Replication Guide for "Clean Growth"

Costas Arkolakis Conor Walsh

Table of Contents

1 Introduction **3**
- 1.1 Remarks . . . 3
- 1.2 Requirements . . . 3
- 1.3 Storage . . . 4

2 Downloads **5**
- 2.1 Software . . . 5
  - 2.1.1 Osmosis . . . 5
  - 2.1.2 PostgreSQL . . . 5
  - 2.1.3 Osm2pgsql . . . 5
  - 2.1.4 PGAdmin . . . 5
- 2.2 Data . . . 6
  - 2.2.1 Open Street Map . . . 6
  - 2.2.2 Geofabrik . . . 6
  - 2.2.3 Global Power Plant Database . . . 6

3 Replication **7**
- 3.1 Filtering . . . 7
  - 3.1.1 Elements . . . 7
  - 3.1.2 Tags . . . 8
  - 3.1.3 Commands . . . 8
- 3.2 Export to Database . . . 9
  - 3.2.1 Commands . . . 9
- 3.3 Abstraction . . . 11
- 3.4 Estimate Line Capacity . . . 11
- 3.5 Aggregation . . . 12
- 3.6 Figures . . . 12

4 Abstraction Algorithm **13**
- 4.1 SQL Algorithm . . . 13
  - 4.1.1 Station Clustering . . . 13
  - 4.1.2 Line Endpoint Replacement . . . 13
  - 4.1.3 Splitting Lines . . . 14
  - 4.1.4 Merging Lines . . . 15
  - 4.1.5 T-Intersections, Auxiliary Stations . . . 15
  - 4.1.6 Abstracting Lines . . . 15
- 4.2 Sliding Box Algorithm . . . 16
- 4.3 Output . . . 16
  - 4.3.1 Vertices . . . 16
  - 4.3.2 Links . . . 17

4.3.3 Intersections 17

5 Capacity Estimation **18**
5.1 Clean Voltages 18
5.2 St. Clair Curve 18

6 Aggregation Process **20**
6.1 Aggregate Assets 20
6.2 Aggregate Lines 20
6.3 Construct Matrices 22

7 Grid Improvements **23**
7.1 TransWest Express Line 23
7.2 SunZia Transmission Line 23
7.3 Grain Belt Express Line 24
7.4 Champlain-Hudson Power Express Line 26

8 References **28**

SECTION 1

Introduction

This a replication manual and data guide for Arkolakis and Walsh (2022). First we describe the necessary software and data sources that need to be downloaded. Then we walk through the steps required to replicate the paper in detail. Next we provide an overview of what the abstraction algorithm is doing to the the raw Open Street Map data for intuition. Finally we discuss how the abstracted power grid data is aggregated and used to produce the adjacency matrices and figures for the paper.

1.1 Remarks

This manual is a complete description of all steps for those interested in following the data construction process from the download of raw data to the output of figures and final estimates. **For those who only wish to replicate the results using intermediate data from our replication package, many of the steps can be ignored.** Most downloads and replication steps are only necessary for the processing and abstraction of the raw Open Street Map data, which has a long run time. If starting from the pre-abstracted files, the user may skip to Section 3.4.

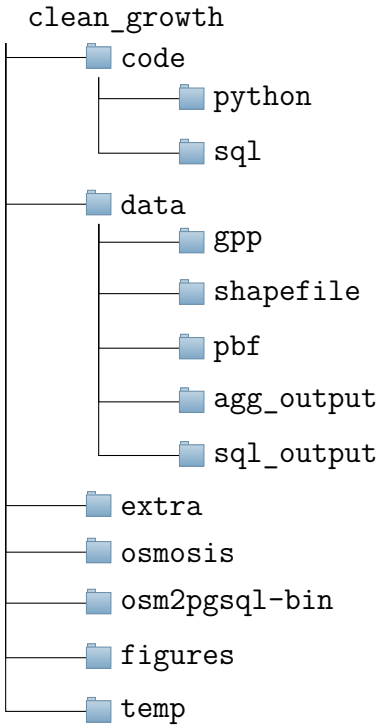
1.2 Requirements

The scripts and commands referenced in this document were tested on Windows though with minimal alterations should be able to run on other systems. The system and software versions that were used to construct the data are listed here.

Windows:	11
osmosis:	0.48.3
PostgreSQL:	13.3
PostGIS:	3.1
osm2pgsql:	1.4.2
PGAdmin 4:	5.2
Python:	3.6

1.3 Storage

The structure of the project folder is displayed here. The folder contains scripts, data, and intermediary outputs. The code includes the SQL scripts and Python script for running the main OSM abstraction algorithm. The data includes Global Power Plant data, region shapefiles, output files for the SQL algorithm, as well as aggregated data and adjacency matrices.



SECTION 2

Downloads

2.1 Software

2.1.1 Osmosis

Osmosis is a Java tool that allows us to extract OSM data and filter out elements we do not need. OSM contains information on far more than just power lines; it includes roads, buildings, public transit routes and so on. *Osmosis* allows us to filter these out. You can find instructions and a link for installing *osmosis* here. [17]

2.1.2 PostgreSQL

PostgreSQL or *Postgres* is an object-relational database management system. It is a powerful tool for querying and manipulating data tables that relate to one another. While we refer to the steps taken in SQL as an "algorithm", it is really a long set of queries that extracts and manipulates the relevant data into output files for analysis. The OSM data is stored on a *Postgres* database while the SQL text files run their queries and operations on it. The download link is here. [22]

2.1.3 Osm2pgsql

Osm2pgsql is a command line software for importing OSM data to the *PostgreSQL* database. You can find instructions and a link for installing *osm2pgsql* here. [20]

2.1.4 PGAdmin

PGAdmin is a Graphical User Interface (GUI) for *PostgreSQL*. It allows us to access databases and run queries more intuitively than doing so via the command prompt. While not strictly necessary, troubleshooting and SQL query editing become much easier with *PGAdmin* so it is highly recommended for those trying to do a complete replication from original data. Debugging in SQL is much simpler from the PGAdmin interface, particularly since the queries we use are complex. Researchers interested in reading or ammending the queries we use are recommended to install it as well. The download link can be found here. [21]

2.2 Data

All intermediate data can be found in the replication package. These are instructions for downloading data from their original sources.

2.2.1 Open Street Map

The file containing all global OSM data is called `planet-{date}.osm` where date is the download date with no punctuation in the order 'year, month, day'. While OSM data can be stored in various formats, we use the `.pbf` format because it is the smallest and contains all the relevant information. Our data was downloaded from Open Street Map on March 22nd, 2021 and is about 57 GB. The latest data can be downloaded here. [18].

2.2.2 Geofabrik

Depending on the research question of interest, we might be interested restricting to a specific country or region. To bound the area of data we are concerned with, we use a `.poly` file during the filtering process, which speeds up this process. This can also be done with a bounding box if the user is not concerned with filtering around national borders. We do not do this for the global analysis, however it is useful for testing aspects of the algorithm on smaller geographical areas. The source site, Geofabrik, is a consulting firm that specializes in Open Street Map services. They provide `.poly` extracts from OSM here. [12].

2.2.3 Global Power Plant Database

The file `gpp_bleed.csv` contains production capacity and geolocation data on power plants around the world. We use it to measure power assets by region. It can be downloaded from the World Resources Institute here. [13].

SECTION 3

Replication

The filtering, export, and abstraction steps are only necessary for taking raw data from Open Street Map directly and constructing the data from scratch. This is useful for updating the raw data to the most recent OSM version or if a full replication is necessary. For those only interested in replicating the exact results from the paper using our pre-abstracted data, we recommend starting with the aggregation in step 3.4 using the files in the replication package.

3.1 Filtering

The purpose of installing *osmosis* is to filter through only the data related to the power grid and in the geographic area we want. Though the abstraction algorithm is original, commands for filtering and exporting the data come from the SciGRID transmission network model [19]. Below we provide some context about how OSM data is structured to help explain the filtering process.

3.1.1 Elements

OSM data is stored as either a *node*, *way*, or *relation*. An individual object in one of these categories is called an *element*.

- A **node** is a point in space.
- A **way** is a line, or set of line segments. A 'closed' way forms a polygon.
- A **relation** is a collection of nodes and ways that are locally related.

All transmission lines and power station are stored as one of these objects. Lines are nearly always stored as ways. Stations however may be stored as either nodes, if they are very small, or ways, if they are buildings or larger objects that require a polygon representation. We do not consider relations because they contain a combination of nodes and ways which are already captured when we consider these two types.

3.1.2 Tags

Each element in the OSM data is labelled with a *tag*, which is made up of two parts: a *key* and a *value*. A tag is written in the form <'key'='value'>. The tag can be thought of as a short description the kind of element we are working with, to help us differentiate between buildings, roads, power lines and so on. The key is the category of the object and the value is the type of object it is within that category. For example, nearly all objects related to the power grid are labelled with the 'power' key. A value for the 'power' key might be 'plant' or 'line'.

3.1.3 Commands

The following commands are for the Windows command prompt. Syntax may be different for Mac or Linux. We use `PATH` to denote the path where the requisite file is stored.

1. Create a temporary folder to store the temporary data created during the filtering process.

```
set JAVACMD_OPTIONS="-Djava.io.tmpdir={PATH}"
```

2. Set the directory to where `osmosis.bat` is saved.

```
cd "{PATH}\osmosis\bin"
```

3. Filter `planet-{DATE}.osm.pbf` for objects with the 'power' key, using the United States as the bounding polygon. This process can take a few hours for large polygons like the US. Name the file `FILENAME`.

```
osmosis ^
--read-pbf file=~{PATH}\planet-{DATE}.osm.pbf ^
--tag-filter accept-relations route=power ^
--used-way ^
--used-node ^
--bounding-polygon file=~{PATH}\us.poly completeRelations=yes ^
--buffer outPipe.0=route ^
--read-pbf file=~{PATH}\planet-{DATE}.osm.pbf ^
--tag-filter accept-relations power=* ^
--used-way ^
--used-node ^
--bounding-polygon file=~{PATH}\us.poly completeRelations=yes ^
--buffer outPipe.0=power ^
```

```
--read-pbf file=~{PATH}\planet-{DATE}.osm.pbf ^
--tag-filter reject-relations ^
--tag-filter accept-ways power=* ^
--used-node ^
--bounding-polygon file=~{PATH}\us.poly completeRelations=yes ^
--buffer outPipe.0=pways ^
--read-pbf file=~{PATH}\planet-{DATE}.osm.pbf ^
--tag-filter reject-relations ^
--tag-filter reject-ways ^
--tag-filter accept-nodes power=* ^
--bounding-polygon file=~{PATH}\us.poly ^
--buffer outPipe.0=pnodes ^
--merge inPipe.0=route inPipe.1=power ^
--buffer outPipe.0=mone ^
--merge inPipe.0=pways inPipe.1=pnodes ^
--buffer outPipe.0=mtwo ^
--merge inPipe.0=mone inPipe.1=mtwo ^
--write-pbf file=~{PATH}\path{FILENAME}.osm.pbf
```

A few remarks about these commands. We can removing the commands starting with `--bounding-polygon` if filtering the entire world. We can use a bounding box by changing `--bounding-polygon` to `--bounding-box` and list coordinates by writing `top=x bottom=y left=z right=w` separated by spaces to determine the sides of the box. For more information on each individual command, see the SciGrid user guide here [19].

3.2 Export to Database

After filtering, the data needs to be exported to a database before we can analyze it or make changes. Here we make use of the *osm2pgsql* program to export the data, as well as *PostgreSQL*. We add an extension to the database called *PostGIS* that allows us to work with geometries and geographic aspects of the data. If filtering the global data

3.2.1 Commands

1. Set the directory to where *PostgreSQL* programs are saved.

```
cd "{PATH}\PostgreSQL\{VARIABLE}\bin"
```

2. Create a database called DATABASE.

```
createdb ^
--username=postgres ^
--port=5432 ^
--host=127.0.0.1 {DATABASE}
```

3. Add *PostGIS* extension to the database.

```
psql ^
--dbname={DATABASE} ^
--username=postgres ^
--host=127.0.0.1 ^
--port=5432 -c "CREATE EXTENSION postgis;"
```

4. Add *hstore* extensions to the database.

```
psql ^
--dbname={DATABASE} ^
--username=postgres ^
--host=127.0.0.1 ^
--port=5432 -c "CREATE EXTENSION hstore;"
```

5. Change to the directory where *osm2pgsql.bat* is located.

```
cd "{PATH}\osm2pgsql-bin"
```

6. Export the data to the newly created database using *osm2pgsql*. The style file is stored in the replication data folder under 'extra'.

```
osm2pgsql ^
-r pbf ^
--username=postgres ^
--host=127.0.0.1 ^
--port=5432 ^
--database={DATABASE} ^
--style {PATH}\power.style ^
--hstore ^
```

```
--number_processes nb_processors ^
~{PATH}\{FILENAME}.osm.pbf
```

3.3 Abstraction

This step is implemented by running the script `osm_abstraction_algorithm.py`, which calls a set of SQL queries from another folder. Abstraction takes the following SQL tables as inputs:

- 'planet_osm_line'
- 'planet_osm_point'
- 'planet_osm_polygon'

It outputs the following .CSV files:

- `vertices.csv`
- `links.csv`
- `intersections.csv`

The input SQL tables can be seen in the *Postgres* database from PGAdmin. They are produced by the export process described in Section 3.2. The abstraction script performs a sequence of operations on them to clean, manipulate, and structure them for analysis. These output tables contain the information necessary to visualize our model of the grid and construct an adjacency matrix of stations or regions. Details of the abstraction algorithm can be found in Section 4.

3.4 Estimate Line Capacity

This step is implemented by running the script `line_capacity_estimation.py` which takes as inputs:

- `links.csv`

It outputs:

- `links_cap.csv`

The estimation script cleans the voltage data and estimates the capacity of each line. The output `links_cap.csv` file contains all of the data present in `links.csv` with additional columns for mean voltage (over merged lines) in kilovolts and line capacity in megawatts. More details about the assumptions and methods for estimating power line capacity can be found in Section 5.

3.5 Aggregation

Aggregation refers to aggregation of line and asset data across regions. It is done by running the script `simplify_grid.py`. This code takes in the following files as inputs:

- `selected_regions.shp`
- `gpp_bleed.csv`
- `links_cap.csv`
- `vertices.csv`
- `intersections.csv`

It outputs:

- `csr_edges.csv`
- `csr_aggcap.csv`
- `csr_adjmat.csv`
- `csr_adjmat_count.csv`
- `csr_adjmat_dist.csv`
- `csr_adjmat_volt.csv`
- `csr_adjmat_lcap.csv`

`Selected_regions.shp` is a shapefile dividing countries of the world into more granular regions. The `gpp_bleed.csv` file contains Global Power Plant Database data on power assets. It outputs seven files: a table of edges between regions, a table of aggregate capacities by region, and five different kinds of adjacency matrices. The prefix 'csr' refers to 'country selected regions', to differentiate it from the earlier data which is at a finer resolution. More details about how the data is aggregated and how the matrices are constructed can be found in Section 6.

3.6 Figures

Figures for the paper are plotted by running the script `plot_figures.py`. This script takes in files from all of the previous steps and creates images used in the final paper.

SECTION 4

Abstraction Algorithm

The abstraction algorithm takes raw transmission line and power station data from Open Street Map (OSM) and constructs a simplified model of the grid for analysis. There are two primary components: the SQL algorithm which performs the actual geometric operations and data adjustments to produce the desired model, and the Python script which acts as a meta-algorithm to iteratively apply the SQL algorithm to geographic subsets of the data. SQL scripts are numbered in the order they are called and named with reference to the main operations they contain. The names help differentiate the files but do not perfectly summarize the contents of each script, which often contain other operations.

4.1 SQL Algorithm

4.1.1 Station Clustering

We use the term "station" as a catch-all to describe plants, stations, substations and transformers. At any given location, there may be multiple polygons representing different buildings or components within the same "station". For example, a hydro-electric dam may be physically separate from the accompanying substation which transmission lines actually connect to. This is more granularity than is feasible to work with. As such, we cluster stations within 500 meters of each other. This involves grouping stations within 500 meters of one another and taking the convex hull of each group as the final station.

4.1.2 Line Endpoint Replacement

The algorithm replaces line endpoints with the relevant station centroids at each step. Relevant stations here mean stations that either intersect the line at the endpoint or are within 500 meters of the line endpoint. The reason for doing this is that, in the output files, stations are represented by their centroids. This makes it important for the endpoints of lines to intersect those centroids instead of elsewhere on the station polygon. We also do this for lines which have endpoints within 500 meters of stations to correct for data entry mistakes. Spot checks on Google Earth confirm that many lines that actually connect to

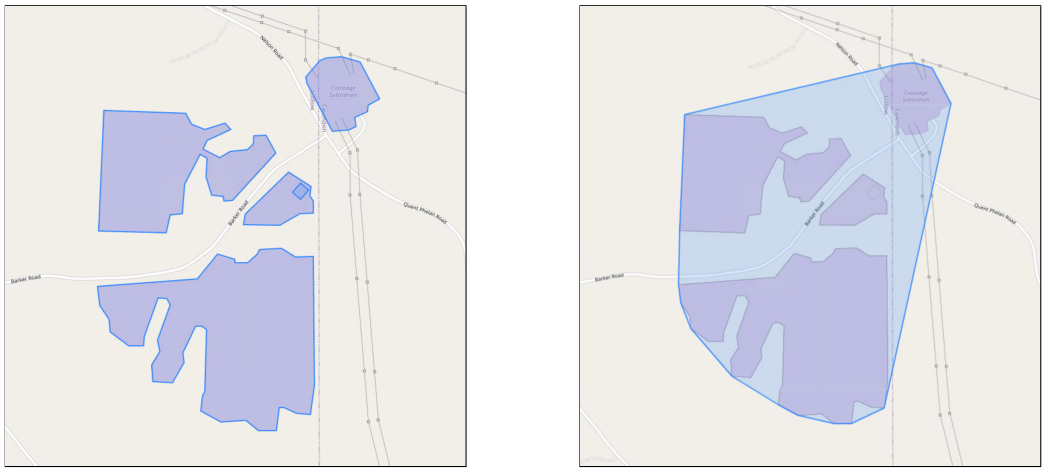


Figure 1: A collection of power plants, stations, and substations that are within 500 meters of one another, before (left) and after (right) they are clustered into the convex hull

Figure 2: A set of lines that intersect a station before (left) and after (right) their endpoints have been replaced with the station centroid

nearby stations are not captured by an intersection in the OSM data; instead the lines end outside of the polygon.

4.1.3 Splitting Lines

A single line may be intersected by one or more stations. We define a unique line as one that connects at most two stations. Thus we split each line by any stations that intersect it by subtracting the intersection of the station and line and replacing the endpoints of the new lines with the station centroid. This is also done for stations that are within 500 meters of lines by taking the closest point on the line to the nearby station, buffering it, and subtracting that buffered object from the line. A single line can only be split by one station at a time because it creates new lines from which line-station intersections must be recalculated, hence this is written as a loop.

Figure 3: A line that is intersected by a station before (left) and after (right) it has been split by the station. The blue and red lines represent separate lines after the split.

4.1.4 Merging Lines

Many power lines with separate IDs connect to one another without a station between. We treat this as a single line. Lines with unique IDs that intersect outside of stations are grouped and unioned. We also do this for lines with endpoints within 500 meters of one another. Line grouping is a transitive relationship. If line A connects to B and B connects to C, all three need to be grouped together prior to the merge.

4.1.5 T-Intersections, Auxiliary Stations

A T-intersection is defined as an intersection between two lines A and B where A intersects (or lies within a 500 meter radius of) Line B but not at line B's endpoint. It describes a intersection which looks like the letter-T. The T-relationship is not symmetric; A may act as the bottom of the T for line B but this does not imply that B does the same for A at any point. In cases where the relationship is symmetric, we treat this as a single line. As such the line segments between the intersection points are dropped, replaced with a straight line between the points, and all of these segments are merged. When the T-intersection is not symmetric (the more common case), the top of the T is split by the bottom, and an artificial station called an "auxiliary" station is constructed at that point to keep track of the intersection.

4.1.6 Abstracting Lines

Lines which do not intersect any other lines at endpoints are removed and all lines are straightened to a direct line between the two endpoints. We refer to the straightening as 'abstraction' because it abstracts away from details about the line's geometry.

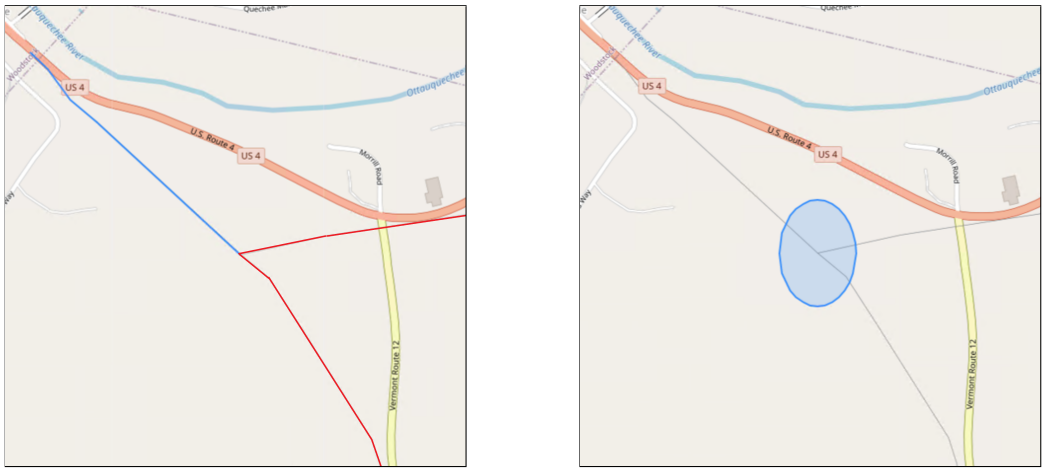


Figure 4: A T-intersection (left) and the auxiliary station constructed for it (right) where the blue and red lines represent the "bottom" and "top" of the T respectively.

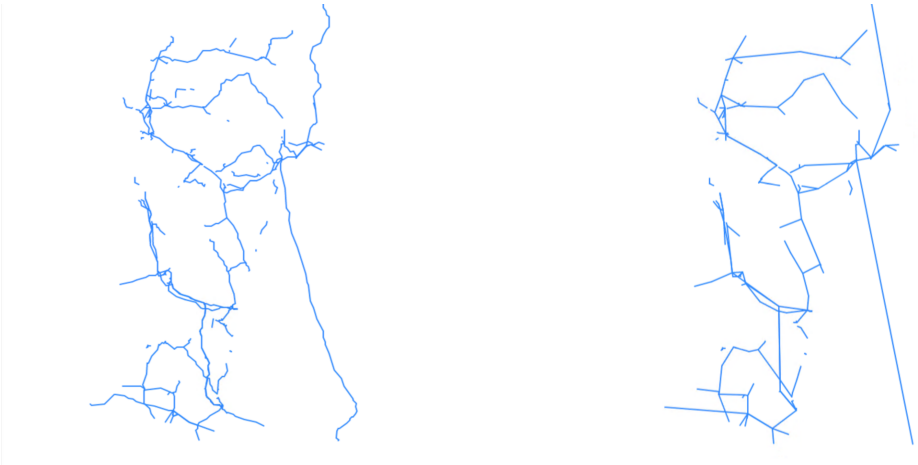


Figure 5: The power transmission grid in Vermont, shown in the PGAdmin geometry viewer before (left) and after (right) cleaning.

4.2 Sliding Box Algorithm

The above steps become computationally expensive for large geographies like the United States. To make computation feasible, we apply the SQL algorithm iteratively to subset overlapping boxes of the geography. This is the "sliding box" aspect of the algorithm. We make the boxes overlap to ensure that no connections across boxes are unprocessed.

4.3 Output

As noted in Section 3.3, the abstraction algorithm outputs three files. We describe these files in more detail below.

4.3.1 Vertices

`Vertices.csv` describes the stations. It contains columns for an identification number, longitude and latitude coordinates, a list of all types of station objects in the cluster, and

a corresponding list of voltages if available. Note that these stations are used only in the SQL algorithm for the abstraction of the power lines because the lines need to be split and merged. We use the Global Power Plant database to measure power assets by region.

station_id	longitude	latitude	power	voltage
1	-74.54	39.15	plant	350000
2	-118.13	33.23	station, substation	50000
3	-98.22	36.66	substation	150000
⋮	⋮	⋮	⋮	⋮

Table 1: Example vertices data

4.3.2 Links

`Links.csv` describes the lines. It contains columns for a line identification number, longitude and latitude coordinates of both endpoints, OSM tag values, the number of cables in each line in the merge, a corresponding list of voltages, and wire information We are primarily concerned with the geographic information and the voltage. “Power”, “cables” and “wires” data are not used in our analysis, however they are preserved in case they have use cases to other researchers. “Power” refers to the value listed in the OSM tag. “Cables" data is very sparse and denotes the number of cables in the power line. “Wire" contains information on whether the line uses single, double, triple, or quadruple wires. Other than the ID and geographic data, each column may contain lists of items instead of scalar or single-string entries if the line was merged from smaller lines.

line_id	lon_1	lat_1	lon_2	lat_2	power	cables	voltage	wires
1	-74.54	39.15	-76.32	39.15	line	3	90000	double
2	-118.13	33.23	-120.10	33.23	line	6	135000	quad
3	-98.22	36.66	-97.41	36.66	cable	1	50000	single
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 2: Example links data

4.3.3 Intersections

`Intersections.csv` describes which lines intersect which stations. It contains only two columns, one for the station identification number, and one for the line. It does not contain geographic information.

station_id	line_id
1	100
2	99
3	98
⋮	⋮

Table 3: Example intersections data

SECTION 5

Capacity Estimation

The Open Street Map data only contains voltage and distance information for power lines the while model requires a measure of line capacity. As such we use a fit of the St. Clair curve as per Gutman et. al (1979) [1] to estimate line capacity using voltage and distance. Our curve is a non-parametric fit of the original curve from Gutman et. al.

5.1 Clean Voltages

Voltage data from the abstraction algorithm is aggregated across a number of lines when those lines have been merged. The individual voltages are collected in a list separated by semicolons. The first step to estimating capacities is to destring, separate, and sum the voltages for each line. Since the data is open source, there are entry errors that need to be corrected. We assume lines with voltages recorded under 1000 are recorded in kilovolts instead of volts and all voltages above are recorded in volts. We set all values to kilovolts. Some voltage records contain strings that are hard to account for. We set these voltages to missing values. We group lines into 25 bins based on their length and calculated mean voltages for each bin. Lines that contain only missing voltage values are imputed with the mean voltage of their distance bin.

5.2 St. Clair Curve

The St. Clair curve is a tool used in electrical engineering to estimate load capacity limits using limited information about the power line. It relates the distance of a power line to a standardized unit of capacity call that is independent of the line voltage. We can then use voltage information of each line to back out values of capacity in the more familiar megawatt units.

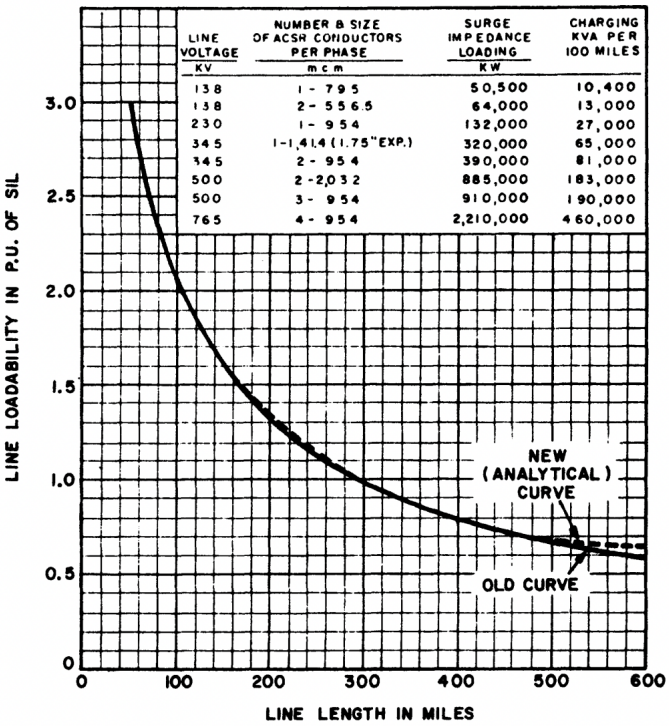


Figure 6: St. Clair Curve from Gutman et. al.

We use the following fitted equation for the St. Clair curve which matches the above curve. Here d_i is the distance of line i in miles for comparability with Gutman et. al.

$$u_i = 0.1021681 + \frac{11.43091 - 0.1021681}{1 + \left(\frac{d_i}{12.49187}\right)^{0.7700907}} \tag{1}$$

For a given voltage, one standardized unit u_i is equal to its corresponding surge impedance loading (SIL) in MW as detailed in the table in Figure 6. In general the SIL of a line is given by:

$$SIL = \frac{V_i^2}{z} \tag{2}$$

Here z is the surge impedance of a line. We use SIL values and the voltages listed in Figure 6 table along with equation (2) to back out values for the surge impedance of various voltages. We take the most conservative (ie. largest) value of approximately 400 ohms for all lines. Thus we have z, V_i and d_i for each line i. Combining equations (1) and (2) we can estimate capacities c_i in MW:

$$c_i = \frac{V_i^2}{z}\left(0.1021681 + \frac{11.43091 - 0.1021681}{1 + \left(\frac{d_i}{12.49187}\right)^{0.7700907}}\right)$$

SECTION 6

Aggregation Process

The abstracted data is more granular than we need for region-level model simulation. As such we aggregate power line and power asset data by region for our analysis. The size of the region depends on the available data for each country. For example in the United States the regions are commuting zones, whereas in China the regions are provinces. This process is done entirely in Python.

6.1 Aggregate Assets

We use the Global Power Plant (GPP) database from the World Resources Institute which includes data on the capacity, fuel types, and location of global power assets. To aggregate the assets by region, we identify which region intersects the coordinates of a given power plant and sum the capacity over all power plants in a region. We also decompose the data by fuel type to measure total renewable and non-renewable capacity by region. The aggregate asset data is stored in `csr_aggcap.csv`.

csr_id	capacity_mw	rnw_capacity$_{mw}$	ffl_capacity_mw
1	1810.4	55.4	1755
2	709.1	47.6	661.5
3	48.5	11.9	36.6
⋮	⋮	⋮	⋮

Table 4: Example of aggregated assets data

6.2 Aggregate Lines

We consider each line in `links_cap.csv` and identify which region-polygon the line endpoint intersects. Then for a given pair of regions A and B, we sum the voltages, capacities, and number of lines over all lines between those regions. This produces a single aggregate “power line" representation, or edge, between each region if at least one actual line exists. The end points of the edge are set to the centroids of the intersecting regions.

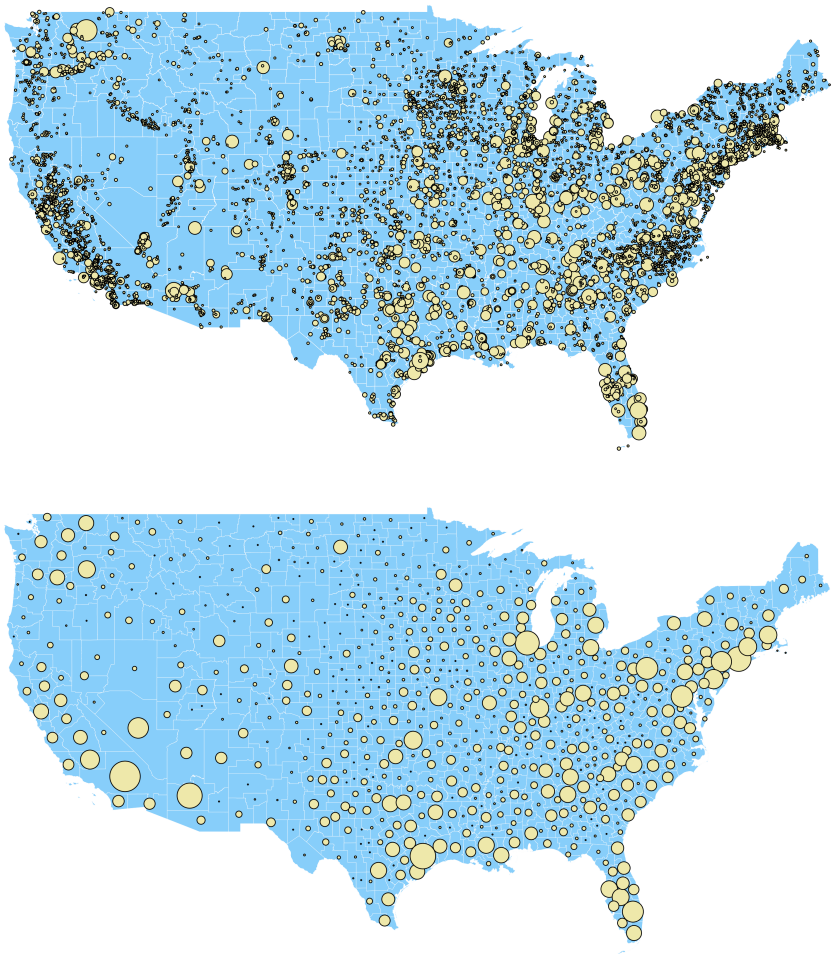


Figure 7: Power assets before (top) and after (bottom) aggregation. Size of bubble denotes capacity.

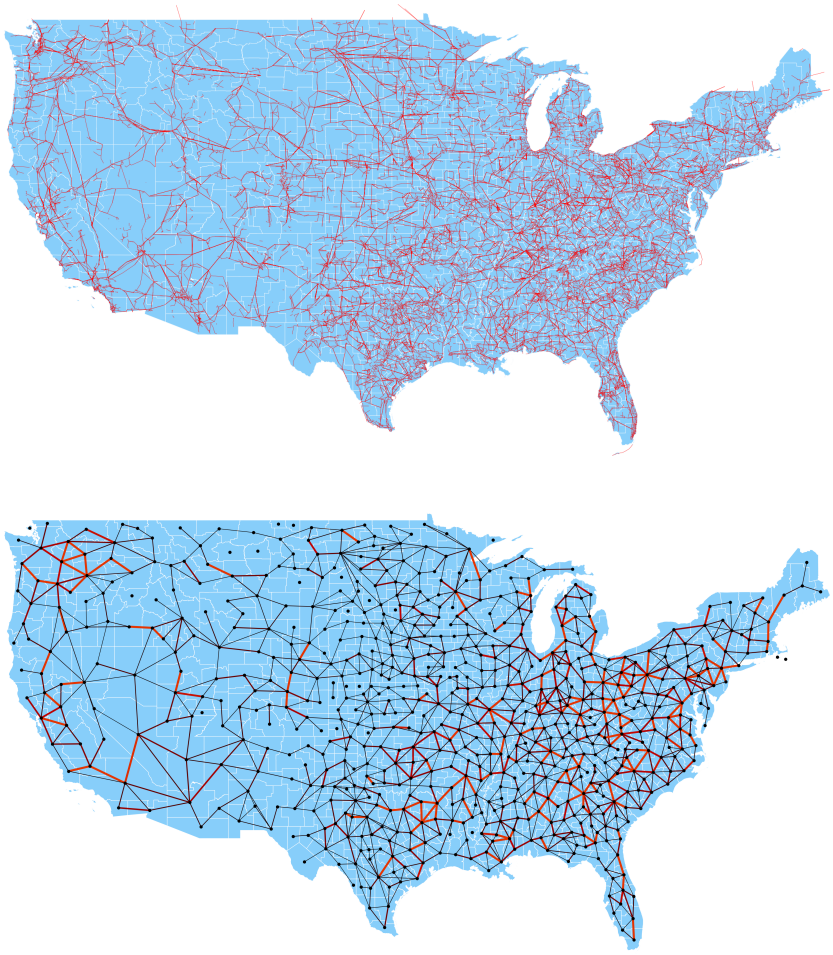


Figure 8: Power line data before (top) and after (bottom) aggregation. Redder lines on the aggregated map indicates greater capacity.

csr_id_A	csr_id_B	num_of_edge	voltage_kv	max_capacity_mw	distance_km
1	13	5	819.76	2835.92	56.35
1	25	4	1317	5244.89	103.29
2	37	1	500	2194.73	56.95
⋮	⋮	⋮	⋮	⋮	⋮

Table 5: Example of aggregated lines data

6.3 Construct Matrices

An adjacency matrix is a matrix where the rows and columns indicate regions and the entries indicate the presence of an edge between the row and column regions. We construct several adjacency matrices for our simulations using the Pandas crosstab function with an indicator, voltage, capacity, number of lines, and distance as the entries. Distance is defined as the distance between centroids of the regions, not total distance of the aggregated lines.

SECTION 7

Grid Improvements

In order to assess the value and impact of upgrades to the existing power grid, we incorporate four major planned improvements to the power grid. These high voltage lines are the TransWest Express line from Wyoming to Nevada planned to begin construction in 2023, the SunZia line in the Southwest also planned for 2023, the GrainBelt Express in the Midwest planned for 2024, and the Champlain-Hudson Power Express which began construction in 2022. The intermediary stations are sourced from company websites and media outlets. They are then hard-coded into the grid during the aggregation stage. Due to some ambiguity from these data sources, a few of assumptions are made about where stations for local power distribution lie on the lines, which are detailed below.

7.1 TransWest Express Line

The TransWest Express Line, shown in Figure 9, is a 500 kV line running from Rawlins, Wyoming to Las Vegas, Nevada [16]. It consists of three segments, Rawlins-Utah, Utah-Crystal, and Crystal-Eldorado. The Crystal-Eldorado segment is fully contained within one commuting zone and as such does not affect the aggregated network. The Rawlins-Utah segment has a capacity of 3000 MW while the Utah-Crystal segment has 1500 MW [16]. We create links between the commuter zones that contain the relevant endpoints, ignoring the series compensation station in Iron County. The TransWest Express is expected to cost around 3 billion USD [8].

7.2 SunZia Transmission Line

The SunZia Transmission Line, depicted in Figure 10, is a 525 kV, 3000 MW capacity line between Pinal County, Arizona and Lincoln County, New Mexico [11]. It consists of four segments. The first segment runs from Pinal County to just north of Willcox, Arizona, the second segment connects Willcox to Deming, New Mexico, the third joins Deming to Socorro and the final segment runs from Socorro to Corona in Lincoln County. Public company information only indicates aggregate voltage and capacity of the line so we assume constant capacity and voltage across the line. Alternate maps indicate a

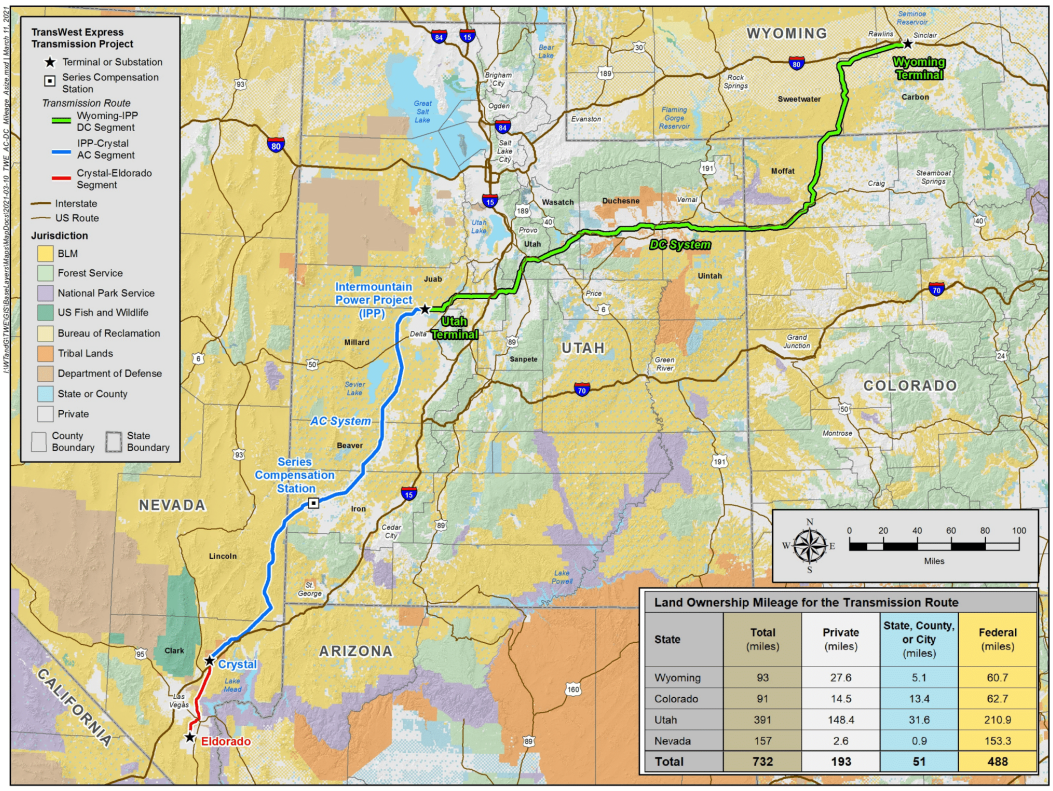


Figure 9: Map of the planned TransWest Express Line. [16]

substation at Lordsburg, New Mexico in the Willcox-Deming portion of the line, however Lordsburg resides in the same commuter zone as Deming. While the main map indicates no substation at some connection points, alternate sources suggest there are substations at each connection between segments except for at Socorro [4]. We assume there are substations at each connection including at Socorro. The line is being developed in tandem with wind assets, with a total estimated cost of 8 billion USD [6]. The line itself is expected to cost between 2 to 2.5 billion USD [7] [3].

7.3 Grain Belt Express Line

The Grain Belt Express Line joins Kansas, Missouri, Illinois, and Indiana, as depicted in Figure 11. It is a 5000 MW, 600 kV line from Dodge City, Kansas to the border between Illinois and Indiana. While the primary map indicates only three segments, we divide the line into five. Media sources indicate that there is an additional delivery station east of Moberly, as shown in Figure 12. Furthermore, official descriptions of the line indicate that it transmits to both Illinois and Indiana [15], which is supported by the fact that the line is meant to join ultimately with the eastern grid as indicated by Figure 11. As such we assume there are six stations with segments in between; from Dodge City, Kansas to Monroe County, Missouri, on to Perry, Missouri then to West Union, Illinois, terminating in Sullivan County, Indiana. The sixth station is not part of the main line but rather a

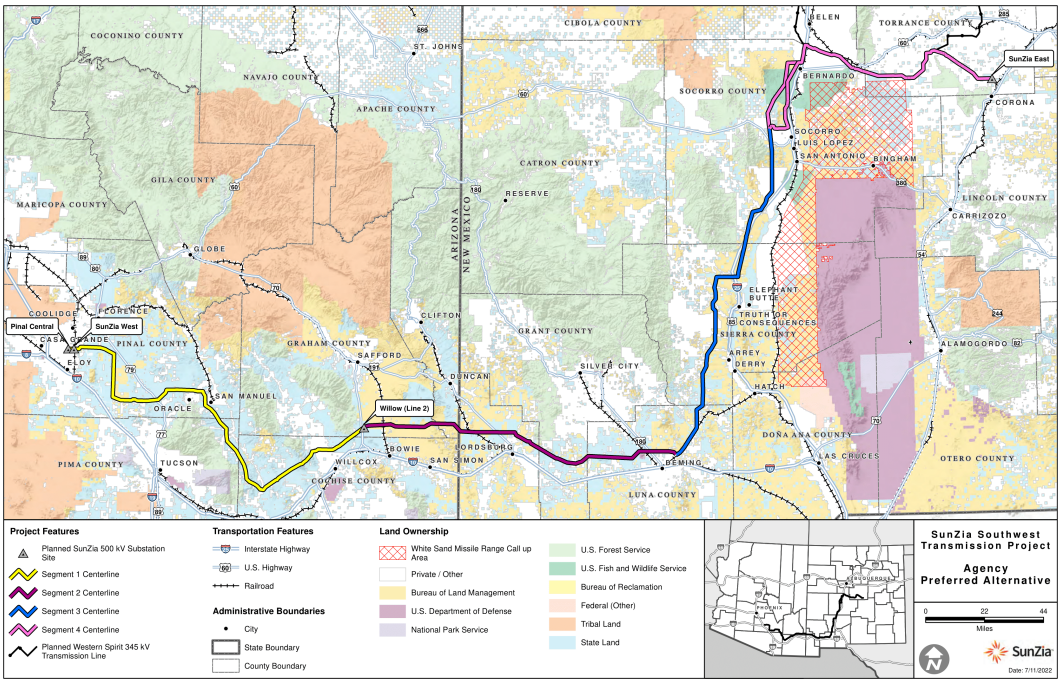


Figure 10: Map of the planned SunZia Transmission Line. [11]

terminal node off of a branch from Monroe to Kingdom City, Missouri. Once again we assume constant voltage and capacity, save for the branch to Kingdom City, which has a capacity of 2500 MW [23]. The Grain Belt Express has an estimated cost of 2 billion USD [10].

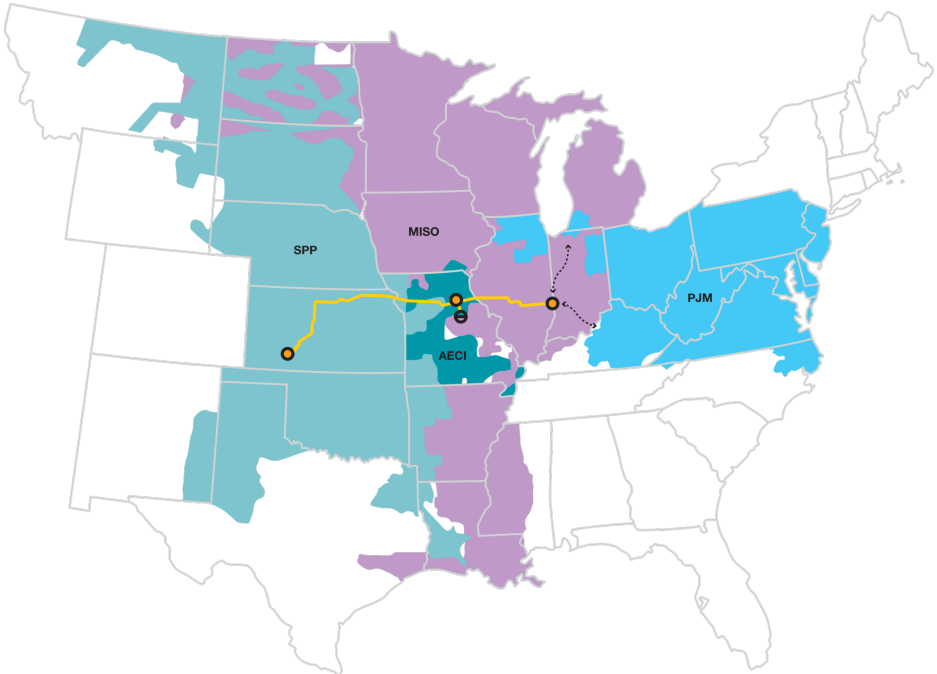


Figure 11: Map of the planned Grain Belt Express Line. [15]

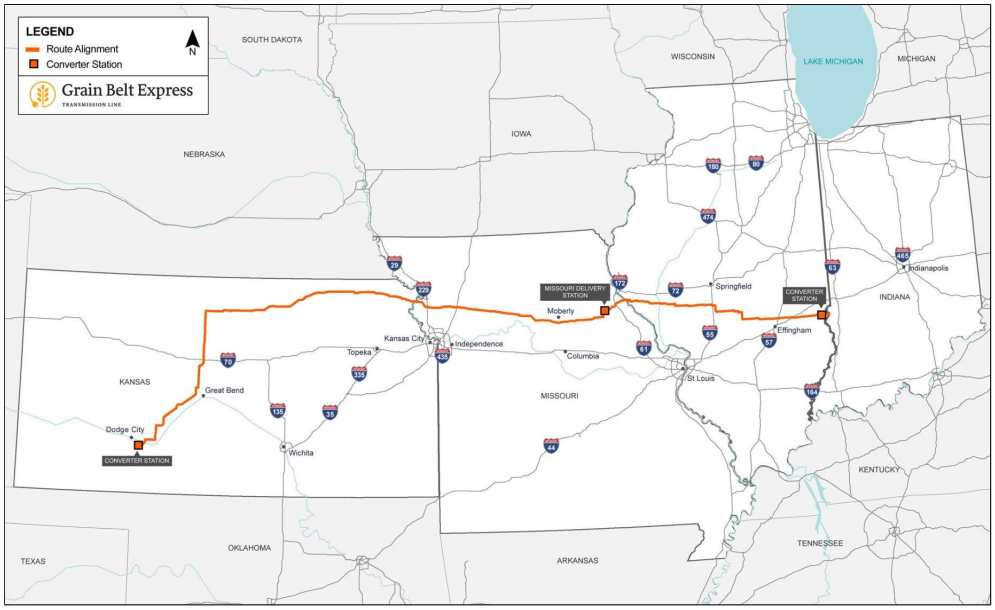


Figure 12: Media map of the Grain Belt Express. [5]

7.4 Champlain-Hudson Power Express Line

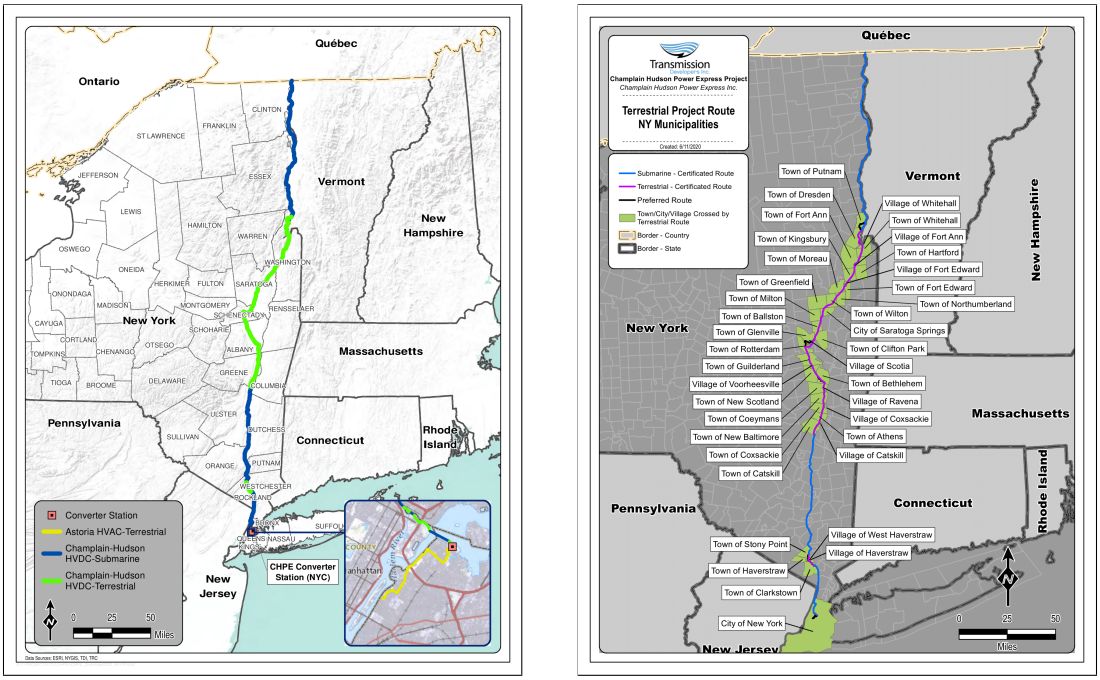


Figure 13: Maps of the Champlain-Hudson Power Express Line. [14]

The Champlain-Hudson Power Express is a transmission line under construction running from Quebec, Canada to New York City with a capacity of 1250 MW [14] and a voltage of 320 kV [2]. The line runs underground through Lake Champlain, upstate New York, and the Hudson River, with above-ground connections at two points in New York state as shown in Figure 13. We assume that the line connects to stations in the above-ground

locations. While the map suggests that the line joins many smaller upstate towns as indicated by the green segments, each disjoint segment of municipalities lies in the same commuter zone, simplifying the analysis. As such we model the Champlain-Hudson route as having four stations, one in Quebec, another in upstate New York, which includes Putnam to Catskill, another downstate, which includes Stony Point to Clarkstown, and finally one in New York City. Again we assume constant capacity and voltage. The Champlain-Hudson line is expected to cost 3 billion USD [9].

8
SECTION

References

[1] Richard Gutman, Raymond D. Dunlop, and Peter P. Marchenko. “Analytical Development of Loadability Characteristics for EHV and UHV Transmission Lines”. In: *Institute of Electrical and Electronics Engineers* PAS-98.2 (1979), pp. 606–617.

[2] Annie St-Pierre. “Hydro-Québec export project to New York”. In: *Argent* (Jan. 29, 2013). URL: https://archive.ph/20130219184408/http://argent.canoe.ca/lca/affaires/quebec/archives/2013/01/projet-exportation-hydro-quebec-new-york.html#selection-959.0-959.16.

[3] Tina Casey. “$2 Billion SunZia Solar-Wind Project Gets Green Light From White Sands”. In: *Clean Technica* (May 28, 2014). URL: https://cleantechnica.com/2014/05/28/2-billion-sunzia-wind-solar-project-gets-green-light/.

[4] Steve Terrell. “PRC rejects SunZia transmission line project”. In: *Las Cruces Sun News* (Sept. 6, 2018). URL: https://www.lcsun-news.com/story/news/local/new-mexico/2018/09/06/prc-rejects-sunzia-transmission-line-project/1211636002/.

[5] Clayton Anderson. “Grain Belt Express power line looks to push forward”. In: *News Press Now* (June 20, 2021). URL: https://www.newspressnow.com/news/local_news/business/grain-belt-express-power-line-looks-to-push-forward/article_caf36d4e-cfaf-11eb-9a10-33dadd3a126e.html.

[6] Tina Casey. “Massive $8 Billion SunZia Renewable Energy Project Is On Again, For Now”. In: *Clean Technica* (July 19, 2022). URL: https://cleantechnica.com/2022/07/19/massive-8-billion-sunzia-renewable-energy-project-is-on-again-for-now/.

[7] Daniel Moore. “Renewable-Rich States Push Feds to End Electricity Grid Logjam”. In: *Bloomberg Law* (June 15, 2022). URL: https://news.bloomberglaw.com/environment-and-energy/renewable-rich-states-push-feds-to-end-electricity-grid-logjam.

[8] Josh Saul, Naureen Malik, and Dave Merrill. “The Clean-Power Megaproject Held Hostage by a Ranch and a Bird”. In: *Bloomberg* (Apr. 12, 2022). URL: https://www.bloomberg.com/graphics/2022-clean-energy-power-lines-transwest-wind-maps-private-property/#xj4y7vzkg.

[9] Will Wade. “NYC’s Big Clean Energy Plan Is Under Attack From One-Time Advocate”. In: *Bloomberg* (Feb. 18, 2022). URL: https://www.bloomberg.com/news/articles/2022-02-18/champlain-hudson-power-express-plan-to-bring-hydro-power-to-nyc-faces-pushback.

[10] Missouri Public Service Commission. *Grain Belt Express Project Fact Sheet*. URL: https://www.efis.psc.mo.gov/mpsc/commoncomponents/viewdocument.asp?DocId=935827705.

[11] Pattern Energy. *SunZia Transmission*. URL: https://patternenergy.com/projects/sunzia-transmission/.

[12] Geofabrik. *OpenStreetMap Data Extracts*. https://download.geofabrik.de/index.html.

[13] World Resources Institute. *Global Power Plant Database*. https://datasets.wri.org/dataset/globalpowerplantdatabase.

[14] Champlain Hudson Power Express LLC. *Champlain Hudson Power Express*. URL: https://chpexpress.com/.

[15] Grain Belt Express LLC. *Grain Belt Express*. URL: https://grainbeltexpress.com/.

[16] TransWest Express LLC. *TransWest Express*. URL: https://www.transwestexpress.net/.

[17] Open Street Map. *Osmosis/Quick Install (Windows)*. URL: https://wiki.openstreetmap.org/wiki/Osmosis#How_to_install.

[18] Open Street Map. *Planet-OSM*. https://planet.openstreetmap.org/.

[19] Wided Medjroubi. *SciGRID Power*. https://www.power.scigrid.de/.

[20] Osm2pgsql. *Installation*. https://osm2pgsql.org/doc/install.html.

[21] PGAdmin. *Download*. https://www.pgadmin.org/download/.

[22] PostgreSQL. *Downloads*. https://www.postgresql.org/download/.

[23] Federal Register. *Notice of Intent To Prepare an Environmental Impact Statement for the Grain Belt Express Transmission Line Project*. URL: https://www.federalregister.gov/documents/2022/12/16/2022-27099/notice-of-intent-to-prepare-an-environmental-impact-statement-for-the-grain-belt-express.