GMNS Workshop
Modeling Mobility Conference

2025 September 14

Agenda

1. What's the Problem?
2. Network Wrangler
   a. What is it?
   b. History
3. GMNS+
   a. What is it?
   b. History
4. What's next?
5. Walkthrough: Create a San Francisco network and …
   a. Network Wrangler - apply projects
   b. GMNS - accessibility viz

If this material is already familiar to you, please feel free to view/try the workshop walkthroughs:

https://colab.research.google.com/drive/1eUXGbDmos_bnzXYhs-vgPhd28lnxdeTF?usp=drive_link

aka https://tiny.cc/SF_netwrangler

What's the Problem?

As a travel modeler at a transportation planning agency, I need to be able to build a base year network and a series of networks for future scenarios (Plan 2025, 2035, 2040, 2045, 2050) that are:

- Built on top of a baseline network representing what's on the ground today
- Flexible and transparent, so can:
  - Easily see when a project gets built
  - Move projects between build years
  - Update a project and have it reflected in all years
  - Easily understand what projects are included in any given network

What's the Problem?

As an academic researcher in transportation and optimization, I need results and algorithms that are reproducible and easily shared with other researchers or partner agencies.

- Aligned with open science principles
- Reproducible research enables:
  - Verification and validation of results (traffic assignment, vehicle routing, choice models, etc.)
  - Transfer of research into real-world applications
  - Systems for benchmarking of new approaches
- The key are accessible, shareable, universally understood datasets & algorithms

The Challenge

- Reproducibility gap: Computer Science >70% vs. Transportation <8%
- Reliance on limited datasets (e.g., TNTP for assignment, NGSIM for traffic flow)
- Comparisons of accessibility or demand forecasts across regions difficult due to lack of data standardization

What is Network Wrangler?

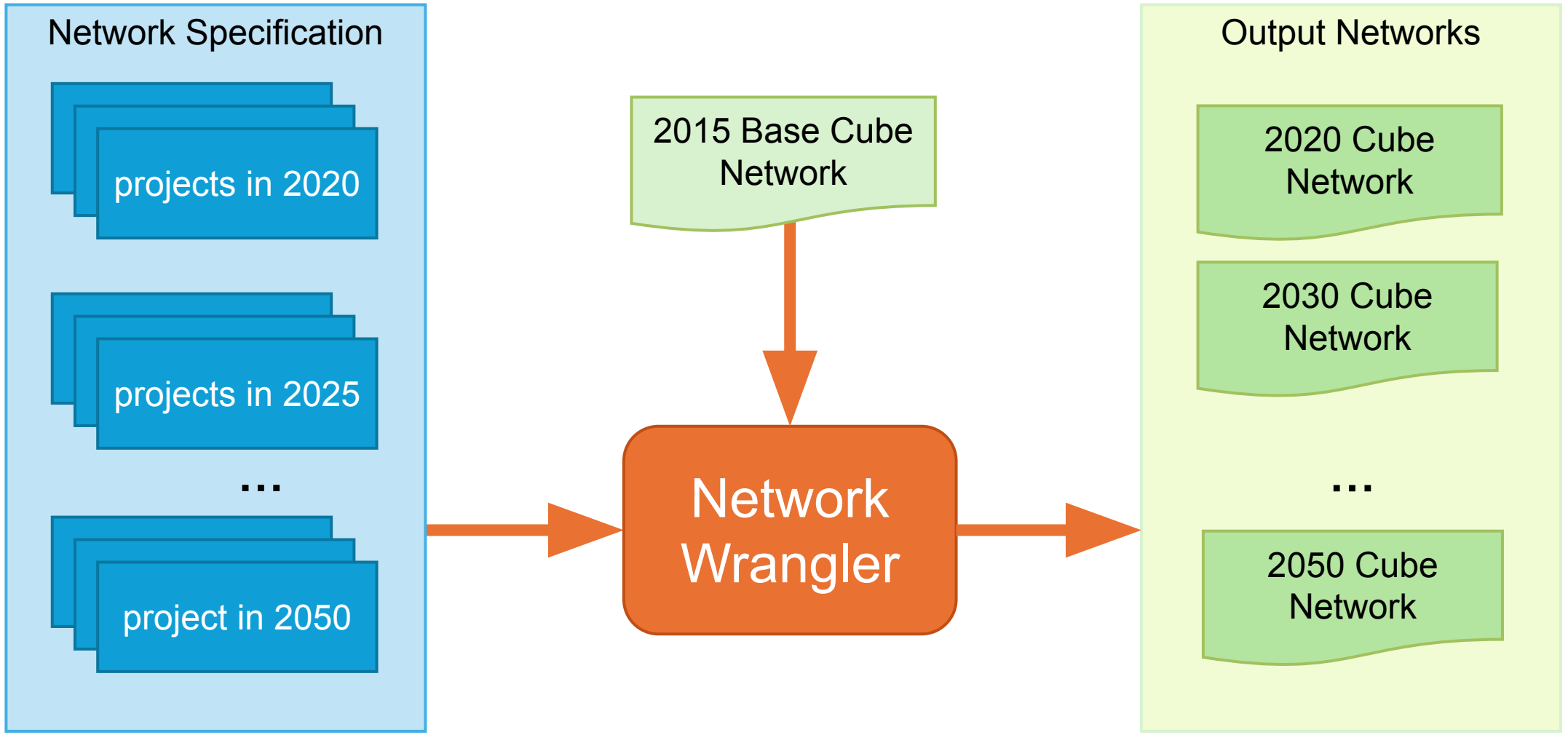
Network Wrangler is a tool that
builds up a **series of future networks**
from a **base network** and
an ordered list of **projects**

What is Network Wrangler?

Design principles:

- Building a network is a repeatable process
- Git and tagging are used for version control
  - For example, if a Plan Amendment occurs three years after the original model analysis, the network can be rebuilt with an update to a single project or the addition or removal of a single project
- A single network specification can be used to define multiple related scenarios (e.g., No Project and Plan)
- Assignment software agnostic

What is Network Wrangler? (v1)



Network Wrangler (v1) : Scenario Specification

Scenario Specification example:

https://github.com/BayAreaMetro/NetworkWrangler/blob/master/scripts/net_spec_blueprint.py

Network Wrangler (v1) History

2010

Network Wrangler started @ SFCTA
https://github.com/sfcta/NetworkWrangler

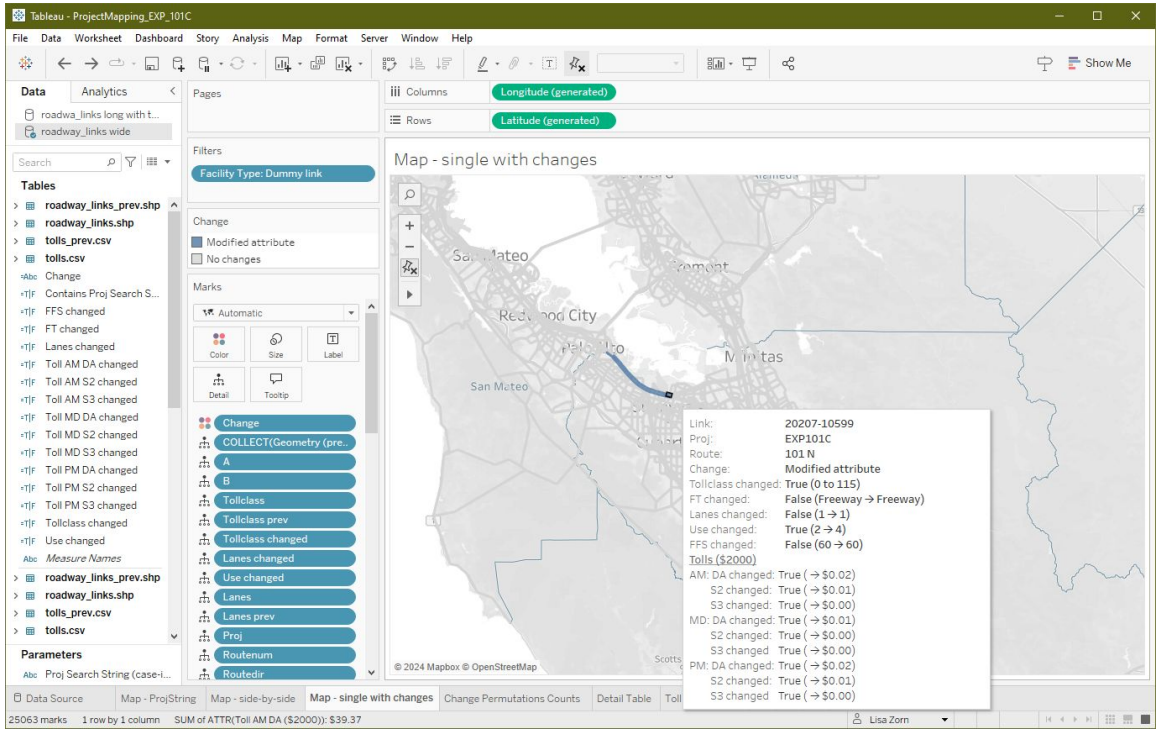
2018

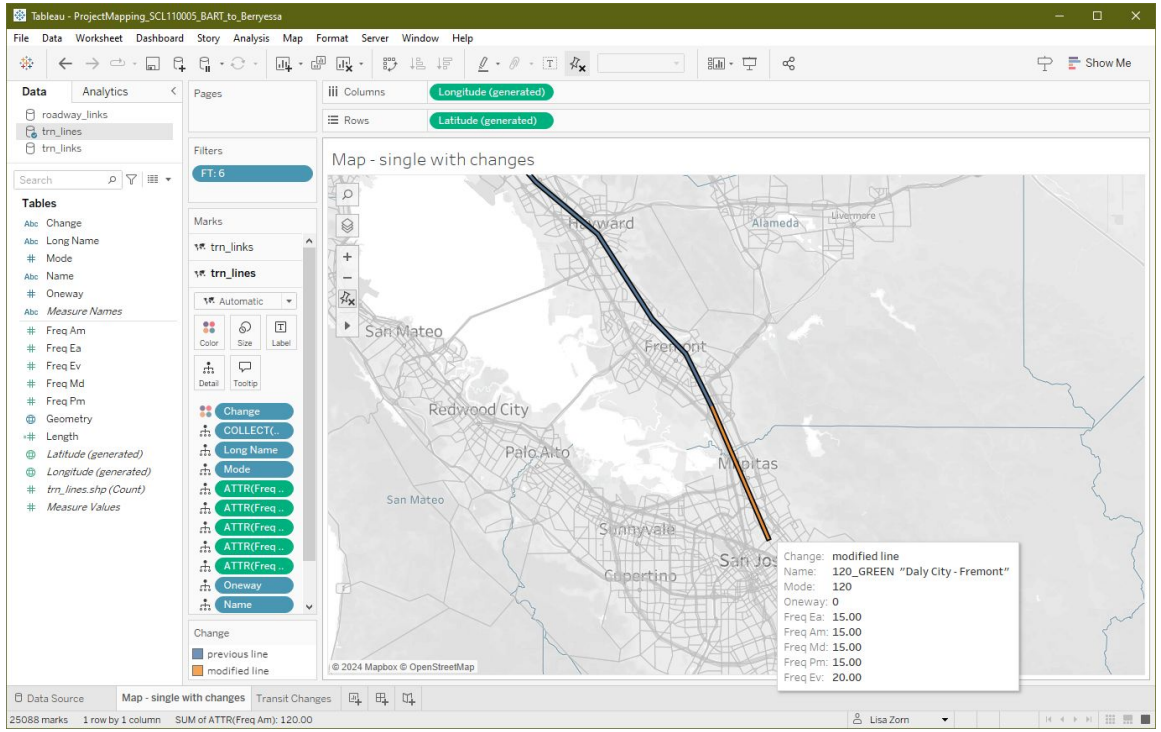
Network Wrangler came to MTC
https://github.com/BayAreaMetro/NetworkWrangler

2024

Project QA visualization option added last year

Network Wrangler (v1): Project QA Visualization

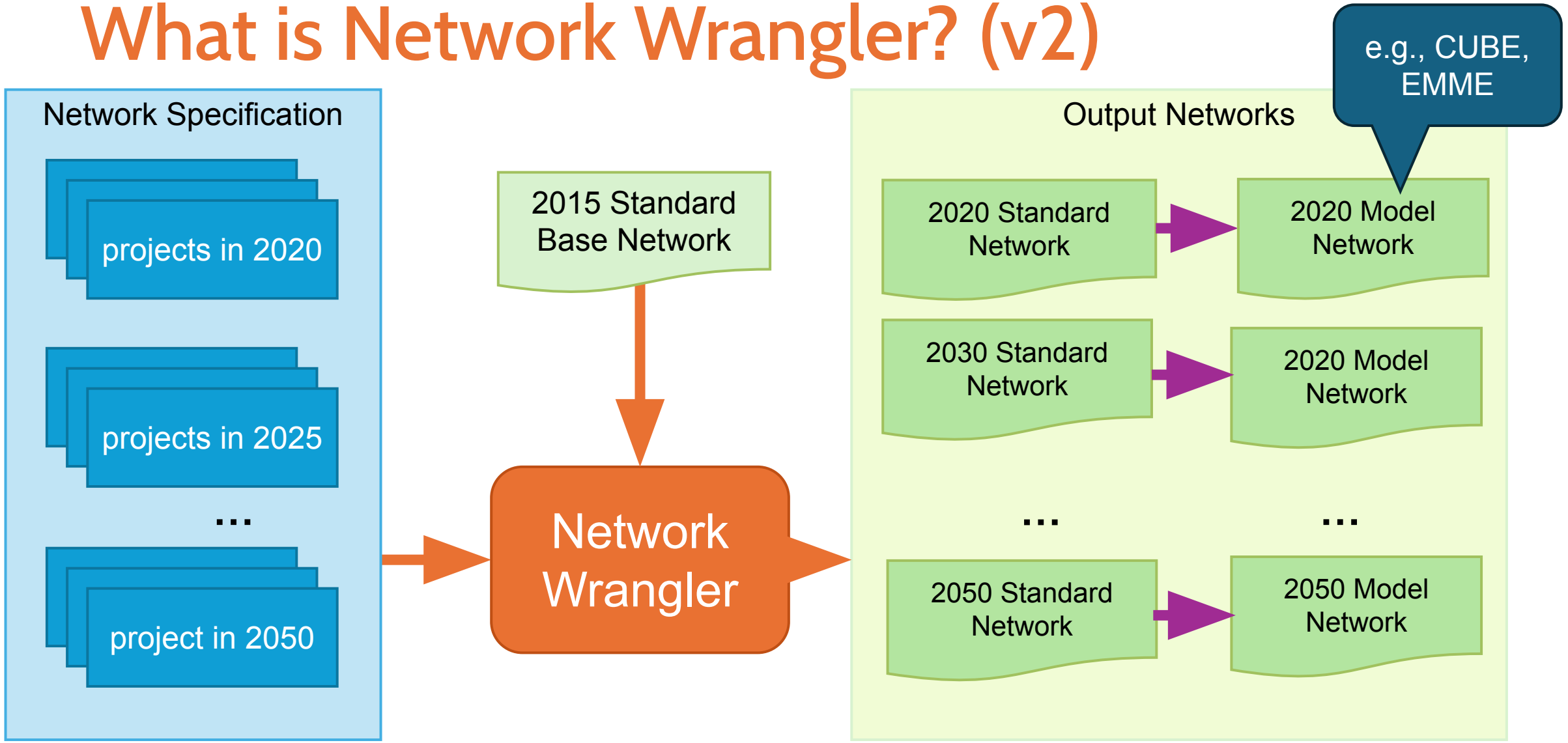




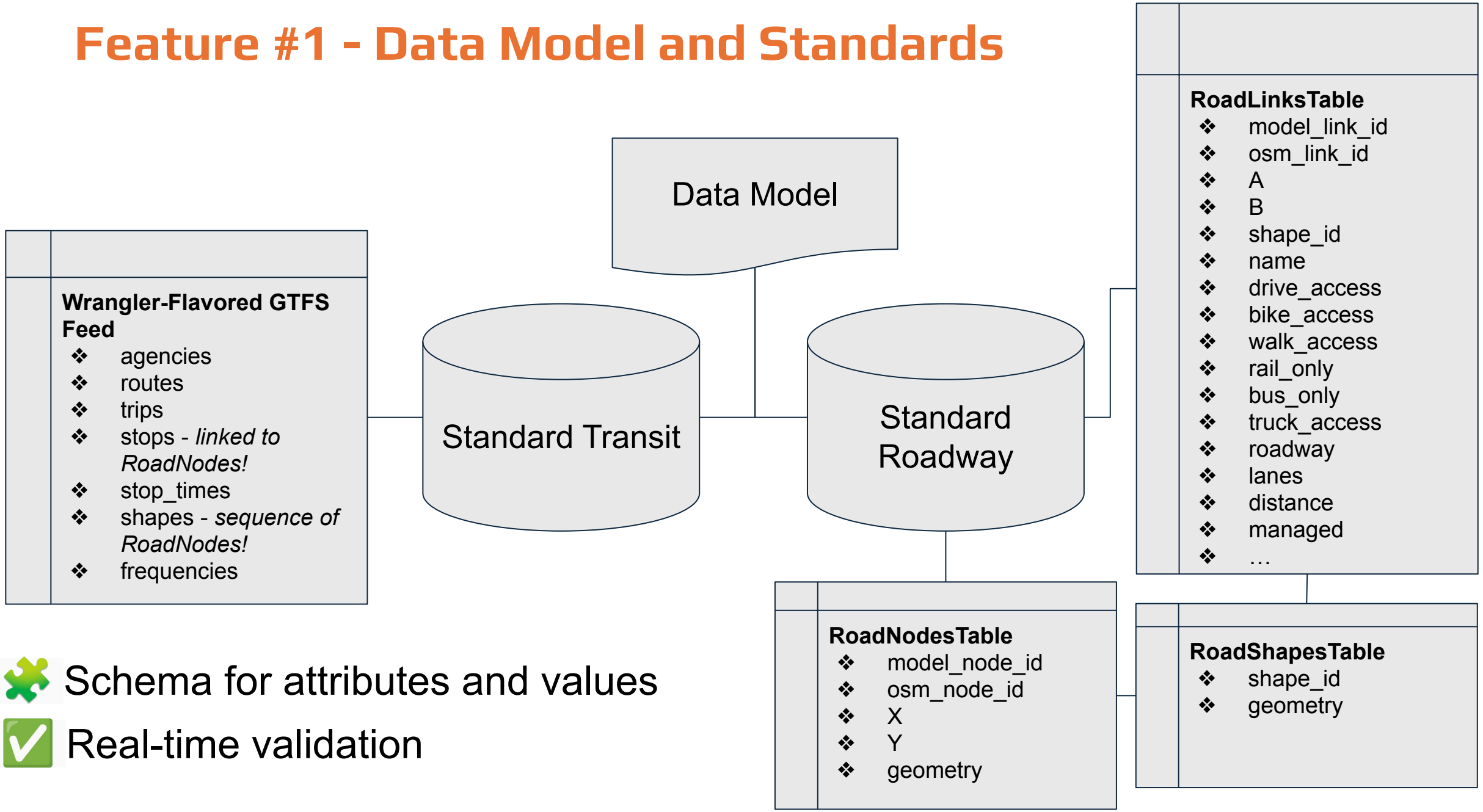
What is Network Wrangler? (v2)

- Builds upon v1 concepts
- Uses modern Python packages
- Defines network standards
- Operates on standard networks
- Stays agnostic to commercial vendor software platform

What is Network Wrangler? (v2)



Feature #1 - Data Model and Standards



🧩 Schema for attributes and values

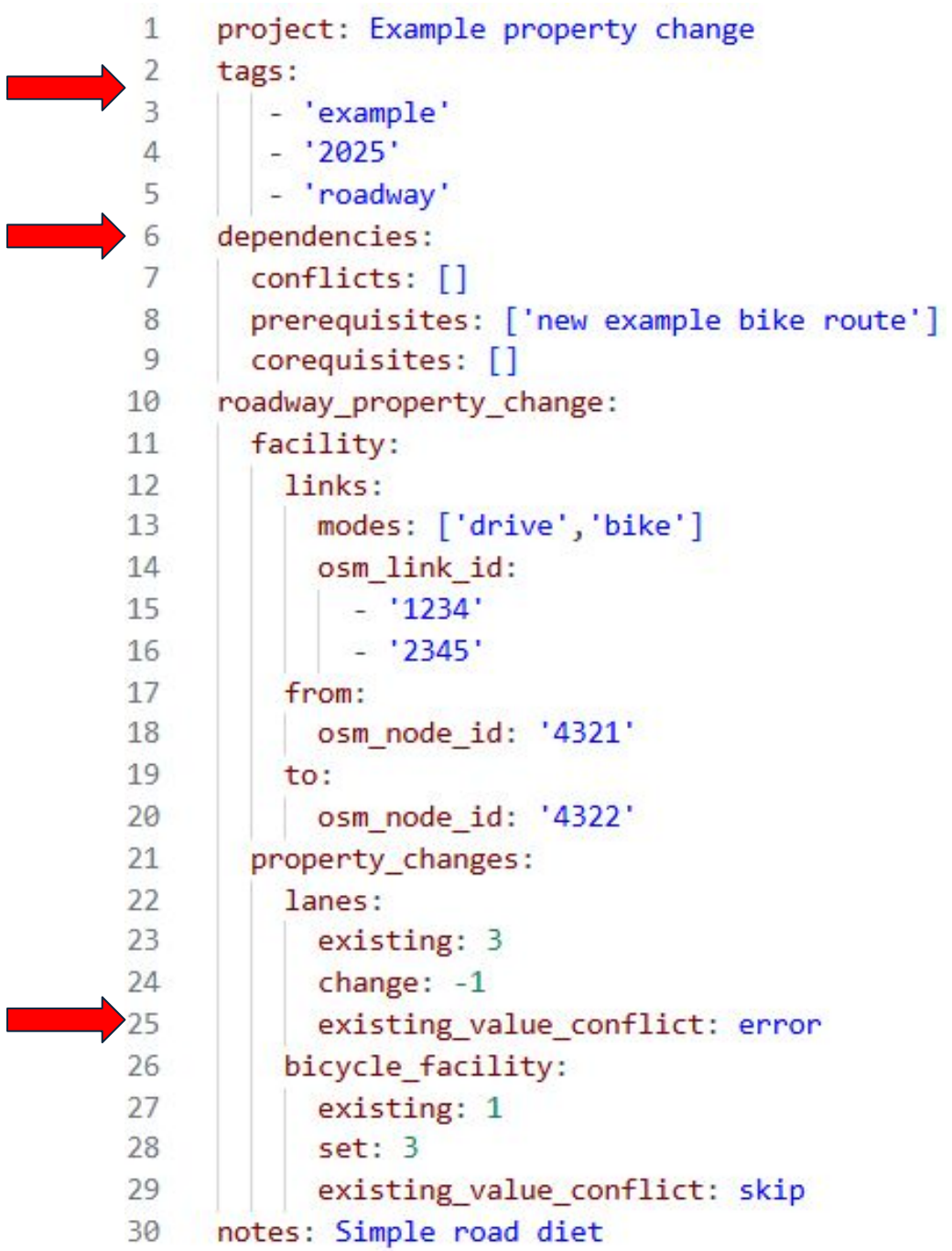
✅ Real-time validation

Feature #2 - Project Cards

📝 Document network changes in human-friendly text files

✅ Use and promote standards

🔁 Allow reuse

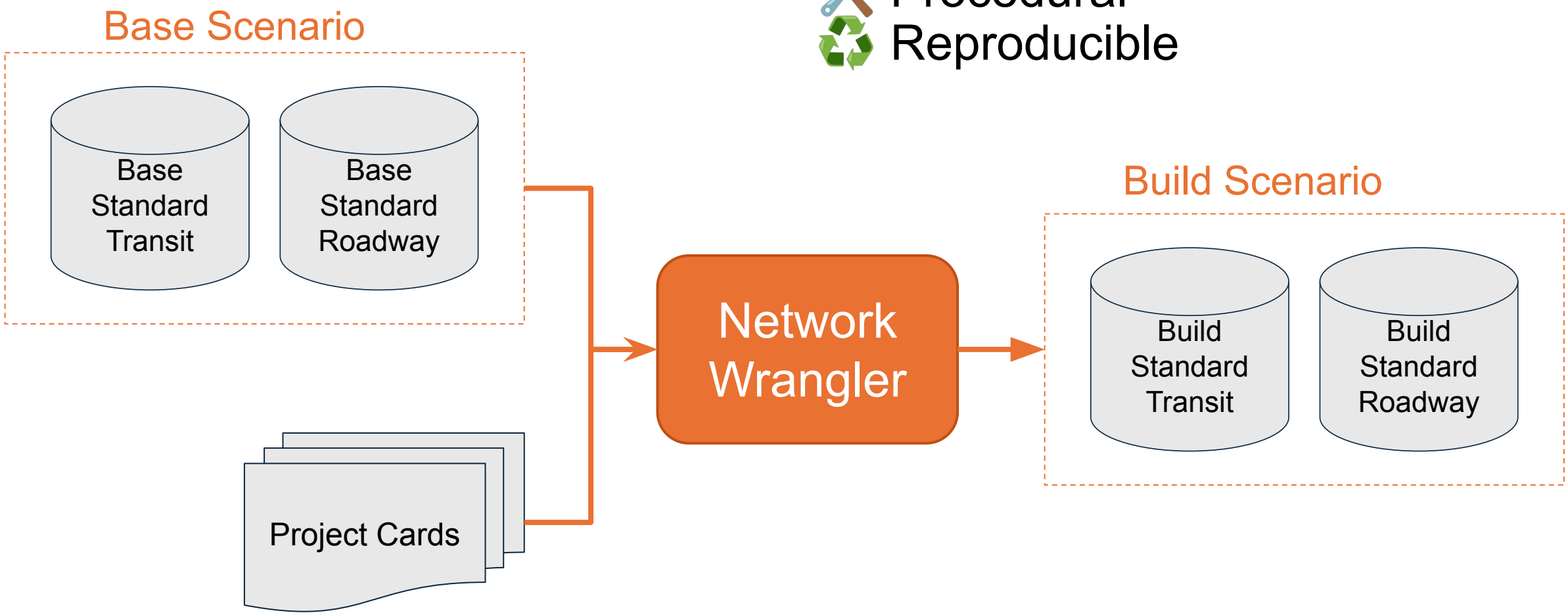


```
project: Example property change
tags:
  - 'example'
  - '2025'
  - 'roadway'
dependencies:
  conflicts: []
  prerequisites: ['new example bike route']
  corequisites: []
roadway_property_change:
  facility:
    links:
      modes: ['drive','bike']
      osm_link_id:
        - '1234'
        - '2345'
    from:
      osm_node_id: '4321'
    to:
      osm_node_id: '4322'
  property_changes:
    lanes:
      existing: 3
      change: -1
      existing_value_conflict: error
    bicycle_facility:
      existing: 1
      set: 3
      existing_value_conflict: skip
notes: Simple road diet
```

Feature #3 - Scenarios

Scenarios are created in a way that is:

- Prescriptive
- Procedural
- Reproducible



Feature #4 - Auditing

🕵️ Which projects are included in my network?

Network Wrangler can tell you in a split second!

Network Wrangler (v2) would be great for people who:

🐍 Prefer open, scriptable tools like Python
🔗 Encourage workflow sharing
👥 Seek collaborative project coding
🧭 Support or may switch between between commercial vendor software platforms *(e.g., CUBE, EMME, TransCAD, Visum)*

Want to see Network Wrangler (v2) in action?

💻 Stay for our demo!

🤝 Reach out to the Network Wrangler team - https://github.com/orgs/network-wrangler/people

What is GMNS?

The **General Modeling Network Specification (GMNS)** defines a common machine (and human) readable format for sharing routable road network files. It is designed to be used in multi-modal static and dynamic transportation planning and operations models.

What is GMNS?

High level requirements

1. GMNS is a data specification, not tied to any specific software tool
2. GMNS is extensible, not universal
   - The only required files are nodes and links, to support static network assignment
   - Specification accommodates user-defined fields
3. Extensions include data needed for dynamic, multi-modal networks
4. GMNS reflects infrastructure, services and policies:
   - physical roads
   - intersections
   - traffic controls
   - tolls
   - time-of-day restrictions
5. GMNS is human and machine readable

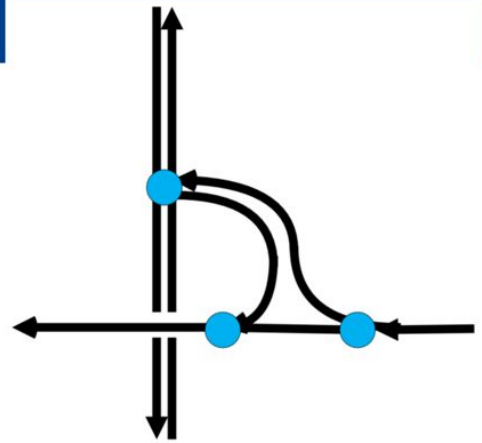
Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

Required elements

Node— a point that connects links

- Required fields: node_id, x_coord, y_coord
- Optional fields: name, node_type, ctrl_type, zone_id, parent_node_id



Any GMNS element can have user defined fields

Link— a directed or undirected line object in a network, defined by the nodes it travels from and to.

- Required fields: link_id, from_node_id, to_node_id, directed
- Optional fields: name, geometry_id, geometry, parent_link_id, dir_flag, length, grade, facility_type, capacity, free_speed, lanes, bike_facility, ped_facility, parking, allowed_uses, toll, jurisdiction, row_width
- Links are directed for vehicle travel, undirected for pedestrian travel

Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

Segments and Lanes

Segment—portion of a link defined by linear references

Lane— Lanes are numbered left to right with 1 as the left-most through lane. Left turn lane is –1. A bike lane is a lane with allowed_uses = BIKE

Turn pockets are defined via segments.

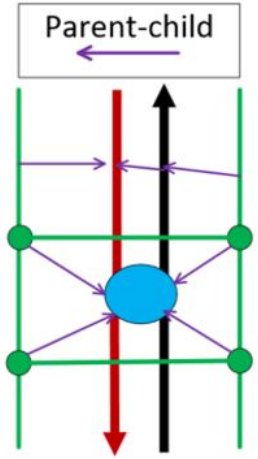
Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

Multimodal accommodation

The **allowed_uses** field indicates what may flow on a **link** or **lane** (e.g., walk, bike, bus, truck, auto, hov2, hov3+), as well as non-travel uses (shoulder, parking)

Location—a point that is associated with a specific location along a link, using a linear reference

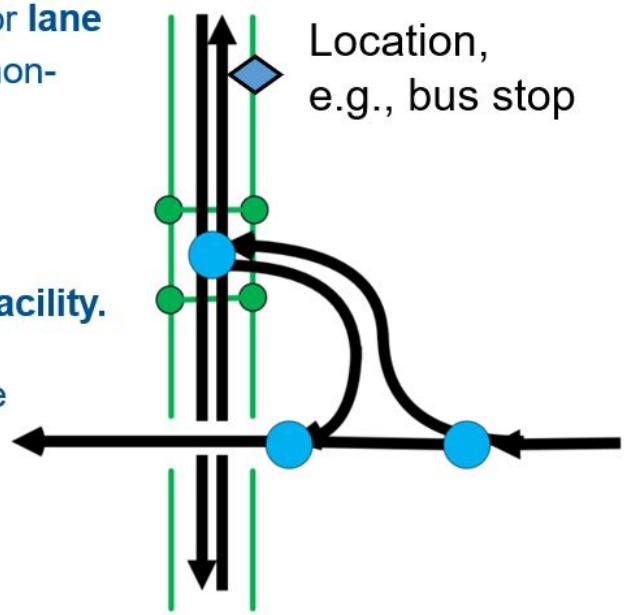


Links include fields for **ped_facility, bike_facility.**

Sidewalks and crosswalks may optionally be handled via their own undirected links.
Parent-child relationships:

- Sidewalk with associated road
- Crosswalk and intersection nodes

Separated bike facilities may also be handled as their own pairs of directed links

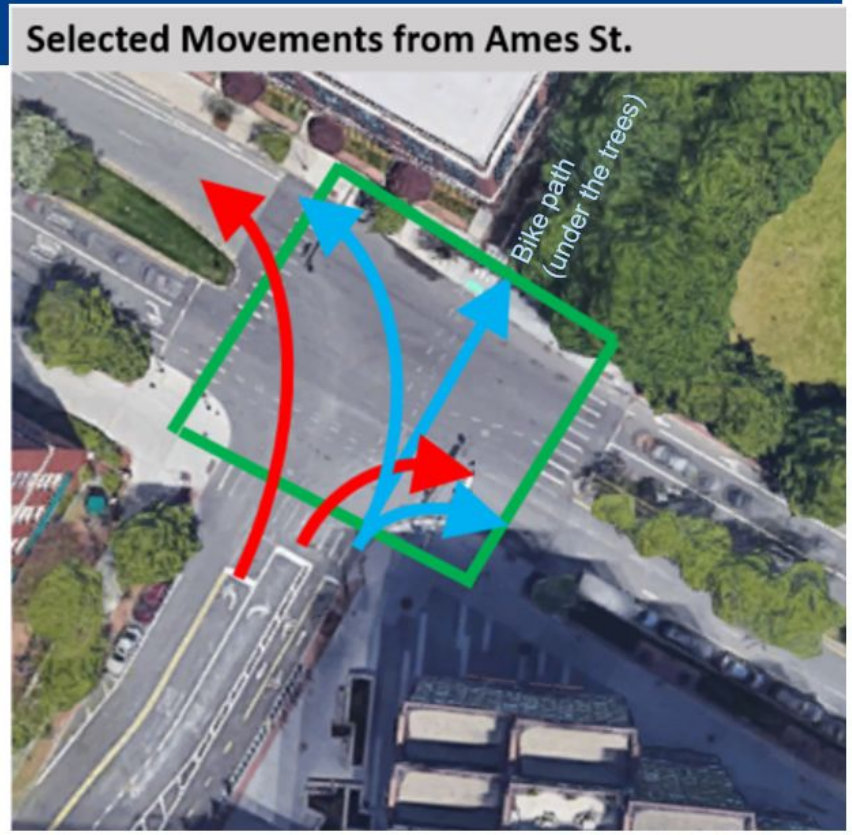


Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

Movements

- Movements define connections and traffic control types (none, yield, stop, signal) between inbound and outbound links or lanes at an intersection.
- Example:
  - Pedestrian movements in green
  - General traffic (red) and bike (blue) movements are shown from the south



Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

Traffic signals

Signal_controller— association of one or more intersections whose signals use the same controller

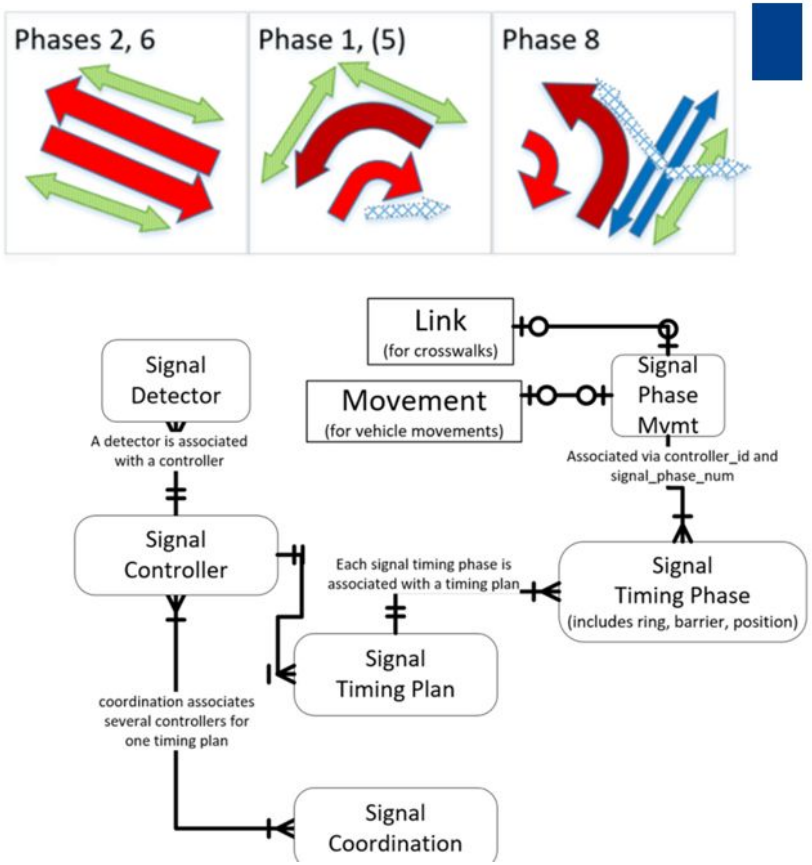
Signal_phase_mvmt— signal_phase mapped to its associated traffic movements and pedestrian links (e.g., crosswalks)

Signal_timing_phase— timing and concurrency information for each signal phase

Signal_timing_plan— timing plan for the signal, by controller, time period

Signal_coordination— coordination for several signal controllers, associated with a timing plan

Signal_detector— traffic detector associated with a controller, a phase and a group of lanes



Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

What is GMNS?

It takes a community…

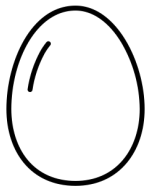
Tools that work with GMNS

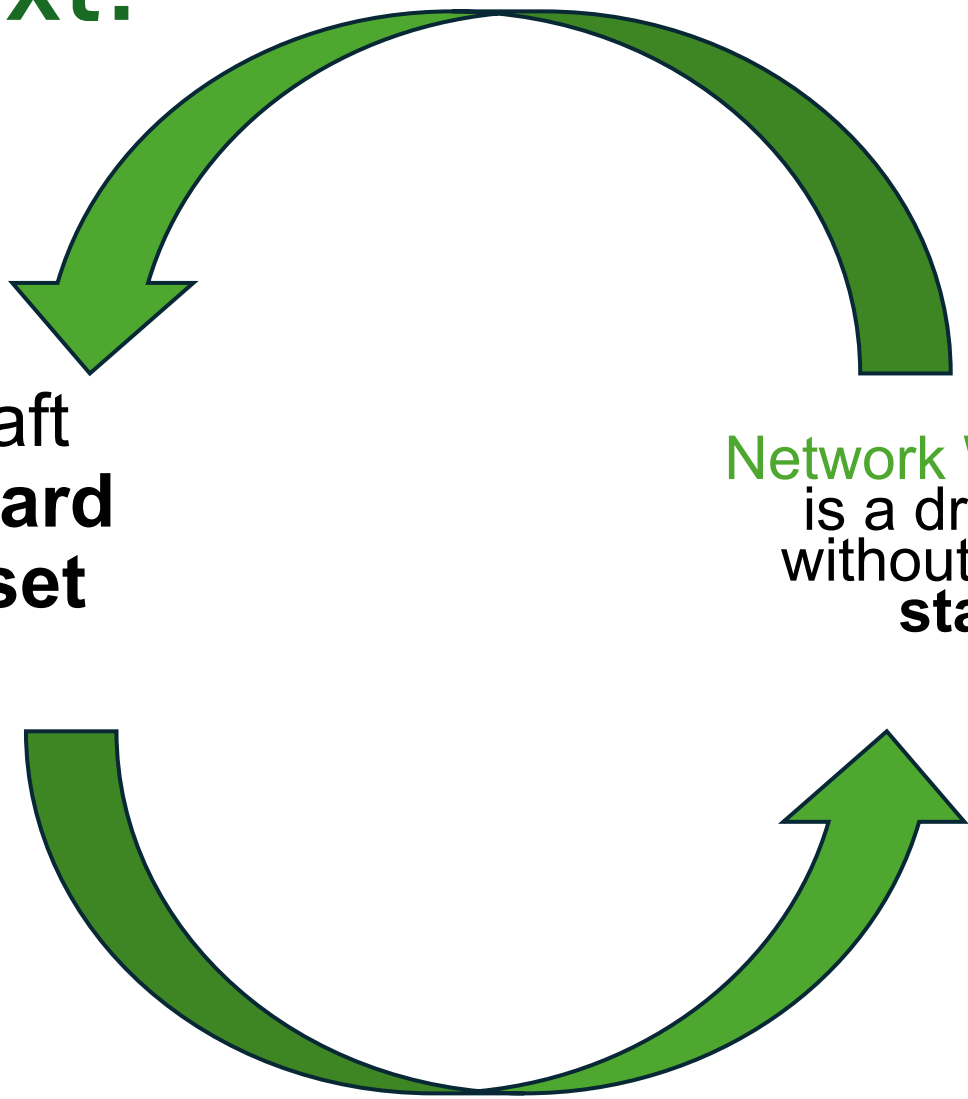
- **Network synthesis:**
  - **osm2gmns for nodes, links, movements**
  - **signal4gmns for traffic signals**
  - **NeXTA4gmns for visualization and editing**
- **Multiresolution network expansion:**
  - **net2cell**
- **Validation tools:**
  - **gmnspy for format validation (does the network conform to the spec?)**
  - **Graph validation (is the network connected?)**
- **Shortest path and routing:**
  - **path4gmns, with connection to DTALite**
  - **AequilibraE**

Source: Scott Smith, presentation at the **Joint Traffic Analysis and Modeling Workshop & Mid-Year Meeting**, August 2024

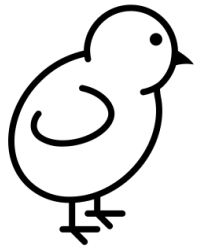
What is next?

GMNS is a draft **network standard** without a **toolset**





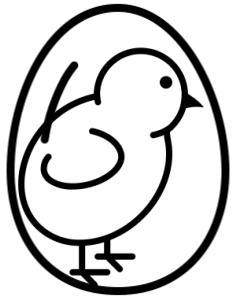
Network Wrangler (v2) is a draft **toolset** without a **network standard**



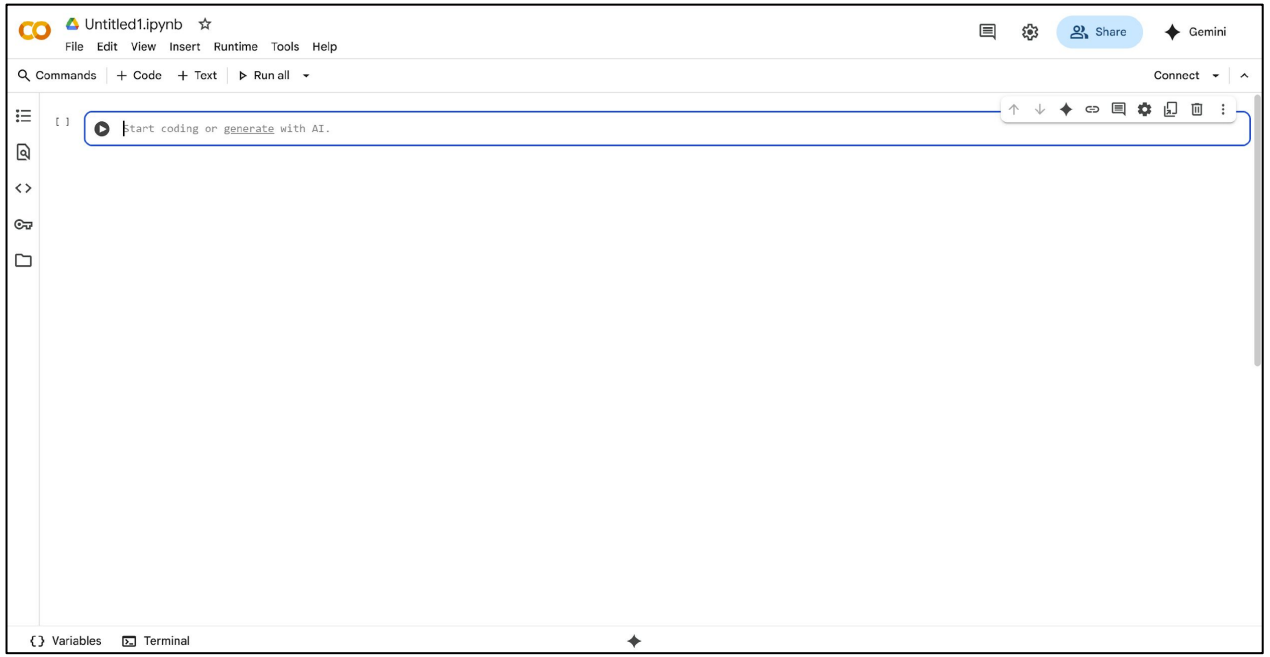
What is next?

Goals:

- Ease Network Wrangler adoption by implementing a universal network standard (GMNS+)
- Make GMNS+ more useful by pairing it with network tools (Network Wrangler)



Google Colab



- Free, browser-based Python interface
- No installation needed

Requirement:

- Google Account

GMNS->Accessibility Workflow Overview

From Zone to Assignment:

Step	What Happens (Planner's View)
1. Define Zones (TAZ/Census Tracts)	Identify where trips start/end (communities, districts).
2. Prepare Physical Network	Map roads, transit, bike lanes with capacity & speed.
3. Generate Connected Network	Connect zones to physical network for assignment
4. Readiness Validation Check	Verify network structure and configuration match required inputs
5. Traffic Assignment	Estimate how travel loads the system.
6. Accessibility Check	Measure access to different zones.

Presenters: Dr. Xuesong (Simon) Zhou, Henan Zhu (Arizona State University)

Why It Matters

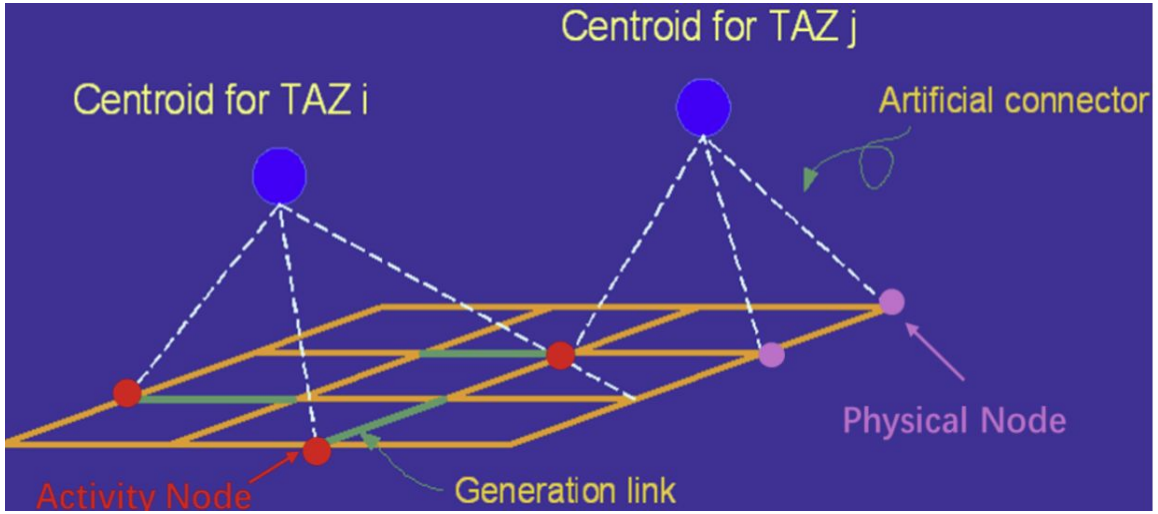
From OSM Data to Assignment-Ready Networks:

- **OSM-Generated Network**
  - Geographic foundation only
  - Raw nodes and links, inconsistent attributes
  - No direct link between zones, demand, and the physical network
- **Assignment-Ready Network**
  - Zones connected to the network
  - Forward-star structure for efficient path search
  - Demand integrated with network supply
  - Enables traffic assignment and accessibility analysis
- 💡 **Motivation**:
  OpenStreetMap provides *where the roads are*, but an assignment-ready network provides *how people and vehicles move through them*.

Connecting Zones to the Network

Forward-Star Network Structure

- Each zone has a centroid
- Connectors link centroid nodes to activity nodes in the real network
- Provides efficient, standardized inputs for assignment models



- Centroid nodes = where trips originate/terminate (demand side).
- Activity nodes = where those trips enter/exit the real network (supply side).

Multi-Level Validation Framework

Checking Readiness

- GMNS+ Dataset project introduces a standardized multi-level modeling readiness and validation framework.
- **Files ready? (node/link/demand checks)**
- **Attributes valid? (capacity, units, config)**
- **Modeling ready? (assignment & accessibility checks)**

Level 1 – Basic File Check
✓ Required files & fields (node/link)
✓ Data types, sorting, link endpoint validity

Level 2 – Demand & Zone Consistency
✓ Centroid-node match, connectors
✓ Demand file structure & zone consistency

Level 3 – Network Attribute Validation
✓ Units for speed/length, capacity checks
✓ VDF fields and conversion consistency

Level 4 – Single Mode Configuration Validation
✓ Config file presence
✓ One-mode setups with consistent parameters

Level 5 – Observed Volume Checks and ODME Preparation
✓ Observed volume validity
✓ Sufficient data for ODME

Level 6 – Accessibility Assessment
✓ OD path feasibility
✓ Compute access metrics & identify gaps

Level 7 – Traffic Assignment Validation
✓ Volume, VMT/VHT, avg. speed checks
✓ Compare to observed data (R^2, RMSE, etc.)

Level 8 – Extended ODME Post-Quality Checks
✓ Demand-target matching
✓ High-demand route feasibility & anomaly flags

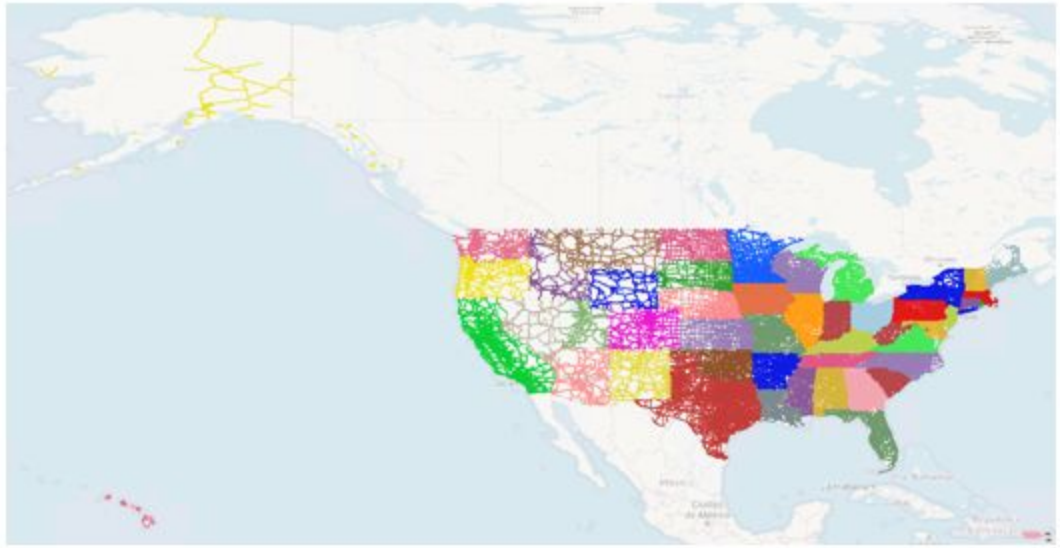
File Output Overview & Accessibility

Accessibility Evaluation
Once node.csv, link.csv, demand.csv, and settings.csv files are ready, run the TAPLite simulation

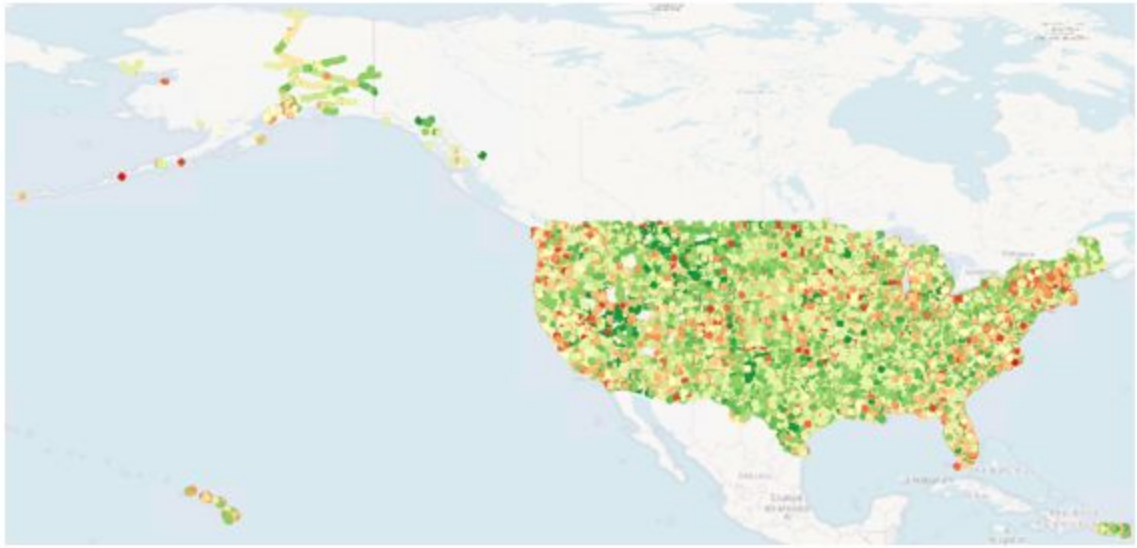
- This will generate: zone_accessibility.csv

Category	Measure	Definition
Origins (to this zone)	origin_count (Count)	Number of zones reaching this zone
	origin_avg_distance_mile origin_avg_distance_km (Avg. Distance)	Mean distance traveled (mi / km)
	origin_avg_free_flow origin_avg_congestion (Avg. Travel Time)	Mean time under free-flow and congested conditions (min)
Destinations (from this zone)	destination_count (Count)	Number of zones reachable from this zone
	destination_avg_distance_mile destination_avg_distance_km (Avg. Distance)	Mean distance to destinations (mi / km)
	destination_avg_free_flow destination_avg_congestion (Avg. Travel Time)	Mean time under free-flow and congested conditions (min)

OSM-> GMNS Networks



State by state networks



City by city networks

Buckeye

Mesa

Tempe

Tucson

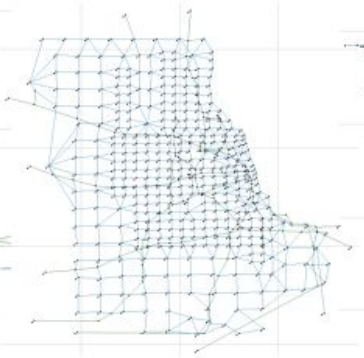
Goodyear

Casa Grande

TNTP Assignment Test Networks -> GMNS

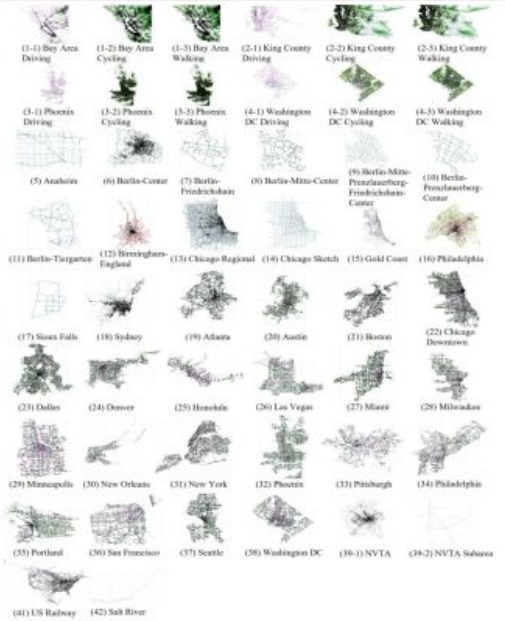
39	Planning Networks for NVTA		The Northern Virginia Transportation Authority (NVTA)	-	-	-	-	-	-	-	-	
39.1	NVTA_Full_Network	Virginia		√	17,826	49,577	√	√	√		√	
39.2	NVTA_Subarea_Network			√	13	28	√	√	√		√	
40	State_level_Networks_MultiModal (USA)	U.S. states and D.C.	Generated from OpenStreetMap by OSM2GMNS https://osm2gmns.readthedocs.io/en/stable/		-	-	√	√	√			
41	Railway Network (USA)	-	National Transportation Atlas Database		27,220	35,257	√	√	√			
42	Waterway Network (Salt River)	Arizona	Generated by OSM2GMNS https://github.com/asu-trans-ai-lab/osm2gmns_water		345	338	√	√	√			

13_Chicago Regional



14_Chicago Sketch

22_Chicago Downton



Source: https://github.com/bstabler/TransportationNetworks

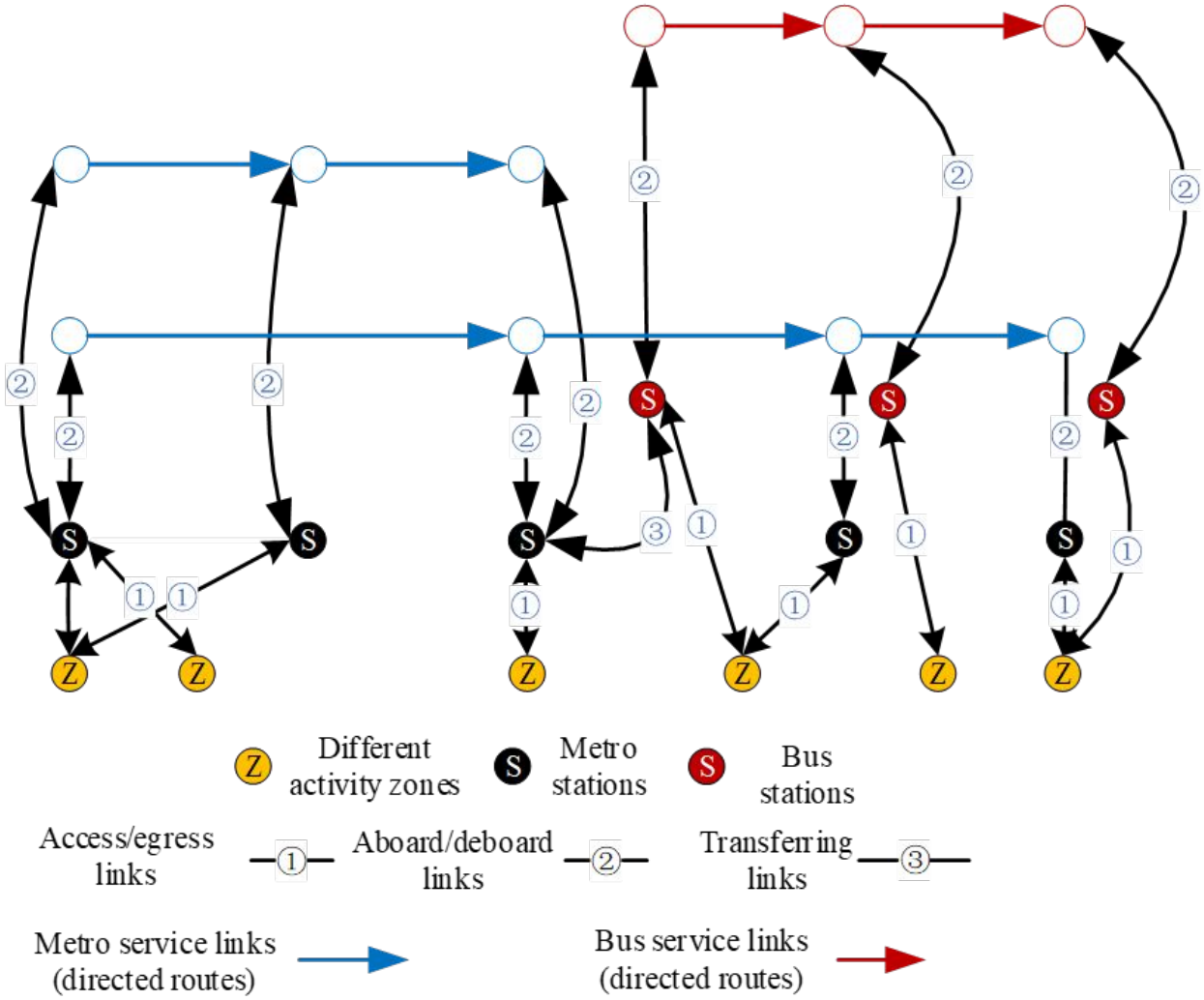
Planning Shapefiles-> GMNS Networks

Washington DC Driving Network

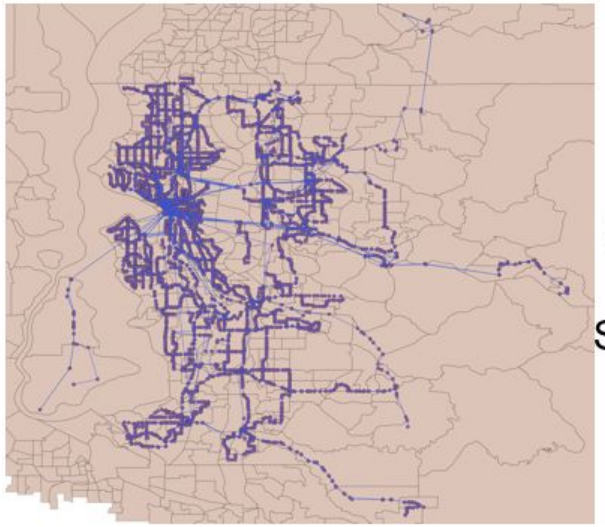
Washington DC Cycling Network

Washington DC Walking Network

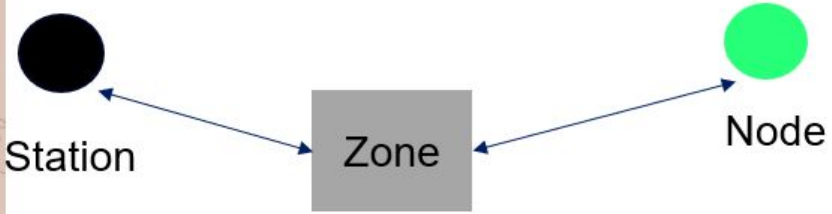
GTFS-> GMNS Networks

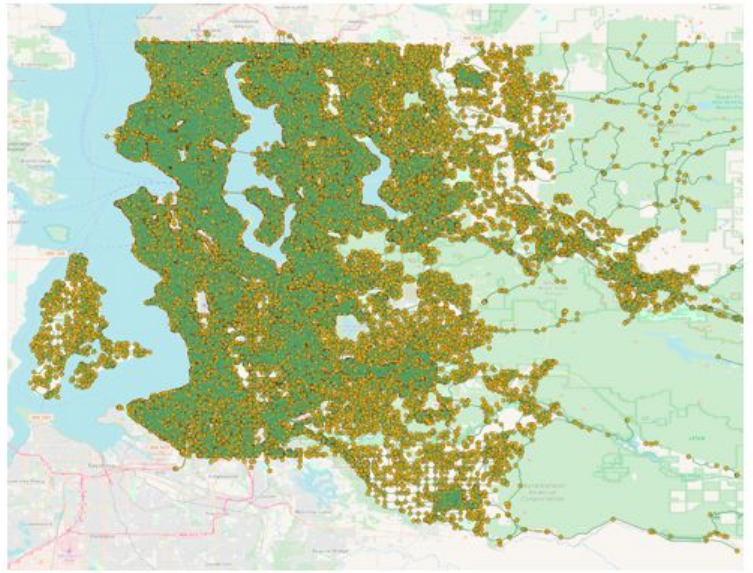


Integrated Multimodal GMNS Network



Transit network for King County





Road network for King County

UTDF->GMNS

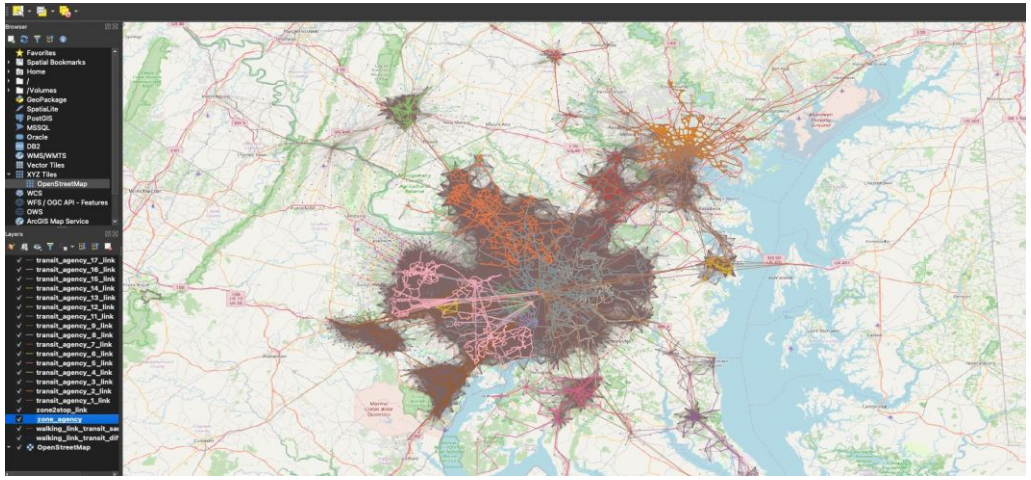
# of nodes = 17,552
- # of stops = 6,095
- # of virtual service node (bus) = 11,408
- # of virtual service node (tram) = 37

# of links = 40,837
- # of service links = 11,133
- # of boarding links = 22,906
- # of transferring links = 6,798

OSM->GMNS

# of nodes = 104,544
# of links = 236,405

GMNS codes multimodal transfers in a standardized way, enabling explicit routing and multi-agency transit integration



The relationship between mode_type and allowed_use

link type	mode_type	w_bus	w_metro	w_bus_metro	d_bus	d_metro	d_bus_metro
sta2sta_1r	bus	√		√	√		√
	metro		√	√		√	√
sta2r	bus	√		√	√		√
	metro		√	√		√	√
r2r	bus	√		√	√		√
	metro		√	√		√	√
sta2sta_2r	bus2bus	√		√	√		√
	bus2metro			√			√
	metro2bus			√			√
	metro2metro		√	√		√	√
z2sta	walk2bus	√		√			
	walk2metro		√	√			
	drive2bus				√		√
	drive2metro					√	√
s2s_2a	bus2bus	√		√	√		√
	bus2metro			√			√
	metro2bus			√			√
	metro2metro		√	√		√	√

Source: Northern Virginia Transportation Authority (NVTA) Multimodal DTA Model Development Documents

Walkthrough

Accessibility Analysis Using GMNS
San Francisco Case Study

Andre L. Carrel, The Ohio State University

Overview

- **Goal:** Conduct an accessibility analysis using GMNS-formatted network data and the DTALite package
- **Case Study:** San Francisco
- **Tools:** Google Colab, DTALite, Folium
- **Required files:** Provided in a Google Drive folder

  Link at https://u.osu.edu/carrel/momo

Getting set up (1)

1. Accessing Colab

- Go to https://colab.research.google.com/
- Sign in with Google account
- Create new Colab file (New → More → Google Colaboratory)
- Notebook will be saved in your Google Drive

2. Mounting Google Drive

- This is required to access workshop files and save outputs to Google Drive
- Code snippet:

```
from google.colab import drive
drive.mount('/content/drive')
```

- Drive available at: /content/drive/MyDrive/

Getting set up (2)

3. Installing Libraries:

- Most libraries we'll use are pre-installed in Colab
- Install additional libraries if needed:

```
!pip install DTALite
```

- Other libraries used in the workflow: pandas, geopandas, numpy, scipy, and folium

4. Accessing Data:

- Copy contents of **" MoMo 2025 Accessibility Analysis – Public"** into **MyDrive**.
- In Chrome you can copy directly. Otherwise, you need to download and re-upload
- Direct link
- Link at https://u.osu.edu/carrel/momo

Overview

Study Area: City of San Francisco

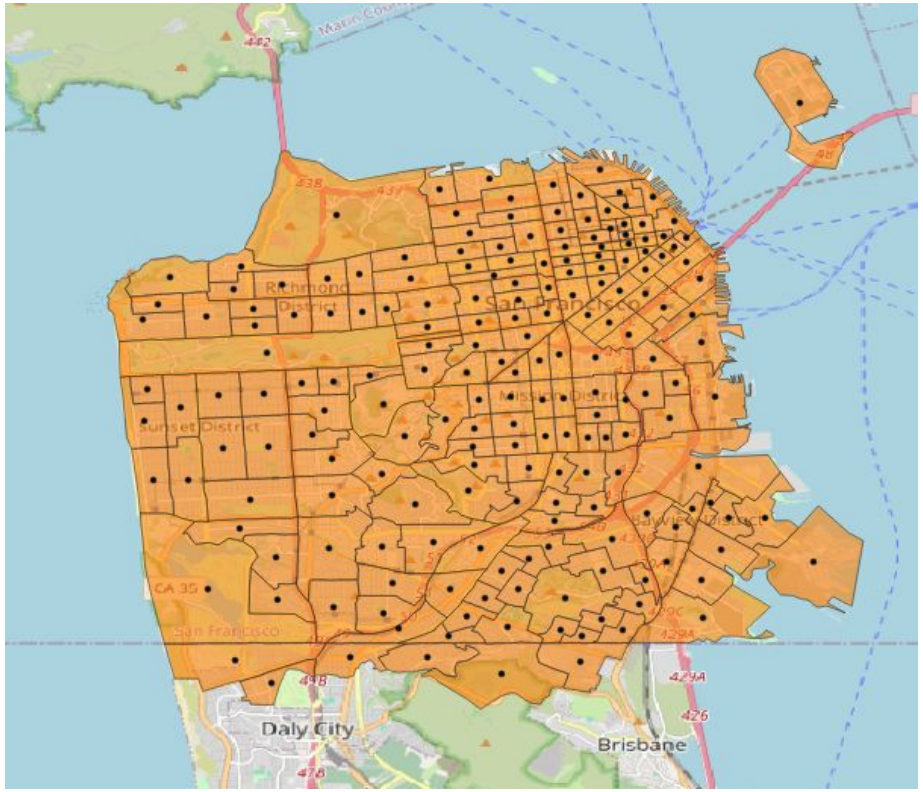
Analysis: Zone-based accessibility. Average travel time from any given zone (TAZ) to all other zones of the city, using walking and public transportation

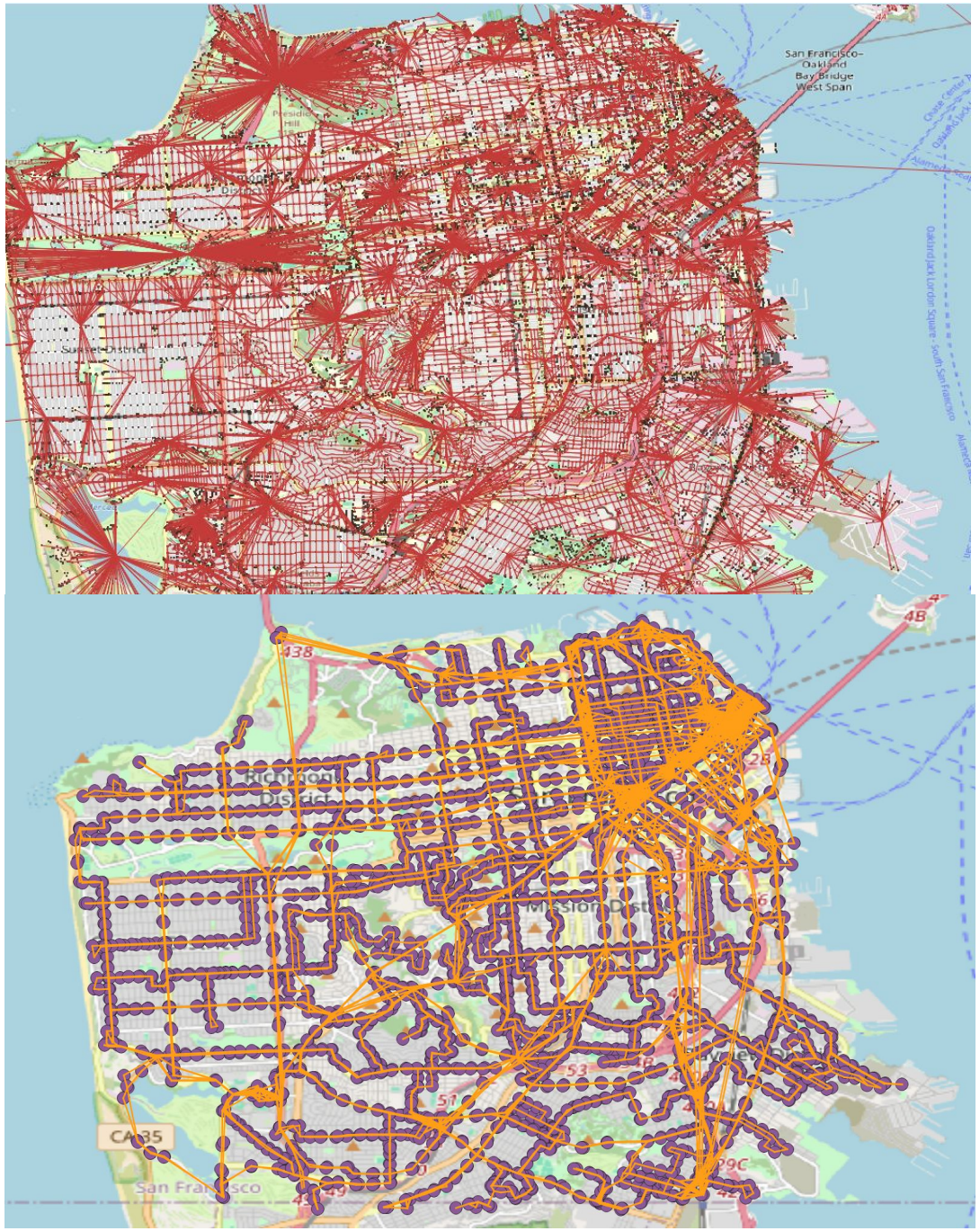
Key Input Files (from Drive):

- Walk network, from OSM: walk_nodes.csv, walk_links.csv
- Transit network, from GTFS: transit_nodes.csv, transit_links.csv
- Zone centroids: TAZ_centroid.csv

All files are in the shared Google Drive folder.

Networks and Zones - Overview





Left: Zones and centroids
Right top: Walking network
Right bottom: Transit network

DTALite

- Dynamic traffic assignment tool created by Arizona State University
- For use with GMNS-formatted networks
- Can also perform accessibility analyses
- More information:
  https://github.com/asu-trans-ai-lab/DTALite/wiki
  https://github.com/itsfangtang/DTALite_release/wiki/DTALite-Inputs-and-Outputs

Step 1: Prepare Networks

Run commands in notebook:
"Renumber links and nodes for DTALite and create stop-to-walk connectors"

- The walking network serves as the base network. The transit network will be overlaid.
- DTALite requires all node and link IDs of the integrated network to be consecutive

What the code does:
- Reassign node and link IDs for both networks
- Create bidirectional connector links between stops and nearest walk nodes

Inputs: Walk and transit network files
Outputs:
- updated_walk_nodes.csv, updated_walk_links.csv
- updated_transit_nodes.csv, updated_transit_links.csv
- transfer_connector_links.csv

Step 2: Integrate Networks

Run commands in notebook: “**Integrate Networks**”

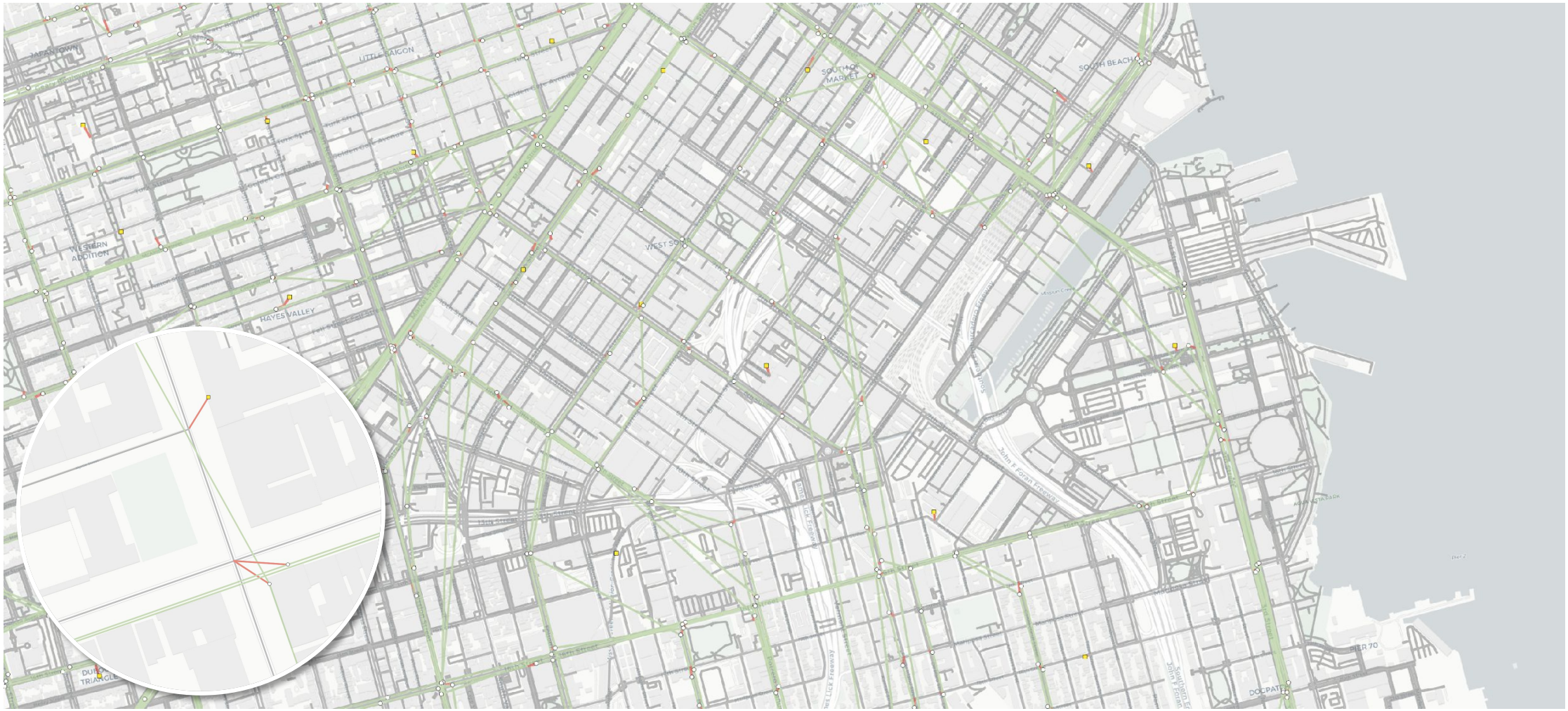
Purpose: Merge walk, transit, and connector networks into one single network

What the code does:
- Integrate walk and transit network data in single node and link files
- Standardize fields for DTALite requirements
- Apply “permitted uses” attribute: walk → “walk”, transit → “bus”

Inputs: Renumbered node and link files from previous step, connector links

Outputs:
- node.csv and link.csv
- These file names are required by DTALite

Integrated Networks - Visualization



Legend Zone Centroid Bus Stop Transit Link Pedestrian Link Connector (centroid-node, stop-node)

Step 3: Prepare Settings File

- DTALite requires settings.csv
- Provided in Google Drive, but if no settings file is available:
- Set parameter: number_of_iterations = 0
  - Run DTALite once to generate sample_settings.csv:

```
import DTALite as dta
dta.assignment()
```

  - Rename sample_settings.csv → settings.csv

Step 4: Perform Accessibility Analysis

Run commands in notebook: **“Integrate Networks”**

- Perform the accessibility analysis by running DTALite with number of iterations set to 0
- The possible origins and destinations are the zone centroids

```
import DTALite as dta
dta.assignment()
```

Inputs: node.csv, link.csv, settings.csv

Outputs:
- od_performance.csv – Travel times/distances by OD pair
- zone_accessibility.csv – Average accessibility by zone

Additional output files produced if the number of iterations is > 0.

Principal Output: Zone Accessibility

zone_accessibility.csv

Fields:

- **origin_count:** # of zones that can reach this zone
- **origin_avg_distance:** Avg. distance from origins
- **origin_avg_free_flow:** Avg. free-flow travel time
- **origin_avg_congestion:** Avg. congested travel time
- **destination_count:** # of zones reachable from this zone
- **destination_avg_distance:** Avg. distance to destinations
- **destination_avg_free_flow:** Avg. free-flow time
- **destination_avg_congestion:** Avg. congested time

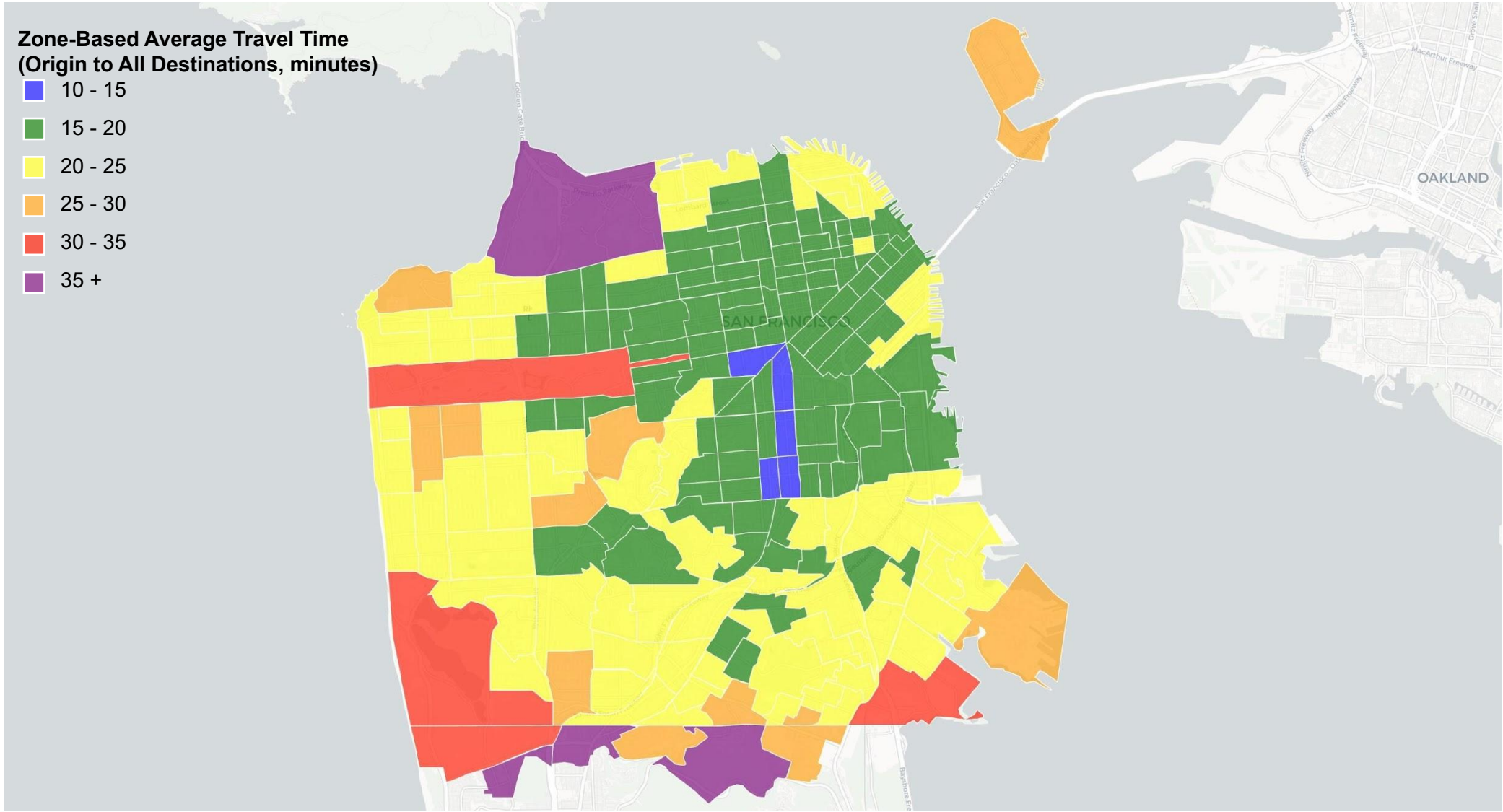
Step 5: Visualization

Run commands in notebook: **“Visualize the Results”**

Inputs:
- zone_accessibility.csv
- TAZ_centroid.csv

Output:
- Choropleth map average travel time to other zones
- Saved as accessibility_map.html

Visualization Result



Transit Network Representation

Currently, transit networks are treated the same as automobile networks:

- Links with travel times/speeds
- No wait times

Work is currently underway to add representation of wait times, transfer times, time-dependent travel times, and travel times based on real-time information.

Other GMNS Learning Resources

https://drive.google.com/file/d/1kT785-GtLg_ow-6-IxCxuEnJ5ytk8dAX/view?usp=sharing

https://github.com/hhhhhenanZ/2025MoMo_Workshop_Demo/blob/main/2025MoMo_Workshop_Demo_User_Guide_v2.ipynb

Thank You!