Using High Resolution Transit Geolocation Data to Analyze Performance and Identify Intersections that Contribute to Transit Delay

Tal Green, P.E.
San Francisco Municipal Transportation Agency
Tal.Green@sfmta.com
June 25, 2019
Presentation Outline

- Title breakdown!
- SF system overview
- Current transit metrics
- Need for high res data analysis
- Data collection and methodology overview
- Metric #1 – transit signal delay
- Metric #2 - % of buses stopped at intersection
- Next steps
Title Breakdown

Using High Resolution Transit Geolocation Data to Analyze Performance and Identify Intersections that Contribute to Transit Delay
Title Breakdown

Using **High Resolution** Transit Geolocation Data to Analyze Performance and Identify Intersections that Contribute to Transit Delay
Title Breakdown

Using High Resolution Transit Geolocation Data to Analyze Performance and Identify Intersections that Contribute to Transit Delay
Title Breakdown

Using High Resolution Transit Geolocation Data to **Analyze Performance** and Identify Intersections that Contribute to Transit Delay
Title Breakdown

Using High Resolution Transit Geolocation Data to Analyze Performance and **Identify** Intersections that Contribute to Transit Delay
Title Breakdown

Using High Resolution Transit Geolocation Data to Analyze Performance and Identify Intersections that Contribute to Transit Delay
System Overview

- 1200+ signals (~1000 serve Muni)
- 700K rides per day
- Transit First policy
- Transit and Traffic Engineering work together
Currently Available Metrics

**Mode Based On Time Performance**

**Stop Level Dwell Times**

**Timepoint to Timepoint Travel Time**

**Route Based On Time Performance**

**Route Level Ridership**
Data Collection Effort

- On board equipment collects data
  - Approximately 500 buses equipped
- Daily data dump in yard over WiFi
- Weekly transmission of all data (~4 GB)
  - Began collecting data in November, 2018
- Data processing occurs weekly
  - Currently on local machine
  - Future: through IT in a production data warehouse

2.7 million GPS points from one day of data
TSP Equipment on Bus
Methodology Overview
Metric #1 – Transit Signal Delay

• Processing
  • Isolated data to individual trips
  • Matched trip to schedule
  • Determined points of interest, matched to nearest

• Consumable data
  • Pushed to Tableau server
  • Can be pivoted by route, direction, location, time of day, etc.

<table>
<thead>
<tr>
<th>Intersection Name</th>
<th>Median Delay (sec)</th>
<th>Number of Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>25th St &amp; Mission</td>
<td>4</td>
<td>372</td>
</tr>
<tr>
<td>24th St &amp; Mission</td>
<td>14</td>
<td>380</td>
</tr>
<tr>
<td>23rd St &amp; Mission</td>
<td>11</td>
<td>367</td>
</tr>
<tr>
<td>22nd St &amp; Mission</td>
<td>19</td>
<td>365</td>
</tr>
<tr>
<td>21st St &amp; Mission</td>
<td>2</td>
<td>250</td>
</tr>
<tr>
<td>20th St &amp; Mission</td>
<td>3</td>
<td>347</td>
</tr>
<tr>
<td>19th St &amp; Mission</td>
<td>16</td>
<td>343</td>
</tr>
<tr>
<td>18th St &amp; Mission</td>
<td>20</td>
<td>346</td>
</tr>
<tr>
<td>17th St &amp; Mission</td>
<td>2</td>
<td>343</td>
</tr>
<tr>
<td>16th St &amp; Mission</td>
<td>3</td>
<td>339</td>
</tr>
<tr>
<td>15th St &amp; Mission</td>
<td>7</td>
<td>329</td>
</tr>
<tr>
<td>14th St &amp; Mission</td>
<td>17</td>
<td>332</td>
</tr>
</tbody>
</table>

Analysis – Tabulated and mapped delay information by TOD, direction, route
Metric #2 - % of Buses Stopped at Intersection

- Processing
  - Isolated data to individual trips
  - Matched trip to schedule
  - Draw approach zones for signals
  - Determine if buses traveled <2MPH
- Consumable data
  - Pushed to Tableau server
  - Can be pivoted by route, direction, location, time of day, etc.

Analysis – Tabulated and mapped bus stopped % information by TOD, direction, route

<table>
<thead>
<tr>
<th>Intersection Name</th>
<th>% Stopped</th>
<th>Number of Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>25th St &amp; Mission</td>
<td>29%</td>
<td>400</td>
</tr>
<tr>
<td>24th St &amp; Mission</td>
<td>47%</td>
<td>408</td>
</tr>
<tr>
<td>23rd St &amp; Mission</td>
<td>41%</td>
<td>415</td>
</tr>
<tr>
<td>22nd St &amp; Mission</td>
<td>69%</td>
<td>379</td>
</tr>
<tr>
<td>21st St &amp; Mission</td>
<td>5%</td>
<td>374</td>
</tr>
<tr>
<td>20th St &amp; Mission</td>
<td>14%</td>
<td>378</td>
</tr>
<tr>
<td>19th St &amp; Mission</td>
<td>55%</td>
<td>396</td>
</tr>
<tr>
<td>18th St &amp; Mission</td>
<td>70%</td>
<td>394</td>
</tr>
<tr>
<td>17th St &amp; Mission</td>
<td>4%</td>
<td>377</td>
</tr>
<tr>
<td>16th St &amp; Mission</td>
<td>33%</td>
<td>375</td>
</tr>
<tr>
<td>15th St &amp; Mission</td>
<td>34%</td>
<td>380</td>
</tr>
<tr>
<td>14th St &amp; Mission</td>
<td>59%</td>
<td>362</td>
</tr>
</tbody>
</table>

Processing – define and utilize approach zones
For More Details...

# During Daylight Saving (comment out if not) - March to November
allPoints[\'TimestampPacific\'] = allPoints[\'TimestampNew\'] - pd.Timedelta(
    # Process details: Conference compendium
    # Code: Contact me
    # Calculate delay
    mergedData[\"delay\"] = mergedData[\"travelTime\"] - mergedData[\"travelTime\"]
    # Flag out max outliers (True if > 10 minutes to get through intersection)
    mergedData[\"maxOutlier\"] = (mergedData[\"travelTime\"] > 10)

    # Drop extra data
    mergedData.drop([\"FOCSN\"], axis=1, inplace=True)

    # resultsDF = pd.concat([resultsDF,mergedData])

    return mergedData

except:
    print("ERROR!")
    # Add code here to better track errors (create list and add to list)
    if progressBar (chunkCounter, totalChunks, currentSegment, totalSegments, progress=0):
        newProgress = int(100 * float(currentSegment)/float(totalSegments))
        if newProgress >= |progress|:
            clear_output()
        print("Progress: ", newProgress, ")"complete ",currentSegment,"of",totalSegments,"trips analyzed", newProgress
    return progress

# LOOK INTO THIS: at some point, the order of the columns changes!
# Display all points
display(allPoints.tail())

# Loop through each zone to get minimums in each process
for i, zone in allZones.iterrows():
    try:
        pointsInZone = allPoints.loc[(allPoints[\'tripID\'])]
        (allPoints[\'tripID\'])
        (allPoints[\'tripID\']

    # Determine the minimum value in each zone
    minPoints = pointsInZone.loc[pointsInZone]

    # True if stopped, False if didn't stop
    minPoints[\'stopped\'] = minPoints[\"speed\"]

    # Add values from current Zone series to each record
    minPoints[\'zoneID\'] = zone[\"zoneID\"]
    minPoints[\"CNN\"] = zone[\"CNN\"]
    minPoints[\"intName\"] = zone[\"intName\"]
    minPoints[\"intSequence\"] = zone[\"intSequence\"]
    minPoints[\"includesNearsideStop\"] = zone[\"includesNearsideStop\"]

# Display all points
display(allPoints.tail())
Next Steps

- Conduct before/after studies
- Productionalize processing efforts
- Include intersections with nearside stops
- Address poor GPS readings in NE due to Urban canyon effect
Summary

• Title breakdown!
• SF system overview
• Current performance metrics
• Need for high res data analysis
• Data collection and methodology overview
• Metric #1 – transit signal delay
• Metric #2 - % of buses stopped at intersection
• Next steps
Acknowledgement

Ian Martin
Student Intern
Summers 2018, 2019
Questions?