# ENGR 298: Engineering Analysis and Decision Making – Finding Data in Noise

Dr. Jason Forsyth
Department of Engineering
James Madison University

# Identifying Signals of Interest

- Previous work with EKG utilized an existing algorithm

- What happens when there is not a standard approach? How can we write code to isolate events of interest in a given time-series set.

- Will focus on calculating the Reactive Strength Index (RSI) from Drop-Jump exercises. Later, pull data from tensile test.

# Reactive Strength Index (RSI)

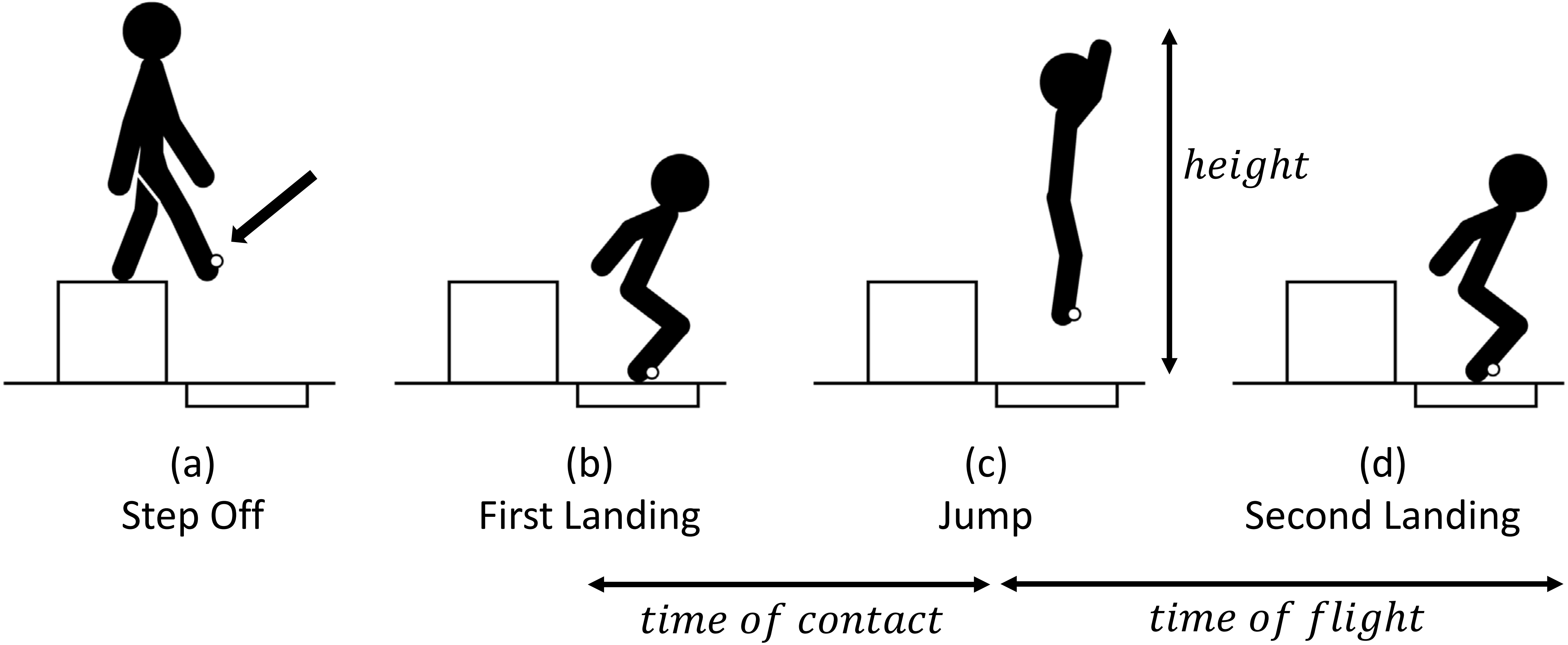
- Measures an athlete’s capacity for explosive movement. Utilized in training regimes to determine fatigue levels and/or improvement.

- Typically measured by force plates in research laboratories; bulky, immobile, and expensive.

- Can RSI be determined with lower-cost equipment such as an inertial measurement unit? Will compare RSI measurements during a Drop Jump test with force plates and IMUs to assess accuracy.

# Drop Jump Test

$$\text{RSI} = \frac{h}{t_c} = \frac{g * {t_f}^2}{8 * t_c}$$



$height$

(a)
Step Off

(b)
First Landing

(c)
Jump

(d)
Second Landing

$time\ of\ contact$

$time\ of\ flight$

# Calculating RSI

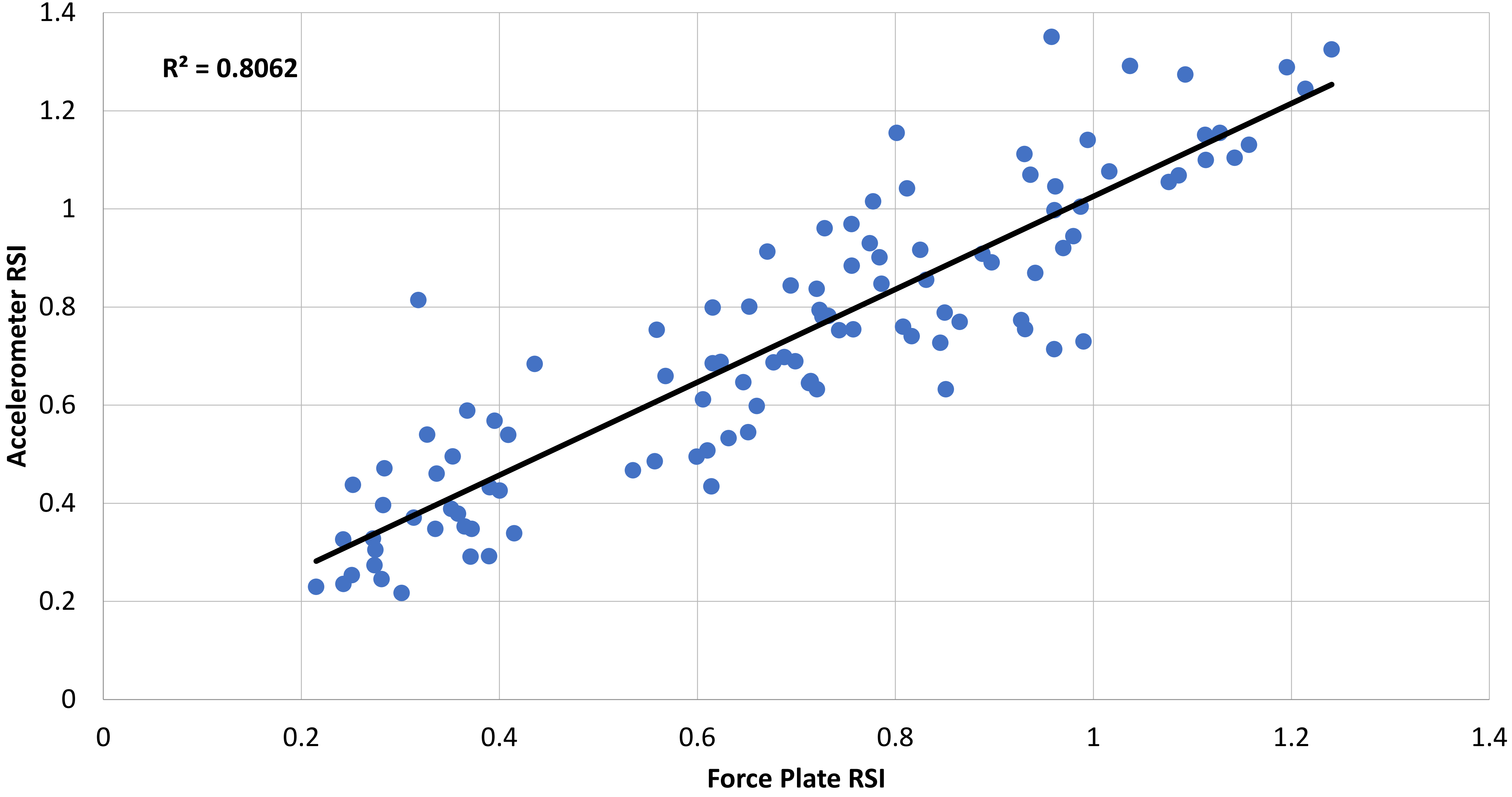
- Once the various landing and take-offs points have been identified, then the remainder is just comparison.

- **Time of Contact** $(t_c)$= jump point – first landing
- **Time of Flight** $(t_f)$ = second landing – jump point

- **RSI** =$\frac{g * {t_f}^2}{8 * t_c}$ where g is local gravitational acceleration.

# Experimental Setup

- Attach Inertial Measurement Unit (IMU) to person's foot. Stream acceleration (g's) at 800Hz.
- Monitor forces (Netwons) on plate at 1000Hz
- Calculate points for first/second landing and take-off from both IMU and Force Plate. Compare results to see if "sufficiently" accurate.


R² = 0.8062
Accelerometer RSI
Force Plate RSI
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
0.2
0.4
0.6
0.8
1
1.2
1.4

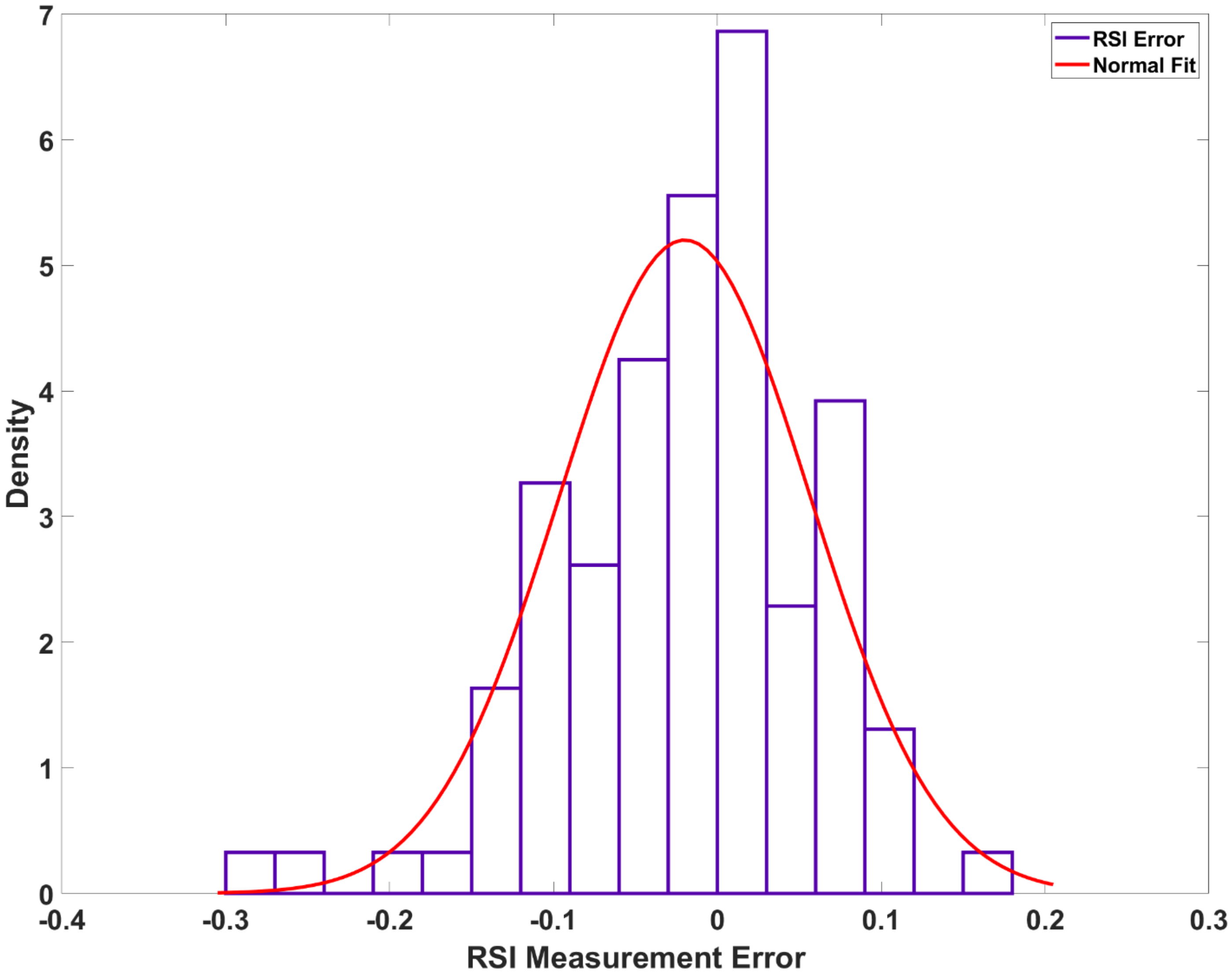
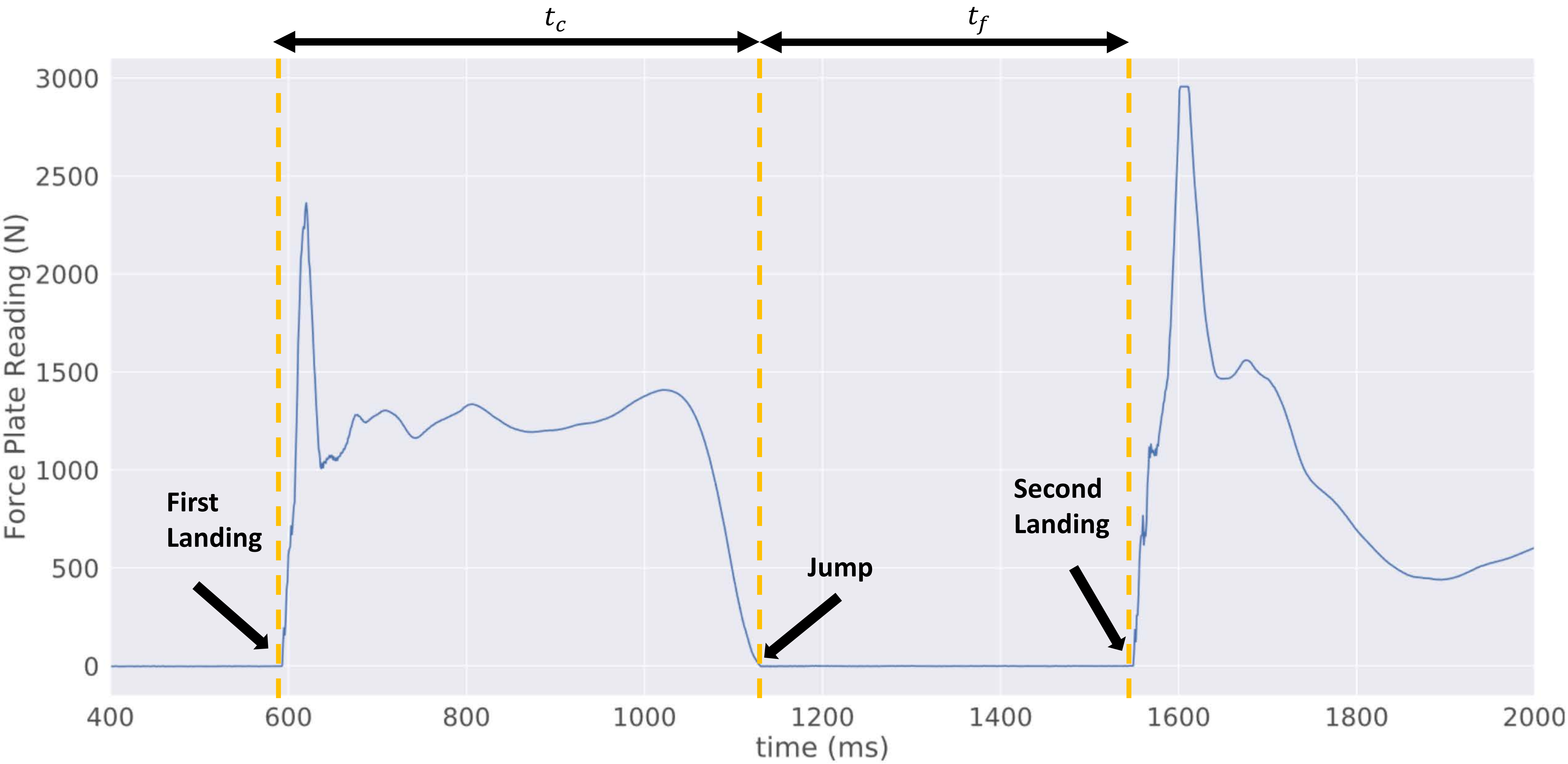
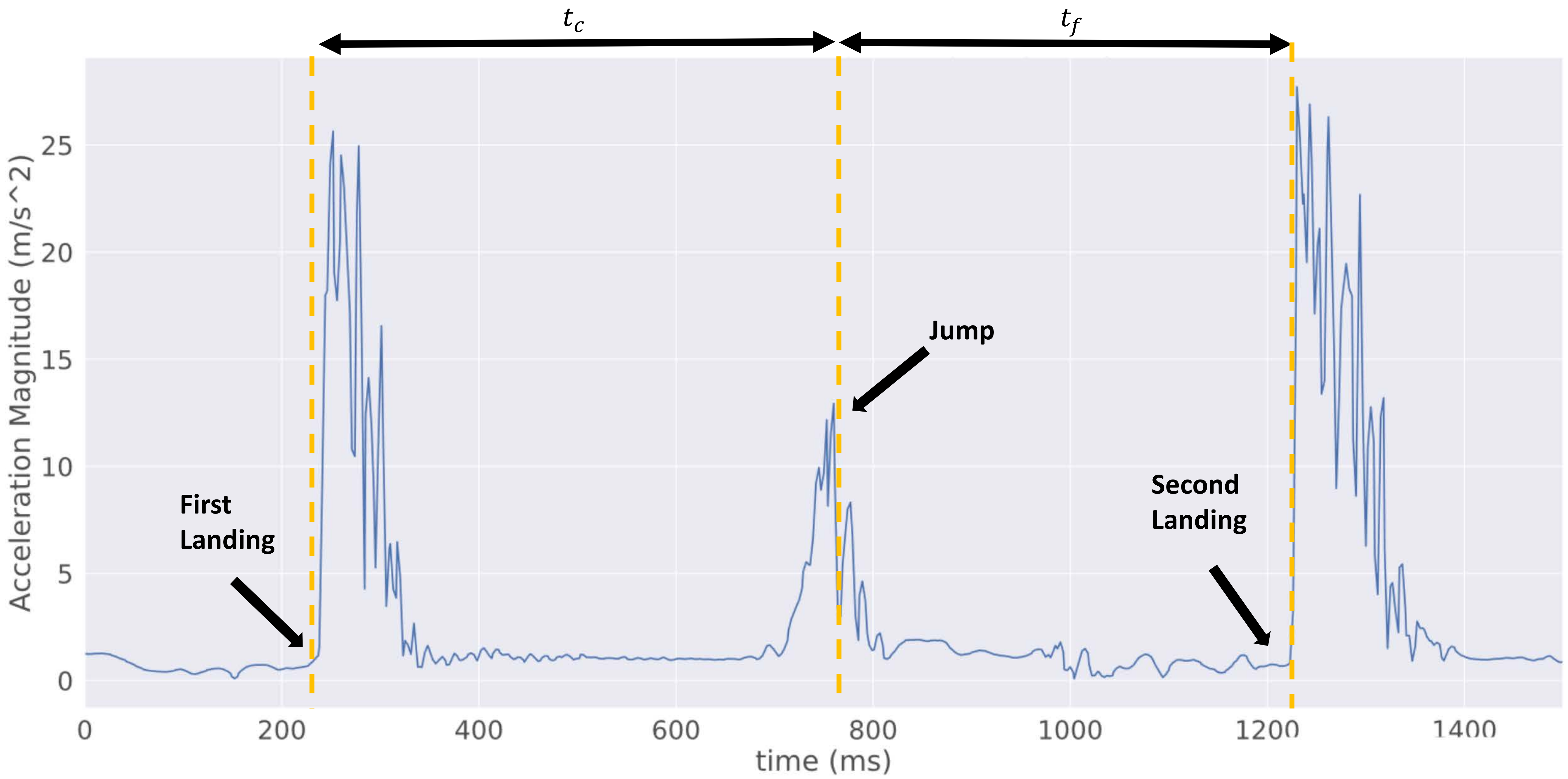


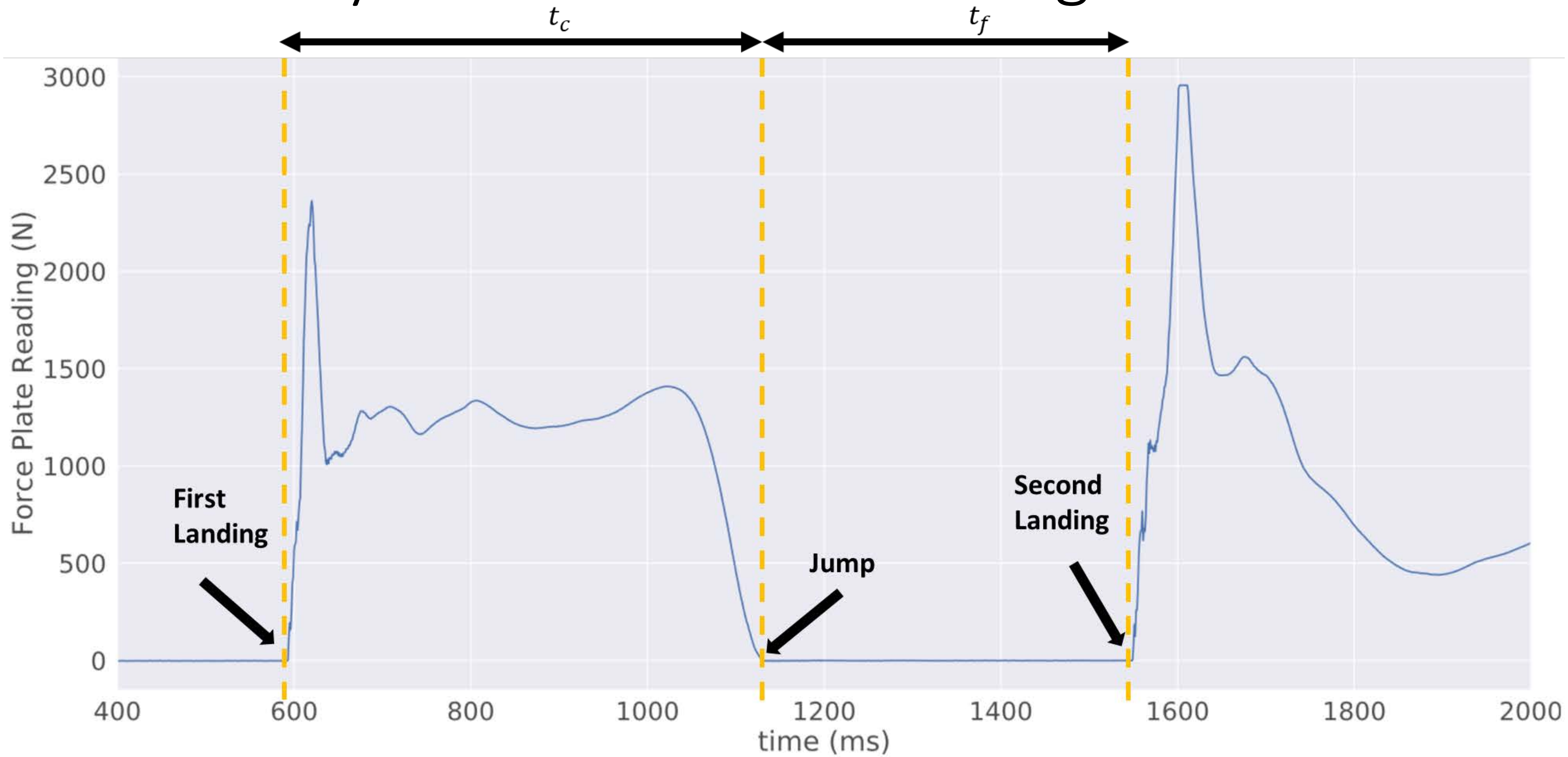
Fig. 4: Distribution of measurement error with fitted normal distribution



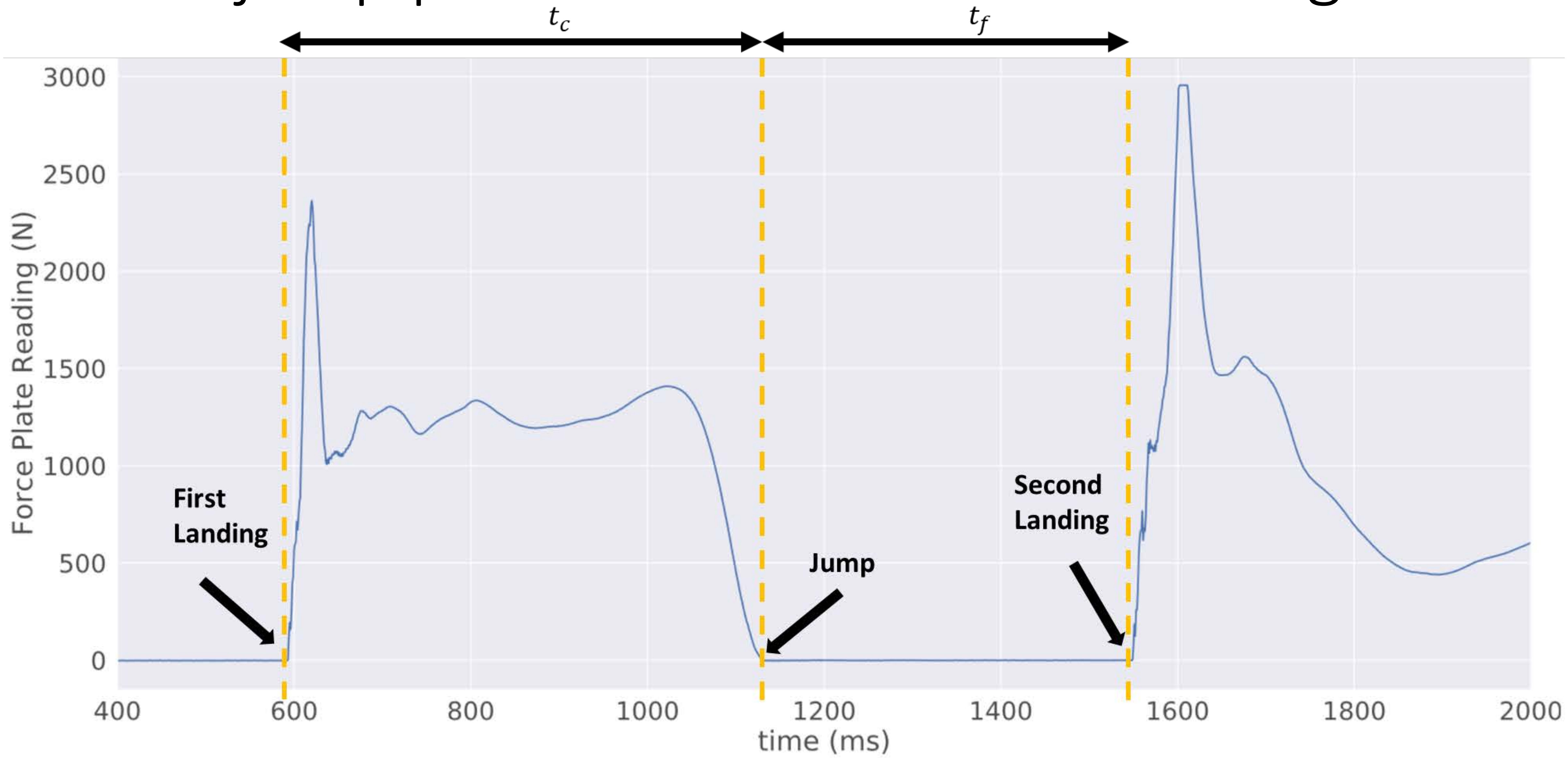
$t_c$
$t_f$
3000
2500
2000
1500
1000
500
0
Force Plate Reading (N)
First Landing
Jump
Second Landing
400
600
800
1000
1200
1400
1600
1800
2000
time (ms)


$t_c$
$t_f$
Jump
First Landing
Second Landing
Acceleration Magnitude (m/s^2)
0
5
10
15
20
25
time (ms)
0
200
400
600
800
1000
1200
1400

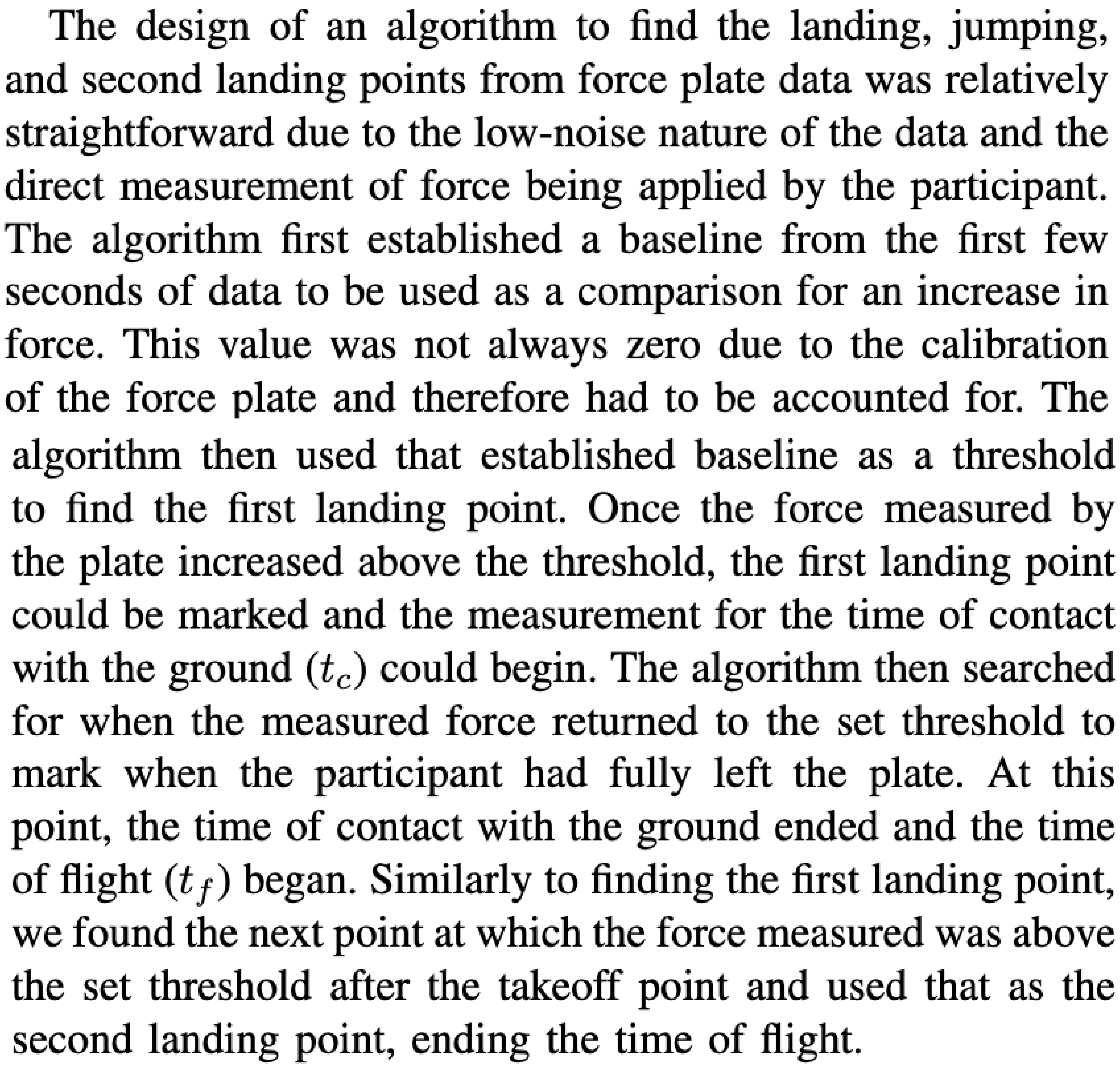
# Imagine this data was a list/array in Python. How would you find the first landing?



# If you knew the first landing, how would you find the jump point and then second landing?



# We need to translate this algorithm into something that we can realize in Python.



The design of an algorithm to find the landing, jumping, and second landing points from force plate data was relatively straightforward due to the low-noise nature of the data and the direct measurement of force being applied by the participant. The algorithm first established a baseline from the first few seconds of data to be used as a comparison for an increase in force. This value was not always zero due to the calibration of the force plate and therefore had to be accounted for. The algorithm then used that established baseline as a threshold to find the first landing point. Once the force measured by the plate increased above the threshold, the first landing point could be marked and the measurement for the time of contact with the ground ($t_c$) could begin. The algorithm then searched for when the measured force returned to the set threshold to mark when the participant had fully left the plate. At this point, the time of contact with the ground ended and the time of flight ($t_f$) began. Similarly to finding the first landing point, we found the next point at which the force measured was above the set threshold after the takeoff point and used that as the second landing point, ending the time of flight.

# Force Plate Algorithm for First Landing, Take Off, and Second Landing

1. Establish **baseline** measurement for no contact on plate.
2. Once force measurement rises above the baseline by some threshold, the user has contacted the plate. Consider that point the time of **first landing**.
3. When force measurements return to the initial baseline the user has left the plate. Consider this the **take off** point.
4. The plate should remain near baseline while the user is in the air (there is no load). Once it rises above the baseline again, the user has landed. Consider this the **second landing**.
5. Calculate *time of contact ($t_c$)* and *time of flight ($t_f$)* based upon these points.
6. Calculate RSI from $t_c$ and $t_f$.

Imagine that all the force plate data is loaded from a file and provided to you as a Numpy array or List. What else do you need?

# What variables, methods, packages, loops/conditions might be needed….

1. Establish **baseline** measurement for no contact on plate.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| | | | | |

# What variables, methods, packages, loops/conditions might be needed….

1. Establish **baseline** measurement for no contact on plate.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| **Baseline** (the average value over some period of time)<br>**Duration** (how long to take that average) | Average() | Numpy | None | List of force plate data point |

# What variables, methods, packages, loops/conditions might be needed….

2. Once force measurement rises above the baseline by some threshold, the user has contacted the plate. Consider that point the time of **first landing**.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| | | | | |

# What variables, methods, packages, loops/conditions might be needed….

2. Once force measurement rises above the baseline by some threshold, the user has contacted the plate. Consider that point the time of **first landing**.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| **Measurement** (whatever the current force plate measurement is)<br>**Threshold** (some value that if the measurement rises above we consider it the landing)<br>**First Landing** (index in the list of the first landing point) | Comparison? | None | FOR loop to iterate through list<br>IF statement to perform comparison | A list containing all force plate data points. |

# What variables, methods, packages, loops/conditions might be needed….

3. When force measurements return to the initial baseline the user has left the plate. Consider this the **take off point**.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| | | | | |

# What variables, methods, packages, loops/conditions might be needed….

3. When force measurements return to the initial baseline the user has left the plate. Consider this the **take off point**.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| **Measurement** (whatever the current force plate measurement is)<br>**Threshold** (some value that if the measurement rises above we consider it the landing)<br>**Take off point** (index in the list of the take off point) | Comparison? | None | FOR loop to iterate through list<br>IF statement to perform comparison | A list containing all force plate data points. |

# What variables, methods, packages, loops/conditions might be needed….

4. The plate should remain near baseline while the user is in the air (there is no load). Once it rises above the baseline again, the user has landed. Consider this the **second landing**.

| Variables | Functions / Methods | Packages | Loops / Conditionals | Data Structures |
|---|---|---|---|---|
| **Baseline** (the average value over some period of time)<br>**Measurement** (whatever the current force plate measurement is)<br>**Threshold** (some value above the baseline)<br>**Second Landing** (index in the force plate data that is the second landing) | Comparison? | None | FOR loop to iterate through list<br>IF statement to perform comparison | A list containing all force plate data points. |

With all the take-off and landing points identified, then can directly calculate RSI.

# Assignments This Week

- Will focus on Force Plate data (less noise and easier to accomplish).

- An example/template is provided showing how find the first landing point. Read algorithm. Run it. Try out on different files to see how it performs.

- **Deliverable**: extend example and complete template to identify jump off point and second landing. Use results to calculate RSI.

- Hint: nothing more than FOR loops and IF statements are required.

- **Future Deliverable**: will assign Acceleration data as culminating assignment in coming week(s).