.99)
    ll = x['heart_rate'].to_numpy()
    be = []
    p70 = []
    p75 = []
    p80 = []
    p85 = []
    p90 = []
    p95 = []
    for i in range(len(ll)):                    #     print(ll[i]) #correct hr
        if ll[i] >= q70 and ll[i] <= q75:
            p70.append(ll[i])
        if ll[i] >= q75 and ll[i] <= q80:
            p75.append(ll[i])
        if ll[i] >= q80 and ll[i] <= q85:
            p80.append(ll[i])
        if ll[i] >= q85 and ll[i] <= q90:
            p85.append(ll[i])
        if ll[i] >= q90 and ll[i] <= q95:
            p90.append(ll[i])
        if ll[i] >= q95 and ll[i] <= q99:
            p95.append(ll[i])
```

```
   # print("70: {0}\n 75: {1}\n 80: {2}\n 85:{3}\n 90:{4}\n 95:{5}".
↪format(max(p70), max(p75),
   #     max(p80), max(p85), max(p90), max(p95)))
   df70 = x.loc[x['heart_rate'] == max(p70)] #df with percentile values, max␣
↪below
   df75 = x.loc[x['heart_rate'] == max(p75)]
   df80 = x.loc[x['heart_rate'] == max(p80)]
   df85 = x.loc[x['heart_rate'] == max(p85)]
   df90 = x.loc[x['heart_rate'] == max(p90)]
   df95 = x.loc[x['heart_rate'] == max(p95)]
   graphItems = []
   msp70 = df70['enhanced_speed'].max()
   dis70 = df70['distance'].max()
   se70 = df70['heart_rate'].idxmin()
   hr70 = df70['heart_rate'].max()
   graphItems.append({"pct":70 , "maxspeed":msp70 , "maxdistance":dis70 ,␣
↪"seq":se70, "heart":hr70})
   msp75 = df75['enhanced_speed'].max()
   dis75 = df75['distance'].max()
   se75 = df75['heart_rate'].idxmin()
   hr75 = df75['heart_rate'].max()
   graphItems.append({"pct":75 , "maxspeed":msp75 , "maxdistance":dis75, "seq":
↪se75, "heart":hr75})
   msp80 = df80['enhanced_speed'].max()
   dis80 = df80['distance'].max()
   se80 = df80['heart_rate'].idxmin()
   hr80 = df80['heart_rate'].max()
   graphItems.append({"pct":80 , "maxspeed":msp80 , "maxdistance":dis80, "seq":
↪se80, "heart":hr80})
   msp85 = df85['enhanced_speed'].max()
   dis85 = df85['distance'].max()
   se85 = df85['heart_rate'].idxmin()
   hr85 = df85['heart_rate'].max()
   graphItems.append({"pct":85 , "maxspeed":msp85 , "maxdistance":dis85,"seq":
↪se85, "heart":hr85})
   msp90 = df90['enhanced_speed'].max()
   dis90 = df90['distance'].max()
   se90 = df90['heart_rate'].idxmin()
   hr90 = df90['heart_rate'].max()
   graphItems.append({"pct":90 , "maxspeed":msp90 , "maxdistance":dis90,"seq":
↪se90, "heart":hr90})
   msp95 = df95['enhanced_speed'].max()
   dis95 = df95['distance'].max()
   se95 = df95['heart_rate'].idxmin()
   hr95 = df95['heart_rate'].max()
```

```
    graphItems.append({"pct":95 , "maxspeed":msp95 , "maxdistance":dis95,"seq":
↪se95, "heart":hr95})
    df42 = pd.DataFrame(graphItems)
    df42['weight'] = round(np.divide((df42['maxspeed'] * df42['maxdistance']),␣
↪1000.0), 4)
    # df42.head(20)
    # df95.head(5)
    # print(graphItems)
    triggerGraph(df42)
    return 0
```

0.19 Building node and edge datum for graphs

Graphs depend on three basic data structures: 1. nodes, which correspond to scatter or line plots, on a 2D graph 2. edges, which represent lines, connecting nodes 3. weights, which detail how force connects nodes, via edges.

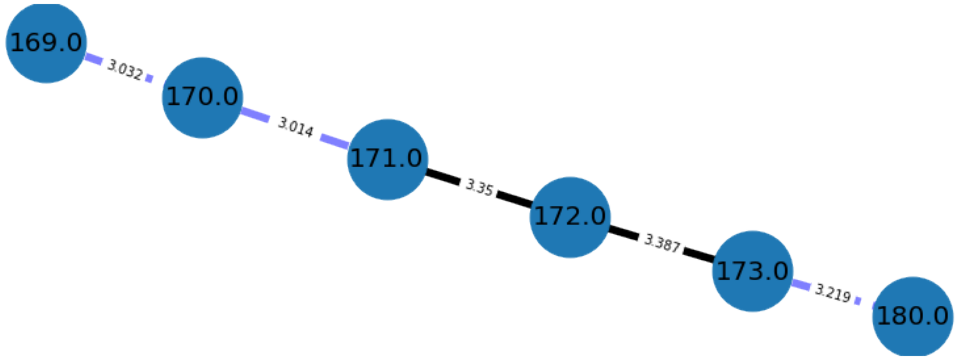
In a force directed graph, an input/output set of edges is normally plotted, for each node. Given the freedom of various graph systems to express multiple edges, the FDG attempts to model stepped systems, similar to teh linear structure of events within one workout.

Hence, the relevance of distance, speed on heart rate is visualized. Does the athlete contol his/her heart rate with the choice of speed, or does speed deliver heart rate changes?

[27]:
```
def triggerGraph(df):
    G = nx.Graph()
    w = df['maxspeed'].to_numpy()
    seq = df['heart'].to_numpy()
    for j in range(len(seq)):
        if j < len(seq)-1:
    #         print("heart: ", j, seq[j],  w[j])
            G.add_edge(seq[j], seq[j+1], weight = w[j]  )
    G.add_edge(seq[4], seq[5], weight=w[5])

    elarge = [(u, v) for (u, v, d) in G.edges(data=True) if d["weight"] > 3.3]
    esmall = [(u, v) for (u, v, d) in G.edges(data=True) if d["weight"] <= 3.3]

    pos = nx.spring_layout(G, seed=7)  # spring_layout
    # nodes
    nx.draw_networkx_nodes(G, pos, node_size=3700)
    # edges
    nx.draw_networkx_edges(G, pos, edgelist=elarge, width=6)
    nx.draw_networkx_edges(
        G, pos, edgelist=esmall, width=6, alpha=0.5, edge_color="b",␣
↪style="dashed")
    # node labels
    nx.draw_networkx_labels(G, pos, font_size=20, font_family="sans-serif")
    # edge weight labels
```

```
    edge_labels = nx.get_edge_attributes(G, "weight")
    nx.draw_networkx_edge_labels(G, pos, edge_labels)

    ax = plt.gca()
    ax.margins(0.08)
    plt.axis("off")
    plt.tight_layout()
    plt.show()
    print(list(G.edges))
    df.head(20)
    return 0
```

[28]:
```
prepGraph(42)
```




```
[(169.0, 170.0), (170.0, 171.0), (171.0, 172.0), (172.0, 173.0), (173.0, 180.0)]
```

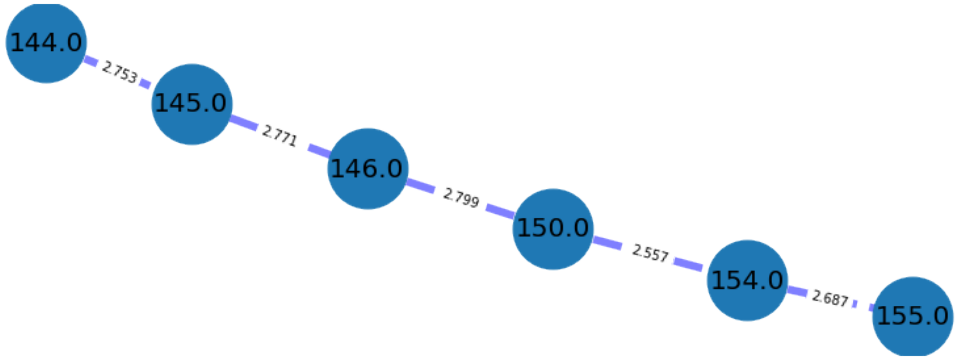
[28]:
```
0
```

From the graph of a mid training phase (42nd out of 75), it appears that it requires roughly 10% more exertion to move into the 80th percentile of heart output. However, moving into the 99th percentile from the mid 80s and 90's, requires less speed. Technically, moving the training session beyond the 75th percentile, of total heart output, creates a much easier output needed to move the heart into the 99th percentile.

However, the dramatic increase in heart rate between 90th and 95th percentiles creates a motivating transition, in high-intensity heart output. It requires only 3% more energy output to move the heart between 173 and 180 bpm. Thus, this is a much cheaper energy output than moving the heart between the 75th and 80th.

This efficient movement between high-output heart states is an attribute of mid-phase training, where improving between high, and highest tier productivity is easier.

[29]:
```
prepGraph(9)
```



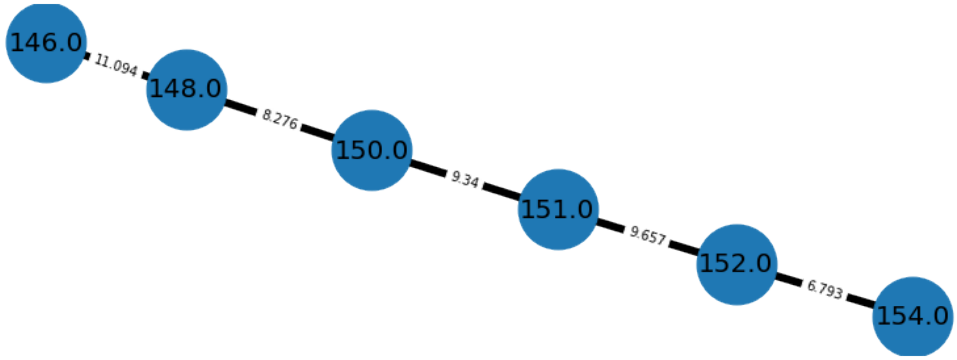

```
[(144.0, 145.0), (145.0, 146.0), (146.0, 150.0), (150.0, 154.0), (154.0, 155.0)]
```

[29]: 0

At the start of the training phase, it is clear that high heart output is harder to come by. Our data showed that only on one occasion did the heart rate reach a record point. This sample from the first few workouts shows how low the maximal heart rates are, compared to the middle of the training regimen.

Also, moving between the differing tiers of performance required a similar distribution of speed. The difference in heart rate among 70, through the 99th percentile represented less than a 10% increase in heart rate. This stands in difference from the range that was accomplished at the middle of the regimen.

[30]:
```
prepGraph(71)
```




```
[(146.0, 148.0), (148.0, 150.0), (150.0, 151.0), (151.0, 152.0), (152.0, 154.0)]
```

[30]: 0

The above sample was taken from a late-stage workout (71/75). This represented a highpoint in the workout regimen, from the standpoint of heart rate. Note how moving between states required

speeds which were several times higher than at the start of the regimen. It is clear that the heart rate remains much lower, despite reaching regional highs. This is taking place while travelling at much higher speeds, at some junctions 3 to 4 times higher than at the start. The heart fitness was much higher. It is likely that the body reached a point of much higher efficiency, where extreme heart rates were unnecessary, in order to deliver much higher speeds.

0.20 Conclusion

Given the practicality of the FDG to visualize the chronology of distance and speed as precursors to heart rate, we can contrast the high heart rate of early workouts to speed, against the relative low response by the heart to much higher speed, and higher distances, in later training.

We have two major observations to make: 1. early in the workout chronology, maximum heart rates are possible, likely due to the onset of the new regime. 2. in the middle of the training regimen, maximum heart rates are possible, but likely due to the controlled use of speed and distance by the athlete, to introduce higher levels of performance. 3. in late stage workouts, heart rates do not resume the maximums, present at the outset and middle. Average heart rates have improved, but do not reach the same maxima, despite much higher speeds and distances.

In short, the FDG helps draw comparisons between the binned workouts, where time series datum on heart rate, distance and speed are present. We can visualize how early, mid and late stage workouts proceed, with the potential for debate on the causal events, leading up to speed, and higher heart rate.

0.21 Notes for Future Studies

Take SIADS 630! Just Kidding

Use the Directed Graph structure as a norm for studying time series events, and introducing a basis for studying causal events, in multi-factor time series phenomena.

0.22 References

1. Networkx documentation on Fruchterman-Reingold force-directed algorithms, and path-graphs in python. https://networkx.org/documentation/stable/reference/generated/networkx.drawing.layou
2. Fruchterman, T.M., & Reingold, E.M. (1991). Graph drawing by force-directed placement. Software: Practice and Experience, 21. https://doi.org/10.1002/spe.4380211102