

Matrix Shuffler

Andrej Knaus, Laura Thaci, and Esma Karic

30 Jun 2025

Abstract

Matrix Shuffler is an interactive web-based tool designed to facilitate the visual exploration of matrix-structured datasets through manual and algorithmic reordering. Inspired by Jacques Bertin's reorderable physical matrices, the tool enables users to uncover hidden patterns, clusters, and anomalies by rearranging rows and columns. Matrix Shuffler supports multiple encoding options, advanced sorting algorithms, and intuitive drag-and-drop interactions within a clean, browser-based interface. Its open-source architecture emphasizes accessibility, extensibility, and performance, making it a versatile resource for researchers, data analysts, and educators across disciplines.

© Copyright 2025 by the author(s), except as otherwise noted.

This work is placed under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence.

Contents

Contents	i
List of Figures	iii
1 Introduction	1
2 Motivation	3
3 Overview and Features	5
3.1 Core Functionality	5
3.2 Technology Stack	5
3.3 Dataset Management	5
3.4 Matrix Operations	6
3.4.1 Manual Reordering	6
3.4.2 Algorithmic Sorting	6
3.4.3 Matrix Utilities	8
3.4.4 Visualization and Encoding	8
3.5 Export and Sharing	10
4 User Interface and Interaction	11
4.1 Data Panel	11
4.2 Visualization Panel	12
4.3 Settings Panel	12
4.3.1 Normalization	13
4.3.2 Label Rotation	13
4.3.3 Cell Size	13
4.3.4 Visualization Colors	14
4.3.5 Color Scheme Presets.	14
4.3.6 Matrix Sorting	14
4.3.6.1 Algorithmic Solutions	14
4.3.6.2 Statistical Sorting	15
4.3.6.3 Similarity Sorting	15
4.4 Menu Bar	16
5 Future Work	19
6 Concluding Remarks	21
Bibliography	23

List of Figures

1.1	Township Dataset Before Reordering	2
1.2	Township Dataset After Reordering.	2
3.1	Matrix Shuffler: Data Panel	6
3.2	European Values Dataset: Standard Color Encoding.	7
3.3	European Values Dataset: After 2D Sort	7
3.4	European Values Dataset: After Greedy Seriation	8
3.5	European Values Dataset: Circle Encoding	9
3.6	European Values Dataset: Hybrid Encoding	9
4.1	Matrix Shuffler: User Interface	12
4.2	Matrix Shuffler: Tooltip Display	12
4.3	Matrix Shuffler: Settings Panel	13
4.4	Matrix Shuffler: Color Scheme Presets	14
4.5	Matrix Sorting: Statistical Sorting	15
4.6	Matrix Sorting: Similarity Sorting	16
4.7	Matrix Sorting: Similarity Sorting Example	17

Chapter 1

Introduction

Matrix Shuffler [Knaus et al. 2025b; Knaus et al. 2025a] is an open-source web application that builds on Jacques Bertin’s concept of reorderable matrices [Bertin 1983] to enhance the visual exploration of structured data. By allowing users to rearrange rows and columns, the tool helps uncover patterns and structures that are often obscured in default data orderings, as shown in Figures 1.1 and 1.2.

Matrix Shuffler offers the same core functionality as Bertin’s physical matrices, the ability to reorder rows and columns, but translated into a modern digital context. Users can interactively reorder matrices either manually, through drag-and-drop operations, or automatically, using built-in algorithmic sorting methods. This process often reveals structures such as clusters, groupings, gradients, and anomalies, which are not easily noticeable in the original order of the data.

Developed as a web-based application, Matrix Shuffler offers cross-platform compatibility and is distributed under the MIT license, making it completely open-source. The tool combines a clean and user-friendly interface with an efficient rendering engine, capable of handling both small and large datasets. It supports tasks ranging from exploratory data analysis to the creation of presentation-quality visualizations.

The key objective is to enable users to interactively explore their datasets, just as Bertin did with physical matrices, but in a more accessible, flexible, and scalable way. By offering an intuitive interface paired with powerful algorithmic tools, Matrix Shuffler lowers the barrier to entry for pattern discovery in tabular data. This empowers users in various domains, including data science, research, and education, to uncover insights that might otherwise remain hidden.

Additionally, the tool is designed to be easily extensible. Its architecture allows for future integration of more advanced sorting algorithms, encoding options, and interaction techniques.

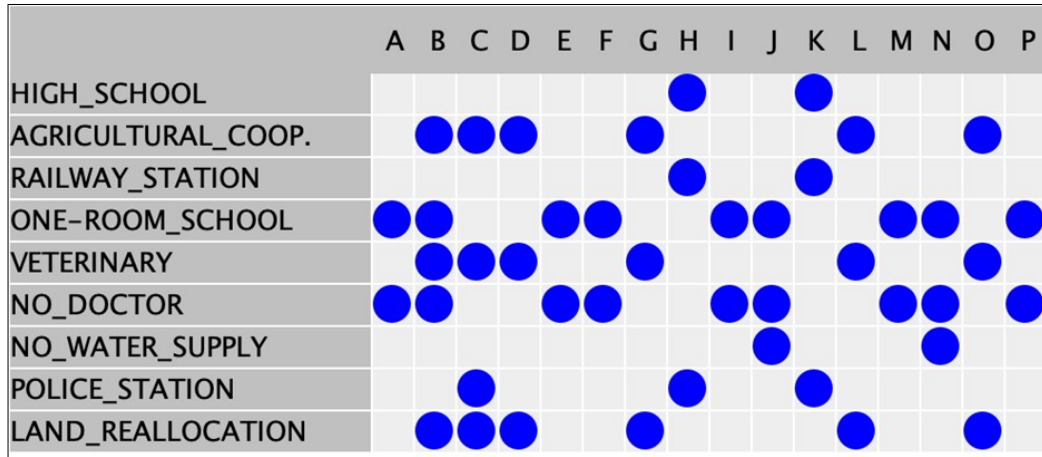


Figure 1.1: Bertin's township dataset before reordering. [Screenshot taken by Andrej Knaus.]

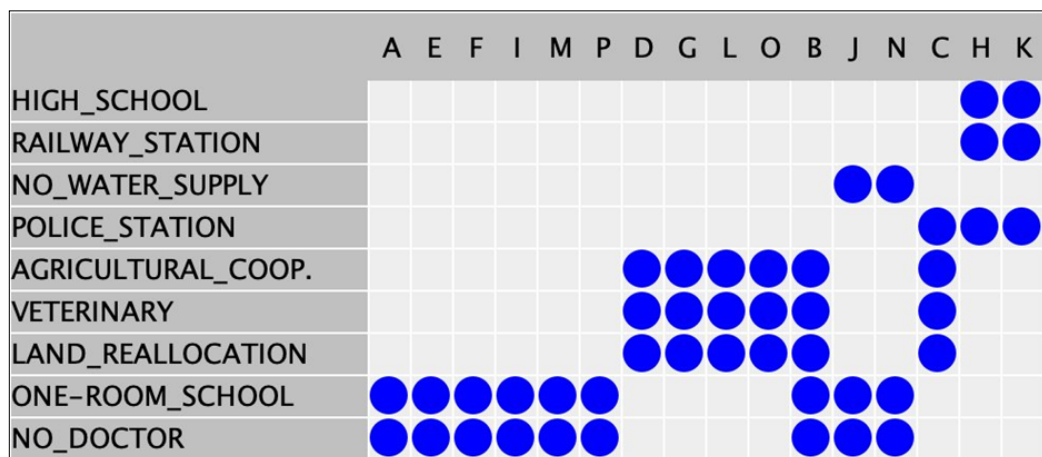


Figure 1.2: Bertin's township dataset after reordering. Groupings and patterns are now clearly visible. [Screenshot taken by Andrej Knaus.]

Chapter 2

Motivation

Large matrix-based data representations are a common occurrence across various fields, including data science, bioinformatics, social network analysis, and machine learning. Despite their widespread use, these datasets are often difficult to interpret in their raw form. The default order of rows and columns typically reflects data entry or arbitrary indexing, rather than any meaningful structure. As a result, interesting patterns or insights may remain hidden.

Reordering rows and columns transforms the matrix from a chaotic collection of values into a structured, interpretable visualization. This simple yet powerful technique often reveals underlying patterns that are otherwise invisible. For example, clusters of similar entities become visible as blocks along the diagonal, or patterns of correlation may emerge in a reordered similarity matrix.

This motivation is not new, it draws heavily from the pioneering work of Jacques Bertin, who demonstrated with his physical reorderable matrices that order is itself a form of visual encoding. The act of reordering does not change the data, but transforms the ability of users to perceive structure within it.

In modern data analysis workflows, this principle remains highly relevant. Yet, most visualization tools either overlook matrix reordering or limit it to static, algorithmic clustering results. Matrix Shuffler addresses this gap by allowing users to reorder matrices both manually and with algorithmic methods.

The goal is to make matrix reordering accessible, fast, and interactive, enabling users to explore their datasets visually, uncover hidden patterns, validate hypotheses, and ultimately derive deeper insights — without requiring specialized technical skills or manual data preprocessing.

Chapter 3

Overview and Features

Matrix Shuffler is designed to support interactive matrix reordering through both manual adjustments and algorithmic operations. Additional features focus on dataset management, flexible visualization options, and export capabilities.

3.1 Core Functionality

Matrix Shuffler’s core functionality supports interactive exploration of matrix-based data. These functionalities can be grouped into four main categories: dataset management, matrix operations, visual encoding, and export. The application allows users to load, rename, and manage multiple datasets during a session. A key feature is the ability to reorder rows and columns, either manually by dragging or automatically using sorting algorithms such as 2D Sort and Greedy Seriation. The Visualization Panel provides several encoding options, including color gradients and circle-based representations, as well as hybrid combinations of both. Users can further adjust the visualization through label rotation, matrix transposition, and normalization. To support reporting and further analysis, Matrix Shuffler includes export functionality for both the visualizations (PNG, SVG) and the processed datasets (CSV, TSV).

3.2 Technology Stack

Matrix Shuffler is implemented as a modern web application. The frontend is developed in TypeScript with Vue.js [Vue.js 2025], providing a robust and reactive framework for managing user interactions and application state. For rendering the matrix visualization, the application uses PixiJS [PixiJS 2025], a high-performance 2D rendering engine that uses WebGL for GPU acceleration. This choice enables smooth interactions. Application state, including dataset management, normalization settings, and user preferences, is handled using Pinia, a Vue-compatible state management library. Build tooling is managed with Vite, which provides fast hot-module reloading and optimized production builds, alongside PNPM for efficient dependency management. The entire application is open-source and released under the MIT license, encouraging contributions, modifications, and reuse by the wider community. This architecture ensures that Matrix Shuffler is both highly accessible—running directly in a browser without installation—and easily extendable for future enhancements.

3.3 Dataset Management

Matrix Shuffler offers flexible dataset handling to support iterative exploration. Users can import datasets in CSV or TSV formats through the Import Data option. Upon import, datasets are stored in a session-based dataset manager, which allows switching between previously loaded datasets without needing to re-upload files. The dataset manager uses the local storage of the browser, which dependent on the browser is limited to 5MB to 10 MB. Figure 3.1 shows the appearance of the Data Panel.

	Belgium	Czech Republic	Denmark	Finland	France	Germany	Greece	Italy	Norway	Poland	Portugal	Russia	Spain	Sweden	United Kingdom
Household income	26874	16957	24682	25739	28310	28799	20440	24216	31459	15371	19366	15286	22847	26242	26904
Women's suffrage date	1948	1920	1915	1906	1944	1918	1952	1945	1913	1918	1976	1918	1931	1921	1928
Against cohabitation without marriage	12	42	4	18	8	20	30	46	12	39	17	39	16	6	19
Belief in God	61	36	63	69	52	63	93	91	56	96	86	77	76	46	65
Confidence in Government	32	21	55	42	34	29	22	28	51	23	30	60	35	54	19
Confidence in the	50	34	72	83	73	58	70	75	57	63	75	73	57	41	89

Figure 3.1: Matrix Shuffler: The Data Panel display the dataset in tabular form.

Each dataset can be renamed for clarity, and users can remove datasets that are no longer needed. The application also maintains the order of loaded datasets, ensuring users can easily return to earlier configurations. Additionally, a sample dataset is provided to quickly demonstrate the application's functionality without requiring initial data preparation.

3.4 Matrix Operations

The core functionality of Matrix Shuffler revolves around matrix reordering. Users can interactively adjust row and column order to reveal patterns and clusters that are difficult to detect in raw matrix forms.

3.4.1 Manual Reordering

Users can manually reorder rows and columns by dragging cells directly in the visualization. This tactile interaction mimics the physical reorderable matrices pioneered by Jacques Bertin and supports outlier detection, and cluster identification.

3.4.2 Algorithmic Sorting

Matrix Shuffler provides two primary algorithmic sorting methods:

- **2D Sort:** Sorts rows and columns independently based on summary statistics. Users can choose metrics such as sum, mean, variance, maximum, minimum, or median. Each axis can be sorted in either ascending or descending order. An example of a dataset before and after 2D sort is shown in Figures 3.2 and 3.3 respectively.
- **Greedy Seriation:** An ordering algorithm based on pairwise similarity. It iteratively builds an ordering that minimizes the distance between adjacent rows (or columns), making clusters and continuous patterns more apparent. An example of a dataset before and after greedy seriation is shown in Figures 3.2 and 3.4 respectively.

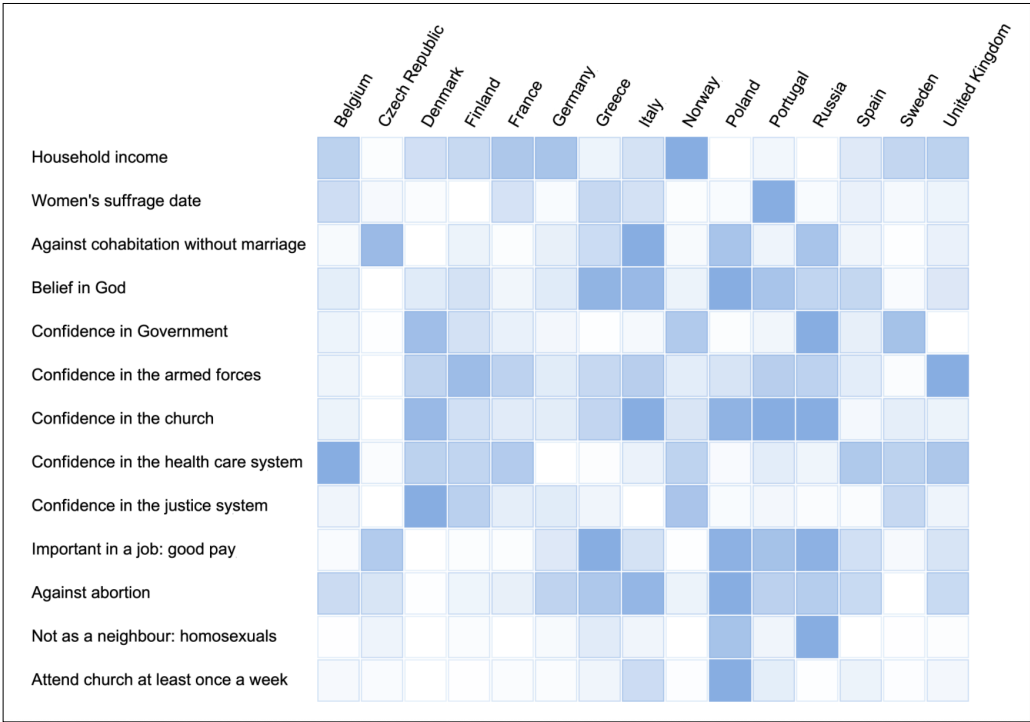


Figure 3.2: European Values Dataset: Standard color encoding.

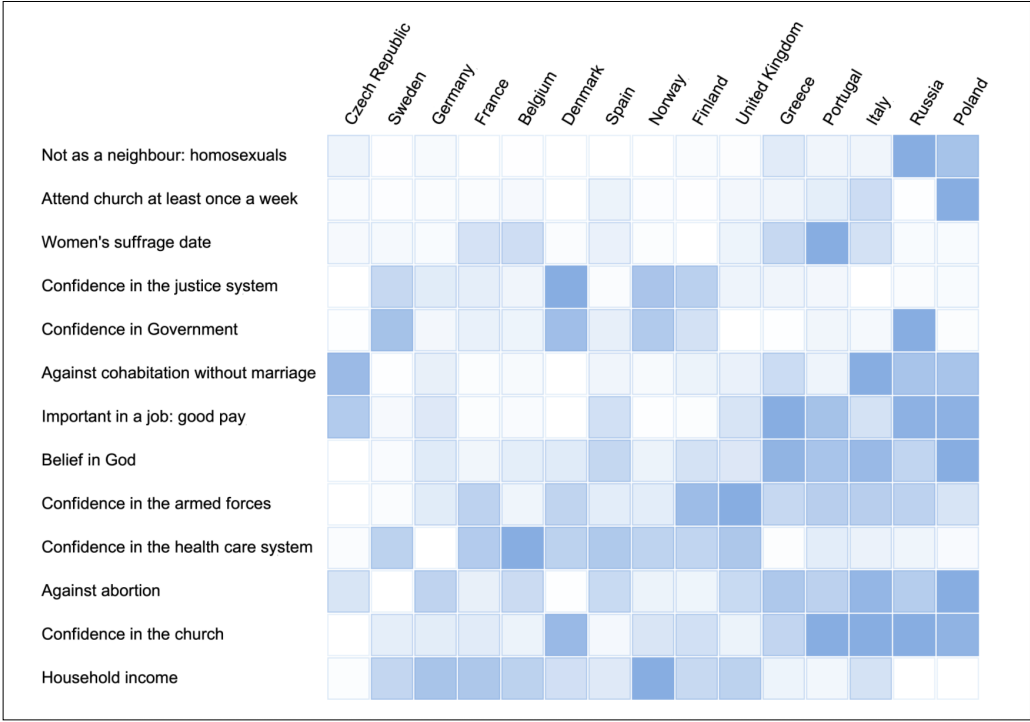


Figure 3.3: European Values Dataset: After 2D Sort.

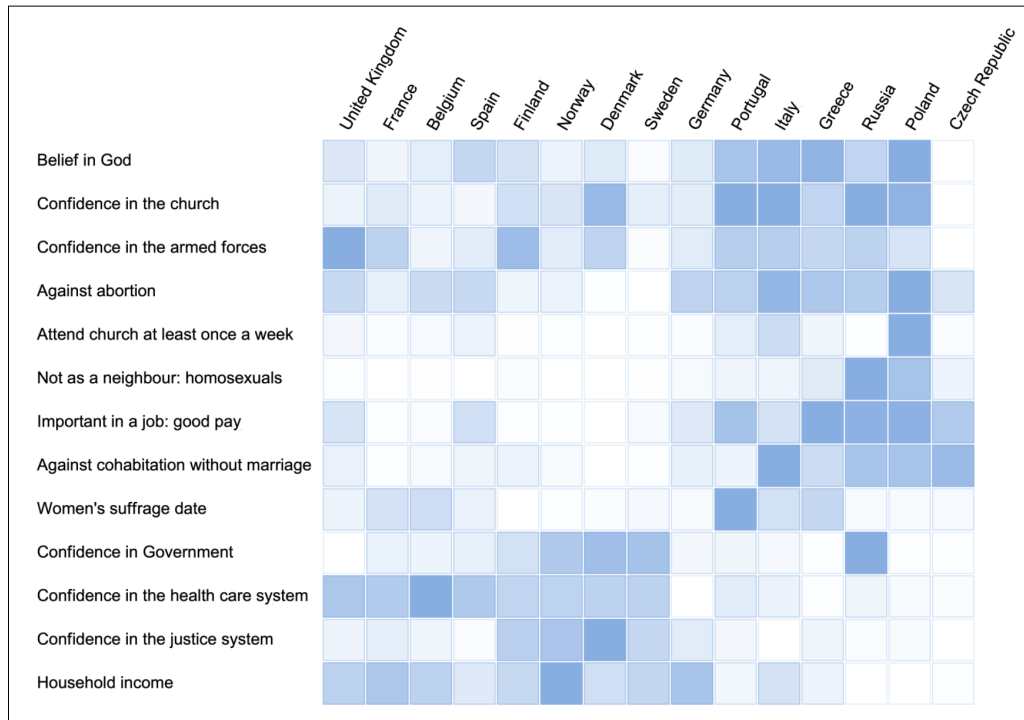


Figure 3.4: European Values Dataset: After Greedy Seriation.

3.4.3 Matrix Utilities

Additional matrix manipulation tools include:

- *Transpose*: Flips rows and columns.
- *Normalization*: Adjusts data scales to improve visual interpretability. Options include None, Row-wise, Column-wise, and Global normalization.
- *Reset*: Restores the matrix to its original input order.

3.4.4 Visualization and Encoding

Matrix Shuffler offers multiple encoding options that enhance the readability of matrix data:

- *Color Encoding*: Cells are filled with a color gradient representing the magnitude of values, as shown in Figure 3.2. Users can adjust minimum and maximum colors or select from preset color schemes.
- *Circle Encoding*: The magnitude of values is represented by circle size within each cell, as shown in Figure 3.5.
- *Hybrid Encoding*: Combines color with circle or text, providing redundant encoding for clarity, as shown in Figure 3.6.

Label readability is improved with adjustable label rotation (from 0 to 90 degrees), allowing users to adapt the matrix layout depending on data density. The interface supports zooming, panning, and resizable panels to facilitate exploration of large datasets.

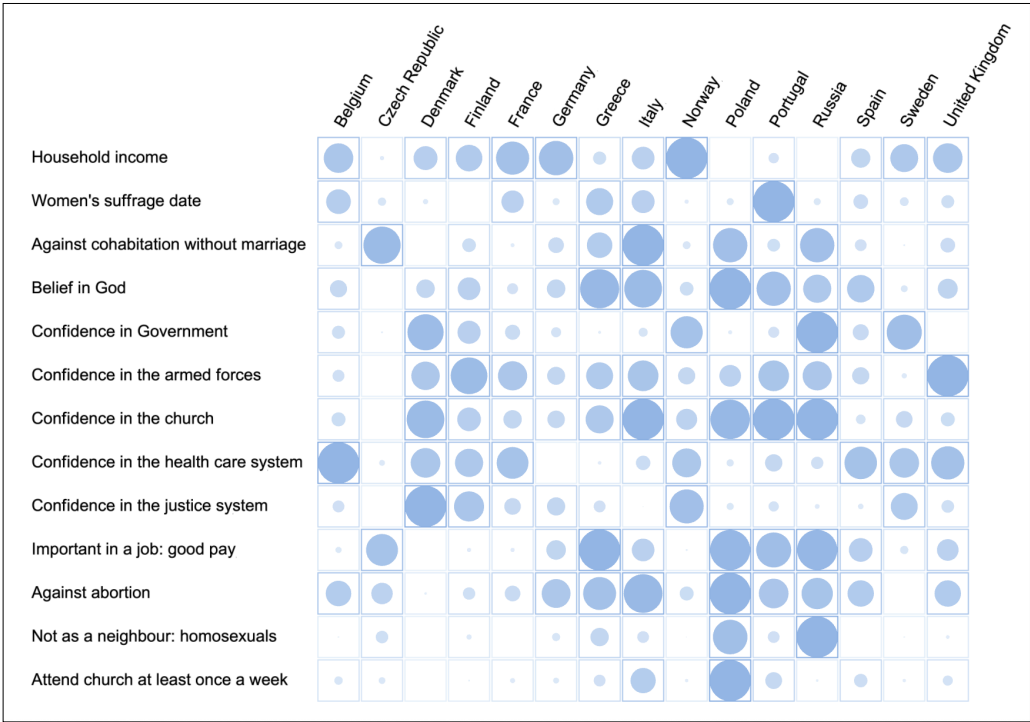


Figure 3.5: European Values Dataset: Circle encoding.

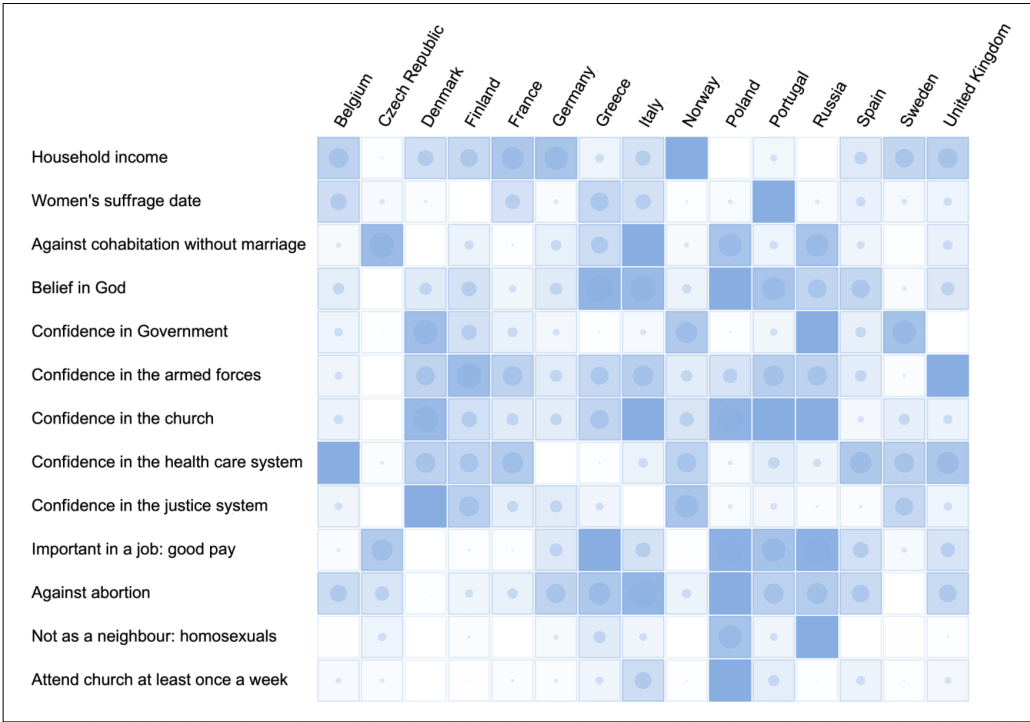


Figure 3.6: European Values Dataset: Hybrid encoding.

3.5 Export and Sharing

Matrix Shuffler provides export functionality to support documentation, presentation, and downstream analysis. Users can export the reordered dataset in CSV format, preserving the current row and column order. This allows for consistent data output aligned with the visual exploration.

The matrix visualization can also be exported in two formats: PNG and SVG. For PNG export, the current canvas content is extracted and downloaded as a raster image, providing a quick and high-quality snapshot suitable for presentations. The SVG export, on the other hand, reconstructs the entire matrix visualization using the current dataset and configuration settings. Each visual element is generated through the `generateMatrixSVG` function, ensuring that the exported graphic is fully scalable.

These export options make it easy to share, document, and reuse insights discovered through matrix reordering, enhancing both collaborative and individual analytical workflows.

Chapter 4

User Interface and Interaction

The user interface of Matrix Shuffler is designed to facilitate intuitive exploration of matrix data through a combination of manual and algorithmic interactions. It is shown in Figure 4.1. The application is structured into four main components:

- **Data Panel (Left Sidebar):** Manages datasets, including importing, renaming, deleting, and switching between datasets. This panel is resizable via a drag handle, allowing users to adjust its width based on preference or screen size.
- **Visualization Panel (Center):** The main workspace where the matrix is displayed. This view supports direct interactions such as manual reordering, zooming, and panning.
- **Settings Panel (Right Sidebar):** Contains controls for visualization settings, sorting algorithms, encoding options, normalization, and matrix operations. This panel can be toggled open or closed to optimize screen space.
- **Menu Bar (Top):** Provides access to file-related actions (import, export), display modes (encoding changes), matrix operations (reset, apply sorting), and help resources.

4.1 Data Panel

The Data Panel is initially hidden and can be opened either by clicking on the Data icon on the left side or by uploading / selecting a dataset. The opened Data Panel, shown on the left of Figure 4.1 (and in more detail in Figure 3.1), allows users to:

- Access recently loaded datasets within the current session.
- Switch between datasets by clicking on a recent dataset entry. The matrix view updates immediately to reflect the selected dataset.
- Clear the current dataset visualization.
- Load a sample dataset for quick testing or demonstration.
- View the dataset in a table view.

The panel width can be resized using a drag handle, providing flexibility for larger datasets or compact viewing.



Figure 4.1: Matrix Shuffler: User interface with opened Data Panel, and Settings Panel. The Visualization Panel in the center is showing the European Values dataset.

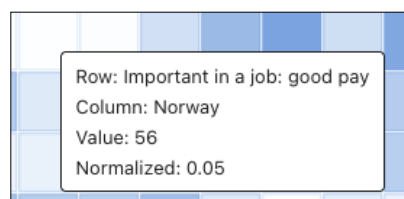


Figure 4.2: Matrix Shuffler: Tooltip displaying detailed information about the hovered cell.

4.2 Visualization Panel

The central Visualization Panel displays the matrix as a grid of cells representing individual data values, accompanied by labeled rows and columns. Users can reorder the matrix interactively by dragging cells. The direction of the initial drag determines whether a row or column is moved. As the selected element is repositioned, the matrix dynamically updates to reflect the new order. Upon release, the element snaps precisely into place. This drag-and-drop mechanism is designed to be smooth and responsive, enabling intuitive pattern discovery through manual exploration.

Additional interactive features include tooltip display and navigation controls. Holding the Alt or Meta (Command) key displays a tooltip containing detailed information about the hovered cell, including both raw and normalized values, as can be seen in Figure 4.2. For navigation, holding the spacebar enables panning, allowing users to move the view freely. Pressing the “r” key recenters the visualization within the viewport.

4.3 Settings Panel

The Settings Panel provides comprehensive controls for adjusting the matrix view, sorting, and appearance, as shown in Figure 4.3. It is designed as the central hub for visualization customization and data manipulation.

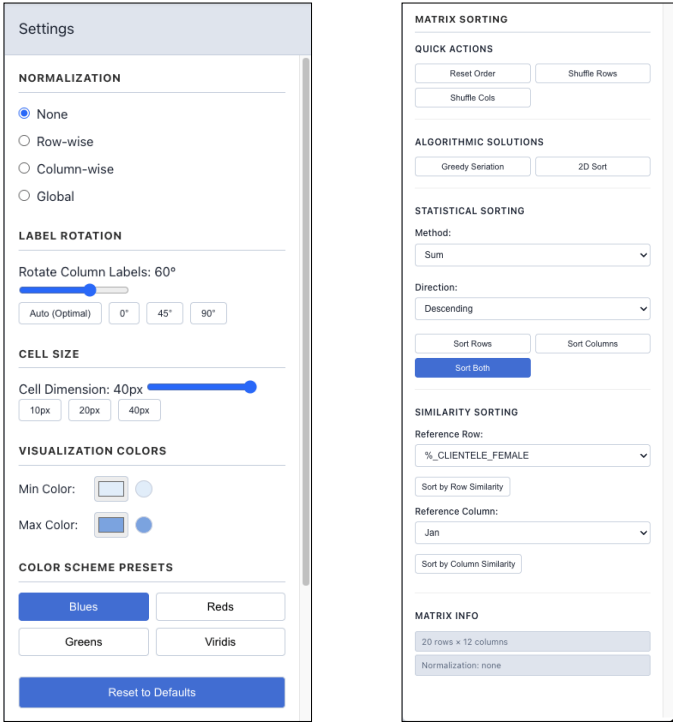


Figure 4.3: Matrix Shuffler: Settings Panel (shown split in two vertically).

4.3.1 Normalization

Users can choose how data values are normalized to make comparisons meaningful:

- None: Raw values are used without modification.
- Row-wise: Each row is scaled independently.
- Column-wise: Each column is scaled independently.
- Global: The entire matrix is normalized based on the global minimum and maximum.

Normalization directly affects how values are represented visually in the matrix.

4.3.2 Label Rotation

Column labels can be rotated to enhance readability, particularly in wide matrices where horizontal space is limited. A rotation slider allows users to adjust the angle continuously between 0° and 90°. Additionally, several preset options (0°, 45°, 90°, and an Auto (Optimal) mode) are provided for quick selection. This feature is essential for preventing label overlap and maintaining legibility in dense visualizations.

4.3.3 Cell Size

Matrix Shuffler allows users to adjust the cell size of the visualization, which is especially useful when working with large datasets. A slider enables continuous adjustment within a range of 10 px to 40 px, while three predefined options (10 px, 20 px, and 40 px) offer convenient presets. Modifying the cell size also dynamically scales the label size, ensuring a consistent and readable layout.

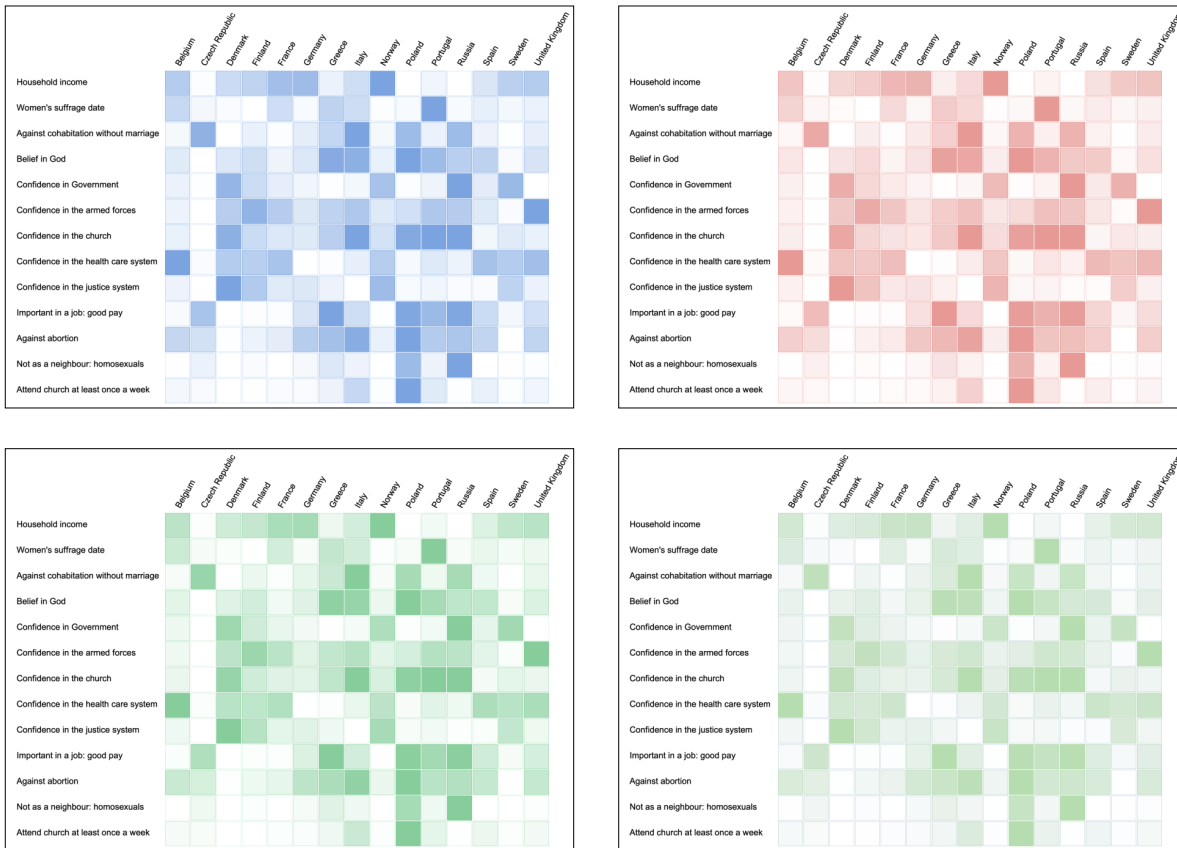


Figure 4.4: Matrix Shuffler: The four color scheme presets: Blues, Reds, Greens, and Viridis.

4.3.4 Visualization Colors

Users can customize the color encoding of the matrix by setting the minimum and maximum values of the color gradient. Any adjustments to these colors are reflected immediately in the visualization, allowing real-time feedback and fine-tuning of the display.

4.3.5 Color Scheme Presets

To simplify the selection of effective color schemes, Matrix Shuffler includes several predefined color palettes: Blues, Reds, Greens, and Viridis, as shown in Figure 4.4. These presets help users quickly choose visually appropriate schemes for their data without manual configuration.

4.3.6 Matrix Sorting

Under the Matrix Sorting section, the user can through various methods automatically reorder the matrix. At the top there are some quick action buttons which either reset the matrix order or randomize it. Below them are powerful sorting tools grouped into three sections.

4.3.6.1 Algorithmic Solutions

The Algorithmic Solutions section offers two fully automated techniques for matrix reordering: *Greedy Seriation* and *2D Sort*. These algorithms aim to uncover latent structure in the data by optimizing the order of rows and columns based on similarity or distributional characteristics:

- *Greedy Seriation*: Reorders matrix rows and columns to highlight underlying patterns by maximizing local similarity relationships. The algorithm proceeds in two main phases. First, a similarity matrix is constructed for each dimension (rows and columns) using Pearson correlation coefficients; each

STATISTICAL SORTING

Method:

Direction:

Figure 4.5: Matrix Sorting: The Statistical Sorting section.

entry $S[i, j]$ in the similarity matrix represents the correlation between elements i and j . Second, a greedy ordering procedure begins with the element having the highest total similarity score (i.e., the sum of its correlations to all others). The algorithm then iteratively selects the next element that maximizes average similarity to all previously selected elements. This approach is particularly effective for exposing clusters or gradual transitions in the data, making it well-suited for visual exploration and pattern recognition tasks.

- **2D Sort:** 2D Sort is an iterative optimization algorithm that simultaneously reorders both matrix dimensions based on distributional weights. The process alternates between row and column reordering until convergence. In each iteration, columns are first normalized to unit sum, and row weights are computed as the sum of these normalized values. Rows are then reordered using a simple bubble sort according to their weights. The same procedure is repeated with rows normalized and column weights calculated, followed by column reordering. This loop continues until no further changes occur or a maximum of 500 iterations is reached. 2D Sort is effective in cases where the goal is to arrange both rows and columns according to their relative “mass” or importance in the dataset. It is particularly useful for enhancing visual symmetry and structure in heatmaps or matrix-based visualizations.

4.3.6.2 Statistical Sorting

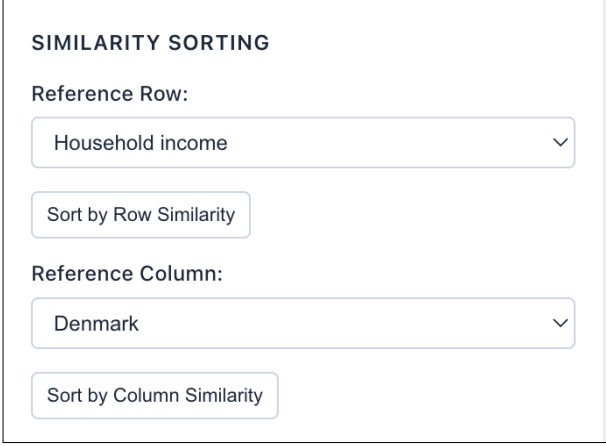
In the Statistical Sorting section, shown in Figure 4.5, users can reorder the matrix based on common statistical functions applied to its rows or columns. The available metrics include sum, mean, median, maximum, minimum, and variance. Users may specify the sorting direction (either ascending or descending) and can apply the selected sorting operation to rows, columns, or both simultaneously.

This feature provides a simple yet effective way to structure the matrix according to dominant statistical properties, facilitating rapid identification of high- or low-value entities and improving the interpretability of the visualization.

4.3.6.3 Similarity Sorting

In the Similarity Sorting section (see Figure 4.6), users can reorder rows or columns based on their similarity to a selected reference row or column. This operation is particularly effective for emphasizing local clusters and uncovering pattern similarities relative to a user-defined point of interest.

To perform similarity sorting, users first select a reference row or column. The application then reorders the corresponding axis by computing similarity scores between the reference and all other elements. Rows or columns are then arranged in order of decreasing similarity to the reference. An



The screenshot shows a section titled "SIMILARITY SORTING". It contains two main controls: "Reference Row:" and "Reference Column:". The "Reference Row:" control has a dropdown menu with "Household income" selected and a "Sort by Row Similarity" button below it. The "Reference Column:" control has a dropdown menu with "Denmark" selected and a "Sort by Column Similarity" button below it.

Figure 4.6: Matrix Sorting: The Similarity Sorting section.

example of this operation is shown in Figure 4.7, where rows are sorted based on similarity to the *Household Income* row, and columns based on their similarity to *Denmark* within the European Values dataset.

This technique helps surface context-specific patterns and comparisons that may not be evident through global ordering methods.

4.4 Menu Bar

The final component of the application interface is the Menu Bar, located at the top of the screen. It provides access to key file operations, view settings, actions, and help resources through four main dropdown menus:

- **File:** Allows users to import their own datasets or load one of the available example datasets. Additionally, it supports exporting the current dataset as a CSV file, or saving the matrix visualization as either a PNG image or a scalable SVG graphic.
- **View:** Provides options to switch between different encoding styles, enabling users to choose the most appropriate visual representation for their data.
- **Actions:** Includes utilities such as resetting the dataset to its original ordering and transposing the matrix.
- **Help:** Offers guidance on using the application, including access to documentation and an About item describing the tool.

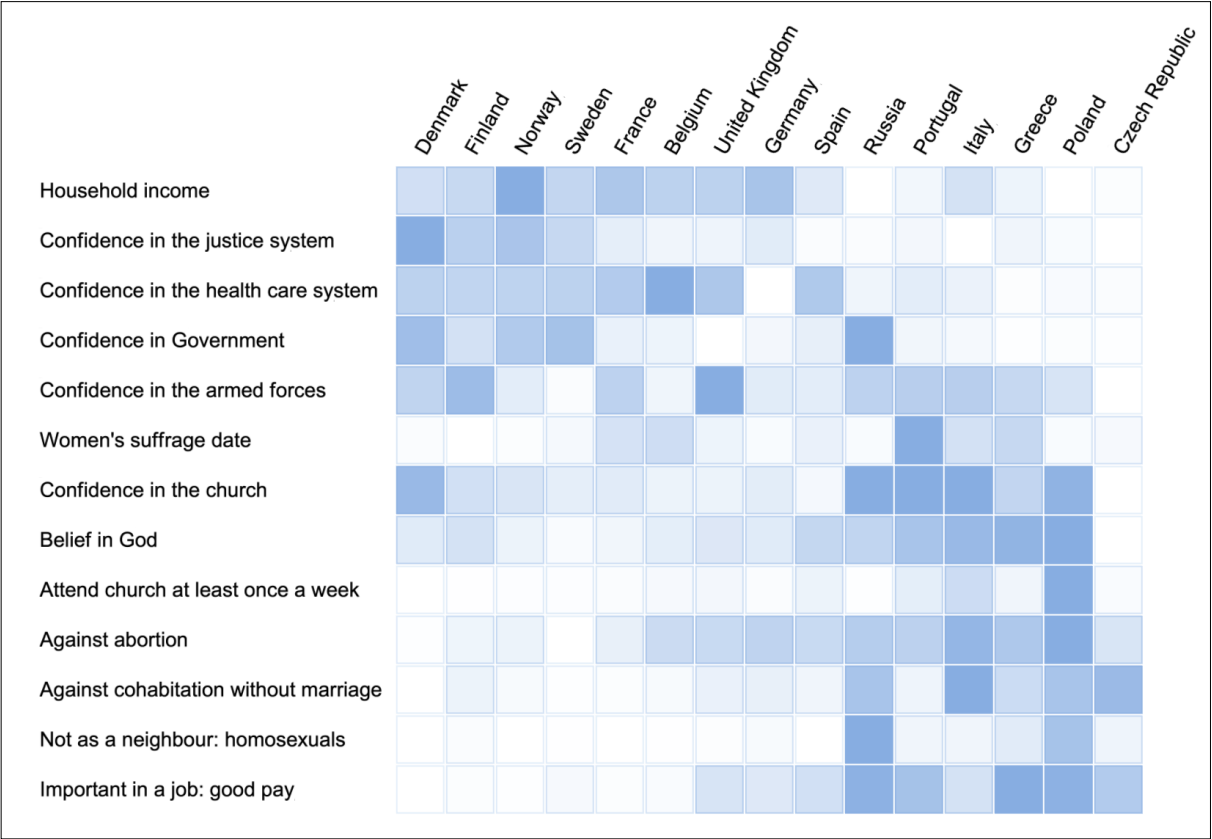


Figure 4.7: Similarity sorting example with the European Values dataset. Rows are sorted by *Household Income* and columns by *Denmark*.

Chapter 5

Future Work

While Matrix Shuffler already offers a comprehensive set of features, several directions for future enhancement have been identified:

- *HTML Label Overlays*: Replace PixiJS text elements with HTML-based overlays to improve text clarity. The current rendering can result in blurry or pixelated labels, particularly on high-resolution displays. HTML elements would ensure sharper, more readable text across platforms.
- *Glue Functionality*: Add support for merging or fixing rows and columns during reordering, inspired by the functionality available in Bertifier [Perin et al. 2014]. This would allow users to preserve specific data relationships while exploring other structures.
- *Additional Encoding Styles*: Introduce new visual encoding methods, including the original Bertin-style texture and pattern encodings. These alternatives would increase the expressiveness and flexibility of matrix representations beyond color and size alone.
- *Expanded Algorithmic Solutions*: Integrate advanced matrix reordering techniques such as BiMax and multidimensional scaling (MDS). These algorithms would support more diverse use cases and uncover complex structures in data.
- *Sticky Labels*: Introduce sticky row and column labels that remain fixed to the left and top edges of the canvas, respectively, when panning. This behavior would be optionally toggleable and would function similarly to the "freeze" feature in spreadsheet applications like Excel. It would significantly enhance usability when navigating large datasets by preventing label loss during exploration.
- *Performance Optimization for Large Datasets*: Enhance rendering efficiency and scalability. Although PixiJS provides GPU-accelerated graphics, it has some limitations drawing sharp text and lines. Alternative technologies such as [three.js](#) [Three.js 2025](#) or [WebGPU](#) [W3C 2025](#) could offer improved performance and rendering capabilities.
- *Enhanced Navigation*: Implement features such as zooming and a minimap to facilitate exploration of large or complex matrices. These tools would improve usability and user orientation within the visualization.

These enhancements would increase the power, flexibility, and user-friendliness of Matrix Shuffler, making it a more robust tool for matrix visualization and exploratory data analysis.

Chapter 6

Concluding Remarks

Matrix Shuffler is an open-source web application for matrix reordering. Inspired by Jacques Bertin's physical reorderable matrices, the tool lets users explore tabular data through reordering, revealing patterns, clusters, and anomalies that are often hidden in raw matrix views.

The application has a clean and responsive user interface. Key features include manual reordering, algorithmic sorting, statistical sorting, similarity-based sorting, and customizable visualization options that let users adapt the view to their specific needs.

By providing both flexibility and ease of use, Matrix Shuffler helps users gain deeper insights into their data, whether for exploratory analysis, pattern recognition, or visual storytelling.

Bibliography

- Bertin, Jacques [1983]. *Semiology of Graphics: Diagrams, Networks, Maps*. Originally published in French as *Sémiologie Graphique* in 1967. University of Wisconsin Press, 31 Dec 1983. ISBN 0299090604 (cited on page 1).
- Knaus, Andrej, Esma Karic and Laura Thaçi [2025a]. *Matrix Shuffler Demo*. 30 Jun 2025. <https://andrejknaus.github.io/matrix-shuffler/> (cited on page 1).
- Knaus, Andrej, Esma Karic and Laura Thaçi [2025b]. *Matrix Shuffler Repository*. 30 Jun 2025. <https://github.com/AndrejKnaus/matrix-shuffler> (cited on page 1).
- Perin, Charles, Pierre Dragicevic and Jean-Daniel Fekete [2014]. *Bertifier: An Interactive Reorderable Matrix Visualization Tool*. 2014. https://aviz.fr/bertifier_app (cited on page 19).
- PixiJS [2025]. *PixiJS*. 30 Jun 2025. <https://pixijs.com/> (cited on page 5).
- Three.js [2025]. *Three.js – JavaScript 3D Library*. 30 Jun 2025. <https://threejs.org/> (cited on page 19).
- Vue.js [2025]. *Vue.js: The Progressive JavaScript Framework*. 30 Jun 2025. <https://vuejs.org/> (cited on page 5).
- W3C [2025]. *WebGPU*. 2025. <https://w3.org/TR/webgpu/> (cited on page 19).