PRINCIPLES OF COMPLEX SYSTEMS

HW01 WRITE-UP

September 8, 2018

David W. Landay
University of Vermont
Graduate Student, Comp. Systems and Data Sci.

0.1 Problem 1

Use a back-of-an-envelope scaling argument to show that maximal rowing speed V increases as the number of oarspeople N as $V \propto \mathrm{N}^{\frac{1}{9}}$ Assume the following:

$$\text{Rowing Shells are geometrically similar (isometric) } l --> w \propto l \tag{1}$$

$$\text{The resistance encountered by a shell is due largely to drag on its wetted surface} \tag{2}$$

$$D_f \propto V^2 \times l^2 \tag{3}$$

$$P \propto D_f \times V \tag{4}$$

$$V_d \propto N \tag{5}$$

$$\text{The depth of water } d \text{ scales isometrically with boat length } l \tag{6}$$

$$P \propto N \tag{7}$$

Solution:

$$\begin{aligned}
P &\propto (V^2 \times l^2) \times V && \text{(3,4)} \\
\implies P &\propto V^3 \times l^2 && \text{(assoc.)} \\
\implies N &\propto V^3 \times l^2 && \text{(7)} \\
\implies V_d &\propto V^3 \times l^2 && \text{(5)} \\
\implies l^2 \times d &\propto V^3 \times l^2 && \text{(def.)} \\
\implies l^3 &\propto V^3 \times l^2 && \text{(6)} \\
\implies V^3 &\propto l && \text{(assoc.)} \\
\implies N &\propto V^3 \times (V^3)^2 && \text{(trans. line 3, line 6)} \\
\implies N &\propto V^9 && \text{(assoc.)} \\
\implies V &\propto N^{\frac{1}{9}} \therefore
\end{aligned}$$

0.2 Problem 2

Find the modern day world record times for 2000 metre races and see if this scaling still holds up. Of course, our relationship is approximate as we have neglected numerous factors, the range is extremely small (1–8 oarspeople), and the scaling is very weak (1/9). But see what you can find.

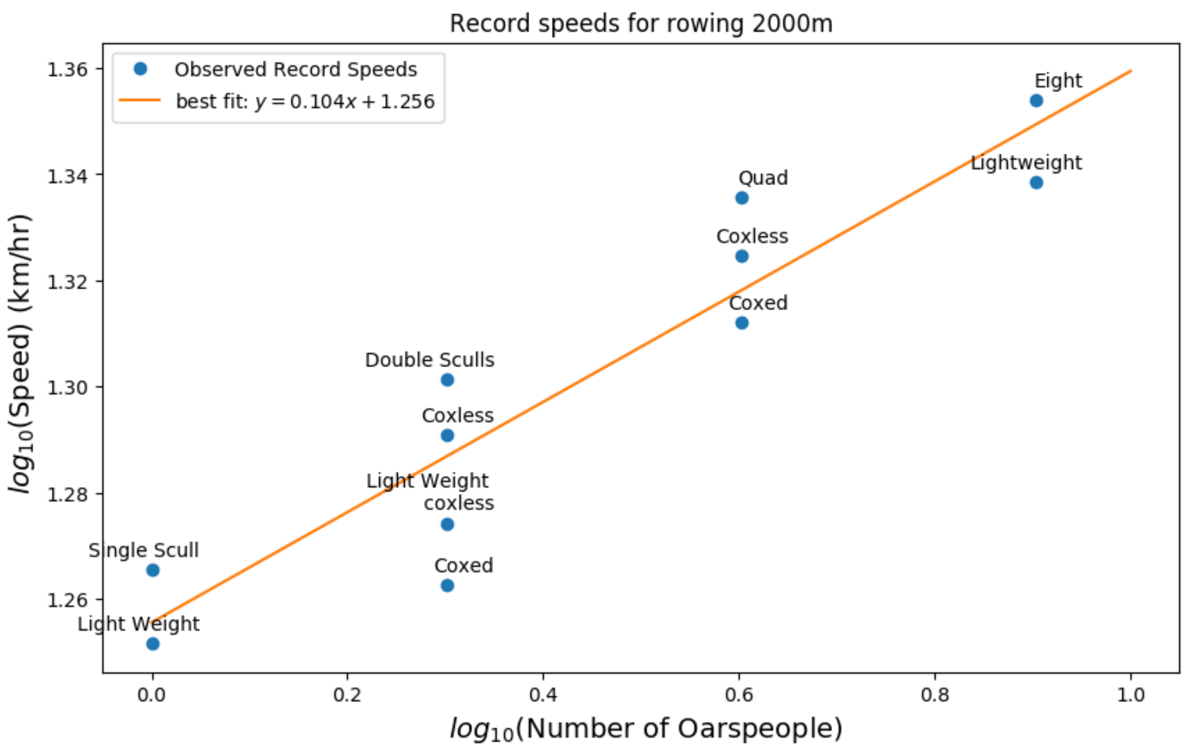

Figure 1: Speed of Boat vs Number of Oarspeople, 2000m world records

The slope determined from the linear regression (with an R^2 value = 0.86 and p-value = 3.17×10^-5) over the data suggests that a $\frac{1}{10}$ scaling more closely defines the relationship, but it is close and may not be represented by the amount of data.

0.3 Problem 3

Check current weight lifting records for the snatch, clean and jerk, and the total for scaling with body mass (three regressions).

(a) Does $\frac{2}{3}$ scaling hold up?

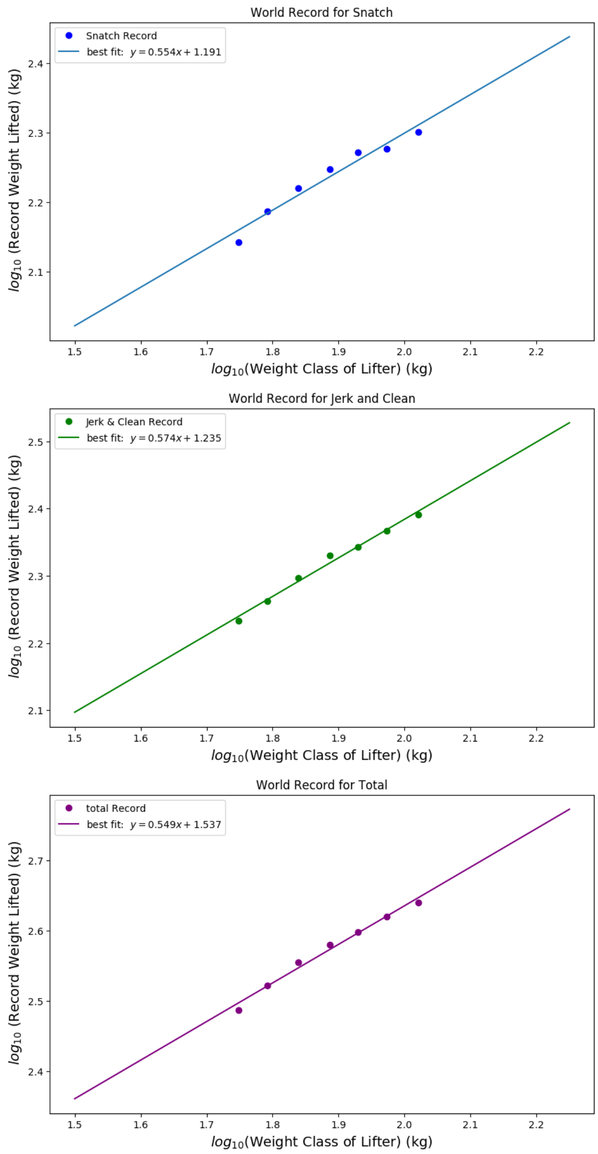

Figure 2: Weight lifted

$\frac{2}{3}$ scaling does not hold up for any category based on a least-squares linear regression. We find that the slopes of the log-log plots are not $\frac{2}{3}$. (b) Normalized by the appropriate scaling, who holds the overall, re-scaled world record?

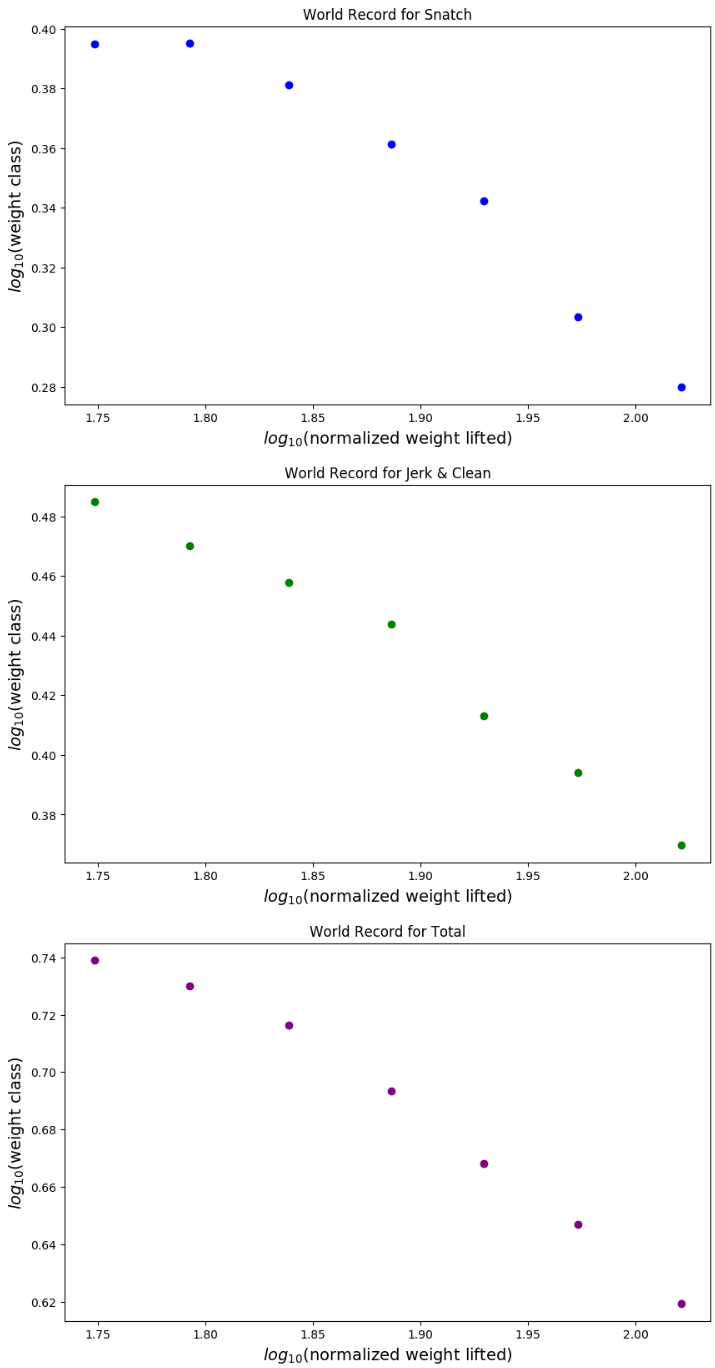

Figure 3: Weight Lifted vs. Weight of Lifter noramlized by weight lifted by category to weight class

If we normalize weight classes by the ratio of total weight lifted to the upper limit of each competitive weight class, and re-plot the values. We discover that lighter weight lifters are actually lifting more weight compared in comparison to their own body weight. This indicates that Wu Jiangbiao is the record holder in the snatch, Om Yun-chol is the record holder for the clean and Jerk, and Long Qingquan is the record holder for the total.

0.4 Problem 4

Finish the calculation for the platypus on a pendulum problem so show that a simple pendulum's period τ is indeed proportional to $\sqrt{\frac{l}{g}}$. Basic plan from lectures: Create a matrix A where ij_{th} entry is the power of dimension iin the j_{th} variable, and solve by row reduction to find basis null vectors. In lectures, we arrived at:

$$A\vec{x} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

You only have to take a few steps from here.

Solution:

We know that the number of related quantities $n = 4$ and the rank r of our matrix A is 3 as there are 3 dimensions length L, mass M, and time T that are represented in the model. Hence, there are $n - r = 1$ dimensionless quantities. The relationship between the dimensionless quantity π_1 and 4 related quantities $\{q_1, q_2, q_3, q_4\} = \{l,m,g,\tau\}$ can be expressed as follows:

$$[\pi_i] = [l]^{x_1}[m]^{x_2}[g]^{x_3}[\tau]^{x_4} = L^{x_1}M^{x_2}(LT^{-2})^{x_3}T^{x_4} \quad (1)$$

Combining like terms, we get:

$$[\pi_i] = L^{x_1+x_3}M^{x_2}T^{-2x_3+x_4} \quad (2)$$

which yields the matrix above. The matrix A yields a system of linear equations:

$$x_1 + x_3 = 0 \tag{3}$$

$$x_2 = 0 \tag{4}$$

$$-2x_3 + x_4 = 0 \tag{5}$$

Let:

$$\begin{aligned} x_1 &= 1 \\ x_2 &= 0 \\ x_3 &= -1 \\ x_4 &= -2 \end{aligned}$$

then,

$$\begin{aligned} \implies \pi_1 &= l m^0 g^{-1} \tau^{-2} \\ \implies \pi_1 &= l g^{-1} \tau^{-}2 \\ \implies \pi_1 \tau^2 &= l g^{-1} \\ \implies \tau^2 &\propto l g^{-1} \\ \implies \tau &\propto \sqrt{\frac{l}{g}} \therefore \end{aligned}$$

0.5 Problem 5

Show that the maximum speed of animals V_{max} is proportional to their length L. Here are five dimensionful parameters:

- V_{max}, maximum speed
- l, animal length
- ρ, organismal density
- σ, maximum applied force per unit area of tissue
- b, maximum metabolic rate per unit mass (b has the dimensions of power per unit mass)

And here are the three dimensions: L, M, and T
Use a back-of-the-envelope calculation to express V_{max}/l in terms of ρ, σ, and b.

Assumptions:

$$\rho \propto [M][L]^{-3} \tag{1}$$

$$\sigma \propto [M][L][T]^{-2} \propto [L]^2 \tag{2}$$

$$\sigma \propto [M][L]^{-1}[T]^{-2} \tag{3}$$

$$b \propto [L]^2[T]^{-3} \tag{4}$$

$$V_{max} \propto [L][T]^{-1} \tag{5}$$

Solution:

$$
\begin{aligned}
\sigma[T]^2 &\propto [M][L]^{-1}\sigma^{-1} && \text{(assoc. 2)}\\
\implies [T] &\propto \sqrt{[M][L]^{-1}\sigma^{-1}} && \text{(line 1)}\\
\rho[L]^3 &\propto [M] && \text{(assoc. 1)}\\
\implies [T] &\propto \sqrt{\rho[L]^3[L]^{-1}\sigma^{-1}} && \text{(line 2, line 3)}\\
\implies [T] &\propto \sqrt{\rho[L]^2\sigma^{-1}} &&\\
b[T]^3 &\propto [L]^2 && \text{(assoc. 4)}\\
\implies [T] &\propto \sqrt{\rho b[T]^3\sigma^{-1}} && \text{(line 5, line 6)}\\
\implies [T]^2 &\propto \rho b[T]^3\sigma^{-1} &&\\
\implies [T]^2[T]^{-3} &\propto \rho b\sigma^{-1} &&\\
\implies [T]^{-1} &\propto \rho b\sigma^{-1} &&\\
\implies V_{max}/l &\propto \rho b\sigma^{-1} \therefore &&
\end{aligned}
$$

0.6 Problem 6

Use the Buckingham π theorem to reproduce G. I. Taylor's finding the energy of an atom bomb E is related to the density of air ρ and the radius of the blast wave R at time t:

$$E = constant \times \rho \frac{R^5}{t} \tag{1}$$

In constructing the matrix, order parameters as E, ρ, R, and t and dimensions as L, T, and M.

Assumptions:

$$[E] = ML^2T^{-2} \tag{1}$$
$$[\rho] = ML^{-3} \tag{2}$$
$$[R] = L \tag{3}$$
$$[t] = T \tag{4}$$

Solution:

$$[\pi_i] = [E]^{x_1}[\rho]^{x_2}[R]^{x_3}[t]^{x_4}$$
$$[\pi_i] = (ML^2T^{-2})^{x_1}(ML^{-3})^{x_2}L^{x_3}T^{x_4}$$
$$\implies [\pi_i] = M^{x_1+x_2}L^{2x_1-3x_2+x_3}T^{-2x_1+x_4}$$

then,

$$A\vec{x} = \begin{bmatrix} 2 & -3 & 1 & 0 \\ -2 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

so,

$$x_1 + x_2 = 0$$
$$2x_1 - 3x_2 + x_3 = 0$$
$$-2x_1 + x_4 = 0$$

$$\implies$$

$$x_1 = -x_2$$

$$x_3 = -5x_1$$

$$x_4 = 2x_1$$

Let $x_1 = 1$. Then,

$$[\pi_i] = [E][\rho]^{-1}[R]^{-5}[t]^2$$

$$\implies [\pi_i][\rho][R]^5[t]^{-2} = [E]$$

$$\implies E = constant \times \rho R^5 t^{-2} \therefore$$

0.7 Problem 7

Use the Buckingham π theorem to derive Kepler's third law, which states that the square of the orbital period of a planet is proportional to the cube of its semi-major axis. Let's shed some enlightenment and assume circular orbits.

Parameters:

- Planet's mass, m;
- Sun's mass, $M_\odot$;
- Orbital period, τ;
- Orbital radius, r;
- Gravitational constant, G.

(a) What are the dimensions of these five quantities?

$$\begin{aligned}
[m] &= M \\
[M_\odot] &= M \\
[\tau] &= T \\
[r] &= L \\
[G] &= L^3 M^{-1} T^{-2}
\end{aligned}$$

(b) You will find that there are two dimensionless parameters using the Buckingham π theorem, and that you can choose one to be $\pi_2 = m/M$. Find the other dimensionless parameter, π_1.

$$\begin{aligned}
[\pi_i] &= [m]^{x_1}[M_\odot]^{x_2}[\tau]^{x_3}[r]^{x_4}[G]^{x_5} \\
\implies [\pi_i] &= M^{x_1+x_2} T^{x_3} L^{x_4} (L^3 M^{-1} T^{-2})^{x_5} \\
&= M^{x_1+x_2-x_5} T^{x_3-2x_5} L^{x_4+3x_5}
\end{aligned}$$

Then,

$$A\vec{x} = \begin{bmatrix} 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -2 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

which implies a nullity of 2, since number of related quantities is $n = 5$ and the rank $r = 3$. So,

$$\begin{aligned} x_1 + x_2 - x_5 &= 0 \\ x_3 - 2x_5 &= 0 \\ x_4 + 3x_5 &= 0 \end{aligned}$$

and,

$$\begin{aligned} x_1 + x_2 &= x_5 \\ x_3 &= 2x_5 \\ x_4 &= -3x_5 \end{aligned}$$

Let $x_5 = 1$. Then,

$$\begin{aligned} x_1 = x_2 &= \frac{1}{2} \\ x_3 &= 2 \\ x_4 &= -3 \end{aligned}$$

$$\begin{aligned} \implies [\pi_1] &= [m]^{\frac{1}{2}}[M_\odot]^{\frac{1}{2}}[\tau]^2[r]^{-3}[G] \\ \implies \pi_1 &= \sqrt{mM_\odot}\frac{\tau^2}{r^3}G \therefore \end{aligned}$$

(c) Now argue that $\tau^2 \propto r^3$:

Notice that G, m, and $M_\odot$ are constants. So,

$$
\begin{aligned}
[\pi_1] &= [\tau]^2[r]^{-3} \\
\implies \pi_1 &= \tau^2 r^{-3} \\
\implies r^3 &\propto \tau^2 \therefore
\end{aligned}
$$

(d) For our solar system's nine (9) planets (yes, Pluto is on the team here), plot τ^2 vs. r^3, and using basic linear regression report on how well Kepler's third law holds up.

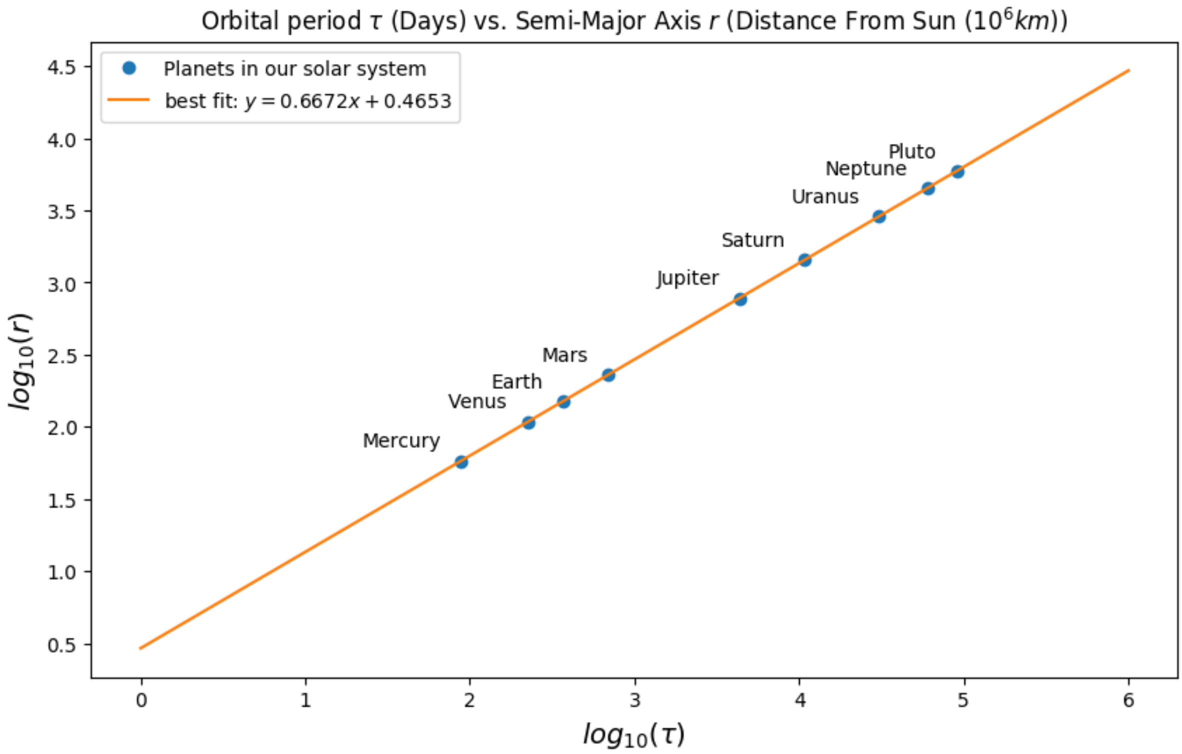

Figure 4: Orbital perdiod (days) vs. Semi-major axis ($10^6 km$)

Based on the fit of the regression, Kepler's third law holds up extremely well, indicating that the orbital period squared is proportional to the semi-major axis of an orbiting body cubed. The R^2 value for this regression was 0.99, indicating that the relationship between orbital period and semi-major axis is well explained.

0.8 Problem 8

Surface area of allometrically growing Minecraft organisms:
Let's consider animals as parallelepipeds (e.g., the well known box cow), with dimensions L_1, L_2, and L_3 and volume $V = L_1 \times L_2 \times L_3$. As we vary in scale of organism, let's assume the lengths scale with volume as $L_i = c_i^{-1} V^{\gamma_i}$ where the exponents satisfy $\gamma_1 + \gamma_2 + \gamma_3 = 1$ and the c_i are prefactors such that $c_1 \ timesc_2 \times c_3 = 1$. Let's also arrange our organisms so that $\gamma_1 \leq \gamma_2 \leq \gamma_3$.

(a) Show that the scalings $L_i = c_i^{-1} V^{\gamma_i}$ mean that indeed $L_1 \times L_2 \times L_3 = V$.

$$\begin{aligned}
L_1 &= c_1^{-1} V^{\gamma_1} \\
L_2 &= c_2^{-1} V^{\gamma_2} \\
L_3 &= c_3^{-1} V^{\gamma_3} \\
\implies L_1 \times L_2 \times L_3 &= c_1^{-1} V^{\gamma_1} \times c_2^{-1} V^{\gamma_2} \times c_3^{-1} V^{\gamma_3} \\
&= \frac{V^{\gamma_1+\gamma_2+\gamma_3}}{c_1 \times c_2 \times c_3} \\
&= \frac{V^1}{1} \\
\implies L_1 \times L_2 \times L_3 &= V \therefore
\end{aligned}$$

(b) Write down the γ_i corresponding to isometric scaling.

Assumptions:

$$\gamma_1 + \gamma_2 + \gamma_3 = 1 \tag{1}$$

$$V \text{ scales isometrically} \tag{2}$$

Solution:

$$\begin{aligned}
L_1, L_2, \text{and } L_3 \text{ scale isometrically} && (2) \\
\implies L_i = c_i^{-1} V^{\gamma_i} \text{ scale isometrically for all } i && \\
\implies \gamma_1 = \gamma_2 = \gamma_3 = \frac{1}{3} \therefore && (1, \text{line } 2)
\end{aligned}$$

(c) Calculate the surface area S of our imaginary beings.

Assumptions:

$$V \text{represents a cuboid} \tag{1}$$

Solution:

$$\begin{aligned} S &= 2(L_1L_2) + 2(L_1L_3) + 2(L_2L_3) && \text{(def. 1)} \\ &= 2(L_1L_2 + L_1L_3 + L_2L_3) \\ &= 2(L_1 \times L_2 \times L_3)(\frac{1}{L_1} + \frac{1}{L_2} + \frac{1}{L_3}) \\ &= 2V(\frac{1}{L_1} + \frac{1}{L_2} + \frac{1}{L_3}) \therefore \end{aligned}$$

(d) Show how S behaves as V becomes large (i.e., which term(s) dominate).

Expanding upon the previous part:

$$S = 2V(\frac{c_3}{V^{\gamma_3}} + \frac{c_2}{V^{\gamma_2}} + \frac{c_1}{V^{\gamma_1}})$$
$$S = 2(V^{1-\gamma_3} + V^{1-\gamma_2} + V^{1-\gamma_1})\text{And as } V \to \infty,$$
$$S = V^{1-\gamma_3} + V^{1-\gamma_2} + V^{1-\gamma_1}$$

The γ_i terms will determine how S scales. But, from the relationship established above:

$$\gamma_1 + \gamma_2 + \gamma_3 = 1$$
$$\text{and}$$
$$\gamma_1 \leq \gamma_2 \leq \gamma_3$$

As V grows and γ_3 approaches 1, then surface area will scale fastest, because γ_1 and γ_2 will need to shrink in order to remain within the constraints of $\gamma_1 + \gamma_2 + \gamma_3 = 1$. Hence, as γ_3 approaches 0, V will scale isometrically and therefore slower. (see e.)

(e) Which sets of γ_i give the fastest and slowest possible scaling of S as a function of V?

See answer above and plots below:

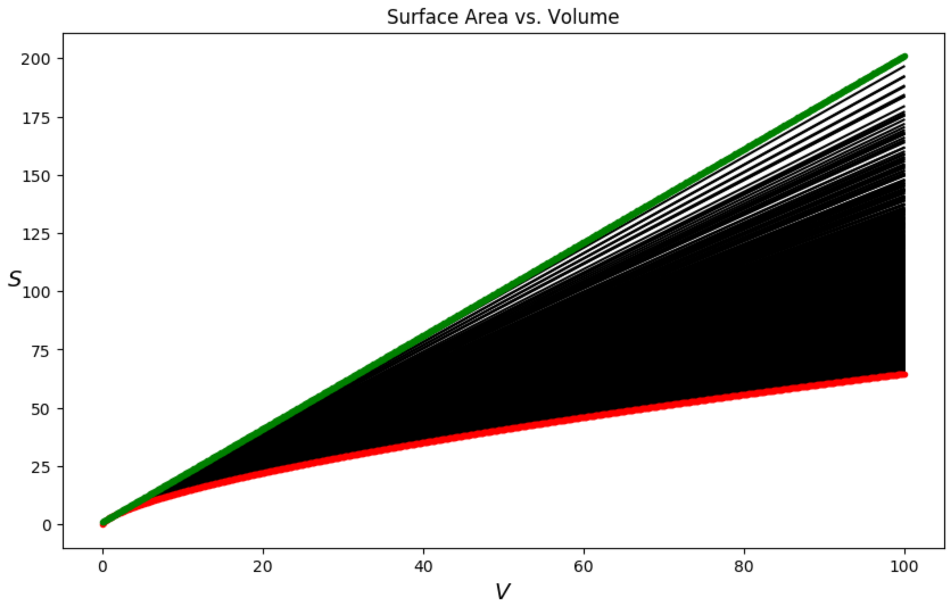

Figure 5: Surface Area vs Volume of a cuboid linear plot

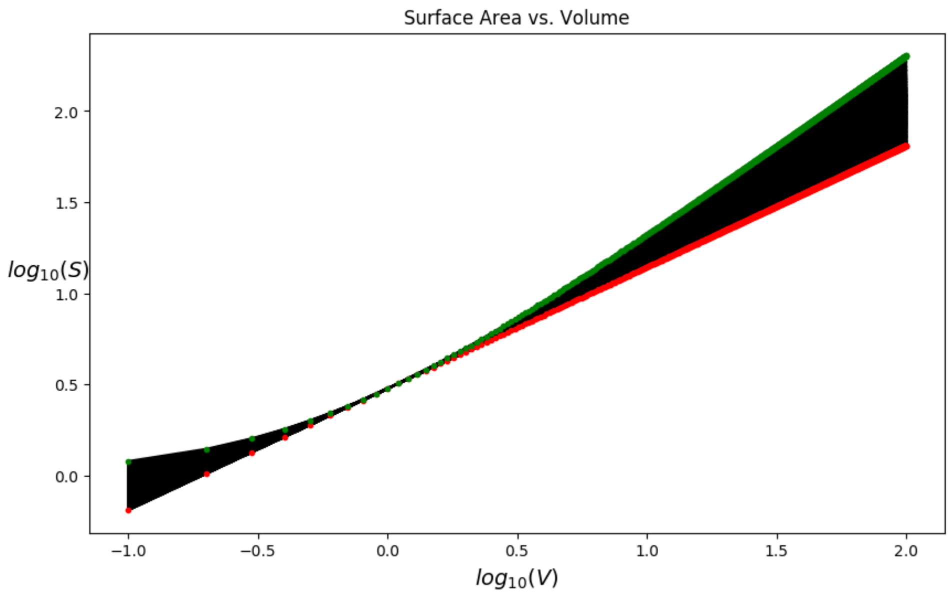

Figure 6: Surface Area vs Volume of a cuboid log-log plot

The two plots above confirm that as Volume scales isometrically ($\gamma_i = \frac{1}{3}$) S scales slower, and as $\gamma_3 \leftarrow 1$ then S scales fastest.

0.9 References

Collaborators:
Maxfield Green, Sophie Hodson

CODE:

```
# Source: https://en.wikipedia.org/wiki/List_of_world_records_in_rowing

#number of oarspeople
N = [1, 1, 2, 2, 2, 2, 4, 4, 4, 8, 8]

#Time (min):
t = [6.5123, 6.723, 6.142, 6.554, 5.995, 6.382, 5.6831, 5.849, 5.539, 5.311,
    5.504]

#boat types (Single Scull, Double Sculls, Coxless, Coxed, Eight, Lightweight):
boat_type = ['Single Scull', 'Light Weight',
             'Coxless', 'Coxed', 'Double Sculls',
             'Light Weight \n coxless', 'Coxless',
             'Coxed', 'Quad', 'Eight', 'Lightweight']

#derive speed:
def speed(time,n_meters=2000):
    """SPEED: returns a speeds for a 2000 meter rowing race given a vector of
        finish times

            ARGS:
                time    : a list of finish times in minutes
                n_meters: number of meters (default 2000)
    """
    min2sec = [t*60 for t in time]
    vel = [3.6*(n_meters/t) for t in min2sec]

    return vel

speeds = speed(t)
```

```
#log-log:
slope, intercept, r_value, p_value, std_err = stats.linregress(np.log10(N),
    np.log10(speeds))
x = np.linspace(0,1,100)
y = slope*x + intercept
#plot:
plt.figure(num=None, figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')

plt.plot(np.log10(N), np.log10(speeds), 'o', label='Observed Record Speeds ')
plt.plot(x,y, label='best fit:' r' $ y = {}x + {}
    $'.format(round(slope,3),round(intercept,3)))

plt.title('Record speeds for rowing 2000m')
plt.xlabel(r' $ log_{10} $(Number of Oarspeople)', fontsize=14)
plt.ylabel(r' $log_{10} $(Speed) (km/hr)', fontsize=14)

#Make pretty:
plt.legend(loc='upper left')
for name, n, s in zip(boat_type, np.log10(N), np.log10(speeds)):
    plt.annotate(
        name,
        xy=(n, s), xytext=(25, 5),
        textcoords='offset points', ha='right', va='bottom')

plt.show()
############################################################################################
#Source:
    https://en.wikipedia.org/wiki/List_of_world_records_in_Olympic_weightlifting

#weights:
weight_classes = np.array([56, 62, 69, 77, 85, 94, 105])

#records:
snatch = np.array([139, 154, 166, 177, 187, 189, 200])
jercln = np.array([171, 183, 198, 214, 220, 233, 246])
total  = np.array([307, 333, 359, 380, 396, 417, 437])
```

```
#regressions:
logwc = np.log10(weight_classes)

log_snatch = np.log10(snatch)
slope_s, intercept_s, r_value_s, p_value_s, std_err_s = stats.linregress(logwc,
    log_snatch)
x = np.linspace(1.5,2.25,100)
y = np.multiply(slope_s, x) + intercept_s
plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.plot(logwc, log_snatch, 'o', color='blue', label='Snatch Record')
plt.plot(x, y, label='best fit: ' r' $ y = {}x + {}
    $'.format(round(slope_s,3),round(intercept_s,3)))

plt.xlabel(r'$ log_{10} $(Weight Class of Lifter) (kg)', fontsize=14)
plt.ylabel(r' $ log_{10} $ (Record Weight Lifted) (kg)', fontsize=14)
plt.title('World Record for Snatch')

plt.legend(loc='upper left')

plt.show()
log_jercln = np.log10(jercln)
slope_jc, intercept_jc, r_value_jc, p_value_jc, std_err_jc =
    stats.linregress(logwc, log_jercln)
x = np.linspace(1.5,2.25,100)
y = np.multiply(slope_jc, x) + intercept_jc
plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.plot(logwc, log_jercln, 'o', color='green', label='Jerk & Clean Record')
plt.plot(x, y, color= 'green',
        label='best fit: ' r' $ y = {}x + {}
            $'.format(round(slope_jc,3),round(intercept_jc,3)))

plt.xlabel(r'$ log_{10} $(Weight Class of Lifter) (kg)', fontsize=14)
plt.ylabel(r' $ log_{10} $ (Record Weight Lifted) (kg)', fontsize=14)
plt.title('World Record for Jerk and Clean')

plt.legend(loc='upper left')
```

```
plt.show()
log_tot = np.log10(total)
slope_t, intercept_t, r_value_t, p_value_t, std_err_t = stats.linregress(logwc,
    log_tot)
x = np.linspace(1.5,2.25,100)
y = np.multiply(slope_t, x) + intercept_t
plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.plot(logwc, log_tot, 'o', color='purple', label='total Record')
plt.plot(x, y, color='purple',
        label='best fit: ' r' $ y = {}x + {}
            $'.format(round(slope_t,3),round(intercept_t,3)))

plt.xlabel(r'$ log_{10} $(Weight Class of Lifter) (kg)', fontsize=14)
plt.ylabel(r' $ log_{10} $ (Record Weight Lifted) (kg)', fontsize=14)
plt.title('World Record for Total')

plt.legend(loc='upper left')

plt.show()
##############################################################################################
norm_snatch = np.divide(snatch,weight_classes)
norm_jrcln = np.divide(jercln,weight_classes)
norm_total = np.divide(total,weight_classes)

log_norm_snatch = np.log10(norm_snatch)
log_norm_jrcln = np.log10(norm_jrcln)
log_norm_total = np.log10(norm_total)
plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
slope_s, intercept_s, r_value_s, p_value_s, std_err_s = stats.linregress(logwc,
    log_norm_snatch)

plt.title('World Record for Snatch')
plt.xlabel(r'$log_{10}$(normalized weight lifted)',fontsize=14)
plt.ylabel(r'$log_{10}$(weight class)',fontsize=14)
plt.plot(logwc, log_norm_snatch, 'o', color='blue')
plt.show()
```

```
plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
slope_jr, intercept_jr, r_value_jr, p_value_jr, std_err_jr =
    stats.linregress(logwc, log_norm_jrcln)

plt.title('World Record for Jerk & Clean')
plt.xlabel(r'$log_{10}$(normalized weight lifted)',fontsize=14)
plt.ylabel(r'$log_{10}$(weight class)',fontsize=14)
plt.plot(logwc, log_norm_jrcln, 'o', color='green')
plt.show()

plt.figure( figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
slope_t, intercept_t, r_value_t, p_value_t, std_err_t = stats.linregress(logwc,
    log_norm_total)

plt.title('World Record for Total')
plt.xlabel(r'$log_{10}$(normalized weight lifted)',fontsize=14)
plt.ylabel(r'$log_{10}$(weight class)',fontsize=14)
plt.plot(logwc, log_norm_total, 'o', color='purple')
plt.show()
##########################################################################################
#pt D: #from NASA: https://nssdc.gsfc.nasa.gov/planetary/factsheet/

#ORDER: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto
planet_names = ['Mercury', 'Venus', 'Earth', 'Mars', 'Jupiter', 'Saturn',
    'Uranus', 'Neptune', 'Pluto']

#orbital period:
t = np.array([88.0, 224.7, 365.2, 687.0, 4331, 10747, 30589, 59800, 90560])

#semi-major axis:
r = np.array([57.9, 108.2, 149.6, 227.9, 778.6, 1433.5, 2872.5, 4495.1, 5906.4])

slope, intercept, r_value, p_value, std_err =
    stats.linregress(np.log10(t),np.log10(r))

x = np.linspace(0, 6,100)
y = slope*x + intercept
```

```
s = round(slope,4)
i = round(intercept,4)
#plot:
plt.figure(num=None, figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.plot(np.log10(t), np.log10(r), 'o', label = 'Planets in our solar system')
plt.plot(x,y, label='best fit: 'r'$ y = {}x + {} $'.format(s,i))

#make pretty:)
plt.title('Orbital period ' r'$ \tau $ (Days) vs. Semi-Major Axis ' r'$ r $
    (Distance From Sun' r' $ (10^6 km) $)')
plt.xlabel(r'$log_{10}( \tau )$', fontsize=14)
plt.ylabel(r'$ log_{10}(r) $', fontsize=14)
plt.legend(loc='upper left')
for name, t, r in zip(planet_names, np.log10(t), np.log10(r)):
    plt.annotate(
        name,
        xy=(t, r), xytext=(-10, 5),
        textcoords='offset points', ha='right', va='bottom')

plt.show()
#############################################################################################
#gammas:
g_1 = np.linspace(0,1,100)
g_2 = np.linspace(0,1,100)
g_3 = np.linspace(0,1,100)


gammas= []
#Volume:
V = np.linspace(0,100,1000)

SA = []

for gamma_3 in g_3:
    for gamma_2 in g_2:
        for gamma_1 in g_3:
```

```
            if gamma_3 + gamma_2 + gamma_1 == 1 and gamma_3 >= gamma_2 >=
                gamma_1:
                gammas.append([gamma_1, gamma_2, gamma_3])

plt.figure(num=None, figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.title('Surface Area vs. Volume')
plt.ylabel(r'$ S $',rotation='horizontal',fontsize=14)
plt.xlabel(r'$ V $',fontsize=14)
for params in gammas:
    plt.plot(V, np.power(V,params[0]+params[1])+np.power(V,params[0]+params[2])
        + np.power(V,params[1]+params[2]),
             '-', color='black')

#isometric scaling:
plt.plot(V,
    np.power(V,gammas[0][0]+gammas[0][1])+np.power(V,gammas[0][0]+gammas[0][2])
    + np.power(V,gammas[0][1]+gammas[0][2]),
             '.', color='red')

#allometric scaling:
plt.plot(V,
    np.power(V,gammas[674][0]+gammas[674][1])+np.power(V,gammas[674][0]+gammas[674][2])
    + np.power(V,gammas[674][1]+gammas[674][2]),
             '.', color='green')


plt.figure(num=None, figsize=(10, 6), dpi=100, facecolor='w', edgecolor='k')
plt.title('Surface Area vs. Volume')
plt.ylabel(r'$ log_{10}(S) $',rotation='horizontal',fontsize=14)
plt.xlabel(r'$ log_{10}(V) $',fontsize=14)
for params in gammas:
    plt.plot(np.log10(V),
        np.log10(np.power(V,params[0]+params[1])+np.power(V,params[0]+params[2])
        + np.power(V,params[1]+params[2])),
             '-', color='black')

#isometric scaling
```

```
plt.plot(np.log10(V),
    np.log10(np.power(V,gammas[0][0]+gammas[0][1])+np.power(V,gammas[0][0]+gammas[0][2])
    + np.power(V,gammas[0][1]+gammas[0][2])),
            '.', color='red')

#allometric
plt.plot(np.log10(V),
    np.log10(np.power(V,gammas[674][0]+gammas[674][1])+np.power(V,gammas[674][0]+gammas[674][
    + np.power(V,gammas[674][1]+gammas[674][2])),
            '.', color='green')
```