arm_pendulum_modeling

October 19, 2021

1 Arm Motion Modeling

1.1 System Description

A double-pendulum system hanging in gravity is shown in the figure above. $q = [\theta_1, \theta_2]$ are the system configuration variables. We assume the z-axis is pointing out from the screen/paper, thus the positive direction of rotation is counter-clockwise. The solution steps are: 1. Computing the Lagrangian of the system. 2. Computing the Euler-Lagrange equations, and solve them for $\ddot{\theta}_1$ and $\ddot{\theta}_2$. 3. Numerically evaluating the solutions for τ_1 and τ_2, and simulating the system for θ_1, θ_2, $\dot{\theta}_1$, $\dot{\theta}_2$, $\ddot{\theta}_1$ and $\ddot{\theta}_2$. 4. Animating the simulation.

[1]:
```
from IPython.core.display import HTML
display(HTML("<table><tr><td><img src='./double-pendulum-diagram.png'␣
↪width=450' height='300'></table>"))
```

```
<IPython.core.display.HTML object>
```

1.2 Import Libraries and Define System Constants

Import libraries:

[9]:
```
# Imports required for data processing
import os
import csv
import pandas as pd

# Imports required for dynamics calculations
import sympy
from sympy.abc import t
from sympy import symbols, Eq, Function, solve, sin, cos, Matrix, Subs,␣
↪substitution, Derivative, simplify, symbols, lambdify
import math
from math import pi
import numpy as np
import matplotlib.pyplot as plt

# Imports required for animation
from plotly.offline import init_notebook_mode, iplot
from IPython.display import display, HTML
```

```python
import plotly.graph_objects as go
```

Define the system's constants:

[10]:
```python
# Masses, length and center-of-mass positions (calculated using the lab␣
↪measurements)
# Mass calculations (mass unit is kg)
# m_body = 90.6                         # Average weights for American adult␣
↪male
#                                       # from "Anthropometric Reference Data␣
↪for Children and Adults:
#                                       # United States, 2015-2018"
m_body_dict = {'ID': 51, 'JD': 79.5, 'JR': 76, 'KS': 59.3, 'KW': 63.8, 'LC': 61.
↪2,
               'LD': 97.3, 'LS': 82.2, 'MK': 93.5, 'MV': 98.5, 'SM': 68.5, 'TD':
↪ 70,
               'TM': 66.2}

# m_upper_arm = 0.028 * m_body          # Average upper arm weights relative␣
↪to body weight, from "Biomechanics
#                                       # and Motor Control of Human Movement"␣
↪by David Winter (2009), 4th edition
m_upper_arm_dict = {'ID': 0.028 * m_body_dict['ID'], 'JD': 0.028 *␣
↪m_body_dict['JD'],
                    'JR': 0.028 * m_body_dict['JR'], 'KS': 0.028 *␣
↪m_body_dict['KS'],
                    'KW': 0.028 * m_body_dict['KW'], 'LC': 0.028 *␣
↪m_body_dict['LC'],
                    'LD': 0.028 * m_body_dict['LD'], 'LS': 0.028 *␣
↪m_body_dict['LS'],
                    'MK': 0.028 * m_body_dict['MK'], 'MV': 0.028 *␣
↪m_body_dict['MV'],
                    'SM': 0.028 * m_body_dict['SM'], 'TD': 0.028 *␣
↪m_body_dict['TD'],
                    'TM': 0.028 * m_body_dict['TM']}

m_lower_arm = 0.7395                    # Average lower prosthetics weights,␣
↪calculated using lab measurements



# Arm length calculations (length unit is m)
# H_body = 1.769                        # Average height for American adult␣
↪male, from "Height and body-mass
#                                       # index trajectories of school-aged␣
↪children and adolescents from
```

```
#                                        # 1985 to 2019 in 200 countries and␣
↪territories: a pooled analysis
#                                        # of 2181 population-based studies␣
↪with 65 million participants"
H_body_dict = {'ID': 1.62, 'JD': 1.76, 'JR': 1.77, 'KS': 1.64, 'KW': 1.62, 'LC':
↪ 1.58,
               'LD': 1.875, 'LS': 1.635, 'MK': 1.78, 'MV': 1.805, 'SM': 1.79,␣
↪'TD': 1.69,
               'TM': 1.735}

# L_upper_arm = 0.186 * H_body           # Average upper arm length relative to␣
↪body height
#                                        # from "Biomechanics and Motor Control␣
↪of Human Movement" by David
#                                        # Winter (2009), 4th edition
L_upper_arm_dict = {'ID': 0.186 * H_body_dict['ID'], 'JD': 0.186 *␣
↪H_body_dict['JD'],
                    'JR': 0.186 * H_body_dict['JR'], 'KS': 0.186 *␣
↪H_body_dict['KS'],
                    'KW': 0.186 * H_body_dict['KW'], 'LC': 0.186 *␣
↪H_body_dict['LC'],
                    'LD': 0.186 * H_body_dict['LD'], 'LS': 0.186 *␣
↪H_body_dict['LS'],
                    'MK': 0.186 * H_body_dict['MK'], 'MV': 0.186 *␣
↪H_body_dict['MV'],
                    'SM': 0.186 * H_body_dict['SM'], 'TD': 0.186 *␣
↪H_body_dict['TD'],
                    'TM': 0.186 * H_body_dict['TM']}

L_lower_arm = 0.42                      # Average lower prosthetics length,␣
↪calculated using lab measurements



# Arm center of mass length calculations (length unit is m)
# L_upper_arm_COM = 0.436 * L_upper_arm  # Average upper arm length from␣
↪shoulder to center of mass relative
#                                        # to upper arm length, from␣
↪"Biomechanics and Motor Control of Human
#                                        # Movement" by David Winter (2009),␣
↪4th edition
L_upper_arm_COM_dict = {'ID': 0.436 * L_upper_arm_dict['ID'], 'JD': 0.436 *␣
↪L_upper_arm_dict['JD'],
                        'JR': 0.436 * L_upper_arm_dict['JR'], 'KS': 0.436 *␣
↪L_upper_arm_dict['KS'],
```

```
                           'KW': 0.436 * L_upper_arm_dict['KW'], 'LC': 0.436 *␣
↪L_upper_arm_dict['LC'],
                           'LD': 0.436 * L_upper_arm_dict['LD'], 'LS': 0.436 *␣
↪L_upper_arm_dict['LS'],
                           'MK': 0.436 * L_upper_arm_dict['MK'], 'MV': 0.436 *␣
↪L_upper_arm_dict['MV'],
                           'SM': 0.436 * L_upper_arm_dict['SM'], 'TD': 0.436 *␣
↪L_upper_arm_dict['TD'],
                           'TM': 0.436 * L_upper_arm_dict['TM']}

L_lower_arm_COM = 0.2388               # Average lower prosthetics length from␣
↪elbow to center of mass,
                                       # calculated using lab measurements
```

1.3 Extracting Data

Extracting angles data and computing angular velocities and angular accelerations from the angles:

[27]:
```
def calculate_Vel(Ang_list, time_list, index):
    return ((Ang_list[index + 1] - Ang_list[index])
          / (time_list[index + 1] - time_list[index]))


def calculate_Acc(Vel_list, time_list, index):
    return ((Vel_list[index + 1] - Vel_list[index])
          / (time_list[index + 1] - time_list[index]))


data_csv_dir = '../../data/control_data/CSV Converted Files'
frame_frequency = 120
print("current directory: ", os.getcwd())

participants_list = []
time_list = []
Elbow_Ang_list = []
sholder_Ang_list = []
Elbow_Vel_list = []
sholder_Vel_list = []
Elbow_Acc_list = []
sholder_Acc_list = []

for file in os.listdir(data_csv_dir):
    file_name = file.split(".")[0]
    participant_name = file.split("_")[0]

    if file.endswith(".csv"):
        frame = 0
```

```
        file_time_list = []
        file_R_Elbow_Ang_list = []
        file_R_sholder_Ang_list = []
        file_L_Elbow_Ang_list = []
        file_L_sholder_Ang_list = []
        file_R_Elbow_Vel_list = []
        file_R_sholder_Vel_list = []
        file_L_Elbow_Vel_list = []
        file_L_sholder_Vel_list = []
        file_R_Elbow_Acc_list = []
        file_R_sholder_Acc_list = []
        file_L_Elbow_Acc_list = []
        file_L_sholder_Acc_list = []

        data_path = os.path.join(data_csv_dir, file)

        # Cutting out weird data behavior on data edges
        if file == 'TD_WN7.csv':
            data_rows = open(data_path).read().strip().split("\n")[40:]
        elif file == 'TD_WN4.csv':
            data_rows = open(data_path).read().strip().split("\n")[24:-12]
        elif file == 'TD_WN11.csv':
            data_rows = open(data_path).read().strip().split("\n")[24:-3]
        else:
            data_rows = open(data_path).read().strip().split("\n")[24:]

        # Extract time [sec], elbow angles [rad], and shoulder angles [rad]␣
↪from data
        for row in data_rows:
            splitted_row = row.strip().split("\t")

            # Check if loop finished all data
            if len(splitted_row) < 80:
                break

            file_time_list.append(frame/frame_frequency)
            file_R_sholder_Ang_list.append(float(splitted_row[11]) * 2*pi/360)
            file_R_Elbow_Ang_list.append(float(splitted_row[9]) * 2*pi/360)
            file_L_sholder_Ang_list.append(float(splitted_row[23]) * 2*pi/360)
            file_L_Elbow_Ang_list.append(float(splitted_row[21]) * 2*pi/360)
            frame += 1

        # Extract elbow and shoulder velocities [rad/sec] from angles
        for i in range(len(file_time_list) - 1):
            R_Elbow_Vel = calculate_Vel(file_R_Elbow_Ang_list, file_time_list,␣
↪i)
```

```
            R_sholder_Vel = calculate_Vel(file_R_sholder_Ang_list,␣
↪file_time_list, i)
            L_Elbow_Vel = calculate_Vel(file_L_Elbow_Ang_list, file_time_list,␣
↪i)
            L_sholder_Vel = calculate_Vel(file_L_sholder_Ang_list,␣
↪file_time_list, i)

            file_R_Elbow_Vel_list.append(R_Elbow_Vel)
            file_R_sholder_Vel_list.append(R_sholder_Vel)
            file_L_Elbow_Vel_list.append(L_Elbow_Vel)
            file_L_sholder_Vel_list.append(L_sholder_Vel)

        # Extract elbow and shoulder Accelerations [rad/sec^2] from velocities
        for i in range(len(file_time_list) - 2):
            R_Elbow_Acc = calculate_Acc(file_R_Elbow_Vel_list, file_time_list,␣
↪i)
            R_sholder_Acc = calculate_Acc(file_R_sholder_Vel_list,␣
↪file_time_list, i)
            L_Elbow_Acc = calculate_Acc(file_L_Elbow_Vel_list, file_time_list,␣
↪i)
            L_sholder_Acc = calculate_Acc(file_L_sholder_Vel_list,␣
↪file_time_list, i)

            file_R_Elbow_Acc_list.append(R_Elbow_Acc)
            file_R_sholder_Acc_list.append(R_sholder_Acc)
            file_L_Elbow_Acc_list.append(L_Elbow_Acc)
            file_L_sholder_Acc_list.append(L_sholder_Acc)

        # Adjust lists length
        file_time_list = file_time_list[:-2]
        file_R_Elbow_Ang_list = file_R_Elbow_Ang_list[:-2]
        file_R_sholder_Ang_list = file_R_sholder_Ang_list[:-2]
        file_L_Elbow_Ang_list = file_L_Elbow_Ang_list[:-2]
        file_L_sholder_Ang_list = file_L_sholder_Ang_list[:-2]

        file_R_Elbow_Vel_list = file_R_Elbow_Vel_list[:-1]
        file_R_sholder_Vel_list = file_R_sholder_Vel_list[:-1]
        file_L_Elbow_Vel_list = file_L_Elbow_Vel_list[:-1]
        file_L_sholder_Vel_list = file_L_sholder_Vel_list[:-1]

        participants_list.append(participant_name)
        participants_list.append(participant_name)

        time_list.append(file_time_list)
        time_list.append(file_time_list)
```

```
        Elbow_Ang_list.append(file_R_Elbow_Ang_list)
        sholder_Ang_list.append(file_R_sholder_Ang_list)
        Elbow_Ang_list.append(file_L_Elbow_Ang_list)
        sholder_Ang_list.append(file_L_sholder_Ang_list)
        Elbow_Vel_list.append(file_R_Elbow_Vel_list)
        sholder_Vel_list.append(file_R_sholder_Vel_list)
        Elbow_Vel_list.append(file_L_Elbow_Vel_list)
        sholder_Vel_list.append(file_L_sholder_Vel_list)
        Elbow_Acc_list.append(file_R_Elbow_Acc_list)
        sholder_Acc_list.append(file_R_sholder_Acc_list)
        Elbow_Acc_list.append(file_L_Elbow_Acc_list)
        sholder_Acc_list.append(file_L_sholder_Acc_list)
```

```
current directory:
/home/yael/Documents/MSR_Courses/ME499-Final_Project/Motorized-Prosthetic-
Arm/motor_control/arm_pendulum_modeling
```

1.4 System Modeling

Computing the Lagrangian of the system:

[28]:
```
m1, m2, g, R1, R1_COM, R2, R2_COM = symbols(r'm1, m2, g, R1, R1_COM, R2,␣
 ↪R2_COM')

# The system torque variables as function of t
tau1 = Function(r'tau1')(t)
tau2 = Function(r'tau2')(t)

# The system configuration variables as function of t
theta1 = Function(r'theta1')(t)
theta2 = Function(r'theta2')(t)

# The velocity as derivative of position wrt t
theta1_dot = theta1.diff(t)
theta2_dot = theta2.diff(t)

# The acceleration as derivative of velocity wrt t
theta1_ddot = theta1_dot.diff(t)
theta2_ddot = theta2_dot.diff(t)

# Converting the polar coordinates to cartesian coordinates
x1 = R1_COM * sin(theta1)
x2 = R1 * sin(theta1) + R2_COM * sin(theta1 + theta2)

y1 = -R1_COM * cos(theta1)
y2 = -R1 * cos(theta1) - R2_COM * cos(theta1 + theta2)

# Calculating the kinetic and potential energy of the system
```

```
KE = 1/2 * m1 * ((x1.diff(t))**2 + (y1.diff(t))**2) + 1/2 * m2 * ((x2.
 ↪diff(t))**2 + (y2.diff(t))**2)
PE = m1 * g * y1 + m2 * g * y2

# Computing the Lagrangian
L = simplify(KE - PE)
print('L: ')
display(L)
```

L:

$$0.5R_{1COM}^2 m_1 \left(\frac{d}{dt}\theta_1(t)\right)^2 + R_{1COM} g m_1 \cos(\theta_1(t)) + g m_2 (R_1 \cos(\theta_1(t)) + R_{2COM} \cos(\theta_1(t) + \theta_2(t))) +$$
$$0.5m_2 \left(R_1^2 \left(\frac{d}{dt}\theta_1(t)\right)^2 + 2R_1 R_{2COM} \cos(\theta_2(t)) \left(\frac{d}{dt}\theta_1(t)\right)^2 + 2R_1 R_{2COM} \cos(\theta_2(t)) \frac{d}{dt}\theta_1(t) \frac{d}{dt}\theta_2(t) + R_{2COM}^2 \left(\frac{d}{dt}\theta\right.\right.$$

Computing the Euler-Lagrange equations:

[29]:
```
# Define the derivative of L wrt the functions: x, xdot
L_dtheta1 = L.diff(theta1)
L_dtheta2 = L.diff(theta2)

L_dtheta1_dot = L.diff(theta1_dot)
L_dtheta2_dot = L.diff(theta2_dot)

# Define the derivative of L_dxdot wrt to time t
L_dtheta1_dot_dt = L_dtheta1_dot.diff(t)
L_dtheta2_dot_dt = L_dtheta2_dot.diff(t)

# Define the left hand side of the the Euler-Lagrange as a matrix
lhs = Matrix([simplify(L_dtheta1_dot_dt - L_dtheta1),
              simplify(L_dtheta2_dot_dt - L_dtheta2)])

# Define the right hand side of the the Euler-Lagrange as a Matrix
rhs = Matrix([tau1, tau2])

# Compute the Euler-Lagrange equations as a matrix
EL_eqns = Eq(lhs, rhs)

print('Euler-Lagrange matrix for this systems:')
display(EL_eqns)
```

Euler-Lagrange matrix for this systems:

$$\left[\begin{matrix} 1.0R_{1COM}^2 m_1 \frac{d^2}{dt^2}\theta_1(t) + R_{1COM} g m_1 \sin(\theta_1(t)) + g m_2 (R_1 \sin(\theta_1(t)) + R_{2COM} \sin(\theta_1(t) + \theta_2(t))) + m_2 \left(R_1^2 \frac{d^2}{dt^2}\theta_1(t\right. \\ R_{2COM} m_2 \left(R_1 \sin(\theta_2(t)) \left(\frac{d}{dt}\right.\right. \end{matrix}\right.$$

$$\begin{bmatrix} \tau_1(t) \\ \tau_2(t) \end{bmatrix}$$

Solve the equations for τ_1 and τ_2:

[30]:
```
# Solve the Euler-Lagrange equations for the shoulder and elbow torques
T = Matrix([tau1, tau2])
soln = solve(EL_eqns, T, dict=True)

# Initialize the solutions
solution = [0, 0]
i = 0

for sol in soln:
    for v in T:
        solution[i] = simplify(sol[v])
        display(Eq(T[i], solution[i]))
        i =+ 1
```

$$\tau_1(t) = R_1^2 m_2 \frac{d^2}{dt^2}\theta_1(t) - 2.0 R_1 R_{2COM} m_2 \sin(\theta_2(t)) \frac{d}{dt}\theta_1(t) \frac{d}{dt}\theta_2(t) - R_1 R_{2COM} m_2 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_2(t)\right)^2 + 2.0 R_1 R_{2COM} m_2 \cos(\theta_2(t)) \frac{d^2}{dt^2}\theta_1(t) + R_1 R_{2COM} m_2 \cos(\theta_2(t)) \frac{d^2}{dt^2}\theta_2(t) + R_1 g m_2 \sin(\theta_1(t)) + R_{1COM}^2 m_1 \frac{d^2}{dt^2}\theta_1(t) + R_{1COM} g m_1 \sin(\theta_1(t)) + R_{2COM}^2 m_2 \frac{d^2}{dt^2}\theta_1(t) + R_{2COM}^2 m_2 \frac{d^2}{dt^2}\theta_2(t) + R_{2COM} g m_2 \sin(\theta_1(t) + \theta_2(t))$$

$$\tau_2(t) = R_{2COM} m_2 \left(R_1 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_1(t)\right)^2 + R_1 \cos(\theta_2(t)) \frac{d^2}{dt^2}\theta_1(t) + R_{2COM} \frac{d^2}{dt^2}\theta_1(t) + R_{2COM} \frac{d^2}{dt^2}\theta_2(t) + g \sin$$

Simulating the system:

[31]:
```
# Substitute the derivative variables with a dummy variables and plug-in the␣
↪constants
solution_0_subs = solution[0]
solution_1_subs = solution[1]

theta1_dot_dummy = symbols('thetadot1')
theta2_dot_dummy = symbols('thetadot2')
theta1_ddot_dummy = symbols('thetaddot1')
theta2_ddot_dummy = symbols('thetaddot2')

solution_0_subs = solution_0_subs.subs([(g, 9.81)])
solution_1_subs = solution_1_subs.subs([(g, 9.81)])

solution_0_subs = solution_0_subs.subs([((theta1.diff(t)).diff(t),␣
↪theta1_ddot_dummy),
                                        ((theta2.diff(t)).diff(t),␣
↪theta2_ddot_dummy)])
solution_1_subs = solution_1_subs.subs([((theta1.diff(t)).diff(t),␣
↪theta1_ddot_dummy),
```

```
                                        ((theta2.diff(t)).diff(t),␣
↪theta2_ddot_dummy)])

solution_0_subs = solution_0_subs.subs([(theta1.diff(t), theta1_dot_dummy),
                                        (theta2.diff(t), theta2_dot_dummy)])
solution_1_subs = solution_1_subs.subs([(theta1.diff(t), theta1_dot_dummy),
                                        (theta2.diff(t), theta2_dot_dummy)])

# Lambdify the thetas and its derivatives
func1 = lambdify([theta1, theta2, theta1_dot_dummy, theta2_dot_dummy,␣
↪theta1_ddot_dummy,
                  theta2_ddot_dummy, m1, m2, R1, R2, R1_COM, R2_COM],␣
↪solution_0_subs, modules = sympy)
func2 = lambdify([theta1, theta2, theta1_dot_dummy, theta2_dot_dummy,␣
↪theta1_ddot_dummy,
                  theta2_ddot_dummy, m1, m2, R1, R2, R1_COM, R2_COM],␣
↪solution_1_subs, modules = sympy)

# Initialize the torque and power lists
sholder_tau_list, Elbow_tau_list = [], []
sholder_current_list, Elbow_current_list = [], []
sholder_power_list, Elbow_power_list = [], []

motor_kv = 115
torque_const = 8.27 / motor_kv

for i in range(len(time_list)):
    # Initialize the torque and power lists
    tau1_list, tau2_list = [], []
    current1_list, current2_list = [], []
    power1_list, power2_list = [], []

    t_list = time_list[i]
    theta1_list = sholder_Ang_list[i]
    theta2_list = Elbow_Ang_list[i]
    dtheta1_list = sholder_Vel_list[i]
    dtheta2_list = Elbow_Vel_list[i]
    ddtheta1_list = sholder_Acc_list[i]
    ddtheta2_list = Elbow_Acc_list[i]

    # Plug-in the angles, angular velocities and angular accelerations for␣
↪every time step to find the torques
    for j in range(len(t_list)):
        tau1_list.append(func1(theta1_list[j], theta2_list[j], dtheta1_list[j],␣
↪dtheta2_list[j],
```

```
                                  ddtheta1_list[j], ddtheta2_list[j],␣
↪m_upper_arm_dict[participants_list[i]],
                                  m_lower_arm,␣
↪L_upper_arm_dict[participants_list[i]], L_lower_arm,
                                  L_upper_arm_COM_dict[participants_list[i]],␣
↪L_lower_arm_COM))

        tau2_list.append(func2(theta1_list[j], theta2_list[j], dtheta1_list[j],␣
↪dtheta2_list[j],
                                  ddtheta1_list[j], ddtheta2_list[j],␣
↪m_upper_arm_dict[participants_list[i]],
                                  m_lower_arm,␣
↪L_upper_arm_dict[participants_list[i]], L_lower_arm,
                                  L_upper_arm_COM_dict[participants_list[i]],␣
↪L_lower_arm_COM))

        # Calculate the current required to reach the required joints torques␣
↪for every time step
        current1_list.append(torque_const * tau1_list[j])
        current2_list.append(torque_const * tau2_list[j])

        # Calculate the power required to reach the required angular velocities␣
↪and joints torques for every time step
        power1_list.append(dtheta1_list[j] * tau1_list[j])
        power2_list.append(dtheta2_list[j] * tau2_list[j])

    sholder_tau_list.append(tau1_list)
    Elbow_tau_list.append(tau2_list)

    sholder_current_list.append(current1_list)
    Elbow_current_list.append(current2_list)

    sholder_power_list.append(power1_list)
    Elbow_power_list.append(power2_list)

    print(f"Trial {i}/{len(time_list) - 1} finished \t maximum torque is␣
↪{format(max(tau2_list), '.3f')} [Nm]\t maximum power is␣
↪{format(max(power2_list), '.3f')} [W]")
```

```
Trial 0/203 finished     maximum torque is 1.724 [Nm]    maximum power is 0.549
[W]
Trial 1/203 finished     maximum torque is 2.262 [Nm]    maximum power is 1.483
[W]
Trial 2/203 finished     maximum torque is 2.195 [Nm]    maximum power is 3.079
[W]
Trial 3/203 finished     maximum torque is 2.162 [Nm]    maximum power is 2.907
[W]
```

```
Trial 4/203 finished     maximum torque is 2.016 [Nm]    maximum power is 3.069
[W]
Trial 5/203 finished     maximum torque is 2.346 [Nm]    maximum power is 3.666
[W]
Trial 6/203 finished     maximum torque is 2.157 [Nm]    maximum power is 1.753
[W]
Trial 7/203 finished     maximum torque is 3.857 [Nm]    maximum power is 1.155
[W]
Trial 8/203 finished     maximum torque is 2.269 [Nm]    maximum power is 3.146
[W]
Trial 9/203 finished     maximum torque is 2.113 [Nm]    maximum power is 2.225
[W]
Trial 10/203 finished    maximum torque is 2.101 [Nm]    maximum power is 2.142
[W]
Trial 11/203 finished    maximum torque is 2.669 [Nm]    maximum power is 4.011
[W]
Trial 12/203 finished    maximum torque is 1.848 [Nm]    maximum power is 2.279
[W]
Trial 13/203 finished    maximum torque is 1.527 [Nm]    maximum power is 1.073
[W]
Trial 14/203 finished    maximum torque is 2.246 [Nm]    maximum power is 2.050
[W]
Trial 15/203 finished    maximum torque is 1.959 [Nm]    maximum power is 2.019
[W]
Trial 16/203 finished    maximum torque is 2.151 [Nm]    maximum power is 1.810
[W]
Trial 17/203 finished    maximum torque is 2.532 [Nm]    maximum power is 3.662
[W]
Trial 18/203 finished    maximum torque is 1.784 [Nm]    maximum power is 1.620
[W]
Trial 19/203 finished    maximum torque is 2.044 [Nm]    maximum power is 3.226
[W]
Trial 20/203 finished    maximum torque is 2.215 [Nm]    maximum power is 2.900
[W]
Trial 21/203 finished    maximum torque is 2.092 [Nm]    maximum power is 2.660
[W]
Trial 22/203 finished    maximum torque is 2.098 [Nm]    maximum power is 1.675
[W]
Trial 23/203 finished    maximum torque is 1.903 [Nm]    maximum power is 1.349
[W]
Trial 24/203 finished    maximum torque is 1.877 [Nm]    maximum power is 0.604
[W]
Trial 25/203 finished    maximum torque is 2.079 [Nm]    maximum power is 1.186
[W]
Trial 26/203 finished    maximum torque is 1.862 [Nm]    maximum power is 2.632
[W]
Trial 27/203 finished    maximum torque is 1.639 [Nm]    maximum power is 1.742
[W]
```

```
Trial 28/203 finished    maximum torque is 2.330 [Nm]    maximum power is 3.189
[W]
Trial 29/203 finished    maximum torque is 2.033 [Nm]    maximum power is 1.699
[W]
Trial 30/203 finished    maximum torque is 1.781 [Nm]    maximum power is 1.580
[W]
Trial 31/203 finished    maximum torque is 1.444 [Nm]    maximum power is 1.096
[W]
Trial 32/203 finished    maximum torque is 1.926 [Nm]    maximum power is 3.662
[W]
Trial 33/203 finished    maximum torque is 2.176 [Nm]    maximum power is 4.196
[W]
Trial 34/203 finished    maximum torque is 1.834 [Nm]    maximum power is 2.813
[W]
Trial 35/203 finished    maximum torque is 2.126 [Nm]    maximum power is 3.386
[W]
Trial 36/203 finished    maximum torque is 1.714 [Nm]    maximum power is 1.628
[W]
Trial 37/203 finished    maximum torque is 1.604 [Nm]    maximum power is 1.134
[W]
Trial 38/203 finished    maximum torque is 1.760 [Nm]    maximum power is 1.442
[W]
Trial 39/203 finished    maximum torque is 1.981 [Nm]    maximum power is 2.659
[W]
Trial 40/203 finished    maximum torque is 1.888 [Nm]    maximum power is 0.964
[W]
Trial 41/203 finished    maximum torque is 2.331 [Nm]    maximum power is 1.417
[W]
Trial 42/203 finished    maximum torque is 1.755 [Nm]    maximum power is 1.278
[W]
Trial 43/203 finished    maximum torque is 1.558 [Nm]    maximum power is 0.905
[W]
Trial 44/203 finished    maximum torque is 2.182 [Nm]    maximum power is 2.536
[W]
Trial 45/203 finished    maximum torque is 2.260 [Nm]    maximum power is 4.148
[W]
Trial 46/203 finished    maximum torque is 1.812 [Nm]    maximum power is 0.658
[W]
Trial 47/203 finished    maximum torque is 2.467 [Nm]    maximum power is 1.763
[W]
Trial 48/203 finished    maximum torque is 1.725 [Nm]    maximum power is 2.373
[W]
Trial 49/203 finished    maximum torque is 1.601 [Nm]    maximum power is 1.229
[W]
Trial 50/203 finished    maximum torque is 2.490 [Nm]    maximum power is 6.810
[W]
Trial 51/203 finished    maximum torque is 2.403 [Nm]    maximum power is 2.430
[W]
```

```
Trial 52/203 finished    maximum torque is 2.869 [Nm]    maximum power is 1.375
[W]
Trial 53/203 finished    maximum torque is 1.446 [Nm]    maximum power is 1.393
[W]
Trial 54/203 finished    maximum torque is 1.987 [Nm]    maximum power is 1.902
[W]
Trial 55/203 finished    maximum torque is 1.536 [Nm]    maximum power is 1.430
[W]
Trial 56/203 finished    maximum torque is 2.087 [Nm]    maximum power is 2.283
[W]
Trial 57/203 finished    maximum torque is 2.191 [Nm]    maximum power is 2.616
[W]
Trial 58/203 finished    maximum torque is 2.107 [Nm]    maximum power is 1.425
[W]
Trial 59/203 finished    maximum torque is 2.375 [Nm]    maximum power is 1.704
[W]
Trial 60/203 finished    maximum torque is 1.938 [Nm]    maximum power is 0.834
[W]
Trial 61/203 finished    maximum torque is 1.921 [Nm]    maximum power is 1.301
[W]
Trial 62/203 finished    maximum torque is 2.376 [Nm]    maximum power is 5.876
[W]
Trial 63/203 finished    maximum torque is 2.244 [Nm]    maximum power is 3.340
[W]
Trial 64/203 finished    maximum torque is 2.061 [Nm]    maximum power is 2.558
[W]
Trial 65/203 finished    maximum torque is 2.100 [Nm]    maximum power is 2.625
[W]
Trial 66/203 finished    maximum torque is 1.663 [Nm]    maximum power is 1.312
[W]
Trial 67/203 finished    maximum torque is 2.116 [Nm]    maximum power is 3.063
[W]
Trial 68/203 finished    maximum torque is 1.895 [Nm]    maximum power is 2.606
[W]
Trial 69/203 finished    maximum torque is 3.397 [Nm]    maximum power is 5.689
[W]
Trial 70/203 finished    maximum torque is 1.729 [Nm]    maximum power is 1.053
[W]
Trial 71/203 finished    maximum torque is 1.908 [Nm]    maximum power is 2.222
[W]
Trial 72/203 finished    maximum torque is 2.131 [Nm]    maximum power is 2.015
[W]
Trial 73/203 finished    maximum torque is 2.156 [Nm]    maximum power is 1.733
[W]
Trial 74/203 finished    maximum torque is 1.723 [Nm]    maximum power is 1.493
[W]
Trial 75/203 finished    maximum torque is 1.992 [Nm]    maximum power is 2.875
[W]
```

```
Trial 76/203 finished    maximum torque is 1.952 [Nm]    maximum power is 0.733
[W]
Trial 77/203 finished    maximum torque is 1.864 [Nm]    maximum power is 0.840
[W]
Trial 78/203 finished    maximum torque is 1.709 [Nm]    maximum power is 1.120
[W]
Trial 79/203 finished    maximum torque is 1.709 [Nm]    maximum power is 2.075
[W]
Trial 80/203 finished    maximum torque is 2.027 [Nm]    maximum power is 2.190
[W]
Trial 81/203 finished    maximum torque is 2.463 [Nm]    maximum power is 3.874
[W]
Trial 82/203 finished    maximum torque is 1.872 [Nm]    maximum power is 2.846
[W]
Trial 83/203 finished    maximum torque is 1.907 [Nm]    maximum power is 2.994
[W]
Trial 84/203 finished    maximum torque is 1.912 [Nm]    maximum power is 0.776
[W]
Trial 85/203 finished    maximum torque is 2.063 [Nm]    maximum power is 0.752
[W]
Trial 86/203 finished    maximum torque is 1.982 [Nm]    maximum power is 2.788
[W]
Trial 87/203 finished    maximum torque is 1.975 [Nm]    maximum power is 2.702
[W]
Trial 88/203 finished    maximum torque is 2.201 [Nm]    maximum power is 2.370
[W]
Trial 89/203 finished    maximum torque is 1.958 [Nm]    maximum power is 1.672
[W]
Trial 90/203 finished    maximum torque is 2.029 [Nm]    maximum power is 1.471
[W]
Trial 91/203 finished    maximum torque is 2.095 [Nm]    maximum power is 1.522
[W]
Trial 92/203 finished    maximum torque is 2.051 [Nm]    maximum power is 2.388
[W]
Trial 93/203 finished    maximum torque is 1.967 [Nm]    maximum power is 1.547
[W]
Trial 94/203 finished    maximum torque is 1.895 [Nm]    maximum power is 1.728
[W]
Trial 95/203 finished    maximum torque is 2.213 [Nm]    maximum power is 3.315
[W]
Trial 96/203 finished    maximum torque is 1.765 [Nm]    maximum power is 1.787
[W]
Trial 97/203 finished    maximum torque is 1.572 [Nm]    maximum power is 1.189
[W]
Trial 98/203 finished    maximum torque is 2.047 [Nm]    maximum power is 2.742
[W]
Trial 99/203 finished    maximum torque is 1.863 [Nm]    maximum power is 2.993
[W]
```

```
Trial 100/203 finished   maximum torque is 2.041 [Nm]    maximum power is 2.836
[W]
Trial 101/203 finished   maximum torque is 2.331 [Nm]    maximum power is 3.646
[W]
Trial 102/203 finished   maximum torque is 1.822 [Nm]    maximum power is 1.137
[W]
Trial 103/203 finished   maximum torque is 2.022 [Nm]    maximum power is 2.148
[W]
Trial 104/203 finished   maximum torque is 1.972 [Nm]    maximum power is 1.944
[W]
Trial 105/203 finished   maximum torque is 2.942 [Nm]    maximum power is 3.463
[W]
Trial 106/203 finished   maximum torque is 2.223 [Nm]    maximum power is 1.528
[W]
Trial 107/203 finished   maximum torque is 2.191 [Nm]    maximum power is 2.091
[W]
Trial 108/203 finished   maximum torque is 2.169 [Nm]    maximum power is 2.189
[W]
Trial 109/203 finished   maximum torque is 1.932 [Nm]    maximum power is 1.626
[W]
Trial 110/203 finished   maximum torque is 1.763 [Nm]    maximum power is 1.116
[W]
Trial 111/203 finished   maximum torque is 1.837 [Nm]    maximum power is 0.803
[W]
Trial 112/203 finished   maximum torque is 1.744 [Nm]    maximum power is 1.151
[W]
Trial 113/203 finished   maximum torque is 1.861 [Nm]    maximum power is 2.469
[W]
Trial 114/203 finished   maximum torque is 2.658 [Nm]    maximum power is 3.810
[W]
Trial 115/203 finished   maximum torque is 1.858 [Nm]    maximum power is 1.957
[W]
Trial 116/203 finished   maximum torque is 2.187 [Nm]    maximum power is 2.410
[W]
Trial 117/203 finished   maximum torque is 1.917 [Nm]    maximum power is 1.363
[W]
Trial 118/203 finished   maximum torque is 1.834 [Nm]    maximum power is 1.439
[W]
Trial 119/203 finished   maximum torque is 1.637 [Nm]    maximum power is 0.672
[W]
Trial 120/203 finished   maximum torque is 2.274 [Nm]    maximum power is 3.163
[W]
Trial 121/203 finished   maximum torque is 2.209 [Nm]    maximum power is 1.725
[W]
Trial 122/203 finished   maximum torque is 2.421 [Nm]    maximum power is 3.402
[W]
Trial 123/203 finished   maximum torque is 1.784 [Nm]    maximum power is 3.929
[W]
```

```
Trial 124/203 finished   maximum torque is 2.082 [Nm]    maximum power is 1.888
[W]
Trial 125/203 finished   maximum torque is 3.691 [Nm]    maximum power is 10.920
[W]
Trial 126/203 finished   maximum torque is 1.976 [Nm]    maximum power is 2.872
[W]
Trial 127/203 finished   maximum torque is 2.707 [Nm]    maximum power is 4.604
[W]
Trial 128/203 finished   maximum torque is 2.103 [Nm]    maximum power is 1.594
[W]
Trial 129/203 finished   maximum torque is 2.238 [Nm]    maximum power is 1.743
[W]
Trial 130/203 finished   maximum torque is 2.155 [Nm]    maximum power is 0.910
[W]
Trial 131/203 finished   maximum torque is 1.895 [Nm]    maximum power is 1.619
[W]
Trial 132/203 finished   maximum torque is 1.766 [Nm]    maximum power is 1.680
[W]
Trial 133/203 finished   maximum torque is 1.435 [Nm]    maximum power is 0.837
[W]
Trial 134/203 finished   maximum torque is 1.886 [Nm]    maximum power is 1.764
[W]
Trial 135/203 finished   maximum torque is 1.689 [Nm]    maximum power is 1.243
[W]
Trial 136/203 finished   maximum torque is 2.039 [Nm]    maximum power is 2.892
[W]
Trial 137/203 finished   maximum torque is 1.969 [Nm]    maximum power is 2.807
[W]
Trial 138/203 finished   maximum torque is 2.024 [Nm]    maximum power is 2.436
[W]
Trial 139/203 finished   maximum torque is 2.062 [Nm]    maximum power is 3.563
[W]
Trial 140/203 finished   maximum torque is 1.750 [Nm]    maximum power is 1.527
[W]
Trial 141/203 finished   maximum torque is 1.509 [Nm]    maximum power is 0.916
[W]
Trial 142/203 finished   maximum torque is 2.147 [Nm]    maximum power is 3.096
[W]
Trial 143/203 finished   maximum torque is 2.105 [Nm]    maximum power is 3.047
[W]
Trial 144/203 finished   maximum torque is 1.832 [Nm]    maximum power is 0.497
[W]
Trial 145/203 finished   maximum torque is 1.894 [Nm]    maximum power is 0.975
[W]
Trial 146/203 finished   maximum torque is 1.854 [Nm]    maximum power is 0.655
[W]
Trial 147/203 finished   maximum torque is 2.585 [Nm]    maximum power is 2.828
[W]
```

```
Trial 148/203 finished   maximum torque is 1.915 [Nm]    maximum power is 1.598
[W]
Trial 149/203 finished   maximum torque is 2.064 [Nm]    maximum power is 1.749
[W]
Trial 150/203 finished   maximum torque is 1.519 [Nm]    maximum power is 0.714
[W]
Trial 151/203 finished   maximum torque is 1.837 [Nm]    maximum power is 1.194
[W]
Trial 152/203 finished   maximum torque is 1.748 [Nm]    maximum power is 2.013
[W]
Trial 153/203 finished   maximum torque is 1.572 [Nm]    maximum power is 1.280
[W]
Trial 154/203 finished   maximum torque is 1.954 [Nm]    maximum power is 1.517
[W]
Trial 155/203 finished   maximum torque is 1.508 [Nm]    maximum power is 1.341
[W]
Trial 156/203 finished   maximum torque is 2.255 [Nm]    maximum power is 2.891
[W]
Trial 157/203 finished   maximum torque is 2.211 [Nm]    maximum power is 2.954
[W]
Trial 158/203 finished   maximum torque is 1.795 [Nm]    maximum power is 0.623
[W]
Trial 159/203 finished   maximum torque is 2.029 [Nm]    maximum power is 0.894
[W]
Trial 160/203 finished   maximum torque is 1.904 [Nm]    maximum power is 1.085
[W]
Trial 161/203 finished   maximum torque is 1.929 [Nm]    maximum power is 1.754
[W]
Trial 162/203 finished   maximum torque is 2.149 [Nm]    maximum power is 2.040
[W]
Trial 163/203 finished   maximum torque is 2.391 [Nm]    maximum power is 3.745
[W]
Trial 164/203 finished   maximum torque is 1.838 [Nm]    maximum power is 2.772
[W]
Trial 165/203 finished   maximum torque is 1.939 [Nm]    maximum power is 2.873
[W]
Trial 166/203 finished   maximum torque is 1.782 [Nm]    maximum power is 1.068
[W]
Trial 167/203 finished   maximum torque is 1.762 [Nm]    maximum power is 1.961
[W]
Trial 168/203 finished   maximum torque is 1.934 [Nm]    maximum power is 1.954
[W]
Trial 169/203 finished   maximum torque is 1.821 [Nm]    maximum power is 2.794
[W]
Trial 170/203 finished   maximum torque is 2.213 [Nm]    maximum power is 3.121
[W]
Trial 171/203 finished   maximum torque is 2.065 [Nm]    maximum power is 2.912
[W]
```

Trial 172/203 finished   maximum torque is 1.623 [Nm]   maximum power is 1.938 [W]
Trial 173/203 finished   maximum torque is 1.539 [Nm]   maximum power is 1.021 [W]
Trial 174/203 finished   maximum torque is 1.684 [Nm]   maximum power is 0.736 [W]
Trial 175/203 finished   maximum torque is 1.990 [Nm]   maximum power is 1.582 [W]
Trial 176/203 finished   maximum torque is 2.001 [Nm]   maximum power is 3.070 [W]
Trial 177/203 finished   maximum torque is 1.943 [Nm]   maximum power is 1.621 [W]
Trial 178/203 finished   maximum torque is 1.602 [Nm]   maximum power is 0.497 [W]
Trial 179/203 finished   maximum torque is 1.916 [Nm]   maximum power is 0.848 [W]
Trial 180/203 finished   maximum torque is 2.081 [Nm]   maximum power is 1.403 [W]
Trial 181/203 finished   maximum torque is 2.197 [Nm]   maximum power is 2.198 [W]
Trial 182/203 finished   maximum torque is 2.069 [Nm]   maximum power is 2.887 [W]
Trial 183/203 finished   maximum torque is 1.959 [Nm]   maximum power is 3.850 [W]
Trial 184/203 finished   maximum torque is 2.478 [Nm]   maximum power is 2.407 [W]
Trial 185/203 finished   maximum torque is 2.325 [Nm]   maximum power is 2.783 [W]
Trial 186/203 finished   maximum torque is 1.818 [Nm]   maximum power is 1.410 [W]
Trial 187/203 finished   maximum torque is 1.918 [Nm]   maximum power is 2.348 [W]
Trial 188/203 finished   maximum torque is 1.702 [Nm]   maximum power is 1.669 [W]
Trial 189/203 finished   maximum torque is 1.904 [Nm]   maximum power is 2.223 [W]
Trial 190/203 finished   maximum torque is 2.137 [Nm]   maximum power is 1.441 [W]
Trial 191/203 finished   maximum torque is 1.854 [Nm]   maximum power is 2.063 [W]
Trial 192/203 finished   maximum torque is 2.015 [Nm]   maximum power is 3.539 [W]
Trial 193/203 finished   maximum torque is 2.479 [Nm]   maximum power is 1.987 [W]
Trial 194/203 finished   maximum torque is 2.281 [Nm]   maximum power is 1.948 [W]
Trial 195/203 finished   maximum torque is 2.028 [Nm]   maximum power is 1.481 [W]

```
Trial 196/203 finished   maximum torque is 1.795 [Nm]    maximum power is 2.051
[W]
Trial 197/203 finished   maximum torque is 1.500 [Nm]    maximum power is 0.869
[W]
Trial 198/203 finished   maximum torque is 1.994 [Nm]    maximum power is 2.683
[W]
Trial 199/203 finished   maximum torque is 1.902 [Nm]    maximum power is 3.710
[W]
Trial 200/203 finished   maximum torque is 1.741 [Nm]    maximum power is 0.904
[W]
Trial 201/203 finished   maximum torque is 1.956 [Nm]    maximum power is 2.327
[W]
Trial 202/203 finished   maximum torque is 2.137 [Nm]    maximum power is 1.852
[W]
Trial 203/203 finished   maximum torque is 1.377 [Nm]    maximum power is 0.993
[W]
```

Calculation summary:

[34]:
```
# print(f"Shoulder max angular velocity:\t{format(max(dtheta1_list), '.3f')}␣
↪[rad/sec]\t\t Shoulder average angular velocity:\t{format(sum(dtheta1_list) /
↪ float(len(dtheta1_list)), '.3f')} [rad/sec]")
# print(f"Shoulder max angular velocity:\t{format(max(dtheta1_list)*60/(2*pi),␣
↪'.3f')} [rpm]\t\t Shoulder average angular velocity:
↪\t{format((sum(dtheta1_list) / float(len(dtheta1_list)))*60/(2*pi), '.3f')}␣
↪[rpm]")
# print(f"Elbow max angular velocity:\t{format(max(dtheta2_list), '.3f')} [rad/
↪sec]\t\t Elbow average angular velocity:\t{format(sum(dtheta2_list) /␣
↪float(len(dtheta2_list)), '.3f')} [rad/sec]")
# print(f"Elbow max angular velocity:\t{format(max(dtheta2_list)*60/(2*pi), '.
↪3f')} [rpm]\t\t Elbow average angular velocity:\t{format((sum(dtheta2_list) /
↪ float(len(dtheta2_list)))*60/(2*pi), '.3f')} [rpm]\n")
# print(f"Shoulder max torque:\t\t{format(max(tau1_list), '.3f')} [Nm]\t\t␣
↪Shoulder average torque:\t\t{format(sum(tau1_list) / float(len(tau1_list)),␣
↪'.3f')} [Nm]")
# print(f"Elbow max torque:\t\t{format(max(tau2_list), '.3f')} [Nm]\t\t Elbow␣
↪average torque:\t\t\t{format(sum(tau2_list) / float(len(tau2_list)), '.3f')}␣
↪[Nm]\n")
# print(f"Shoulder max power:\t\t{format(max(power1_list), '.3f')} [W]\t\t␣
↪Shoulder average power:\t\t{format(sum(power1_list) /␣
↪float(len(power1_list)), '.3f')} [W]")
# print(f"Elbow max power:\t\t{format(max(power2_list), '.3f')} [W]\t\t Elbow␣
↪average power:\t\t\t{format(sum(power2_list) / float(len(power2_list)), '.
↪3f')} [W]\n")


max_Elbow_tau, max_Elbow_power, max_Elbow_Vel = 0, 0, 0
max_Elbow_tau_index, max_Elbow_power_index, max_Elbow_Vel_index = 0, 0, 0
```

```python
for i in range(len(Elbow_tau_list)):
    if max_Elbow_Vel < max(Elbow_Vel_list[i]):
        max_Elbow_Vel = max(Elbow_Vel_list[i])
        max_Elbow_Vel_index = i

    if max_Elbow_tau < max(Elbow_tau_list[i]):
        max_Elbow_tau = max(Elbow_tau_list[i])
        max_Elbow_tau_index = i

    if max_Elbow_power < max(Elbow_power_list[i]):
        max_Elbow_power = max(Elbow_power_list[i])
        max_Elbow_power_index = i

print(f"maximum elbow angular velocity is {format(max_Elbow_Vel, '.3f')} [rad/sec] ({format(max_Elbow_Vel*60/(2*pi), '.3f')} [rpm]), in trial {max_Elbow_Vel_index}")
print(f"maximum elbow torque is {format(max_Elbow_tau, '.3f')} [Nm], in trial {max_Elbow_tau_index}")
print(f"maximum elbow power is {format(max_Elbow_power, '.3f')} [W], in trial {max_Elbow_power_index}")


# The torque equations for the maximum power:
solution_0_subs = solution_0_subs.subs([(m1, m_upper_arm_dict[participants_list[max_Elbow_tau_index]]), (m2, m_lower_arm), (R1, L_upper_arm_dict[participants_list[max_Elbow_tau_index]]), (R2, L_lower_arm), (R1_COM, L_upper_arm_COM_dict[participants_list[max_Elbow_tau_index]]), (R2_COM, L_lower_arm_COM), (g, 9.81)])
solution_1_subs = solution_1_subs.subs([(m1, m_upper_arm_dict[participants_list[max_Elbow_tau_index]]), (m2, m_lower_arm), (R1, L_upper_arm_dict[participants_list[max_Elbow_tau_index]]), (R2, L_lower_arm), (R1_COM, L_upper_arm_COM_dict[participants_list[max_Elbow_tau_index]]), (R2_COM, L_lower_arm_COM), (g, 9.81)])

print("\nThe torque equations for the maximum torque:")
display(Eq(T[0], solution_0_subs))
display(Eq(T[1], solution_1_subs))

display(Elbow_Ang_list[max_Elbow_tau_index])
display(Elbow_Vel_list[max_Elbow_tau_index])
display(Elbow_Acc_list[max_Elbow_tau_index])
display(Elbow_tau_list[max_Elbow_tau_index])
```

```
maximum elbow angular velocity is 4.199 [rad/sec] (40.100 [rpm]), in trial 125
maximum elbow torque is 3.857 [Nm], in trial 7
maximum elbow power is 10.920 [W], in trial 125

The torque equations for the maximum torque:
```

$$\tau_1(t) = 0.1110202357768\ddot{\theta}_1 \cos(\theta_2(t)) + 0.1520553387246 74\ddot{\theta}_1 + 0.0555101178884\ddot{\theta}_2 \cos(\theta_2(t)) + 0.042170312288\ddot{\theta}_2 - 0.1110202357768\dot{\theta}_1\dot{\theta}_2 \sin(\theta_2(t)) - 0.0555101178884\dot{\theta}_2^2 \sin(\theta_2(t)) + 1.732373406 \sin(\theta_1(t) + \theta_2(t)) + 4.915563608124 \sin(\theta_1(t))$$

$$\tau_2(t) = 0.0555101178884\ddot{\theta}_1 \cos(\theta_2(t)) + 0.042170312288\ddot{\theta}_1 + 0.042170312288\ddot{\theta}_2 + 0.0555101178884\dot{\theta}_1^2 \sin(\theta_2(t)) + 1.732373406 \sin(\theta_1(t) + \theta_2(t))$$

```
[0.6678676915681501,
 0.6644293929417213,
 0.6608689212676528,
 0.6571862765459449,
 0.6533290988990375,
 0.6493497482044903,
 0.6452482244623037,
 0.6410768875500372,
 0.6369404572228107,
 0.6330832795759032,
 0.6297497007045939,
 0.6271666134116423,
 0.6254736440372078,
 0.6246882458738104,
 0.6247580590438901,
 0.6255260039147676,
 0.6267128278061239,
 0.6279869181600797,
 0.6290515690037962,
 0.6295926210719145,
 0.6292784618065554,
 0.6279345582825198,
 0.6254910973297279,
 0.6220004388257392,
 0.6176720222807932,
 0.6126280207425296,
 0.6071127803062275,
 0.6013357404821263,
 0.5955587006580251,
 0.5901132733918028,
 0.5852263514862186,
 0.5811597343290718,
 0.5779832350904421,
 0.5756968537703296,
 0.5742656837836941,
 0.5736199119604564,
```

```
0.5736199119604564,
0.5741609640285746,
0.5750510819470918,
0.5762029992534079,
0.5775294494849237,
0.5790129793491188,
0.5806884954310334,
0.5826432641932671,
0.58482492575826,
0.587268386711052,
0.590008553636683,
0.593115239705233,
0.5966408047942615,
0.6006027021962886,
0.6049660253262744,
0.6096784143066591,
0.614687509259883,
0.6199234970158659,
0.6253165644045283,
0.6307445383782307,
0.6361026991818535,
0.6413037803527964,
0.6462256088434204,
0.6508158247761655,
0.654987161688432,
0.6587396195802198,
0.662073198451529,
0.6649529917173197,
0.6674139059626315,
0.6695432076500646,
0.6714456165347384,
0.6732782122493326,
0.6751806211340065,
0.6773622826989992,
0.6800500897470705,
0.6833487620333398,
0.6872582995578072,
0.6916216226877929,
0.696176932035498,
0.7005751617505238,
0.7044148861049113,
0.7073644925407818,
0.7090923685002563,
0.7092494481329357,
0.7076088386360609,
0.7040832735470325,
0.6987425660359298,
0.6917787023204723,
```

```
0.6835058416660192,
0.6742905032154893,
0.664569019281881,
0.6546031892529932,
0.6446548125166255,
0.6348460621204173,
0.6252118446494087,
0.6157172535185595,
0.6064495551904697,
0.5975134694202586,
0.5890835291331261,
0.5813691738393112,
0.5745449364640133,
0.5687504433473921,
0.5640904142445673,
0.5606521156181384,
0.5584530007606255,
0.5574756163795088,
0.5576676025972281,
0.558959146243704,
0.5611408078086968,
0.5640206010744875,
0.567528712870996,
0.5715953300281429,
0.5762029992534079,
0.5812644540841915,
0.5867273346429337,
0.5924345612969552,
0.598333774168696,
0.6042853469179967,
0.6102543729598173,
0.616171039124078,
0.622052798703299,
0.6278123852348803,
0.6333625322562223,
0.6385461601346454,
0.6430665628973107,
0.6467317543264988,
0.6493497482044903,
0.6508856379462453,
0.6513568768442838,
0.6508681846537253,
0.6495766410072495,
0.6477265920001356,
0.6455274771426227,
0.6431538293599104,
0.6407103684071185,
0.6383367206244059,
```

```
0.6360852458893334,
0.6341304771270997,
0.6326120406778647,
0.6316870161743078,
0.6314775766640683,
0.6319662688546268,
0.6330309196983432,
0.6344097298074189,
0.6359107129641339,
0.6372371631956497,
0.6380574679440869,
0.6380923745291269,
0.6372022566106097,
0.6353173010184558,
0.6324200544601453,
0.6285279702281978,
0.6236759549076538,
0.6179861815461523,
0.6115808231913329,
0.6046693193534356,
0.5975134694202587,
0.590340166194562,
0.5834461156491844,
0.5771280237569649,
0.5716127833206629,
0.5670225673879178,
0.5634097358362895,
0.5606870222031783,
0.5586973468559049,
0.5574407097944688,
0.5568822044338307,
0.5570567373590302,
0.5580690283251869,
0.5600063437949006,
0.5628861370606911,
0.5666036883674391,
0.5710891845450645,
0.576185545960888,
0.5818753193223896,
0.5882457710921688,
0.595261994685186,
0.6028192703463214,
0.6107256118578558,
0.6188937527571893,
0.627306239751802,
0.6359805261342137,
0.6448991586119048,
0.6539225108447154,
```

0.6628411433224064,
0.6713234434870988,
0.6792472382911532,
0.6864379948093698,
0.6930178860893885,
0.6990916318863286,
0.7048163118328702,
0.710366458854212,
0.7158293394129543,
0.7212573133866567,
0.7265980208977594,
0.7318340086537423,
0.736808197021926,
0.7414333195397111,
0.7455348432818978,
0.7491302215410062,
0.7520972812693963,
0.754366209296989,
0.7557101128210247,
0.7558846457462242,
0.7545756488072284,
0.7514864160311985,
0.7463202414452953,
0.7389025921243194,
0.7292683746533106,
0.7175397620799087,
0.7040483669619925,
0.689160708442481,
0.6732782122493326,
0.6568372106955459,
0.6402740360941197,
0.6240075674655325,
0.6083170574901036,
0.5934817588481517,
0.5797634709274763,
0.5673716332383166,
0.5565854984609917,
0.5474923330581012,
0.540179403492245,
0.5346118031783831,
0.5308069854090355,
0.5286602304290824,
0.528154084946004,
0.5292710956672804,
0.5320985290555112,
0.5366189318181767,
0.5427624907851967,
0.5503721263238919,

0.559238398924023,
0.5690820559052712,
0.5796063912947969,
0.5905321524122813,
0.6015277266998457,
0.6124360345248102,
0.6231698094245752,
0.6335719717664616,
0.643590161672909,
0.653119659388798,
0.6619859319889292,
0.6700493531331431,
0.6772226563588397,
0.6834360284959395,
0.6887069228369624,
0.6930527926744283,
0.6963863715457375,
0.6986378462808102,
0.6997024971245267,
0.6995454174918472,
0.6981840606752916,
0.6957580530150196,
0.6923721142661505,
0.6883054971090037,
0.6837501877612985,
0.6790377987809137,
0.6744999427257286,
0.6704507788611017,
0.6670822934047527,
0.664516659404321,
0.6627364235672868,
0.6617939457712099,
0.6615845062609704,
0.6619335721113694,
0.6626142505196473,
0.6633647420980048,
0.6639407007511628,
0.6641675935539221,
0.6638185277035233,
0.662701516982247,
0.6606943883424534,
0.6576575154439834,
0.6535036318242368,
0.6482676440682538,
0.6420019120535942,
0.6348635154129374,
0.6270968002415627,
0.618981019219789,

```
0.6109001447830553,
0.6030985230266406,
0.5957855934607843,
0.5890835291331261,
0.5831145030913055,
0.5778959686278424,
0.5735151922053368,
0.5698849073611885,
0.5669353009253181,
0.5645616531426059,
0.5626592442579319,
0.5611757143937368,
0.5600412503799405,
0.5593082120941029,
0.5589765995362239,
0.5591336791689033,
0.559866717454741,
0.5611931676862567,
0.5631653897410103,
0.5658182902040417,
0.5691693223678708,
0.5732359395250176,
0.5779657817979222,
0.5833763024791047,
0.5894675015685648,
0.5963266455289025,
0.6040410008227175,
0.6125582075724498,
0.6218957190706194,
0.6319488155621068,
0.6425604174142323,
0.6535559917017966,
0.6646911923295203,
0.6758438462497642,
0.6868568738298484,
0.6975208355595337,
0.7077484649762207,
0.7174001357397491,
0.7265107544351597,
0.7350803210624518,
0.7431262889141458,
0.750701017867801,
0.7578045079234179,
0.7644891189585562,
0.770737397680696,
0.7765144375047972,
0.7817329719682602,
0.7862359214384056,
```

```
 0.789796393112474,
 0.7922224007727462,
 0.7932521450314226,
 0.7926761863782646,
 0.7902676320105125,
 0.7859043088805266,
 0.7795513104032672,
 0.7712609964562942,
 0.7612777131348867,
 0.7498632598268437,
 0.7372794359199646,
 0.723910213849688,
 0.7099475798337334,
 0.6957056931374597,
 0.6814463531486661,
 0.6673615460850717,
 0.6535908982868366,
 0.6402216762165599,
 0.6273411463368418,
 0.614984215232722,
 0.6031857894892403,
 0.5920505888615164,
 0.58170078639719,
 0.5724330880691002,
 0.564439480094966,
 0.5580515750326669,
 0.5534962656849617,
 0.5509306316845299,
 0.5502499532762521,
 0.5511226179022494,
 0.5529726669093634,
 0.5551717817668762,
 0.5572836301617894,
 0.5588544264885843,
 0.5596747312370216,
 0.5599714372098608]

[-0.4125958351714498,
 -0.42725660088821726,
 -0.44191736660495795,
 -0.4628613176288888,
 -0.47752208334565627,
 -0.4921828490623967,
 -0.5005604294719747,
 -0.49637163926718586,
 -0.46286131762890215,
 -0.4000294645571102,
 -0.3099704751541977,
```

```
-0.20315632493212968,
-0.09424777960768796,
0.008377580409564269,
0.09215338450530021,
0.14241886696275244,
0.15289084247469112,
0.127758101245985,
0.06492624817419303,
-0.03769911184308584,
-0.16126842288428203,
-0.2932153143350292,
-0.4188790204786384,
-0.5194099853935176,
-0.6052801845916279,
-0.6618288523562567,
-0.6932447788921393,
-0.6932447788921526,
-0.6534512719466791,
-0.5864306286700983,
-0.48799405885762,
-0.3811799086355552,
-0.2743657584134987,
-0.17174039839626126,
-0.07749261878853252,
0.0,
0.0649262481741928,
0.10681415022205469,
0.13823007675793655,
0.15917402778189482,
0.17802358370340998,
0.2010619298297563,
0.23457225146803856,
0.2617993877991474,
0.29321531433504155,
0.3288200310757262,
0.37280232822600423,
0.4230678106834138,
0.47542768824325676,
0.5235987755982913,
0.5654866776461696,
0.6010913943868638,
0.6283185307179484,
0.6471680866394871,
0.6513568768442936,
0.642979296434725,
0.6241297405131596,
0.5906194188748718,
0.5508259119294154,
```

0.500560429471973,
0.45029494701453726,
0.4000294645571062,
0.3455751918948836,
0.2953097094374179,
0.2555162024919708,
0.22828906616086053,
0.21991148575129926,
0.22828906616086356,
0.2617993877991341,
0.3225368457685475,
0.3958406743523227,
0.46914450293608023,
0.5235987755982948,
0.546637121724606,
0.5277875658030969,
0.4607669225265026,
0.35395277230445654,
0.20734511513693285,
0.01884955592152966,
-0.19687313962496747,
-0.42306781068341104,
-0.6408849013323282,
-0.8356636458548946,
-0.992743278534375,
-1.1058406140635781,
-1.1665780720330108,
-1.1958996034665323,
-1.1938052083641313,
-1.1770500475449737,
-1.156106096521032,
-1.1393509357019034,
-1.112123799370768,
-1.0723302924253357,
-1.0115928344559046,
-0.9257226352577941,
-0.8189084850357418,
-0.6953391739945427,
-0.5592034923389797,
-0.4125958351714646,
-0.26389378290154497,
-0.11728612573400674,
0.02303834612631847,
0.15498523757710603,
0.2617993877991306,
0.3455751918948836,
0.4209734155810289,
0.4879940588576167,

```
0.5529203070318032,
0.607374579694031,
0.655545667049069,
0.6848671984825683,
0.7079055446089092,
0.7141887299160857,
0.7162831250184735,
0.7099999397112742,
0.7058111495065215,
0.6911503837897539,
0.6660176425610477,
0.6220353454107636,
0.5424483315198377,
0.4398229715025719,
0.3141592653589875,
0.18430676901060106,
0.05654866776461412,
-0.05864306286701669,
-0.1549852375770927,
-0.22200588085367368,
-0.26389378290154847,
-0.2848377339254792,
-0.2932153143350302,
-0.2848377339255058,
-0.2701769682087045,
-0.23457225146804012,
-0.18221237390819997,
-0.11100294042683018,
-0.02513274122873286,
0.05864306286701669,
0.1277581012459721,
0.1654572130890803,
0.1801179788057989,
0.15917402778189482,
0.09843656981246375,
0.004188790204802132,
-0.10681415022206801,
-0.2261946710584625,
-0.34766958699727135,
-0.46705010783368006,
-0.5822418384652849,
-0.6827728033801896,
-0.768643002578327,
-0.8293804605476781,
-0.8587019919812263,
-0.8607963870836142,
-0.8272860654452949,
-0.7581710270663349,
```

```
-0.6618288523562456,
-0.5508259119294154,
-0.43353978619539535,
-0.32672563597334064,
-0.23876104167281562,
-0.15079644737233053,
-0.06702064327656587,
0.020943951023930724,
0.12147491593880887,
0.23247785636563906,
0.3455751918948703,
0.4461061568097618,
0.5382595413150489,
0.6115633698988036,
0.6827728033801896,
0.7644542123735114,
0.8419468311620711,
0.906873079336251,
0.9487609813841258,
0.9801769079200219,
1.0094984393535302,
1.0409143658893854,
1.0702358973229347,
1.0828022679372744,
1.0702358973229213,
1.0178760197630945,
0.9508553764865268,
0.8628907821860019,
0.7895869536022233,
0.7288494956328132,
0.6869615935849918,
0.666017642561021,
0.6555456670490823,
0.6513568768442802,
0.6408849013323282,
0.6283185307179484,
0.5969026041820364,
0.5550147021342176,
0.4921828490623987,
0.4314453910930076,
0.35604716740682235,
0.27227136331112606,
0.1612684228842826,
0.02094395102393017,
-0.15707963267949376,
-0.37070793312358985,
-0.6199409503083841,
-0.8901179185171091,
```

```
-1.1561060965210586,
-1.4074335088082408,
-1.618967414149949,
-1.7865190223413339,
-1.9058995431778158,
-1.9729201864544104,
-1.9875809521711512,
-1.9519762354304662,
-1.8828611970514841,
-1.7802358370342315,
-1.6461945504810125,
-1.4870205226991615,
-1.2943361732789962,
-1.0911798483468653,
-0.8775515479027427,
-0.6681120376634354,
-0.4565781323217138,
-0.2576105975943719,
-0.06073745796940282,
0.13404128655316197,
0.3392920065877071,
0.5424483315198511,
0.7372270760424042,
0.9131562646434276,
1.0639527120157448,
1.1812388377497467,
1.2629202467430867,
1.311091334098138,
1.319468914507729,
1.3089969389957503,
1.2880529879718063,
1.2482594810263725,
1.2021827887736822,
1.1435397259066618,
1.0639527120157448,
0.9676105373056688,
0.8607963870836008,
0.7456046564519818,
0.6325073209227505,
0.5215043804959071,
0.4000294645570982,
0.270176968208725,
0.1277581012459854,
-0.01884955592154298,
-0.1633628179866616,
-0.29112091923264244,
-0.40631264986428806,
-0.4879940588576232,
```

-0.5466371217246265,
-0.5654866776461696,
-0.5445427266222255,
-0.48589966375522214,
-0.404218254761887,
-0.3078760800517977,
-0.21362830044410938,
-0.11309733552923125,
-0.02513274122873286,
0.04188790204787477,
0.08168140899334848,
0.09005898940289465,
0.06911503837896874,
0.02722713633110728,
-0.041887902047848126,
-0.13404128655316197,
-0.24085543677523,
-0.36442474781639994,
-0.4984660343696019,
-0.6283185307179617,
-0.751887841759145,
-0.8566075968788253,
-0.9320058205649706,
-0.9738937226128453,
-0.9697049324080432,
-0.9361946107697727,
-0.8775515479027093,
-0.8042477193189852,
-0.7162831250184735,
-0.626224135615574,
-0.5256931707006826,
-0.4356341812977964,
-0.35395277230444794,
-0.28483773392546585,
-0.22828906616087688,
-0.17802358370341118,
-0.13613568165556303,
-0.08796459430051172,
-0.039793506945473706,
0.01884955592152966,
0.08796459430052504,
0.15917402778187303,
0.23666664657042785,
0.3183480555637763,
0.40212385965948594,
0.4879940588576232,
0.5675810727485573,
0.6492624817418925,

```
0.7309438907352143,
0.8230972752405281,
0.9257226352578074,
1.0220648099678833,
1.1205013797803471,
1.2063715789784977,
1.2733922222550653,
1.3194689145077156,
1.3362240753267864,
1.3383184704292719,
1.3215633096101034,
1.2796754075622419,
1.2273155300024419,
1.1582004916234199,
1.0932742434492664,
1.0283479952750598,
0.9655161422032811,
0.9089674744386255,
0.8524188066740365,
0.8021533242165975,
0.7497934466567839,
0.6932447788921416,
0.6262241356155607,
0.5403539364174212,
0.42725660088820544,
0.2911209192326691,
0.12356931104116997,
-0.06911503837895541,
-0.28902652413025465,
-0.5235987755983081,
-0.762359817271137,
-0.9948376736367628,
-1.1979939985689068,
-1.369734396965168,
-1.5100588688254932,
-1.604306648433195,
-1.6755160819145647,
-1.7090264035528484,
-1.711120798655145,
-1.6901768476313321,
-1.6524777357882194,
-1.6043066484332082,
-1.5456635855661782,
-1.482831732494386,
-1.415811089217805,
-1.3362240753268708,
-1.2419762957191693,
-1.1121237993707829,
```

```
 -0.9592329568961045,
 -0.7665486074758993,
 -0.5466371217246265,
 -0.307876080051811,
 -0.08168140899333516,
 0.10471975511966138,
 0.222005880853687,
 0.26389378290153515,
 0.2534218073895831,
 0.18849555921538985,
 0.09843656981247707,
 0.03560471674069822,
 1.0807078728348867]

[-1.7592918860120932,
 -1.7592918860088824,
 -2.5132741228717044,
 -1.7592918860120936,
 -1.7592918860088496,
 -1.0053096491493567,
 0.5026548245746555,
 4.0212385965940465,
 7.539822368615036,
 10.807078728349504,
 12.817698026648163,
 13.069025438932988,
 12.31504320207027,
 10.053096491488315,
 6.0318578948942685,
 1.2566370614326419,
 -3.0159289474447357,
 -7.539822368615038,
 -12.315043202073467,
 -14.828317324943548,
 -15.833626974089661,
 -15.07964473723311,
 -12.063715789785464,
 -10.304423903773243,
 -6.785840131755451,
 -3.7699111843059194,
 -1.5987211554602258e-12,
 4.775220833456829,
 8.042477193189692,
 11.812388377497403,
 12.817698026647736,
 12.817698026646823,
 12.315043202068454,
 11.30973355292749,
```

9.299114254623873,
7.791149780903164,
5.0265482457434105,
3.769911184305837,
2.5132741228749844,
2.261946710581827,
2.7646015351615487,
4.021238596593887,
3.2672563597330515,
3.7699111843073108,
4.272566008882146,
5.277875658033346,
6.031857894889171,
6.283185307181134,
5.780530482604161,
5.026548245745379,
4.27256600888332,
3.267256359730147,
2.2619467105846516,
0.5026548245767721,
-1.0053096491482352,
-2.261946710587834,
-4.021238596594553,
-4.775220833454748,
-6.031857894893115,
-6.031857894892268,
-6.03185789489175,
-6.534512719466646,
-6.031857894895906,
-4.775220833453668,
-3.2672563597332465,
-1.0053096491473426,
1.005309649147719,
4.021238596592478,
7.288494956329638,
8.796459430052932,
8.796459430050938,
6.5345127194657735,
2.7646015351573503,
-2.2619467105810642,
-8.04247719319135,
-12.817698026645571,
-17.59291886010267,
-22.619467105848464,
-25.886723465579745,
-27.143360527013325,
-26.13805087786981,
-23.37344934270805,

```
-18.84955592153772,
-13.571680263504419,
-7.288494956331846,
-3.5185837720226,
0.25132741228812794,
2.0106192982989217,
2.513274122872986,
2.010619298295431,
3.2672563597362574,
4.775220833451825,
7.288494956331756,
10.304423903773305,
12.817698026646324,
14.828317324943741,
16.336281798667617,
17.592918860101875,
17.844246272390418,
17.592918860104415,
16.838936623239086,
15.833626974094564,
12.817698026642994,
10.053096491490265,
9.047786842337468,
8.042477193190564,
7.791149780902299,
6.534512719467359,
5.780530482604587,
3.5185837720199222,
2.764601535160884,
0.7539822368611864,
0.2513274122865292,
-0.7539822368639175,
-0.5026548245703206,
-1.7592918860121127,
-3.015928947444759,
-5.277875658034113,
-9.550441666911004,
-12.315043202071946,
-15.079644737230176,
-15.582299561806431,
-15.330972149518487,
-13.823007675795377,
-11.561060965209164,
-8.042477193189745,
-5.026548245744993,
-2.5132741228716937,
-1.0053096491461235,
1.005309649142926,
```

1.7592918860161628,
4.272566008879629,
6.2831853071808395,
8.545132017764406,
10.304423903771715,
10.053096491489983,
8.293804605474678,
4.523893421173001,
1.759291886006194,
-2.5132741228685,
-7.288494956331754,
-11.309733552919434,
-13.320352851224465,
-14.32566250036739,
-14.576989912657114,
-14.325662500369097,
-13.823007675792258,
-12.063715789788615,
-10.304423903776515,
-7.288494956322164,
-3.5185837720257975,
-0.25132741228654254,
4.021238596598323,
8.293804605475014,
11.56106096521076,
13.320352851219662,
14.07433508808246,
12.81769802664661,
10.555751316063041,
10.555751316058249,
10.053096491491795,
10.555751316059347,
12.063715789785421,
13.32035285121967,
13.571680263507796,
12.063715789787022,
11.058406140634498,
8.796459430050586,
8.545132017766132,
9.801769079198648,
9.2991142546272,
7.791149780901617,
5.026548245744986,
3.769911184307544,
3.5185837720210147,
3.769911184302628,
3.518583772025837,
1.507964473720774,

-1.5079644737223727,
-6.283185307179244,
-8.042477193188146,
-10.55575131606303,
-8.796459430053464,
-7.2884949563290435,
-5.026548245738591,
-2.5132741228764965,
-1.256637061432646,
-0.5026548245762559,
-1.2566370614342448,
-1.5079644737255835,
-3.769911184309449,
-5.026548245738151,
-7.539822368618292,
-7.288494956326954,
-9.047786842342264,
-10.053096491483592,
-13.320352851221264,
-16.838936623242347,
-21.36283004441038,
-25.635396053291622,
-29.90796206217542,
-32.42123618504711,
-31.91858136047405,
-30.15928947446198,
-25.38406864100508,
-20.10619298296625,
-14.325662500377508,
-8.04247719319137,
-1.759291886008902,
4.272566008882214,
8.293804605477874,
12.315043202070362,
16.08495438638634,
19.100883333821674,
23.122121930419922,
24.37875899185579,
25.635396053294805,
25.132741228716963,
25.384068641006685,
23.876104167281113,
23.62477675499637,
23.373449342707232,
24.6300864041455,
24.378758991857364,
23.373449342706465,
21.11150263212288,

18.095573684678133,
14.074335088080275,
9.801769079200572,
5.780530482606186,
1.005309649150913,
-1.2566370614374422,
-2.5132741228732858,
-4.775220833452075,
-5.529203070322854,
-7.037167544042482,
-9.550441666909812,
-11.561060965209162,
-12.817698026648202,
-13.823007675794333,
-13.571680263507803,
-13.32035285122126,
-14.576989912657117,
-15.58229956180484,
-17.09026403552881,
-17.592918860103467,
-17.3415914478143,
-15.330972149516937,
-13.823007675797523,
-9.801769079200252,
-7.037167544040424,
-2.261946710585171,
2.513274122873299,
7.037167544040424,
9.801769079200252,
11.56106096521076,
11.309733552922635,
12.06371578978542,
10.555751316059844,
8.042477193192944,
4.775220833456863,
1.005309649145544,
-2.5132741228709854,
-5.026548245743393,
-8.293804605474678,
-11.058406140637702,
-12.817698026648209,
-14.828317324940446,
-16.084954386384293,
-15.582299561803234,
-14.828317324942049,
-12.566370614361674,
-9.047786842337475,
-5.026548245744986,

0.5026548245762559,
4.021238596592474,
7.037167544047638,
8.796459430046452,
10.555751316061446,
10.807078728347975,
12.063715789787016,
10.807078728346376,
9.80176907920185,
8.29380460547788,
6.7858401317507,
6.031857894895906,
5.026548245741796,
5.780530482606178,
5.780530482604582,
7.037167544040429,
8.293804605479474,
8.545132017761789,
9.299114254626115,
9.80176907920185,
10.05309649148519,
10.304423903776508,
9.550441666912125,
9.801769079200259,
9.801769079198648,
11.058406140637702,
12.315043202073559,
11.561060965209148,
11.812388377495704,
10.304423903778101,
8.042477193188146,
5.529203070318058,
2.0106192982885034,
0.2513274122982398,
-2.0106192983002273,
-5.026548245743401,
-6.28318530717602,
-8.29380460548267,
-7.791149780898446,
-7.7911497809048145,
-7.539822368613476,
-6.785840131758698,
-6.785840131750704,
-6.031857894892702,
-6.283185307177646,
-6.785840131757099,
-8.042477193189745,
-10.304423903776769,

```
 -13.571680263505222,
 -16.33628179866442,
 -20.106192982979966,
 -23.12212193041513,
 -26.389378290156003,
 -28.14867017616652,
 -28.651325000739565,
 -27.897342763875194,
 -24.378758991857364,
 -20.60884780755143,
 -16.83893662323908,
 -11.309733552924245,
 -8.545132017764402,
 -4.02123859659406,
 -0.2513274122756046,
 2.5132741228574313,
 4.52389342117354,
 5.780530482601363,
 7.037167544043628,
 7.539822368615088,
 8.042477193189745,
 9.550441666912144,
 11.309733552924218,
 15.582299561806431,
 18.346901096961464,
 23.12212193042472,
 26.38937829015282,
 28.651325000737966,
 27.143360527017197,
 22.368139693559662,
 14.074335088082375,
 5.026548245741796,
 -1.2566370614342515,
 -7.791149780903216,
 -10.807078728349572,
 -7.539822368613489,
 125.41237873130305,
 12.31504320207196]

[0.678379348168032,
 0.773081246601525,
 0.836060770874859,
 0.876360537706190,
 0.950325945711220,
 1.03500333508618,
 1.15228977861353,
 1.18118372212757,
 1.27649484374427,
```

1.23138158470890,
1.20014998392148,
1.14002810698724,
1.03920800386012,
0.986056578121280,
0.926015580376722,
0.966936598812291,
1.00809877644574,
1.10526002549516,
1.21486699806614,
1.33418784451178,
1.38834597542906,
1.40999356954432,
1.39791466870564,
1.42168529307569,
1.38922497303320,
1.37933956608757,
1.35691988360842,
1.28747149321530,
1.24073923562878,
1.12526371325116,
1.06892713481085,
0.992671549014616,
0.895762750746597,
0.778180759409375,
0.707895803233052,
0.615209206074249,
0.603849561061220,
0.590295716973236,
0.599430480406486,
0.695931990598818,
0.780190706419686,
0.896554230453080,
0.928535944611594,
0.991850114372735,
1.01111251439866,
1.05180088358690,
1.10433178391136,
1.09177570943287,
1.09199278474861,
0.971541236813495,
0.895150675253635,
0.830114013211203,
0.742941530919544,
0.711428303659332,
0.690270276675316,
0.635556066377179,
0.580777728861215,

0.524066875496445,
0.401621349087760,
0.330885746802219,
0.258688963284467,
0.185442651963706,
0.196366330562134,
0.108264756289215,
0.115402213807172,
0.0454387134410992,
-0.0362723207357470,
-0.119293347517722,
-0.256345946903213,
-0.380905444443602,
-0.524342819433250,
-0.567874080893862,
-0.587659942901666,
-0.489068305895698,
-0.403003507407829,
-0.235642894744557,
-0.0735227436748675,
0.114559709395308,
0.285683700013028,
0.472846058022298,
0.687841033625293,
0.846943505237431,
0.951139786540424,
1.02225134224552,
1.04924327911279,
1.03370524401843,
1.01727558420190,
1.11023709790551,
1.15001295215897,
1.25659892409227,
1.28438952532834,
1.34502141365898,
1.36024539058173,
1.35288541166152,
1.39155429677054,
1.36548081473624,
1.36406900596035,
1.26410593133087,
1.21308153685039,
1.12023728127230,
1.09785800110123,
1.04403265917099,
0.974357603359933,
0.916404376268982,
0.888589213138297,

```
0.906341527792658,
0.822044642089380,
0.808383587351454,
0.727058619077738,
0.738991121169984,
0.702832655050030,
0.724874597220158,
0.676845175007691,
0.696299805828709,
0.635905813439487,
0.644541196620346,
0.586625118550560,
0.573773924371290,
0.562349056805915,
0.568775441889936,
0.682257175806635,
0.758175651836063,
0.843169409206759,
0.937288318642375,
0.998244801653037,
1.06944357891223,
1.12028783341307,
1.12913890060893,
1.11833421808543,
1.10795684945467,
1.06731716367215,
0.992824703428741,
0.919727539914505,
0.836168550996255,
0.841463777090657,
0.806653257058934,
0.884258421161760,
0.899878156017497,
0.958730238054821,
1.02015007646539,
1.03918859563369,
1.06905785066199,
1.16303247230175,
1.27911627149590,
1.36362372804734,
1.40620775783121,
1.39612153477523,
1.35328617011605,
1.24534301704651,
1.21176564310266,
1.20898086438906,
1.20572424250228,
1.22154260594848,
```

1.16930383932772,
1.13921289142618,
1.08906005322706,
1.06331969636895,
1.06538357754211,
1.02705120646430,
0.951451467096996,
0.877412812147026,
0.801374619020438,
0.790212465577657,
0.813572771232545,
0.840110444067223,
0.793554172547767,
0.836484337692538,
0.849932244173852,
0.815211413967077,
0.844498697661295,
0.866719410727867,
0.891542485021477,
0.908216046174386,
0.944674526808952,
0.957547835686279,
0.969672835250341,
0.916801632652853,
0.833409432986102,
0.707520286909748,
0.593791686266713,
0.520059321482410,
0.456487813341793,
0.421603833407802,
0.417496825987241,
0.379821183277633,
0.351513504347688,
0.290419330521018,
0.165062809624902,
0.0390938307818332,
-0.0870123054551354,
-0.276986091048087,
-0.403574225418159,
-0.520518843868416,
-0.604926514847110,
-0.649232361903395,
-0.600174355988023,
-0.585772329078214,
-0.459780980060425,
-0.359422531599069,
-0.211540536364015,
-0.0479427108102170,

0.143435191186412,
0.333100418351875,
0.531333089240539,
0.686982325056326,
0.819936138833361,
0.887392540601213,
0.973717478273837,
1.01465309551850,
1.08572120627082,
1.09192144005324,
1.14080948670313,
1.15665962772672,
1.25152610714108,
1.34532551537648,
1.39095935405877,
1.41620748326722,
1.43702869747458,
1.40180056491218,
1.36270359097494,
1.31111650805234,
1.23941580255982,
1.14340321282346,
1.10036890233740,
1.05096464107030,
0.996383905743551,
0.957159470575982,
0.944209560503370,
0.875647346146592,
0.884628850359297,
0.860124137199294,
0.872907755431500,
0.882400232918438,
0.870621877386327,
0.833449209369868,
0.786356830419898,
0.718258391739966,
0.648884667221192,
0.589089685603427,
0.539450553157230,
0.520784747135530,
0.544575476887497,
0.503911496948233,
0.494428600042252,
0.441480568229295,
0.494034224869801,
0.513410043562020,
0.627689591200948,
0.741564287660421,

0.854746285591128,
0.979208374460718,
1.12488027427098,
1.18563775805664,
1.27953479234257,
1.27884198894586,
1.23708735526337,
1.19825149782988,
1.13877397826784,
1.09346622243545,
1.00744254147708,
0.977581354630757,
0.906216807981064,
0.932954413437470,
0.959848664138869,
1.02070961981967,
1.08249022698587,
1.18833755703521,
1.24257048848176,
1.28794760410188,
1.32366287928106,
1.35047027999231,
1.33415323025109,
1.30582953209609,
1.27574413795185,
1.26346940338264,
1.18534141227021,
1.10532475886295,
1.08040658252270,
0.966355787272737,
0.942440481803591,
0.872560006265012,
0.829902404834570,
0.754466481773496,
0.747792993224778,
0.720123205795894,
0.725196407909742,
0.720422662767765,
0.768084994907466,
0.806708711259786,
0.854086362155023,
0.946567160216025,
0.952523415211825,
1.04747948450491,
1.08801338757918,
1.14114760117758,
1.15049822920200,
1.14054838060270,

1.12919402582353,
1.08549128487138,
1.07297318367471,
0.994188479259514,
0.919460245665551,
0.799376086114842,
0.671586349360271,
0.490602838576890,
0.321343719302325,
0.197017182700309,
0.0816085992708266,
0.00940643793369848,
-0.0309104063439482,
-0.105321489546300,
-0.148638737542114,
-0.172236078311690,
-0.218053861609067,
-0.254125221966570,
-0.332573835278804,
-0.359359753764963,
-0.396445298401289,
-0.413686105876258,
-0.379876475363388,
-0.358301428671233,
-0.257237524308390,
-0.0993482943393087,
0.0542037399476356,
0.201007937894225,
0.333093002092513,
0.469949078196805,
0.604593772843395,
0.707286268824358,
0.799633575126767,
0.884289996080361,
0.958125814916900,
1.01033858336348,
1.05239762119660,
1.08076066405343,
1.09797352590896,
1.08165930053362,
1.10655900360224,
1.09885657303963,
1.12345575025865,
1.16977961526083,
1.11929043187398,
1.13462651767264,
1.12692233682151,
1.13015674807508,

```
1.15119814499059,
1.17587974750144,
1.19721721445396,
1.24599023752414,
1.20973378331380,
1.21956556162985,
1.13924853463546,
1.00267253933263,
0.857684073170001,
0.716626637620560,
0.609486062366760,
0.492424185092715,
0.505181527671631,
3.85669298166448,
-0.218339563032717]
```

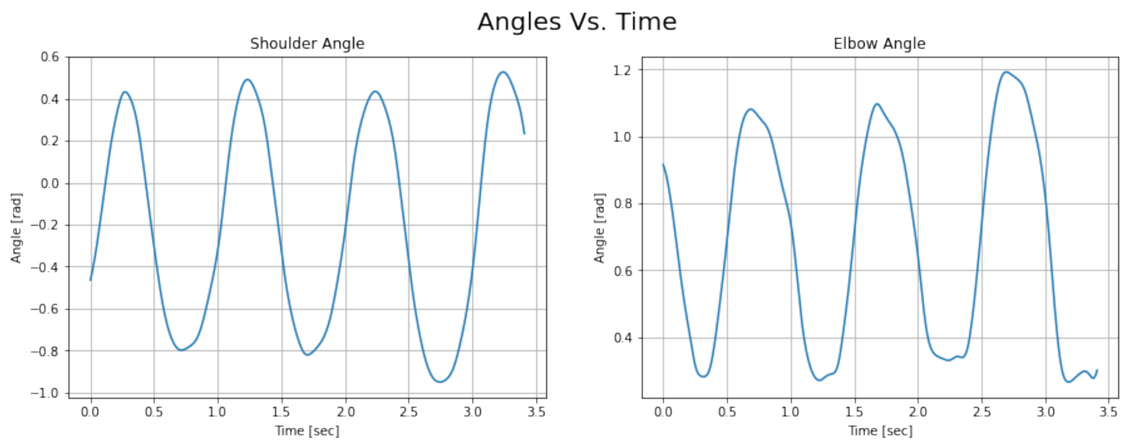
Example for the trial with the largest elbow torque & power:

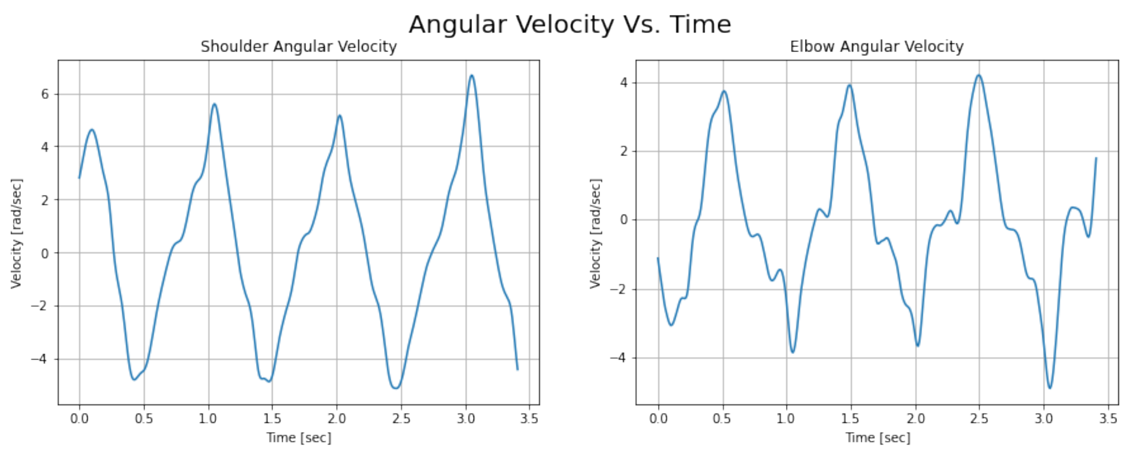
[35]:
```
index = 125
# index = 195
t_list = time_list[index]
theta1_list = sholder_Ang_list[index]
theta2_list = Elbow_Ang_list[index]
dtheta1_list = sholder_Vel_list[index]
dtheta2_list = Elbow_Vel_list[index]
ddtheta1_list = sholder_Acc_list[index]
ddtheta2_list = Elbow_Acc_list[index]
tau1_list = sholder_tau_list[index]
tau2_list = Elbow_tau_list[index]
current1_list = sholder_current_list[index]
current2_list = Elbow_current_list[index]
power1_list = sholder_power_list[index]
power2_list = Elbow_power_list[index]

# Compute the trajectory of the arm's motion
N = int((max(t_list) - min(t_list))/(1/frame_frequency))
tvec = np.linspace(min(t_list), max(t_list), N)
traj = np.zeros((6, N))
for i in range(N):
    traj[0, i] = theta1_list[i]
    traj[1, i] = theta2_list[i]
    traj[2, i] = dtheta1_list[i]
    traj[3, i] = dtheta2_list[i]
    traj[4, i] = ddtheta1_list[i]
    traj[5, i] = ddtheta2_list[i]

# Calculate the length difference between the time list and the trajectory lists
diff = (len(t_list) - len(traj[0]))
```

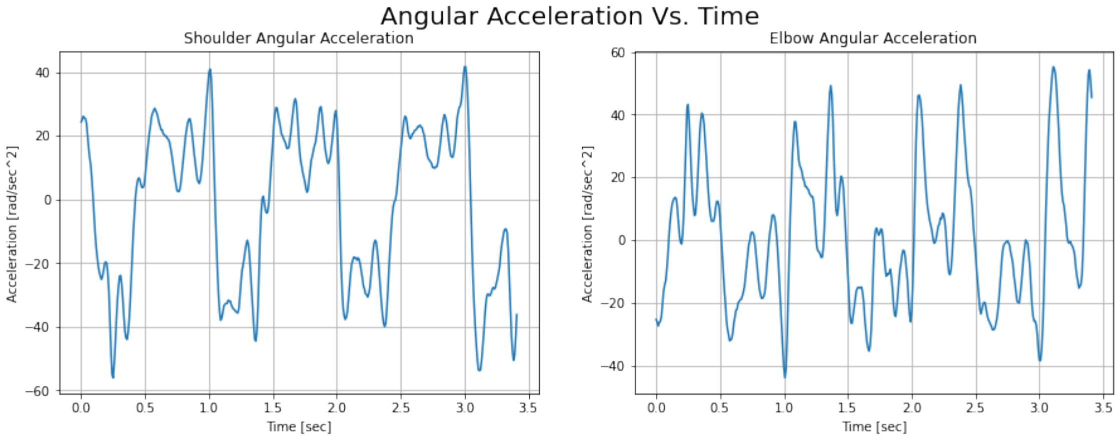
```
# Plot the trajectory lists (angles, velocities, accelerations, torques, and␣
↪power)
plt.figure(figsize=(15,5))
plt.suptitle('Angles Vs. Time', fontsize=20)
plt.subplot(121)
plt.plot(t_list[:-diff], traj[0])
plt.ylabel('Angle [rad]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Shoulder Angle')

plt.subplot(122)
plt.plot(t_list[:-diff], traj[1])
plt.ylabel('Angle [rad]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Elbow Angle')
plt.show()

plt.figure(figsize=(15,5))
plt.suptitle('Angular Velocity Vs. Time', fontsize=20)
plt.subplot(121)
plt.plot(t_list[:-diff], traj[2])
plt.ylabel('Velocity [rad/sec]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Shoulder Angular Velocity')

plt.subplot(122)
plt.plot(t_list[:-diff], traj[3])
plt.ylabel('Velocity [rad/sec]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Elbow Angular Velocity')
plt.show()

plt.figure(figsize=(15,5))
plt.suptitle('Angular Acceleration Vs. Time', fontsize=20)
plt.subplot(121)
plt.plot(t_list[:-diff], traj[4])
plt.ylabel('Acceleration [rad/sec^2]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Shoulder Angular Acceleration')

plt.subplot(122)
```

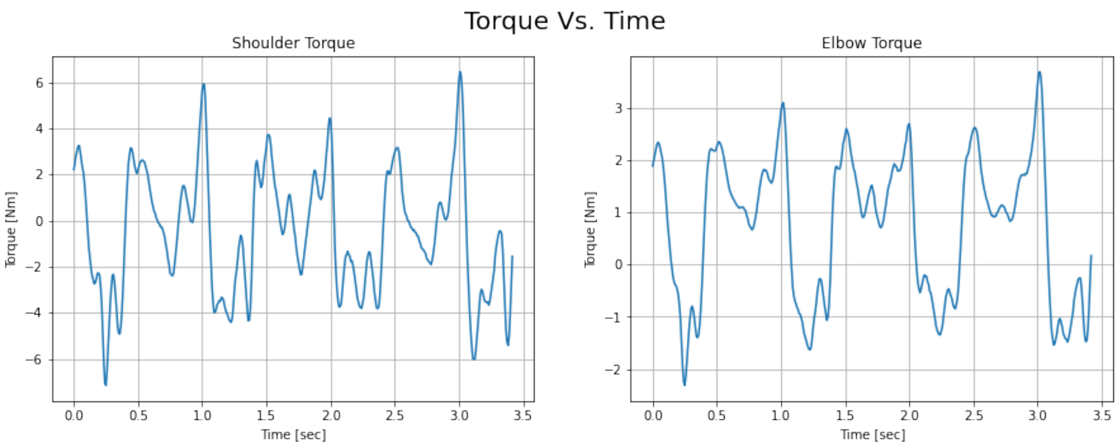
```
plt.plot(t_list[:-diff], traj[5])
plt.ylabel('Acceleration [rad/sec^2]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Elbow Angular Acceleration')
plt.show()

plt.figure(figsize=(15,5))
plt.suptitle('Torque Vs. Time', fontsize=20)
plt.subplot(121)
plt.plot(t_list, tau1_list)
plt.ylabel('Torque [Nm]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Shoulder Torque')

plt.subplot(122)
plt.plot(t_list, tau2_list)
plt.ylabel('Torque [Nm]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Elbow Torque')
plt.show()

plt.figure(figsize=(15,5))
plt.suptitle('Power Vs. Time', fontsize=20)
plt.subplot(121)
plt.plot(t_list, power1_list)
plt.ylabel('Power [W]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Shoulder Power')

plt.subplot(122)
plt.plot(t_list, power2_list)
plt.ylabel('Power [W]')
plt.xlabel('Time [sec]')
plt.grid()
plt.title('Elbow Power')
plt.show()

plt.figure(figsize=(15,5))
plt.suptitle('Speed Vs. Torque', fontsize=20)
plt.subplot(121)
plt.plot(tau1_list[:-diff], traj[2])
plt.ylabel('Velocity [rad/sec]')
plt.xlabel('Torque [Nm]')
```

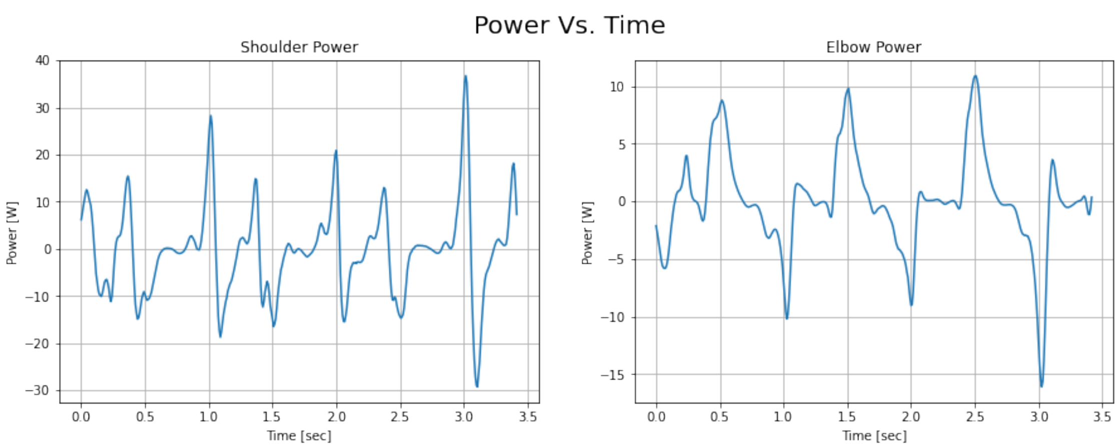
```
plt.grid()
plt.title('Shoulder Speed-Torque')

plt.subplot(122)
plt.plot(tau2_list[:-diff], traj[3])
plt.ylabel('Velocity [rad/sec]')
plt.xlabel('Torque [Nm]')
plt.grid()
plt.title('Elbow Speed-Torque')
plt.show()
```

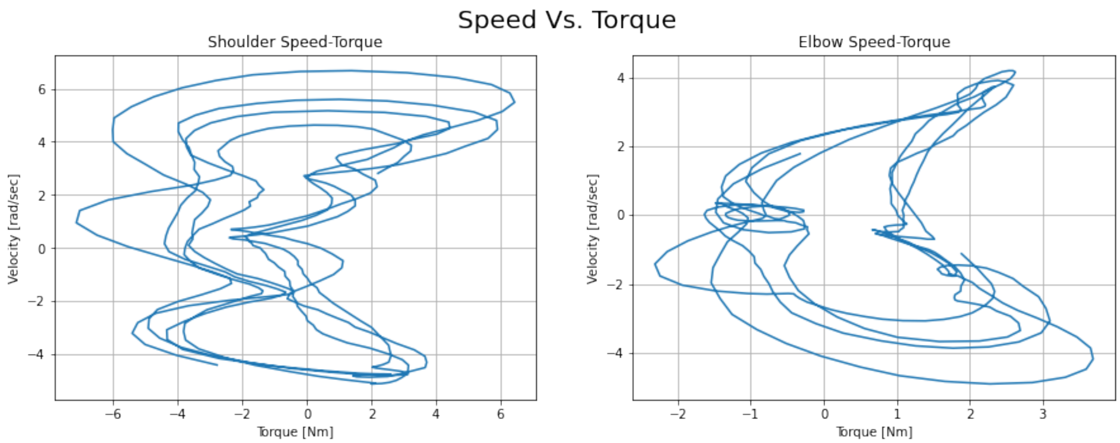





Angular Acceleration Vs. Time
Shoulder Angular Acceleration
Elbow Angular Acceleration
Acceleration [rad/sec^2]
Time [sec]


Torque Vs. Time
Shoulder Torque
Elbow Torque
Torque [Nm]
Time [sec]


Power Vs. Time
Shoulder Power
Elbow Power
Power [W]
Time [sec]



Animating the simulation:

[36]:
```
def animate_double_pend(traj, L1, L2, L1_COM, L2_COM, T):
    """
    Function to generate web-based animation of double-pendulum system

    Parameters:
        traj:         trajectory of theta1 and theta2
        L1:           length of the upper arm
        L2:           length of the lower arm
        L1_COM:       length of the center of mass of the upper arm from the␣
↪shoulder
        L2_COM:       length of the center of mass of the lower arm from the␣
↪elbow
        T:            length/seconds of animation duration

    Returns: None
    """

    # Browser configuration
    def configure_plotly_browser_state():
        import IPython
        display(IPython.core.display.HTML('''
            <script src="/static/components/requirejs/require.js"></script>
            <script>
              requirejs.config({
                paths: {
                  base: '/static/base',
                  plotly: 'https://cdn.plot.ly/plotly-1.5.1.min.js?noext',
                },
              });
            </script>
```

```
          '''))
configure_plotly_browser_state()
init_notebook_mode(connected=False)

# Getting data from pendulum angle trajectories
xx1 = L1 * np.sin(traj[0])
yy1 = -L1 * np.cos(traj[0])
xx1_COM = L1_COM * np.sin(traj[0])
yy1_COM = -L1_COM * np.cos(traj[0])
xx2 = xx1 + L2 * np.sin(traj[0] + traj[1])
yy2 = yy1 - L2 * np.cos(traj[0] + traj[1])
xx2_COM = xx1 + L2_COM * np.sin(traj[0] + traj[1])
yy2_COM = yy1 - L2_COM * np.cos(traj[0] + traj[1])
N = len(traj[0])

# Using these to specify axis limits
xm = np.min(xx1)
xM = np.max(xx1)
ym = np.min(yy1) - 0.6
yM = np.max(yy1) + 0.6

# Defining data dictionary
data = [dict(x=xx1, y=yy1,
             mode='lines', name='Arm',
             line=dict(width=5, color='blue')
            ),
        dict(x=xx1_COM, y=yy1_COM,
             mode='lines', name='Upper Arm Center of Mass',
             line=dict(width=2, color='green')
            ),
        dict(x=xx2_COM, y=yy2_COM,
             mode='lines', name='Lower Arm Center of Mass',
             line=dict(width=2, color='orange')
            ),
        dict(x=xx1, y=yy1,
             mode='markers', name='Elbow Trajectory',
             marker=dict(color="green", size=2)
            ),
        dict(x=xx2, y=yy2,
             mode='markers', name='Hand Trajectory',
             marker=dict(color="orange", size=2)
            )
      ]

# Preparing simulation layout
layout = dict(xaxis=dict(range=[xm, xM], autorange=False,␣
↪zeroline=False,dtick=1),
```

```
                  yaxis=dict(range=[ym, yM], autorange=False,␣
↪zeroline=False,scaleanchor = "x",dtick=1),
                  title='Simulation of Arm Modeled as a Double Pendulum',
                  hovermode='closest',
                  updatemenus= [{'type': 'buttons',
                                 'buttons': [{'label': 'Play', 'method':␣
↪'animate',
                                              'args': [None, {'frame':␣
↪{'duration': T, 'redraw': False}}]},
                                             {'args': [[None], {'frame':␣
↪{'duration': T, 'redraw': False}, 'mode': 'immediate',
                                              'transition': {'duration':␣
↪0}}],'label': 'Pause', 'method': 'animate'}
                                            ]
                                }]
                 )

    # Defining the frames of the simulation
    frames = [dict(data=[dict(x=[0,xx1[k],xx2[k]],
                              y=[0,yy1[k],yy2[k]],
                              mode='lines',
                              line=dict(color='red', width=4)),
                         go.Scatter(
                              x=[xx1_COM[k]],
                              y=[yy1_COM[k]],
                              mode="markers",
                              marker=dict(color="blue", size=12)),
                         go.Scatter(
                              x=[xx2_COM[k]],
                              y=[yy2_COM[k]],
                              mode="markers",
                              marker=dict(color="purple", size=12)),
                        ]) for k in range(N)]

    # Putting it all together and plotting
    figure = dict(data=data, layout=layout, frames=frames)
    iplot(figure)

# Animate the system
L1 = L_upper_arm_dict[participants_list[index]]
L2 = L_lower_arm
L1_COM = L_upper_arm_COM_dict[participants_list[index]]
L2_COM = L_lower_arm_COM
T = 5

animate_double_pend(traj, L1, L2, L1_COM, L2_COM, T)
```

<IPython.core.display.HTML object>

1.5 Motor Selection

Plotting the torque-speed curve of all trials and the torque-speed curve of some motors:

[22]:
```
# Compute the torque and speed vectors of the arm's motion for all trials
tot_dtheta1_list = []
tot_dtheta2_list = []
tot_tau1_list = []
tot_tau2_list = []

for lst in range(len(time_list)):
    for i in range(len(time_list[lst])):
        tot_dtheta1_list.append(sholder_Vel_list[lst][i])
        tot_dtheta2_list.append(Elbow_Vel_list[lst][i])
        tot_tau1_list.append(sholder_tau_list[lst][i])
        tot_tau2_list.append(Elbow_tau_list[lst][i])

# Compute the torque and speed vectors of some motors
######################################## T-Motor␣
↪########################################
# T-Motor, GL80 (KV30):
tau_stall_GL80_30 = 1.75
no_load_speed_GL80_30 = 720*2*pi/60
motor_speed_GL80_30 = [2*no_load_speed_GL80_30, no_load_speed_GL80_30, 0, -.
↪5*no_load_speed_GL80_30]
motor_torque_GL80_30 = [-tau_stall_GL80_30, 0, tau_stall_GL80_30, 1.
↪5*tau_stall_GL80_30]

# T-Motor, GL80 (KV60):
tau_stall_GL80_60 = 2.9
no_load_speed_GL80_60 = 1440*2*pi/60
motor_speed_GL80_60 = [2*no_load_speed_GL80_60, no_load_speed_GL80_60, 0, -.
↪5*no_load_speed_GL80_60]
motor_torque_GL80_60 = [-tau_stall_GL80_60, 0, tau_stall_GL80_60, 1.
↪5*tau_stall_GL80_60]

# T-Motor, G80 (KV30):
tau_stall_G80_30 = 2.9
no_load_speed_G80_30 = 700*2*pi/60
motor_speed_G80_30 = [2*no_load_speed_G80_30, no_load_speed_G80_30, 0, -.
↪5*no_load_speed_G80_30]
motor_torque_G80_30 = [-tau_stall_G80_30, 0, tau_stall_G80_30, 1.
↪5*tau_stall_G80_30]

# T-Motor, G80 (KV60):
```

```
tau_stall_G80_60 = 2.9
no_load_speed_G80_60 = 1400*2*pi/60
motor_speed_G80_60 = [2*no_load_speed_G80_60, no_load_speed_G80_60, 0, -.5*no_load_speed_G80_60]
motor_torque_G80_60 = [-tau_stall_G80_60, 0, tau_stall_G80_60, 1.5*tau_stall_G80_60]

# T-Motor, GL60 (KV25):
tau_stall_GL60_25 = 1.75
no_load_speed_GL60_25 = 600*2*pi/60
motor_speed_GL60_25 = [2*no_load_speed_GL60_25, no_load_speed_GL60_25, 0, -.5*no_load_speed_GL60_25]
motor_torque_GL60_25 = [-tau_stall_GL60_25, 0, tau_stall_GL60_25, 1.5*tau_stall_GL60_25]

# T-Motor, GL60 (KV55):
tau_stall_GL60_55 = 1.75
no_load_speed_GL60_55 = 1200*2*pi/60
motor_speed_GL60_55 = [2*no_load_speed_GL60_55, no_load_speed_GL60_55, 0, -.5*no_load_speed_GL60_55]
motor_torque_GL60_55 = [-tau_stall_GL60_55, 0, tau_stall_GL60_55, 1.5*tau_stall_GL60_55]

# T-Motor, GL100 (KV10):
tau_stall_GL100 = 7.7
no_load_speed_GL100 = 250*2*pi/60
motor_speed_GL100 = [2*no_load_speed_GL100, no_load_speed_GL100, 0, -.5*no_load_speed_GL100]
motor_torque_GL100 = [-tau_stall_GL100, 0, tau_stall_GL100, 1.5*tau_stall_GL100]

# T-Motor, G100 (KV10):
tau_stall_G100 = 7.7
no_load_speed_G100 = 250*2*pi/60
motor_speed_G100 = [2*no_load_speed_G100, no_load_speed_G100, 0, -.5*no_load_speed_G100]
motor_torque_G100 = [-tau_stall_G100, 0, tau_stall_G100, 1.5*tau_stall_G100]

# T-Motor, R60 (KV115):
tau_stall_R60 = 16.96
no_load_speed_R60 = 5520*2*pi/60
motor_speed_R60 = [2*no_load_speed_R60, no_load_speed_R60, 0, -.5*no_load_speed_R60]
motor_torque_R60 = [-tau_stall_R60, 0, tau_stall_R60, 1.5*tau_stall_R60]

# T-Motor, R80 (KV110):
tau_stall_R80 = 17.73
```

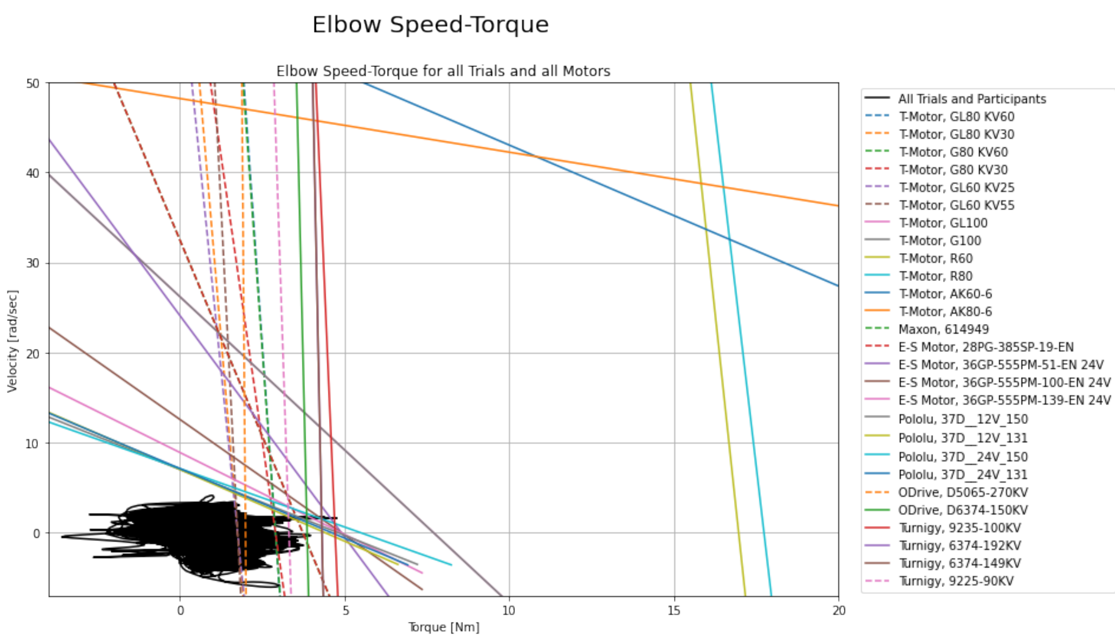
```
no_load_speed_R80 = 5280*2*pi/60
motor_speed_R80 = [2*no_load_speed_R80, no_load_speed_R80, 0, -.
↪5*no_load_speed_R80]
motor_torque_R80 = [-tau_stall_R80, 0, tau_stall_R80, 1.5*tau_stall_R80]

# T-Motor, AK60-6:
tau_stall_AK60_6 = 37.49
no_load_speed_AK60_6 = 560*2*pi/60
motor_speed_AK60_6 = [2*no_load_speed_AK60_6, no_load_speed_AK60_6, 0, -.
↪5*no_load_speed_AK60_6]
motor_torque_AK60_6 = [-tau_stall_AK60_6, 0, tau_stall_AK60_6, 1.
↪5*tau_stall_AK60_6]

# T-Motor, AK80-6:
tau_stall_AK80_6 = 80.888
no_load_speed_AK80_6 = 460*2*pi/60
motor_speed_AK80_6 = [2*no_load_speed_AK80_6, no_load_speed_AK80_6, 0, -.
↪5*no_load_speed_AK80_6]
motor_torque_AK80_6 = [-tau_stall_AK80_6, 0, tau_stall_AK80_6, 1.
↪5*tau_stall_AK80_6]

######################################## Maxon␣
↪########################################
# Maxon, 614949:
tau_stall_614949 = 4.3
no_load_speed_614949 = 4300*2*pi/60
motor_speed_614949 = [2*no_load_speed_614949, no_load_speed_614949, 0, -.
↪5*no_load_speed_614949]
motor_torque_614949 = [-tau_stall_614949, 0, tau_stall_614949, 1.
↪5*tau_stall_614949]

######################################## E-S Motor␣
↪########################################
# E-S Motor, 28PG-385SP-19-EN:
tau_stall_28PG = 3.73
no_load_speed_28PG = 310*2*pi/60
motor_speed_28PG = [2*no_load_speed_28PG, no_load_speed_28PG, 0, -.
↪5*no_load_speed_28PG]
motor_torque_28PG = [-tau_stall_28PG, 0, tau_stall_28PG, 1.5*tau_stall_28PG]

# E-S Motor, 36GP-555PM-51-EN 24V:
tau_stall_36GP_51 = 4.90
no_load_speed_36GP_51 = 230*2*pi/60
motor_speed_36GP_51 = [2*no_load_speed_36GP_51, no_load_speed_36GP_51, 0, -.
↪5*no_load_speed_36GP_51]
```

```
motor_torque_36GP_51 = [-tau_stall_36GP_51, 0, tau_stall_36GP_51, 1.
↪5*tau_stall_36GP_51]

# E-S Motor, 36GP-555PM-100-EN 24V:
tau_stall_36GP_100 = 4.90
no_load_speed_36GP_100 = 120*2*pi/60
motor_speed_36GP_100 = [2*no_load_speed_36GP_100, no_load_speed_36GP_100, 0, -.
↪5*no_load_speed_36GP_100]
motor_torque_36GP_100 = [-tau_stall_36GP_100, 0, tau_stall_36GP_100, 1.
↪5*tau_stall_36GP_100]

# E-S Motor, 36GP-555PM-139-EN 24V:
tau_stall_36GP_139 = 4.90
no_load_speed_36GP_139 = 85*2*pi/60
motor_speed_36GP_139 = [2*no_load_speed_36GP_139, no_load_speed_36GP_139, 0, -.
↪5*no_load_speed_36GP_139]
motor_torque_36GP_139 = [-tau_stall_36GP_139, 0, tau_stall_36GP_139, 1.
↪5*tau_stall_36GP_139]

######################################## Pololu␣
↪########################################
# Pololu, 150:1 Metal Gearmotor 37Dx73L mm 12V with 64 CPR Encoder (Helical␣
↪Pinion):
tau_stall_37D__12V_150 = 4.805
no_load_speed_37D__12V_150 = 67*2*pi/60
motor_speed_37D__12V_150 = [2*no_load_speed_37D__12V_150,␣
↪no_load_speed_37D__12V_150, 0, -.5*no_load_speed_37D__12V_150]
motor_torque_37D__12V_150 = [-tau_stall_37D__12V_150, 0,␣
↪tau_stall_37D__12V_150, 1.5*tau_stall_37D__12V_150]

# Pololu, 131:1 Metal Gearmotor 37Dx73L mm 12V with 64 CPR Encoder (Helical␣
↪Pinion):
tau_stall_37D__12V_131 = 4.41
no_load_speed_37D__12V_131 = 76*2*pi/60
motor_speed_37D__12V_131 = [2*no_load_speed_37D__12V_131,␣
↪no_load_speed_37D__12V_131, 0, -.5*no_load_speed_37D__12V_131]
motor_torque_37D__12V_131 = [-tau_stall_37D__12V_131, 0,␣
↪tau_stall_37D__12V_131, 1.5*tau_stall_37D__12V_131]

# Pololu, 150:1 Metal Gearmotor 37Dx73L mm 24V with 64 CPR Encoder (Helical␣
↪Pinion):
tau_stall_37D__24V_150 = 5.49
no_load_speed_37D__24V_150 = 68*2*pi/60
motor_speed_37D__24V_150 = [2*no_load_speed_37D__24V_150,␣
↪no_load_speed_37D__24V_150, 0, -.5*no_load_speed_37D__24V_150]
```

```
motor_torque_37D__24V_150 = [-tau_stall_37D__24V_150, 0, tau_stall_37D__24V_150, 1.5*tau_stall_37D__24V_150]

# Pololu, 131:1 Metal Gearmotor 37Dx73L mm 24V with 64 CPR Encoder (Helical Pinion):
tau_stall_37D__24V_131 = 4.61
no_load_speed_37D__24V_131 = 79*2*pi/60
motor_speed_37D__24V_131 = [2*no_load_speed_37D__24V_131, no_load_speed_37D__24V_131, 0, -.5*no_load_speed_37D__24V_131]
motor_torque_37D__24V_131 = [-tau_stall_37D__24V_131, 0, tau_stall_37D__24V_131, 1.5*tau_stall_37D__24V_131]

# ODrive, DUAL SHAFT MOTOR - D5065 270KV:
tau_stall_D5065 = 1.99
no_load_speed_D5065 = 8640*2*pi/60
motor_speed_D5065 = [2*no_load_speed_D5065, no_load_speed_D5065, 0, -.5*no_load_speed_D5065]
motor_torque_D5065 = [-tau_stall_D5065, 0, tau_stall_D5065, 1.5*tau_stall_D5065]

# ODrive, DUAL SHAFT MOTOR - D6374 150KV:
tau_stall_D6374 = 3.86
no_load_speed_D6374 = 5760*2*pi/60
motor_speed_D6374 = [2*no_load_speed_D6374, no_load_speed_D6374, 0, -.5*no_load_speed_D6374]
motor_torque_D6374 = [-tau_stall_D6374, 0, tau_stall_D6374, 1.5*tau_stall_D6374]

# Turnigy, 9235-100KV Brushless Multi-Rotor Motor:
tau_stall_9235 = 4.71
no_load_speed_9235 = 3840*2*pi/60
motor_speed_9235 = [2*no_load_speed_9235, no_load_speed_9235, 0, -.5*no_load_speed_9235]
motor_torque_9235 = [-tau_stall_9235, 0, tau_stall_9235, 1.5*tau_stall_9235]

# Turnigy, SK8 6374-192KV Sensored Brushless Motor (14P):
tau_stall_6374_192KV = 4.31
no_load_speed_6374_192KV = 7373*2*pi/60
motor_speed_6374_192KV = [2*no_load_speed_6374_192KV, no_load_speed_6374_192KV, 0, -.5*no_load_speed_6374_192KV]
motor_torque_6374_192KV = [-tau_stall_6374_192KV, 0, tau_stall_6374_192KV, 1.5*tau_stall_6374_192KV]

# Turnigy, SK8 6374-149KV Sensored Brushless Motor (14P):
tau_stall_6374_149KV = 4.31
no_load_speed_6374_149KV = 7373*2*pi/60
motor_speed_6374_149KV = [2*no_load_speed_6374_149KV, no_load_speed_6374_149KV, 0, -.5*no_load_speed_6374_149KV]
```

```python
motor_torque_6374_149KV = [-tau_stall_6374_149KV, 0, tau_stall_6374_149KV, 1.5*tau_stall_6374_149KV]

# Turnigy, 9225-90KV Turnigy Multistar Brushless Multi-Rotor Motor:
tau_stall_9225 = 3.31
no_load_speed_9225 = 3456*2*pi/60
motor_speed_9225 = [2*no_load_speed_9225, no_load_speed_9225, 0, -.5*no_load_speed_9225]
motor_torque_9225 = [-tau_stall_9225, 0, tau_stall_9225, 1.5*tau_stall_9225]

# Plotting the torque-speed curves of the arm's motion and the motors
plt.figure(figsize=(12,8))
plt.plot(tot_tau2_list, tot_dtheta2_list, color='black', label='All Trials and Participants')
plt.plot(motor_torque_GL80_60, motor_speed_GL80_60, '--', label='T-Motor, GL80 KV60')
plt.plot(motor_torque_GL80_30, motor_speed_GL80_30, '--', label='T-Motor, GL80 KV30')
plt.plot(motor_torque_G80_60, motor_speed_G80_60, '--', label='T-Motor, G80 KV60')
plt.plot(motor_torque_G80_30, motor_speed_G80_30, '--', label='T-Motor, G80 KV30')
plt.plot(motor_torque_GL60_25, motor_speed_GL60_25, '--', label='T-Motor, GL60 KV25')
plt.plot(motor_torque_GL60_55, motor_speed_GL60_55, '--', label='T-Motor, GL60 KV55')
plt.plot(motor_torque_GL100, motor_speed_GL100, label='T-Motor, GL100')
plt.plot(motor_torque_G100, motor_speed_G100, label='T-Motor, G100')
plt.plot(motor_torque_R60, motor_speed_R60, label='T-Motor, R60')
plt.plot(motor_torque_R80, motor_speed_R80, label='T-Motor, R80')
plt.plot(motor_torque_AK60_6, motor_speed_AK60_6, label='T-Motor, AK60-6')
plt.plot(motor_torque_AK80_6, motor_speed_AK80_6, label='T-Motor, AK80-6')
plt.plot(motor_torque_28PG, motor_speed_28PG, '--', label='Maxon, 614949')
plt.plot(motor_torque_28PG, motor_speed_28PG, '--', label='E-S Motor, 28PG-385SP-19-EN')
plt.plot(motor_torque_36GP_51, motor_speed_36GP_51, label='E-S Motor, 36GP-555PM-51-EN 24V')
plt.plot(motor_torque_36GP_100, motor_speed_36GP_100, label='E-S Motor, 36GP-555PM-100-EN 24V')
plt.plot(motor_torque_36GP_139, motor_speed_36GP_139, label='E-S Motor, 36GP-555PM-139-EN 24V')
plt.plot(motor_torque_37D__12V_150, motor_speed_37D__12V_150, label='Pololu, 37D__12V_150')
plt.plot(motor_torque_37D__12V_131, motor_speed_37D__12V_150, label='Pololu, 37D__12V_131')
```

```python
plt.plot(motor_torque_37D__24V_150, motor_speed_37D__24V_150, label='Pololu, 37D__24V_150')
plt.plot(motor_torque_37D__24V_131, motor_speed_37D__24V_150, label='Pololu, 37D__24V_131')
plt.plot(motor_torque_D5065, motor_speed_D5065, '--', label='ODrive, D5065-270KV')
plt.plot(motor_torque_D6374, motor_speed_D6374, label='ODrive, D6374-150KV')
plt.plot(motor_torque_9235, motor_speed_9235, label='Turnigy, 9235-100KV')
plt.plot(motor_torque_6374_192KV, motor_speed_6374_192KV, label='Turnigy, 6374-192KV')
plt.plot(motor_torque_6374_149KV, motor_speed_6374_149KV, label='Turnigy, 6374-149KV')
plt.plot(motor_torque_9225, motor_speed_9225, linestyle='--', label='Turnigy, 9225-90KV')

plt.ylabel('Velocity [rad/sec]')
plt.xlabel('Torque [Nm]')
plt.xlim([-4, 20])
plt.ylim([-7, 50])
plt.grid()
plt.legend(loc='upper left', bbox_to_anchor=(1.02,1))
plt.suptitle('Elbow Speed-Torque', fontsize=20)
plt.title('Elbow Speed-Torque for all Trials and all Motors')
plt.show()
```



[ ]: