Edit on GitHub
Edit on GitHub Prose

# Hourly energy model example 2: Variable supply and flat demand (python code included)

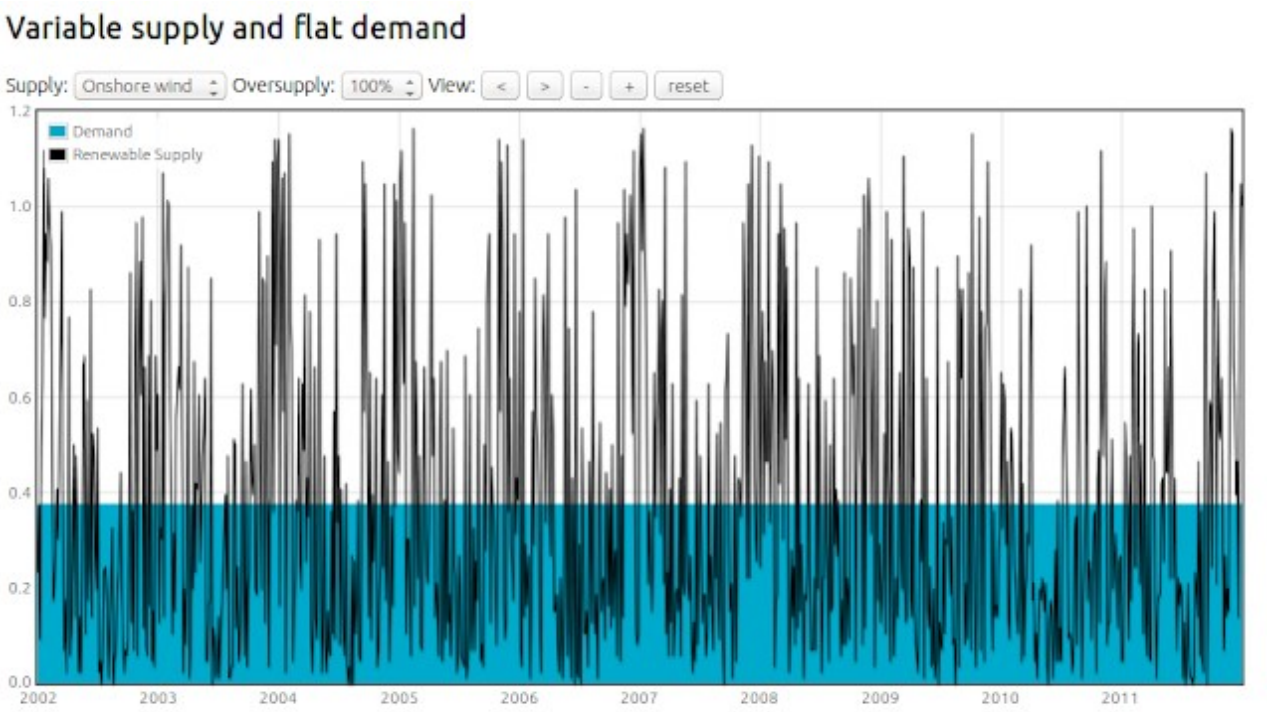
2015-06-12 07:52:00 +00002015-06-12 07:52:00 +0000 Trystan Lea
four minutes reading time

- zerocarbonbritain

Comments

The second example in the hourly energy modelling tool models the degree of supply/demand matching between a variable renewable supply consisting of a single renewable energy generation type and a flat electricity demand profile.

A flat demand may not of course be particularly realistic and the more complex examples later on address this, but I've used it here just to illustrate this particular simple example case.




```
Generator capacity: 1.165kW
Capacity factor: 32.3%

Total supply: 3300.2kWh/y
Total demand: 3300.0kWh/y

Exess generation: 1126kWh/y
Unmet demand: 1126kWh/y

Percentage of demand supplied directly 65.9%
Percentage of time supply was more or the same as the demand 40.1%
```

Online tool: http://openenergymonitor.org/energymodel > navigate to *2. variable supply and flat demand*

The demand is subtracted from the supply for every hour in the 10 year period and the total amount of unmet demand and excess generation is measured as well as the amount of time the supply was more than or equal to the demand.

The demand level is set to an annual average electricity demand of 3300 kWh which is the average UK household annual electricity consumption. The amount of installed capacity is set to match this demand on the 10 year basis of the dataset.

Running the model for each of each generation type, matching total 10 year supply to total 10 year demand of 3300 kWh x 10 we get the following results:

| | Onshore wind | Offshore wind | Wave | Tidal | Solar |
|---|---|---|---|---|---|
| Installed capacity | 1.17kW | 0.79kW | 1.33kW | 1.58kW | 3.98kW |

| | | | | | |
|---|---|---|---|---|---|
| Percentage of demand supplied directly | 65.9% | 76.4% | 73.9% | 57.7% | 40.6% |
| Percentage of time demand is more or the same as the supply | 40.1% | 46.2% | 45.3% | 38.6% | 32.1% |

We can see again here that offshore wind is the clear winner with the lowest installed capacity requirement and highest level of supply/demand matching. Perhaps an interesting result is how less predictable technologies such as wind and wave provide greater levels of matching than power from tidal which is very predictable.

## Python example source code

Alongside the online javascript modelling tool there are a series of python versions of the examples which are simpler to follow as they dont include all the code to create the visual output, they just print out the main results at the end.

I've highlighted the main parts in bold below:

```
# dataset index:
# 0:onshore wind, 1:offshore wind, 2:wave, 3:tidal, 4:solar, 5:traditional electricity
gen_type = 1

installed_capacity = 0.785 # kW

annual_house_demand = 3300 # kWh
house_power = (annual_house_demand * 10.0) / 87648

# Load dataset
with open("../dataset/tenyearsdata.csv") as f:
    content = f.readlines()
hours = len(content)

print "Total hours in dataset: "+str(hours)+" hours"
print

total_supply = 0
total_demand = 0

exess_generation = 0
unmet_demand = 0

hours_met = 0


# for every hour in the dataset
for hour in range(0, hours):
    values = content[hour].split(",")

    # calculate the supply
    supply = float(values[gen_type]) * installed_capacity
    total_supply += supply
```

```
    # calculate demand
    demand = house_power
    total_demand += demand

    # subtract demand from supply to find the balance
    balance = supply - demand

    # record the total amount of exess and unmet demand
    if balance>=0:
        exess_generation += balance
        hours_met += 1
    else:
        unmet_demand += -balance

capacity_factor = total_supply / (installed_capacity*hours) * 100

prc_demand_supplied = ((total_demand - unmet_demand) / total_demand) * 100

prc_time_met = (1.0 * hours_met / hours) * 100




# print out the results
print "Installed capacity: %s kW" % installed_capacity
print "Capacity factor: %d%%" % capacity_factor
print
print "Total supply: %d kWh" % total_supply
print "Total demand: %d kWh" % total_demand
print
print "Exess generation %d kWh" % exess_generation
print "Unmet demand %d kWh" % unmet_demand
print
print "Percentage of demand supplied directly %d%%" % prc_demand_supplied
print "Percentage of time supply was more or the same as the demand %d%%" % prc_time_met
```

To engage in discussion regarding this post, please post on our Community Forum.

## Recent Posts

- Introducing emonDC
- 2018 Development Review
- Demand Side Response Development
- Emoncms Blue Theme
- emonPi Remote Access with Dataplicity
- Introducing IoTaWatt
- CydYnni – Energy Local, community hydro smart grid
- Advances Wales Magazine Article
- Emoncms Android App V2
- emonPi Network Setup Wizard