# Projet 1 Fidget Spinner

## 1.1 Introduction

In this project, we will measure and analyze the behavior of a fidget spinner using a methodology inspired by these two videos (link 1, link 2). It turns out that it is possible to estimate the spinner's angular velocity (rotation rate), $\omega(t)$, as a function of time from a video recording.

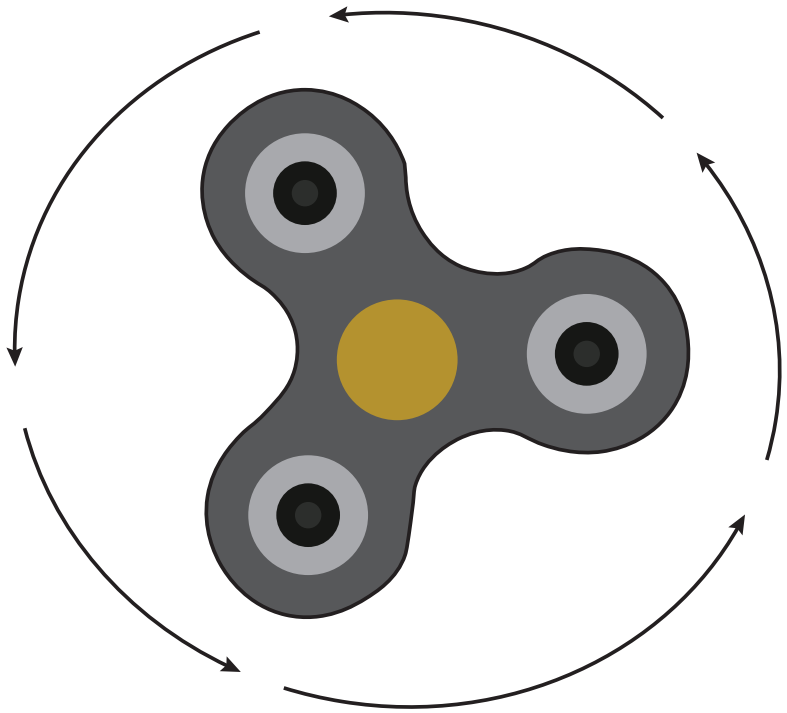

Figure 1.1: A fidget spinner.

As is the running theme of this portion of the course, we'd like to model this time series data ($\omega(t)$ as a function of $t$) using a first-order ODE. However, unlike our models for capacitors and cooling coffee, this ODE will be nonlinear and does not have an exponential function as its solution. Thus, we will not be fitting an exponential to the graph of the dependent variable, $\omega(t)$, as a function of $t$; as we might usually do for a linear first-order system. Since we will not know the analytical solution to our chosen ODE, we will take an alternative approach: fitting a curve to the graph of $\dot{\omega}(t)$ as a function of $\omega(t)$. This will allows us to find an approximation for the explicit form of the ODE:

$$\dot{\omega}(t) = f(t, \omega(t)) \tag{1.1}$$

which we can then use to make predictions about the behavior of the system.

## 1.2 System Model

Our ODE will be based on a simplified dynamics model for a rotating body:

$$I\alpha = \sum \tau \tag{1.2}$$

where:

- $I$ is the moment of inertia of the fidget spinner.
- $\alpha = \dot{\omega}$ is the angular acceleration of the fidget spinner.
- $\sum \tau$: is the sum of external torques acting on the fidget spinner.

Here, we will consider three types of torques acting on the fidget spinner, causing it to slow down:

$$\sum \tau = \tau_C + \tau_V + \tau_Q \tag{1.3}$$

- $\tau_Q$: is the torque due to quadratic drag, which scales with the square of the airspeed:

$$\tau_Q = -d_1\omega^2 \mid \omega > 0, \qquad \tau_Q = d_1\omega^2 \mid \omega < 0 \tag{1.4}$$

- $\tau_V$: is the torque due to viscous drag, which is linear drag with the surrounding air:

$$\tau_V = -d_2\omega \tag{1.5}$$

- $\tau_C$: is the torque due to contact friction, which is generated as a result of sliding contact at the rotating joint. This torque/force is often modeled as being a constant acting opposite of the direction of motion:

$$\tau_C = -d_3 \mid \omega > 0, \qquad \tau_C = d_3 \mid \omega < 0 \tag{1.6}$$

Note that we assume that the external torques do not depend on the angle, $\theta(t)$, of the fidget spinner, due to the rotational symmetry of the system (otherwise, we would need to use a second-order ODE to model its behavior). Combining all of this together, we get a model for the behavior of our system:

$$I\dot{\omega}(t) = -d_1\omega(t)^2 - d_2\omega(t) - d_3 \tag{1.7}$$

which can be rewritten as:

$$\boxed{\dot{\omega}(t) = a\omega(t)^2 + b\omega(t) + c} \tag{1.8}$$

where:

$$a = -\frac{d_1}{I}, \qquad b = -\frac{d_2}{I}, \qquad c = -\frac{d_3}{I} \tag{1.9}$$

Note that 1.8 is a nonlinear ODE, and therefore will not have an exponential function as its solution. Furthermore, most nonlinear ODE's do not have analytical solutions. However, it is still possible to estimate the system parameters, $a$, $b$ and $c$ from the data. We will accomplish this by estimating $\dot{\omega}(t)$ (using the finite difference approximation), and then fitting a parabola to the graph of $\dot{\omega}(t)$ as a function of $\omega(t)$.

## 1.3 Data Collection

We need to capture a video of the fidget spinner spinning until it stops, and then load that video onto a computer for analysis. In order for the analysis code to work properly, please keep the following in mind:

- The video needs to be in a format that is supported by MATLAB:
  - If you are recording the video using a webcam and the camera app on windows, it should probably store the video as a MPEG-4 (.mp4) by default (which is MATLAB compatible).
  - If you are recording the video on an iPhone, the default format is probably HEVC, which is not compatible with MATLAB. To change it to a compatible format, go to Settings > Camera > Formats and select 'Most Compatible'.
  - Android phones should store videos to MPEG-4 (which is MATLAB compatible) by default.
- A higher frame rate (60 fps instead of 30 fps) is preferable:
  - If you are using the windows camera app, go to Settings > Video Settings, and choose a video quality with a good resolution and 60 fps.
  - If you are using an iPhone, go to Settings > Camera > Record Video and choose an option that is 60 fps. On the same page, scroll down, click on Auto FPS and disable it.
  - See this video for instructions on how to change your video recording resolution on Android.

  Remember to change your settings back once you have finished the lab!
- The fidget spinner should already be spinning by the time the recording has started (the video shouldn't contain footage of you spinning up the fidget spinner). If the recording started too early, you can always clip the desired footage (using whichever method you choose).
- The video should record until the fidget spinner has come to a full stop. We recommend that the video contain at least two to three seconds of footage where the fidget spinner is at rest.
- The background of the video (behind the fidget spinner) should be static. This can be accomplished by setting the spinner horizontally on a table and using the table or a piece of paper as the background.
- The position of the fidget spinner should be fixed in the frame of the video. The easiest way to do this is to set the camera on top of something (like a stack of books) facing down. Alternatively, if you have steady hands, you can just hold your phone to record the video.
- The video will need to be transferred to your computer:
  - If you are using a mac and an iPhone, you can airdrop the video.
  - See these instructions for importing a video from iPhone/Android to PC.
  - In the worst case scenario, either try to email the video to yourself, or upload it to google drive.

We recommend using a box or book to hold up your phone above the fidget spinner to record the video. Use whichever means you prefer to send the video to your computer for analysis.

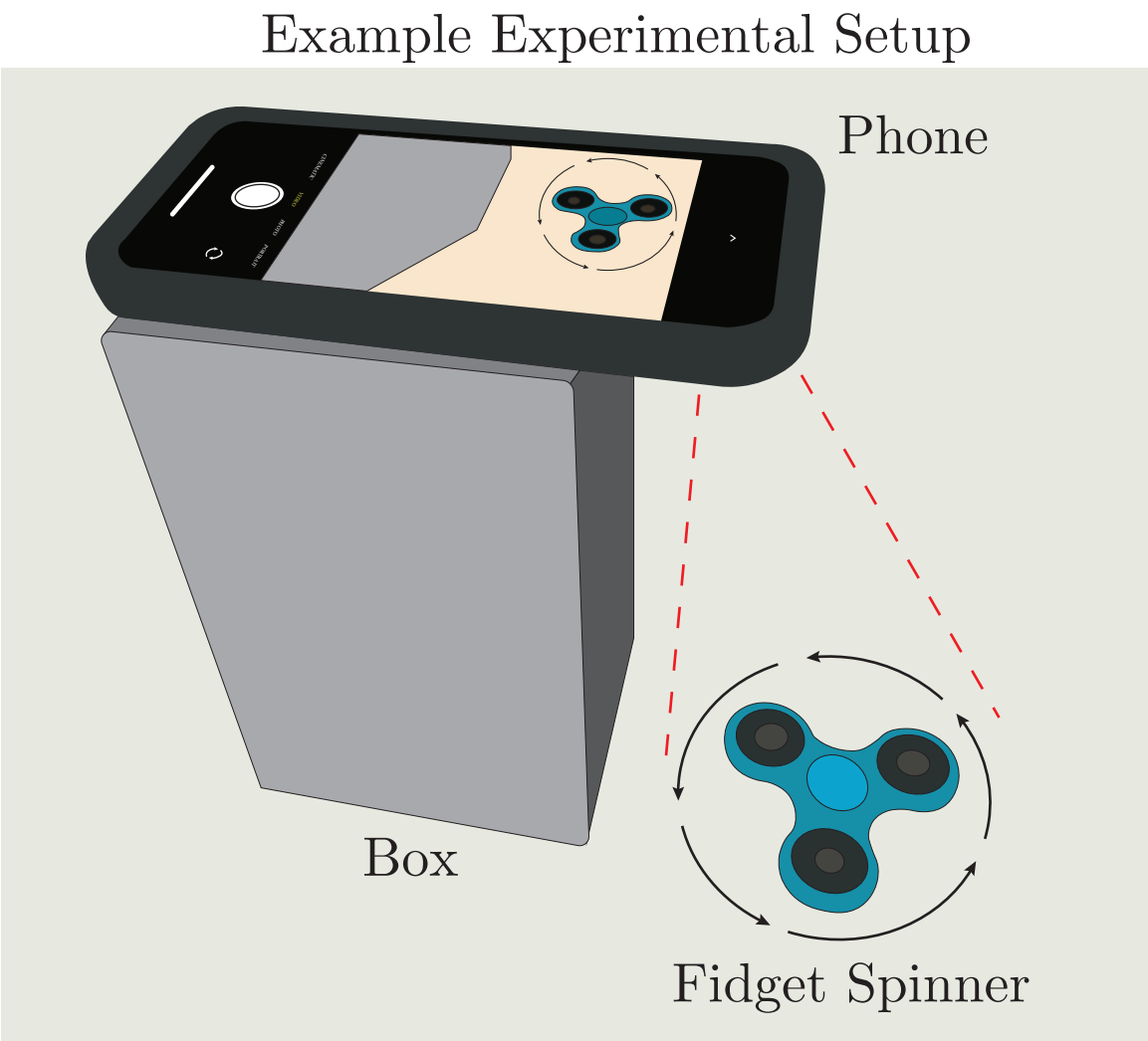


Figure 1.2: An example recording setup. Here, we use a small box to hold up the phone above the spinner.

## 1.4 Video Processing

In order to analyze the behavior of the fidget spinner, we need to estimate its angular velocity from the video data. How are we going to accomplish this? Here, we will exploit the fact that the rotation of the fidget spinner will cause the video pixels to oscillate between the color of the spinner and the color of the background (see Fig. 1.3). The oscillation frequency should be proportional to the rate of rotation (why?). Our plan is to average this oscillating signal spatially across pixels for a small window of the video, and then use tools from frequency analysis to measure the rotation rate of the spinner.

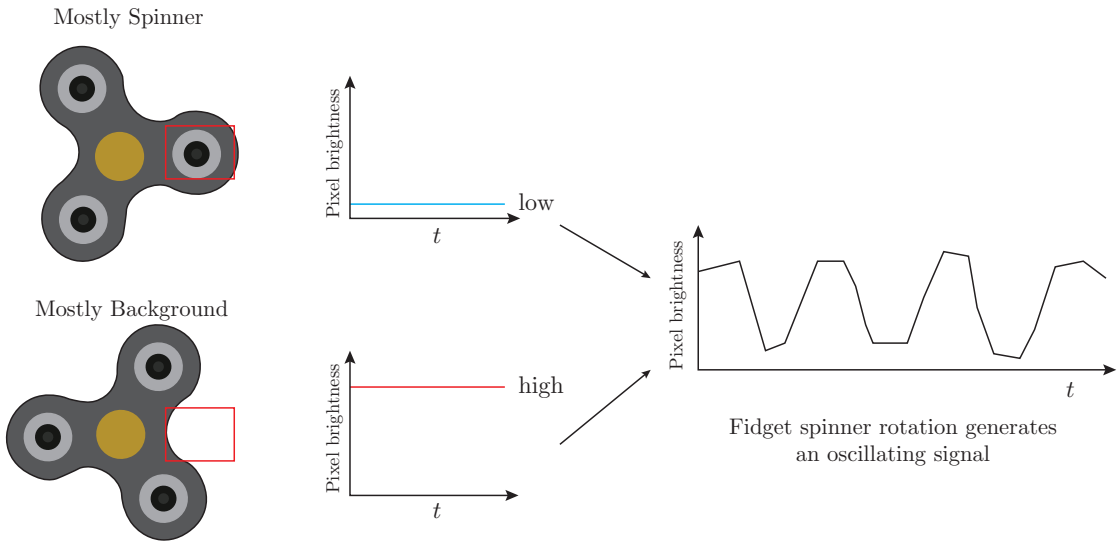


Figure 1.3: The rotation of the fidget spinner causes the pixel values to oscillate between the color of the background and the color of the spinner.

We have written two functions for you to help process the video data:

- `video_to_signal`: converts the video file of the fidget spinner to a time signal by computing the average pixel value in a window for each frame.
- `fidget_spinner_FFT`: uses the Fast Fourier Transform (FFT) and some filtering tricks to extract angular velocity of the fidget spinner as a function of time. This code also corrects for the stroboscopic effect, which is a type of aliasing.

These functions can be found on the Canvas page. Please download them and place them in your working MATLAB folder. The first of these functions, `video_to_signal`, computes an average pixel value across a window (determined by you) for each frame of the video. A usage example can be found on the next page.

Initially, you will want to watch the debugging video, so start by setting the parameter `show_image` to 1. You will need to choose the location of the window to average the pixel values across. Our goal is to get a signal that oscillates at a frequency that is proportional to the rotation rate of the fidget spinner. To do so, change the values of top/bottom/left/right to choose a window that covers a small portion of the fidget spinner (see Fig. 1.4). Ideally, the contents of the window will alternate between the color of the spinner and the color of the background (resulting in an oscillating signal). Keep in mind that the pixel index increases when going from top to bottom, so top $<$ bottom.

```
%fname: a string of the filename of the video you want to process
%     the the video is not in the same folder, fname should include
%     the absolute path to the file
fname = 'C:\Users\taylorott\Pictures\Camera Roll\WIN_20250828_14_12_58_Pro.mp4';

%window_bounds: a MATLAB struct that indicates the boundaries of the
%     window to use for averaging the pixel value.
window_bounds = struct();
window_bounds.top = 300;
window_bounds.bottom = 400;
window_bounds.left = 700;
window_bounds.right = 800;

%show_image: a boolean (0 or 1) that determines whether or not the video
%     of the fidget spinner is displayed during processing
%     set show_image to 1 if you are still trying to figure out
%     the boundaries of the window to use
%     set show_image to 0 to process the video faster
show_image = 1;

%Converts the video file of a fidget spinner to a time signal
%by computing the average pixel value in a window for each frame
%OUTPUTS:
%y: a list of the averaged pixel value in the window
%Fs: framerate of the video
[y,Fs] = video_to_signal(fname,window_bounds,show_image);
```

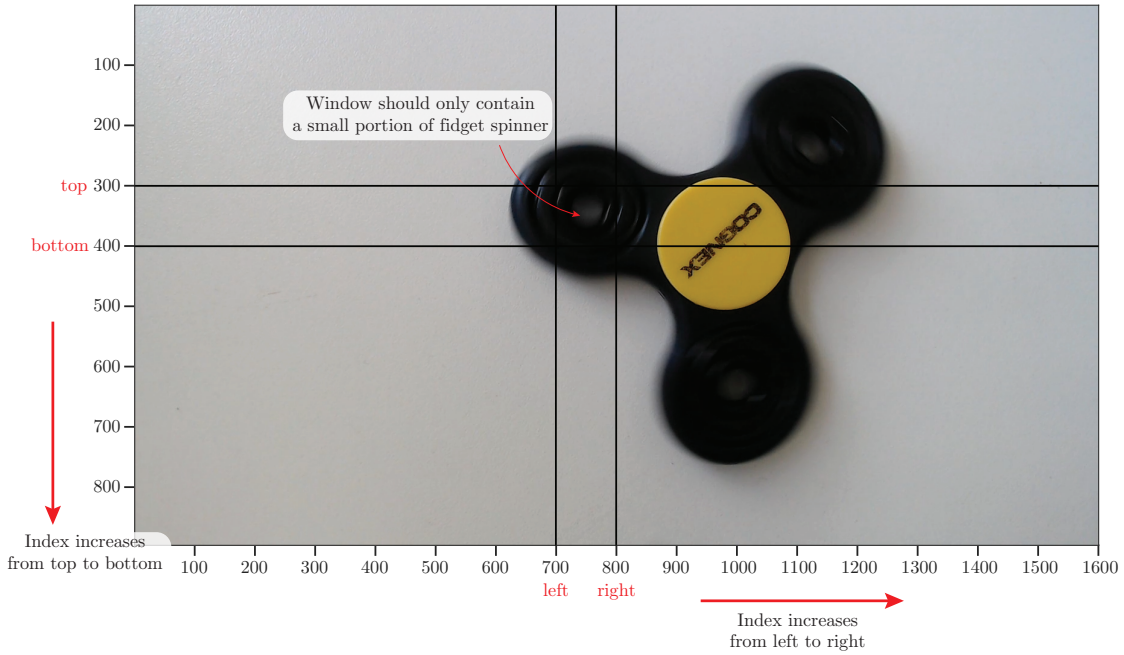


Figure 1.4: Sample frame when running the video processing function. Note that you should choose a window that contains only a portion of the fidget spinner. Additionally, because the pixel indexing convention increases from top to bottom, top $<$ bottom

.

Once you have chosen your window and processed the video, we recommend that you save your results in a file so you don't have to process the video every time you want to look at the data:

```
fname_save = 'your path and file name here.mat'; %should be a .mat file

save(fname_save,'y','Fs'); %save results from video_to_signal

S = load(fname_save,'y','Fs'); %load results from save file
y = S.y; Fs = S.Fs; %place values in variables y and Fs
```

The MATLAB function `fidget_spinner_FFT` extracts the angular velocity of the fidget spinner from the averaged pixel signal using the Fast Fourier Transform. A usage example is provided below. Note that you will need to set the following parameter values:

- `T_window`: the width (in seconds) of the window for the FFT. We've found that .7 usually works well.
- `q`: A parameter (between 0 and 1) used to filter the frequency data. You might need to play with this number a bit, but we've found that a value of .6 usually works well.

```
%T_window: width of FFT window (in seconds) .7 is usually good
T_window = .7;

%q: a parameter used to filter which portions of frequency curve to use
%    (filter is by width of flat portions of curve)
%    value should be in interval [0,1]
%    values closer to 1 are most "exclusive".
%    .6 is a pretty good value to choose
q = .6;

%showAnalysis: boolean that turns visualization on/off
showAnalysis = 1;

%uses FFT and some filtering tricks to extract angular velocity of fidget
%spinner as a function of time
%OUTPUTS:
%tlist: list of times for measured values of angular frequency
%omega_list: measured value of angular frequency
[tlist,freq_list] = fidget_spinner_FFT(y,Fs,T_window,q,showAnalysis);
```

We recommend that you take a look at the debugging videos/plots that are generated by `fidget_spinner_FFT`. How do you think this function works? It's also recommended that you save the processed data in a file so you can easily access it in the next section.

```
%NOTE: you should choose a different file name than your other save file!!!
fname_save2 = 'your path and file name here.mat'; %should be a .mat file

save(fname_save2,'tlist','freq_list'); %save results from fidget_spinner_FFT

S2 = load(fname_save2,'tlist','freq_list'); %load results from save file
tlist = S2.tlist; freq_list = S2.freq_list; %unpack loaded variables
```

## 1.5 Data Analysis

### 1.5.1 Angular Velocity Correction

Using the MATLAB functions we have provided, you have extracted an estimate for the angular velocity, $\omega$, as a function of time, $t$. However, this does not account for the number of spokes/wings of the fidget spinner. Specifically, the 'apparent' rotation rate (the rotation rate estimated directly from the oscillation frequency of the color values) is the product of the 'true' rotation rate and the number of wings/spokes (why?):

$$\omega_{\text{apparent}} = (n_{\text{wings}})(\omega_{\text{true}}) \tag{1.10}$$

Use the 'apparent' rotation rate to compute the 'true' rotation rate.

### 1.5.2 Approximating Angular Acceleration

Assuming our model for the system is given by the ODE

$$\dot{\omega}(t) = a\omega(t)^2 + b\omega(t) + c \tag{1.11}$$

we would like to find the values of the coefficients $a$, $b$, and $c$ that best fit our measured data. To do this, we need measurements for both the angular velocity, $\omega(t)$, (which we have), and the angular acceleration, $\dot{\omega}(t)$, (which we do not currently have). Fortunately, we can approximate $\dot{\omega}(t)$ using the finite difference approximation. Given a pair of data points, $(t_i, \omega_i)$, $(t_{i+1}, \omega_{i+1})$, $\dot{\omega}$ can be approximated as follows:

$$\dot{\omega}_{i+1/2} \approx \frac{\omega_{i+1} - \omega_i}{t_{i+1} - t_i} \tag{1.12}$$

This corresponds to the data point $(\omega_{i+1/2}, \dot{\omega}_{i+1/2})$, which encodes the midpoint and slope of the line segment that connects $(t_i, \omega_i)$ to $(t_{i+1}, \omega_{i+1})$:

$$\omega_{i+1/2} \approx \frac{\omega_{i+1} + \omega_i}{2}, \qquad \dot{\omega}_{i+1/2} \approx \frac{\omega_{i+1} - \omega_i}{t_{i+1} - t_i} \tag{1.13}$$

Generate a plot of the estimated values of $(\omega_{i+1/2}, \dot{\omega}_{i+1/2})$ (Deliverable #6 below). This should be a scatter plot and not a line plot.

### 1.5.3 System Identification

Now that we have estimates for the angular acceleration, $\dot{\omega}$, as a function of angular velocity, $\omega$, we can use linear regression to fit our quadratic model to the data. Use polyfit to estimate the values of $a$, $b$, and $c$. Plot the fit on top of the graph of $(\omega_{i+1/2}, \dot{\omega}_{i+1/2})$ (Deliverable #6 below). Does the quadratic fit seem to capture the dynamics of the fidget spinner? Why or why not?

### 1.5.4 Simulation

Use ODE45 to simulate the system using the values you estimated for the model coefficients, $a$, $b$, and $c$, and the estimated value of $\omega(t = 0)$ generated by the video processing functions. Compare your simulated solution with the measured data. Does the model seem accurate? (Deliverables #9-10 below).

You may notice that, when simulating the system, $\omega(t)$ does not stop at rest, but instead becomes negative. Why is this? What behavior of the system does our ODE not capture? Correct for this by multiplying your rate equation by the sign of $\omega(t)$. How does this fix this simulation issue?

$$\dot{\omega}(t) = \big(a\omega(t)^2 + b\omega(t) + c\big)\text{sign}(\omega(t)) \tag{1.14}$$

### 1.5.5 Model Predictions

We have additional analysis exercises for you, which are detailed in the deliverables section.

## 1.6 Lab Report Deliverables

Make sure to include the following items in your lab report:

1. Please include a picture of your team's fidget spinner.
2. Please include a picture (or labeled sketch) of your team's experimental setup for recording the video.
3. Please include a written description of your experimental procedure.
   - Don't just copy-paste the experimental procedure from this document. We want ***your*** description.
4. What are the units of the angular velocity, $\omega(t)$? What are the units of the angular acceleration, $\dot{\omega}(t)$?
5. Please fill in the following table of values, which should be included in your lab report:

| **Quantity description** | **Symbol** | **Value** | **(units)** |
|---|---|---|---|
| Estimated quadratic drag constant | $a$ | | |
| Estimated viscous damping constant | $b$ | | |
| Estimated Coulomb friction constant | $c$ | | |
| Video frame rate | $F_s$ | | |
| Estimated initial angular velocity | $\omega(t=0)$ | | |

   where:

$$\dot{\omega}(t) = a\omega(t)^2 + b\omega(t) + c \tag{1.15}$$

6. Please include a plot of $\dot{\omega}(t)$ (that you estimated via finite differences) as a function of $\omega(t)$ . This plot should also depict your quadratic fit of the data [from subsections 1.5.2 and 1.5.3].
7. Does the quadratic fit seem to capture the dynamics of the fidget spinner? Why or why not?
8. What about our model (equation 1.15) makes it first order? Is our model linear or nonlinear? Is the system homogeneous or forced?
9. Please include a plot comparing the measured values of $\omega(t)$ with the solution that you simulated using ODE45 (with the same initial condition) [from subsection 1.5.4].
10. Based on a comparison between your simulated solution with the measured data. Does the model seem accurate?
11. Please include a quiver plot that visualizes the ODE using the parameter values of $a$, $b$, and $c$ that you measured. This quiver plot should be on the domain $t > 0, \omega > 0$ (do not consider negative values of time or angular velocity). This plot should also depict three simulated solutions for $\omega(t)$ with different initial conditions. Make sure to choose positive values for the initial condition, $\omega(t=0) = \omega_0$.
12. How are the angular velocity of the fidget spinner and the measured frequency of the video signal related to one another? Can you think of an equation relating these quantities?
13. Why do we need to divide for the number of wings/spokes when computing the angular velocity from the measured frequency of the video signal?
14. Write a short description (in your own words) of the stroboscopic effect. Were you able to observe this phenomenon when recording the video of your fidget spinner?
15. When collecting the data, we can observe that the fidget spinner will come to rest after a finite amount of time. How is this different from the first order systems that we analyzed previously? Do the solutions to this system have the form of an exponential curve? Why or why not?

16. Suppose we wanted to make the fidget spinner spin for longer. One option would be to make it out of a denser material, which would increase its moment of inertia. If we were to keep everything else the same (geometry, friction/drag coefficients etc.), then the new ODE for the system would be given by:

$$\frac{\rho_{new}}{\rho_{old}}\dot{\omega}(t) = a\omega(t)^2 + b\omega(t) + c \tag{1.16}$$

where $\rho_{new}/\rho_{old}$ is the density ratio of the new and old materials. Given the values of $\omega(t=0)$, $a$, $b$, and $c$ that you measured/estimated, use ODE45 to simulate the system for $\rho_{new}/\rho_{old}$ values of .5, 1, 2, and 4. How long does the fidget spinner spin in each of these cases? You are encouraged to just zoom in on the simulated plots to determine when the fidget spinner will come to rest; however, you can use whichever method you prefer.

17. Using the simulated results you just calculated, include a plot of $t_{stop}/t^*_{stop}$ as a function of $\rho_{new}/\rho_{old}$, where $t_{stop}$ is the simulated stop time of the fidget spinner (i.e. the time that $\omega(t)$ first reaches zero), and $t^*_{stop}$ is the simulated stop time when $\rho_{new}/\rho_{old} = 1$. Do you notice anything about the plot?

18. Let's define a new independent variable, $\tilde{t}$, which is the time scaled by the density ratio, $\rho_{new}/\rho_{old}$:

$$\tilde{t} = \frac{t}{\rho_{new}/\rho_{old}} \tag{1.17}$$

Show that our scaled ODE (1.16) can be rewritten as:

$$\frac{d\omega\left(\tilde{t}\right)}{d\tilde{t}} = a\omega\left(\tilde{t}\right)^2 + b\omega\left(\tilde{t}\right) + c \tag{1.18}$$

The chain rule of derivatives will come in handy here:

$$\dot{\omega}(t) = \frac{d\omega\left(t\right)}{dt} = \frac{d\omega\left(\tilde{t}\right)}{d\tilde{t}}\frac{d\tilde{t}}{dt} = \frac{d\omega\left(\tilde{t}\right)}{d\tilde{t}}\frac{\rho_{old}}{\rho_{new}} \tag{1.19}$$

How are this result (1.18) and the plot of $t_{stop}/t^*_{stop}$ vs. $\rho_{new}/\rho_{old}$ related?

19. Using the relationship that you observed in the previous plot, predict the stop time for the following materials, assuming that your fidget spinner is made of either ABS plastic ($\rho \approx 1.2\,\mathrm{g/cm^3}$ or Aluminum ($\rho \approx 2.7\,\mathrm{g/cm^3}$) if your spinner is made of either plastic or metal respectively:

- Diamond: ($\rho \approx 3.5\,\mathrm{g/cm^3}$)
- Gold: ($\rho \approx 19.32\,\mathrm{g/cm^3}$)

Please show your calculations. You should be able to predict the stop time without having to simulate the system again with ODE45.

20. Is increasing the material density an effective way to increase the spin time?