### Motion Analysis Methods:

**Sampling, Fourier Analysis and Filtering**

---

#### Eng315  Lab 67  Winter, 2019

**Video Analysis and Filtering**

<table>
<thead>
<tr>
<th>Name</th>
<th>Partner(s)</th>
<th>Grade</th>
</tr>
</thead>
</table>

**Introduction**

The goals of this lab is to obtain a position-time dataset from video analysis and create a Butterworth filter to obtain useful velocity and acceleration graphs.

**Objectives**

- Understand the basics of video digitization.
- Obtain useful 3D data from a video clip.
- Understand how digital filtering can enhance the quality of results.

**Equipment Provided**

- Computer with appropriate software.

**References**

- Video digitization and digital filtering textbooks and web sites.

**Procedure**

1. On the class website under the heading of Lab 67, you will find a video file showing a side view of a human vertical jump with markers indicating various joint centers, including at the hip. Two forms are provided: original .mp4 and new .mp4. Download the .mp4 video file and save it to your lab computer's S: drive. Open the .mp4 video file and ensure that joint centers are visible through each frame.
Motion Analysis Methods:
Sampling, Fourier Analysis and Filtering

- Ground Reaction Force vs Time
- Counter Motion Vertical Jump

- Vertical Jump - 1200 Hz
Proper Sampling:

"If you can exactly reconstruct the analog signal from the samples, you must have done the sampling properly."

(from Digital Signal Processing by Steven W. Smith, p. 39)
Analysis Methods for Position-Time Data:

First Central Difference formulas:

Velocity: \( V_i = \frac{X_{i+1} - X_{i-1}}{t_{i+1} - t_{i-1}} \)

Acceleration: \( a_i = \frac{X_{i+1} - 2X_i + X_{i-1}}{(t_{i+1} - t_i)^2} \)
Why does taking the derivative of position-time data seem to amplify the noise in a signal?

**Low Frequency Signal (f = 1 Hz)**

\[
f(t) = 100 \sin(2 \pi f t)
\]

\[
f'(t) = 200 \pi f \cos(2 \pi f t)
\]

\[
f''(t) = -400 \pi^2 f^2 \sin(2 \pi f t)
\]

**High Frequency Noise (f = 10 Hz)**

\[
f_n(t) = 1 \sin(2 \pi f t)
\]

\[
f'_n(t) = 2 \pi f \cos(2 \pi f t)
\]

\[
f''_n(t) = -4 \pi^2 f^2 \sin(2 \pi f t)
\]

**Signal to Noise Ratio:**

- 100 : 1
- 10 : 1
- 1 : 1
Why does taking the derivative of position-time data seem to amplify the noise in a signal?

How can one smooth out a signal to minimize problems with velocity and acceleration calculations?
Butterworth filtering:

\[ X'_i = a_0 X_i + a_1 X_{i-1} + a_2 X_{i-2} + b_1 X'_{i-1} + b_2 X'_{i-2} \]

\( X \) = raw data points
\( X' \) = previously filtered data points (recursion)

\[ \omega_c = \tan \left( \frac{\pi f_c}{f_i} \right) \]

\( K_1 = \sqrt{2} \omega_c \) for a Butterworth filter,

or, \( 2 \omega_c \) for a critically damped filter

\[ K_2 = \omega_c^2, \quad a_0 = \frac{K_2}{(1 + K_1 + K_2)} \quad a_1 = 2a_0, \quad a_2 = a_0 \]

\[ K_3 = \frac{2a_0}{K_2}, \quad b_1 = -2a_0 + K_3 \]

\[ b_2 = 1 - 2a_0 - K_3, \quad \text{or,} \quad b_2 = 1 - a_0 - a_1 - a_2 - b_1 \]
Butterworth Low-Pass Filter is easy to implement in Excel:

<table>
<thead>
<tr>
<th>f_s [Hz]</th>
<th>a_0</th>
<th>a_1</th>
<th>a_2</th>
<th>b_1</th>
<th>b_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.29289</td>
<td>0.58579</td>
<td>0.29289</td>
<td>0.00000</td>
<td>-0.1715</td>
</tr>
<tr>
<td>5</td>
<td>0.20657</td>
<td>0.41314</td>
<td>0.20657</td>
<td>0.36953</td>
<td>-0.19582</td>
</tr>
<tr>
<td>6</td>
<td>0.15505</td>
<td>0.31010</td>
<td>0.15505</td>
<td>0.62021</td>
<td>-0.24041</td>
</tr>
<tr>
<td>7</td>
<td>0.12123</td>
<td>0.24247</td>
<td>0.12123</td>
<td>0.80303</td>
<td>-0.28796</td>
</tr>
<tr>
<td>8</td>
<td>0.09763</td>
<td>0.19526</td>
<td>0.09763</td>
<td>0.94281</td>
<td>-0.33333</td>
</tr>
<tr>
<td>9</td>
<td>0.08042</td>
<td>0.16085</td>
<td>0.08042</td>
<td>1.05333</td>
<td>-0.37502</td>
</tr>
<tr>
<td>10</td>
<td>0.06746</td>
<td>0.13491</td>
<td>0.06746</td>
<td>1.14298</td>
<td>-0.41290</td>
</tr>
<tr>
<td>11</td>
<td>0.05742</td>
<td>0.11484</td>
<td>0.05742</td>
<td>1.21719</td>
<td>-0.44687</td>
</tr>
<tr>
<td>12</td>
<td>0.04949</td>
<td>0.09898</td>
<td>0.04949</td>
<td>1.27963</td>
<td>-0.47759</td>
</tr>
<tr>
<td>13</td>
<td>0.04311</td>
<td>0.08621</td>
<td>0.04311</td>
<td>1.33291</td>
<td>-0.50333</td>
</tr>
<tr>
<td>14</td>
<td>0.03789</td>
<td>0.07578</td>
<td>0.03789</td>
<td>1.37889</td>
<td>-0.53045</td>
</tr>
<tr>
<td>15</td>
<td>0.03357</td>
<td>0.06714</td>
<td>0.03357</td>
<td>1.41896</td>
<td>-0.55327</td>
</tr>
<tr>
<td>16</td>
<td>0.02995</td>
<td>0.05991</td>
<td>0.02995</td>
<td>1.45424</td>
<td>-0.57406</td>
</tr>
<tr>
<td>17</td>
<td>0.02689</td>
<td>0.05379</td>
<td>0.02689</td>
<td>1.48550</td>
<td>-0.59307</td>
</tr>
<tr>
<td>18</td>
<td>0.02428</td>
<td>0.04856</td>
<td>0.02428</td>
<td>1.51338</td>
<td>-0.61051</td>
</tr>
<tr>
<td>19</td>
<td>0.02205</td>
<td>0.04407</td>
<td>0.02205</td>
<td>1.53842</td>
<td>-0.62655</td>
</tr>
<tr>
<td>20</td>
<td>0.02008</td>
<td>0.04017</td>
<td>0.02008</td>
<td>1.56102</td>
<td>-0.64135</td>
</tr>
<tr>
<td>21</td>
<td>0.01838</td>
<td>0.03676</td>
<td>0.01838</td>
<td>1.58152</td>
<td>-0.65505</td>
</tr>
<tr>
<td>22</td>
<td>0.01689</td>
<td>0.03378</td>
<td>0.01689</td>
<td>1.60020</td>
<td>-0.66776</td>
</tr>
<tr>
<td>23</td>
<td>0.01557</td>
<td>0.03114</td>
<td>0.01557</td>
<td>1.61730</td>
<td>-0.67958</td>
</tr>
<tr>
<td>24</td>
<td>0.01440</td>
<td>0.02880</td>
<td>0.01440</td>
<td>1.63299</td>
<td>-0.69060</td>
</tr>
<tr>
<td>25</td>
<td>0.01336</td>
<td>0.02672</td>
<td>0.01336</td>
<td>1.64746</td>
<td>-0.70090</td>
</tr>
</tbody>
</table>
How to do simple 2D motion analysis?

LoggerPro
LoggerPro Software can be used to analyze video images: