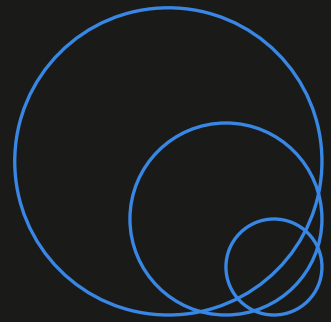
Ex.1 How to Proceed with the Exercise — Learning with Volve Field Data —

Practical Reservoir Engineering — Data-Driven Simulator Series, Session 1: Exercise Slides



Today's Agenda (1/2)

- Purpose and goals of the exercise (Ex.1)
- Setting up the Python / Jupyter Notebook environment
- Data used: Equinor Volve field data (excerpt for the exercise)
- Data loading and preprocessing
- Visualization and exploratory data analysis (EDA)
- Feature organization and deliverables

Part E

Ex.1 How to Proceed with the Exercise

E1: Purpose of the Exercise (Ex.1)

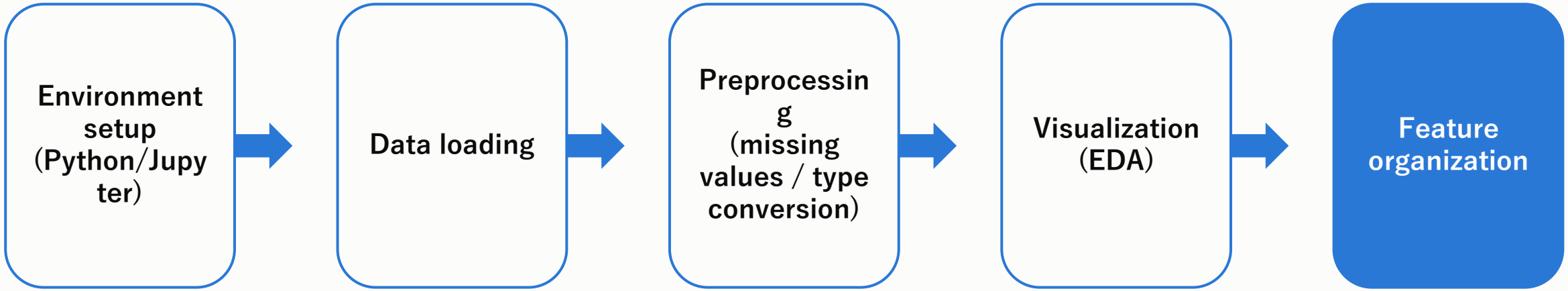
■ Goal 1: Set up a Python (Jupyter Notebook) environment on your own PC and verify that it works

■ Goal 2: Load and visualize the Equinor Volve field data (exercise excerpt in Research/References/Volve)

■ Goal 3: Experience the approach to organizing features in preparation for surrogate-model building (Session 2)

Research/References/Volve/README.md

Flow of Today's Exercise (Ex.1)



E2: Steps to Set Up the Python Environment

■ Installing Anaconda (Individual Edition) or Miniconda (lightweight version) is recommended

■ Create a course-specific virtual environment, e.g. `conda create -n reservoir-ml python=3.11`, and activate it with `conda activate reservoir-ml`

■ Install JupyterLab with `conda install jupyterlab`, launch it with `jupyter lab`, and verify it works in the browser

■ Online alternative: you can also use Google Colab without a local setup (governance checks are required when using internal data)

Anaconda official documentation https://docs.anaconda.com/anaconda/install/ ; Jupyter official documentation https://jupyter.org/install

E5–E7: Required Libraries

■ pandas / numpy: used to load, process, and aggregate the CSV-format Volve data (production rates and bottomhole pressure)

■ matplotlib / seaborn: used for time-series plots, visualizing missing values, and comparing wells

■ scikit-learn: regression, random forest, etc.

■ You can install them all at once with `pip install pandas numpy matplotlib seaborn scikit-learn`

scikit-learn official documentation https://scikit-learn.org/stable/ ; matplotlib official https://matplotlib.org/

Data Used

Equinor Volve field data (excerpt for the exercise)

› Real data from the Volve oil field in the Norwegian North Sea (operated 2008–2016), released by Equinor in 2018

› The full public dataset comprises about 40,000 files, so this exercise uses only an excerpt of the production and bottomhole-pressure data (5 files)

E8: Overview of the Volve Dataset and Its Use in This Exercise

■ A large public dataset including well logs, seismic surveys, production/injection history, simulation models, etc. (research and educational use is permitted under the Equinor Open Data Licence)

■ This exercise uses, as teaching material, five CSV files (an exercise excerpt) covering production and per-well bottomhole pressure

■ Note that the excerpt is a redistribution by a third party (the pyreservoir repository); rights to the original data remain governed by the Equinor Open Data Licence

Equinor, 'Volve field data set' https://www.equinor.com/energy/volve-data-sharing ; energistics.org, "Equinor's Volve Field Test Data" https://energistics.org/equinors-volve-field-test-data

Structure of the Stored Data (1): Production and PVT

■ Volve_production.csv: monthly data, Jan 2008–Nov 2008 (11 rows). Pressure p, cumulative production Np/Gp/Wp, cumulative injection Gi/Wi, etc.

■ Volve_Production_Data_Processed.csv: monthly data, Jan 2007–Dec 2016 (112 rows). Lets you follow production and pressure trends over a longer period

■ volve_input_to_mbal.csv: input data formatted for material-balance analysis (Sep 2007–Dec 2016, 112 rows). Holds the same family of variables as above in model-input form

Research/References/Volve/README.md

Structure of the Stored Data (2): Per-Well Bottomhole Pressure

■ Volve_BHFP_data.csv: daily bottomhole flowing pressure for 7 wells (15/9-F-1C, F-4, F-5, F-11, F-12, F-14, F-15D), 15,634 rows in total (2007–2016)

■ The AVG_DOWNHOLE_PRESSURE column has about 42.6% missing values, but broken down by well the two water-injection wells (15/9-F-4, F-5) are 100% missing while the remaining five production wells have almost none — an ideal case study for examining the structure of missingness (well attributes)

■ Volve_BHP_15_9-F-1C.csv: excerpt for well 15/9-F-1C only (746 rows, Jan 2014–Dec 2016). Used as an introductory material to start with visualizing a single well

Research/References/Volve/README.md

Roles of the Five Files

Production / PVT	Per-well bottomhole pressure	MBAL input
Volve_production.csv	Volve_BHFP_data.csv	volve_input_to_mbal.csv
Volve_Production_Data_Processed.csv	Volve_BHP_15_9-F-1C.csv	

E11: How to Load and Preprocess the Data

■ Start with a small file such as Volve_BHP_15_9-F-1C.csv (one well, 746 rows), load it with `pandas.read_csv`, and get used to the operations

■ Convert the DATEPRD/Date column to datetime with `pd.to_datetime`. Note: Volve_Production_Data_Processed.csv uses DD/MM/YYYY notation, so without `dayfirst=True` the monthly data is misread as pseudo-daily data (confirmed in actual testing)

■ Check missing values such as AVG_DOWNHOLE_PRESSURE — by well, actual-data testing revealed a structural bias in which the two water-injection wells (15/9-F-4, F-5) are 100% missing while the other five have almost none

■ Group by well using the NPD_WELL_BORE_NAME column and organize the data into per-well DataFrames

■ A numeric "0" can also be a physically impossible missing/anomalous value (e.g., well shut-in periods). Don't rely on `isna()` alone — actually plot the data and check visually

E12: Feature Organization

■ Candidate inputs: ON_STREAM_HRS (operating hours), well, elapsed days/period

■ Candidate outputs: AVG_DOWNHOLE_PRESSURE (bottomhole pressure), monthly cumulative production Np, Gp, Wp

■ Experience organizing simple input–output relationships such as "operating hours / elapsed time → bottomhole pressure" as a foundation for surrogate-model building in Session 2

E13: Visualization (Exploratory Data Analysis)

■ Create time-series plots of bottomhole pressure by well using matplotlib/seaborn

■ Visualize the missing intervals to see which wells and periods lack data

■ Overlay the pressure trends of multiple wells (7) to compare differences in behavior between wells

■ Use visualization to check not only for NaNs but also for physically impossible "0" values (e.g., well shut-in periods) — these were found in several wells during actual-data testing

Verification Notes: What We Learned by Actually Running the Exercise

■ Following this material, we actually ran through Ex.1 end-to-end on the Volve data and recorded it as Research/Deliverables/DataDrivenSimulator_S1_Exercise_notebook.ipynb

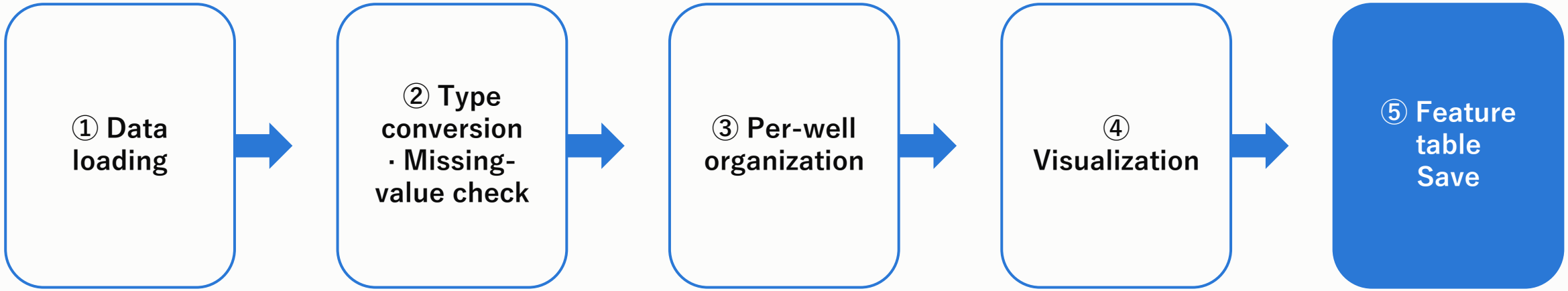
■ Insight 1 (preprocessing): dates in Volve_Production_Data_Processed.csv use DD/MM/YYYY notation. Forgetting dayfirst=True breaks the monthly data into pseudo-daily data

■ Insight 2 (structure of missingness): the 42.6% missing bottomhole-pressure values are due to 100% concentration in the two water-injection wells (15/9-F-4, F-5) — structural missingness that simple interpolation cannot handle

■ Insight 3 (spurious zeros): missingness appears not only as NaN but also as physically impossible "0" values (shut-in periods in several wells), so visual EDA is indispensable

Research/Deliverables/DataDrivenSimulator_S1_Exercise_notebook.ipynb (verification record)

Proposed Structure of the Exercise Notebook



List of References (Sources) (1/2)

■ Equinor, 'Volve field data set' https://www.equinor.com/energy/volve-data-sharing

■ energistics.org, "Equinor's Volve Field Test Data" https://energistics.org/equinors-volve-field-test-data

■ yohanesnuwara/pyreservoir (data redistribution source) https://github.com/yohanesnuwara/pyreservoir/tree/master/data/volve

■ Research/References/Volve/README.md (details of stored data, sources, license notes)

■ Research/Deliverables/DataDrivenSimulator_S1_Exercise_notebook.ipynb (record of verifying this material's content on real data)

■ Anaconda official documentation https://docs.anaconda.com/anaconda/install/

■ Jupyter official documentation https://jupyter.org/install

■ scikit-learn official documentation https://scikit-learn.org/stable/

List of References (Sources) (2/2)

- matplotlib official documentation https://matplotlib.org/