BEE 4750 Homework 4: Generating Capacity Expansion

Name: Anthony Nicolaides

ID: ajn68

> **Due Date**
>
> Friday, 10/27/23, 9:00pm

Overview

Instructions

- In Problem 1, you will formulate, solve, and analyze a standard generating capacity expansion problem.
- In Problem 2, you will add a CO_2 constraint to the capacity expansion problem and identify changes in the resulting solution.

Load Environment

The following code loads the environment and makes sure all needed packages are installed. This should be at the start of most Julia scripts.

In [ ]:
```julia
import Pkg
Pkg.activate(@__DIR__)
Pkg.instantiate()
```

```
  Activating project at `~/Documents/BEE4750/hw/hw04-anthonynic28`
```

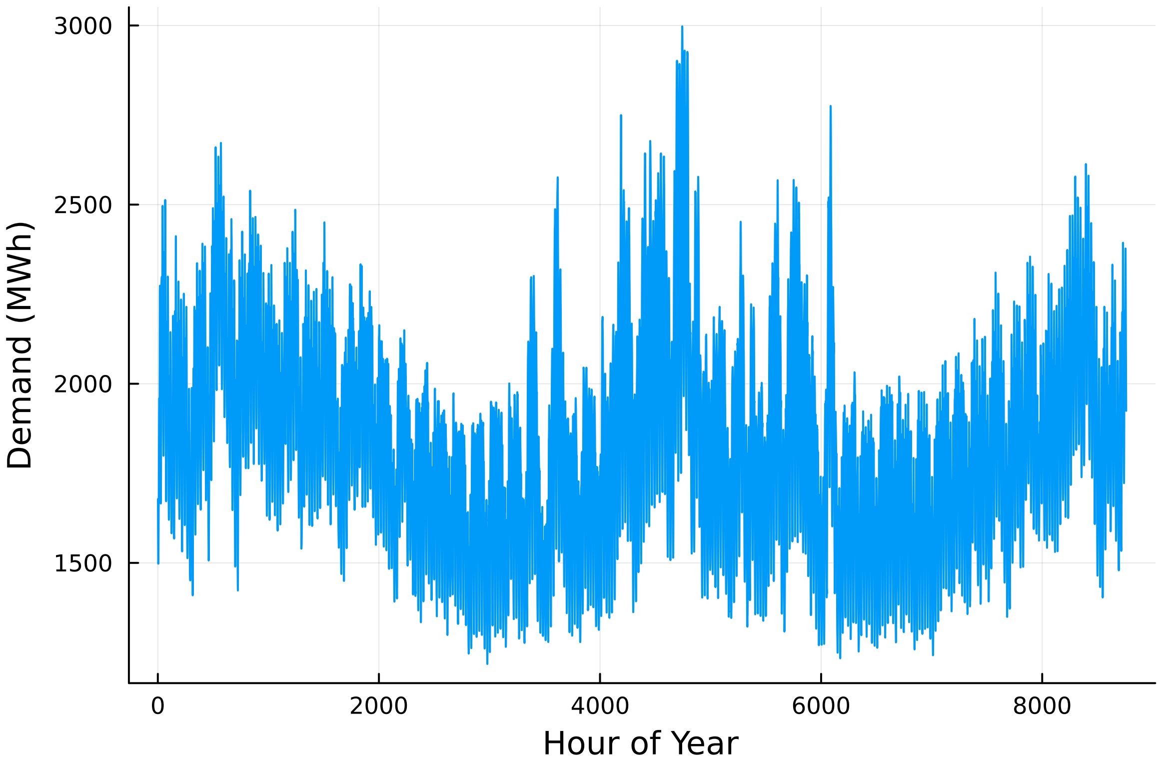
In [ ]:
```julia
using JuMP
using HiGHS
using DataFrames
using Plots
using Measures
using CSV
using MarkdownTables
```

Problems (Total: 100 Points)

For this problem, we will use hourly load (demand) data from 2013 in New York's Zone C (which includes Ithaca). The load data is loaded and plotted below in Figure 1.

In [ ]:
```
# load the data, pull Zone C, and reformat the DataFrame
NY_demand = DataFrame(CSV.File("data/2013_hourly_load_NY.csv"))
rename!(NY_demand, :"Time Stamp" => :Date)
demand = NY_demand[:, [:Date, :C]]
rename!(demand, :C => :Demand)
demand[:, :Hour] = 1:nrow(demand)

# plot demand
plot(demand.Hour, demand.Demand, xlabel="Hour of Year", ylabel="Demand (MWh)
```



Next, we load the generator data. This data includes fixed costs (/MWinstalled), variablecosts(/MWh generated), and CO_2 emissions intensity (tCO_2/MWh generated).

In [ ]:
```
gens = DataFrame(CSV.File("data/generators.csv"))
```

6×4 DataFrame

Row	Plant	FixedCost	VarCost	Emissions
	String15	Int64	Int64	Float64
1	Geothermal	450000	0	0.0
2	Coal	220000	24	1.0
3	NG CCGT	82000	30	0.43
4	NG CT	65000	40	0.55
5	Wind	91000	0	0.0
6	Solar	70000	0	0.0

Finally, we load the hourly solar and wind capacity factors, which are plotted in Figure 2. These tell us the fraction of installed capacity which is expected to be available in a given hour for generation (typically based on the average meteorology).

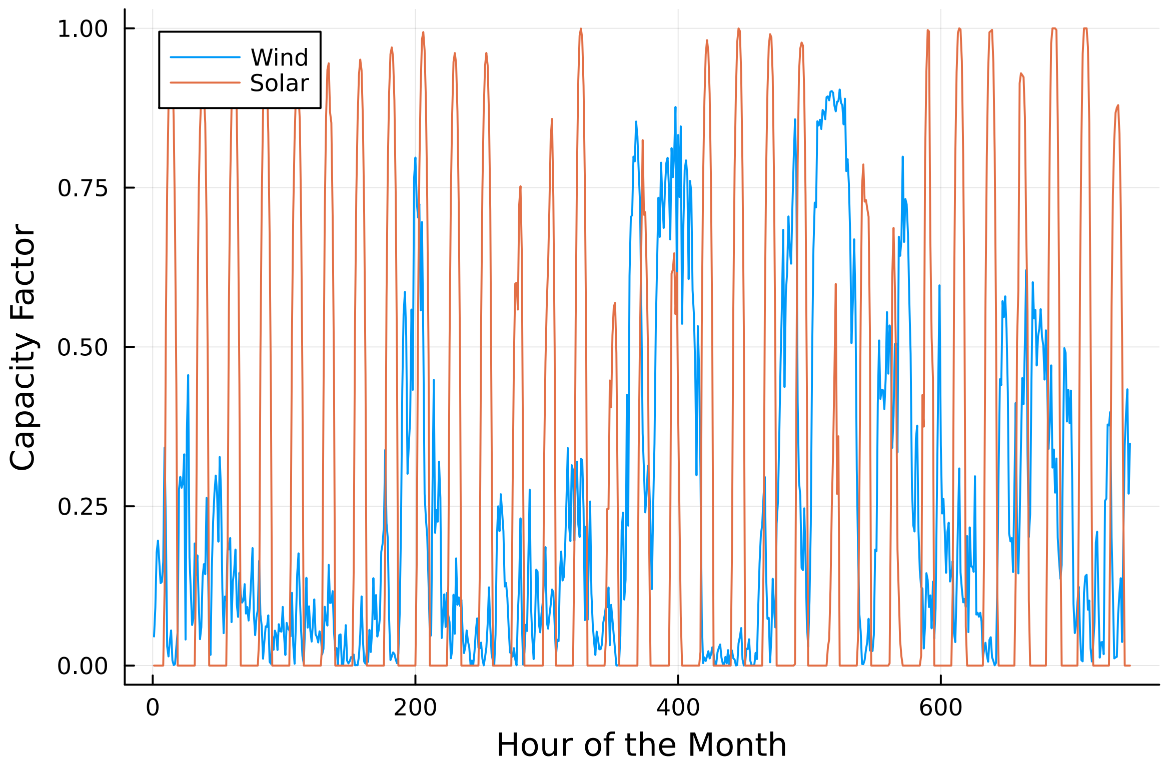
In [ ]:

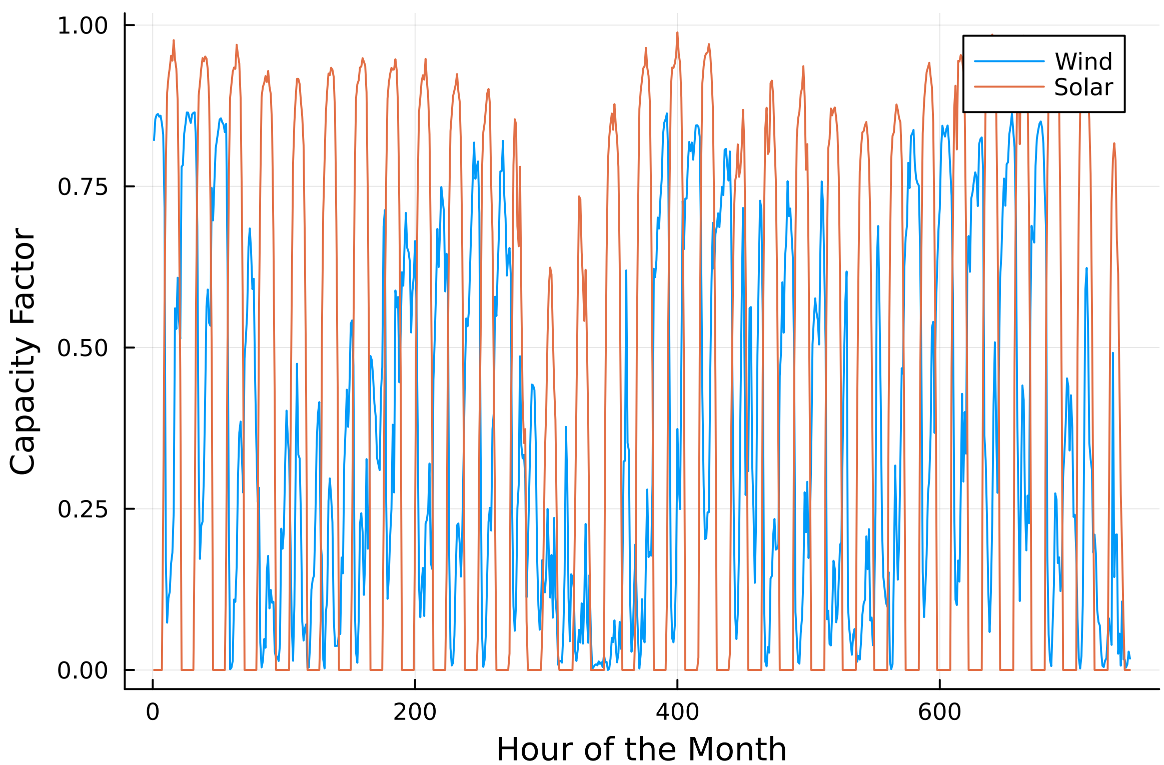
```
# load capacify factors into a DataFrame
cap_factor = DataFrame(CSV.File("data/wind_solar_capacity_factors.csv"))

# plot January capacity factors
p1 = plot(cap_factor.Wind[1:(24*31)], label="Wind")
plot!(cap_factor.Solar[1:(24*31)], label="Solar")
xaxis!("Hour of the Month")
yaxis!("Capacity Factor")

p2 = plot(cap_factor.Wind[4344:4344+(24*31)], label="Wind")
plot!(cap_factor.Solar[4344:4344+(24*31)], label="Solar")
xaxis!("Hour of the Month")
yaxis!("Capacity Factor")

display(p1)
display(p2)
```





You have been asked to develop a generating capacity expansion plan for the utility in Riley, NY, which currently has no existing electrical generation infrastructure. The utility can build any of the following plant types: geothermal, coal, natural gas combined cycle gas turbine (CCGT), natural gas combustion turbine (CT), solar, and wind.

While coal, CCGT, and CT plants can generate at their full installed capacity, geothermal plants operate at maximum 85% capacity, and solar and wind available capacities vary by the hour depend on the expected meteorology. The utility will also penalize any non-served demand at a rate of $1000/MWh.

> **Significant Digits**
>
> Use `round(x; digits=n)` to report values to the appropriate precision!

> **Getting Variable Output Values**
>
> `value.(x)` will report the values of a `JuMP` variable `x`, but it will return a special container which holds other information about `x` that is useful for `JuMP`. This means that you can't use this output directly for further calculations. To just extract the values, use `value.(x).data`.

> **Suppressing Model Command Output**
>
> The output of specifying model components (variable or constraints) can be quite large for this problem because of the number of time periods. If you end a cell with an `@variable` or `@constraint` command, I *highly* recommend suppressing output by adding a semi-colon after the last command, or you might find that your notebook crashes.

Problem 1 (22 points)

Your first task is to find a capacity expansion plan which minimizes total costs of investment and operation.

Problem 1.1 (2 points)

Identify and define the decision variables for the problem. Make sure to include units.

Variable	Meaning	Units
x	installed capacity of each generator	MWh
y	production capacity of each generator	MWh
NSE	non-served demand	MWh

Problem 1.2 (3 points)

Formulate the objective function. Make sure to include any needed derivations or justifications for your equation(s) and define any additional required notation beyond that introduced in Problem 1.1.

Variable	Meaning	Units
NSEcost	peanlized cost for non-served demand	$/MWh
fixedCost	cost to install generator at a given capacity	$/MWh
varCost	cost to operate generator for served energy	$/MWh

The total price of the generators is the investment cost + the operating cost.

The investment cost is simply the fixed cost to install the type of generator at a given capacity times the capacity the generator needs to be installed at:

$$\sum_{g \in \mathcal{G}} (fixedCost_g * x_g) \tag{1}$$

The operating cost is broken up into two parts:

The amount of served energy produced for each generator times the variable cost to operate each generator:

$$\sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} (varCost_g * y_{g,t}) \tag{2}$$

The amount of nonserved demand to the area times the peanlized cost for the nonserved demand:

$$\sum_{t \in \mathcal{T}} (NSEcost * NSE_t) \tag{3}$$

Adding up the terms gives us the objective function:

$$\begin{aligned} \min_{x,y,NSE} \qquad & \sum_{g \in \mathcal{G}} (fixedCost_g * x_g) + \\ & \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} (varCost_g * y_{g,t}) + \\ & \sum_{t \in \mathcal{T}} (NSEcost * NSE_t) \end{aligned} \tag{4}$$

Problem 1.3 (4 points)

Derive all relevant constraints. Make sure to include any needed justifications or derivations.

Variable	Meaning	Units
d	the energy demand each hour	MWh
c	capacity factor of each generator each hour	Unitless

Variable	Meaning	Units
T	time array of the year	hours
t	a specific hour	hour
G	generators being considered	Unitless
g	a specific generator	Unitless

First, it is a waste of cost and resources to produce more energy than the demand. It is also undesired to produce less energy than the demand, however this may be unavoidable at times hence why a non-served demand variable is included. Therefore, the amount of demand must be equal to the amount of demand served by the generators and the unserved demand:

$$\sum_{g \in \mathcal{G}} (y_{g,t}) + NSE_t = d_t \quad \forall t \in \mathcal{T} \tag{5}$$

Also, the amount of energy that is being generated can not exceed the max installed capacity of the generator, while taking into account the capacity factor of the generator at a given time:

$$x_g * c_{g,t} \geq y_{g,t} \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G} \tag{6}$$

Lastly, it is important that all installed capacities, served energy produced, and non-served demand are non-negative:

$$\begin{aligned} x_g \geq 0 \quad \forall g \in \mathcal{G} \\ NSE_t \geq 0 \quad \forall t \in \mathcal{T} \\ y_{g,t} \geq 0 \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G} \end{aligned} \tag{7}$$

The complete objective function with constraints:

$$\begin{aligned} \min_{x,y,NSE} \quad & \sum_{g \in \mathcal{G}} (fixedCost_g * x_g) + \\ & \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} (varCost_g * y_{g,t}) + \\ & \sum_{t \in \mathcal{T}} (NSEcost * NSE_t) \\ \text{subject to} \quad & \\ & \sum_{g \in \mathcal{G}} (y_{g,t}) + NSE_t = d_t \quad \forall t \in \mathcal{T} \\ & x_g * c_{g,t} \geq y_{g,t} \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G} \\ & x_g \geq 0 \quad \forall g \in \mathcal{G} \\ & NSE_t \geq 0 \quad \forall t \in \mathcal{T} \\ & y_{g,t} \geq 0 \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G} \end{aligned} \tag{8}$$

Problem 1.4 (3 points)

Implement your optimization problem in `JuMP`.

In [ ]:
```
# define sets
G = 1:length(gens.Plant) # num_gen is the number of generators
T = 1:length(demand.Hour) # number of time periods

# define indexed for array c in accordance to how array gens orders
#   the generators
geothermal = 1
coal = 2
CCGT = 3
CT = 4
wind = 5
solar = 6

# coefficents
NSEcost = 1000 # $/MWh
fixedCost = gens.FixedCost # $/MWh
varCost = gens.VarCost # $/MWh

capacity_model = Model(HiGHS.Optimizer)

# variable notation for non-decision variables
d = demand.Demand # MWh
c = zeros(length(G), length(T)) # G x T matrix of capacity factors
c[geothermal, :] = 0.85 .+ zeros(length(T)) # 85% capacity
c[coal, :] = 1 .+ zeros(length(T)) # full capacity
c[CCGT, :] = 1 .+ zeros(length(T)) # full capacity
c[CT, :] = 1 .+ zeros(length(T)) # full capacity
c[wind, :] = cap_factor.Wind # varied capacity
c[solar, :] = cap_factor.Solar # varied capacity

# decision variables
@variable(capacity_model, x[g in G] >= 0) # installed capacity
@variable(capacity_model, y[g in G, t in T] >= 0) # generated power
@variable(capacity_model, NSE[t in T] >= 0) # non served energy

# non-negativity constraints
@constraint(capacity_model,
    constraint_x[g in G], x[g in G] >= 0)
@constraint(capacity_model,
    constraint_y[g in G, t in T], y[g in G, t in T] >= 0)
@constraint(capacity_model,
    constraint_NSE[t in T], NSE[t in T] >= 0)

# main constraints
@constraint(capacity_model, constraint_capacity[g in G, t in T],
    y[g, t] <= x[g] * c[g, t]) # capacity constraint

@constraint(capacity_model, constraint_demand[g in G, t in T],
    d[t] == sum(y[g, t] for g in G) + NSE[t]) # demand constraint
```

```
@objective(capacity_model, Min,
    sum(fixedCost[g] * x[g] for g in G) +
    sum(sum(varCost[g] * y[g, t] for g in G) for t in T) +
    sum(NSEcost * NSE));

println() # supress output
```

Problem 1.5 (5 points)

Find the optimal solution. How much should the utility build of each type of generating plant? What will the total cost be? How much energy will be non-served?

In [ ]:
```
optimize!(capacity_model)
```

```
Running HiGHS 1.5.3 [date: 1970-01-01, git hash: 45a127b78]
Copyright (c) 2023 HiGHS under MIT licence terms
Presolving model
100656 rows, 56862 cols, 437328 nonzeros
56856 rows, 56862 cols, 153048 nonzeros
56856 rows, 56862 cols, 153048 nonzeros
Presolve : Reductions: rows 56856(-109590); columns 56862(-4464); elements 1
53048(-376854)
Solving the presolved LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0     0.0000000000e+00 Pr: 8760(4.52383e+06) 0s
      41818     6.4455540055e+08 Pr: 0(0); Du: 0(3.41061e-13) 2s
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Simplex   iterations: 41818
Objective value     :  6.4455540055e+08
HiGHS run time      :          2.31
```

In [ ]:
```
installed_cap = value.(x).data
println("Installed Capacity of each generator")
[println("  ", gens.Plant[g], ": ",
    round(installed_cap[g], digits=3), " MWh") for g in G];

installed_cost = sum(value.(x).data .* fixedCost)
println("Total cost of installation")
println("  \$", round(installed_cost, digits=2))
```

```
Installed Capacity of each generator
  Geothermal: 0.0 MWh
  Coal: 0.0 MWh
  NG CCGT: 1644.257 MWh
  NG CT: 687.094 MWh
  Wind: 534.512 MWh
  Solar: 1944.552 MWh
Total cost of installation
  $3.642493601e8
```

In [ ]:
```
println("Total cost of installation and operation")
println("  \$", round(objective_value(capacity_model), digits=2))
```

```
Total cost of installation and operation
  $6.4455540055e8
```

In [ ]:
```julia
non_served = sum(value.(NSE).data)
println("Demand that is non-served")
println("   ", round(non_served, digits=3), " MWh")
```

```
Demand that is non-served
   5966.048 MWh
```

Problem 1.6 (5 points)

What fraction of annual generation does each plant type produce? How does this compare to the breakdown of built capacity that you found in Problem 1.5? Do these results make sense given the generator data?

In [ ]:
```julia
y_annual = sum(value.(y[:, t]).data for t in T)
y_annual_percentage = (y_annual ./ sum(y_annual))
println("Fraction of annual generation of each generator")
[println("  ", gens.Plant[g], ": ",
    round(y_annual_percentage[g], digits=4)) for g in G];
```

```
Fraction of annual generation of each generator
  Geothermal: 0.0
  Coal: 0.0
  NG CCGT: 0.5223
  NG CT: 0.0274
  Wind: 0.0958
  Solar: 0.3545
```

NG CCGT is generating energy the most often out of all the generators (52.23%), however it has the second highest installed capacity (1644 MWh). This is because it is relatively expensive to install a higher capacity NG CCGT plant (82000 USD/MWh). Furthermore, depsite the high operating cost (30 USD/MWh) NG CCGT always has the potential to run at full capacity (capacity factor of 1), therefore NG CCGT can produce more energy over time (compared to the highest installed capacity plant, solar), which is why it generates energy the most often.

Solar is generating energy second most often (35.45%), and has the highest installed capacity (1944 MWh). This makes sense because it has one of the cheaper costs to install a higher capacity plant (70000 USD/MWh). Furthermore, while operating solar is has a very cheap operating cost (0 USD/MWh), solar is limited by its fluctuating capacity factor and therefore can not produce as much energy over time compared to NG CCGT.

Wind is generating energy the third most often (9.58%) and has the lowest installed capacity (534 MWh). This makes sense because it is the most expensive out of all the plants to install at a higher capacity (91000 USD/MWh). Despite is fluctuating capacity factor and low installed capacity, the operating cost of the wind plant is very cheap (0 USD/MWh), so it can be operating for longer than NG CT without impacting the cost.

This is inversely the case for solar vs NG CCGT because NG CCGT has a much more steady and higher average capacity factor than solar despite solar having more installed capacity, meaning it's more benefical to the cost to operate NG CCGT for longer compared to solar.

Inversely, wind has a much cheaper operating cost compared to NG CT despite NG CT having more installed capacity and higher capacity factor, meaning it's more benefical to the cost to operate wind for longer compared to NG CT.

NG CT is generating energy the least often (2.74%) and has the third highest installed capacity (687 MWh). This makes sense because it is the cheapest out of the plants to install it at a higher capacity (65000 USD/MWh). Despite always having the potential to run at full capacity (capacity factor of 1), it is used the least often due to having the highest operating cost (40 USD/MWh). The high operating cost is why it is the third highest installed capacity even though it has the cheapest cost for installed capacity.

Both geothermal and coal are not installed due to their high installed capacities (450000 USD/MWh and 220000 USD/MWh, respectively), so it is not worth the cost to install them despite having high capacity factors (0.85 and 1, respectively).

Overall, these results make sense when taking into account all the data of each generator.

Problem 2 (18 points)

The NY state legislature is considering enacting an annual CO_2 limit, which for the utility would limit the emissions in its footprint to 1.5 $MtCO_2$/yr.

Problem 2.1 (3 points)

What changes are needed to your linear program from Problem 1? Re-formulate the problem and report it in standard form.

Variable	Meaning	Units
emissions	emissions produced by each generator	tCO2/MWh
CO2_limit	the regulatory utility limit for CO2 emissions	tCO2/yr

A new constraint would need to be added to the model that takes into account emissions of each generator, the emissions of each generator is then added up to give the total emissions of the utility, which must meet regulation standards:

$$\sum_{g \in \mathcal{G}} \left(\sum_{t \in \mathcal{T}} (y_{g,t}) * emissions_g \right) \leq CO2_{limit} \tag{9}$$

Now the re-formulated objective function with constraints:

$$
\begin{aligned}
\min_{x,y,NSE} \quad & \sum_{g \in \mathcal{G}} (fixedCost_g * x_g) + \\
& \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} (varCost_g * y_{g,t}) + \\
& \sum_{t \in \mathcal{T}} (NSEcost * NSE_t) \\
\text{subject to} \quad & \\
& \sum_{g \in \mathcal{G}} (\sum_{t \in \mathcal{T}} (y_{g,t}) * emissions_g) \leq CO2_{limit} \\
& \sum_{g \in \mathcal{G}} (y_{g,t} + NSE_t) = d_t \quad \forall t \in \mathcal{T} \\
& x_g * c_{g,t} \geq y_{g,t} \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G} \\
& x_g \geq 0 \quad \forall g \in \mathcal{G} \\
& NSE_t \geq 0 \quad \forall t \in \mathcal{T} \\
& y_{g,t} \geq 0 \quad \forall t \in \mathcal{T} \quad \forall g \in \mathcal{G}
\end{aligned} \tag{10}
$$

Problem 2.2 (3 points)

Implement the new optimization problem in `JuMP`.

In [ ]:
```
# initialize values
co2_limit = 1.5e3 # tCO2/yr
emissions = gens.Emissions

# use same model and add new constraint to take emissions into account
@constraint(capacity_model, constraint_co2,
    co2_limit >= sum(sum(y[g, t] for t in T) .* emissions[g] for g in G))

println() # supress output
```

Problem 2.3 (5 points)

Find the optimal solution. How much should the utility build of each type of generating plant? What is different from your plan from Problem 1? Do these changes make sense?

In [ ]:
```
optimize!(capacity_model)
```

```
Solving LP without presolve or with basis
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0     6.4458431941e+08 Pr: 1(980058) 2s
       2009     6.7963497808e+08 Pr: 1299(8.06857e+06) 8s
       4652     6.7963792593e+08 Pr: 1384(5.45923e+06) 13s
       7694     6.7997264446e+08 Pr: 11514(1.01769e+07) 19s
       9542     7.0600483066e+08 Pr: 9894(3.48267e+07); Du: 0(8.20118e-08) 2
5s
      11724     7.4592170266e+08 Pr: 1220(1.14889e+07); Du: 0(2.77032e-07) 3
0s
      14038     7.8080111054e+08 Pr: 3369(2.75292e+06); Du: 0(2.77032e-07) 3
5s
      16235     8.5227135562e+08 Pr: 3302(2.35081e+06); Du: 0(2.77032e-07) 4
0s
      18520     9.5174208067e+08 Pr: 3333(1.58185e+06); Du: 0(3.74019e-07) 4
6s
      21245     1.0396160153e+09 Pr: 24208(7.61679e+06); Du: 0(2.77032e-07)
51s
      24629     1.0476109971e+09 Pr: 26333(1.94626e+06); Du: 0(1.9502e-07) 5
6s
      27754     1.0789436339e+09 Pr: 7165(4.53634e+06); Du: 0(2.68365e-08) 6
1s
      30305     1.1480216064e+09 Pr: 11897(544147); Du: 0(2.68376e-08) 67s
      33716     1.2579059739e+09 Pr: 0(0); Du: 0(5.38876e-11) 71s
Model   status      : Optimal
Simplex   iterations: 33716
Objective value     :  1.2579059739e+09
HiGHS run time      :         71.88
```

In [ ]:

```
installed_cap = value.(x).data
println("Installed Capacity of each generator")
[println("  ", gens.Plant[g], ": ",
    round(installed_cap[g], digits=3), " MWh") for g in G];

installed_cost = sum(value.(x).data .* fixedCost)
println("Total cost of installation")
println("  \$", round(installed_cost, digits=2))
```

```
Installed Capacity of each generator
  Geothermal: 2452.369 MWh
  Coal: 0.0 MWh
  NG CCGT: 6.675 MWh
  NG CT: 0.0 MWh
  Wind: 390.647 MWh
  Solar: 607.567 MWh
Total cost of installation
  $1.18219197233e9
```

In [ ]:

```
println("Total cost of installation and operation")
println("  \$", round(objective_value(capacity_model), digits=2))
```

```
Total cost of installation and operation
  $1.25790597389e9
```

In [ ]:

```
non_served = sum(value.(NSE).data)
println("Demand that is non-served")
```

```
println("   ", round(non_served, digits=3), " MWh")
```

```
Demand that is non-served
   75609.35 MWh
```

The types of generators built have changed in response to the new constraint of limited emissions. Now due to geothermal's low emission rate, geothermal becomes worth the cost of installation. Furthermore, due to NG CT's high emission rate, NG CT is no longer worth it's cheap installation cost, so it is not longer built. For similar reasons, NG CCGT's installed capacity has be drastically lowered because of its high emission rate. Overall, these changes make sense when considering emission rates of each generator.

Problem 2.4 (5 points)

What fraction of annual generation does each plant type produce? How does this compare to the breakdown of built capacity that you found in Problem 2.3? What are the differences between these results and your plan from Problem 1?

In [ ]:
```
y_annual = sum(value.(y[:, t]).data for t in T)
y_annual_percentage = (y_annual ./ sum(y_annual))
println("Fraction of annual generation of each generator")
[println("  ", gens.Plant[g], ": ",
    round(y_annual_percentage[g], digits=4)) for g in G];
```

```
Fraction of annual generation of each generator
  Geothermal: 0.815
  Coal: 0.0
  NG CCGT: 0.0002
  NG CT: 0.0
  Wind: 0.0716
  Solar: 0.1132
```

Geothermal is generating energy the most often (81.50%) and had the highest installed capacity (2452 MWh). This makes sense because despite the expensive installation cost (450000 USD/MWh), geothermal has a very steady and high capacity factor (0.85), operating cost (0 USD/MWh) and a very low emission rate (0 tCO2/MWh). This makes up for the expensive cost for a higher installed capacity plant.

The more constant and high capacity factor of geothermal is why it has a higher installed capacity and is operated for longer compared to solar and wind.

Solar is generating energy the second most often (11.32%) and has the second highest installed capacity (607 MWh). This makes sense because it has a low emission rate (0 tCO2/MWh), a very cheap operation cost (0 USD/MWh), and has a relatively cheap cost to install a higher capacity plant (70000 USD/MWh), these factors make up for the fluctuating capacity factor.

Wind is generating energy the third most often (7.16%) and has the third highest installed capacity (390 MWh). This makes sense because it has a low emission rate (0 tCO2/MWh), a very cheap operation cost (0 USD/MWh), these factors make up for the expensive cost to install a higher capacity plant (91000 USD/MWh) and fluctuating capacity factor.

The higher cost compared to solar to install a higher capacity wind plant is why solar has a higher installed capacity and operates for longer.

NG CCGT is generating energy the least often (0.0002%) and has the lowest installed capacity (6 MWh). This makes sense because it has the lowest non-zero emission rate (0.43 tCO2/MWh) and very steady and always has the potential to run at full capacity (capacity factor of 1). Due to its high operating cost (30 USD/MWh), relatively expensive isntallation cost (82000 USD/MWh), and having a non-zero emission rate, it's not worth operating NG CCGT at a high capacity nor for very long.

Coal and NG CT have the highest emission rates (1 and 0.55, respectively), meaning it is not worth the cost to install and operate them because they contribute too much to the overall emission rate.

This plan is different from the plan in Problem 1 because this plan minimizes how many generators with non-zero emission rates are installed and operated because of the strict regulatory limit on CO2 emissions the utility can have. This results in the plan in Problem 1 being able to install and operate both natural gas generators (NG CCGT and NG CT), while the plan can only afforable, emissions-wise, to operate NG CT plant at a very small installed capacity and only for a very small fraction of the time. Furthermore, the CO2 restriction forced this plan to take full advantage of the zero emission rate generators, which is why this plan utilized geothermal while the plan in Problem 1 does not, as the plan in Problem 1 does not need to.

Problem 2.5 (2 points)

What would the value to the utility be of allowing it to emit an additional 1000 tCO_2/yr? An additional 5000?

In [ ]:
```
# the shadow price of the CO2 constraint would show how the value ($) of
#   the utility would change if the regulatory CO2 limit is weakened by 1
#   unit of the constraint, which is 1 tCO2/yr
shadow_price_co2 = shadow_price(constraint_co2)

# shadow price has units of $/(tCO2/yr)
shadow_price_1000 = shadow_price_co2 * 1000
shadow_price_5000 = shadow_price_co2 * 5000

println("Cost would change by ", round(shadow_price_1000, digits=2),
    " USD \n    if the utility could emit an additional 1000 tCO2/yr")
println("Cost would change by ", round(shadow_price_5000, digits=2),
    " USD \n    if the utility could emit an additional 5000 tCO2/yr")
```

```
Cost would change by -1.88695965e6 USD
    if the utility could emit an additional 1000 tCO2/yr
Cost would change by -9.43479825e6 USD
    if the utility could emit an additional 5000 tCO2/yr
```

References

List any external references consulted, including classmates.

BEE 4750 10/13 Lecture "Power Systems: Generating Capacity Expansion" Slides & provided code (syntax for vectorized code for JuMP)