Project 1 Report

2023-03-07

Names, EID: Solomon Leo, sal4294, Vardhan Koripally vrk385

Introduction
Texas is known as the energy hub of the United States. In 2021, Texas produced more energy than any other state (Texas.gov). However, unlike the rest of the United States, Texas is isolated with its own power grid. This keeps the Texas grid free from federal regulation and lowers wholesale electricity prices, but it also makes it incredibly vulnerable to power outages because it's not interconnected with the rest of the United States. This was seen in February 2021, when Winter Storm Uri caused massive power outages to homes across Texas. This report analyzes the reliability and carbon footprint of the Texas power grid in 2022.

There are three data sets used in this analysis, all of them come from the US Electricity Information Administration (EIA).
The first two data sets give an hourly grid monitor of the United States in 2022 by balancing authority. The first set is from January to June, and the second data set is from July to December. Each unique row represents a certain hour, and each unique value is a number of Megawatts. Some unique columns are electricity interchange between balancing authorities, demand, and generation by type (wind, solar, coal, etc).
The third data set is also an hourly grid monitor, but specifically for the Electricity Reliability Council of Texas (ERCOT) balancing authority from 2015 to 2023. This data set gives CO2 emissions by generation type (coal, natural gas, and other sources). Each unique row represents a certain hour, and each unique value is a number of metric tons of CO2. This data set also gives carbon intensity, a metric of how "clean" the generated electricity is in terms of carbon dioxide emissions. This value is given in pounds of CO2 per kilowatt-hour of electricity.

We have several categorical variables such as the date and balancing authority. Every other variable such as emissions, generation, and demand are numeric.

The goal of this analysis is to evaluate the Texas energy grid's reliability and carbon footprint in 2022. Since all data sets contain the time at end of hour in UTC time which we will use as the keys to join the data sets together. We will join the two EIA data sets by attaching the data from January to June with the data from July to December to make a year. It's hypothesized that generation by fossil fuels are positively correlated with carbon intensity, and that as demand raises, generation is less likely to meet it. The following research questions will be answered in this analysis:
How often does generation fail to meet demand in the Texas power grid?
How does the carbon emissions from electricity generation vary over the year?

Joining
First, the data is read into R and stripped of special characters to make manipulation easier.

```
# Read all data from spreadsheet and csv files and remove all spaces, periods, and other special cha
racters from all column names
EIA1 = read_csv('EIA930_BALANCE_2022_Jan_Jun.csv',show_col_types = FALSE) |> as_tibble() |> select_a
ll(~gsub("\\s+|\\.", "_", .)) |>
  select_all(tolower) |> select_all(~gsub(":","",.))
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
EIA2 = read_csv('EIA930_BALANCE_2022_Jul_Dec.csv',show_col_types = FALSE) |> as_tibble() |> select_a
ll(~gsub("\\s+|\\.", "_", .)) |>
  select_all(tolower) |> select_all(~gsub(":","",.))
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
CARBON = read_csv('ERCO.csv',show_col_types = FALSE) |> as_tibble() |> select_all(~gsub("\\s+|\\.",
"_", .)) |>
  select_all(tolower) |> select_all(~gsub(":","",.))
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

These parsing issues can be ignored. `EIA1` and `EIA2` are the first two data sets, and `CARBON` is the third data set.

```
# Look at the first few rows of each data set
EIA1 |> head()
```

```
## # A tibble: 6 × 42
##   balancing_au…¹ data_…² hour_…³ local…⁴ utc_t…⁵ deman…⁶ deman…⁷ net_g…⁸ total…⁹
##   <chr>          <chr>     <dbl> <chr>   <chr>     <dbl>   <dbl>   <dbl>   <dbl>
## 1 AECI           01/01/…       1 01/01/… 01/01/…    2235    2251    1986    -265
## 2 AECI           01/01/…       2 01/01/… 01/01/…    2217    2208    2039    -169
## 3 AECI           01/01/…       3 01/01/… 01/01/…    2193    2204    2080    -124
## 4 AECI           01/01/…       4 01/01/… 01/01/…    2255    2234    2110    -124
## 5 AECI           01/01/…       5 01/01/… 01/01/…    2325    2287    2138    -149
## 6 AECI           01/01/…       6 01/01/… 01/01/…    2419    2378    2218    -160
## # … with 33 more variables: `sum(valid_dibas)_(mw)` <dbl>,
## #   `demand_(mw)_(imputed)` <dbl>, `net_generation_(mw)_(imputed)` <dbl>,
## #   `total_interchange_(mw)_(imputed)` <lgl>, `demand_(mw)_(adjusted)` <dbl>,
## #   `net_generation_(mw)_(adjusted)` <dbl>,
## #   `total_interchange_(mw)_(adjusted)` <dbl>,
## #   `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>, …
```

```
EIA2 |> head()
```

```
## # A tibble: 6 × 42
##   balancing_au…¹ data_…² hour_…³ local…⁴ utc_t…⁵ deman…⁶ deman…⁷ net_g…⁸ total…⁹
##   <chr>          <chr>     <dbl> <chr>   <chr>     <dbl>   <dbl>   <dbl>   <dbl>
## 1 AECI           07/01/…       1 07/01/… 07/01/…    2669    2626    2330    -296
## 2 AECI           07/01/…       2 07/01/… 07/01/…    2492    2451    2141    -310
## 3 AECI           07/01/…       3 07/01/… 07/01/…    2372    2321    2015    -306
## 4 AECI           07/01/…       4 07/01/… 07/01/…    2296    2253    1903    -350
## 5 AECI           07/01/…       5 07/01/… 07/01/…    2258    2229    1942    -287
## 6 AECI           07/01/…       6 07/01/… 07/01/…    2301    2266    1936    -330
## # … with 33 more variables: `sum(valid_dibas)_(mw)` <dbl>,
## #   `demand_(mw)_(imputed)` <dbl>, `net_generation_(mw)_(imputed)` <dbl>,
## #   `total_interchange_(mw)_(imputed)` <lgl>, `demand_(mw)_(adjusted)` <dbl>,
## #   `net_generation_(mw)_(adjusted)` <dbl>,
## #   `total_interchange_(mw)_(adjusted)` <dbl>,
## #   `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>, …
```

```
CARBON |> head()
```

```
## # A tibble: 6 × 70
##   ba    utc_time     local…¹  hour local…² time_…³ gener…⁴    df     d    ng    ti
##   <chr> <chr>        <chr>   <dbl> <chr>   <chr>   <chr>   <dbl> <dbl> <dbl> <dbl>
## 1 ERCO  01Jul2015… 01Jul2…     1 01Jul2… Central N       39708 37456 37462     6
## 2 ERCO  01Jul2015… 01Jul2…     2 01Jul2… Central N       37338 35119 35124     4
## 3 ERCO  01Jul2015… 01Jul2…     3 01Jul2… Central N       35697 33638 33642     5
## 4 ERCO  01Jul2015… 01Jul2…     4 01Jul2… Central N       34772 32798 32805     6
## 5 ERCO  01Jul2015… 01Jul2…     5 01Jul2… Central N       34773 32805 32812     7
## 6 ERCO  01Jul2015… 01Jul2…     6 01Jul2… Central N       36046 34121 34128     7
## # … with 59 more variables: imputed_d <dbl>, imputed_ng <dbl>,
## #   imputed_ti <lgl>, adjusted_d <dbl>, adjusted_ng <dbl>, adjusted_ti <dbl>,
## #   ng_col <dbl>, ng_ng <dbl>, ng_nuc <dbl>, ng_oil <lgl>, ng_wat <dbl>,
## #   ng_sun <dbl>, ng_wnd <dbl>, ng_oth <dbl>, ng_unk <lgl>,
## #   imputed_col_gen <dbl>, imputed_ng_gen <dbl>, imputed_nuc_gen <dbl>,
## #   imputed_oil_gen <lgl>, imputed_wat_gen <dbl>, imputed_sun_gen <dbl>,
## #   imputed_wnd_gen <dbl>, imputed_oth_gen <dbl>, imputed_unk_gen <lgl>, …
```

```
# Look at the number of rows in each data set
nrow(EIA1)
```

```
## [1] 273621
```

```
nrow(EIA2)
```

```
## [1] 275330
```

```
nrow(CARBON)
```

```
## [1] 67369
```

EIA 1 and EIA2 have 42 columns, while `CARBON` has 70 columns. EIA1 has 273,621 rows, `EIA2` has 275,330 rows, and `CARBON` has 67,369 rows.

Before joining the three data sets, they must be filtered and cleaned. `EIA1` and `EIA2` have data for 2022, however they have data for all of Texas and must be filtered to only the Texas balancing authority, ERCOT. The `CARBON` data set only has data for ERCOT, but it must be filtered to only contain data from 2022. `EIA1` and `EIA2` can easily be joined by using the `rbind()` function because they contain the same columns, and `EIA2` is simply a continuation of `EIA1` .

```
# Take only data from 2022 from the ERCO data and remove all columns that are NA
CARBON = CARBON |> filter(substr(local_date, 6,9) == '2022') |> select_if(~ !any(is.na(.)))

# Join EIA1 and EIA2 to form an entire year then select only ERCO, finally remove all NA columns
EIA = rbind(EIA1, EIA2) |> dplyr::filter(balancing_authority == 'ERCO') |> select_if(~ !any(is.na
(.)))

# Look at cleaned data
EIA |> head()
```

```
## # A tibble: 6 × 28
##   balancing_au…¹ data_…² hour_…³ local…⁴ utc_t…⁵ deman…⁶ deman…⁷ net_g…⁸ total…⁹
##   <chr>          <chr>     <dbl> <chr>   <chr>     <dbl>   <dbl>   <dbl>   <dbl>
## 1 ERCO           01/01/…       1 01/01/… 01/01/…   37515   38123   37963    -159
## 2 ERCO           01/01/…       2 01/01/… 01/01/…   37119   37122   36757    -365
## 3 ERCO           01/01/…       3 01/01/… 01/01/…   36864   35936   35826    -109
## 4 ERCO           01/01/…       4 01/01/… 01/01/…   36308   35132   35033     -98
## 5 ERCO           01/01/…       5 01/01/… 01/01/…   35776   34602   34582     -20
## 6 ERCO           01/01/…       6 01/01/… 01/01/…   34917   34451   34330    -120
## # … with 19 more variables: `sum(valid_dibas)_(mw)` <dbl>,
## #   `demand_(mw)_(adjusted)` <dbl>, `net_generation_(mw)_(adjusted)` <dbl>,
## #   `total_interchange_(mw)_(adjusted)` <dbl>,
## #   `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>,
## #   `net_generation_(mw)_from_nuclear` <dbl>,
## #   `net_generation_(mw)_from_hydropower_and_pumped_storage` <dbl>, …
```

```
CARBON |> head()
```

```
## # A tibble: 6 × 52
##   ba    utc_time   local…¹  hour local…² time_…³ gener…⁴    df     d    ng    ti
##   <chr> <chr>      <chr>   <dbl> <chr>   <chr>   <chr>   <dbl> <dbl> <dbl> <dbl>
## 1 ERCO  01Jan2022… 01Jan2…     1 01Jan2… Central N       37515 38123 37963  -159
## 2 ERCO  01Jan2022… 01Jan2…     2 01Jan2… Central N       37119 37122 36757  -365
## 3 ERCO  01Jan2022… 01Jan2…     3 01Jan2… Central N       36864 35936 35826  -109
## 4 ERCO  01Jan2022… 01Jan2…     4 01Jan2… Central N       36308 35132 35033   -98
## 5 ERCO  01Jan2022… 01Jan2…     5 01Jan2… Central N       35776 34602 34582   -20
## 6 ERCO  01Jan2022… 01Jan2…     6 01Jan2… Central N       34917 34451 34330  -120
## # … with 41 more variables: adjusted_d <dbl>, adjusted_ng <dbl>,
## #   adjusted_ti <dbl>, ng_col <dbl>, ng_ng <dbl>, ng_nuc <dbl>, ng_wat <dbl>,
## #   ng_sun <dbl>, ng_wnd <dbl>, ng_oth <dbl>, adjusted_col_gen <dbl>,
## #   adjusted_ng_gen <dbl>, adjusted_nuc_gen <dbl>, adjusted_wat_gen <dbl>,
## #   adjusted_sun_gen <dbl>, adjusted_wnd_gen <dbl>, adjusted_oth_gen <dbl>,
## #   cen <dbl>, swpp <dbl>, subregion_coas <dbl>, subregion_east <dbl>,
## #   subregion_fwes <dbl>, subregion_nrth <dbl>, subregion_ncen <dbl>, …
```

These data sets are now both filtered to 2022 and only contain columns with values. However, some columns are redundant and contain no valuable information for this analysis. For example, both data sets have columns representing balancing authority and local time, which aren't needed because this data is only for one balancing authority, and UTC time is already available. Both data sets also have separate data date columns, but they are redundant because the UTC time column contains all date and time information.

The data sets are further reduced to only the necessary information using the `select` function.

```
# Select useful columns
EIAf = EIA |> select(c(5:8,14:20))
ERCOf = CARBON |> select(c(2,31:48,51,52))

# Look again
EIAf |> head()
```

```
## # A tibble: 6 × 11
##   utc_time_at_…¹ deman…² deman…³ net_g…⁴ net_g…⁵ net_g…⁶ net_g…⁷ net_g…⁸ net_g…⁹
##   <chr>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 01/01/2022 7:…   37515   38123   37963   10957    9739    5099      33       0
## 2 01/01/2022 8:…   37119   37122   36757   10128    8564    5099       8       0
## 3 01/01/2022 9:…   36864   35936   35826    8944    7351    5099       7       0
## 4 01/01/2022 10…   36308   35132   35033    7165    6259    5099       7       0
## 5 01/01/2022 11…   35776   34602   34582    5609    5454    5100       6       0
## 6 01/01/2022 12…   34917   34451   34330    5642    5356    5100       6       0
## # … with 2 more variables: `net_generation_(mw)_from_wind` <dbl>,
## #   `net_generation_(mw)_from_other_fuel_sources` <dbl>, and abbreviated
## #   variable names ¹utc_time_at_end_of_hour, ²`demand_forecast_(mw)`,
## #   ³`demand_(mw)`, ⁴`net_generation_(mw)`, ⁵`net_generation_(mw)_from_coal`,
## #   ⁶`net_generation_(mw)_from_natural_gas`,
## #   ⁷`net_generation_(mw)_from_nuclear`,
## #   ⁸`net_generation_(mw)_from_hydropower_and_pumped_storage`, …
```

```
ERCOf |> head()
```

```
## # A tibble: 6 × 21
##   utc_time       subre…¹ subre…² subre…³ subre…⁴ subre…⁵ subre…⁶ subre…⁷ subre…⁸
##   <chr>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 01Jan2022 7:0…   11593    1348    3875     860    9544    3336    6268    1321
## 2 01Jan2022 8:0…   11349    1292    3805     851    9177    3289    6055    1340
## 3 01Jan2022 9:0…   11007    1293    3841     854    8756    3167    5772    1276
## 4 01Jan2022 10:…   10851    1241    3896     835    8538    3057    5530    1201
## 5 01Jan2022 11:…   10646    1216    3954     827    8481    2983    5419    1085
## 6 01Jan2022 12:…   10628    1204    3902     835    8465    2941    5368    1106
## # … with 12 more variables: co2_factor_col <dbl>, co2_factor_ng <dbl>,
## #   co2_factor_oil <dbl>, co2_emissions_col <dbl>, co2_emissions_ng <dbl>,
## #   co2_emissions_other <dbl>, co2_emissions_generated <dbl>,
## #   co2_emissions_imported <dbl>, co2_emissions_exported <dbl>,
## #   co2_emissions_consumed <dbl>,
## #   co2_emissions_intensity_for_generated_electricity <dbl>,
## #   co2_emissions_intensity_for_consumed_electricity <dbl>, and abbreviated …
```

Since both data sets are time series data, they can be joined by the UTC time at the end of the hour. In this case the keys will be the date. To make sure that this approach will work, the `anti_join()` function checks to see if any keys are missing.

```
# Check to see if keys match
EIAf |> anti_join(ERCOf, by = c('utc_time_at_end_of_hour' = 'utc_time'))
```

```
## # A tibble: 8,760 × 11
##    utc_time_at…¹ deman…² deman…³ net_g…⁴ net_g…⁵ net_g…⁶ net_g…⁷ net_g…⁸ net_g…⁹
##    <chr>           <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 01/01/2022 7…   37515   38123   37963   10957    9739    5099      33       0
##  2 01/01/2022 8…   37119   37122   36757   10128    8564    5099       8       0
##  3 01/01/2022 9…   36864   35936   35826    8944    7351    5099       7       0
##  4 01/01/2022 1…   36308   35132   35033    7165    6259    5099       7       0
##  5 01/01/2022 1…   35776   34602   34582    5609    5454    5100       6       0
##  6 01/01/2022 1…   34917   34451   34330    5642    5356    5100       6       0
##  7 01/01/2022 1…   33851   34571   34385    5531    5127    5101      30       0
##  8 01/01/2022 2…   33894   34635   34406    5665    5207    5101      30       8
##  9 01/01/2022 3…   35578   35606   35475    5676    5936    5101      30    1349
## 10 01/01/2022 4…   37141   37917   37731    5582    6469    5102      30    3459
## # … with 8,750 more rows, 2 more variables:
## #   `net_generation_(mw)_from_wind` <dbl>,
## #   `net_generation_(mw)_from_other_fuel_sources` <dbl>, and abbreviated
## #   variable names ¹utc_time_at_end_of_hour, ²`demand_forecast_(mw)`,
## #   ³`demand_(mw)`, ⁴`net_generation_(mw)`, ⁵`net_generation_(mw)_from_coal`,
## #   ⁶`net_generation_(mw)_from_natural_gas`,
## #   ⁷`net_generation_(mw)_from_nuclear`, …
```

Although the IDs exist in both data sets, the two date formats are incompatible with each other, which means it's not possible to join by date. To resolve this, a new column will be added using the `mutate()` function, which encodes the hour of the year. The data can be joined by this new key since it will be consistent across both data sets. This only works if each data set starts at the same time and increments hourly, which in this case they both do.

```
# Add a column to track the hour of the year
EIAf = EIAf |> mutate(hour = c(1:8760))
ERCOf = ERCOf |> mutate(hour = c(1:8760))
# Check anti_join agaim
EIAf |> anti_join(ERCOf, by = 'hour')
```

```
## # A tibble: 0 × 12
## # … with 12 variables: utc_time_at_end_of_hour <chr>,
## #   demand_forecast_(mw) <dbl>, demand_(mw) <dbl>, net_generation_(mw) <dbl>,
## #   net_generation_(mw)_from_coal <dbl>,
## #   net_generation_(mw)_from_natural_gas <dbl>,
## #   net_generation_(mw)_from_nuclear <dbl>,
## #   net_generation_(mw)_from_hydropower_and_pumped_storage <dbl>,
## #   net_generation_(mw)_from_solar <dbl>, …
```

No values are missing, therefore all values will be matched in the joined data set and no values will be missed. The data sets can now be joined by the hour of the year.

```
# Join the data sets by the hour of the year
ed = left_join(ERCOf, EIAf, by = 'hour')
ed |> head()
```

```
## # A tibble: 6 × 33
##   utc_time       subre…¹ subre…² subre…³ subre…⁴ subre…⁵ subre…⁶ subre…⁷ subre…⁸
##   <chr>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 01Jan2022 7:0…   11593    1348    3875     860    9544    3336    6268    1321
## 2 01Jan2022 8:0…   11349    1292    3805     851    9177    3289    6055    1340
## 3 01Jan2022 9:0…   11007    1293    3841     854    8756    3167    5772    1276
## 4 01Jan2022 10:…   10851    1241    3896     835    8538    3057    5530    1201
## 5 01Jan2022 11:…   10646    1216    3954     827    8481    2983    5419    1085
## 6 01Jan2022 12:…   10628    1204    3902     835    8465    2941    5368    1106
## # … with 24 more variables: co2_factor_col <dbl>, co2_factor_ng <dbl>,
## #   co2_factor_oil <dbl>, co2_emissions_col <dbl>, co2_emissions_ng <dbl>,
## #   co2_emissions_other <dbl>, co2_emissions_generated <dbl>,
## #   co2_emissions_imported <dbl>, co2_emissions_exported <dbl>,
## #   co2_emissions_consumed <dbl>,
## #   co2_emissions_intensity_for_generated_electricity <dbl>,
## #   co2_emissions_intensity_for_consumed_electricity <dbl>, hour <int>, …
```

```
# Remove bad date format, and reorder columns, to make neat and easy to tidy later
ed = ed |> select(-c(1)) |> rename('time' = 'utc_time_at_end_of_hour')
ed = ed[,c(21,22,1:20,23:32)]
ed = ed |> rename_all(~str_replace_all(.,"\\:","")) |>  select(-c(3:10))
ed |> head()
```

```
## # A tibble: 6 × 24
##    hour time     co2_f…¹ co2_f…² co2_f…³ co2_e…⁴ co2_e…⁵ co2_e…⁶ co2_e…⁷ co2_e…⁸
##   <int> <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1 01/01/2…    2.28    0.92    2      11327    4082     115   15523      76
## 2     2 01/01/2…    2.28    0.92    2      10473    3587     120   14180     132
## 3     3 01/01/2…    2.28    0.92    2       9252    3081     130   12463      62
## 4     4 01/01/2…    2.28    0.92    2.01    7416    2622     144   10183      63
## 5     5 01/01/2…    2.28    0.92    2.01    5809    2286     157    8252      42
## 6     6 01/01/2…    2.28    0.92    2       5843    2243     155    8242      59
## # … with 14 more variables: co2_emissions_exported <dbl>,
## #   co2_emissions_consumed <dbl>,
## #   co2_emissions_intensity_for_generated_electricity <dbl>,
## #   co2_emissions_intensity_for_consumed_electricity <dbl>,
## #   `demand_forecast_(mw)` <dbl>, `demand_(mw)` <dbl>,
## #   `net_generation_(mw)` <dbl>, `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>, …
```

The final data set, `ed` ,has 8760 rows, the number of hours in a year, and 24 columns. The original data sets `EIA1` , `EIA2` , and `CARBON` were reduced by 264861, 266570, and 58609 observations respectively. The columns of the joined data set are rearranged and some are removed to make tidying easier. Moving date and hour to the first two columns also makes it easier to check, view, and manipulate later. Finally the data is now ready to be tidied.

Tidying

```
# Tidy the data by pivoting longer, making each row its own observation
el = ed |> pivot_longer(c(3:24), names_to = 'id')
el |> head()
```

```
## # A tibble: 6 × 4
##    hour time                   id                     value
##   <int> <chr>                  <chr>                  <dbl>
## 1     1 01/01/2022 7:00:00 AM co2_factor_col          2.28
## 2     1 01/01/2022 7:00:00 AM co2_factor_ng           0.92
## 3     1 01/01/2022 7:00:00 AM co2_factor_oil          2
## 4     1 01/01/2022 7:00:00 AM co2_emissions_col   11327
## 5     1 01/01/2022 7:00:00 AM co2_emissions_ng     4082
## 6     1 01/01/2022 7:00:00 AM co2_emissions_other   115
```

The data set `ed` is pivoted longer such that each observation has its own row and each variable has its own column. This is done by using the `pivot_longer()` function to set all column names from 3 to 24 to be an id variable and the value of each to be the value of the observation. The tidied data set is useful for grouping generation and emissions by type together to compute summary statistics and create visualizations. Every column in the original data set is transformed to be an identifier for each observation. The date and hour of the year remain columns, and a new column, value is created to give the value of the distinct observation.

Wrangling

Summary statistics for CO2 emissions in metric tons, and generation by type in Megawatts:

```
# Summary stats for all emissions by generation type
el |> filter(id %in% c('co2_emissions_col','co2_emissions_ng','co2_emissions_other')) |> group_by(i
d) |> summarise(n = n(), mean = mean(value), sd = sd(value))
```

```
## # A tibble: 3 × 4
##   id                      n  mean     sd
##   <chr>               <int> <dbl>  <dbl>
## 1 co2_emissions_col    8760 8529. 1815.
## 2 co2_emissions_ng     8760 8685. 4117.
## 3 co2_emissions_other  8760  133.   41.6
```

```
# Summary stats for all generation by type
gen_type = c('net_generation_(mw)_from_coal','net_generation_(mw)_from_natural_gas','net_generation_
(mw)_from_nuclear', 'net_generation_(mw)_from_hydropower_and_pumped_storage', 'net_generation_(mw)_f
rom_solar', 'net_generation_(mw)_from_wind')
el |> filter(id %in% gen_type) |> group_by(id) |> summarise(n = n(), mean = mean(value), sd = sd(val
ue))
```

```
## # A tibble: 6 × 4
##   id                                                         n    mean     sd
##   <chr>                                                  <int>   <dbl>  <dbl>
## 1 net_generation_(mw)_from_coal                           8760  8210.  1753.
## 2 net_generation_(mw)_from_hydropower_and_pumped_storage  8760    36.2   42.5
## 3 net_generation_(mw)_from_natural_gas                    8760 20860.  9941.
## 4 net_generation_(mw)_from_nuclear                        8760  4778.   529.
## 5 net_generation_(mw)_from_solar                          8760  2702.  3363.
## 6 net_generation_(mw)_from_wind                           8760 12258.  6512.
```

To calculate summary statistics, the tidy data set `el` is filtered to only the desired ids, in this case for co2 emissions type and generation type. A vector `gen_type` is also created for later use to easily filter to only generation types. The filtered data is given to the summarize function that returns the number of observations, mean, and standard deviation. The average coal, natural gas, and other CO2 emissions were found to be 8528.7523MW, 8685.4475MW, and 132.5945MW respectively. Texas emits almost all of its CO2 from coal and natural gas, with minimal coming from other generation sources. The standard deviation of coal emissions and natural gas emissions are 1814.68171 and 4116.91141 respectively. The average CO2 emitted from natural gas and coal are quite close to each other, however natural gas varies much more than coal since its standard deviation is much higher.

As can be seen in the summary statistics for generation type, Texas gets most of its power from natural gas with wind being second, and coal being third. The mean for these three respectively were 20860.4430MW, 12258.3498, and 8209.5861. The standard deviation of natural gas emissions was 9940.90744 and likely comes from the fact that its variability in energy generation is much larger than other sources. This is due to natural gas being used as a backup to the rest of the grid due to its fast response time. Wind power also accounts for a large share of the average energy produced, producing an average of 12258.3498MW. This was more than coal or nuclear which is surprising, since I never really realized that Texas had a lot of wind generation since it seems to rely heavily on oil and gas. Wind also had a standard deviation of 6511.94927 which is quite large. This makes sense since the wind is quite random and predicable. Nuclear, however, had a standard deviation of 528.94566 which is far below the other major sources of energy generation. This makes sense since nuclear plants have to operate at a consistent power and suggests that Nuclear could be a valid means to build a strong foundation for the grid.

To evaluate the cleanliness and reliability of the grid, two categorical variables are created. The first indicates whether the generation met the demand for that hour. This represents reliability of the grid. The second categorical variable indicates if the electricity produced for that hour is “clean” or “dirty” based on carbon emissions. For this analysis, a carbon intensity larger than 0.875 lbs/kWh is dirty, intensity less than 0.625 lbs/kWh is clean, and anything within that range is acceptable.

```
# Create a new column that says if the demand is met by the generation
ed = ed |> mutate(difference = `net_generation_(mw)`-`demand_(mw)`) |> mutate(reliable = case_when(d
ifference < 0 ~ 'not', difference >= 0 ~ 'met'))
# classify electricity as clean or dirty
ed = ed |> mutate(cleanliness =
            case_when(co2_emissions_intensity_for_generated_electricity < 0.625 ~ 'clean',
                      co2_emissions_intensity_for_generated_electricity >= 0.625 & co2_emissions_
intensity_for_generated_electricity <= 0.875 ~ 'acceptable',
                      co2_emissions_intensity_for_generated_electricity > 0.875 ~ 'dirty'))
# Take a look at the mutated data set
ed |> head()
```

```
## # A tibble: 6 × 27
##    hour time     co2_f…¹ co2_f…² co2_f…³ co2_e…⁴ co2_e…⁵ co2_e…⁶ co2_e…⁷ co2_e…⁸
##   <int> <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1 01/01/2…    2.28    0.92    2      11327    4082     115   15523      76
## 2     2 01/01/2…    2.28    0.92    2      10473    3587     120   14180     132
## 3     3 01/01/2…    2.28    0.92    2       9252    3081     130   12463      62
## 4     4 01/01/2…    2.28    0.92    2.01    7416    2622     144   10183      63
## 5     5 01/01/2…    2.28    0.92    2.01    5809    2286     157    8252      42
## 6     6 01/01/2…    2.28    0.92    2       5843    2243     155    8242      59
## # … with 17 more variables: co2_emissions_exported <dbl>,
## #   co2_emissions_consumed <dbl>,
## #   co2_emissions_intensity_for_generated_electricity <dbl>,
## #   co2_emissions_intensity_for_consumed_electricity <dbl>,
## #   `demand_forecast_(mw)` <dbl>, `demand_(mw)` <dbl>,
## #   `net_generation_(mw)` <dbl>, `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>, …
```

Reliability is calculated by the proportion of the time the grid meets demand and the proportion of the time it does not. Cleanliness is quantified by calculating the proportion of the time the grid is considered clean, acceptable, and dirty. This is done by selecting the desired variable, converting it to a table, and dividing by the number of observations, in this case the number of hours in the year.

```
# calculate percentage of time met and not met
ed |> select(reliable) |> table()/8760*100
```

```
## reliable
##      met      not
## 33.13927 66.86073
```

```
# Calculate percentage of cleanliness
ed |> select(cleanliness) |> table()/8760*100
```

```
## cleanliness
## acceptable      clean      dirty
##   51.55251   19.54338   28.90411
```

The grid meets demand 33.13927% of the time and fails to meet demand 66.86073% of the time. This makes the grid very unreliable since it cannot meet demand a majority of the time. The grid's carbon emissions are acceptable 51.55251% of the time, clean 19.54338% of the time, and dirty 28.90411% of the time. The grid may not be very reliable but it is quite clean, being acceptable or clean 71.09589% of the time. This is quite surprising since a majority of the power comes from natural gas and coal. This is also quite alarming because Texas is isolated from the rest of the United States power grid and it must be able to at least support itself.

To truly evaluate the reliability, peak demand hours should be considered. It's important that the grid can handle times when the demand is at its highest. The data is arranged in descending order by demand and indexed to the first 100 rows. The percentages are computed the same as before.

```
# Find 100 times with most demand
high_demand = ed |> arrange(desc(`demand_(mw)`))
high_demand = high_demand[c(1:100),]
# Take a look at the first few rows
high_demand |> head()
```

```
## # A tibble: 6 × 27
##    hour time     co2_f…¹ co2_f…² co2_f…³ co2_e…⁴ co2_e…⁵ co2_e…⁶ co2_e…⁷ co2_e…⁸
##   <int> <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1  4816 07/20/2…    2.29    0.91    2.01   12050   19237     139   31426     505
## 2  4817 07/20/2…    2.29    0.91    2.01   12161   18999     141   31302     504
## 3  4792 07/19/2…    2.29    0.91    2.01   12065   17454     171   29689     130
## 4  4793 07/19/2…    2.29    0.91    2.01   11967   17710     166   29843     121
## 5  4815 07/20/2…    2.29    0.91    2.02   11773   19301     137   31211     509
## 6  4791 07/19/2…    2.29    0.91    2.01   11928   16986     176   29090     210
## # … with 17 more variables: co2_emissions_exported <dbl>,
## #   co2_emissions_consumed <dbl>,
## #   co2_emissions_intensity_for_generated_electricity <dbl>,
## #   co2_emissions_intensity_for_consumed_electricity <dbl>,
## #   `demand_forecast_(mw)` <dbl>, `demand_(mw)` <dbl>,
## #   `net_generation_(mw)` <dbl>, `net_generation_(mw)_from_coal` <dbl>,
## #   `net_generation_(mw)_from_natural_gas` <dbl>, …
```

```
# Calculate reliability and cleanliness data for the 100 most demand hours
high_demand |> select(reliable) |> table()/100
```

```
## reliable
##  met  not
## 0.03 0.97
```

```
high_demand |> select(cleanliness) |> table()/100
```

```
## cleanliness
## acceptable      dirty
##        0.6        0.4
```

The times with the highest demand are all during the summer. This is expected since it requires more energy to cool than it does to heat. During these 100 times, the grid is extremely unreliable, not meeting demand 97% of the time. This is quite terrible since it means that during the summer Texas cannot support itself. As the climate continues to warm and the population of Texas continues to increase, this disparity will only continue to grow unless something is done. The cleanliness of the grid is still relatively good, being acceptable 60% of the time, and dirty 40% of the time. The grid was never clean during this time. This is surprising since we would have expected that it would spend a majority of the time being dirty to produce as much energy as possible to meet demand. Again as we move toward more sustainable solutions, steps will need to be taken to upgrade the grid with better, more efficient, forms of energy production and storage.

Visualizing

In all plots regarding the time the year is split into 4 quarters (Q) to make analysis easier.

```
#Plot of carbon intensity over the year
ed |> ggplot(aes(x = hour, y = co2_emissions_intensity_for_generated_electricity)) + geom_line() + g
eom_smooth(col="red") +
      labs(title = "Carbon Intensity of ERCOT Power Grid (2022)", x = "Hour of the Year", y = "Carbo
n Intensity (lbs/kWh)") +
      scale_x_continuous(breaks = seq(0,8760,2190))
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

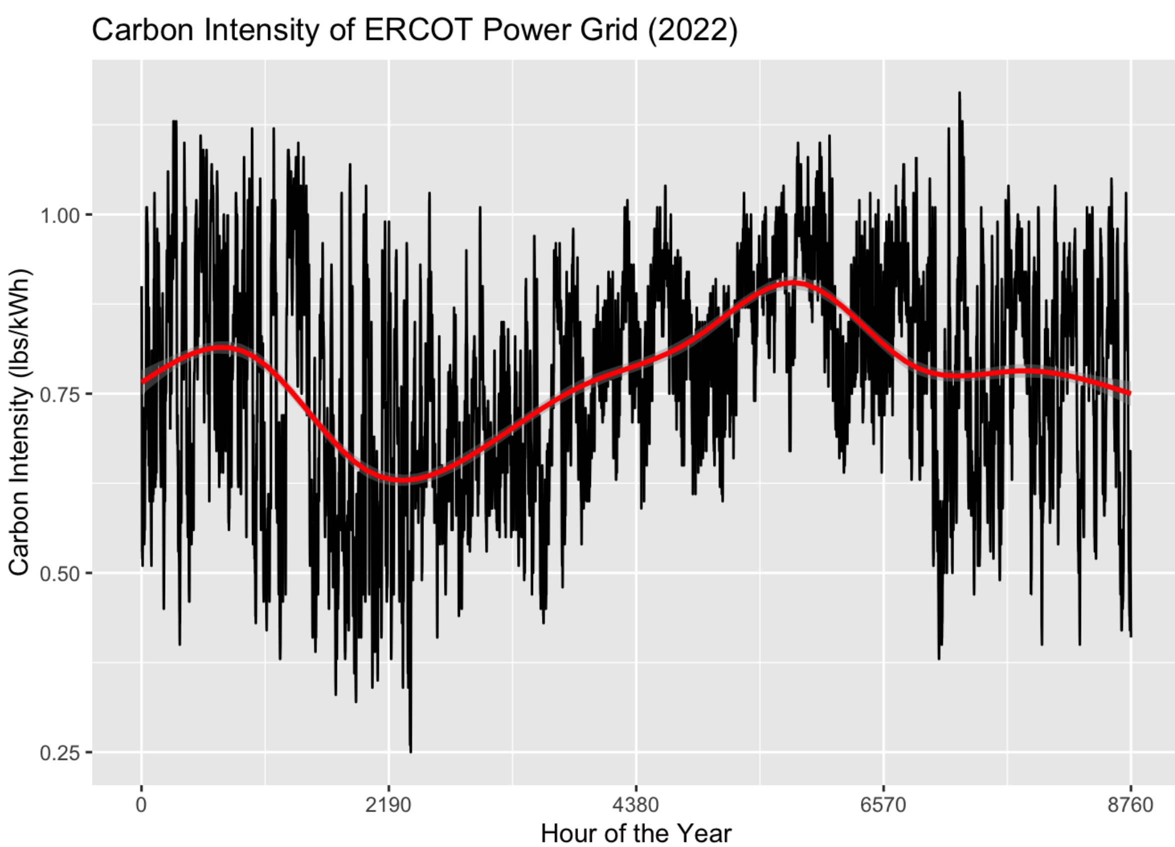

Figure (1): Graph of carbon intensity (lbs/kWh) over time

As seen in figure (1), carbon intensity seems to be at a low at the end of Q1, and at a high during Q3. This is expected since at the end of Q1 and the start of Q2 it is spring and there is minimal need for heating or cooling. The maximum in Q3 is also expected since there is high demand for electricity for cooling, and natural gas is the primary source of electricity in this grid (see

Figure (3)). This then remains constant through Q4 as demand for cooling rapidly turns into demand for heating. The intensity never reaches this maximum again since heating is much more energy efficient compared to cooling.

```
# Plot of diff vs demand where color is mapped to cleanliness and shape is mapped to reliablity
ed |> ggplot(aes(x = `demand_(mw)`, y = difference)) +
  geom_point() +
  geom_smooth(method = lm, col = 'aquamarine')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

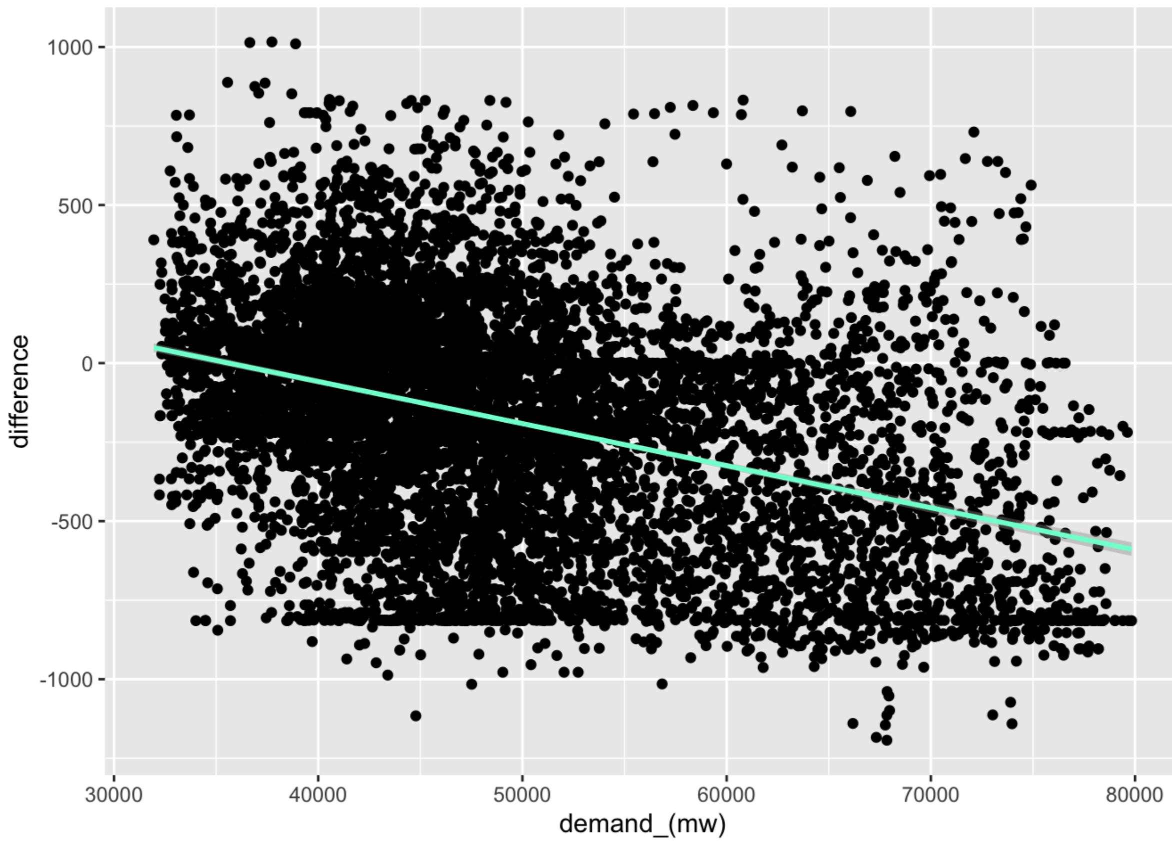

Figure (2): Graph of difference between demand and production as a function of demand.

Figure (2) shows a clear negative relationship between the demand and difference in production and demand. This shows that at high demands, such as in the summer, the Texas power grid is unable to keep up with demand. Also seen in figure (2) is that at lower demand the grid can meet it relatively cleanly and consistently, however at high demand and large deficits, it becomes very dirty. This is likely due to dependence on natural gas and coal for fast response and high demand situations. If Texas were to invest in energy storage technologies it would help the grid be cleaner and prepared for situations when it needs a large amount of power.

```
#Plot of generation by type
el |> filter(id %in% gen_type) |> ggplot(aes(x = hour, y = value)) +
  geom_smooth(aes(color = id), show.legend = TRUE) +
  labs(title = "ERCOT Generation of Electricity by Type (2022)", y = "Generation (MWh)", x = "Hour o
f the Year") +
  scale_color_hue(name = "", labels = c('Coal', 'Hydropwer', 'Natural gas', 'Nuclear', 'Solar', 'Win
d')) +
  theme(legend.position = c(1, 1.05),
    legend.justification = c("right", "top"),
    legend.box.just = "right",
    legend.margin = margin(6, 6, 6, 6),
    legend.background = element_blank()) +
  scale_x_continuous(breaks = seq(0,8760,2190))
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

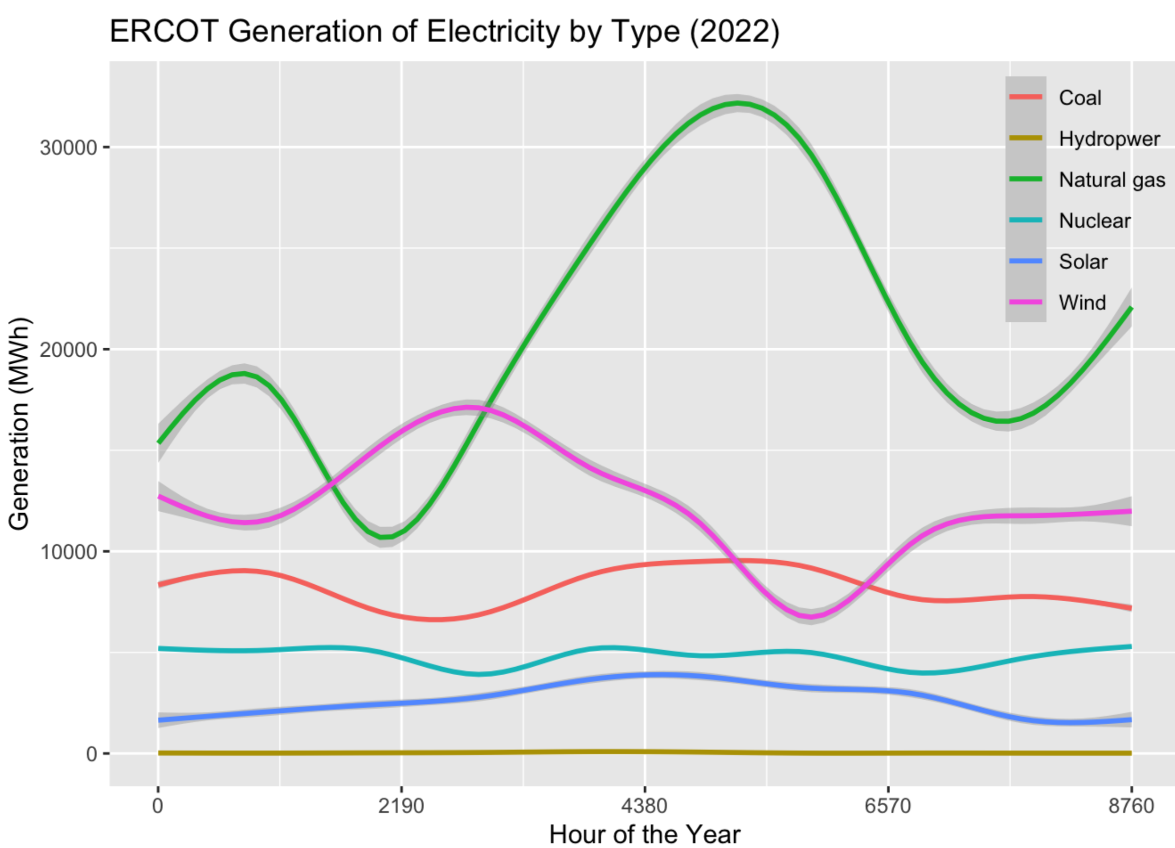

Figure (3): Generation over time for all sources

Figure (3) is quite interesting since it shows the amount of power generated per source over the year. One surprise is that at the end of Q1, the start of spring, wind power actually becomes the dominant energy source. In Q3, however, wind dips very low and natural gas spikes, again due to summer cooling demand. One prospective way to relieve this spike and reliance on natural gas is to invest in more solar power. In Q3 solar power reaches its maximum, and it should be used to help even out the load on the grid. Coal is also used heavily in Q3 to meet demand, again contributing to the ‘dirty’ power produced in the summer. As was expected nuclear tends to stay at a relatively stable generation throughout the year. Hydroelectic produces almost nothing throughout the year. If Texas wants to improve the cleanliness and reliability of the grid, it needs to focus on implementing more solar and nuclear generation to meet baseline power demands, and use wind with energy storage for response to demand fluctuations. It should still keep natural gas for times like Q3 when demand goes beyond the variation of the rest of the year.

```
ed |> ggplot(aes(x = reliable)) + geom_bar(aes(fill = reliable)) +
  labs(title = "Reliability of the ERCOT Power Grid (2022)", x="Generation Meeting Demand", y="Numbe
r of Hours") +
  scale_y_continuous(breaks = seq(0,7000,500)) +
  scale_fill_brewer(type = 'qual', palette = 'Set4')
```

```
## Warning in pal_name(palette, type): Unknown palette Set4
```

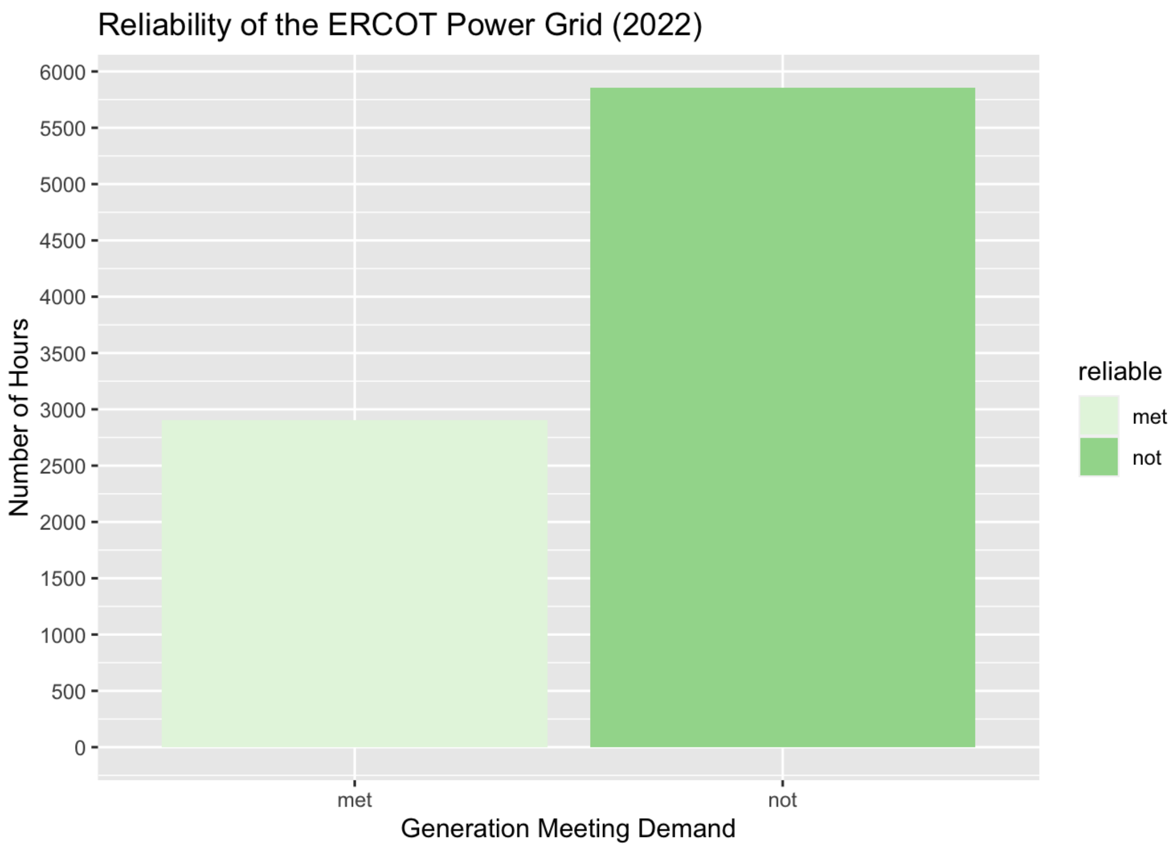

Figure (4): Bar plot of the number of times the grid met demand and did not meet demand

Figure (4) shows that for a majority of the time, Texas' power grid is unable to meet demand. This indicates a larger underlying issue with the power grid. This plot, however shows the true scale of how little Texas can meet its own demand. This is larger than just one off quarters like Q3, summer, it is an underlying issue that effects the entire grid. This also shows that this is an immediate issue that needs to be fixed as soon as possible to reduce the amount of time that the grid does not meet demand.

```
ed |> ggplot(aes(x = cleanliness)) + geom_bar(aes(fill = cleanliness)) +
  labs(title = "Cleanliness of the ERCOT Electricity (2022)", x="Cleanliness Rating", y="Number of H
ours") +
  scale_y_continuous(breaks = seq(0,5000,500)) +
  scale_fill_brewer(type = 'qual', palette = 'Set3')
```

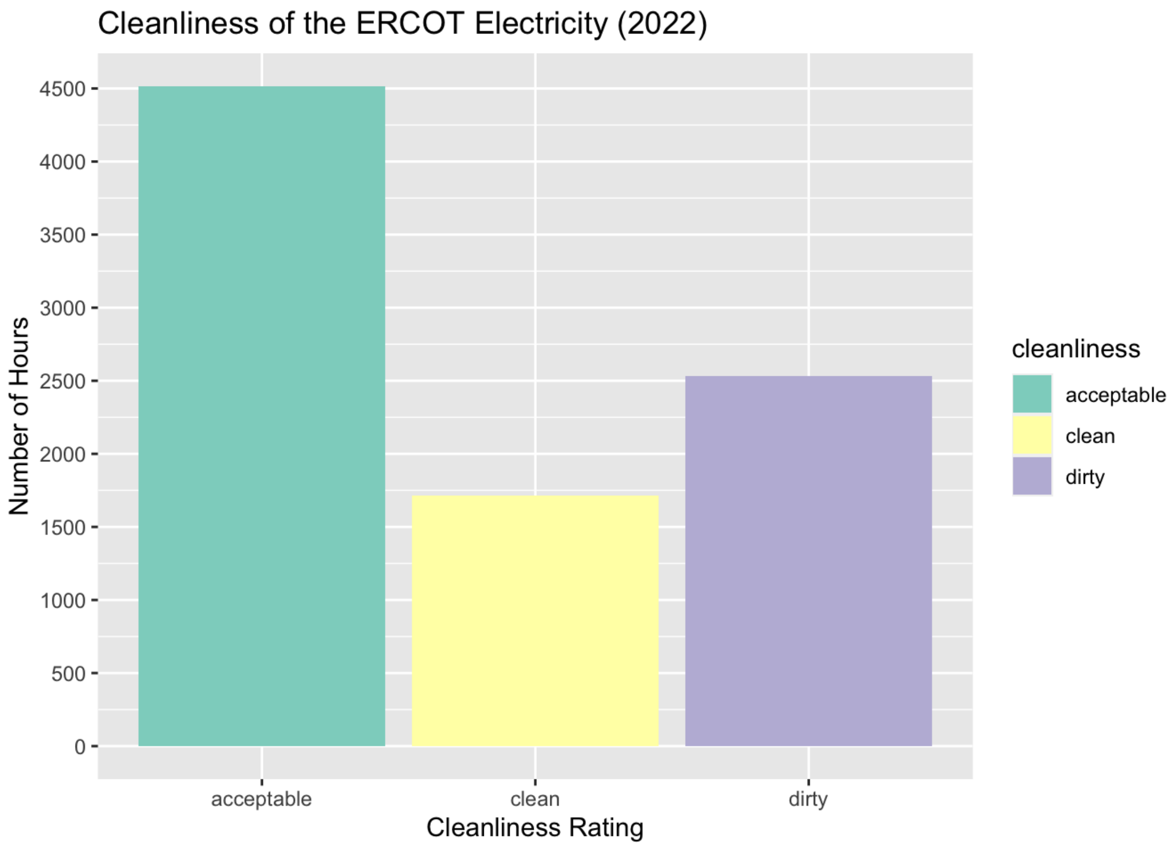

Figure (5): Bar plot of how clean the energy produced by the grid is.

This plot shows that for a majority of the time, Texas' power grid is quite environmentally friendly. It is acceptable or clean a large majority of the time. This plot shows that Texas needs to continue implementing environmentally friendly upgrades to the grid when they do improve it. Considering the poor state of the grid it is quite commendable that it is relatively green.

```
# Plot of generation over time where color is mapped to cleanliness and shape is mapped to reliabili
ty
ed |> ggplot(aes(x = hour, y = `net_generation_(mw)`)) +
  geom_point(aes(color = cleanliness), size = 0.7) +
  scale_color_manual(values=c("cadetblue3", "chartreuse3", "firebrick2")) +
  scale_x_continuous(breaks = seq(0,8760,2190)) +
  ylab("Net Generation (MW)") +
  labs(title = "Generation over Time (2022)")
```

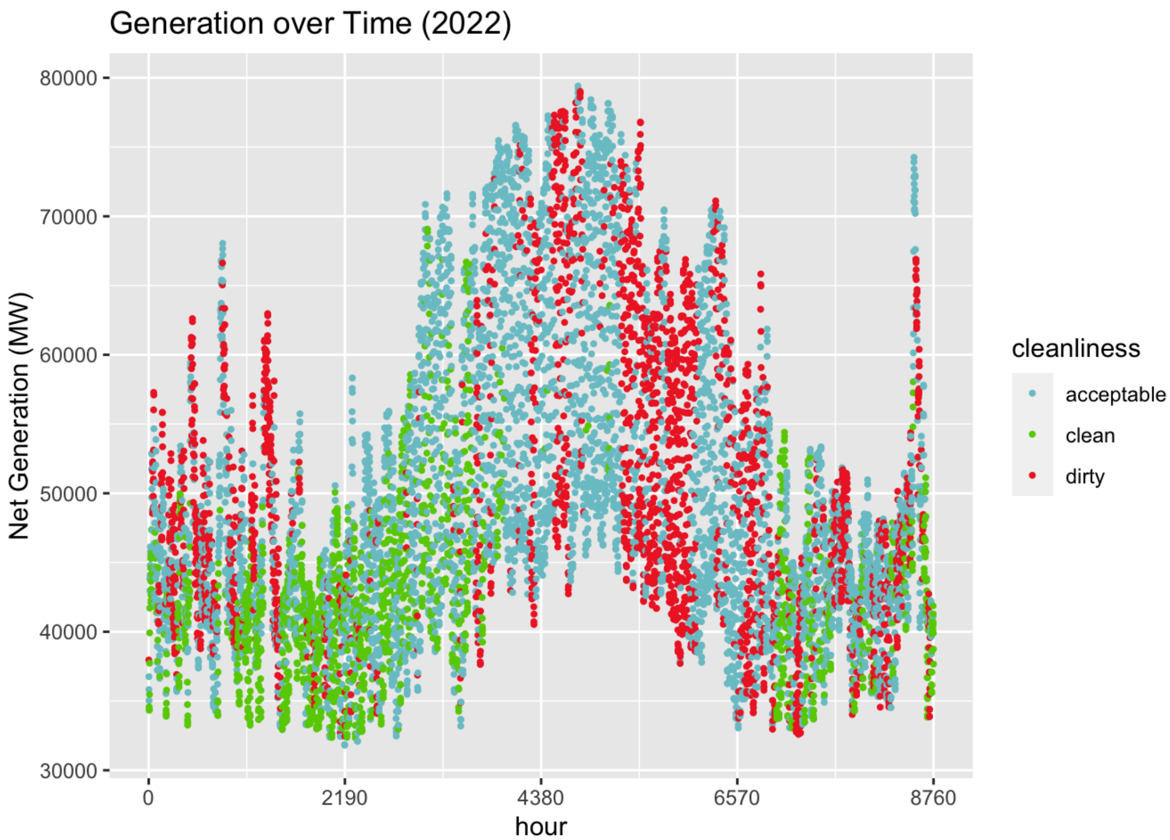

Figure (6): Scatter plot of energy generation over time where color is mapped to cleanliness.

This scatter plot shows the trend also seen in figure (1). Demand spikes in Q3 in response to the summer heat. This plot also shows that when the demand spikes, the cleanliness seems to also decrease. For the first half of Q3 the emissions are acceptable, however, at the end of Q3 emissions get quite bad. This suggests that the Texas energy grid is not equipped to handle high demand while being environmentally friendly. At lower demand in Q1, Q2, and Q3 the grid is capable of providing clean energy. This is consistent with figure (1) that in the summer emissions are much higher than in winter.

```
el |> filter(id %in% gen_type) |> ggplot(aes(x = id, y = value, fill = id)) +
  geom_bar(stat = 'summary', fun = 'mean') +
  scale_fill_discrete(labels = c('Coal', 'Hydropwer', 'Natural gas', 'Nuclear', 'Solar', 'Wind')) +
  geom_errorbar(stat = 'summary', fun.data = 'mean_se', width = 0.5) +
  theme(axis.text.x = element_blank()) +
  ylab("Average Energy Generated (MW)") +
  labs(title = "Average Energy Generated by type (2022)")
```

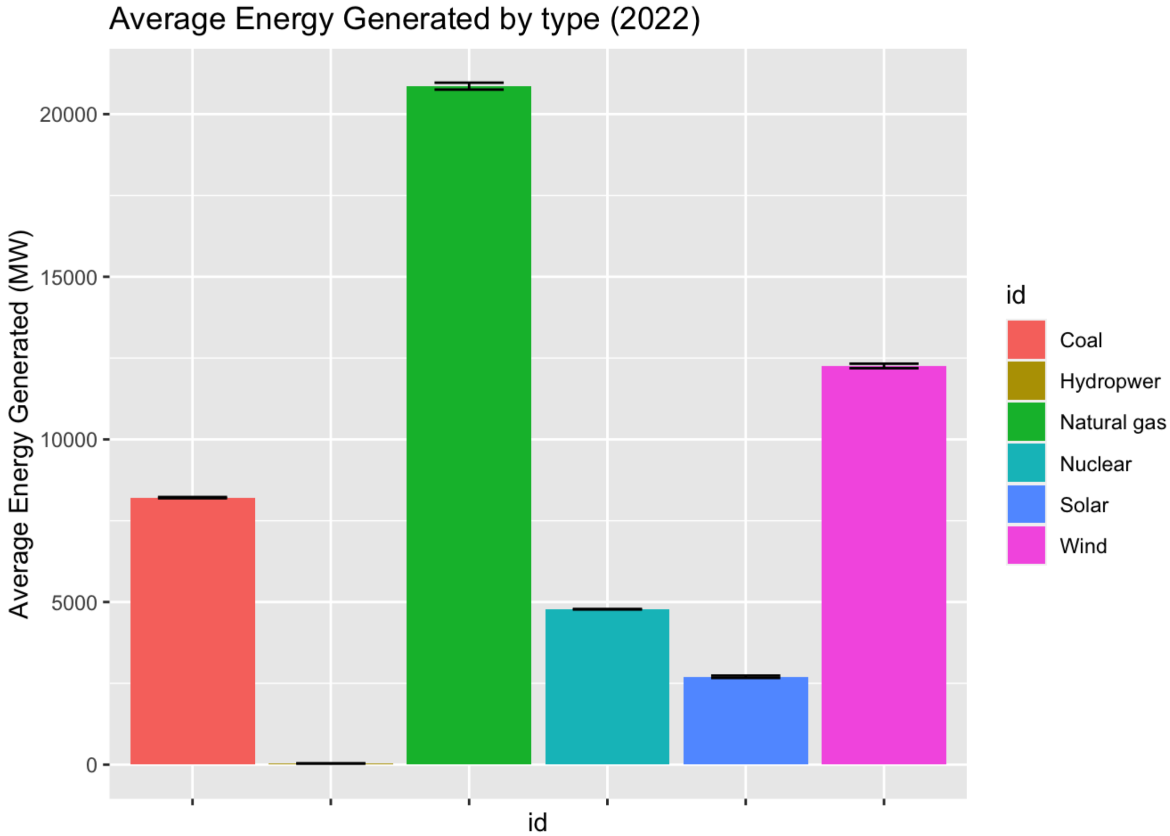

Figure (7): Bar plot of average energy generated by type.

For the entire year, natural gas produced the most energy and had the largest variability. Wind was second, with coal in third. Wind was more variable than coal but less variable than natural gas. This plot again shows Texas' reliance on natural gas. Texas relies on natural gas due to its ability to quickly respond to demand, but again investing in more energy production and storage technology would be more beneficial for long term sustainability.

Discussion How often does generation fail to meet demand in the Texas power grid?
The Texas power grid fails to meet demand approximately 66.86% of the time during the year. As we hypothesized, generation is less likely to meet demand as demand increases. This can be seen in Figure (2) and in the summary statistics calculated for the top 100 hours of peak demand, which stated that generation didn't meet demand for 97 of the 100 peak hours.
One limitation of this analysis is that there is no information on energy storage. For hours where the generation is larger than the demand (i.e. difference > 0), it is not known where that excess energy is stored and how it's distributed. It's possible that for a majority of the hours where the demand is greater than the generation, the difference is supplemented with accumulated energy in the grid, which means the demand was satisfied. However, this data wasn't available, and the storage capabilities of the ERCOT grid are unknown.

How does the carbon emissions from electricity generation vary over the year?
As seen in Figure (1), the carbon intensity is at its minimum in Q1 with spring and peaks in Q3 with summer. The minimum is due to a large output of wind power in the spring, and the maximum is due to increased natural gas usage because of the large cooling demand in peak summer time. A key solution to reducing carbon intensity and also increasing grid stability is to expand the capacity of constant energy sources like nuclear and hydropower. These energy sources are renewable, but not variable like wind or solar, which means they can consistently output energy throughout the year. Increasing the capacity of nuclear and hydropower will create a larger baseline output and reduce the need for natural gas and coal from demand fluctuations. Once this has been achieved, solar and wind energy should be coupled with efficient energy storage technologies to satisfy demand during these peak hours to reduce coal and natural gas dependencies even more.

The most challenging part of this analysis was cleaning and joining the two data sets together. The original data sets were so large that finding the needed observations and columns for this analysis was tricky. We learned that the majority of the work in data science comes from gathering and compiling the data into a usable format, the actual analysis, visualization and discussion is only about 20% of the process.

Solomon: joining, cleaning, tidying, visualizing
Vardhan : introduction, discussion, comments, visualizing

Sources Introduction: https://comptroller.texas.gov/economy/fiscal-notes/2022/sep/energy.php#(https://comptroller.texas.gov/economy/fiscal-notes/2022/sep/energy.php#):~:text=Texas%20leads%20the%20nation%20in,nation’s%20total%20net%20energy%20generation.

This data was obtained from the United States Electricity Information Administration:
https://www.eia.gov/electricity/gridmonitor/dashboard/electric_overview/US48/US48
(https://www.eia.gov/electricity/gridmonitor/dashboard/electric_overview/US48/US48)
Download Data -> Six-Month Files -> 2022 -> Both balance files are `EIA1` and `EIA2`
Download Data -> Balancing Authority / Region Files -> the ERCO file is `CARBON`