Version 10 - User experiences at Vattenfall

2019-03-14, Roger Halldin

Agenda

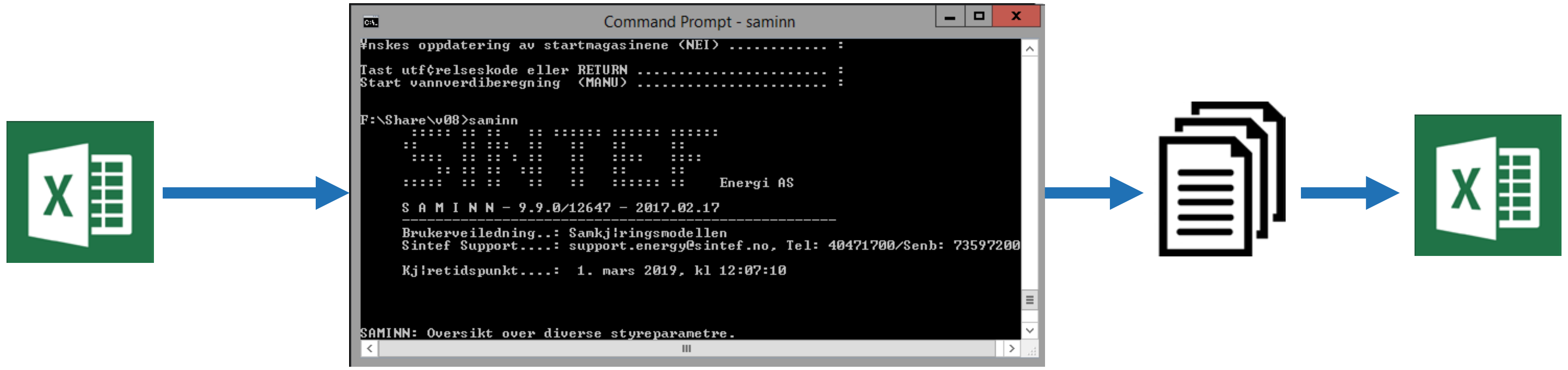
1. **History of EMPS at Vattenfall**
2. **Data handling with Python and Power BI**
3. **Some thoughts regarding V.10**

History of EMPS at Vattenfall

- Vattenfall have used EMPS since the 90’s
- All the experienced users (+15 years of EMPS) are retired
- New users with other requirements for development as well as programming knowledge
- Need for streamlining and automatisation and more time for analysing

In the beginning…

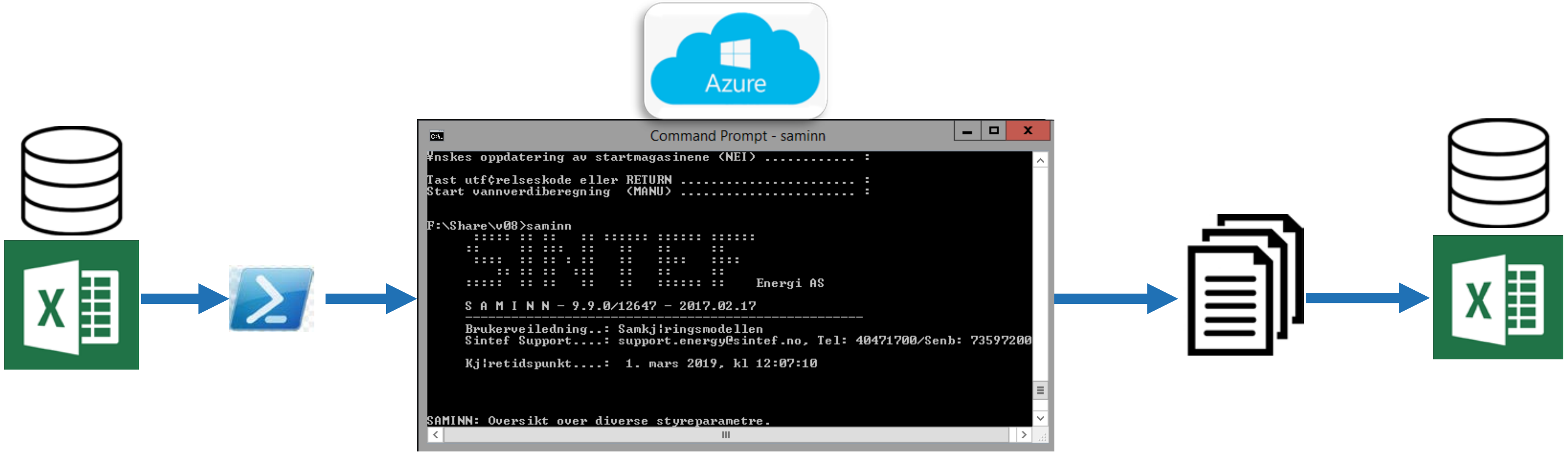
- Running EMPS from dos-prompt
- Some Fortran code to help running and small scripts
- Excel for indata and results
- A lot of manual work, copy and paste



```
#nskes oppdatering av startmagasinene (NEI) ............ :

Tast utf¢relseskode eller RETURN ....................... :
Start vannverdiberegning  (MANU) ....................... :


F:\Share\v08>saminn
      ::::: :: ::   :: :::::: :::::: ::::::
     ::      :: :::  ::   ::    ::     ::
      ::::  :: :: : ::   ::    ::::   ::::
         :: :: ::  :::   ::    ::     ::
     :::::  :: ::   ::   ::    :::::: ::    Energi AS

     S A M I N N - 9.9.0/12647 - 2017.02.17
     ----------------------------------------------------
     Brukerveiledning..: Samkj|ringsmodellen
     Sintef Support....: support.energy@sintef.no, Tel: 40471700/Senb: 73597200

     Kj|retidspunkt....:  1. mars 2019, kl 12:07:10



SAMINN: Oversikt over diverse styreparametre.
```

Second step

- Excel to steer EMPS with VBA and PowerShell
- Results in Excel
- Most input data automated from db
- Run in cloud

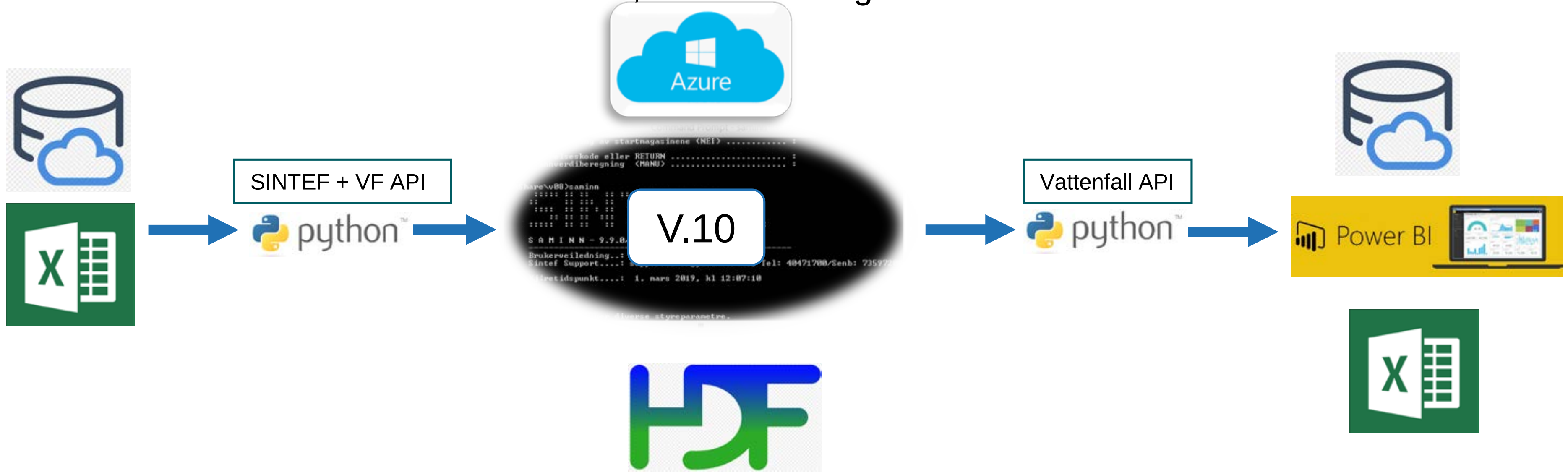


```
¥nskes oppdatering av startmagasinene (NEI) ............ :

Tast utf¢relseskode eller RETURN ........................ :
Start vannverdiberegning  (MANU) ........................ :


F:\Share\v08>saminn
      ::::: :: ::   :: :::::: :::::: ::::::
     ::     :: :::  ::   ::   ::     ::
      ::::  :: :: : ::   ::   ::::   ::::
          :: :: ::  :::   ::   ::     ::
     :::::  :: ::   ::   ::   :::::: ::    Energi AS

     S A M I N N - 9.9.0/12647 - 2017.02.17
     ----------------------------------------------------

     Brukerveiledning..: Samkj¦ringsmodellen
     Sintef Support....: support.energy@sintef.no, Tel: 40471700/Senb: 73597200

     Kj¦retidspunkt....:  1. mars 2019, kl 12:07:10



SAMINN: Oversikt over diverse styreparametre.
```

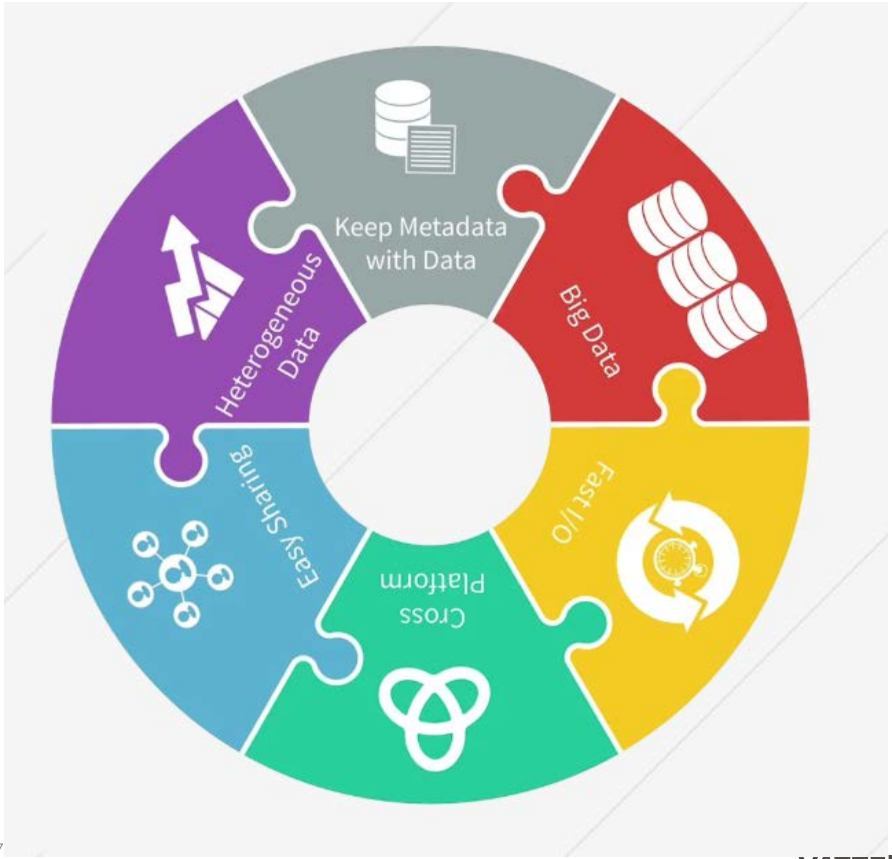
VATTENFALL

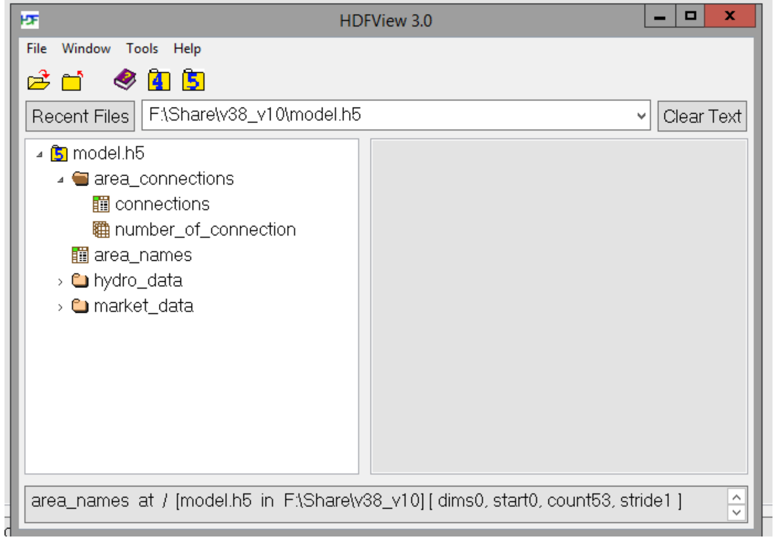
Third step

- Database and Excel for data input
- Use Python for data import, transformation and validation
- Run in cloud
- Possible for more automatic runs, data checking and transformations



What is HDF5?

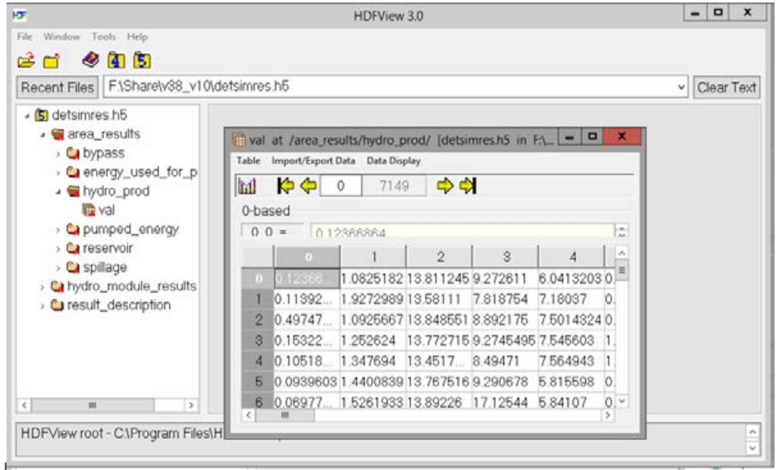




F:\Share\v38_v10\model.h5

- model.h5
  - area_connections
    - connections
    - number_of_connection
  - area_names
  - hydro_data
  - market_data

area_names at / [model.h5 in F:\Share\v38_v10] [dims0, start0, count53, stride1]



F:\Share\v38_v10\detsimres.h5

- detsimres.h5
  - area_results
    - bypass
    - energy_used_for_p
    - hydro_prod
      - val
    - pumped_energy
    - reservoir
    - spillage
  - hydro_module_results
  - result_description

HDFView root - C:\Program Files\H

val at /area_results/hydro_prod/ [detsimres.h5 in F:\...

0 7149

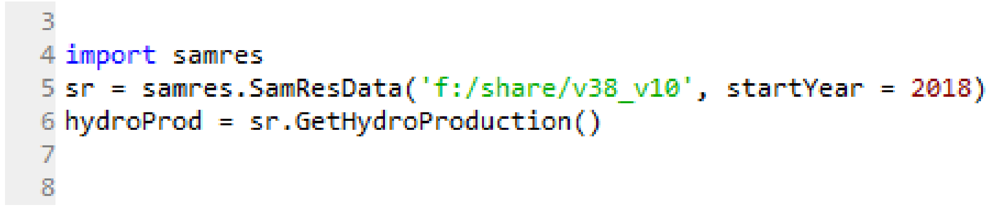
0-based

0, 0 = 0.12366864

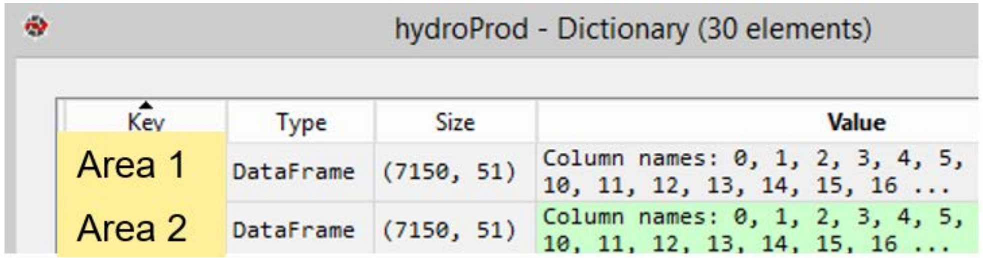
	0	1	2	3	4	
0	0.12366	1.0825182	13.811245	9.272611	6.0413203	0.
1	0.11392...	1.9272989	13.58111	7.818754	7.18037	0.
2	0.49747...	1.0925667	13.848551	8.892175	7.5014324	0.
3	0.15322...	1.252624	13.772715	9.2745495	7.545603	1.
4	0.10518...	1.347694	13.4517...	8.49471	7.564943	1.
5	0.0939603	1.4400839	13.767516	9.290678	5.815598	0.
6	0.06977...	1.5261933	13.89226	17.12544	5.84107	0.

Data handling using Python

- A few lines of code to get data into dataframe (results)

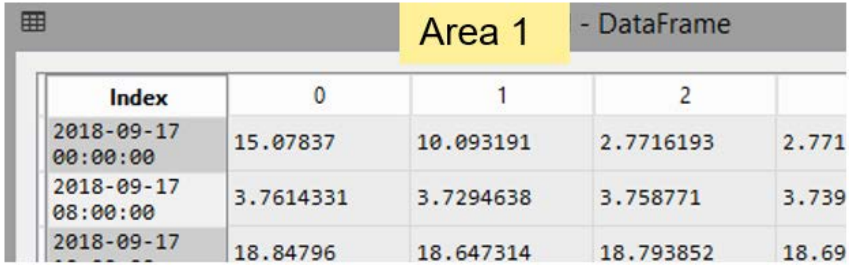


```python

import samres
sr = samres.SamResData('f:/share/v38_v10', startYear = 2018)
hydroProd = sr.GetHydroProduction()


```



hydroProd - Dictionary (30 elements)

Key	Type	Size	Value
Area 1	DataFrame	(7150, 51)	Column names: 0, 1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16 ...
Area 2	DataFrame	(7150, 51)	Column names: 0, 1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16 ...



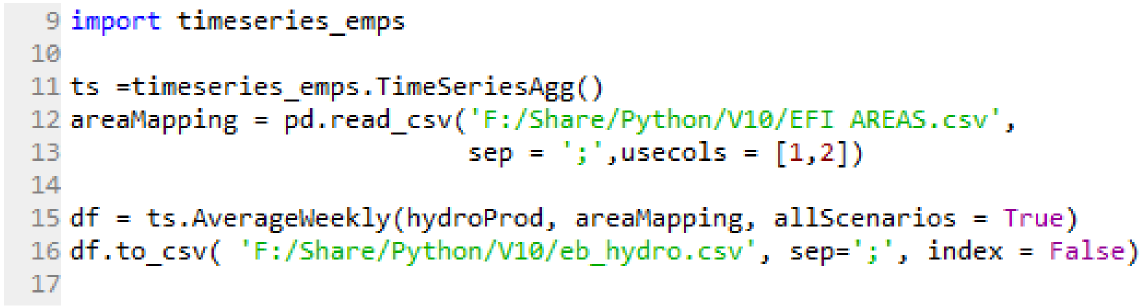
Area 1 - DataFrame

Index	0	1	2	
2018-09-17 00:00:00	15.07837	10.093191	2.7716193	2.771
2018-09-17 08:00:00	3.7614331	3.7294638	3.758771	3.739
2018-09-17	18.84796	18.647314	18.793852	18.69

- Class SamResData simple and generic
  - areas, time resolution and so on read from .h5 and thus generic
    - Example - changing from 5 time steps to hourly will not change any code
    - Example – Adding new cable or thermal plant will not change any code

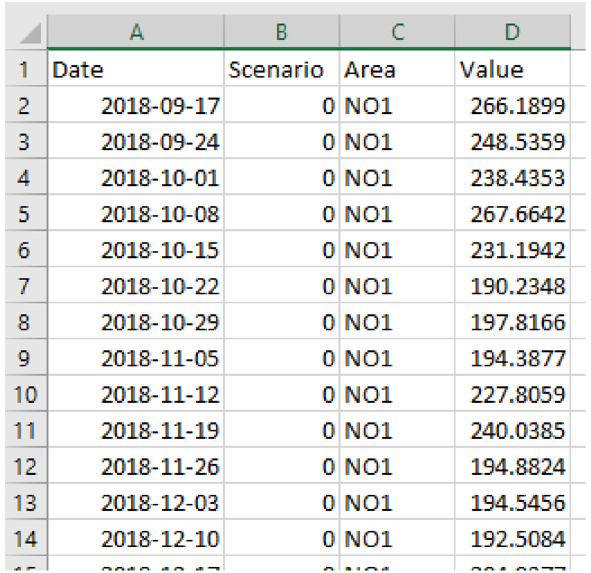
Data handling using Python

- A few more lines of code for typical aggregation:



```python
import timeseries_emps

ts =timeseries_emps.TimeSeriesAgg()
areaMapping = pd.read_csv('F:/Share/Python/V10/EFI AREAS.csv',
                             sep = ';',usecols = [1,2])

df = ts.AverageWeekly(hydroProd, areaMapping, allScenarios = True)
df.to_csv( 'F:/Share/Python/V10/eb_hydro.csv', sep=';', index = False)
```

- Getting data and creating files less than 30 seconds
  - Demand, hydro, wind, thermal, exchange, inflows, reservoirs
  - Aggregated to NP areas, weekly, all scenarios



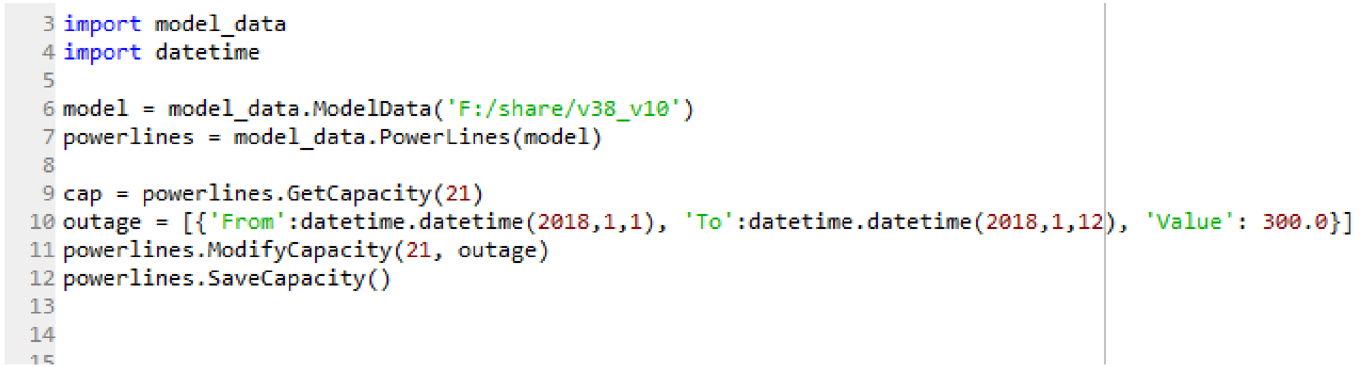
	A	B	C	D
1	Date	Scenario	Area	Value
2	2018-09-17	0	NO1	266.1899
3	2018-09-24	0	NO1	248.5359
4	2018-10-01	0	NO1	238.4353
5	2018-10-08	0	NO1	267.6642
6	2018-10-15	0	NO1	231.1942
7	2018-10-22	0	NO1	190.2348
8	2018-10-29	0	NO1	197.8166
9	2018-11-05	0	NO1	194.3877
10	2018-11-12	0	NO1	227.8059
11	2018-11-19	0	NO1	240.0385
12	2018-11-26	0	NO1	194.8824
13	2018-12-03	0	NO1	194.5456
14	2018-12-10	0	NO1	192.5084

VATTENFALL

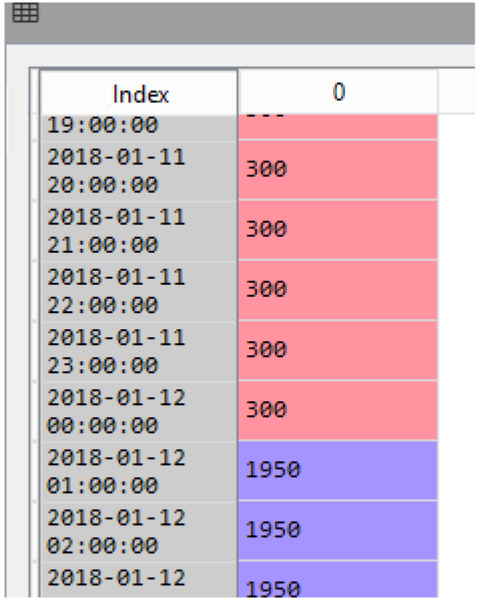
Data handling using Python

Input Data

- A few lines of code to get capacity for line and modify it:



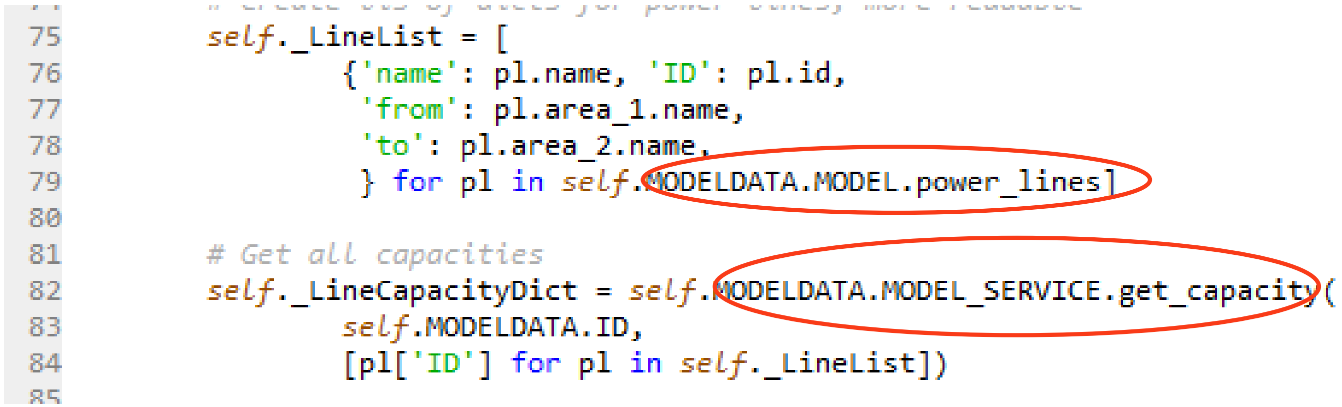
```python
import model_data
import datetime

model = model_data.ModelData('F:/share/v38_v10')
powerlines = model_data.PowerLines(model)

cap = powerlines.GetCapacity(21)
outage = [{'From':datetime.datetime(2018,1,1), 'To':datetime.datetime(2018,1,12), 'Value': 300.0}]
powerlines.ModifyCapacity(21, outage)
powerlines.SaveCapacity()
```



Index	0
19:00:00	...
2018-01-11 20:00:00	300
2018-01-11 21:00:00	300
2018-01-11 22:00:00	300
2018-01-11 23:00:00	300
2018-01-12 00:00:00	300
2018-01-12 01:00:00	1950
2018-01-12 02:00:00	1950
2018-01-12	1950

Data handling using Python

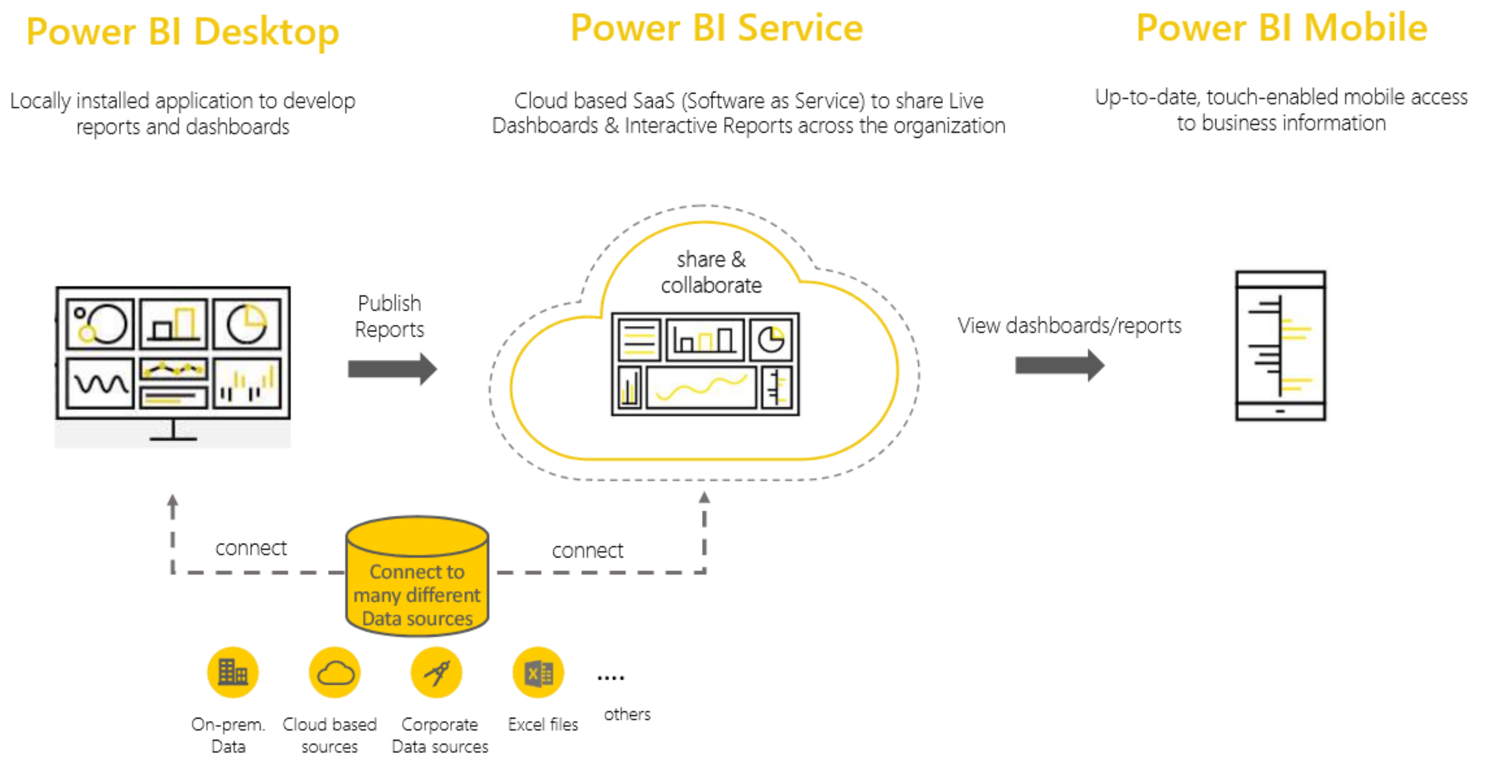
- Class PowerLines uses Sintef’s API
  - Functionality to fit most common tasks, initiation is reading all lines and capacities



```python
        # Create list of dicts for power lines, more readable
        self._LineList = [
            {'name': pl.name, 'ID': pl.id,
             'from': pl.area_1.name,
             'to': pl.area_2.name,
             } for pl in self.MODELDATA.MODEL.power_lines]

        # Get all capacities
        self._LineCapacityDict = self.MODELDATA.MODEL_SERVICE.get_capacity(
            self.MODELDATA.ID,
            [pl['ID'] for pl in self._LineList])
```

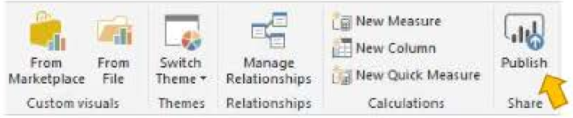
- Added functionality:
  - Get capacity from one area to another by name (both directions)
  - Modify capacity for a list of from/to dates (as in example)
  - And so on..

Result Handling: Basic Components of Power BI

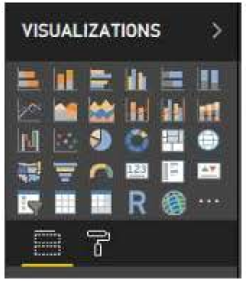

Power BI Desktop
Locally installed application to develop reports and dashboards
Power BI Service
Cloud based SaaS (Software as Service) to share Live Dashboards & Interactive Reports across the organization
Power BI Mobile
Up-to-date, touch-enabled mobile access to business information
Publish Reports
share & collaborate
View dashboards/reports
connect
Connect to many different Data sources
connect
On-prem. Data
Cloud based sources
Corporate Data sources
Excel files
....
others

Result Handling: Power BI Desktop – How biuld your own report

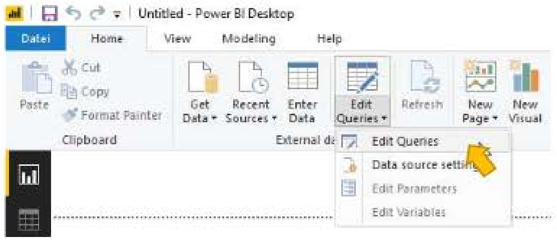
Share Reports → Publish to powerbi.com



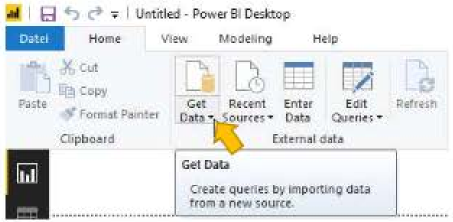
Create Visuals



Transform and clean data in Query Editor



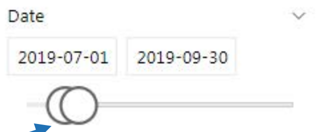
Connect to Data



Get Data

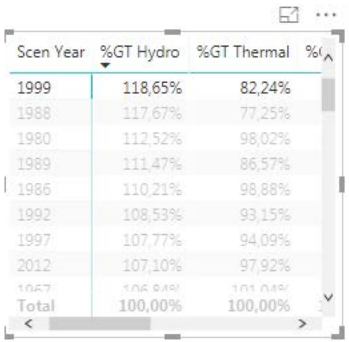
Create queries by importing data from a new source.

Examples of analysing result in Power BI


Date

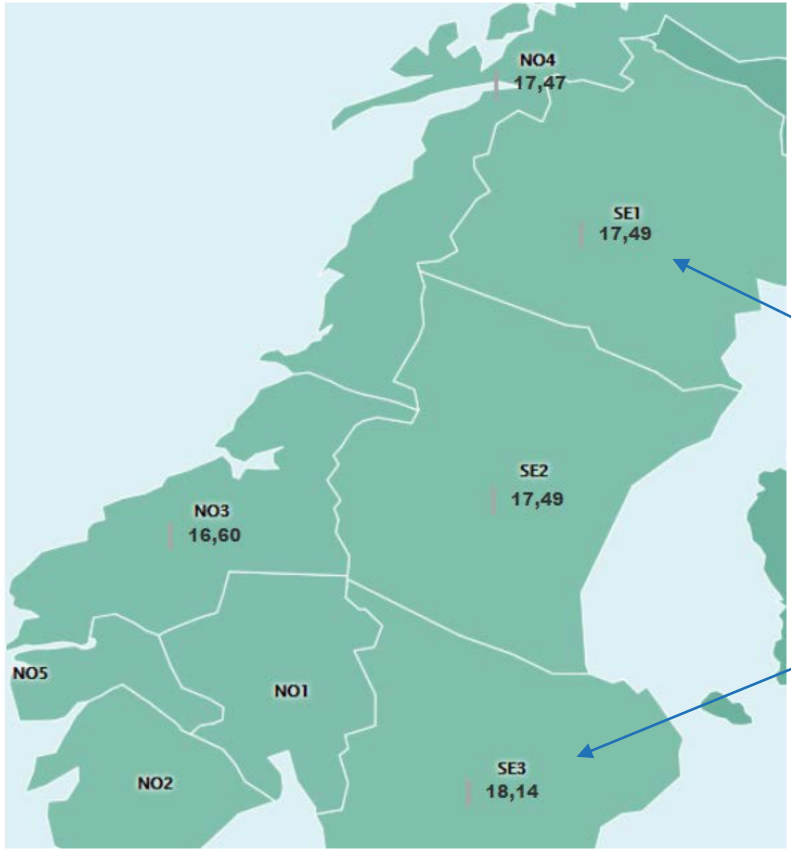
2019-07-01 2019-09-30

Drag to change period



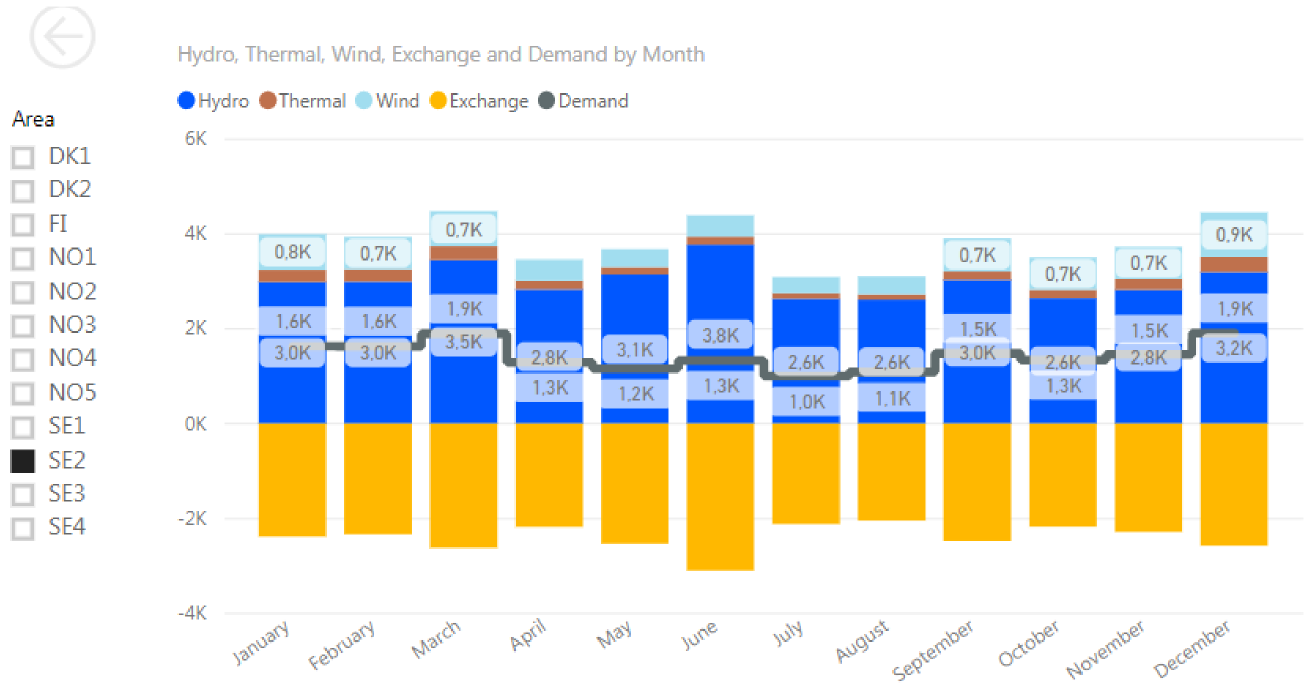
Scen Year	%GT Hydro	%GT Thermal
1999	118,65%	82,24%
1988	117,67%	77,25%
1980	112,52%	98,02%
1989	111,47%	86,57%
1986	110,21%	98,88%
1992	108,53%	93,15%
1997	107,77%	94,09%
2012	107,10%	97,92%
1967	106,84%	101,04%
Total	100,00%	100,00%

Choose specific weather years, or all weather years



Corresponding price or what ever fundamental variable you choose will change automatically

Examples of analysing result in Power BI



With a click:

- Change price area
- Change time discretization, e.g. daily, weekly, monthly, yearly
- View different fundamentals, e.g. production, price, demand etc

Some thought about version 10

- ”SINTEF Upgrade” from v9 to v10 works and even found issues in our own data that was wrong
- Saminn, enmdat, med, etc -> LTM – Long Term Model
- Better error checking (from Sintef and Vattenfall) and warnings in v.10 compared to v.9
- Easier to compare input and output data
- Faster and easier to get the result from HDF
  - Get the result when you need it and fast!
  - No need to use kurvetegn to get textfiles
  - Easy transforming between different time resolutions and areas using Python, Vattenfall API
- Easier to use time series
- EMPS date format = Python ISO-calendar
- Better separation of model and input data