An Analysis of Jolts Experienced on Amtrak Railways
Trips we took

A.k.a. public transport where we scared unsuspecting commuters with our devices
Specific Railways Studied

1. Illinois Service Line between Chicago & Champaign
   - Multiple trips with three or more DAQ devices present
   - Primary route; most detailed analysis

2. Italian Intercity Trains
   - For comparison with American railways

3. Hiawatha Line between Chicago & Milwaukee, Wi.
The State of Railways in the United States

- **Amtrak**: 2,142 railway cars and 425 locomotives
- 21,400 miles within the contiguous United States & Southern Canada.
- Recent years: reputation for being rougher, less efficient and slower than international counterparts (i.e. European, East Asian railways).
- Conditions of rails: below what American Society of Civil Engineers considers optimal.
- Current annual budget just enough to keep rails safe
THE ARDUINO

- Arduino MEGA 2560
  - Features 256 kilobytes of volatile flash memory
  - 8 kilobytes of RAM
- Built in 16 channel 10 bit Analog-to-Digital Converter (ADC)
- Hardware Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit communication (I2C) protocol
Ultimate GPS

- Can 22 satellites
- -165 dBm sensitivity, 10 Hz updates, 66 channels
- 5V friendly design and only 20mA current draw
- Breadboard friendly + two mounting holes
- Built-in datalogging
- PPS output on fix
- Internal patch antenna + u.FL connector for external active antenna
LSM9DS1

9 DOF sensor:

- **3-axis accelerometer**
  - Measures gravity
  - Informs the user how fast the board is accelerating in 3D space

- **3-axis magnetometer**
  - Detects which direction the magnetic north lies
    - Done by sensing where the strongest magnetic force is coming from

- **3-axis gyroscope**
  - Uses Earth's gravity to measure spin and twist
  - Ultimately help determine orientation.
**DS3231 RTC**

- Precision clock
- Coin cell on the back of the sensor
  - The user to take years of data even if the main power is lost.
- Synchronizes time read outs
  - Matches GPS data with position data
- Contains an extremely accurate internal crystal
  - Accounts for drifting caused by external crystals

**INA219**

- A high side DC current sensor
- A precision amplifier that measures up to ±3.2 Amps
- Used for power-monitoring related problems
- Uses I2C to measure both the high side voltage and DC current draw
  - 1% precision
Breadboarded & PCB Versions of DAQ
writing to data001.txt
Software

- Arduino Data Acquisition Code
- Python Processing
  - Repairing Data
  - Calculating Data
  - Averaging Data
Arduino Data Acquisition

- Automatically updating filenames
- Using Keypad for multiple functions
  - Electrical information
  - File Closing
  - Current Filename
  - The Magic Conch --->
- GPS function
  - And subsequent issues
- Writing to file
  - Inserting NaNs when appropriate
# The -Python Sequence-

<table>
<thead>
<tr>
<th>Processing Phase</th>
<th>Operation(s)</th>
<th>Filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Repair GPS data</td>
<td>GPS_repair.py</td>
</tr>
<tr>
<td>II</td>
<td>Organize data into arrays, perform minor calculations, &amp; prepare for mapping in Tableau</td>
<td>TrainPy3.py</td>
</tr>
<tr>
<td>III</td>
<td>Calculations of time-averaged acceleration &amp; jerk for use in jolt analysis</td>
<td>GPS_calculations.py</td>
</tr>
</tbody>
</table>

Table 4: Brief overview of the processing pipeline through which all data collected during both test-runs and Amtrak rides passes. **Note:** the filename of step II includes an integer corresponding to the latest version of the program.
GPS -troubles-

Error in GPS Parsing Conversions caused this pattern to appear in our maps.

This is why we needed a repair script.
Repairing

- Our GPS gives out values in DDMM.MMMM, so the minutes are in decimals.
- Issue was stripping zeros from minute string when converting from string to integer. (ex: 4000.0063 would show up in data as 4000.63)
- Split string at period, and count digits that remain in the second one
- If less than 4, add appropriate number of zeros to string
- Write to output file

Us, when we saw the first map that didn’t look like a glitch in the simulation
Arrays & Calculations

- Initially read in data with pandas, also used numpy
- Had to convert DDMM.MMMM values to only degrees (1)
- Located indices where GPS had values.
  - Used this to apply a mask to other arrays to visualize raw data and interpolate (2)
- Interpolate GPS coordinates using original and masked millis arrays (2)
- Multiplied longitude by -1
- Subtract 1 from z acceleration array
- Output to CSV, ready to plot and analyze

```python
def degrees(coordinate):
    coordinate_mod = coordinate % 100
    coordinate_floor = coordinate // 100
    coordinate_degrees = (coordinate_mod / 60) + coordinate_floor
    return coordinate_degrees

longitude = degrees(longitude_raw)
latitude = degrees(latitude_raw)
```

```python
lat_index = np.array([index for index, value in enumerate(latitude_raw) if np.isnan(value) == False])
millis_lat = millis[lat_index]
lat_int = np.interp(millis, millis_lat, latitude_stripped)
```
Final Averaging and Output

- Large amount of small-timestep acceleration points didn’t show us the bigger picture
- Decided to average over a quarter second to get better idea of rough patches in track (1)
- Inserted “jerk” calculation
  - Difference of subsequent acceleration averages (2)
- Output all data to final CSV, ready for plotting and analyzation of bigger picture
DAQ Synchronization

Acceleration data along x-axis measured on three different boards.
DAQ Synchronization

Acceleration data along x-axis measured on three different boards.

Millis vs. Avg, acc x A, Avg, acc x C and Avg, acc x D. Color shows details about Avg, acc x A, Avg, acc x C and Avg, acc x D.
Test-Runs - Champaign-Urbana MTD

Sample Acceleration Profile for a DAQ Suspened By Springs (MTD Bus)
Test-Runs - Champaign-Urbana MTD

Fourier Transform: $a_{\text{net}}$ from MTD Ride w/ Board Suspended from Spring

$f = 0.45 \pm 0.005 \,[\text{Hz}]$
Test-Runs - Champaign-Urbana MTD
Test-Runs - Chicago Metra trains

BNSF Outbound from Chicago to Clarendon Hills, IL.
2:40pm - 3:25pm
No antenna.

Map based on Longitude and Latitude. Color shows Millis.
Test-Runs - Chicago Metra trains

Metra BNSF Inbound (3/16/2019)
Repaired GPS Data
Board D

Map based on Longitude and Latitude. Color shows Millis. Size shows Acc.

<table>
<thead>
<tr>
<th>Net-Acceleration [g]</th>
<th>Millis</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0436</td>
<td>1,626</td>
</tr>
<tr>
<td>0.2000</td>
<td>2,428,578</td>
</tr>
<tr>
<td>0.4000</td>
<td>1,626</td>
</tr>
<tr>
<td>0.6000</td>
<td>2,428,578</td>
</tr>
<tr>
<td>0.8734</td>
<td>1,626</td>
</tr>
</tbody>
</table>
Test-Runs - Chicago Metra trains

Fourier Transform: $a_{net}$ for Metra Chicago (Inbound)

$f = 0.24 \pm 0.0001 \, [Hz]$  
$f = 1.29 \pm 0.03 \, [Hz]$  

Fourier Transform: $a_{net}$ for Metra Chicago (Outbound)

$f = 0.24 \pm 0.001 \, [Hz]$  
$f = 2.14 \pm 0.05 \, [Hz]$
Data Analysis - Characterizing Jolts/Lurches

- Analyze both acceleration & jerk to characterize severity of jolts
- Overall ‘roughness’ of ride characterized w/ RMS of net-acceleration

\[ \bar{J} = \frac{d\vec{a}}{dt} = \frac{d^2\vec{v}}{dt^2} = \frac{d^3\vec{r}}{dt^3} \]

\[ a_{RMS} = \sqrt{\langle a_{net} \rangle^2} = \sqrt{\frac{1}{N} \sum_{i}^{N} (a_{x,i}^2 + a_{y,i}^2 + a_{z,i}^2)} \]
Data Analysis - Characterizing Jolts/Lurches

- Net-acceleration calculated by adding respective x-y-z components in quadrature
- Extremely important to subtract-off the constant 1 [g] of acceleration measured in the z-direction due to Earth’s gravity
- Dilution; washes-out the jolts we are actually interested in

\[ a_{\text{net}} = \sqrt{a_{\hat{x}}^2 + a_{\hat{y}}^2 + a_{\hat{z}}^2} \]
Net Acceleration (All Data, Full Track)

- Indicates a scheduled stop.

Map based on Longitude and Latitude. Color shows A Net (G). Details are shown for Table Name.
Italian Railways

Florence to Rome

Map based on Longitude and Latitude. Color shows Jerk.
Italian Railways

Scaled graphs
Acceleration: 0-1g
Time: 1000k-4000k millis

