bead

a provenance tool

Krisztián Fekete Miklós Koren

2025-09-10

The Editor Gives You One Week

You need to:

1. Address reviewer concerns about source data
2. Redo analysis with new data
3. Recreate Figure 1
4. Submit within one week

But the Submission Was Months Ago

- Research submitted months ago
- Team has been improving data cleaning since then
  - Some team members left
- Different statistical methods now
- **First question**: How *exactly* was Figure 1 produced?

Research Results are Functions

$$\text{Figure 1} = \text{code}(\text{data})$$

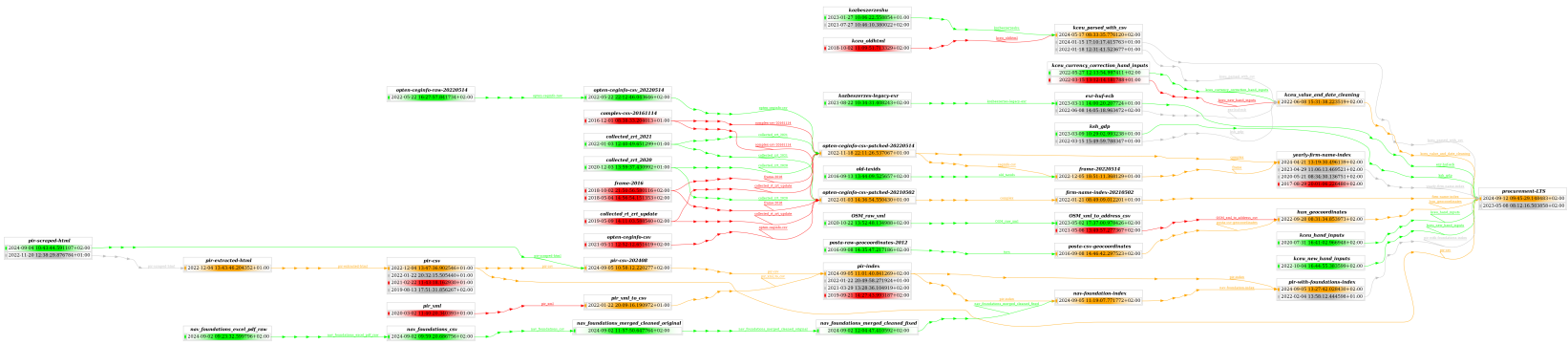
- Results depend on both algorithms and data
- Code under version control (Git)
- Tagged commit at submission
- **But what about the data?**

Data is Also a Function

$$\text{data}_1 = \text{code}_2(\text{data}_2)$$

- Data transformed by wrangling/cleaning steps
  - countries dropped
  - transformations applied
  - feature engineering details
- **Chain of data provenance**

A Real-World Data Pipeline



The Data Provenance Problem

Why It Is Complex:

1. **Frequent changes**: Code and data both evolve
2. **Complex pipelines**: Many steps, multiple datasets
3. **Tool heterogeneity**: Python, R, SQL, DuckDB all in one project
4. **Team dynamics**: People join, leave, change roles

Existing Solutions

Version Control (Git)

- Great for code
- **Not suitable for large binary data**

Data Version Control (DVC)

dvc.org

- Similar spirit to bead, but delivery/versioning focused
- **More complex than needed for provenance tracking**

Orchestration Tools

- Apache Airflow (Python) - airflow.apache.org
- dbt (SQL) - getdbt.com
- KNIME (no-code) - knime.com
- **Too complex for heterogeneous teams**

Enter bead

A command-line tool that captures your data's story, step by step.

- Much simpler than alternatives
- Language agnostic
- Supports heterogeneous teams

What bead Does NOT Do

Not a code runner

- You run your own code
- Python, R, Stata, SQL - doesn't matter

Not a file delivery system

- File system stores your files
- You copy/move files yourself

Only requirement:

- Works with flat files on file system
- Files not too big (20GB works fine)

Core bead Concepts

The bead

- Self-contained unit of computation
- Contains code, *reference* to input data, results
- Packaged as ZIP file
- Remembers exact provenance

Simple Commands

```
bead new my-analysis
bead input add source-data
bead save
```

Demo Time

Inspiration

- Data Journalism
- Human concept of story
  - Characters, setting, events, theme or message
  - Variable scope
  - Stories are composable
  - Helps us to make sense of the world

Design

- ISBN / ISSN
  - Identification of books and series with code
- Task scope
- Local first
- Integrity

Demo

Conference session classification by title

Inputs:

- list of sessions
- session theme classification rules

Output:

- list of sessions extended with theme

Workspace and Box

Output: Plain data

Workspace

- name (*sessions*)
  - input
  - output
  - temp

Calculation 1: CSV to DuckDB

Calculation 2: Match Themes

Versions: Inputs and Improvements

Visual Overview

Visual Overview

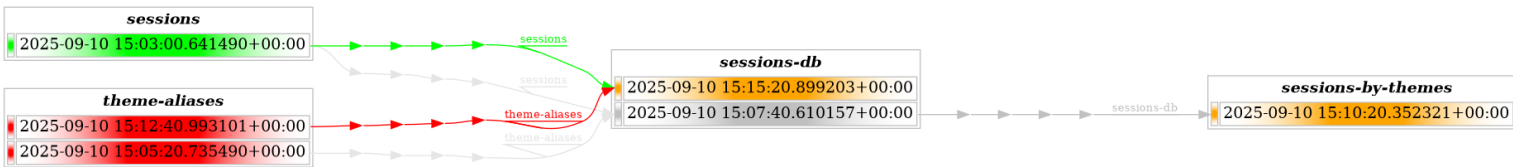


Figure 1: session-themes.png

Internal Details

Internal Details

Similar ideas

Similar ideas

- kaggle notebook
  - https://www.kaggle.com/code#:~:text=New,-Notebook
- nix flake
  - https://nixos.wiki/wiki/Flakes#Flake_schema
- orderly2 (RSECON24 talk)
  - https://youtu.be/IkPgihFQbrk

Source

Source

https://github.com/e3krisztian/bead {{< qrcode https://github.com/e3krisztian/bead qrgithub 300 300 >}}

Case Study From Our Research Lab

Case Study From Our Research Lab

- Used since 2017 in our research groups: CEU MicroData, MACROMANAGERS.eu
  - internally
  - when sharing data with others
- Saved about 600+ beads, two versions on average
- Interquartile range of bead sizes: 10 to 500 MB, largest is 23 GB
- Median time between saving new versions: 51 days

Practices We Adopted

- *Everything* is a bead: raw data, intermediate data, analysis sample, research results
- Never load data directly, from outside a bead
  - Side product: always use relative paths
- We don't often recompute everything, but good to know we *could*

Lessons For Research Software Engineers

- Minimal learning curve for researchers
- No infrastructure requirements
- Works with existing workflows
- Complements version control
- Enables true reproducibility

Key Takeaways

1. **Data provenance is hard** - especially with changing teams
2. **Existing tools too complex** - for heterogeneous research teams
3. **bead keeps it simple** - focuses on one thing well
4. **Reproducibility becomes automatic** - not an afterthought

Contact and Acknowledgements

- Web: bead.zip
- Installation: https://bead.zip/install
- GitHub: github.com/e3krisztian/bead

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 313164). The views expressed are those of the authors and do not necessarily reflect those of the ERC or the European Commission.

References

- **World Development Indicators**: data.worldbank.org/indicator
- **DVC (Data Version Control)**: dvc.org
- **Apache Airflow**: airflow.apache.org
- **dbt**: getdbt.com
- **KNIME**: knime.com