Engineered Muscle Contractility Analysis Pipeline - Master User Guide

Version: 1.0

Developed By: Philip Barrett

Date: January 12, 2026

CITATION

If you use this pipeline in your research, please cite the software as follows:

Barrett, P. (2026). Engineered Muscle Contractility Analysis Pipeline (EMCAP) (Version #) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.18228700

TABLE OF CONTENTS

---

1. Introduction & Overview

Welcome to the **E**ngineered **M**uscle **C**ontractility **A**nalysis **P**ipeline! (EMCAP) This suite of Python scripts is designed to batch process and analyze electrical stimulation data from

engineered human muscle tissues from a time force 24 well data format. The core strength of this pipeline lies in its custom, in-house peak finding algorithms, offering tailored analysis beyond standard platform outputs.

The pipeline includes four primary analysis scripts and four secondary "Prism Merge" scripts for post-processing and formatting data for GraphPad Prism.

- **Transparency:** Provides complete analytical visibility via open-source code and auto-generated calculation reference files, ensuring every peak detection and kinetic metric is fully traceable and reproducible.
- **Flexibility:** Deploys protocol-specific logic that automatically adapts peak detection strategies to match the distinct physiological profiles of twitch, 75Hz tetanic, and Force-Frequency protocols.
- **Robustness:** Ensures data integrity through a multi-layered validation system: an "Enhanced Detection Algorithm" (3-consecutive-point rule) and smoothing filters signal noise, while automatic baseline subtraction and configurable artifact rejection (e.g., Fast Peak Warnings, Minimum Twitch Thresholds) eliminate drift and non-physiological false positives.

---

2. Getting Started - System Setup

To ensure the analysis scripts work correctly, you'll use a pre-configured Python environment (UW_environment.yml) that contains all necessary packages and dependencies at the exact versions used to develop these scripts.

Prerequisites:

- A computer (Windows/Mac/Linux) with at least 8GB RAM recommended.
- Anaconda or Miniconda installed (see step 1 below).
- Time vs. Force data collected from 24-well plates in CSV (Recommended) or Excel formats
- A plate map in CSV format (.csv).

Installation Steps:

Step 1: Install Anaconda

Download and install Anaconda Individual Edition from anaconda.com/download, accepting all default settings during installation.

Step 2: Set Up the Python Environment

The UW_environment.yml file contains the exact Python configuration needed to run all analyses. Follow these steps carefully:

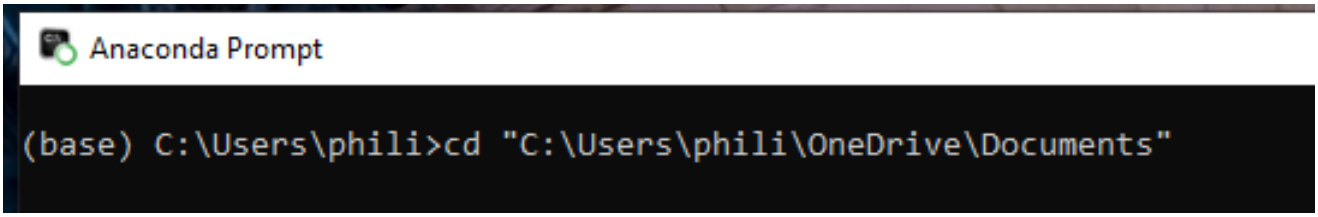
2a. Save the Environment File

Locate the UW_environment.yml file provided with these scripts. Save it to an easily accessible location, such as C:\Users\YourUsername\Documents\.

2b. Create the Environment

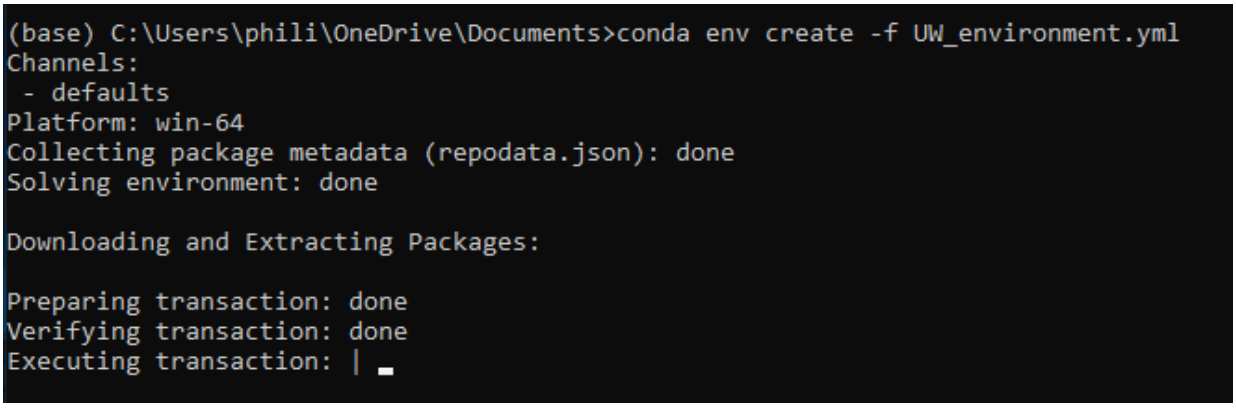
Open Anaconda Prompt (Windows) or Terminal (Mac/Linux). Navigate to the folder where you saved the .yml file:

cd C:\Users\YourUsername\Documents\
Press Enter



Create the environment by running:

conda env create -f UW_environment.yml
Press Enter



2c. Activate the Environment

Once installation completes, activate the environment:

conda activate UW
Press Enter

You should see (UW) appear at the beginning of your command prompt.

```
(base) C:\Users\phili\OneDrive\Documents>conda activate uw

(uw) C:\Users\phili\OneDrive\Documents>
```

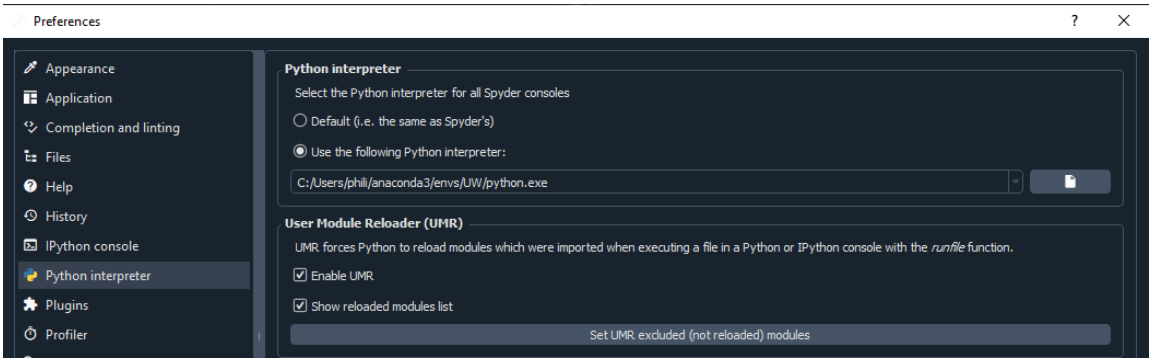
2d. Verify Installation

Run python --version. You should see: Python 3.12.4 (or similar).

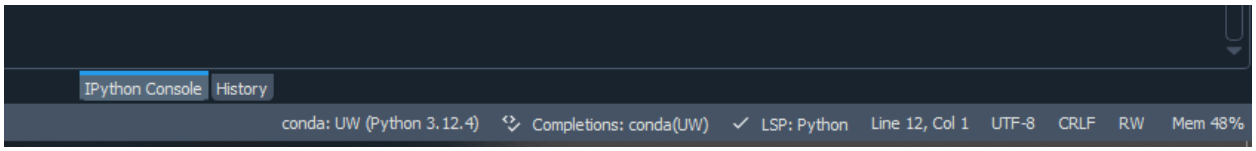
Step 3: Set up Spyder (Recommended IDE)

Spyder provides an integrated development environment optimized for scientific Python workflows.

1. Launch **Anaconda Navigator**.
2. Install and Launch **Spyder**.
3. Configure Spyder to use the UW environment:
    - Go to **Tools → Preferences → Python interpreter**.
    - Select "Use the following Python interpreter".
    - Browse to the Python executable in your UW environment folder (e.g., .../anaconda3/envs/UW/python.exe).
4. Click Apply and OK. Restart Spyder.



5. Verify the Connection (Crucial Step) Before running any scripts, ensure Spyder is actually talking to the UW environment.
    - Look at the IPython Console (usually the bottom-right pane) in Spyder, directly below the console window; this shows the Python executable being used.



---

3. Input Data Requirements

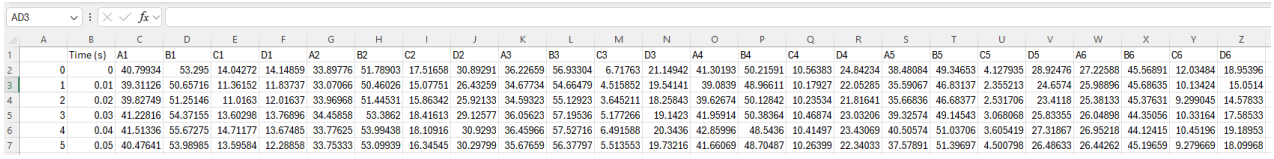
*The pipeline requires **Time vs. Force** data collected from **24-well plates**. While developed using data from Mantarray systems, the scripts are compatible with **any** recording device that outputs data in the following structures.*

3.1. Supported File Formats

You may upload data in either of the following formats. The script automatically detects the file type and standardizes the data structure internally.

Option A: CSV (.csv) - Recommended This is the simplest format, typically exported directly from force analysis instrument

- **Structure:** A simple comma-separated file.
- **Time Column:** Must be a single column named Time (s).
- **Well Columns:** Must use simple well coordinates as headers (e.g., A1, B1, C1…).
- **Note:** This matches the direct output of many controller interface logs.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1		Time (s)	A1	B1	C1	D1	A2	B2	C2	D2	A3	B3	C3	D3	A4	B4	C4	D4	A5	B5	C5	D5	A6	B6	C6	D6
2	0	0	40.79934	53.295	14.04272	14.14859	33.89776	51.78903	17.51658	30.89291	36.22659	56.93304	6.71763	21.14942	41.30193	50.21591	10.56383	24.84234	38.48084	49.34653	4.127935	28.92476	27.22588	45.56891	12.03484	18.95396
3	1	0.01	39.31126	50.65716	11.36152	11.83737	33.07066	50.46026	15.07751	26.43259	34.67734	54.66479	4.515852	19.54141	39.0839	48.96611	10.17927	22.05285	35.59067	46.83137	2.355213	24.6574	25.98896	45.68635	10.13424	15.0514
4	2	0.02	39.82749	51.25146	11.0163	12.01637	33.96968	51.44531	15.86342	25.92133	34.59323	55.12923	3.645211	18.25843	39.62674	50.12842	10.23534	21.81641	35.66836	46.68377	2.531706	23.4118	25.38133	45.37631	9.299045	14.57833
5	3	0.03	41.22816	54.37155	13.60298	13.76896	34.45858	53.3862	18.41613	29.12577	36.05623	57.19536	5.177266	19.1423	41.95914	50.38364	10.46874	23.03206	39.32574	49.14543	3.068068	25.83355	26.04898	44.35056	10.33164	17.58533
6	4	0.04	41.51336	55.67275	14.71177	13.67485	33.77625	53.99438	18.10916	30.9293	36.45966	57.52716	6.491588	20.3436	42.85996	48.5436	10.41497	23.43069	40.50574	51.03706	3.605419	27.31867	26.95218	44.12415	10.45196	19.18953
7	5	0.05	40.47641	53.98985	13.59584	12.28858	33.75333	53.09939	16.34545	30.29799	35.67659	56.37797	5.513553	19.73216	41.66069	48.70487	10.26399	22.34033	37.57891	51.39697	4.500798	26.48633	26.44262	45.19659	9.279669	18.09968

***Option B: Excel (.xlsx)** This is the standard export format multi-sheet Excel formats.*

- **Sheet Name:** The file MUST contain a sheet named exactly: continuous-waveforms.
- **Time Column:** Must be named Time (seconds).
- **Well Columns**: Must be named using the convention: A1 - Active Twitch Force (μN), B1 - Active Twitch Force (μN) etc.

3.2. File naming convention (CRITICAL)

To enable high-throughput batch processing, the scripts rely on a standardized filename structure to sort data by Experimental Day, Plate, and Protocol. You must rename your input files to match this pattern.

Required Pattern: *[AnyText]__[Day]_[PlateName]_[Protocol]_[AnyText].[extension]*

Example*: Export_001__d28_Plate1_Fatigue.csv*

Interpretation Logic: *The script parses the filename by locating the "Day" token (e.g., d7) and reading sequentially:*

1. ***Day:*** *The token starting with 'd' (e.g., d7).*
2. ***Plate Name:*** *The text segment immediately following the Day.*
3. ***Protocol:*** *The text segment immediately following the Plate Name.*

3.3. Plate Map Format (.csv)

You must provide a ***CSV file*** *that maps each well to an experimental condition (e.g., "WT", "KO", "Drug_A").*

- ***Format:*** *A standard grid layout matching a 24-well plate*
- ***Row 1:*** *Header row (can be blank or labels like 1, 2, 3...).*
- ***Column A:*** *Row labels (A, B, C, D...).*
- ***Cells:*** *The condition name for that well.*

Example Plate Map Structure (Excel view):

	A	B	C	D	E	F	G
1		1	2	3	4	5	6
2	A	WT+Rp	DMD+Rp	EF1a-Delta-ABD	EF1a udys-V	EF1a udys p2a	EF1a udys FLAG
3	B	WT+Rp	DMD+Rp	EF1a-Delta-ABD	EF1a udys-V	EF1a udys p2a	EF1a udys FLAG
4	C	WT+Rp	DMD+Rp	EF1a-Delta-ABD	EF1a udys-V	EF1a udys p2a	EF1a udys FLAG
5	D	WT+Rp	DMD+Rp	EF1a-Delta-ABD	EF1a udys-V	EF1a udys p2a	EF1a udys FLAG

- ***Important:*** *Ensure condition names are consistent (e.g., don't use "WT" in one cell and "WildType" in another).*

3.4. Processing Multiple Plates (Batch Mode)

The scripts are capable of processing multiple plates from the same experimental day in a single run. This is highly efficient for batch analysis.

- ***How it works:*** *When selecting input files, you can select multiple Excel files at once (e.g., d7_Plate1_Twitch.xlsx, d7_Plate2_Twitch.xlsx, d21_Plate1_Twitch.xlsx & d21_Plate2_Twitch.xlsx).*
- ***Automatic Pairing:*** *The script uses the filename parser to identify the unique* ***Plate Name*** *for each file. It then processes each file independently but combines all the data into a single final output file (e.g., d7_Twitch_Analysis.xlsx & d21_Twitch_Analysis.xlsx).*
- ***Data Tagging:*** *In the final output, every row of data is tagged with a Plate column so you can distinguish between Plate 1 and Plate 2 results.*

LIMITATION: Single Plate Map

- *Currently, you can only upload* ***ONE*** *plate map CSV file per analysis run.*
- ***Implication:*** *This means* ***all plates in your batch must have the exact same experimental layout.***
- *If your plates have different layouts (e.g., Plate 1 is Drug A, Plate 2 is Drug B), you must run the analysis script separately for each plate so you can select the correct map for each.*

---

4. Protocol Files (JSON)

Reference JSON files are provided to replicate the specific stimulation timings (Delay, Duration, Frequency) used as defaults in these scripts. These timings can be manually replicated on any programmable stimulator.. These files exactly match the "Default" settings in the analysis scripts.

- Twitch 02Hz. Twitch protocol
- Fatigue: The standard 4-block fatigue test with decreasing rest periods.
- 75Hz: Three trains of 75Hz stimulation (0.5s duration) - Great for kinetics on tetanic contractions as these produce near maximal force with clear defined/sharp peak.
- FF: A stepped protocol from 1Hz to 100Hz.
- Priming: A warm-up protocol designed to stimulate the muscle prior to data recording. This helps stabilize the tissue, preventing artifacts such as artificially slow relaxation or inflated peaks often seen in the first few contractions of a cold muscle.

- *Twitch 1Hz**:** Requires editing the Twitch Analysis Script configuration. (See Section 10).*

To use: Either upload the relevant JSON file (if supported by your hardware) or manually program these parameters during experiment setup. Matching these timings ensures the script's peak detection windows align with your data.

---

5. Overview of Analysis Scripts & Peak Detection Strategies

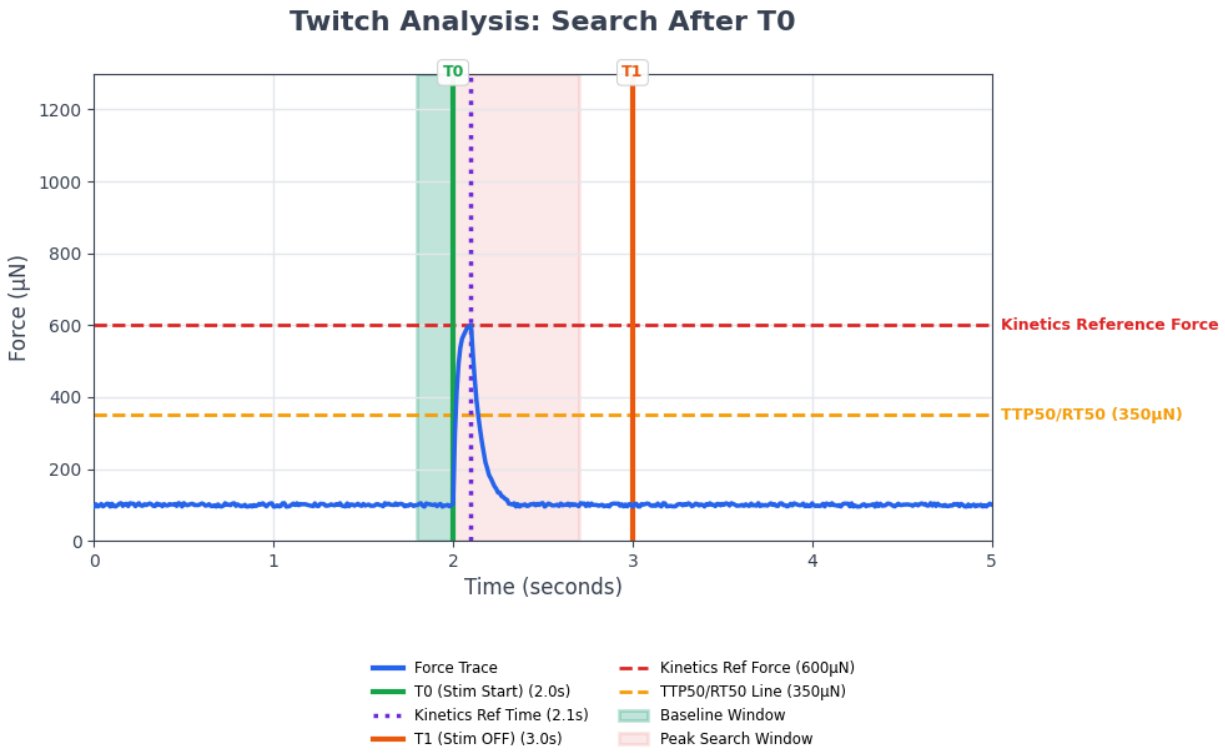
Each script is tailored for specific experimental designs and employs a distinct method for identifying peaks.

Key Definitions for Peak Detection:

- **T0 (Time 0, Stimulation Start):** The time point when an electrical stimulation pulse begins.
- **T1 (Time 1, Stimulation End):** The time point when an electrical stimulation pulse ends.
    - T1 is also the name of twitch 1 in the fatigue and twitch protocols, this will be amended in the future versions.
- **Enhanced Detection Algorithm:** All scripts use a robust algorithm for identifying when a force trace crosses a kinetic threshold (e.g., for R50 or TTP90). It requires **3 consecutive data points** to cross the threshold and incorporates a small (~2%) tolerance to guard against signal noise. All metrics are calculated from the smoothed force trace with baseline subtracted.

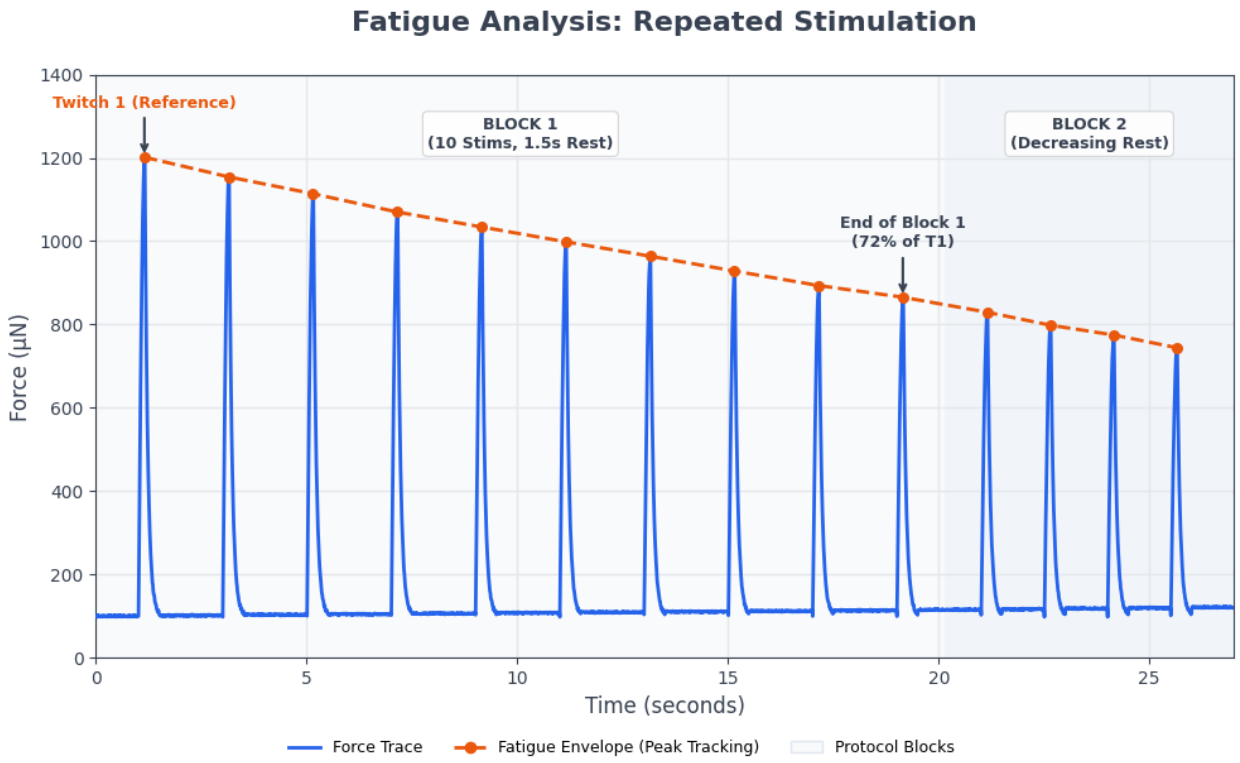
5.1 Twitch Analysis (Twitch Script v1.2.1.py)

- **Purpose:** Designed for analyzing low-frequency twitch protocols. It focuses on detailed kinetics of individual twitches, baseline drift, and fatigue metrics.
- **Peak Detection Strategy: Search After T0**
    - The script uses T0 as the anchor and searches for the highest force value within a defined window (default 0.7s) that occurs **after** T0. All kinetics are calculated relative to T0 and this dynamically determined kinetics reference peak time.



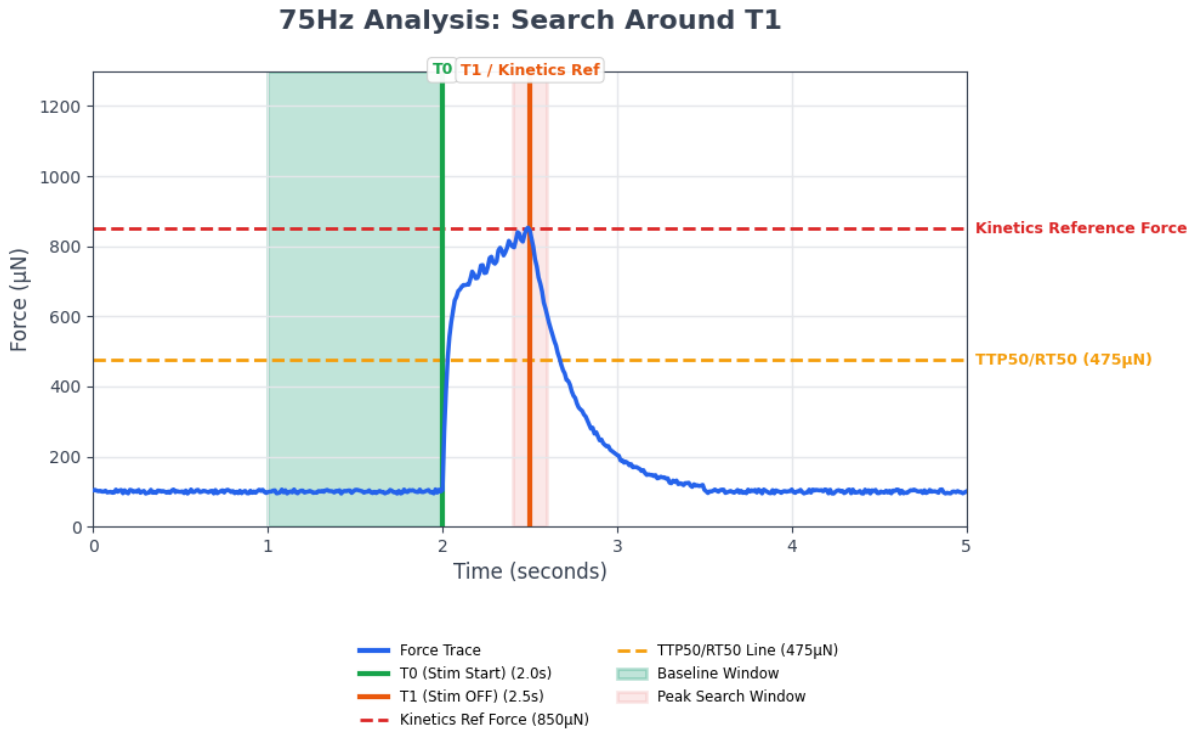
5.2 Fatigue Analysis (Fatigue Script v1.1.2.py)

- **Purpose:** Specialized for a specific 4-block fatigue protocol with decreasing rest periods. The primary goal is to accurately track force decline, baseline changes, and calculate the **Fatigue Resistance Index (FRI)**.
- **Peak Detection Strategy: Search After T0**
    - Similar to the Twitch script, it uses T0 as an anchor for each stimulation and searches for the highest force afterward. This ensures the true, diminishing peak is captured for each contraction even as the muscle fatigues.



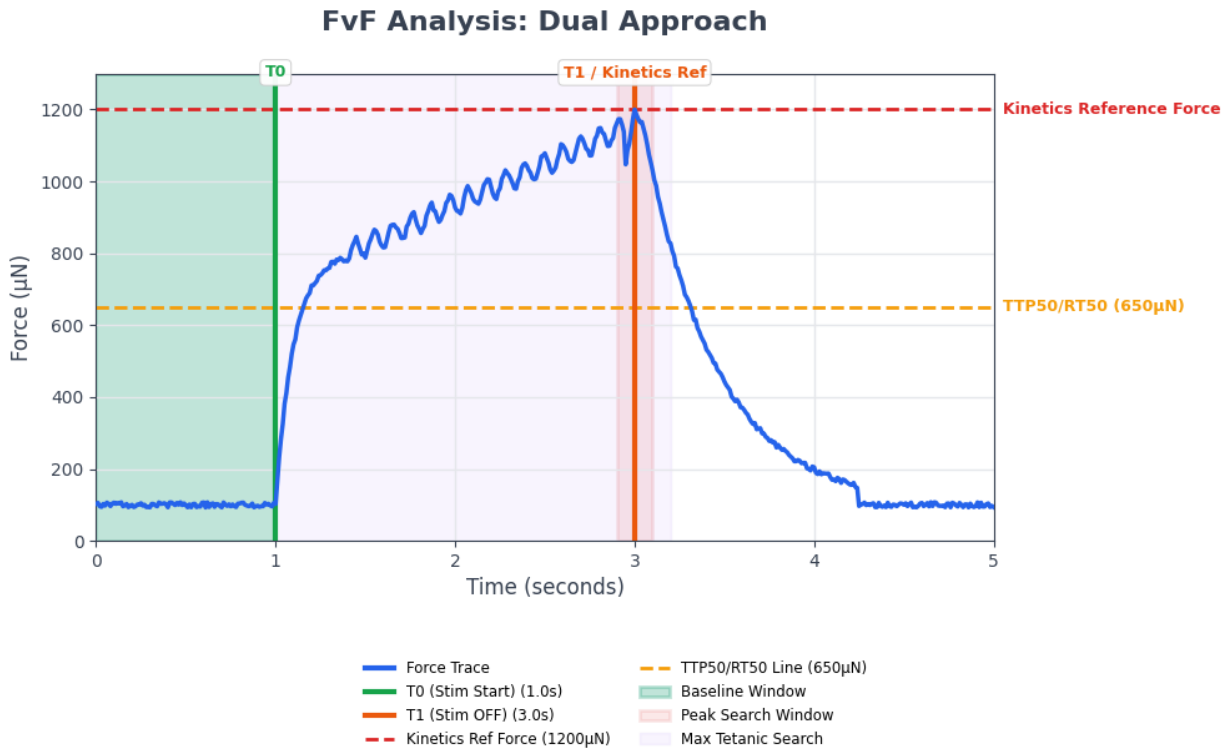
5.3 Continuous Waveform (75Hz) Analysis (75Hz Script v1.7.0.py)

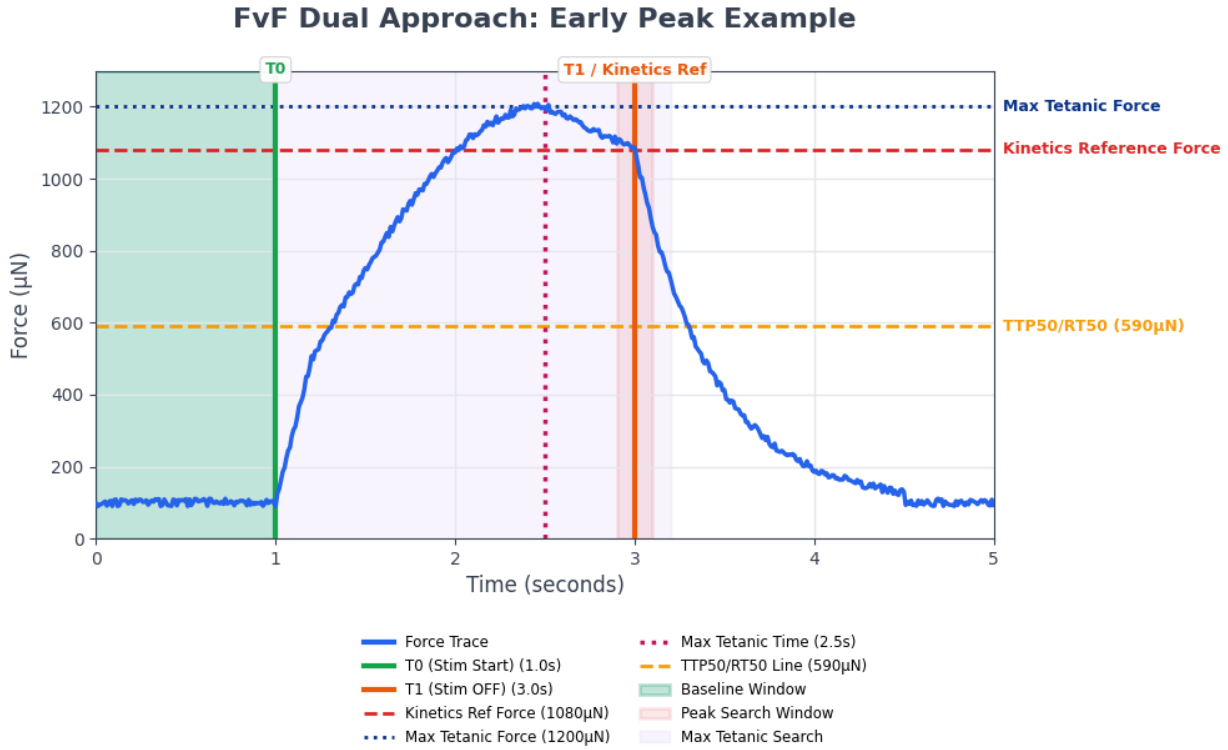
- **Purpose:** Optimized for higher-frequency tetanic stimulations (e.g., 75Hz) where individual twitches fuse into a sustained contraction.
- **Peak Detection Strategy: Dual Approach**
    1. **Kinetics Peak Force:** Searches a narrow window (±100ms) around the protocol-defined T1. This peak is used as the reference time for all relaxation kinetics (R10-R90).
    2. **Max Tetanic Force:** Searches a broader window from T0 to T1+200ms to find the absolute maximum strength of the contraction.
- **Timing:** Contraction kinetics (TT50P, TTP90) are timed from T0. Relaxation kinetics are timed from the **Kinetics Peak**.



5.4 Force-Frequency (FvF) Analysis (Force Frequency Script v1.0.5.py)

- **Purpose:** Specifically built for Force-vs-Frequency relationship experiments. Assigns Hz to a stimulation from the hardcoded parameters.
- **Peak Detection Strategy: Dual Approach**
    1. **Kinetics Peak Force:** It uses the "Search Around T1" method, finding a peak in a ±100ms window around T1 to ensure consistent kinetic calculations.
    2. **Max Tetanic Force:** It performs a separate, broader search for the "Max Tetanic Force" within a window from T0 up to T1+200ms. This ensures the absolute highest force achieved during the sustained stimulation is captured for the FvF curve.


FvF Analysis: Dual Approach
T0
T1 / Kinetics Ref
Kinetics Reference Force
TTP50/RT50 (650µN)
Force (µN)
Time (seconds)
0
200
400
600
800
1000
1200
0
1
2
3
4
5
Force Trace
T0 (Stim Start) (1.0s)
T1 (Stim OFF) (3.0s)
Kinetics Ref Force (1200µN)
TTP50/RT50 Line (650µN)
Baseline Window
Peak Search Window
Max Tetanic Search


FvF Dual Approach: Early Peak Example
T0
T1 / Kinetics Ref
Max Tetanic Force
Kinetics Reference Force
TTP50/RT50 (590µN)
Force (µN)
Time (seconds)
0
200
400
600
800
1000
1200
0
1
2
3
4
5
Force Trace
T0 (Stim Start) (1.0s)
T1 (Stim OFF) (3.0s)
Kinetics Ref Force (1080µN)
Max Tetanic Force (1200µN)
Max Tetanic Time (2.5s)
TTP50/RT50 Line (590µN)
Baseline Window
Peak Search Window
Max Tetanic Search

6. When to Use Which Script: A Guide to Peak Detection

If your primary goal is...	And your protocol involves...	Use this script:	Because its peak detection...
Analyzing individual twitch kinetics.	Single, well-separated pulses (e.g., 1Hz).	**Twitch Script**	Searches for peak **after T0**.
Tracking force decline during fatigue.	Multiple blocks of repeated stimulations.	**Fatigue Script**	Searches for peak **after T0** for each stimulus.
Analyzing kinetics of tetanic contractions.	Short (0.5s) high-frequency trains.	**75Hz Script**	Searches for **Kinetics Peak near T1** (timing) and **Max Tetanic** (strength).
Determining force-frequency relationships.	A series of increasing frequencies.	**FvF Script**	Uses **Dual Approach** (Kinetics near T1, Max Strength window).

---

7. Running an Analysis

The workflow is designed for a simple **"Click and Run"** experience.

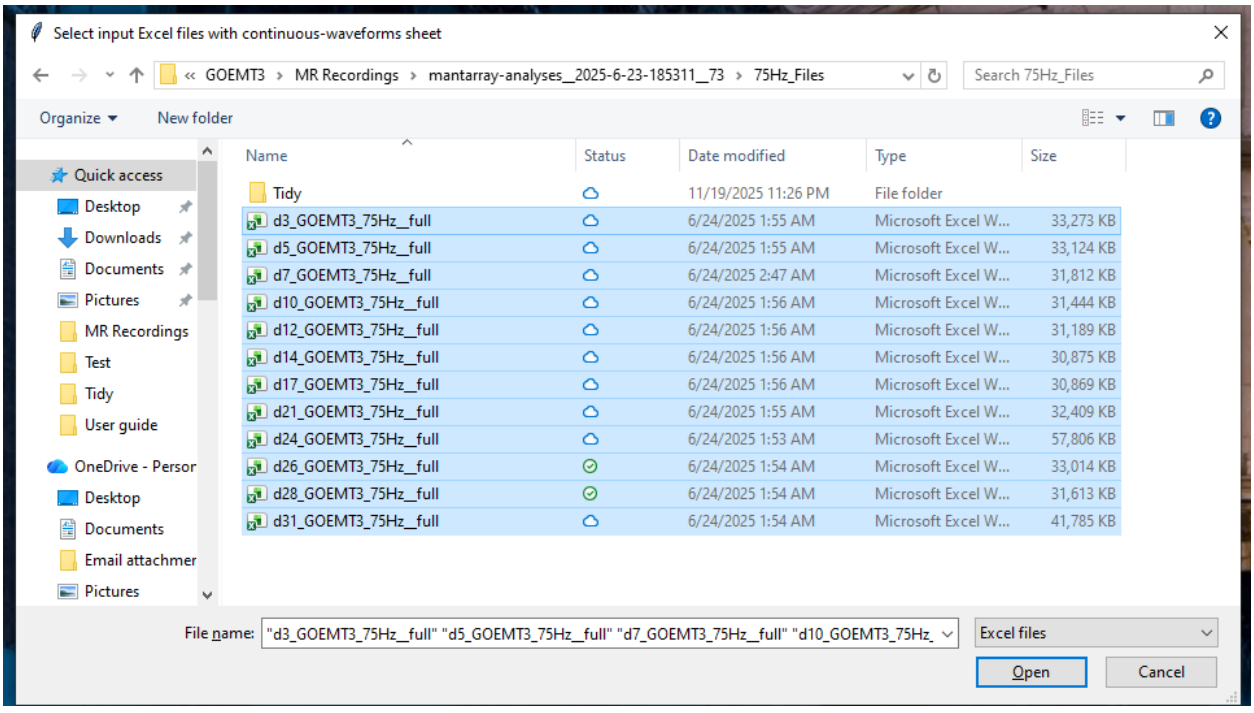
1. **Launch Spyder** with your UW environment active.
2. **Open** the appropriate Python script (.py file).
3. **Check Configuration:** If your protocol timings (Stim Duration, Number of Stims, Max Time) differ from the default, you must edit the **CONFIGURATION SECTION** at the very top of the script *before* running it.

```
# ========================================
#        CONFIGURATION SECTION
# ========================================

# Recording Settings
MAX_RECORDING_TIME = 350.0    # Maximum time (seconds) to process. Trims data beyond this point.

# FvF Protocol Settings (Hard-coded from JSON)
INITIAL_DELAY = 21.0          # Initial delay before first stimulation
STIM_DURATION = 2.0           # Duration of each stimulation
INTER_STIM_DELAY = 16.0       # Delay between stimulations
BASELINE_WINDOW = 1.0         # Window before T0 to calculate baseline
FVF_PROTOCOL_HZ = [1, 2, 3, 5, 10, 15, 20, 30, 40, 60, 80, 100]  # Hz sequence

# ========================================
#      END CONFIGURATION SECTION
# ========================================
```

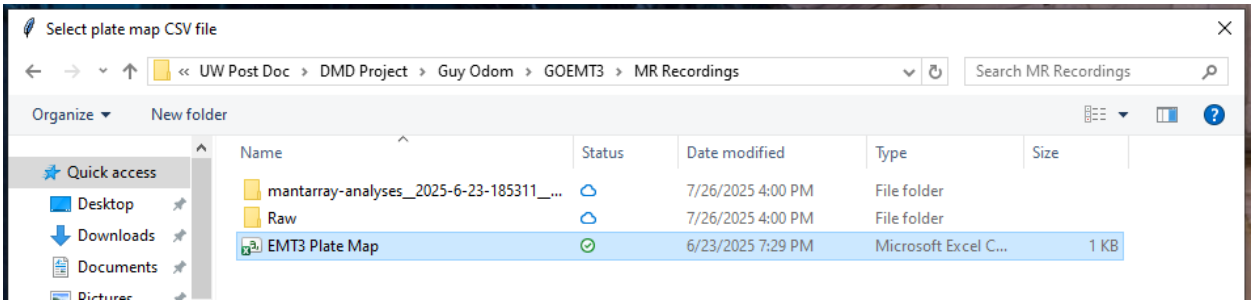
4. **Run** the script (F5).

5. **Select Input Files (CSV or xlsx):** A file dialog will open. Select **one or more** data files.

    - **Batch Processing:** You can hold Ctrl (Windows) or Cmd (Mac) to click and select multiple files (e.g., d7_Plate1.CSV and d19_Plate2.CSV) to analyze them together.
    - *Note:* The script will automatically parse the plate name from the filename (the text immediately following the day, e.g., d7_PlateName_...).

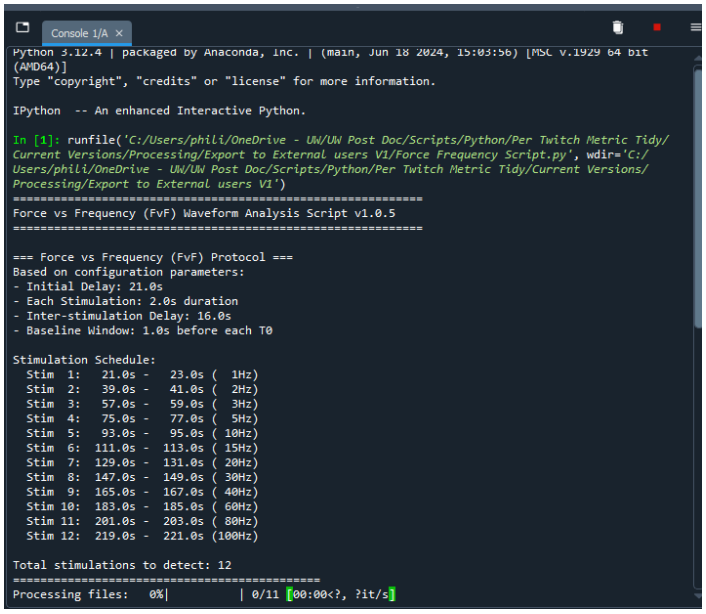


6. **Select Plate Map (CSV):** A second dialog will prompt you to select your single .csv plate map. This map will be applied to **ALL** files selected in the previous step. Ensure all plates share this layout.



7. **Select Output Directory:** Choose where to save the results.

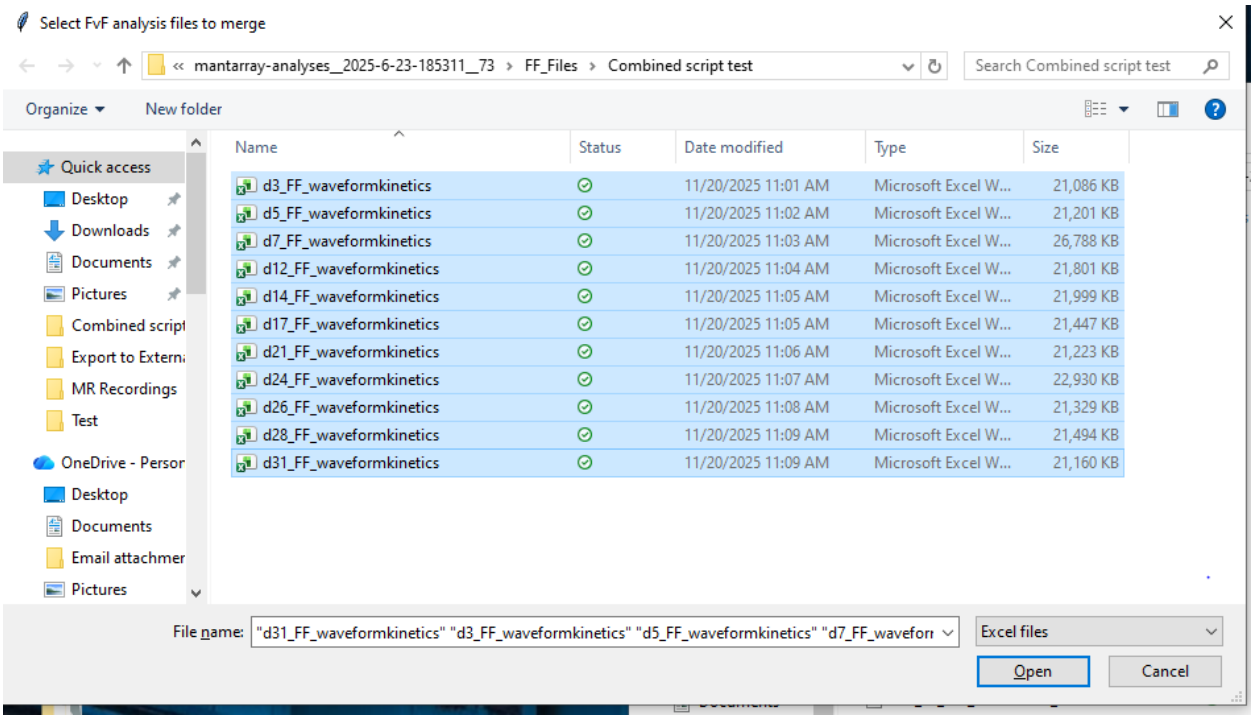
8. **Processing:** The script will process your files without further interruptions (no interactive menus). *Processing may take 5-30 minutes depending on file size.*



9. **Review Outputs (Step 8 below):** Navigate to your output directory to find the generated Excel and Markdown files.

10. **(Optional) Proceed to step 9 below: Post-Processing with Prism Merge Scripts:** Creates master sheet with all day of the experiment in one and formats for easy use with Prism.

    1. **Open** the appropriate Python script (.py file) and run
    2. Select files generated with the analysis scripts



3. Save output location

---

8. Understanding the Outputs

Each script generates a primary Excel file and a Markdown reference document. The Excel file typically contains several sheets:

- **Condition Sheets (e.g., 'WT', 'KO'):** Full time-course data for all wells belonging to a specific condition, including raw force, smoothed force, and all calculated kinetic parameters.
- **(KEY) Summary / Detail Sheet:** This is a critical output, providing a row for each detected peak/stimulation event. It contains all the key calculated metrics.
- **(KEY) Representative Traces:** Shows the *average* waveform data per condition, plotted against time, with standard deviations. Great for graphing the shape of the contraction.
- **Calculation Reference (.md file):** A Markdown file detailing the parameters and methods used for the analysis.

Script-Specific Sheets & Metrics:
Twitch Script (v1.2.1):

- **Contractile Averages:** The primary summary sheet containing averaged metrics per well:
    - **Mean BK Corrected Force (µN):** The average baseline-corrected force (Peak - Baseline) for all valid twitches.
    - **Fatigue Index (Late/Early):** The ratio of the mean force of the last 5 twitches to the first 5 twitches. (<1 indicates fatigue).
    - **Early / Late Twitch Mean (µN):** The specific average force values used to calculate the Fatigue Index.
    - **BK Force CV (%):** Coefficient of Variation, indicating the stability of twitch force across the recording.
- **Relaxation Detail:** A row-by-row breakdown of every single detected twitch:
    - **Fast Peak Warning:** A flag (0 or 1) indicating if a peak occurred too quickly after stimulation (artifact check).
    - **Normalized Force (% of T1):** The force of the current twitch expressed as a percentage of the first valid twitch in that well.
    - **R50/90 Time & TT50P/90 Time:** Individual kinetic timings for every twitch.
- **Baseline Drift Summary:**
    - **Drift Rate (µN/s):** The rate of baseline change over the course of the recording.
    - **Max Drift (µN):** The maximum deviation from the initial baseline observed.

Fatigue Script (v1.1.2):

- **Fatigue Summary:** Contains the primary metrics, positioned in the leftmost columns for easy access:
    - **Fatigue Resistance Index (%):** The mean normalized force across the entire protocol.
    - **FRI Block 1-4 (%):** The fatigue resistance index calculated specifically for each block.
- **Fatigue Detail:** Per-stimulation data including block number and all kinetic parameters.

- **Condition Avg Fatigue:** The average Normalized Force (% of Twitch1) for each stimulation number, grouped by condition.

Continuous (75Hz) Script (v1.7.0):

- **Relaxation Summary:** Contains per-stimulation data with aligned naming conventions:
    - **Max tetanic-BK (µN):** The absolute maximum force found during the entire stimulation window (Strength) minus Baseline.
    - **Kinetics Peak-BK (µN):** The peak force near T1, used for relaxation timing calculations minus Baseline
    - **Peak Time Delta (s):** The time lag between the end of stimulation (T1) and the actual Kinetics Peak.

Force-Frequency Script (v1.0.5):

- **Relaxation Summary:** Contains all per-stimulation data including Hz and:
    - **Max tetanic-BK (µN):** The absolute maximum force found during the entire stimulation window (Strength) minus Baseline.
    - **FF Normalized (%):** Force normalized to the maximum force produced by that specific well- The primary metric for plotting Force-Frequency curves.
    - **TTMFP (s):** Time to Max Force Peak (from T0).
    - **Time Over 90% (%):** A measure of fatigue resistance (percentage of stim duration spent near max force).
    - *Note: Kinetics (R10-90, TT50P) are only calculated for frequencies > 20Hz due to how the peak search window works, it is not compatible with the low frequency, non-fused contractions*.

---

9. Post-Processing with Prism Merge Scripts

After running the primary analysis, use these scripts to combine outputs from multiple days into Prism-friendly formats.

9.1 Twitch Merge.py

- **Purpose:** To merge twitch analysis outputs from multiple days.

- **Inputs:** Reads Contractile Averages and Relaxation Detail sheets.
- **Outputs:** Creates Twitch_Final_Tables.xlsx.

9.2 Fatigue Prism Merge (v1.9.1)

- **Purpose:** To merge fatigue analysis outputs from multiple days.
- **Inputs:** Reads Fatigue Summary (for FRI) and Fatigue Detail (for TimeAt) sheets.
- **Outputs:** Creates Fatigue_Final_Filtered_Tables.xlsx.
- **Tab Organization:** The most important metrics are now shifted to the left:
    1. **Fatigue Resistance Total / Per Block:** Pivoted directly from the Summary sheet.
    2. **TimeAtXX Point:** Calculated from the Detail sheet using linear interpolation.
    3. **Force at Specific Stims:** Pivoted values for Stims 5, 10, 20.
    4. **Curves:** Normalized Force vs Stim / Time.

9.3 75Hz Prism Merge (v1.1.3)

- **Purpose:** To merge 75Hz analysis outputs from multiple days.
- **Inputs:** Reads the Relaxation Summary sheet.
- **Outputs:** Creates Relaxation_75Hz_Final_Tables.xlsx.

9.4 Force Frequency Prism Merge v1.0.py

- **Purpose:** To merge Force-Frequency analysis outputs.
- **Inputs:** Reads the Relaxation Summary sheet.
- **Outputs:** Creates FvF_Final_Filtered_Tables.xlsx with F-F Curves and Kinetics vs. Day (for 100Hz).

9.5 Understanding the Prism Merge Outputs

The Merge scripts are designed to create a master tidy sheet and transform raw "Long/Tidy" data into "Wide/Pivoted" matrices specifically formatted for direct copy-pasting into GraphPad Prism.

General formatting Logic

- **Tidy_Master Sheets:** These contain every single data point in a long list. Use these for archiving or importing into statistical software like R or Python.
- **Pivoted Sheets (Prism Ready):** These are formatted so that **Rows** represent the X-axis (e.g., Day, Hz, Stim #) and **Columns** represent individual Replicates (wells).

A. Twitch Merge Outputs (Twitch_Final_Tables.xlsx)

Designed for Metric vs. Timepoint.

- **Tidy_Twitch_Master:** The full, raw dataset containing all days and all metrics.
- **Metric Sheets (e.g., Mean_BK_Corrected_Force_vs_Day):**
    - **Rows:** Days (d7, d14, etc.).
    - **Columns:** Individual replicates (e.g., WT_Plate1_A1, KO_Plate1_A2).
    - **Prism Usage:** Copy into a **Grouped** table to plot the metric over time.

B. Fatigue Merge Outputs (Fatigue_Final_Filtered_Tables.xlsx)

Designed for a mix of Grouped plots (Bar graphs) and XY plots (Fatigue Curves).

- **Priority Metrics (Leftmost Tabs):**
    - Fatigue_Resistance_Total: The overall FRI % per well vs. Day.
    - TimeAtXX_Point: The interpolated time (seconds) to reach 75% or 50% force vs. Day.
- **Curve Data (Day-Segregated):**
    - NormForce_vs_Stim_dX: Normalized force (Y) vs. Stimulation Number (X). Created as separate sheets for each day (e.g., _d7, _d14).

C. 75Hz Merge Outputs (Relaxation_75Hz_Final_Tables.xlsx)

Designed for plots tracking kinetics over time.

- **Metric Sheets (e.g., R50_Time_vs_Day):**
    - Standard pivot tables (Row=Day, Col=Replicate) for all kinetic parameters.
    - **Timecourse Max:** The bottom rows of these sheets contain a summary of the maximum value found across the entire timeline for quick reference.

D. Force-Frequency Merge Outputs (FvF_Final_Filtered_Tables.xlsx)

Designed for FvF Curves, Max Capacity and relaxation kinetics over time.

- **FvF Curve Sheets (e.g., FF_Normalized_Norm_ vs_Freq_d28):**
    - **Rows:** Frequency (1Hz to 100Hz).
    - **Columns:** Replicates.
    - **Prism Usage:** Copy into XY tables. Fit a sigmoidal curve and interpolate metrics such as F50 and F75 (Frequency required to reach 50% or 75% max force, a metric of EC coupling efficiency.
- **100Hz Summary Sheets (e.g., Max_tetanic-bk_force):**
    - Isolates the 100Hz data point (Max Capacity) and plots it against Day.

---

10. Customizing Analysis Parameters

All scripts (Twitch, Fatigue, 75Hz, FvF) now feature a dedicated **CONFIGURATION SECTION** at the very top of the code. You do not need to be a coder to change these.

How to Edit:

1. Open the script in Spyder.
2. Look for the block between the === CONFIGURATION SECTION === headers.
3. Change the numbers to the right of the = sign.
4. **Save (Ctrl+S)** before running.

General Parameters (All Scripts):

- **MAX_RECORDING_TIME:** Set the cutoff time (seconds) for analysis. Data after this time is ignored.
- **INITIAL_DELAY:** Time (seconds) when the *first* stimulation occurs.
- **STIM_DURATION / REST_DURATION:** Timing of the electrical pulses.

Twitch Specific Settings (Twitch Script)

Stimulation Timing (Hz Control): To change the pacing frequency (e.g., from the default 0.2Hz to 1Hz), you must adjust the timing variables.

For 0.2Hz (Default - 5s interval):

- *INITIAL_DELAY = 4.97*
- *STIM_DURATION = 0.01*
- *REST_DURATION = 4.99*

For 1Hz (1s interval):

- *INITIAL_DELAY = 2.97*
- *STIM_DURATION = 0.01*
- *REST_DURATION = 0.99*

Artifact Rejection Settings:

- **MINIMUM_TWITCH_THRESHOLD_UN:** Sets the minimum baseline-corrected force (μN) required for a twitch to be considered valid. Peaks below this value are ignored for kinetic calculations.
- **FAST_PEAK_WARNING_S:** Sets the time threshold (seconds from T0) for flagging "Fast Peaks." Peaks occurring faster than this value are flagged as potential artifacts.

Force-Frequency Specific Settings (FvF Script): To change the frequencies assigned to each stimulation step, you must edit the **FVF_PROTOCOL_HZ** list.

- **Format:** A list of numbers separated by commas, enclosed in square brackets [].
- **How to edit:** Simply change the numbers, add new ones, or remove them to match your specific protocol.

Example:

- **Default:** FVF_PROTOCOL_HZ = [1, 2, 3, 5, 10, 15, 20, 30, 40, 60, 80, 100]
- **Modified:** FVF_PROTOCOL_HZ = [10, 20, 40, 80]

  CRITICAL WARNING: The script applies these labels sequentially (1st stim gets the 1st Hz in the list, 2nd stim gets the 2nd Hz, etc.). You **MUST** ensure this list exactly matches the order of frequencies programmed into your JSON protocol. If they do not match, your output data will be mislabeled.

---

11. Troubleshooting

- **Environment Problems (ModuleNotFoundError):** Ensure your UW conda environment is activated before launching Spyder.
- **File/Sheet Not Found:** Double-check that all Excel files contain a sheet named exactly continuous-waveforms.
- **Processing Errors:** Check data integrity (clean numerical data), plate map consistency, and review the console for specific error messages.
- **Output Issues:** Review the .md reference file generated with the analysis to check exactly what parameters were used.

---

12. Getting Help

If you encounter problems, contact the developer:

Philip Barrett

Email: pb92@uw.edu

When reporting an issue, please include the script name, the exact error message, and a brief description of what you were doing.