NVE
Reguleringsmyndigheten
for energi – RME

RME EKSTERN RAPPORT

Nr. 1/2021

Developing Methods for Combining Data that Can Be Used for Calculating Power Distance

Multiconsult

RME Ekstern rapport nr. 1/2021

Developing methods for combining data that can be used for calculating power distance

Redaktør: Tore Langset

Forfatter: Multiconsult AS

Forsidefotofotograf: Ole-Petter Kordahl (fotokollasj)

ISSN: 2535-8243

ISBN: 978-82-410-2094-0

Sammendrag: Multiconsult AS har på oppdrag fra RME laget et rammeverk for å sammenstille data som er nødvendig for å beregne oppgavevariablene effekt- og energiavstand. Disse beregningene krever detaljerte data om både strømforbruk og hvor forbruk, utveksling og produksjon av kraft befinner seg i nettet. Beregningen krever også at disse dataene kobles sammen. Rapporten er en del av et større arbeid for å videreutvikle modellen som beregner nettselskapenes effektivitet.

Emneord: Inntektsramme, nettselskaper, økonomisk regulering, effektivitet, effektavstand, energiavstand, eksogene oppgavevariabler, Elhub, nettdata

Reguleringsmyndigheten for energi
Middelthuns gate 29
Postboks 5091 Majorstuen
0301 Oslo

Telefon: 22 95 95 95
E-post: rme@nve.no
Internett: www.reguleringsmyndigheten.no

januar, 2021

Forord

Reguleringsmyndigheten for energi (RME) regulerer nettselskapenes inntekter. Formålet er å bidra til effektiv drift, utnyttelse og utvikling av nettet. RME gjennomfører hvert år en effektivitetsanalyse som måler selskapene mot hverandre, og rangerer dem ut fra hvor mye ressurser de bruker på å bygge, drifte og vedlikeholde nettinfrastrukturen. Nettselskapenes avkastning bestemmes deretter av hvor kostnadseffektivt de løser sine oppgaver.

RME har i de to siste årene utforsket nye variabler som kan brukes i effektivitetsanalysene. Effektavstand beskriver hvor mye effekt hvert nettselskap skal levere, og over hvor lang avstand denne effekten må transporteres. Energiavstand måles på samme måte, men gjenspeiler hvor mye energi som skal fraktes over ulike avstander over en periode.

En matematisk beregning av energi- og effektavstand krever omfattende data om både strømforbruk og hvor forbruk, utveksling og produksjon av kraft befinner seg i nettet. Beregningen krever også at disse dataene kobles sammen. Vi har bedt Multiconsult lage et rammeverk for innsamling, rensing og kobling av data som er nødvendig for å beregne effektavstand og tilhørende variabler, og arbeidet deres presenteres i denne rapporten. Alle vurderinger og konklusjoner i rapporten er konsulentenes egne.

En referansegruppe bestående av Glitre Nett, Jæren Everk, Klepp Energi og Mørenett har bistått med bransjekunnskap og data som har blitt brukt for å verifisere metodene. Vi er takknemlig for den innsatsen disse selskapene har bidratt med i prosjektet. Selskapene har imidlertid intet ansvar for konsulentens konklusjoner

Vi inviterer alle til å komme med innspill til arbeidet innen 15. mars 2021. Tilbakemeldinger merkes med referansenummer 202100566 og sendes til rme@nve.no. Vi tar med oss Multiconsult sitt arbeid og innspill på dette i det videre arbeidet med reguleringsmodellen.

Oslo, januar 2021

Ove Flataker
direktør
Reguleringsmyndigheten for energi

Tore Langset
seksjonssjef

REPORT

Developing Methods for Combining Data that Can Be Used for Calculating Power Distance

CLIENT

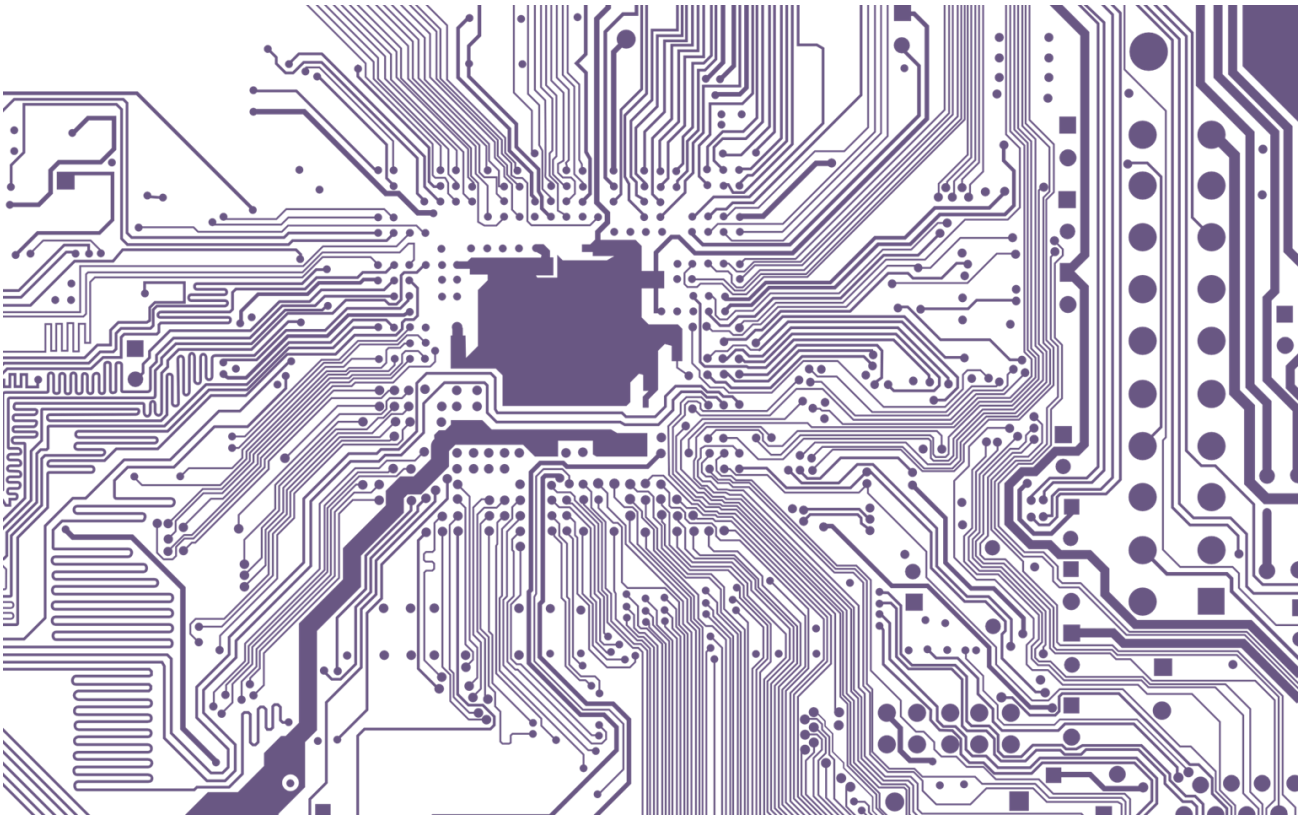
Norges vassdrags- og energidirektorat (NVE-RME)

SUBJECT

Final Report

DATE: / REVISION: November 6, 2020 / 03

DOCUMENT CODE: 10219088-TVF-RAP-001-03



Multiconsult

This report has been prepared by Multiconsult on behalf of Multiconsult or its client. The client's rights to the report are regulated in the relevant assignment agreement. If the client provides access to the report to third parties in accordance with the assignment agreement, the third parties do not have other or more extensive rights than the rights derived from the client's rights. Any use of the report (or any part thereof) for other purposes, in other ways or by other persons or entities than those agreed or approved in writing by Multiconsult is prohibited, and Multiconsult accepts no liability for any such use. Parts of the report are protected by intellectual property rights and/or proprietary rights. Copying, distributing, amending, processing or other use of the report is not permitted without the prior written consent from Multiconsult or other holder of such rights.

Cover photo by OpenClipart-Vectors from Pixabay.

Multiconsult

REPORT

PROJECT	**Developing Methods for Combining Data that Can Be Used for Calculating Power Distance**	DOCUMENT CODE	10219088-TVF-RAP-001
SUBJECT	Final Report	ACCESSIBILITY	Open
CLIENT	**Norges vassdrags- og energidirektorat (NVE-RME)**	PROJECT MANAGER	Magnus Sletmoe Dale
CONTACT	Ole-Petter Kordahl	PREPARED BY	Peder Persen Fostvedt, Thomas Haugstenrød, Jan Ohlenbush, Magnus Sletmoe Dale
		RESPONSIBLE UNIT	10105080 Renewable Energy Advisory Services

03	13.01.2021	Table 8 column name swap	msd	joh	joa
02	07.01.2021	Correction to regulatory drivers in introductory text	msd	joh	joa
01	06.11.2020	Final Report	msd	joh	joa
00	15.10.2020	Draft Final Report	msd	joh	msd
REV.	DATE	DESCRIPTION	PREPARED BY	CHECKED BY	APPROVED BY

TABLE OF CONTENTS

Contents

TABLE OF EXHIBITS INCLUDED IN THIS REPORT

Figures

Tables

1 Introduction

1.1 Mandate/Introduction/Background

The Norwegian power market currently finds itself in a transitional phase characterized by substantial change to legislation, infrastructure, and consumer/producer behavior.

This transition is driven by a diverse set of factors of which many falls within the context of electrification of society and renewable energy build-out. Some of these include:

- Increase in distributed power generation across both industrial, commercial, and private segments
- Strong uptake in demand for electric vehicles; increased electrification of society in general
- Policies/regulation/support programs that spur investments in Renewable Energy Sources (RES)-fueled power generation, across different size segments. These include the Elcertificates system[1] driving growth in utility-scale production facilities, Plusskundeordningen allowing producers up to 100 kW to sell surplus electricity back to the grid, and Enova investment subsidies for residential solar PV plants sized up to 15 kWp.
- Increased digitization of the Norwegian power market, driven by milestone initiatives such as the installation of automatic meter readings (AMS) in all Norwegian households in by January 2019[2], and the centralization of key power market data in Elhub, launched in February 2019[3]

Together, this creates an exciting backdrop against which both Norwegian power market players and regulators may seek new opportunities for increased efficiency. This applies not least to regulatory bodies such as Norwegian Energy Regulatory Authority (NVE-RME).

1.2 Power distance and related R&D initiatives

A key task of NVE-RME is annually setting the amount Distribution System Operators (DSOs) can collect in revenue through network tariffs.

A Data Envelopment Analysis (DEA) tool is used as a benchmarking tool to achieve this. Metrics customer base size, number of substations and high-voltage (HV) transmission lines currently serve as output variables to the DEA model. However, it is likely that the variable “power distance” more precisely, independently, and exogenously capture the real task of the DSOs.

The power distance variable was defined in a report published in 2018. As illustrated in Table 1, five other R&D initiatives, including this assignment, have been launched in order to devise power distance and related concepts.

[1] Open for new plants that start generation by year-end 2021.
[2] https://www.nve.no/reguleringsmyndigheten/stromkunde/smarte-strommalere-ams/
[3] https://www.statnett.no/en/about-statnett/news-and-press-releases/news-archive-2019/elhub-is-now-operational/

Table 1 - R&D initiatives initiated by NVE-RME on power distance and related concepts (Source: NVE)

Project	NVE Report #	Focus	Year	Status
Investigating the minimal power distance	5/2019	Theory	2018	Completed
Developing power and energy distances	1/2019	Methodology and application	2019	Completed
Developing geographical datasets	N/A	Data handling	2018	Completed
Developing methods for combining data that can be used for calculating power and energy distance	N/A	Data handling	2020	**This project**
Developing and testing methods for calculating power and energy distance.	N/A	Theory, methodology and application	2020	Not started
Developing new variables for measuring the task of supplying reliability	N/A	Theory, methodology and application	2020	Not started

In parallel to this assignment, consultancy group Thema developed infrastructure for calculating the power distance variable based on datasets produced by Multiconsult. Multiconsult worked closely with this consultancy to ensure compatibility between our final datasets and Thema's interface descriptions.

1.3 Report structure

This project proposes a framework for collecting, cleaning, and combining data needed to calculate the power distance variable.

Specifically, in Chapter 2, we devise a Formal Framework ("the framework") for combining data needed to calculate the power distance variable. This provides a theoretical and high-level methodology to solving the task at hand, ignores lower-level considerations related to infrastructure, software, and code.

In Chapter 3 and 4, we discuss the application of the Formal Framework to datasets obtained four DSOs participating in a project Reference Group, with certain adaptations, and key results.

In Chapter 5, we identify aspects that are relevant for the potential, future roll-out of the framework on a national scale. Lastly, all relevant material, including such as specifications and code, are all enclosed as appendices in Chapter 6.

Final datasets are transferred to NVE-RME separately through a designated and secure file transfer system. This document, including all appendices, and resulting datasets described in Chapter 6.2 constitute the deliverables of this assignment.

2 Proposed Formal Framework

2.1 Key concepts

Multiconsult has endeavored to establishing a robust and flexible framework for combining solicited datasets.

Key characteristics of the proposed framework include (key concepts in bold):

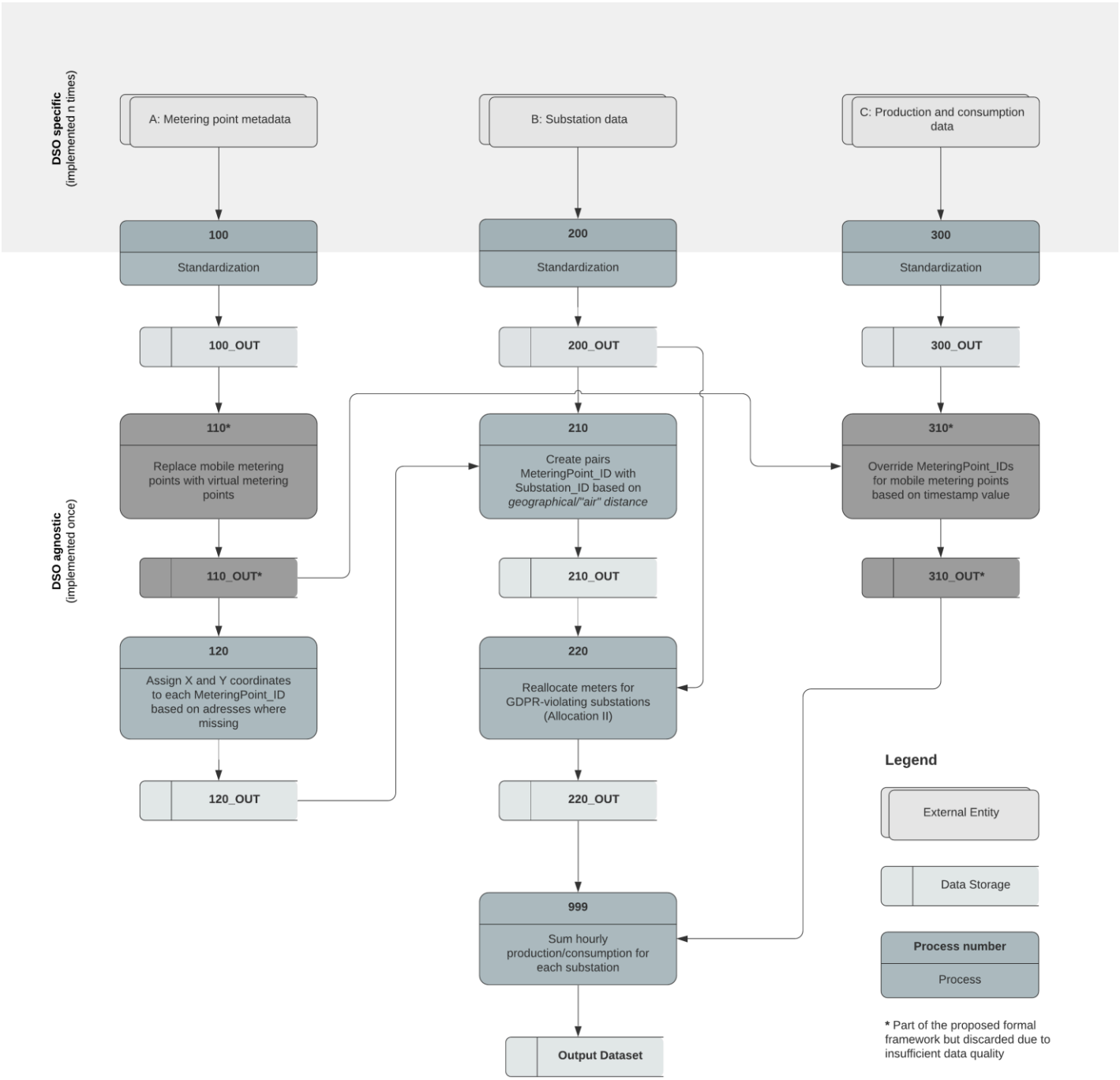
- The proposed dataset is essentially a collection of **processes** that perform some manipulation to **datasets**. These processes may vary in complexity, ranging from relatively straightforward lookups and replace by-operations, to more complex allocation and data restructuring tasks. Yet these are defined independently from its implementation, thereby ensuring validity even as access to software, processing and storage power evolves over time.
- A process may accept any number of datasets as parameters and yields a resulting output dataset which may or may not feed into other processes. A process K will yield one dataset named K_OUT. While only the last process yields a final dataset, **999_OUT**, all other processes yield intermediary datasets that in turn feed into other processes.
- Processes are assigned the smallest index possible taking data dependencies into consideration. This means that a process *k* will only depend on process numbers <k. As an example, process 120 will depend solely on datasets created by processes numbered <120. In addition, processes are indexed such that they can be carried out incrementally as per their assigned index.
- The framework relies on three datasets provided by the DSO which are discussed in further detail in chapter 3.5.2. Processes 100, 200 and 300 standardize these raw datasets for further analysis. Thus, to the extent raw data received differs between DSOs, these three processes should be considered **DSO-specific**. In contrast, since all other processes act on standardized data, these can be thought of **DSO-agnostic**, meaning they will need to implemented only once and can be run on data from all DSOs.
- Multiconsult has intentionally spaced process indices apart: This accommodates for future insertions of additional data processing steps all while preserving the integrity of the framework.

2.2 Tying it all together

The interplay and order of execution of the devised data processing steps are illustrated below in Figure 1.

DSO-specific elements of the framework are highlighted with a grey horizontal rectangle. Select processes are part of the Formal Framework but omitted in the application to data as provided by the DSOs in the Reference Group. These are given a dark-grey color and further discussed in Chapter 3.6.7.

Figure 1 - Proposed Formal Framework



3 Application of Proposed Formal Framework to Reference Group

3.1 Norwegian Distribution System Operators

The Norwegian electricity grid system can be divided into the transmission grid, regional grid and local distribution grid. These are largely characterized by differences in voltage, grid type (radial versus meshed) and role in the overall grid system.

Table 2 - Structure of the Norwegian grid (Source NVE-RME, Multiconsult)

Grid layer	Topology	Line length	Typical voltage	Grid level
Transmission grid	Meshed	12,500 km	300 – 420 kV	1
◆ *Substation ("transformatorstasjon")*				
Regional grid / R-Grid	Mostly meshed	19,000 km	33 kV – 132 kV	2
◆ *Substation ("transformatorstasjon"/"innmatingspunkt")*				
High-voltage distribution grid / HVD-grid	Mostly radial	>300,000 km	1 kV – 22kV	3
◆ *Substation ("nettstasjon")*				
Low-voltage distribution grid / LVD-grid			400 V / 230 V	4

While state-owned Statnett is the sole operator of the transmission grid as the country's Transmission System Operator (TSO), more than 110 other grid companies operate the regional and distribution grid networks[4]. These are mandated to operate within precisely defined areas ("områdekonsesjon") awarded by NVE-RME. An overview of these are included as appendix in Chapter 6.1.

3.2 Reference Group

A Reference Group consisting of four DSOs operating in the high-voltage (HVD) and low-voltage distribution (LVD) grids was established by NVE-RME.

Table 3 - DSOs participating in the Reference Group (Source: DSOs, Multiconsult)

DSO	Assigned identifier	Primary area(s) of operation ("områdekonsesjon")	Size		Website
			Customers (#)	*Annual Load (GWh)*	
• Glitre Energi Nett	GE	Multiple municipalities, Viken	~94,000	2,463	glitreenergi-nett.no
• Jæren Everk	JE	Hå kommune, Rogaland	~9 000	~350	jev.no

[4] Source: NVE (for the year 2021).

• Klepp Energi	KE	Klepp kommune, Rogaland	8 983	349	klepp-energi.no
• Mørenett	MN	Multiple municipalities in Sunnmøre and Nordfjord	~65 000	1,756	morenett.no

The Reference Group makes up a diverse set of DSOs with respect to size, geography, and data management practices. Combined, the four DSOs serve over 170,000 customers with a total annual delivered load of about 5 TWh.

In addition, Ringerikskraft Nett was designated for inclusion in the Reference Group as a back-up, although data from this DSO was never solicited during the assignment.

3.3 DSO data organization

Grid companies manage substantial amounts of data on grid infrastructure, customers, and their electricity consumption/production.

These may choose to organize data in varying ways, including opting for outsourcing of data certain data storage, verification, and management needs. Despite this variety of approaches, data is typically organized across various key internal and external repositories listed in Table 4.

Table 4 - Key grid company data repositories, select key vendors (source DSOs, Multiconsult)

Repository/container	Description	Select key vendors/sponsors
• Advanced Metering Systems (“AMS-device”/“smart meters”)	Physical devices installed at end-customer monitoring electricity consumption on an hourly basis	• NURI • Aidon • GE Energy
• Intermediary databases	Depending on set-up, grid companies may choose to clean/validate/restructure high-resolution data in intermediary databases	• N/A
• Customer Information System (CIS)	Typically extensive system that handles handling electricity production and consumption data, customer data and billing.	• CGI • Hansen CX • CISCO
• Network Information System (NIS)	Systems managing data reflecting physical grid components. Include geographical data on transmission lines, substations, metering points.	• Powel • Digpro • Trimble
• Elhub	Centralized, national repository of power market data launched in February 2019 at the decision of NVE-RME.	• Statnett

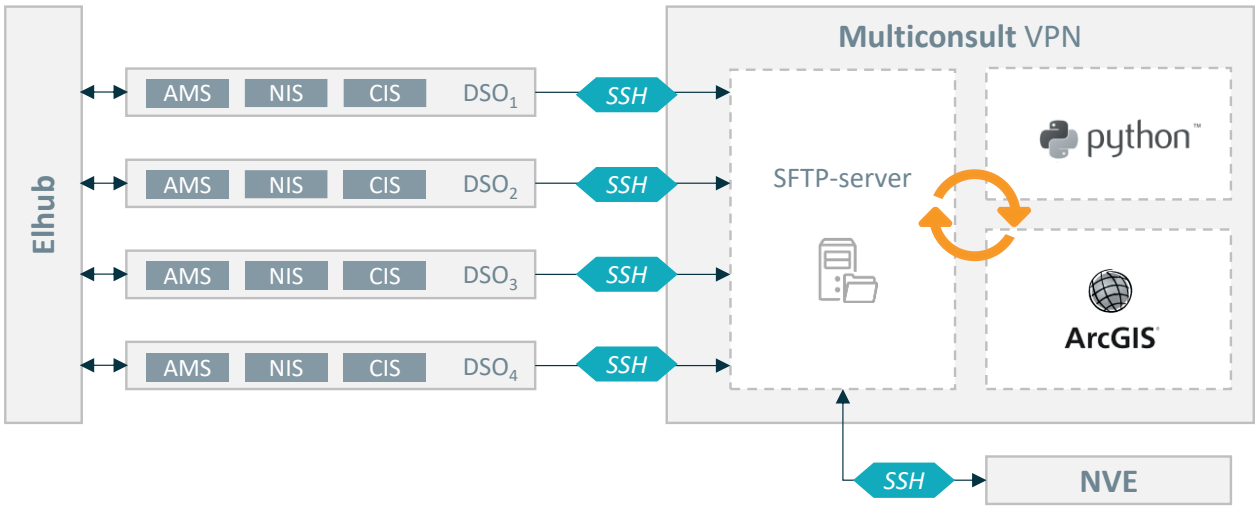
Data solicited from DSOs by Multiconsult in this assignment is drawn largely from the repositories listed above. The role of Elhub will be particularly important in a potential future roll-out of this framework on a national scale. As such, this milestone initiative is discussed in further detail in Chapter 5.

3.4 Infrastructure and software

3.4.1 Project Infrastructure

For the application of the Formal Framework, Multiconsult established an infrastructure for secure transfer and processing of data between DSOs in the Reference Group, NVE-RME and on-premise Multiconsult servers.

Figure 2 - Infrastructure established for application of the Proposed Formal Framework



Measures such as Virtual Private Networks (VPN) and encrypted network communication (through use of the SSH protocol) were taken to ensure data security.

3.4.2 Python

Python is an interpreted scripting language. Its flexibility and quick execution compared to similar languages make it ideal for use in this assignment.

The following table provides an overview of Python libraries and respective version numbers as used in this assignment.

Table 5 - Overview of relevant Python libraries

Library	Version	Documentation	Relevant Processes
• Pandas	1.1.0	https://pandas.pydata.org/docs/	100, 110, 120, 200, 210, 220, 300, 310, 999
• NumPy	1.18.1	https://numpy.org/	100, 200, 300, 999
• GeoPy	1.21.0	https://geopy.readthedocs.io/en/stable/	120

• SciPy	1.5.2	https://docs.scipy.org/doc/scipy/reference/	210, 220
• Re	Python 3.7	https://docs.python.org/3.7/	100
• Datetime			100, 200, 300, 310
• Pathlib			300, 310, 999
• Math			120
• Time			120

These libraries can be installed via the pip Python package manager. To replicate an exact copy of the environment used in this assignment, the version numbers should be considered (please see 6.6 for a description of this).

Scripts were written in the Jupyter notebook format which supports an interactive computational environment, where code execution can be combined with elements such as text, plots, etc.

During processing, data was temporarily stored on an encrypted external hard drive and directly called via Python. Data calls can be made directly to the SFTP server, but this puts pressure on server bandwidths significantly increasing processing time.

3.4.3 ArcGIS Pro

ArcGIS Pro is the leading software for analyzing and processing geographical data. Main benefits of using ArcGIS Pro for this project are that it is powerful, easy to use and widely adopted by practitioners who work with power grids in Norway. ArcGIS Pro also makes it possible to work with different coordinate systems which is an important advantage given the project at hand.

ArcGIS Pro was used for selected data-processing tasks with a geographical component. Specifically, the software was mainly used in the following processes:

- Allocating X- and Y-coordinates to metering points where such were missing (Process 120)
- Allocation of metering points to nearest substation (Process 210 / “Allocation 1“)
- Reallocation of metering points to alternative substations in case of GDPR violations (Process 220 / “Allocation II”)

In addition, the software was used to convert between different geographical coordinate systems and for comparing the use of Euclidian distance instead of technical grid distance in the two rounds of allocations.

ArcGIS Pro is a licensed software priced depending on the number of users. For organizations that do not already use this software, cost may be considered prohibiting. However, for organizations already in possession of user licenses, using ArcGIS Pro does not incur extra cost. Multiconsult used ArcGIS Pro version 2.6 in this assignment.

3.5 Raw data provided by the DSOs

3.5.1 Requested datasets

A Data Request Document was elaborated together with NVE-RME and the third-party consultant to ensure a single and well-defined data request could be issued to the Reference Group. This document is attached as appendix in Chapter 6.4.

Three datasets were requested (key data fields in parenthesis):

- Metering points (including customer type and location
- Substations (including X- and Y-coordinates, grid-level)
- Power consumption/production (including timestamps, metering values, direction)

These three datasets constitute the raw data feeding into our data standardization processes 100, 200 and 300, respectively, as defined in the Formal Framework.

DSOs were permitted to deviate from the data request, but only to the extent defined (idealized) datasets could be constructed based on received datasets. This concession allowed Multiconsult to document variations in data available to DSOs across systems available to them (as discussed in Chapter 3.3), and to underscore inherent challenges in dealing with large power market data.

The period 1 March 2019 - 29 February 2020 was chosen as the Reference Period for which solicited data applies. This period succeeds the launch of Elhub in February 2019 and was chosen in agreement with NVE-RME and the Reference Group.

3.5.2 Received datasets

The following table summarizes data received from the DSOs in response out Data Request Document. Files that were submitted but were not relevant to this assignment are not included in this overview.

Table 6 - Overview of submitted datasets by DSOs

DSO	Name	Format	# Files	Approx. Size (MB)	Related dataset
Glitre Energi	mp	XLSX	1	11	100 – Metering Metadata
	Uttrekk-Netbas-Nettstasjoner-07-08-2020 (003)		1	<1	200 – Substation Data
	Effektdistanse_Innmatingspunkt		1	<1	
	Meterreading_{month&year}	TXT	12	39,000	300 – Meter readings
	Meterreadings_all_prod		1	47	
Jæren Everk	Kunder med NS og koordinat_ny	CSV	1	<1	100 – Metering Metadata
	NS med koordinat		1	<1	200 – Substation Data
	Forbruk_{month&year}	SDV	12	3,639	300 – Meter readings
	Kombi_Forbruk & Kombi_Produksjon		2	17	

Klepp Energi	Metering Point metadata	CSV	1	1	100 – Metering Metadata
	Substation data		1	<1	200 – Substation Data
	Production and consumption data {date} – {date}		2	3,891	300 – Meter readings
Mørenett	AB-data	XLSX	1	5	100 – Metering Metadata
	NS-data		1	<1	200 – Substation Data
	TS-data		1	<1	
	Export_{date}_{date}	DSV	4	3,895	300 – Meter readings

Received files carried one of 6 different extensions. However, for most practical purposes, it is often more relevant to distinguish between plaint text formats and proprietary formats (i.e. files that require special/proprietary software to be opened).

Except for XLSX which is developed by Microsoft, all other encountered formats (TXT, CSV, SDV and DSV) were such plain-text formats. These are highly portable and can be processed across systems.

However, particular care should be taken upon dealing with such plain text files since their encoding can be different. This could impact the representation of non-standard Latin characters such as Norwegian vowels Æ, Ø and Å.

3.6 Data processing

This chapter describes the implementation of the proposed Formal Framework introduced in 2.2 in context of our Reference Group. A more detailed and technical log of data processing steps is provided as an appendix in chapter 6.5.

Please note that processes that address mobile meters are not discussed here since they were not applied in the context of the Reference Group (please see Chapter 3.6.7 for a discussion on the reason for this omission). Processes discussed here is thus limited to 100, 200, 300, 110, 120, 210, 220, 310 and 999.

Despite the issuance of an elaborate Data Needs Document, received datasets revealed a range of differences, challenges, and potential pitfalls for processing. These provide valuable insight to overall data quality, differences in data organization, and challenges for a potential national operationalization of the framework in the future.

3.6.1 Standardization of metering point metadata (Process 100)

Received metering point metadata largely reflected instructions given in the Data Needs Document.

But mobile meters and their historical deployment during the Reference Year represented a notable exception. In fact, only one DSO provided such data according to specification. Remaining 3 either entirely omitted or only indicatively included such data in descriptive text fields.

The standardization process of this dataset included the data harmonization, reclassification of customer groups, removal of mobile meters, standardization of grid level, and removal of duplicate entries.

Table 7 - Standardization of metering point metadata

Characteristic		GE	JE	KE	MN	100_OUT
Meta	• **File format**	.xlsx	.csv	.csv	.xlsx	.csv
	• **Delimiter**	Semicolon	Semicolon	Semicolon	Semicolon	Comma
	• **File Size (MB)**	11.4	0.6	1.3	5.2	N/A
	• **Duplicates**	None	876	159	None	None
	• **Total meters**	94,465	8,884	8,997	64,633	176,979 (for all DSOs)
GIS	• **Coordinates missing**	1801 (1795 with full address data)	None	1	170 (160 with full address data)	None
	• **Coordinate separator**	Dot	Comma	Dot	Dot	Dot
	• **CGS**	EUREF89 UTM zone 33	UTM/EUREF 89 sone 32 + NN2000	WGS 1984	EUREF89 UTM sone 32	WGS 1984
	• **Comments geographical data**	-	-	X & Y coordinates swapped	-	N/A
Mobile meters	• **Mobile meter data**	268 “Kasse”	No	142	250 "Anleggs-kasse”	Discarded
	• **Mobile meter period**	No	No	Yes	No	N/A
Other	• **Customer Group**	Codes 1-37	Codes 1-37	Codes 1-37	Codes 1-37	0 (private), 1 (non-private)
	• **Grid Level**	3 or 4	3 or 4	3 or 4	In voltage	3 or 4

Key challenges identified during standardization of the metering point metadata include:

- **Duplicate observations**: Duplicate metering point IDs were observed for several DSOs. Although these observations shared identical geographical coordinates, customer groups varied. We were thus dealing with partial duplicate as opposed to full duplicate observations.

  Multiconsult suspects these duplicates originated from changes of customers behind respective metering points. For the purpose of standardization, duplicates were removed, and private customer was assumed. This in order to avoid inadvertent violations of GDPR in subsequent rounds of allocation metering point to substations.

- **Erroneous GIS coordinates:** Select X and Y coordinates were found to have been reversed in the provided dataset. Such erroneous coordinates were identified through general sanity

checks using GIS software. Translating DSO license areas ("områdekonsesjon") into polygons should flag instances of this problem in later implementations of the Framework.

- **Lacking GIS coordinates:** For select DSOs, a considerable share of meters had no coordinate information provided. This emphasizes the importance of a high-quality geolocator, and need for complete address data
- **Lacking mobile meter data:** In the data provided by DSOs, there was insufficient information on mobile meters which prohibited accounting for movement of mobile meters during the Period of Interest.

3.6.2 Standardization of substation data (Process 200)

Received substation data largely reflected instructions given in the Data Needs Document needs document. For select DSOs, grid level 3 substations ("nettstasjon") and grid level 2 substations ("transformatorstasjon") were provided in separate files.

The standardization process mainly encompassed harmonization of data formats, removal of duplicates, conversion of coordinate systems, merging of substation data sets, and removal of substations without assigned coordinate systems.

Table 8 - Standardization of substation data

Category		GE	JE	MN	KE	200_OUT
Meta	• **File format**	XLSX	CSV	XLSX	CXV	CSV
	• **Delimiter**	Semicolon	Semicolon	Semicolon	Semicolon	Comma
	• **File size (kB)**	175	12	14	184	N/A
	• **Duplicates**	1	0	0	10	No duplicates
	• **Total substations**	3,570	408	2,635	318	Total 6,931
GIS	• **Coordinates missing**	2	0	17	0	N/A
	• **Coordinates decimal separator**	Dot	Comma	Dot	Dot	Dot
	• **CGS**	EUREF89 UTM zone 32	EUREF89 UTM sone 32 + NN2000	EUREF89 UTM sone 32	WGS 1984	WGS 1984
	• **Other GIS remarks**	-	-	-	X & Y coordinates were swapped	N/A
Other	• **Grid level**	Separate datasets	As Grid Level 2/3	Separate datasets	As Grid Level 2/3	As Grid Level 2/3
	• **Comment**	-	One spennings-hever removed.	-	Includes one 16-digit ID similar to meter IDs	-

Key challenges encountered in the standardization of substation data include:

- **Erroneous GIS coordinates:** Select pairs of X and Y coordinates seemed to have been reversed in the provided dataset. In some other cases, coordinates values contained zero values instead of null values which may be misleading.

  Such erroneous coordinates were identified through general sanity checks using GIS software. Translating DSO license areas ("områdekonsesjon") into polygons should flag instances of this problem in subsequent implementations of the Framework.

- **Inactive and/or decommissioned substations:** After inquiring about values for a specific substation, Multiconsult was informed the substation in question was decommissioned. This underscores the challenges of evolving/extending grid topology for which a validation mechanism should be elaborated[5].

3.6.3 Standardization of metering data (Process 300)

Metering data, i.e. electricity consumption and production data, was provided in a range of different formats. This prompted Multiconsult to issue additional questions and instructions to the DSOs to ensure senseful standardization this data.

The standardization of metering data was related mainly to harmonizing formatting, adjustment for Daylight savings time (DST), time referencing, and netting of consumption and production data.

Table 9 - Standardization of metering data

Category		GE	JE	KE	MN	300_OUT
Meta	• **File format**	TXT	SDV	CSV	DSV	CSV
	• **Delimiter**	Semicolon	Semicolon	Semicolon	Semicolon	Comma
	• **File Size (GB)**	36.3	3.5	3.7	3.7	N/A
	• **Production**	Separate file	Separate File for plusskunder	Combined	Combined	Netted
	• **Observations (post standardization)**	822,210,478	78,632,494	78,619,997	581,131,960	1,560,594,929 (Total across DSOs)
Time	• **Timestamp**	DD.MM.YYYY HH.MM.SS	DD.MM.YYYY HH.MM.SS	DD.MM.YYYY HH:MM	DD.MM.YYYY (hour via column)	DD.MM.YYYY HH:MM:SS
	• **Reference**[6]	Backward	Forward	Backward	N/A	Backward
	• **Observance of DST**	Yes	No	Yes	No	No

[5] Multiconsult has not mapped the extent of inactive/decommissioned substations among DSOs in the Reference Group. Explicit mentions of this could be done in future data requests to limit the extent of this problem.

[6] *Timestamp referencing* refers to whether the hour associated with the timestamp is referring to the preceding or subsequent hour.

Meter data	• **Unit**	kWh	kWh	kWh	kWh	Wh
	• **Consumption Sign**	+	+	+	+ ("Direction" –)	+ (netted)
	• **Production Sign**	+	+	+	+ ("Direction" +)	- (netted)
	• **Decimal Separator**	Comma	Dot	Comma	Comma	Dot
	• **Thousand Separator**	None	None	Space	None	None
Other	• **Comments**	-	18 outlier MeteringPoint-IDs removed	1 faulty meter removed	-	-

Key challenges related to the standardization of metering data include:

- **Data volumes**: The substantial volumes of metering data, particularly for large DSOs, required "tranching" of the standardized output dataset 300_OUT by month to ensure efficient processing in subsequent steps.
- **Observance of Daylight savings time**: DSOs accounted for daylight savings time in several different ways. While one DSO provided metering data in UTC+1 (i.e. Norwegian time with no DST), other DSOs approached this in different ways, including duplicate timestamping and aggregating of metering values for hours when time is advanced/turned back[7].
- **Timestamp reference**: Timestamp data needed to be interpreted in different ways. Specifically, whether a timestamp such as "01.01.2020 03:00:00" refers to the period leading up or instead following 03 AM needed to be clarified for each DSO.
- **Outlier meter readings**: Multiconsult attempted identify erroneous metering data by performing standard deviation tests. This effort revealed some outliers for select DSOs that would have significantly impacted final datasets if left unnoticed. Multiconsult has considered more sophisticated data outlier detection to fall outside the core mandate of this assignment.
- **Profiled metering values** ("profilmåling"): One DSO provided metering values where missing data were replaced by "profiled" metering values. This may lead to select negative consumption values. Since these values are accurate in the context of annual consumption, these values were retained despite introducing (negligible) distortions in hourly consumption at the substation level.

3.6.4 Geocoding of addresses (Process 120)

Geocoding was performed in ArcGIS Pro using GeoLokasjon2, a geocoding service provided by Geodata.

[7] During the Reference Year, this took place on 10.03.2019 02:00 and 03.11.2019 03:00.

The tool selected meters with missing X and Y coordinates and assigned a location based on the address field provided by the DSO. This enabled successful geocoding of between 90%-100% of metering points for which coordinates were missing

However, despite the tool can be calibrated to accommodate for common errors related to punctuation and letters, certain metering points could not be geocoded due to low quality address strings.

3.6.5 Approach to Allocation I and II (Processes 210 and 220)

Allocation I refers to the allocation of metering points to the nearest substation using distance metric of choice (Euclidian).

Allocation II refers to the reallocating, using the same distance metric, of meters for (intermediary) GDPR non-compliant substations. In understanding with NVE-RME, a GDPR compliant substation is defined to be a substation to which at least one non-private customer or 3 or more private customers are allocated. It follows from this definition that a substation to which one non-private and one private meter are allocated is considered compliant.

In terms of implementing these two allocation rounds, each metering point was assigned to the nearest substation. Each substation was assigned an attribute based on whether or GDPR compliance criteria were met. Non-compliant substations were filtered out of the dataset and the model rerun. With the datasets at hand, it was sufficient to perform this process three times to produce fully GPDR-compliant datasets.

The output data from the analysis consisted of a Shape file[8] containing all meters with X and Y coordinates, meter ID and an allocated substation ID. This was then exported to csv.

Please note that the allocation described above was performed in two different instances in order to account for different grid levels: First connecting grid level 4 meters to grid level 3 substations, and secondly connecting grid level 3 meters to level 2 substations.

3.6.6 Preparation of final dataset (Process 999)

The purpose of process 999 is to create the final output dataset. This datasets aggregates consumption at each substation for each hour during the Reference Year.

Table 10 - Schematic illustration of final dataset 999_OUT

	01.01.2020 01:00:00	**01.01.2020 02:00:00**	**01.01.2020 03:00:00**	**…**
Substation_ID_1	Wh	Wh	Wh	
Substation_ID_2	Wh	Wh	Wh	
…				

[8] A geospatial vector data format for Geographic Information Systems (GIS)

The final dataset is created in two main steps. First, aggregated metering values are grouped by substations based on descibred allocation rounds. Second, this dataset is pivoted to take the form illustated in Table 10, as requested by the consultant that will continue the analysis.

3.6.7 Adaptations of Proposed Formal Framework

Despite its inclusion in the data needs document, little or no data on mobile meters and their historical deployment was provided by DSOs. Multiconsult therefore slightly adapted the application of the Formal Framework in its application to the Reference Group.

Accordingly, processes 110 and 310 were not implemented. Therefore, output dataset 300_OUT was used to generate the final dataset, as no mobile meters had to be replaced.

4 Results and Findings

4.1 Output dataset characteristics

Multiconsult has discarded some metering data in compilation of the final 999_OUT datasets. Each such instance is explained by one of five reasons related to either lack of data, or erroneous or insufficient data. The extent of such omissions for each of the DSOs in the Reference Groups is discussed in Table 12.

Table 11 - Reasons for discarding metering data

Reason	Type
1. Metering point not in metadata set (for valid or invalid reason)	Lack of data
2. Faulty meters - as confirmed by the DSO	Erroneous data
3. Outliers in dataset	
4. Assumed or confirmed mobile meters (“anleggskasser”)	Insufficient data
5. Insufficient/erroneous GIS for metering point	

Applying omissions describe above, the final 999_OUT datasets comprise aggregated metering values of ~4.7 TWh. This figure for each DSO reflects total electricity supplied by DSOs in their annual reporting as recapped in Table 3.

Table 12 - Characteristics of 999_OUT and omissions of metering values

DSO	Glitre Energi	Klepp Energi	Jæren Everk	Mørenett
Net MWh in provided dataset (no values excluded)	2 250 579	332 315	626 220	5 692 032
Diff in net MWh (due to Reason 2 and 3)	-	- 12 698	289 940	-

Net MWh in standardized dataset (after Removal of 2 and 3)	2 250 579	345 013	336 279	5 692 032
Difference in net MWh (due to Reason 1, 4 and 5)	3 552	1 495	5 092	3 954 729
Net MWh in provided in 999_OUT	2 247 027	343 518	331 188	1 737 303
Percent omitted (due to reason 1, 4 and 5)	0.16%	0.43%	1.51%	69.48%[9]

4.2 Precision of Allocation I

NVE-RME has expressed preference for the use of Euclidian distance in allocation of metering points to substations. Many valid arguments underpin this choice, including ease of data maintenance, simplicity of computation, and suitability for the evaluating the power distance of DSOs (i.e.. more sophisticated approaches are not strictly called for).

Several DSOs provided a data attribute asserting the substation to which each metering point is connected in the real grid. This allowed Multiconsult to evaluate the precision of applying Euclidian distance compared to allocation based on physical grid topology.

As a benchmark, based on an analysis of such data provided by Klepp Energi, Multiconsult found that the share of metering points allocated to substations deviating from real grid topology amounted to ~30%.

4.3 Alternative approaches to Allocation II

The problem of reallocating metering points to alternative substations in violations of GDPR can be solved in a wide variety of ways.

Multiconsult believes allocation algorithms should be evaluated based on the following criteria:

1. **Must yield valid final state**: In case of iterative algorithms, the algorithm should ensure all metering points are allocated to a substation and that no one substation violates GDPR criteria
2. **Algorithmically efficient**: Lower usage requirements of resources including storage and processing power should be favored; lower algorithmic complexity[10] ensures robustness faced with large and/or increasing data size
3. **Minimizes reallocation of metering points:** This criterion ensures grid topology characteristics to the highest degree possible. It also ensures undue preference is not given to trivial solutions such as allocating all metering points to one single substation.

[9] Whereas omitted consumption was small/negligible for most DSOs, omissions in datasets provided by Mørenett raised initial concerns. After investigation, we revealed 2,980 meters in the metering dataset but absent in metadata. These together made up substantial consumption: In fact, we omitted nearly 70% of consumption found in the raw consumption file due to this reason. Multiconsult suspects Mørenett's raw consumption file contained also aggregated values drawn from substations. We chose not to pursue further clarifications on this toward Mørenett since i) the 1.7 GWh represented by metering points in the metadata file reflects the company's delivered power volume, and ii) Our base assumption is DSO provided complete metering point metadata.

[10] Algorithmic complexity can be compared using so-called Big O-notation – the details of such analysis is considered beyond the scope of this assignment.

On request from NVE-RME, and for completeness, Multiconsult is providing an alternative algorithm for achieving Allocation II, in addition to that provided in chapter 3.6.5 and implemented in ArcGIS.

This conceptually simple algorithm first classifies substations into either “non-compliant” or “compliant” based on the definition of compliant substations. For non-compliant substations, the closest compliant substation is identified. Meters allocated to a non-compliant substation are reallocated to the newly identified compliant substation, yielding a valid state of substations that are all GDPR compliant. This algorithm is implemented in Python and provided as appendix in Chapter 6.7.

A weakness of both proposed algorithms discussed in this report is the inability to “merge” two non-compliant substations into to a compliant substation. However, both algorithms can be refined to capture this and other sorts of dynamic in order to minimize the number of metering point allocations.

4.4 Performance analysis

The proposed Formal Framework could form the backbone of an operationalization on a national level. In such scenario, sensitivity to processing time and storage requirements provide valuable context to choices of software infrastructure, automation, and organization.

To this end, Multiconsult developed a Test harness consisting of 3 test cases to provide a high-level indication of processing speed and storage requirements.

4.4.1 Test harness configuration

Three test cases were defined for up to 250,000 customers. For comparison, nearly all DSOs in Norway have less than 100,000 customers[11], with Elvia being the largest DSO in the country with some 900,000 residential customers[12].

Apart from differing in substations and metering points, configurations were kept similar across test cases.

Table 13 - Test harness

Test cases	**Small**	**Medium**	**Large**
• Metering points (#)	10,000	50,000	250,000
• Substations (#)	1,000	5,000	25,000
Configuration			
• Metering points operating at grid level 3 versus 4	10% / 90%		
• Substations at grid level 2 versus 3	10% / 90%		
• Private customers	90%		
• Mobile metering points	3%		
• Share of metering points with missing coordinates	2%		
Test execution engine			

[11] Source: https://www.nve.no/norwegian-energy-regulatory-authority/
[12] Source: https://www.elvia.no/hva-er-elvia/vart-stromnett. Formed by merger of Hafslund Nett and Eidsiva Nett at beginning of 2020.

• RAM	32 GB
• Processor	Intel Core i7-8650U (8 core)
• Operating System	Windows 10
• Storage	External hard drive (USB 3.0)

4.4.2 Test harness results

Execution of our Test harness yields computational and memory performance as indicated in the table below.

As expected, the number of customers significantly increases running time requirements which vary between 8 and 1,412 minutes (~23 hours) depending on the test case. Further, processes 310 and 999 significantly outweigh preceding processes in terms of both storage and processing requirements.

The configuration of the Test harness, its execution engine and implementation all impact the results of the performance tests. Results should therefore be interpreted indicatively.

Table 14 - Test harness results

Process	**Small**		**Medium**		**Large**	
	Running Time (s)	**Dataset Size** (MB)	**Running Time** (s)	**Dataset Size** (MB)	**Running Time** (s)	**Dataset Size** (MB)
• **110**	2.47	2,781.06	12.85	14,292.66	69.00	71,417.64
• **120**	232.45 (28.45)	2,781.06	1072.89 (120.89)	14,292.67	5,659.94 (623.94)	71,417.74
• **210**	8.52	2,780.68	121.62	14,290.61	2,509.61	71,407.86
• **220**	0.69	2,780.68	1.72	14,290.61	28.72	71,407.86
• **310**	358.68	2,785.12	4192.32	14,316.65	78,489.24	71,527.86
• **999**	104.88	33.48	664.68	167.88	3,054.36	818.4
Total	**707.69** (11 min)	33.48	**6,066.08** (101 min)	167.88	**89,811** (1,497 min)	818.4
Total (without geolocator)	**503.69** (8 min)		**5,114.08** (85 min)		**84,774** (1,412 min)	
Methodological remarks	Geolocation lookups in Process 120 spaced 1000 ms. apart. Python employed for these. Performance of Processes 310 and 999 estimated on subset of test data.					

.

Figure 3 - Aggregate running time of processes

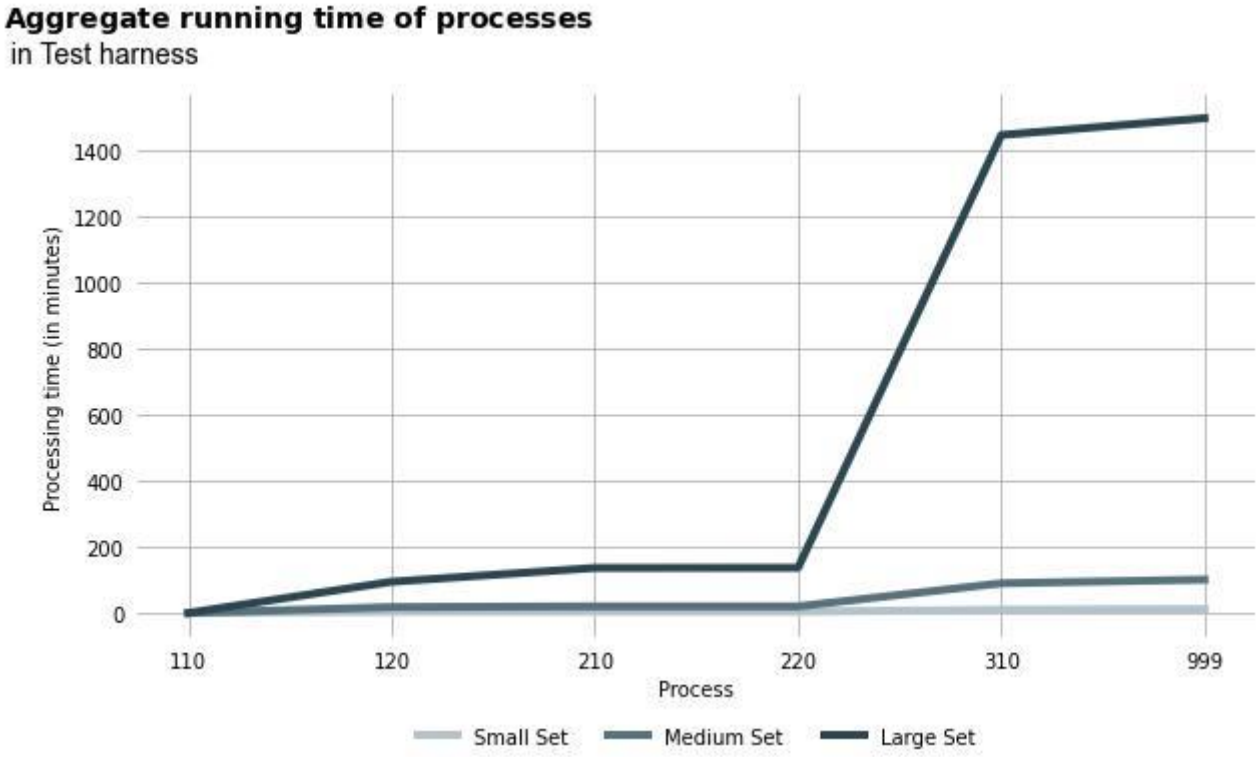
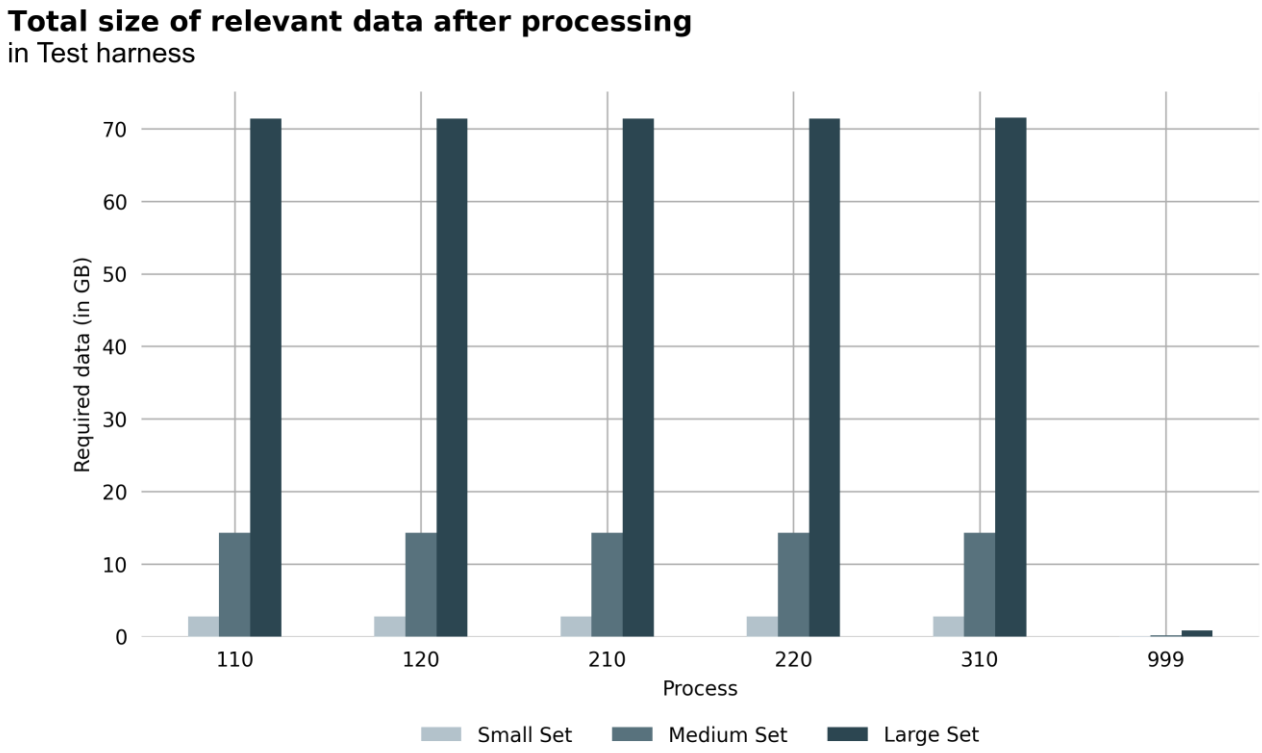


Figure 4 - Total size of relevant data after processing



Processing time. The process for creating virtual meter IDs (110) for mobile meters is very efficient. Even for large DSOs there is no concern for fast implementation. The Python implementation of geolocating addresses of missing meter coordinates (120) does add considerably to the overall processing time.[13] This is largely due to the geolocation service (Nominatim by OpenStreetMap), which requires a *sleep* time of circa one second after each address location.

The allocation of meters to substations and second allocation to account for GDPR compliance of meters contributes significantly to the overall processing time as the number of meters increases – ranging from 8 seconds for 10,000 meters to 42 minutes for 250,000 meters in the test bench environment. The reallocation of load from non-compliant substations to compliant substations is almost negligible, taking less than a minute for all sample sizes. Particularly the replacement of meter IDs with virtual meter IDs is processing intense (310). Particularly, this relates to multiple iterations over the entire consumption dataset containing more than two billion observations in the large dataset. A more efficient implementation may, however, reduce the processing time significantly. The creation of the final dataset (999), takes around 10 minutes for the large set.

5 Future roll-out of Proposed Formal Framework

Multiconsult would like to discuss key aspects that should be carefully considered in such scenario.

Table 15 - Aspect for considerations at nationwide roll-out

Category	Aspect	Remarks
Infrastructural	• **Data flow and general infrastructure**	Any implementation of the Formal Framework requires support for storage and processing of large data files, and GIS capabilities. These core elements may be set up and interact in a vast number of ways. Considerations that need to be made are manifold and include procurement model, software platforms, data security and more. Data management software platforms such as FME and/or AWS may be considered for use. This aspect is fundamental to a roll-out of the Proposed Framework nation-wide and should be carefully considered by domain experts in areas including IT security, networking, and data processing.
	• **Procurement model**	One aspect of the general infrastructure relates to the adopted procurement model. Questions that should be addressed include the extent to which data processing and storage are sourced externally (for example through cloud computing services) or instead are provided internally (such as on-premise). Various hybrid approaches may naturally be optimal. Choices may depend not only on cost, efficiency, and security, but also on laws governing processing of sensitive data on international data servers may also apply.
	• **Elhub**	The possibility of extracting suitable and high-quality data from Elhub is a premise for the roll-out out of the Formal Framework on a national scale. A report by Statistics Norway (SSB) from July 2020 revealed weaknesses in data extracted from this centralized power data repository. These, and overall availability of performing extractions from Elhub, need to be carefully understood in preparation of a nationwide operationalization of the framework.

[13] The geolocation of the final dataset was performed in ArcGIS using a more accurate, Norwegian geolocation service. Therefore, the processing time does not correspond to the processing time in ArcGIS during implementation.

	• **APIs**	An Application Programming Interfaces (APIs) is a computing interface governing the interaction between different parts of a software system. APIs can contribute to data consistency, reduced storage requirements and a distribution of computing efforts if employed cleverly. APIs form a key concept which could find application in all parts of a full-scale implementation of the Proposed Framework, including at NVE, Elhub, DSOs and intermediaries.
	• **Data security & GDPR**	The computation of the power distance variable for all Norwegian DSOs involve the management of vast amounts of potentially GDPR-sensitive data – the extent of this may depend on adopted infrastructure. Nevertheless, a holistic understanding of GPDR sensitive elements of the system, as well as a vetted approach to secure network communication and encryption protocols, need to be elaborated ahead of launch.
	• **Automation & Frequency**	Full automation of data combination tasks pose some inherent threats: It often delinks processes and the personnel in charge of them and is often sensible to (even small) changes in the external environment (such as file naming or data formats). Given the only annual computation of power distance, one can argue part of the data combination efforts need not be fully automated, possibly limiting the extent of such challenges.
Technical	• **Data Quality**	Quality data and subsequent correct calculation of power distance is fundamental. Yet data received from DSOs in the Reference Group as part of this assignment demonstrate many sources of poor data quality. Processes that ensure data quality requirements are met should thus be established.
	• **Data Volume**	Challenges arising from large data volumes are often inherent in managing power data. They are typically manifested in lower performance related to computing time and storage requirements. But combining data for use in calculation of power distance lends itself well to parallel computing. This means raw datasets may be processed as separate smaller pieces and large data processing frameworks such as Hadoop may be considered.
	• **Data Standardization & Validation**	High-quality data may still need standardization and validation. This is particularly true when data originates from a large number of disparate sources which 130+ Norwegian DSOs would represent in a national roll-out. Strict and detailed data instructions and validation processes would need to be established. Most DSOs should have gained accustomed to such standardization efforts through preparations toward the launch of Elhub in 2019.
	• **Data Lags**	While Elhub provides an infrastructure for extraction of real production/consumption values, datasets related to customers and substations face certain distinct challenges. For one, such data typically experiences inherent lags before changes to customers (such as churn) or substation (such as decommissioning) are reflected in relevant NIS/CIS systems. NVE-RME needs likely to address the extent to which such “data lags” be permissible in the context of calculating power distance.
	• **Mobile Meters**	Mobile meters may make up only a small minority of metering points. But by nature, since often deployed for high-consumption applications such as construction sites, consumption may be significant. It is thus important to devise a robust framework for dealing with such meters and historical deployments of the same. Indeed, discussions with the Reference Group as part of this assignment reveals that such data is not routinely stored in a

		standardized way which could threaten the integrity of the power distance variable as a new output variable in the DEA-model for DSOs.
	• **GIS**	A similar lack of standardization is noticeable when dealing with geographical data which is needed both for metering points and substations. Specifically, a number of different geographic systems may be chosen. Further, the use of multiple such geographic systems may differ not only between DSOs but also between datasets sought from individual DSOs.
Procedural	• **Neutrality**	DEA-models need to be calibrated carefully due to inherent sensitivity of its output variables. Yet Norwegian DSOs represent a very heterogeneous set of players as they differ in respects not limited to size, number and type of customers, grid infrastructure, topology, landscape. Power distance may more accurately capture the task of DSOs ("delivering power") instead of merely their efforts to do the same. Nevertheless, Multiconsult wishes to emphasize that DSO bias/methodological favoring may in the process of combining datasets for calculation of power distance, not least through the distance metric and allocation algorithms. Such bias should be carefully understood before adopting a final implementation of a data combination framework.
	• **Ownership**	Key to the reform of the current DEA-models is the definition and elaboration of the power distance variable. The process in achieving this is well documented through publication of reports listed in Table 1. This report documents a range of potential pitfalls in harvesting, standardizing, and combining data such that the calculation of power distance by DSO be permissible. Multiconsult believes similar transparency on data combination efforts be beneficial in ensuring reform ownership not only within the energy regulator but also in each the 130+ Norwegian DSOs affected by the reform.

In a future roll-out of the Proposed Formal Framework in which metering data is drawn from a centralized data repository such as Elhub, all DSO-specific process will become DSO-agnostic.

5.1 Relevant data management frameworks, cloud systems

A future organization need to comprise the following fundamental components as a minimum: i) Storage, ii) Processing (either local or in cloud), and iii) GIS-capabilities.

Multiconsult would like to discuss select data management frameworks that could be relevant in a future roll-out of our proposal formal network in the future. This is not meant to represent an exhaustive overview of relevant data management frameworks. These could also be relevant in combination with one another.

5.1.1 FME

Feature Manipulation Engine (FME) is a powerful management platform developed by SAFE Software for designing, interconnecting, and operating data flows.

FME is available under license but is often used by large public entities that deal with complex data flows. Multiconsult believes this software could be relevant in the context of establishing a future data framework for calculating power distance.

Specifically, FME provides the following key capabilities[14]:

- **Data Integration:** Allowing conversion and transformation of data to form uniform views of collected information
- **Data transformation:** Allowing alterations of structure, content and data characteristics in order to adopt to specific needs
- **Support for spatial data:** Key GIS capabilities that make the framework particularly relevant to deal with geocoded data on metering points and substations
- **Integrates well across applications:** The software allows for connecting to various applications which allow for designing of adapted and flexible data flows
- **Conversion capabilities and verification:** Support for data conversion tasks and functionality for verifying data and ensuring consistent quality

FME can be integrated most leading data technology providers, such as AWS, Azure, Oracle and SAP.

5.1.2 AWS

One possible way of implementing the data processing on a national scale is via a cloud service that provide the necessary storage, processing power and security. There are a range of cloud service available, such as Azure and Amazon Web Services (AWS) that cover the requirements.

As an example, AWS offers a comprehensive set of services dealing with this situation to obtain, store, and analyze data of almost all types of scales and formats. For data storage, AWS offers alternatives for all types of formats. For structured data – which we are mostly dealing with in this assignment (if standardized) - a traditional SQL-based relational database is best suited. This is available in different implementations at AWS – such as Aurora and RDS. Alternatively, it is possible to establish a direct connection to Elhub, given that appropriate APIs or other gateways are in place. Elhub could then be queried and cleaning, formatting & validation can be performed (e.g. via AWS Glue).

For processing, various cloud computing services (e.g. EC2) are available. Computing instances can run scripts of all major programming language – and be employed independent of the storage location of the data. This can ensure that sufficient memory is available and speed up the overall processing. Intermediary and final outputs can be saved to AWS's storage solutions and made available for further processing and calculation of power distance.

5.1.3 Other systems

The list of highlighted data management frameworks is not exhaustive. A range of other data management platforms and cloud computing services could be considered, such as Azure and Heruku owned by Microsoft and Salesforce, respectively.

[14] Source: Multiconsult, SAFE Software

6 Appendix

6.1 Norwegian regional and distribution grid companies

6.2 Data set description

6.3 Process description

6.4 Data needs document

6.5 Data standardization log

6.6 Python environment initialization

6.7 Python Code

6.8 ArcGIS Model (screenshot)

6.9 Description of structural adjustments to final dataset (solicited by Thema)

6.10 Processing of meters operating at 1 kV voltage

Nettselskap Områderkonsejoner

1 : 1 000 000

Områderkonsejoner

Koordinatsystem: UTM, sone 33. Datum er Euref89 (WGS 84)
Grunnlagsdata: Statens kartverk, N1000
Temainformasjon: Forsyningsområde
Oppdatert: 1. mars 2020

Norges vassdrags- og energidirektorat (NVE)
Hydrologisk avdeling (H)
Seksjon for Geoinformasjon (HG)
Postboks 5091, Majorstua 0301 Oslo
Tlf: +47 22959595
e-post: gis@nve.no

NVE

Kartet viser områdekonsesjons eiere lik eller større enn 4 km². På NVE's internettside; atlas.nve.no er alle navn på områdekonsesjons eiere vist når man zoomer inn til større målestokk

Multiconsult

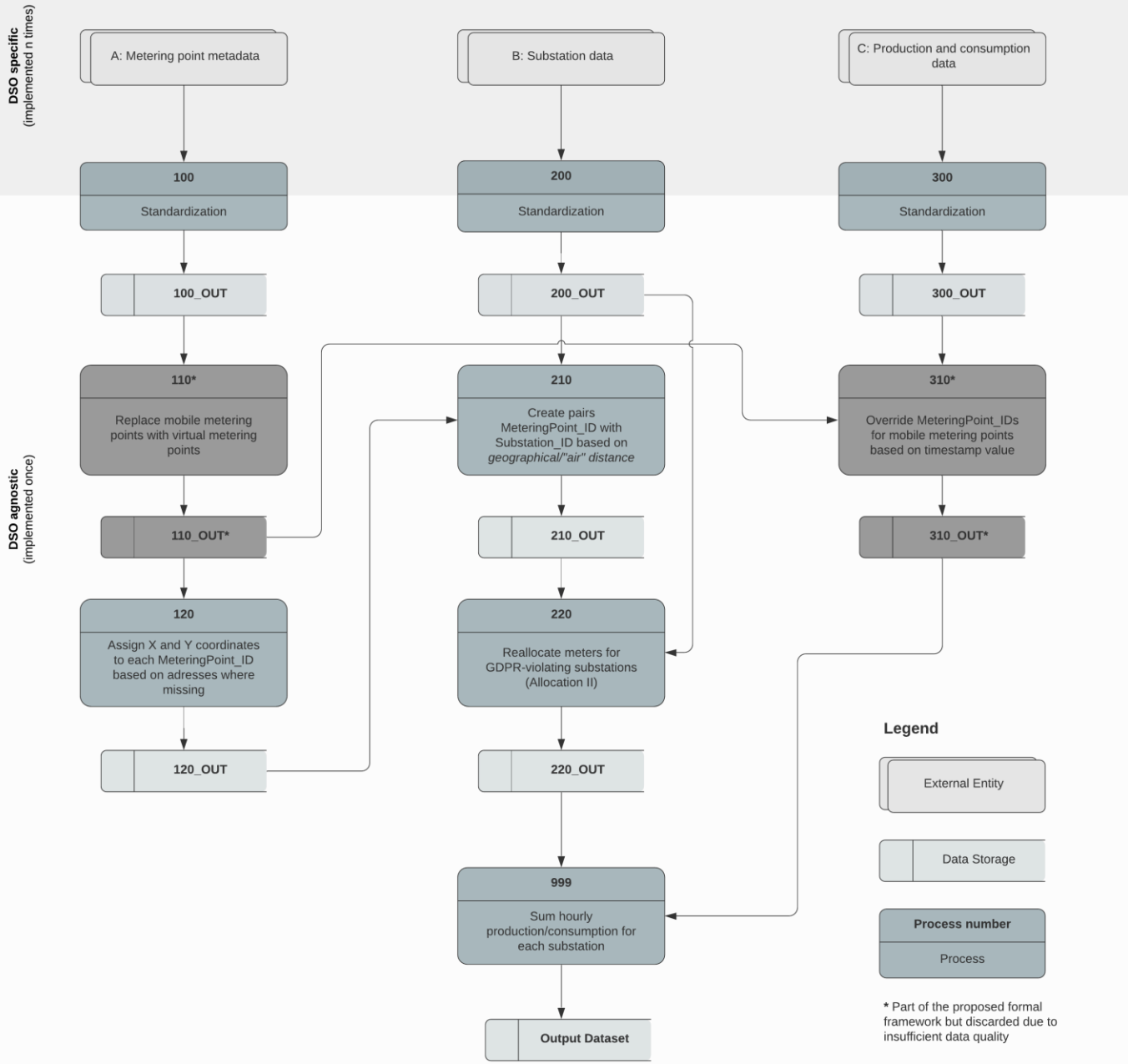
Data Documentation Document (Appendix 2 and 3)

NVE Assignment

15.10.2020

Contents

Formal Framework (schematic overview)



DSO Data

Participating DSOs indexed as per the following:

DSO ID	DSO
MN	Mørenett
JE	Jæren Everk
GE	Glitre Energi Nett
KE	Kleppe Energi

Folder Structure

- **/input** all data provided by DSOs
- **/data** all data processing & CSV files
- **processing files** all scripts on the highest level. Data is called via the */input* & */data* directory

Contents

Processes Overview

Process ID	Description	Implementation	Comment
100	Standardize Metering Point Data	Python	
110	Replace mobile meter ID with virtual meter ID	Python	Not implemented
120	Assign X and Y coordinates to each Meter ID based on address	ArcGIS Pro	Python draft
200	Standardize Substation Data	Python	
210	Create pairs of Meter ID with associated Substation ID	ArcGIS Pro	Python draft
220	Reallocate meters that violate GDPR	ArcGIS Pro	Python draft
300	Standardize Production and Consumption Data Inputs	Python	
310	Override Meter ID of mobile meter based on timestamp	Python	Not implemented
999	Sum hourly production/consumption for each substation	Python	

Process 100

Description
Standardize Metering Point Data of DSOs

Indicative Complexity	Running time	Input	Output
O(n)	Dependent on DSO	Dependent on DSO	100_OUT

Functional description	Pseudocode
This process reads the input data on metering data from the DSO and transforms it into a DSO agnostic dataset that will have an identical data structure to the input from the other DSOs. It will define the datatype for every field and export it to a CSV file.	n/a

Remarks: All code provided in the Appendix and as individual files.

Process 110

Description
Replace mobile meter ID with virtual meter ID

Indicative Complexity	Running time*	Input	Output
O(n)	12.85 seconds	100_OUT	110_OUT

Functional description	Pseudocode
This process will first filter for all mobile meters. A new field will be created called MeteringPoint_ID_New and a new ID will be assigned. All fields will be dropped that are not required - only [MeteringPoint_ID_New, MeteringPoint_ID_New, FromDate, ToDate] will remain, and exported as a CSV file.	Load meter metadata Filter data for mobile meters and add to a list For meter in list: Set counter to 1 For each occurrence of mobile meter ID in metadata set: Add a suffix based on counter to the meter ID in the metadata set Increase counter by 1 Save updated meter metadata as 110_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters.

Process 120

Description
Assign X and Y coordinates to each Meter ID based on address

Complexity	Running time*	Input	Output
O(n)	1,072.89 (120.89) seconds	110_OUT	120_OUT

Functional description	Pseudocode
This process will first read data 110_OUT, and create a temporary table with all Meters that do not have X,Y coordinates. Based on the address of these meters, X & Y coordinates will be determined. These will then be inserted into the initial data, and exported as a new CSV file.	Load meter metadata 110_OUT Filter for meters with missing coordinate data and store in list For each meter in list: Query address and use geolocator (e.g. Nominatim) to find coordinates if coordinates are found: Insert coordinates into 110_OUT of the respective meter Drop meters from 110_OUT that do not have coordinates after the process Save updated meter metadata as 120_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters. Bracket indicates running time of no artificaial break after geolocation of a meter.

Process 200

Description
Standardize Substation Data of each DSO

Complexity	Running time	Input	Output
O(n)	Dependent on DSO	Dependent on DSO	200_OUT

Functional description	Pseudocode
This process reads the input data on metering data from the DSO and transforms it into a DSO agnostic dataset that will have an identical data structure to the input from the other DSOs. It will define the datatype for every field and export it to a CSV file.	n/a

Remarks: All code provided in the Appendix and as individual files.

Process 210

Description
Create tuples of Meter ID with associated Substation ID

Complexity	Running time*	Input	Output
$O(n^2)$	121.62 seconds	120_OUT, 200_OUT	210_OUT

Functional description	Pseudocode

Based on the geographical distance, find the closest substation to each of the meters in 120_OUT, and substations in the standardized 200_OUT. The allocation is considering the grid level of meters and substations. Based on this, a new table is created where every unique MeteringPoint_ID_New has exactly one Substation_ID assigned. This will be exported to CSV.	Load 120_OUT and 200_OUT dataset
	For each meter in 120_OUT
	if grid level of meter is 3:
	calculate distance to all substations at grid level 2 based on coordinates
	append substation ID of closest substation to the entry
	if grid level of meter is 4:
	calculate distance to all substations at grid level 3 based on coordiantes
	append substation ID of closest substation to the entry
	Save updated 120_OUT as 210_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters.

Process 220

Description
Reallocate meters that violate GDPR

Complexity	Running time*	Input	Output
$O(n^2)$	1.72 seconds	200_OUT, 210_OUT	220_OUT

Functional description	Pseudocode
Check in 210_OUT if there is a substation that has less than 3 private meters and no non-private meter assigned to it. If this is the case, find an alternative substation (method to be determined), and replace the substation_ID of these meters with the alternative substation_ID. The new allocation considers the grid level of meters and substations. Export the new table in CSV format.	Load 210_OUT Assign value to each meter in 210_OUT based on customergroup (0->1, 1->3) Aggregate this value for each unique substation in 210_OUT Create a list with value of less than 3 as “non-compliant” , value of more than 2 “compliant” For each substation in non-compliant: Calculate distance to all compliant substations Find the closest substation and append substation ID to the non-compliant ID Replace non-compliant substations with compliant substations based on tuples identified in loop in the 210_OUT dataset Save updated 120_OUT as 220_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters.

Process 300

Description
Standardize Production and Consumption Data Inputs for each DSO

Complexity	Running time	Input	Output
O(n)	Dependent on DSO	Dependent on DSO	300_OUT

Functional description	Pseudocode
This process reads the input data on production and consumption from the DSO and transforms it into a DSO agnostic dataset that will have an identical data structure to the input from the other DSOs. It will define the datatype for every field, and export it to a CSV file.	n/a

Remarks: All code provided in the Appendix and as individual files.

Process 310

Description
Override Meter ID of mobile meter based on timestamp

Complexity	Running time*	Input	Output
$O(n^2)$**	4,192.32 seconds	110_OUT, 300_OUT	310_OUT

Functional description	Pseudocode
This process will identify all meters in dataset 300_OUT that are mobile meters. It will replace the MeteringPoint_ID with the MeteringPoint_ID_New value from 110_OUT, if it falls in the time period specified in 110_OUT. Finally, the column MeteringPoint_ID will be renamed MeteringPoint_ID_New.	Load 110_OUT and 300_OUT Filter 110_OUT for mobile meters For each unique initial meter ID: For each entry with the same meter ID in 300_OUT: Assign new (virtual) meter ID associated with the initial ID based on timestamp of 300_OUT to the entry in 300_OUT Save updated 300_OUT as 310_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters. **In current implementation. A more efficient implementation is feasible.

Process 999

Description
Sum hourly production/consumption for each substation

Complexity	Running time*	Complexity	Output
O(n)	664.68 seconds	220_OUT, 310_OUT	999_OUT

Functional description	Pseudocode
Create a new field in 110_OUT to assign a substation based on 330_OUT. Now, aggregate the consumption (Value_Wh) for every unique combination of Timestamp and Substation_ID. Drop the meter ID field, and export it to CSV.	Load 220_OUT and 310_OUT Many-to-one merge of 310_OUT to substations in 220_OUT associated with the (virtual) meter ID Aggregate the consumption (Value_Wh) grouped by unique combination of (Substation, Timestamp). Pivot the dataset, such timestamps are included as columns, and substation IDs as rows Save the final dataset as 999_OUT

Remarks: All code provided in the Appendix and as individual files. *On standard testbench with 50,000 meters.

Contents

Dataset Overview

Process ID	Description	Type
100_OUT	(MeteringPoint_ID, CustomerGroup, GridLevel, Meter_LocX, Meter_LocY, MeterAddress, IsMobile, FromDate, ToDate)	CSV
110_OUT*	(MeteringPoint_ID, MeteringPoint_ID_New, CustomerGroup, GridLevel, Meter_LocX, Meter_LocY, MeterAddress, IsMobile, FromDate, ToDate)	CSV
120_OUT	(MeteringPoint_ID_New, CustomerGroup, GridLevel, Meter_LocX, Meter_LocY)	CSV
200_OUT	(Substation_ID, GridLevel, Substation_LocX, Substation_LocY)	CSV
210_OUT	(MeteringPoint_ID_New, Substation_ID, CustomerGroup, GridLevel)	CSV
220_OUT	(MeteringPoint_ID_New, Substation_ID)	CSV
300_OUT	(MeteringPoint_ID, Timestamp, Value_Wh)	CSV
310_OUT*	(MeteringPoint_ID_New, Timestamp, Value_Wh)	CSV
999_OUT	(Substation_ID, Timestamp, Value_Wh)	CSV

*Not implemented due to lack of data.

Dataset 100_OUT | 110_OUT | 120_OUT | 200_OUT | 210_OUT | 220_OUT | 300_OUT | 310_OUT | 999_OUT

Dataset 100_OUT.CSV

Field	Format	Description
MeteringPoint_ID	String	An ID uniquely identifying the metering point to which the meter is connected
CustomerGroup	Int	(0=Residential/household/cabin/vacation home, 1=Other)
GridLevel	Int	The grid-level at which this metering point is connected
Meter_LocX	Float	X coordinate of the metering point
Meter_LocY	Float	Y coordinate of the metering point
MeterAddress	String	Address of the MeteringPoint_ID
IsMobile*	Int	Boolean value indicating if this metering point is non-stationary (0)/mobile (1)
FromDate*	Datetime	Date from which the metering point was active at this location
ToDate*	Datetime	Date to which the metering point was active at this location

*Not included in implementation, since mobile meters were not considered in the analysis.

Dataset 110_OUT.CSV*

Field	Format	Description
MeteringPoint_ID	String	An ID uniquely identifying the metering point to which the meter is connected
MeteringPoint_ID_New**	String	Updated ID that assigns mobile meters a unique ID for each location
CustomerGroup	Int	(0=Residential/household/cabin/vacation home, 1=Other)
GridLevel	Int	The grid-level at which this metering point is connected
Meter_LocX	Float	X coordinate of the metering point
Meter_LocY	Float	Y coordinate of the metering point
MeterAddress	String	Address of the MeteringPoint_ID
FromDate	Datetime	Date from which the metering point was active at this location
ToDate	Datetime	Date from which the metering point was active at this location

*Not implemented due to lack of data. **In implementation called MeteringPoint_ID, since no mobile meters were considered in the analysis

Dataset 120_OUT.CSV

Field	Format	Description
MeteringPoint_ID_New*	String	Updated ID that assigns mobile meters a unique ID for each location
CustomerGroup	Int	(0=Residential/household/cabin/vacation home, 1=Other)
GridLevel	Int	The grid-level at which this metering point is connected
Meter_LocX	Float	X coordinate of the metering point
Meter_LocY	Float	Y coordinate of the metering point

*In implementation called MeteringPoint_ID, since no mobile meters were considered in the analysis.

Dataset 200_OUT.CSV

Field	Format	Description
Substation_ID	String	An ID uniquely identifying the substation
GridLevel	Int	The grid-level at which this substation is operating
Substation_LocX	Float	X coordinate of this substation
Substation_LocY	Float	Y coordinate of this substation

Dataset 210_OUT.CSV

Field	Format	Description
MeteringPoint_ID_New*	String	Updated ID that assigns mobile meters a unique ID for each location
Substation_ID	Float	Substation_ID of the substations that is closest to the meter
CustomerGroup	Int	(0=Residential/household/cabin/vacation home, 1=Other)
GridLevel	Int	The grid-level at which this substation is operating

*In implementation called MeteringPoint_ID, since no mobile meters were considered in the analysis.

Dataset 220_OUT.CSV

Field	Format	Description
MeteringPoint_ID_New*	String	Updated ID that assigns mobile meters a unique ID for each location
Substation_ID	String	Substation_ID of the substations that is closest to the meter, after accounting for GDPR compliance (no Substation_ID with less than 3 private meters and no non-private meter)

*In implementation called MeteringPoint_ID, since no mobile meters were considered in the analysis.

Dataset 300_OUT.CSV

Field	Format	Description
MeteringPoint_ID	String	An ID uniquely identifying the metering point to which the meter is connected
Timestamp	Datetime	Timestamp to which the meter reading applies
Value_Wh	Int	Net consumption in Wh

Remarks
Due to the massive amount of data, the number of fields should be limited as much as possible to improve processing efficiency.

Dataset 310_OUT.CSV*

Field	Format	Description
MeteringPoint_ID_New	String	Updated ID that assigns mobile meters a unique ID for each location
Timestamp	Datetime	Timestamp to which the meter reading applies
Value_Wh	Int	Consumption in Wh

Remarks
Due to the massive amount of data, the number of fields should be limited as much as possible to improve processing efficiency. ValueType may still be dropped.

*Not implemented due to lack of data.

Dataset 999_OUT.CSV

Field	Format	Description
Substation_ID	String	Substation_ID of the substations that is closest to the meter, after accounting for GDPR compliance (no Substation_ID with less than 3 private meter and no none private meter)
Timestamp	Datetime	Timestamp to which the meter readings apply
Value_Wh	Int	Consumption in Wh aggregated from all meters that are associated to the Substation_ID at the point in time (Timestamp). Positive value – consumption > production (net consumption) Negative value – consumption < production (net production)

Multiconsult

MEMO

PROJECT	**Developing Methods for Combining Data that Can Be Used for Calculating Power Distance**	DOCUMENT CODE	10219088-TVF-NOT-01
SUBJECT	Description of Data Needs from DSOs in Reference Group	ACCESSIBILITY	Open
CLIENT	**NVE**	PROJECT MANAGER	Magnus Sletmoe Dale
CONTACT	Magnus Sletmoe Dale	PREPARED BY	Magnus Sletmoe Dale
COPY TO	NVE, THEMA Consulting Group	RESPONSIBLE UNIT	Multiconsult ASA

1 Background

Multiconsult and THEMA Consulting Group are requesting the following datasets in order to assist NVE in the reform of its DSO tariff setting methodology:

Chapter	Dataset	Point of contact	Priority
3.1	• Production and consumption data	Multiconsult	1
3.2	• Metering point metadata	Multiconsult	
3.3	• Substation data	Multiconsult	
3.4	• Outage data	THEMA	2
3.5	• Power exchange data between R-grid and HVD-grid	THEMA	

All datasets are required but may be compiled at different times according to the specified priority.

Supplied data files must be clearly structured and formatted consistently across files. Supplied datasets should be described in a separate document. All data fields should be considered mandatory unless specified otherwise.

All GIS-data needs to be in georeferenced. Please specify the coordinate system that is used by specifying its common name or WKID. Additional fields may be included in the dataset at the discretion of DSO.

Additional requirements are given in the paragraphs below. Multiconsult remains flexible on other aspects of these datasets, such as its structure, format, naming and/or split into multiple files.

2 Definitions

The period for which data is requested is 01.03.2019-01.03.2020 and is referred to as the *period of interest*.

The following grid structure and naming comventions are employed in this document:

02	01.07.2020		Msd	Msd	Msd
01	29.06.2020		Msd	Msd	Msd
REV.	DATE	DESCRIPTION	PREPARED BY	CHECKED BY	APPROVED BY

Grid layer	Typical voltage	Grid level
Transmission grid	300 – 420 kV	1
◆ *Substation (“transformatorstasjon”)*		
Regional grid / R-Grid	33 kV – 132 kV	2
◆ *Substation (“transformatorstasjon”/”innmatingspunkt”)*		
High-voltage distribution grid / HVD-grid	1 kV – 22kV	3
◆ *Substation (“nettstasjon”)*		
Low-voltage distribution grid / LVD-grid	400 V / 230 V	4

3 Dataset description

3.1 Production and consumption data

Meter reading data must be provided in a non-proprietary format such as a Comma Separated Values-file (CSV).

A dataset for all meters connected to the HVD- or LVD-grid which at any time during the period of interest consumed power from or exported power to the grid, containing the data fields below.

Field Name	Description
• MeteringPoint_ID	An ID uniquely identifying the meatering point (“målepunkt”) to which the meter is connected. Matches values used for metering point metadata. All MeteringPoint_IDs included in this dataset must appear in the metadata file.
• Timestamp	Timestamp to which the meter reading applies. May be split into date or hour.
• Value_kWh*	Production/consumption registered by the meter (cumulative meter readings to be avoided). Should be empty if value is missing (value 0 instead denoting real zero reading).
• ValueType*	0= Consumption-production (net consumption value), 1=Consumption data only, 2=Production data only.

*Duplicated for “prosumers”/”plusskunder” where separated consumption and production data series are available.

3.2 Metering point metadata

A dataset for all metering points connected to the HV or LV distribution grid in the period of interest containing the fields below.

Please include one entry for each deployment of mobile/non-stationary metering points such as “anleggskasser” during the period of interest.

Field Name	Description
• MeteringPoint_ID	An ID uniquely identifying the metering point to which the meter is connected (“målepunkt”). Matches values used for production and consumption data.
• Substation_ID	An ID uniquely identifying the substation to which the metering point is connected. Matches values in the substation dataset.
• CustomerGroup	A value uniquely identifying the type of customer nehind this metering point. DSOs are invited to use their own customer group breakdown on the conditions i) All metering points are assigned to a customer group, and ii) Households and holiday homes/cabins can be uniquely idenfitied
• GridLevel	Grid level at which this metering point is operating (3 or 4)
• Meter_LocX	The X coordinate of this metering point
• Meter_LocY	The Y coordinate of this metering point
• MeterAddress	Address at which the meter is installed. Mandatory where Meter_LocX and MeteR_LocY are not both provided. Should reflect the physical location of the meter (as opposed to postal adress of customer, for example). Address values should be formatted using the format <Gateadresse>_<Number>_<Letter>,_<Postkode>_<Poststed> (where “_” denotes a space).
• IsMobile	Boolean value indicating whether this metering point is non-stationary/mobile such as “anleggskasser”. Should be FALSE or 0 for most entries.
• FromDate	The date from which the metering point was active at this location (for mobile metering points only).
• ToDate	The date until which the metering point was active at this location (for mobile metering points only).

3.3 Substation data

For each substation operating at level 2 (“innmatingspunkt”) and 3 (“nettstasjon”), please provide the following data:

Field Name	Description
• Substation_ID	An ID uniquely identifying the substation, matching values given for this field in the Metering point meta dataset.
• Substation_LocX	The X coordinate of this substation
• Substation_LocY	The Y coordinate of this substation
• GridLevel	The grid-level at which this substation is operating

3.4 Outage data

In addition to data requirements posed by Multiconsult in the context *of Combining Data that Can Be Used for Calculating Power Distance*, THEMA Consulting Group requires data on outages for an extension of the project related to measuring the task of reliability.

Outage data should be provided in a dataset for the period of interest that covers the entirety of the owned grid area. Each outage should be assigned to affected metering points identified by ID and customer type. The dataset should contain the following data:

Field Name	Description
• MeteringPoint_ID	An ID uniquely identifying the meter. Matches values used for meter metadata and metering data. All meter IDs included in this dataset must appear in the metadata file.
• CustomerGroup	A value uniquely identifying the type of customer to which the meter is connected. DSOs are invited to use their own customer group breakdown on the conditions i) All metering points are assigned to a customer group, and ii) Households and holiday homes/cabins can be uniquely idenfitied
• Timestamp	Timestamp for the occurrence of an outage. May be split into date or hour.
• Duration	Duration of occurred outage. The duration can be provided separately or implicitly from the timestamp of an occurrence. i.e. outage occurred at 01.02. 03.00 with a duration of 2:00h or an outage occurred in the hours 01.02. 03.00 and 01.02. 04.00
• Value_kW	Expected demand at time of outage (avbrutt effekt) per meter and time step
• OutageType	Type of outage (1 = varslet avbrudd 2 = ikke varslet avbrudd) per outage

3.5 Power exchange data between R-grid and HVD-grid

Data must be provided in a non-proprietary format such as a Comma Separated Values-file (CSV).

A dataset for power exchange between R-grid and HVD-grid in the period of interest containing the fields below.

Field Name	Description
• Substation_ID	An ID uniquely identifying the transformer station between R-grid and HVD-grid (“innmatingspunkt”). Matches values used in the substation dataset. All Transformer_IDs included in this dataset must appear in the substation dataset.
• Timestamp	Timestamp to which the meter reading applies. May be split into date or hour.
• Value_kWh	Transferred power between grid levels at given time step and transformer station.

	Positive values should indicate power flow from the R-grid (grid level 2) to the HVD-grid (grid level 3), negative readings imply power exchange from the HVD-grid to the R-grid. Should be empty if value is missing (value 0 instead denoting real zero reading).

4 Data privacy

Multiconsult acknowledges the applicability of GDPR regulation and the sensitivity of the requested data. Data providers are invited to present applicable NDAs/confidentiality agreement templates for signature by Multiconsult and THEMA Consulting.

5 Data delivery

Multiconsult is facilitating a secure FTP server for delivery of requested files. One account will be created for each DSO participating in the reference group.

Please request server and account details from Multiconsult, specifying the following information about the user:

- Full name
- Company name
- E-mail
- Mobile telephone number

6 Contact

Multiconsult and THEMA Consulting Group wish to facilitate the extraction of the requested datasets.

Please address any questions to Magnus Sletmoe Dale (magnus.dale@multiconsult.no / +47 957 08 707) or Jan Ohlenbusch (jan.ohlenbusch@multiconsult.no / +47 412 35 475 during weeks 28 and 29). Please note our offices will be closed during weeks 30 and 31.

For questions concerning outage data and power exchange data between the R- and HVD-grid, please contact Lisa Zafoschnig (lisa.zafoschnig@thema.no / +47 404 03 742)

Data Standardization Log

Methods for Combining Data for Use in Calculation of Power Distance

Multiconsult Norge AS – 15.10.2020

1 Meter Metadata – Standardization (P100)

The following will outline the steps performed to standardize the metering metadata (200_OUT). The final, standardized dataset will have the following structure:

Field	Format	Description
MeteringPoint_ID	String (int)	ID uniquely identifying the point to which the meter is connected
CustomerGroup	Int	(0=Residential/household/cabin/vacation home, 1=Other)
GridLevel	Int	The grid-level at which this metering point is connected
Meter_LocX	Float	X coordinate of the metering point
Meter_LocY	Float	Y coordinate of the metering point
MeterAddress	String	Address of the MeteringPoint_ID

Notably, after receiving the data, it has been decided to exclude mobile meters from the analysis – due to unavailable data/insufficient data quality. This is further detailed in the separate section of each DSO. Therefore, the final dataset does not include the fields *IsMobile*, *FromDate* and *ToDate*.

Furthermore, duplicated values were observed in multiple meter metadata sets (MeteringPoint_ID not unique). Based on feedback from DSOs, it is expected that this change is due to changes in customers at the relevant meter. Accordingly, duplicate entries sometimes have different customer types classified. Duplicate entries are removed from the dataset. To ensure compliance with GDPR guidelines, it has been decided that in the case that any of the customer groups of duplicate entries includes a residential customer, the remaining entry is classified as a residential customer. This will solely influence the GDPR compliant allocation of meters to substations.

1.1 Jæren Everk

Data provided

The DSO provided one CSV file named “Kunder med NS og koordinat_ny.csv”, including metadata for **9760 meters**. All fields that were requested were provided, with the exception of *IsMobile*, *FromDate* and *ToDate* – meaning that **no information on mobile meters was provided**. Customer groups were provided in classes based on “industry” (Code 1-37). Coordinates were provided for every meter. Coordinates were provided in *UTM/EUREF89 sone 32 + NN2000*

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- Coordinates were formatted to points as the decimal separator (from commas).

- The customer group field was adjusted for the purpose of this project, assigning a 0 for residential/private customers and 1 for non-private customers.
- Multiple duplicate entries were identified. Duplicate entries were removed, and the customer group field was assigned a 0, if any of the duplicate meters were assigned a residential customer group. This **reduced the dataset from 9760 meters to 8884 meters**.

1.2 Klepp Energi

Data provided

The DSO provided one CSV file named “Metering point metadata.csv”, including metadata for **9298 meters**. All fields that were requested were provided, including *IsMobile*, *FromDate* and *ToDate*. In total, 142 meters were indicated to be mobile meters (88 unique meter IDs) – however, without geographical information. Furthermore, customer groups were provided in classes based on “industry” (Code 1-37). Coordinated were provided in WGS 1984.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- X and Y coordinates were backwards, which was corrected.
- The 142 meter observations indicated as mobile meters were removed in agreement with NVE, reducing the number of meters from 9298 to 9156.
- The customer group field was adjusted for the purpose of this project, assigning a 0 for residential/private customers and 1 for non-private customers.
- Multiple duplicate entries were identified. Duplicate entries were removed, and the customer group field was assigned a 0, if any of the duplicate meters were assigned a residential customer group. 159 duplicated entries were identified and removed. This **further reduced the dataset from 9156 meters to 8997 meters.**
- Coordinates for one meter were missing

MeteringPoint_ID	GridLevel	Meter_LocX	Meter_LocY	MeterAddress
707057500026038774	4	NaN	NaN	Fjellvegen , 4351 KLEPPE

1.3 Mørenett

Data provided

The DSO provided one Excel file named “AB-data_03092020.xlsx”, including metadata for **64,883 meters**. All fields that were requested were provided, except for mobile meter data (only noting “Anleggskasse” in a meter name and address field). The format of data provided deviated from the requested format, therefore, some adjustments were necessary. All meter IDs were unqiue IDs in the provided data.Furthermore, customer groups were provided in classes based on “industry” (Code 1-37). Coordinated were provided in *EUREF89 UTM sone 32.*

Methodology

Since all data was provided, however, in deviating formats, some adjustments were necessary in the standardization process:

- The 250 meter observations indicated as “ANLEGGSKASSE” in the “Adresse” or “Lastkategori”, which were designated as mobile meters in discussions with the DSO. These meters were removed in agreement with NVE, reducing the number of meters **from 64,883 to 64,633**.
- The customer group field was adjusted for the purpose of this project, assigning a 0 for residential/private customers and 1 for non-private customers.
- The “spenning” field provided by the DSO was translated into a “GridLevel” field, where <1kV equals grid level 4, and >=1kV equals grid level 3.
- For missing “spenning” values, it is assumed that all of these are at grid level 4 – supported by the ascertain by the DSO that most of the missing observations are 230/400V.
- 170 meters only have 0 values for coordinates (10 of which with low-resolution address data). These have to be adjusted based on Address information in further processing of the assignment.

1.4 Glitre Energi

Data provided

The DSO provided one Excel file named “mp.xlsx”, including metadata for **94,733 meters**. All fields that were requested were provided, except for mobile meter data (only noting “Anleggskasse” in a field with meter name. The format of data provided deviated from the requested format, therefore, some adjustments were necessary. All meter IDs were unqiue IDs in the provided data. Furthermore, customer groups were provided in classes based on “industry” (Code 1-37). Coordinated were provided in *EUREF89 UTM zone 33.*

Methodology

Since all data was provided, however, in deviating formats, some adjustments were necessary in the standardization process:

- The 268 meter observations indicated as “kasse” in the “MLPKTNAMN” or “BRUKSOMRAADE”, which are mobile meters, as agreed in discussions with the DSO. These meters were removed in agreement with NVE, reducing the number of meters **from 94,733 to 94,465**.
- The customer group field was adjusted for the purpose of this project, assigning a 0 for residential/private customers and 1 for non-private customers.
- **1801 meters were missing coordinates and only had addresses**. Six of these only had postcodes. These will be geocoded during data processing
- **10 meters were missing Grid Level values**. All of these are non-private. It is assumed that these are at Grid Level 4.

2 Substation Data – Standardization (P200)

The following will outline the steps performed to standardize the substation data (300_OUT). The final, standardized dataset will have the following structure:

Field	Format	Description
Substation_ID	String (int)	An ID uniquely identifying the substation
Substation_LocX	Float	X coordinate of this substation
Substation_LocY	Float	Y coordinate of this substation
GridLevel	Int	The grid-level at which this substation is operating

In the following, the steps followed for each DSO are described in more detail.

2.1 Jæren Everk

Data provided

The DSO provided one CSV file named “NS med koordinat.csv”, which contained substation data for **408 substations.** All fields requested were provided, including the *GridLevel*. All substation IDs were unique IDs – and no values were missing. Coordinates were provided in *UTM/EUREF89 sone 32 + NN2000*.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- X and Y coordinates were backwards, which was corrected.

The final dataset remains at 408 substations.

Identified issues relevant for nationwide implementation

As noted before, a validation mechanism for coordinates may be introduced. Apart from that, no issues were identified based on the provided dataset.

2.2 Klepp Energi

Data provided

The DSO provided one CSV file named “Substation data.csv”, which contained substation data for **328 substations.** All fields requested were provided, including *GridLevel*. All substation IDs were unique IDs – and no values were missing. Coordinates were provided in *WGS 1984*.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- For 5 unique X & Y coordinate and GridLevel observations 10 duplicate entries were observed. All duplicate entries are specified below, indicating the omitted Substation IDs. This may become relevant at a later validation stage, when the substation IDs indicated in the meter metadata are compared against the geographical distance approach. This reduced the total number of substation from 328 to 318 substations.
- One strange ID that has a 16-digit code remains in the data: 707057500071879810

Substation_ID	Substation_LocX	Substation_LocY	GridLevel	Omitted
K060	5.658542	58.74479	3	NO
K888	58.77773	5.631918	3	NO
K999	58.77773	5.631918	3	YES
KL	5.616488	58.78614	2	NO
HA	5.660284	58.77864	2	NO
TU	5.658542	58.74479	2	NO
707057500071879711	5.616488	58.78614	2	YES
707057500071879728	5.616488	58.78614	2	YES
707057500071879650	5.660284	58.77864	2	YES
707057500071879667	5.660284	58.77864	2	YES
707057500071879674	5.660284	58.77864	2	YES
707057500071879742	5.658542	58.74479	2	YES
707057500071879766	5.658542	58.74479	2	YES
707057500071879780	5.658542	58.74479	3	YES
707057500071879797	5.658542	58.74479	3	YES

2.3 Mørenett

The DSO provided two Excel file named “NS-data.xlsx” and “TS-data.xlsx”, which contained substation data for **2653 substations.** All fields requested were provided, however, instead of *GridLevel* the a voltage (Spenning) field was provided for transfornerstasjoner, and no information on nettstasjoner (assumed to be at GridLevel 3, as confirmed by the DSO). All substation IDs were unique IDs – and no values were missing. Coordinates were provided in *EUREF89 UTM sone 32*.

Methodology

The data was largely provided in the requested format. Therefore, only limited adjustments were necessary in the standardization process:

- The TS and NS datasets were merged.
- A grid-level of 3 was assigned to all nettstasjoner (as confirmed by the DSO). The grid-level for Transformerstasjon was verified against the “voltage” field provided in the TS dataset, and a 2 was assigned accordingly.
- One entry in the TS dataset was removed based on discussion with the DSO, as this is a “spenningshever” in the grid - a 15MVA transformer, switching system in / out, voltage regulator, etc. This reduced the number of substations from 2653 to 2652.

Substation_ID	Substation_LocX	Substation_LocY	Voltage	Omitted
421612	392607.102479	6926771	22kV - 2	YES

- 17 substations (nettstasjoner, GridLevel 3) have missing coordinates, and were omitted from the analysis. This reduced the number of substations from 2652 to 2635.

Substation_ID	Substation_LocX	Substation_LocY	GridLevel	Omitted
3006709	0.0	0.0	3	YES
2435930	0.0	0.0	3	YES
2198700	0.0	0.0	3	YES
2189513	0.0	0.0	3	YES
2131502	0.0	0.0	3	YES
2131467	0.0	0.0	3	YES
2130487	0.0	0.0	3	YES
2130107	0.0	0.0	3	YES
2129962	0.0	0.0	3	YES
2129266	0.0	0.0	3	YES
2061412	0.0	0.0	3	YES
2058421	0.0	0.0	3	YES
2058201	0.0	0.0	3	YES
2058178	0.0	0.0	3	YES
2058159	0.0	0.0	3	YES
2058113	0.0	0.0	3	YES
2054309	0.0	0.0	3	YES

2.4 Glitre Energi

Data provided

The DSO provided two Excel file named “Uttrekk-Netbas-Nettstasjoner-07-08-2020 (003).xlsx” and “Effektdistanse_Innmatingspunkt.xlsx”, which contained substation data for **3573 substations.** All fields requested were provided, however, instead of *GridLevel* nettstasjoner and tranformerstasjoner were provided separately (assumed to be at GridLevel 3 & 2). One substation IDs was used twice – and no values were missing. Coordinates were provided in *EUREF89 UTM sone 32*.

Methodology

The data was largely provided in the requested format. Therefore, only limited adjustments were necessary in the standardization process:

- A duplicate entry was removed that had slightly deviating coordinates, reducing the dataset from 3573 to 3572.

Substation_ID	Substation_LocX	Substation_LocY	GridLevel	Omitted
20097	577176.526296	6689949	3	NO
20097	577174.353242	6689950	3	YES

- Two entries with missing coordinates were removed, as discussed with the DSO (not in use in this area). This further **reduced the dataset from 3572 to 3570 substations.**

Substation_ID	Substation_LocX	Substation_LocY	GridLevel	Omitted
NSUVDAL	0.0	0.0	3	YES
NS0005	0.0	0.0	3	YES

- For substations at *GridLevel* 2, the provided Meter ID was applied as the Substation_ID in the dataset.
- Some of the observations provided were production transformers. These were *not* excluded from the analysis.

3 Electricity Consumption & Production Data – Standardization (P300)

The following will outline the steps performed to standardize the metering data (consumption and production data - 100_OUT). The final, standardized dataset will have the following structure:

Field	Format	Description
MeteringPoint_ID	String (int)	ID uniquely identifying the point to which the meter is connected
Timestamp	Datetime	Timestamp to which the meter reading applies
Value_Wh	Int	Net consumption in Wh (consumption with a positive sign)

As discussed with the Client, is has been decided to output all values as net consumption, as the approach to supplying consumption & production values varies significantly across the DSOs.

3.1 Jæren Everk

Data provided

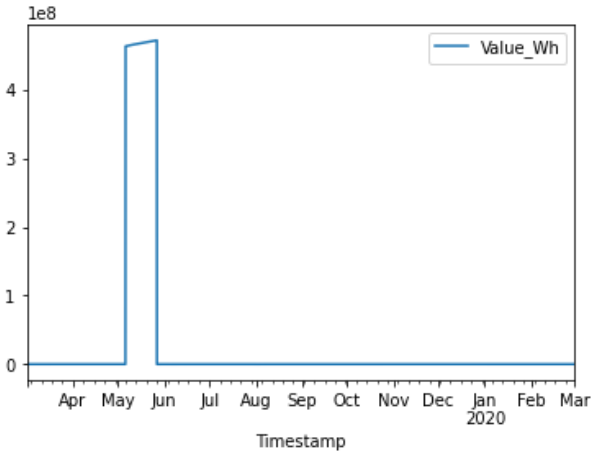
The DSO provided 14 SDV files named “ForbrukYYYYMM.sdv” – and “Kombi_Forbruk.sdv” and “Kombi_Produksjon.sdv” for Plusskunder. These included in total **78,973,466 hourly metering observations**. All fields that were requested were provided.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- For efficiency purposes, consumption & production values were converted from kWh to Wh and converted from floats to integers.
- As confirmed with the DSO, the indicated timestamp is “forwards-referencing” meaning that the consumption for e.g. 10.03.2019 3:00 refers to the consumption between 03:00 and 04:00. For consistency, **the hourly observations were moved ahead by one hour**, such that the timestamp for consumption data for all DSOs is backwards referencing.
- Both production and consumption values are positive. Consumption and production values were netted, whereas all values with ValueType 2 were multiplied by (-1), and consumption and production values were aggregated, such that only one observation for every Timestamp & MeteringPoint_ID combination remains. Furthermore, the ValueType column was removed to make the data more memory efficient. The netting process reduced the number of observations from **78,973,466 hourly metering observations to 78,780,502** (by 192,964).
- Some observation have significant outliers identified in the data and appear to display the absolute meter readings for a period of time. These were identified via the standard deviation across the dataset. The following meters were removed from the dataset. Please also find an illustrative example of one of these meters. This further reduced the number of observations from **78,780,502 to 78,632,494** (by 148,008).

MeteringPoint_ID	Std – Value_Wh
707057500025437158	1.089052e+08
707057500025453196	8.092718e+06
707057500025464314	6.107470e+06
707057500025468848	5.800787e+06
707057500025459754	5.547066e+06
707057500025470667	4.636315e+06
707057500025438667	4.564741e+06
707057500025477673	4.486413e+06
707057500025466264	4.432559e+06
707057500025438278	4.165466e+06
707057500025464819	3.961795e+06
707057500025451789	3.948241e+06
707057500025440615	3.775842e+06
707057500025540780	3.380995e+06
707057500025457415	3.340703e+06
707057500025529013	3.096447e+06
707057500025506854	2.390555e+06
707057500025449342	1.843546e+06



3.2 Klepp Energi

Data provided

The DSO provided two CSV files named Production and consumption data 0103-0109.csv" and "Production and consumption data 0109-0103.csv", which included **78,710,399 hourly metering observations**. All fields that were requested were provided.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- Formatting of consumption/production values was adjusted, changing the decimal separator from comma to points. Furthermore, thousand separators (space) were removed.
- For efficiency purposes, consumption & production values were converted from kWh to Wh, and converted from floats to integers.
- As confirmed with the DSO, data was provided in CET and CEST. In agreement with NVE it was decided to convert the timezone to *UTC+1* for the dataset. To that end, all observations after 31.03.2019 3:00 and before "27.10.2019 03:00" were moved back by one hour. Since there exist two consumption entries for each meter at timestamp "27.10.2019 03:00", the mean of these values was calculated and applied to both timestamp "27.10.2019 03:00" and "27.10.2019 02:00", removing the initial observations. This step increased the number of observations from **78,710,399 hourly metering observations to 78,710,750** (by 351), due to meters that only had one observation at timestamp "27.10.2019 03:00".
- Observations from one metering point with significant outliers was removed, after confirmation from the DSO that this is due to an error in their CIS system and that the meter is not in use. This reduced the number of observations from **78,710,750 hourly metering observations to 78,707,799** (by 2,951).

MeteringPoint_ID	Timestamp	Value_Wh	ValueType
707057500026039030	2019-05-25 01:00:00	-2.147484e+09	1
707057500026039030	2019-06-01 01:00:00	-2.147484e+09	1
707057500026039030	2019-06-08 01:00:00	-2.147484e+09	1
707057500026039030	2019-06-15 01:00:00	-2.147484e+09	1
707057500026039030	2019-06-22 01:00:00	-2.147484e+09	1
707057500026039030	2019-06-29 01:00:00	-2.147484e+09	1

- After these adjustments, only positive values for ValueType 1 and 2 remained.
- Consumption and production values were netted, whereas all values with ValueType 2 were multiplied by (-1), and consumption and production values were aggregated, such that only one observation for every Timestamp & MeteringPoint_ID combination remains. Furthermore, the ValueType column was removed to make the data more memory efficient. The netting process reduced the number of observations from **78,707,799 hourly metering observations to 78,619,997** (by 87,802).
- As confirmed with the DSO, the indicated timestamp is "backwards-looking" meaning that the consumption for e.g. 10.03.2019 3:00 refers to the consumption between 02:00 and 03:00. Therefore, no adjustments were made.

3.3 Mørenett

Data provided

The DSO provided four **DSV** files named “export_DD.MM.YY-DD.MM.YY.dsv”, which included **605,462,520 hourly metering observations**. All fields that were requested were provided.

Methodology

Some adjustments were necessary in the standardization process:

- For the purpose of memory efficient processing, the four DSV files were split into 12 CSV files for every month.
- The provided dataset was presented with each hour as a separate column. The data was “unpivoted”/”melted”, for consistent data structure with the other DSOs. To obtain the timestamp in the appropriate data format, further manipulations were necessary. The timestamp is established as “backwards-looking” meaning that the consumption for e.g. 10.03.2019 3:00 refers to the consumption between 02:00 and 03:00.
- NaN values were observed and dropped in the subsequent step, reducing the dataset from **605,462,520 to 605,407,096** (by 55,424).
- For efficiency purposes, consumption & production **values were converted from kWh to Wh**, and converted from floats to integers.
- Multiple duplicate entries for MeteringPoint_ID and Timestamp were observed. After further investigations, it was discovered that these meters could report up to three times the monthly hours, with varying Wh consumption values. This leads us to the working assumptions that some meters report in 30-minute and 29-minute resolution instead of hourly resolution. For reference, 95.4% of meters have the number observations according to the number of hours in a month. 4.4% of meters have twice as many observations (likely 30-minute resolution), and 0.2% of meters have three times as many observations (likely 20-minute resolution). As average of the monthly sum at each meter is very similar across these categories (with the exception of the 0.2% which have a low sample size), which further supports this assumption. Therefore, as part of the netting process, the duplicate entries were aggregated for the respective hour.
- Consumption and production values were netted, whereas all values with ValueType 2 were multiplied by (-1), and consumption and production values were aggregated, such that only one observation for every Timestamp & MeteringPoint_ID combination remains. Furthermore, the ValueType column was removed to make the data more memory efficient. Together with the previous bulletpoint, this reduced **the number of observations from 605,407,096 to 581,131,960 (by 24,275,136).**

Month	Provided rows	NaN values	After dropping Nan	Netting	After netting
Mar-19	51 603 240	1 222	51 602 018	2 645 085	48 956 933
Apr-19	49 849 416	7 885	49 841 531	2 443 603	47 397 928
May-19	51 428 760	7 720	51 421 040	2 404 331	49 016 709
Jun-19	49 772 712	7 807	49 764 905	2 259 604	47 505 301
Jul-19	51 373 944	6 830	51 367 114	2 205 611	49 161 503
Aug-19	51 336 792	7 376	51 329 416	2 149 073	49 180 343
Sep-19	49 658 832	8 949	49 649 883	2 002 194	47 647 689
Oct-19	51 288 144	7 456	51 280 688	1 980 378	49 300 310
Nov-19	49 603 248	32	49 603 216	1 805 392	47 797 824
Dec-19	51 198 912	30	51 198 882	1 760 250	49 438 632
Jan-20	50 924 520	24	50 924 496	1 471 872	49 452 624
Feb-20	47 424 000	93	47 423 907	1 147 743	46 276 164
Total	**605 462 520**	**55 424**	**605 407 096**	**24 275 136**	**581 131 960**

3.4 Glitre Energi

Data provided

The DSO provided 13 text files with comsumption named “meterreading_MMYY.txt” and production values “meterreading_all_prod.txt”, which included **823,192,040 hourly metering observations**. All fields that were requested were provided.

Methodology

Since the data was provided in the requested format, only limited adjustments were necessary in the standardization process:

- The **timestamp format provided had some inconsistency**, with the first hour of every day not having an hourly value associated with it (e.g. “01.02.2020” instead of “01.02.2020 00.00.00”). This was corrected for.
- Formatting of consumption/production values was adjusted, changing the decimal separator from comma to points.
- For efficiency purposes, consumption & production **values were converted from kWh to Wh**, and converted from floats to integers.
- Some null values are observed in the dataset. These were removed, which reduced the dataset from **823,192,040** observations to **823,182,064** observations (by 9,976).
- Since consumption and production data was provided separately, the **monthly consumption values were merged with the production values in the same period** (no simultaneous processing of the entire annual dataset possible due to memory constraints).
- As confirmed with the DSO, data was provided in CET and CEST. In agreement with NVE it was decided to convert the timezone to *UTC+1* for the dataset. To that end, all observations after 31.03.2019 3:00 and before “27.10.2019 03:00” were moved back by one hour. Since observations for each meter at timestamp “27.10.2019 03:00” contained meter readings for two hours, the total consumption was divided by two and appended as consumption for “27.10.2019 03:00” and “27.10.2019 02:00” - removing the initial observations for “27.10.2019 03:00”. This increased the dataset

from **823,182,064** observations to **823,203,472** observations (by 1,456) due to meters that only had an observation at timestamp “27.10.2019 03:00”.

- Consumption and production values were netted, whereas all values with ValueType 2 were multiplied by (-1), and consumption and production values were aggregated, such that only one observation for every Timestamp & MeteringPoint_ID combination remains. Furthermore, the ValueType column was removed to make the data more memory efficient. This reduced the dataset from **823,203,472** observations to **822,210,478** observations (by 992,994).
- A brief validation of consumption values was performed – confirming that all values have positive consumption values, and no significant outliers.
- As confirmed with the DSO, the indicated timestamp is “backwards-looking” meaning that the consumption for e.g. 10.03.2019 3:00 refers to the consumption between 02:00 and 03:00. Therefore, no adjustments were made.

	Starting	NA values	After NA exclusion	Production values	After adding production	Adjusting for summertime	After summertime adjustment	Netting	Final data
Mar-19	68 849 976	-	68 849 976	68 160	68 918 136	- 92 701	68 825 435	- 60 638	68 764 797
Apr-19	66 743 617	-	66 743 617	67 824	66 811 441	-	66 811 441	- 60 624	66 750 817
May-19	69 070 009	-	69 070 009	75 840	69 145 849	-	69 145 849	- 68 400	69 077 449
Jun-19	66 989 232	-	66 989 232	75 888	67 065 120	-	67 065 120	- 68 688	66 996 432
Jul-19	69 364 008	-	69 364 008	85 056	69 449 064	-	69 449 064	- 77 616	69 371 448
Aug-19	69 459 600	-	69 459 600	92 352	69 551 952	-	69 551 952	- 84 912	69 467 040
Sep-19	67 419 529	-	67 419 529	92 495	67 512 024	-	67 512 024	- 85 296	67 426 728
Oct-19	69 873 575	-	69 873 575	98 399	69 971 974	94 157	70 066 131	- 91 084	69 975 047
Nov-19	67 836 648	-	67 836 648	101 376	67 938 024	-	67 938 024	- 94 176	67 843 848
Dec-19	70 277 096	9 976	70 287 072	106 608	70 393 680	-	70 393 680	- 99 168	70 294 512
Jan-20	70 342 416	-	70 342 416	110 448	70 452 864	-	70 452 864	- 103 008	70 349 856
Feb-20	65 885 544	-	65 885 544	106 344	65 991 888	-	65 991 888	- 99 384	65 892 504
Total	**822 111 250**	**9 976**	**822 121 226**	**1 080 790**	**823 202 016**	**1 456**	**823 203 472**	**- 992 994**	**822 210 478**

Initialization of the Python environment as used in this assignment

The packages can be installed via the following command in the terminal (assuming that pip is available).

```
pip install pandas==1.1.0
pip install numpy==1.18.1
pip install re==2.2.1
pip install geopy==1.21.0
pip install scipy==1.5.2
```

If an environment manager (such as Anaconda) is used to manage Python libraries, the install procedures may vary. A virtual environment was established to ensure that dependencies between the packages are satisfied and not disrupted by new versions of a package (e.g. a newer version of the pandas library may not be supported by the other libraries that are used).

Implementation of Processes

The process during the assignment were implemented in Python and ArcGIS. The table below provides an overview of the respective software that was applied, including comments.

Process number	Software	Comment
100_MN	Python	Standardization process for Mørenett
100_JE	Python	Standardization process for Jæren
100_KE	Python	Standardization process for Klepp
100_GE	Python	Standardization process for Glitre
200_MN	Python	Standardization process for Mørenett
200_JE	Python	Standardization process for Jæren
200_KE	Python	Standardization process for Klepp
200_GE	Python	Standardization process for Glitre
300_MN	Python	Standardization process for Mørenett
300_JE	Python	Standardization process for Jæren
300_KE	Python	Standardization process for Klepp
300_GE	Python	Standardization process for Glitre
110	Python	Draft available, not implemented
120	ArcGIS	
210	ArcGIS	Draft available in Python, not implemented
220	ArcGIS	Draft available in Python, not implemented
310	Python	Draft available, not implemented
999	Python	

** GE = Glitre Energi, JE = Jæren Everk, KE = Klitre Energi, MN = Mørenett*

The Python scripts to the above processes are provided on the following pages. Please note that the in- and out-paths are specific to the workstations. In the provided scripts, these are referring to an external hard drive. Please also note that different version numbers than stated in the final report could lead to compatibility issues.

P100_MN – Meter Metadata Standardization Mørenett

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt
import re

# Define global variable
path_in = 'D:/input/MN/'
path_out = 'D:/data/'

# Import data
df = pd.read_excel(str(path_in)+'mp.xlsx')

# Create an empty dataframe for 100 dataset
df_100 = pd.DataFrame(columns=['MeteringPoint_ID','CustomerGroup', 'GridLevel', 'Meter_LocX','Meter
_LocY','MeterAddress','IsMobile','FromDate','ToDate', 'Substation_ID'])

# Assign data to new df
df_100['MeteringPoint_ID'] = df['MAALEPUNKT']
df_100['Meter_LocY'] = df['GPS pos y mp'].astype('float')
df_100['Meter_LocX'] = df['GPS pos x mp'].astype('float')
df_100['Temp_C'] = df['SLUTTBRUKERGRUPPE']
df_100['MeterAddress'] = df['MP_ADRESSE']
df_100['GridLevel'] = df['Nivå']

# Create mobile meter boolean based on "kasse" string
df_100['Temp_MM'] = df['MLPKTNAMN']
df_100['Temp_MM2'] = df['BRUKSOMRAADE']
df_100['IsMobile'] = 0
df_100['IsMobile'].loc[df_100['Temp_MM'].str.contains('kasse', flags=re.IGNORECASE, regex=True)==Tr
ue] = 1
df_100['IsMobile'].loc[df_100['Temp_MM2'].str.contains('kasse', flags=re.IGNORECASE, regex=True)==T
rue] = 1

# Allocate metering points to customer groups (0 = Private, 1= Non-private)
df_100['CustomerGroup'] = 1
df_100['CustomerGroup'].loc[(df_100['Temp_C']=='35 - Husholdninger') | (df_100['Temp_C']=='36 - Hyt
ter og fritidshus')] = 0

# Filter mobile meters
df_100 = df_100[df_100['IsMobile']==0]

# Drop unnecessary columns
df_100 = df_100.drop(columns=['Temp_MM', 'Temp_C', 'IsMobile','FromDate','ToDate', 'Temp_MM2'])

# Assume that missing values for grid level at meteringpoint ID are at grid-level 4
df_100['GridLevel'].loc[df_100['GridLevel'].isna()==True] = 4

# Set index for memory efficiency
df_100 = df_100.set_index('MeteringPoint_ID')

# Save to hardrive and encode to latin to ensure Norwegian letters are retained in CSV file
df_100.to_csv(str(path_out)+'100_OUT_MN.csv', encoding='latin-1')
```

P100_JE – Meter Metadata Standardization Jæren

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/JE/'
path_out = 'D:/data/'

# Import data
df = pd.read_csv(str(path_in)+'Kunder med NS og koordinat_ny.csv', delimiter=';', encodi
ng='latin-1')

# Create an empty dataframe for 100 dataset
df_100 = pd.DataFrame(columns=['MeteringPoint_ID','CustomerGroup', 'GridLevel', 'Meter_L
ocX','Meter_LocY','MeterAddress','Substation_ID'])

# Rename column in imported file for ease of import
df = df.rename(columns={"MeterAdress": "MeterAddress"})

# Append dataframe to new 100 dataframe
df_100 = df_100.append(df)

# Replace comma separator with dot separator, format coordiantes as floats
df_100['Meter_LocX'] = df_100['Meter_LocX'].str.replace(',','.')
df_100['Meter_LocY'] = df_100['Meter_LocY'].str.replace(',','.')
df_100['Meter_LocX'] = df_100['Meter_LocX'].astype('float')
df_100['Meter_LocY'] = df_100['Meter_LocY'].astype('float')

# Allocate metering points to customer groups (0 = Private, 1= Non-private).
df_100['CustomerGroup_New'] = 1
df_100['CustomerGroup_New'].loc[(df_100['CustomerGroup']=='35') | (df_100['CustomerGroup
']=='36')] = 0
df_100 = df_100.drop(columns='CustomerGroup')
df_100 = df_100.rename(columns={'CustomerGroup_New':'CustomerGroup'})

# Switch coordinates based on feedback from GIS expert (wrong allocation in initial data
set)
df_100 = df_100.rename(columns={'Meter_LocX':'Meter_LocY', 'Meter_LocY':'Meter_LocX'})

# Sort to ensure that residential customers appear first
df_100 = df_100.sort_values(by='CustomerGroup', ascending=True)

# Remove duplicate meters (if residential customers, CustomerGroup 0 remains - which is
subject to GDPR adjustments)
df_100.drop_duplicates(subset='MeteringPoint_ID', keep='first', inplace=True)

# Set index for memory efficiency
df_100 = df_100.set_index('MeteringPoint_ID')

# Save to hardrive and encode to latin to ensure Norwegian letters are retained in CSV f
ile
df_100.to_csv(str(path_out)+'/100_OUT_JE.csv',encoding='latin-1')
```

P100_KE – Meter Metadata Standardization Klepp

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/KE/'
path_out = 'D:/data/'

# Import data
df = pd.read_csv(str(path_in)+'Metering point metadata.csv', delimiter=';')

# Create an empty dataframe for 100 dataset
df_100 = pd.DataFrame(columns=['MeteringPoint_ID','CustomerGroup', 'GridLevel', 'Meter_L
ocX','Meter_LocY','MeterAddress','IsMobile','FromDate','ToDate', 'Substation_ID'])

# Assign data to the 100 dataframe
df_100['MeteringPoint_ID'] = df['METERINGPOINT_ID']
df_100['Meter_LocY'] = df['Meter_LocX'] # Coordinates switched due to error
df_100['Meter_LocX'] = df['Meter_LocY'] # Coordinates switched due to error
df_100['MeterAddress'] = df['METERADRESS']
df_100['CustomerGroup'] = df['CustomerGroup']
df_100['GridLevel'] = df['GRIDLEVEL']
df_100['IsMobile'] = df['ISMOBILE']
df_100['FromDate'] = df['FROMDATE']
df_100['ToDate'] = df['TODATE']
df_100['Substation_ID'] = df['SUBSTATION_ID']

# Drop all mobile meter observations and associated columns
df_100 = df_100[df_100['IsMobile']==False]
df_100 = df_100.drop(columns=['IsMobile', 'FromDate', 'ToDate'])

# Allocate metering points to customer group boolean (0 = Private, 1= Non-private)
df_100['CustomerGroup_New'] = 1
df_100['CustomerGroup_New'].loc[(df_100['CustomerGroup']=='35 - Husholdninger') | (df_10
0['CustomerGroup']=='36 - Hytter og fritidshus')] = 0
df_100 = df_100.drop(columns='CustomerGroup')
df_100 = df_100.rename(columns={'CustomerGroup_New':'CustomerGroup'})

# Sort to ensure that residential customers appear first
df_100 = df_100.sort_values(by='CustomerGroup')

# Remove duplicate meters (if residential customers, CustomerGroup 0 remains - which is
subject to GDPR adjustments)
df_100.drop_duplicates(subset='MeteringPoint_ID', keep='first', inplace=True)

# Set index for memory efficiency
df_100 = df_100.set_index('MeteringPoint_ID')

# Save to hardrive and encode to latin to ensure Norwegian letters are retained in CSV f
ile
df_100.to_csv(str(path_out)+'100_OUT_KE.csv',encoding='latin-1')
```

P100_GE – Meter Metadata Standardization Glitre

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt
import re

# Define global variable
path_in = 'D:/input/MN/'
path_out = 'D:/data/'

# Import data
df = pd.read_excel(str(path_in)+'mp.xlsx')

# Create an empty dataframe for 100 dataset
df_100 = pd.DataFrame(columns=['MeteringPoint_ID','CustomerGroup', 'GridLevel', 'Meter_L
ocX','Meter_LocY','MeterAddress','IsMobile','FromDate','ToDate', 'Substation_ID'])

# Assign data to new df
df_100['MeteringPoint_ID'] = df['MAALEPUNKT']
df_100['Meter_LocY'] = df['GPS pos y mp'].astype('float')
df_100['Meter_LocX'] = df['GPS pos x mp'].astype('float')
df_100['Temp_C'] = df['SLUTTBRUKERGRUPPE']
df_100['MeterAddress'] = df['MP_ADRESSE']
df_100['GridLevel'] = df['Nivå']

# Create mobile meter boolean based on "kasse" string
df_100['Temp_MM'] = df['MLPKTNAMN']
df_100['Temp_MM2'] = df['BRUKSOMRAADE']
df_100['IsMobile'] = 0
df_100['IsMobile'].loc[df_100['Temp_MM'].str.contains('kasse', flags=re.IGNORECASE, rege
x=True)==True] = 1
df_100['IsMobile'].loc[df_100['Temp_MM2'].str.contains('kasse', flags=re.IGNORECASE, reg
ex=True)==True] = 1

# Allocate metering points to customer groups (0 = Private, 1= Non-private)
df_100['CustomerGroup'] = 1
df_100['CustomerGroup'].loc[(df_100['Temp_C']=='35 - Husholdninger') | (df_100['Temp_C']
=='36 - Hytter og fritidshus')] = 0

# Filter mobile meters
df_100 = df_100[df_100['IsMobile']==0]

# Drop unnecessary columns
df_100 = df_100.drop(columns=['Temp_MM', 'Temp_C', 'IsMobile','FromDate','ToDate', 'Temp
_MM2'])

# Assume that missing values for grid level at meteringpoint ID are at grid-level 4
df_100['GridLevel'].loc[df_100['GridLevel'].isna()==True] = 4

# Set index for memory efficiency
df_100 = df_100.set_index('MeteringPoint_ID')

# Save to hardrive and encode to latin to ensure Norwegian letters are retained in CSV f
ile
df_100.to_csv(str(path_out)+'100_OUT_MN.csv', encoding='latin-1')
```

P200_MN – Meter Metadata Standardization Mørenett

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/MN/'
path_out = 'D:/data/'

# Import nettstasjoner and transformerstasjoner datasets
df = pd.read_excel(str(path_in)+'NS-data.xlsx')
df2 = pd.read_excel(str(path_in)+'TS-data.xlsx')

# Create an empty dataframe for 200 dataset
df_200 = pd.DataFrame(columns=['Substation_ID', 'Substation_LocX','Substation_LocY','Gri
dLevel'])

# Assign data of nettstasjoner to new df
df_200['Substation_ID'] = df['Objektnummer']
df_200['Substation_LocX'] = df['Geografisk øst']
df_200['Substation_LocY'] = df['Geografisk nord']
df_200['Placement'] = df['Plassering']
df_200['GridLevel'] = 3

# Create temporary dataframe in format to be appended to 200 dataset
df_temp = pd.DataFrame(columns=['Substation_ID', 'Substation_LocX','Substation_LocY','Gr
idLevel'])
df_temp['Substation_ID'] = df2['Objektnummer']
df_temp['Substation_LocX'] = df2['Geografisk øst']
df_temp['Substation_LocY'] = df2['Geografisk nord']
df_temp['GridLevel'] = 2
df_temp['Voltage'] = df2['Spenning']
df_temp['Substation_Name'] = df2['Stasjonsnavn']
df_temp['Other_ID'] = df2['Driftsmerking']

# Append transformatorstasjoner to 200 dataset
df_200 = df_200.append(df_temp)

# Drop duplicates & nettstasjoner with missing coordinates
df_200.drop_duplicates(subset=['Substation_LocX','Substation_LocY','GridLevel'], keep=Fa
lse, inplace=True)

# Drop 22kV substation (based on DSO feedback)
df_200 = df_200[df_200['Voltage']!=22]

# Drop unnecessary columns
df_200 = df_200.drop(columns=['Placement', 'Voltage', 'Substation_Name','Other_ID'])

# Set index for memory efficiency
df_200 = df_200.set_index('Substation_ID')

# Save to hardrive
df_200.to_csv(str(path_out)+'200_OUT_MN.csv')
```

P200_JE – Meter Metadata Standardization Jæren

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/JE/'
path_out = 'D:/data/'

# Read dataset
df_200 = pd.read_csv(str(path_in)+'NS med koordinat.csv', delimiter=';')

# Rename columns
df_200 = df_200.rename(columns={'Substation_LocX':'Substation_LocY','Substation_LocY':'S
ubstation_LocX'})

# Set index for memory efficiency
df_200 = df_200.set_index('Substation_ID')

# Save to hardrive
df_200.to_csv(str(path_out)+'200_OUT_JE.csv')
```

P200_KE – Meter Metadata Standardization Klepp

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/KE/'
path_out = 'D:/data/'

# Read data
df_200 = pd.read_csv(str(path_in)+'Substation data.csv', delimiter=';')

# Drop duplicates
df_200.drop_duplicates(subset=['Substation_LocX', 'Substation_LocY', 'GridLevel'], keep=
'first', inplace=True)

# Set index
df_200 = df_200.set_index('Substation_ID')

# Save to harddrive
df_200.to_csv(str(path_out)+'200_OUT_KE.csv')
```

P200_MN – Meter Metadata Standardization Mørenett

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/GE/'
path_out = 'D:/data/'

# Read both datasets (nettstasjoner and transformatorstasjoner)
df = pd.read_excel(str(path_in)+'Uttrekk-Netbas-Nettstasjoner-07-08-2020 (003).xlsx')
df2 = pd.read_excel(str(path_in)+'Effektdistanse_Innmatingspunkt.xlsx')

# Create an empty dataframe for 200 dataset
df_200 = pd.DataFrame(columns=['Substation_ID','Substation_LocX', 'Substation_LocY', 'Gr
idLevel'])

# Assign data of nettstasjoner to new df
df_200['Substation_ID'] = df['Driftsmerking']
df_200['Substation_LocX'] = df['UTM(EUREF89) SONE 32 ØST']
df_200['Substation_LocY'] = df['UTM(EUREF89) SONE 32 NORD']
df_200['GridLevel'] = 3

# Drop duplicate substations
df_200.drop_duplicates(subset='Substation_ID', keep='first', inplace=True)

# Drop two substations with 0.0 coordinates
df_200 = df_200[df_200['Substation_LocX']!=0]

# Create temporary dataframe in format to be appended to 200 dataset
df_temp = pd.DataFrame(columns=['Substation_ID', 'Substation_LocX','Substation_LocY','Gr
idLevel'])
df_temp['Substation_ID'] = df2['MpktID']
df_temp['Substation_LocX'] = df2['UTM(EUREF89) SONE 32 ØST']
df_temp['Substation_LocY'] = df2['UTM(EUREF89) SONE 32 NORD']
df_temp['GridLevel'] = 2

# Append transformatorstasjoner to 200 dataset
df_200 = df_200.append(df_temp)

# Drop empty rows
df_200.dropna(subset=['Substation_ID'], inplace=True)

# Set index for memory efficiency
df_200 = df_200.set_index('Substation_ID')
# Save to hardrive
df_200.to_csv(str(path_out)+'200_OUT_GE.csv')
```

P300_GE – Meter Metadata Standardization Glitre

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt
from pathlib import Path

# Define global variable
path_in = 'D:/input/GE/'
path_out = 'D:/data/'

# Define function for memory-efficient datetime parsing (applied later)
def lookup(s):
    dates = {date:pd.to_datetime(date, format='%d.%m.%Y %H.%M.%S') for date in s.unique(
)}
    return s.apply(lambda v: dates[v])

# Create a list "filenames" with all meterreading data
filepath = Path(path_in)
filenames = [fname for fname in filepath.iterdir() if fname.is_file() and fname.suffix =
= '.txt']

# Create a counter for loop
counter = 1

# To run on a regular system (requires ca 32 GB of memory), a loop is create to process
each month separately
for filename in filenames:

    ####################
    # Consumption Values

    # Create empty 300 dataframe
    df_300 = pd.DataFrame(columns=['MeteringPoint_ID', 'Timestamp', 'Value_kWh', 'ValueT
ype'])

    # Read dataset
    df_300 = pd.read_csv(filename, sep=";")

    # Rename columns to fit final dataset structure
    df_300 = df_300.rename(columns={'METERING_POINT_ID':'MeteringPoint_ID','TIMESTAMP':'
Timestamp','VALUE_KWH':'Value_kWh','VALUETYPE':'ValueType'})

    # Add missing timestamps (some hour suffixes are missing)
    df_300['Timestamp'] = df_300['Timestamp'].apply(lambda x: x+str(' 00.00.00') if len(
x)<14 else x)

    # Parse timestamps
    df_100['Timestamp'] = lookup(df_100['Timestamp'])

    # Change decimal separator
    df_300['Value_kWh'] = df_300['Value_kWh'].str.replace(',','.').astype('float')

    # Drop empty consumption values
    df_300 = df_300.dropna(subset=['Value_kWh'])

    # Convert consumption from kWh to Wh
    df_300['Value_kWh'] = df_300['Value_kWh']*1000

    # Convert to int (memory optimization)
    df_300['Value_kWh'] = df_300['Value_kWh'].astype('int')
```

```
    # Rename column from kWh to Wh
    df_300 = df_300.rename(columns={'Value_kWh': 'Value_Wh'})

    ####################
    # Production Values

    # Import production values
    df_prod = pd.read_csv(str(path_in)+'/prod/meterreading_all_prod.txt', sep=";")

    # Rename columns
    df_prod = df_prod.rename(columns={'METERING_POINT_ID':'MeteringPoint_ID','TIMESTAMP'
:'Timestamp','VALUE_KWH':'Value_kWh','VALUETYPE':'ValueType'})

    # Add missing timestamps (some hour suffixes are missing)
    df_prod['Timestamp'] = df_prod['Timestamp'].apply(lambda x: x+str(' 00.00.00') if le
n(x)<14 else x)

    # Parse timestamps
    df_prod['Timestamp'] = pd.to_datetime(df_prod['Timestamp'], format='%d.%m.%Y %H.%M.%
S')

    # Change decimal separator
    df_prod['Value_kWh'] = df_prod['Value_kWh'].str.replace(',','.').astype('float')

    # Convert consumption from kWh to Wh
    df_prod['Value_kWh'] = df_prod['Value_kWh']*1000
    df_prod['Value_kWh'] = df_prod['Value_kWh'].astype('int')

    # Rename columns
    df_prod = df_prod.rename(columns={'Value_kWh': 'Value_Wh'})

    # Keep only entries relevant to period in consumption data
    df_prod = df_prod.loc[df_prod['Timestamp'].isin(df_300['Timestamp'])]

    # Append production values to main dataframe
    df_300 = df_300.append(df_prod)

    ####################
    # Processing

    # Remove observations on 31.03.2019 03:00, as all are empty due to DST
    df_300 = df_300[df_100['Timestamp'] != '2019-03-31 03:00:00']

    # Shift observations back an hour within summertime
    df_300['Timestamp'] = np.where((df_300['Timestamp'] > '31.03.2019 03:00') & (df_300[
'Timestamp'] < '2019-10-
27 03:00:00'), df_300['Timestamp'] - dt.timedelta(hours=1), df_300['Timestamp'])

    # Create a temporary dataframe of all observations at 27.10.2019 03:00 since these o
ccur twice (due to summertime)
    df_temp = df_300[df_300['Timestamp'] == '27.10.2019 03:00']

    # Divide consumption at '27.10.2019 03:00' by two since it aggregated data for two h
ours
    df_temp['Value_Wh'] = df_temp['Value_Wh']/2

    # Create a copy of the dataframe
    df_temp2 = df_temp.copy()

    # Try to move observations one hour back
    try:
        df_temp2['Timestamp'] = df_temp2['Timestamp'] - dt.timedelta(hours=1)
    except:
```

```
        print('No data for 27.10.2019 03:00')

    # Remove initial observations for 27.10.2019 03:00
    df_300 = df_300[df_300['Timestamp'] != '2019-10-27 03:00:00']

    # Append temp and temp2 dataframe (including new '27.10.2019 03:00', allocated to '2
7.10.2019 03:00' and '27.10.2019 02:00')
    df_300 = df_300.append(df_temp)
    df_300 = df_300.append(df_temp2)

    # Convert production values to negative values
    df_300['Value_Wh'] = np.where(df_300['ValueType'] == 2, df_300['Value_Wh'] * (-
1), df_300['Value_Wh'])

    # Drop unnecessary columns
    df_300 = df_300.drop(columns=['ValueType'])

    # Net production and consumption values for each meter and timestamp
    df_300 = df_300.groupby(['MeteringPoint_ID','Timestamp'])['Value_Wh'].sum().reset_in
dex()

    # Set index for memory efficiency
    df_300 = df_300.set_index('MeteringPoint_ID')

    # Save to harddrive
    df_300.to_csv(str(path_out)+'GE/300_OUT_GE_'+str(counter)+'.csv')

    print('Successfully processed and exported '+str(filename))

    # Update counter
    counter = counter+1
```

P300_JE – Meter Metadata Standardization Jæren

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt
from pathlib import Path

# Define global variable
path_in = 'D:/input/KE/'
path_out = 'D:/data/'

# Create a list "filenames" with all meterreading data
directory = 'D:/input/JV/MeterReadings'
filepath = Path(directory)
filenames = [fname for fname in filepath.iterdir() if fname.is_file() and fname.suffix =
= '.sdv']

# Create empty 300 dataframe
df_300 = pd.DataFrame(columns=['MeteringPoint_ID', 'Timestamp', 'Value_kWh', 'ValueType'
])

# Read data of all files identified in filenames and append to 300
for i in filenames:
    df_temp = pd.read_csv(i, delimiter=';', names=['MeteringPoint_ID','Timestamp', 'Valu
e_kWh', 'ValueType'])
    df_300 = df_300.append(df_temp)
    print('Done '+str(i))

# Define function for memory-efficient datetime parsing
def lookup(s):
    dates = {date:pd.to_datetime(date, format='%d.%m.%Y %H.%M.%S') for date in s.unique(
)}
    return s.apply(lambda v: dates[v])

# Parse timestamps to datetime (to have consistent timestamo format)
df_300['Timestamp'] = lookup(df_300['Timestamp'])

# Convert kWh to Wh and convert type to int for memory efficiency
df_300['Value_kWh'] = df_300['Value_kWh'].astype('float')*1000
df_300['Value_kWh'] = df_300['Value_kWh'].astype('int')

# Rename variable from kWh to Wh
df_300 = df_300.rename(columns={'Value_kWh': 'Value_Wh'})

# Move all observations ahead by one hour to make timesstamps "backwards referencing" (D
SO informed that raw data is forward-looking)
df_300['Timestamp'] = df_300['Timestamp'] + dt.timedelta(hours=1)

# Convert production (ValueType =2) to negative values
df_300['Value_Wh'] = np.where(df_300['ValueType'] == 2, df_300['Value_Wh'] * (-
1), df_300['Value_Wh'])

# Drop unnecessary column
df_300 = df_300.drop(columns=['ValueType'])

# Net production and consumption values for each meter and timestamp
df_300 = df_300.groupby(['MeteringPoint_ID','Timestamp'])['Value_Wh'].sum().reset_index(
)

# Create a list of outliers based on standarddeviation. The 18 largest were identified t
o have period of absolute meterreadings
```

```
std = df_300.groupby(by='MeteringPoint_ID')['Value_Wh'].std().reset_index()
x = std.nlargest(18,'Value_Wh')

# Remove outlier meters
df_300 = df_300[(~df_300['MeteringPoint_ID'].isin(x['MeteringPoint_ID']))]

# Set index for memory efficiency
df_300 = df_300.set_index('MeteringPoint_ID')

# Save to hardrive
df_300.to_csv(str(path_out)+'JE/300_OUT_JE.csv')
```

P300_KE – Meter Metadata Standardization Klepp

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt

# Define global variable
path_in = 'D:/input/KE/'
path_out = 'D:/data/'

# Read data and append both sources
df_300 = pd.read_csv(str(path_in)+'Production and consumption data 0103-
0109.csv', delimiter=';')
df2 = pd.read_csv(str(path_in)+'Production and consumption data 0109-
0103.csv', delimiter=';')
df_300 = df_300.append(df2)

# Define function for memory-efficient datetime parsing
def lookup(s):
    dates = {date:pd.to_datetime(date, format='%d.%m.%Y %H:%M') for date in s.unique()}
    return s.apply(lambda v: dates[v])

# Parse timestamps to datetime (required for DST adjustment)
df_300['Timestamp'] = lookup(df_300['Timestamp'])

# Change decimal & thousand separator formatting for values
df_300['Value_kWh'] = df_300['Value_kWh'].str.replace(',','.')
df_300['Value_kWh'] = df_300['Value_kWh'].str.replace(' ','')

# Transform values from kWh to Wh and convert type to int (mempry efficiency)
df_300['Value_kWh'] = df_300['Value_kWh'].astype('float')*1000
df_300['Value_kWh'] = df_300['Value_kWh'].astype('int')

# Rename variable from kWh to Wh
df_300 = df_300.rename(columns={'Value_kWh': 'Value_Wh'})

# Move summertime period one hour back (final dataset in UTC+1)
df_300['Timestamp'] = np.where((df_300['Timestamp'] > '31.03.2019 03:00') & (df_300['Tim
estamp'] < '2019-10-
27 03:00:00'), df_300['Timestamp'] - dt.timedelta(hours=1), df_300['Timestamp'])

# Create a temporary dataframe of all observations at 27.10.2019 03:00 since these occur
 twice (due to summertime)
df_temp = df_300[df_300['Timestamp'] == '27.10.2019 03:00']

# Calculate the mean and return dataframe with unique meteringpoint IDs
df_temp = df_temp.groupby(['MeteringPoint_ID','Timestamp','ValueType'])['Value_Wh'].mean
().reset_index()

# Create a copy of the dataframe and move observations one hour back
df_temp2 = df_temp.copy()
df_temp2['Timestamp'] = df_temp2['Timestamp'] - dt.timedelta(hours=1)

# Remove all observations from this dataframe from the main dataset
df_300 = df_300[df_300['Timestamp'] != '27.10.2019 03:00']

# Append temp and temp2 dataframe (average of '27.10.2019 03:00', allocated to '27.10.20
19 03:00' and '27.10.2019 02:00')
df_300 = df_300.append(df_temp)
df_300 = df_300.append(df_temp2)
```

```
# Drop faulty meter observations (based on DSO feedback)
df_300 = df_300[df_300['MeteringPoint_ID'] != 707057500026039030]

# Convert production (ValueType =2) to negative values
df_300['Value_Wh'] = np.where(df_300['ValueType'] == 2, df_300['Value_Wh'] * (-
1), df_300['Value_Wh'])

# Drop unnecessary column
df_300 = df_300.drop(columns=['ValueType'])

# Net production and consumption values for each meter and timestamp
df_300 = df_300.groupby(['MeteringPoint_ID','Timestamp'])['Value_Wh'].sum().reset_index(
)

# Set index for memory efficiency
df_300 = df_300.set_index('MeteringPoint_ID')

# Save to hardrive
df_300.to_csv(str(path_out)+'KE/300_OUT_KE.csv')
```

P300_MN – Meter Metadata Standardization Mørenett

```
# Import libraries
import pandas as pd
import numpy as np
import datetime as dt
from pathlib import Path

# Define global variable
path_in = 'D:/input/MN/'
path_out = 'D:/data/'

# Process to split initial datasets into 12 monthly datasets (not automized)
df = pd.read_csv('D:/input/MN/export_01.03.19-01.06.19.dsv', delimiter=';')
df[df['READING_TIME'].str.contains("05.2019")].to_csv('D:/input/MN/Meterreadings_201905.
csv')

# Create a list "filenames" with all meterreading data
filepath = Path(path_in)
filenames = [fname for fname in filepath.iterdir() if fname.is_file() and fname.suffix =
= '.csv']

# Create a counter for loop
counter = 1

# To run on a regular system (requires ca 32 GB of memory), a loop is create to process
each month separately
for i in filenames:
    print('##################### Read dataset '+str(i))
    # Read dataset
    df_300 = pd.read_csv(i)

    # Due to structure of dataset, melt the dataset ("unpivot")
    df_300 = pd.melt(df_300, id_vars=['SERIE_OBJECTID', 'DIRECTION', 'READING_TIME'], va
lue_vars=['HOUR1', 'HOUR2', 'HOUR3', 'HOUR4','HOUR5','HOUR6', 'HOUR7', 'HOUR8', 'HOUR9',
'HOUR10','HOUR11', 'HOUR12', 'HOUR13', 'HOUR14','HOUR15','HOUR16', 'HOUR17', 'HOUR18', '
HOUR19','HOUR20','HOUR21', 'HOUR22', 'HOUR23', 'HOUR24'])

    # Rename columns to fit final dataset structure
    df_300 = df_300.rename(columns={'SERIE_OBJECTID':'MeteringPoint_ID', 'READING_TIME':
'Timestamp', 'value':'Value_kWh','DIRECTION':'ValueType'})

    # Convert kWh to Wh and convert type to int for memory efficiency
    df_300['Value_kWh'] = df_300['Value_kWh'].str.replace(',','.')
    df_300['Value_kWh'] = df_300['Value_kWh'].astype('float')*1000

    # Drop empty observations & convert field to integer for memory efficiency
    df_300 = df_300.dropna(subset=['Value_kWh'])
    df_300['Value_kWh'] = df_100['Value_kWh'].astype('int')

    # Rename column from kWh to Wh
    df_300 = df_300.rename(columns={'Value_kWh': 'Value_Wh'})

    # Remap Valuetype from +/- to 1/2
    df_300['ValueType'] = df_300['ValueType'].str.replace('-','1')
    df_300['ValueType'] = df_300['ValueType'].str.replace('+','2')

    # Strip string "HOUR" from hour variable that was created as part of the melt
    df_300['variable'] = df_300['variable'].apply(lambda x: x.lstrip('HOUR'))

    # Convert to integer and deduct one hour for datetime process (see next step)
    df_300['variable'] = df_300['variable'].astype('int') -1
```

```
    # Transform single digit hours from "1" to "01" for parsing
    df_300['variable'] = df_300['variable'].apply(lambda x: str('0') + str(x) if len(str
(x))==1 else x).astype('str')

    # Concatenate timestamp (only date) with hour ("variable")
    df_300['Timestamp'] = df_300['Timestamp'] + str(' ') + df_300['variable']

    # Drop variable column
    df_300 = df_300.drop(columns=['variable'])

    # Parse datetime based on adjusted  "Timestamp" field
    df_300['Timestamp'] = pd.to_datetime(df_300['Timestamp'], format='%d.%m.%Y %H')

    # Add hour back to make the data backwards-referencing (see deductio above)
    df_300['Timestamp'] = df_300['Timestamp'] + dt.timedelta(hours=1)

    # Convert production (ValueType =2) to negative values
    df_300['Value_Wh'] = np.where(df_300['ValueType'] == 2, df_300['Value_Wh'] * (-
1), df_300['Value_Wh'])

    # Drop unnecessary column
    df_300 = df_300.drop(columns=['ValueType'])

    # Net production and consumption values for each meter and timestamp
    df_300 = df_300.groupby(['MeteringPoint_ID','Timestamp'])['Value_Wh'].sum().reset_in
dex()

    # Set index for memory efficiency
    df_300 = df_300.set_index('MeteringPoint_ID')

    # Save to harddrive
    df_300.to_csv(str(path_out)+'MN/300_OUT_MN_'+str(counter)+'.csv')

    # Add counter
    counter = counter + 1
```

P110 – Assignment of Virtual Meter IDs

```
# Import libraries
import pandas as pd

# Define global variable
path = 'D:/data/'
DSO = MN

# Load standardized meter data
df_100 = pd.read_csv(str(path)+'100_OUT_'+str(DSO)+'.csv', encoding='latin-1')

# Filter for mobile meters
df_100 = df_100[df_100['IsMobile'] == 1]

# Get a list with all unique IDs
unique = list(df_100['MeteringPoint_ID'].unique())

# Create new dataframe for 110 dataset
df_110 = pd.DataFrame(columns=['MeteringPoint_ID', 'FromDate', 'ToDate', 'MeteringPoint_
ID_New'])

# Create loop on each unique meter
for x in unique:

    # Create a temporary dataframe df_101 for each unique ID
    df_101 = df_100[df_100['MeteringPoint_ID'] == x]

    # Set counter i
    i=0

    # Loop over each entry in the temporary dataframe
    for y in range(0,len(df_101)):

        # Create a new table for each entry in the dataframe
        df_102 = df_101.iloc[[y]]

        # Add a suffix starting at 0 for each unqiue ID and define new field name
        df_102['MeteringPoint_ID_New'] = df_102['MeteringPoint_ID']+'_'+str(i)

        # Add this to the dataframe initialized earlier
        df_110 = df_110.append(df_102)

        # Count upwards to create a unique identifier for duplicate meter entries (mm)
        i=i+1

# Read intitial meter ID location again
df_100 = pd.read_csv(str(path)+'100_OUT_'+str(DSO)+'.csv', encoding='latin-1')

# Filter for non-mobile meters
df_100 = df_100[df_100['IsMobile'] == 0]

# Rename the column to reflect changes to virtual meter points
df_100 = df_100.rename(columns={'MeteringPoint_ID': 'MeteringPoint_ID_New'})

df_110 = df_100.append(df_110)

# Set index to reduce file size
df_110 = df_110.set_index('MeteringPoint_ID_New')

# Save to harddrive
df_110.to_csv(str(path)+'110_OUT_'+str(DSO)+'.csv', encoding='latin-1')
```

P120 – Assign Coordinates to Meter IDs

```
# Import libraries
import pandas as pd
import geopy as gp
import math
from geopy.geocoders import Nominatim
import time

# Define global variable
path = 'D:/data/'
DSO = 'MN'

# Read data with addresses. Encoding added, due to Norwegian symbols in address line. Ma
y need to be updated in other instances
df_110 = pd.read_csv(str(path)+'110_OUT_'+str(DSO)+'.csv', encoding = "latin1")

# Identify meters where either X, Y or both coordinates are missing. Save as df_111
df_111 = df_110[(df_110['Meter_LocX'].isnull())|(df_110['Meter_LocY'].isnull())]

# Initialize open source geolocator (using Nominatim). Note that too many requests in a
short period can result in ban of IP.
geolocator = Nominatim(user_agent='NVE_Test')

# Reset index to allow for calling on index
df_111 = df_111.reset_index()

# Delete redundant column
df_111 = df_111.drop(columns=['index'])

# For each meter with missing coordinate do the following
for x in range(0, len(df_111)):

    # Define address field as the string used for search
    search = df_111.iloc[x]['MeterAddress'] # Define address as search

    # Only address is not empty
    print(str(x)+'/'+str(len(df_111)))

    try: # Use geolocator to locate coordinates
        location = geolocator.geocode(search)

        # Add one percent lag to avoid IP address getting flagged
        time.sleep(1)

        # Print coordinates
        print(search, location.latitude, location.longitude)

        # Write coordiantes to dataframe
        df_111.at[x,'Meter_LocX'] = location.latitude
        df_111.at[x,'Meter_LocY'] = location.longitude

    # In case the address is empty, jump to next meter, which will be omitted
    except:
        print('Error')

# Create temporary dataframe based on 110 with meters that have complete coordinates
df_112 = df_110[((df_110['Meter_LocX'].isnull()==False)&(df_110['Meter_LocY'].isnull()==
False))]

# Append dataframe with new coordinates to the one with existing coordinates
df_120 = df_112.append(df_111)
```

```
# Drop meters that could not be geolocated
x = len(df_120)
df_120 = df_120.dropna(subset=['Meter_LocX'])
y = len(df_120)
print(str(x-y)+' Meters were omitted!')

# Change index to  reduce file size
df_120 = df_120.set_index('MeteringPoint_ID_New')

# Save to harddrive
df_120.to_csv(str(path)+'120_OUT_'+str(DSO)+'.csv', encoding = "latin1")
```

P210 – Associate Meters with Substation ID (All. 1)

```
# Import relevant libraries
import pandas as pd
from scipy.spatial.distance import cdist

# Define global variable
path = 'D:/data/'
DSO = 'MN'

# Load standardized meter data
df_120 = pd.read_csv(str(path)+'120_OUT_'+str(DSO)+'.csv', encoding='latin-1')

# Load substation locations
df_200 = pd.read_csv(str(path)+'200_OUT_'+str(DSO)+'.csv')

# Zip coordinates to tuples for calculating distance
df_200['point_sub'] = [(x, y) for x,y in zip(df_200['Substation_LocX'], df_200['Substati
on_LocY'])]
df_120['point_meter'] = [(x, y) for x,y in zip(df_120['Meter_LocX'], df_120['Meter_LocY'
])]

def closest_sub(point, points):

    # Return closest point to a list of other points
    return points[cdist([point], points).argmin()]

# Define function to return ID of the substation that is associated to a metering point
def match_subid(df, col1, x, col2):
    """ Match value x from col1 row to value in col2. """
    return df[df[col1] == x][col2].values[0]

# Create temporary dataframes for each grid level
df_200_gl2 = df_200[df_200['GridLevel']==2]
df_200_gl3 = df_200[df_200['GridLevel']==3]
df_120_gl3 = df_120[df_120['GridLevel']==3]
df_120_gl4 = df_120[df_120['GridLevel']==4]

# List comprehension, applying closest_sub function to every meter coordinate and return
s closest substation points for Grid Level 3 meters
df_120_gl3['Closest'] = [closest_sub(x, list(df_200_gl2['point_sub'])) for x in df_120_g
l3['point_meter']]

# List comprehension, applying closest_sub function to every meter coordinate and return
s closest substation points for Grid Level 4 meters
df_120_gl4['Closest'] = [closest_sub(x, list(df_200_gl3['point_sub'])) for x in df_120_g
l4['point_meter']]

# Based on closest substation points, Substation ID is obtained via match_subid function
 and saved as a new field for Grid Level 3 meters
df_120_gl3['Substation_ID'] = [match_subid(df_200_gl2, 'point_sub', x, 'Substation_ID')
for x in df_120_gl3['Closest']]

# Based on closest substation points, Substation ID is obtained via match_subid function
 and saved as a new field for Grid Level 4 meters
df_120_gl4['Substation_ID'] = [match_subid(df_200_gl3, 'point_sub', x, 'Substation_ID')
for x in df_120_gl4['Closest']]

# Append allocation for both grid levels
df_210 = df_120_gl3.append(df_120_gl4)

# Drop unnecessary columns
```

```
df_210 = df_210.drop(columns=['Meter_LocX','Meter_LocY','point_meter','Closest','Meterin
gPoint_ID', 'FromDate','ToDate','MeterAddress','IsMobile'])

# Change index to reduce file size
df_210 = df_210.set_index('MeteringPoint_ID_New')

# Save to harddrive
df_210.to_csv(str(path)+'210_OUT_'+str(DSO)+'.csv', encoding='latin-1')
```

P220 – Consider GDPR Compliance in Allocation (All. 2)

```
# Import libraries
import pandas as pd
from scipy.spatial.distance import cdist

# Define global variable
path = 'D:/data/'
DSO = 'MN'

# Load substation locations
df_200 = pd.read_csv(str(path)+'200_OUT_'+str(DSO)+'.csv')

# Load initial allocation
df_210 = pd.read_csv(str(path)+'210_OUT_'+str(DSO)+'.csv')

# Create mechanism to identify GDPR compliance
df_210['GDPR'] = 0
df_210.loc[df_210['CustomerGroup'] == 1, 'GDPR'] = 3
df_210.loc[df_210['CustomerGroup'] == 0, 'GDPR'] = 1

 # Create datafrma that counts number of meters associated with every substation
df_nc = df_210.groupby(by='Substation_ID')['GDPR'].sum()

# Filter dataframe for substations with less score 3
df_nc = df_nc[df_nc<3]

# Create a list with Substation ID of these meters
nc = list(df_nc.index)

# Create dataframe of substations in compliant and non-compliant substations.
df_nc = df_200[df_200.Substation_ID.isin(nc)]
df_c = df_200[(df_200.Substation_ID.isin(nc)) == False]

# Zip coordinates to tuples for calculating distance
df_c['point_c'] = [(x, y) for x,y in zip(df_c['Substation_LocX'], df_c['Substation_LocY'
])]
df_nc['point_nc'] = [(x, y) for x,y in zip(df_nc['Substation_LocX'], df_nc['Substation_L
ocY'])]

# Define unctions for identfing closest substations
def closest_sub(point, points):

    # Identify closest "compliant" substation to each "non-compliant" substation
    return points[cdist([point], points).argmin()]

# Find substation ID based on substation point
def match_subid(df, col1, x, col2):
    """ Match value x from col1 row to value in col2. """
    return df[df[col1] == x][col2].values[0] #

# Create temporary dataframes for each grid level
df_nc_gl2 = df_nc[df_nc['GridLevel']==2]
df_c_gl2 = df_c[df_c['GridLevel']==2]
df_nc_gl3 = df_nc[df_nc['GridLevel']==3]
df_c_gl3 = df_c[df_c['GridLevel']==3]

# Calculate closest substation point for each non-compliant substation for grid level 2
df_nc_gl2['Closest'] = [closest_sub(x, list(df_c_gl2['point_c'])) for x in df_nc_gl2['po
int_nc']]

# Calculate closest substation point for each non-compliant substation for grid level 3
```

```
df_nc_gl3['Closest'] = [closest_sub(x, list(df_c_gl3['point_c'])) for x in df_nc_gl3['po
int_nc']]

# Find Substation_ID of the closest compliant substation for each non-
compliant substation for grid level 2
df_nc_gl2['Substation_ID_Compliant'] = [match_subid(df_c_gl2, 'point_c', x, 'Substation_
ID') for x in df_nc_gl2['Closest']]

# Find Substation_ID of the closest compliant substation for each non-
compliant substation for grid level 3
df_nc_gl3['Substation_ID_Compliant'] = [match_subid(df_c_gl3, 'point_c', x, 'Substation_
ID') for x in df_nc_gl3['Closest']]

# Merge grid level 2 and 3 to a nc dataset
df_nc = df_nc_gl2.append(df_nc_gl3)

#Drop unnecessary columns
df_nc = df_nc.drop(columns=['point_nc','Substation_LocY','Substation_LocX','Closest'], a
xis=1)

# Set index to reduce file size
df_nc = df_nc.set_index('Substation_ID')

# Create a dictionary based on dataframe for convenient replacement
dict_nc = df_nc.to_dict()

# Filter dictionary to remove unnecessary index
dict_nc = dict_nc["Substation_ID_Compliant"]

# Reload initial meter-subsation allocation
df_210 = pd.read_csv(str(path)+'210_OUT_'+str(DSO)+'.csv')

# Replace non-compliant Substation ID with the ID of the closest Substation to that ID
df_220 = df_210.replace({'Substation_ID': dict_nc})

# Change index to reduce file size
df_220 = df_220.set_index('MeteringPoint_ID_New')

# Save to harddrive
df_220.to_csv(str(path)+'220_OUT_'+str(DSO)+'.csv', encoding='latin-1')
```

P310 – Override Meter IDs with Virtual Meter ID

```
# Import libraries
import pandas as pd
import datetime as dt
from pathlib import Path

# Define global variable
path = 'D:/data/'
DSO = 'MN'

# Load  meter data
df_110 = pd.read_csv(str(path)+'110_OUT_'+str(DSO)+'.csv', encoding='latin-1')

# Filter 110 dataset for mobile meters (reduce processing time)
df_110 = df_110[df_110['IsMobile']==1]

# Transform field into datetime type for pandas to interpret dates
df_110['FromDate'] =  pd.to_datetime(df_110['FromDate'])
df_110['ToDate'] =  pd.to_datetime(df_110['ToDate'])

# Create a counter
counter = 1

# Create a list of all consumption files
filepath = Path(path + DSO + str('/'))
filenames = [fname for fname in filepath.iterdir() if fname.is_file() and fname.suffix =
= '.csv']

# Iterate over all consumption files
for filename in filenames:

    # Import consumption data
    df_300 = pd.read_csv(filename)

    # Transform field into datetime type for pandas to interpret dates
    df_300['Timestamp'] =  pd.to_datetime(df_300['Timestamp'])

    # Reduce dataset to only relevant consumptions observations (reduce processing time)
    df_300 = df_300[df_300['MeteringPoint_ID'].isin(df_110['MeteringPoint_ID'])]

    # Fiter dataset such that
    df_110 = df_110[df_110['MeteringPoint_ID'].isna()==False]

    # For each meter that a new ID was assigned to associate the new meter ID
    for x in range(0, len(df_110)):

        # Define original meter ID as ID
        ID = df_110.iloc[x]['MeteringPoint_ID']

        print(str(ID))

        # Define from date
        FromDate = df_110.iloc[x]['FromDate']

        # Define to date
        ToDate = df_110.iloc[x]['ToDate']

        # Define the new Meter ID
        ID_New = df_110.iloc[x]['MeteringPoint_ID_New']

        #print(FromDate, ToDate, ID, ID_New)
```

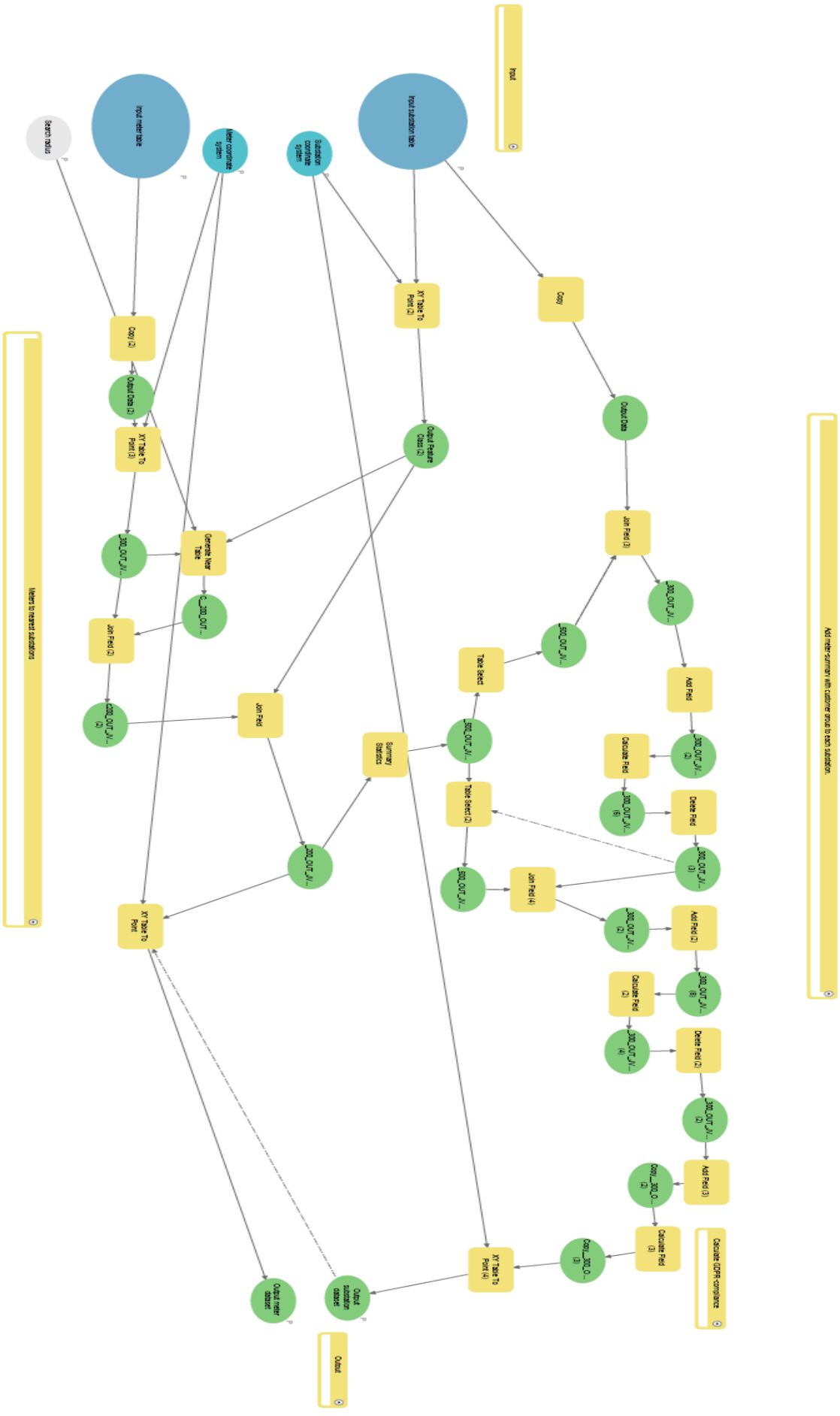
```
        #Replace the MeteringPoint ID in the original dataset, if 1) it matches the orig
inal ID, and is withing the time period
        df_300.loc[(df_300.MeteringPoint_ID == ID) & (df_300.Timestamp > FromDate) & (df
_300.Timestamp < ToDate),'MeteringPoint_ID']=ID_New

        print('Next meter')

    # Read original dataset again
    df_temp = pd.read_csv(filename)

    # Drop all meters that were assigned a new id
    df_temp = df_temp[~df_temp['MeteringPoint_ID'].isin(df_110['MeteringPoint_ID'])]

    # Merge the meters that do not need a new ID with the new dataframe that contains th
e newly assigned meters)
    df_300 = df_temp.append(df_300)

    # Rename column to indicate that the meter ID was changes (consistent acros proccess
es)
    df_310 = df_300.rename(columns={'MeteringPoint_ID': 'MeteringPoint_ID_New'})

    # Set index to reduce file size
    df_310 = df_310.set_index('MeteringPoint_ID_New')

    # Save to harddrive
    df_310.to_csv(str(path)+str(DSO)+'/310_OUT_'+str(DSO)+str('_')+str(counter)+'.csv')

    # Add counter
    counter = counter + 1
```

P999 – Creating Final Dataset

```
import pandas as pd
import numpy as np
from pathlib import Path

# Set directory & DSO
path = 'D:/data/'
DSO = 'MN'

# Load  meter data
df_220 = pd.read_csv(str(path)+'220_OUT_'+str(DSO)+'.csv')

# Create a counter
counter = 0

# Create a list of all consumption files
filepath = Path(path + DSO + str('/'))
filenames = [fname for fname in filepath.iterdir() if fname.is_file() and fname.suffix =
= '.csv']
filenames = [x for x in filenames if "310" in x.name]

# Iterate over all consumption files
for filename in filenames:

    # Import consumption data
    df_310 = pd.read_csv(filename)

    print('############ Start processing: '+str(filename))

    # Merge substation allocation with consumption data. A left join is selected to reta
in all consumption data at this stage, but add the associated substation to each observa
tion
    df_temp = pd.merge(df_310,df_220,on='MeteringPoint_ID_New',how='left')

    # Keep only relevant columns
    df_temp = df_temp[['MeteringPoint_ID_New','Timestamp','Value_Wh','Substation_ID']]

    # Report ommitted and included GWh
    print('### Total GWh omitted: '+str(df_temp[df_temp['Substation_ID'].isna()==True]['
Value_Wh'].sum()/1000000000))
    print('### Total GWh included: '+str(df_temp[df_temp['Substation_ID'].isna()==False]
['Value_Wh'].sum()/1000000000))

    # Exclude observations that do not have a substation allocated and keep in seperate
dataframe
    df_temp_excl = df_temp[df_temp['Substation_ID'].isna()==True]
    df_temp = df_temp[df_temp['Substation_ID'].isna()==False]

    # Drop unnecessary columns for processing
    df_temp =df_temp.drop(columns=['MeteringPoint_ID_New'])

    # For first iteration, create a new dataframe
    if counter ==0:
        # Excluded dataframes
        df_excluded = df_temp_excl.copy()

        # Aggregate consumption by substation ID and timestamp
        df_999 = df_temp.groupby(['Substation_ID', 'Timestamp']).sum()

        # Pivot table to obtain timestamps as columns
```

```
        df_999 = pd.pivot_table(df_999,values='Value_Wh', index=['Substation_ID'], colum
ns=['Timestamp'])

    # For second iteration append data to the existing dataframes
    else:
        # Add excluded observations to df_excluded
        df_excluded = df_excluded.append(df_temp_excl)

        # Aggregate consumption by substation ID and timestamp
        df_temp = df_temp.groupby(['Substation_ID', 'Timestamp']).sum()

        # Pivot table to obtain timestamps as columns
        df_temp = pd.pivot_table(df_temp,values='Value_Wh', index=['Substation_ID'], col
umns=['Timestamp'])

        # Append data to the df_999 dataframe
        df_999 =df_999.merge(df_temp, left_index=True, right_index=True, how='outer')

    # Add one to the counter
    counter = counter+1

# Fill missing values in final dataset with 0
df_999 = df_999.fillna(0)

# Create a list including unique meteringpoints
df_missing = pd.DataFrame(df_excluded['MeteringPoint_ID_New'].unique())

# Save final dataset and excluded values to the harddrive
df_999.to_csv(str(path)+'999_OUT_'+str(DSO)+'.csv')
df_excluded.to_csv(str(path)+'999_OUT_EXCLUDED_VALUES_'+str(DSO)+'.csv')
df_missing.to_csv(str(path)+'999_OUT_EXCLUDED_METERS_'+str(DSO)+'.csv')
```

ArcGIS model – Included for illustrative purposes



Appendix 9: Adjustment to output data set structure after report submission

Some adjustments were made to the datasets after the submission of data to Thema and discussion thereof. These adjustments relate to the reporting of consumption & production of meters that are connected to substations at grid level 2.

In the approach outlined in the report, meters at grid-level 3 were connected to substations at grid level 2. The aggregated consumption/production at substations level were included in the 999 dataset for both grid level 3 and grid level 2 substations (see picture in the top left). The 100 datasets included all consumption data from meters at grid level 4 and grid level 3 (bottom left).

Based on this submission, Thema requested changes to the structure of the output datasets, namely, that the consumption of meters at grid level 3 is appended to the 999 dataset, and excluded form the 100 dataset. This was requested to cater for the current set-up for the algorithm to calculate power distance.

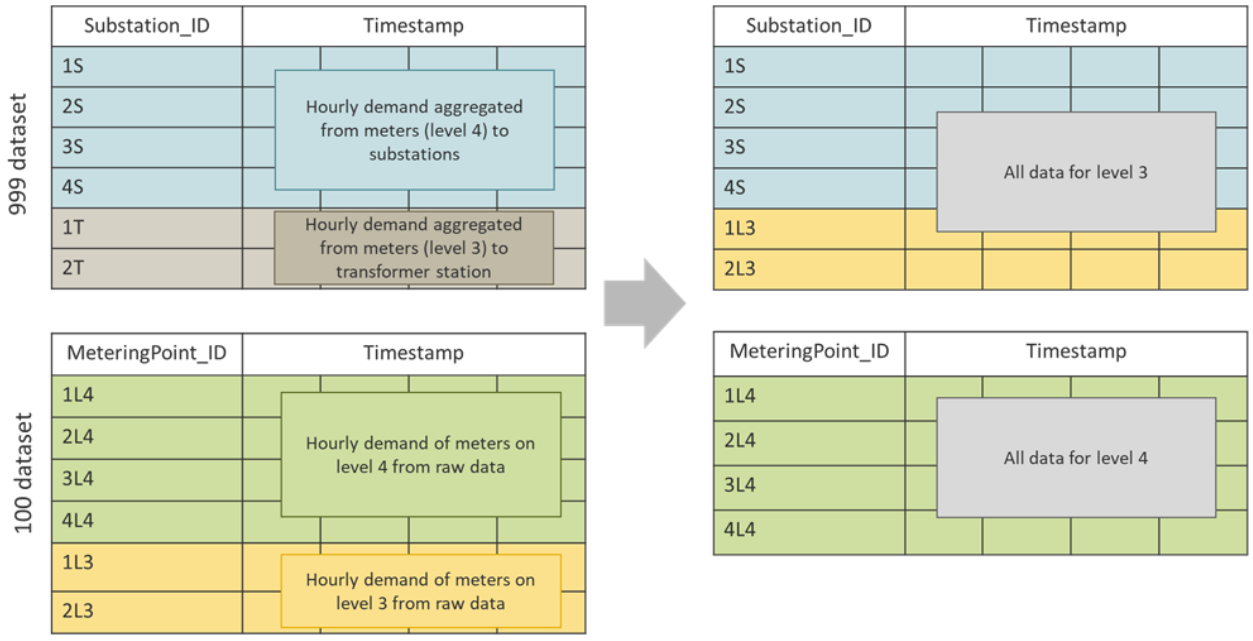


Figure 1 Initial submission of data versus amended submission to Thema (Source: Thema)

Appendix 10: Comment on dealing with meters at voltage level 1kV

In the data needs document that was sent to DSOs, the grid level of each meter was requested. In the data needs document, the following table was included as guidance for the DSOs:

Grid layer	Typical voltage	Grid level
Transmission grid	300 – 420 kV	1
◆ *Substation ("transformatorstasjon")*		
Regional grid / R-Grid	33 kV – 132 kV	2
◆ *Substation ("transformatorstasjon"/"innmatingspunkt")*		
High-voltage distribution grid / HVD-grid	1 kV – 22kV	3
◆ *Substation ("nettstasjon")*		
Low-voltage distribution grid / LVD-grid	400 V / 230 V	4

The classification was based on an NVE report, which is accessible online[1]. From 3 out of the 4 DSOs, we have only received the classified grid level, which is assumed to follow the above table. Only one DSO has provided the voltage level instead of the grid level.

In the feedback to the report, NVE commented that the voltage level 1kV should be included in the low-voltage category (grid level 4). Since most DSOs have provided meters already in a classified format, it is not possible to reallocate these meters from grid level 3 to grid level 4 without requesting an updated dataset. A check was performed on the DSO that has provided the voltage level in volt. Only 3 meters were affected, which amounts to approximately 0.005% of the total meter sample. We do not have a reason to believe that this share is significantly at other DSOs.

Based on the observations above, it was agreed to not update the dataset, but include a note as an appendix to the final report (this document).

Considerations for a national implementation

In case of a nationwide implementation, the grid level classification should be updated, to include 1kV under grid level 4. Alternatively, one might request voltage data instead of already classified grid level data from DSOs/Elhub.

[1] http://publikasjoner.nve.no/rapport/2014/rapport2014_02.pdf

NVE
Reguleringsmyndigheten
for energi – RME

Reguleringsmyndigheten for energi

MIDDELTHUNS GATE 29
POSTBOKS 5091 MAJORSTUEN
0301 OSLO
TELEFON: (+47) 22 95 95 95

www.reguleringsmyndigheten.no