Project 2 – Exploratory visualization

PERE-PAU VÁZQUEZ & OSCAR ARGUDO & IMANOL MUÑOZ

Information Visualization | 2025 - 2026

1. Objective

The goal of the project is to create an exploratory visualization using Altair and Streamlit to analyze the all NSF grants from the last 5 years, and the NSF terminated from the Trump administration.

You have to find data for the NSF grants of the last 5 years (there are many sources and ways you can get them) and let the user answer the following questions with an exploratory tool.

- **Q1:** How are the grants distributed by states every year?
- **Q2:** How are the grants distributed per directorates? And for a certain year?
- **Q3:** Are the cancelled grants especially hitting a certain directorate?
- **Q4:** How have the total grants amount evolved over the years?
- **Q5:** For a selected state, how have the grants evolved? Are there cancelled grants?
- **Q6:** Select some attribute that has not been mentioned previously (e.g., party governing, population of the state, number of funded institutions in the state…), and let the user interactively explore the information around the attribute to get insights.

The second project is all about exploration, which means that some questions require selection of elements. Therefore, you need to think of the Visualization Mantra: overview, zoom and filter, then, details on demand. For example, Q2 should show an overview (there are many ways beyond simple bar charts to show that), and when a year is asked for, you need to allow the user to pick one year (there are also different ways to address that).

We will value the way the interactions help the users solve the tasks. You also need to provide a text of a maximum of 200 words per question (and for the final visualization too) where you explain the design decisions and how these help the users to answer the questions properly. This should include aspects such as: what type of chart did you select and why, what were the different steps of design process you followed, what changes you applied to improve legibility, reduce clutter, distinguish elements, how would a person answer the question with your chart, what other alternatives did not work. Be concise. You do not need to erase the previous steps in design you followed, they are actually encouraged. But only a final description per task is necessary.

A final visualization is also required, in Google Colab, that includes all questions. In this case, you also have to explain, in a maximum of 200 words, what changes were made to make charts consistent, aesthetically pleasant, etc. The final visualization must be then incorporated in a Streamlit application (https://streamlit.io/).

We will pay special attention on how interactions are designed in the final visualization, e.g., if one year is selected in one chart, all the charts containing time data may (or should) be updated accordingly. Take this into account when thinking and designing the final visualization.

DATA

You need to find it yourselves. Different sources will provide different fields. The field directorate is included both in the previous dataset and in many of the others we have checked, so ensure it is there. Some datasets might require you to transform the data. For example, one source lets you

download all the projects in json format (per year). It contains a single json file per project with a lot of details. You could transform this data to CSV and use the transformation step to erase the fields you do not require.

The datasets will contain more than 5000 rows, therefore you will need to use VegaFusion or other techniques to accelerate the generation. And do not forget to remove all the information you do not need. Creation of specific DataFrames with only the necessary information for each component is encouraged.

DELIVERY INSTRUCTIONS

The work can be implemented in pairs or individually. You have to provide the clean data and the final data.

The delivery must consist on a single ZIP file with a name that includes the authors, that contains the datasets (raw and clean), the Colab file(s) (*ipnyb*), the Python Streamlit application, and optional extra documents if required. The ZIP, Streamlit, and Colab files must be named after the names of the authors. Treat the Colab document as a report, include titles, boldfaces, etc., to make it easier to read.

The deadline for the delivery of this lab project is the 30th of December.

IMPORTANT REMARKS

Besides your performance during the interview, the project grade will consider the number of variables and interactions included in the visualizations. We will value the number of non-trivial tasks (adequately described in the documentation) that can be properly solved with your visualization tool. In this sense, adding extra tasks/questions/attributes to explore is valued positively.

Don't leave the project for the last day or do the minimum amount of work. In case of doubt, ask us whether the current work is enough or needs more effort.

THIS CHECKLIST SERVES AS GUIDANCE FOR YOUR DELIVERY

Global checklist:

- Name the file after the name(s) of the author(s).
- Include the name(s) of the authors also as the first line in your notebook.
- Include the clean data.
- Compress all files in a single zip (do not use RAR or other formats) file.
- Also include the names of the authors in the notebook (e.g., a text cell showing who authored the document) and in the Streamlit app (e.g., in an "About" option).
- Ensure the names of the data files inside the notebook correspond to the ones you deliver.
- Make a single delivery per group.
- Ensure you properly cleaned the data.
- Ensure the code executes without errors: last-minute changes may lead to typos.

- Ensure you include a step-by-step design process (does not need to include all minimal steps, if you prefer, but do not forget to include the discussion on why do you change something).
- Do not mix charts with other technologies, everything must be created using altair.
- Ensure they do not include non-properly cleaned data (e.g., N/A or undefined fields).

Google Colab document. For every chart, you must consider:

- Color blindness (e.g., coding anything just with a red-green palette may be problematic).
- Check the consistency of colors across the whole visualization (same color, same meaning in different charts).
- Do not forget to add meaningful titles, labels, messages if necessary...
- When using colors with opacity different from zero, check the interactions with the other elements (are they visible?).
- Think whether you need to normalize values.

For the Streamlit vis:

- All questions should be solved in a single page (with not much scrolling, e.g., maximum double the available screen on a 15" laptop screen).
- The application should run with a simple "streamlit run <application_name>" in any computer, everything should be available in the same folder or addressed properly.
- Do not forget to include a final visualization that answers all the questions.
- Align things, be consistent. You can make use of both vertical and horizontal alignments to facilitate comparisons.
- Extra questions you answer must also go into the final vis.
- Ensure charts can be visually compared properly (consider changing scales, palettes...)
- Use space cleverly (put related things together and unrelated things far away).
- The higher the number of variables (and questions answered) included, the better.
- If you do not use all your data in your vis, ensure you have filtered it previously, don't force Streamlit to execute with data it is not used.
- **Ensure the default values make sense** (e.g., if including selections, check that the initial configuration of the chart has a default state that is meaningful).
- **Minimize the number of interactions required to solve the problems.**
- **Integrate multiple tasks in a single chart if suitable.**
- **Ensure no interactions produce invalid charts** (e.g., empty charts, charts with N/A values, etc.). Most of this is typically originated from an incomplete data cleaning process.