Pollis

Bright Insights through Statistical Learning, Simulation and Optimisation

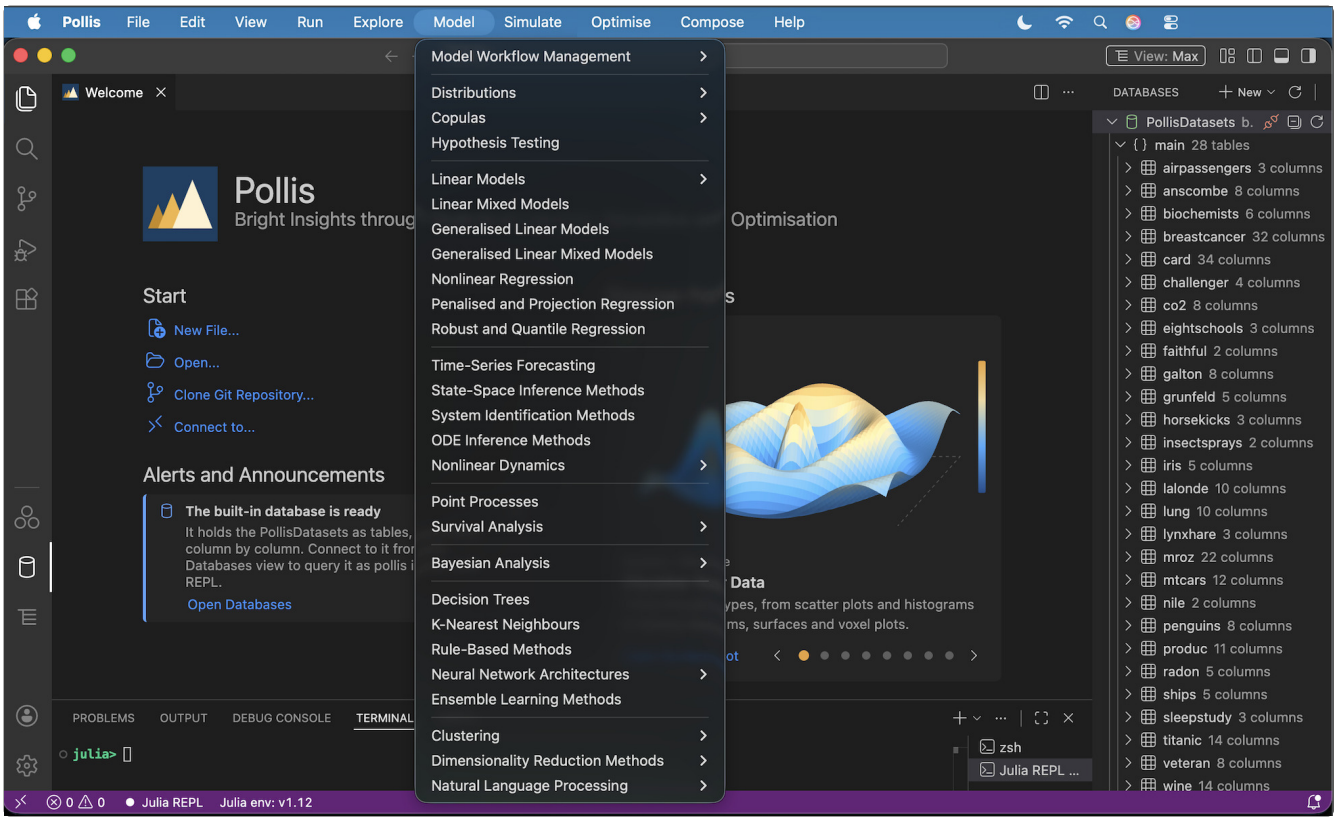
From eureka to evidence.

Had an idea? Pollis helps you test it, in code you own, and then improve and expand it.

Contents

1 From eureka to evidence
2 What Pollis is
3 Inside a panel
4 Teaching built in
5 Coverage
6 Customisation
7 Extensions and Workflows
8 Reports
9 Practice
10 Collaboration
11 Free and open
12 Everything in Pollis

A free, open-source desktop environment to explore data, build models, run simulations and solve optimisation problems: from your first click to your own code.



Start by clicking
Choose a method, fill in a few boxes, run the example.

Keep the code
Every panel writes readable Julia code that is yours.

Learn good practice
Which method, how to check it, where to read more.

All in one place
656 procedures in 203 panels, from statistics to optimal control.

Free · **Open source** (AGPL 3.0) · **Windows, macOS, Linux**

© 2026 Antonio Saragga Seabra · Licensed under CC BY-SA 4.0 · 15 October 2026

WHAT POLLIS IS

Data and a question? Start here.

One desktop application for the whole job, organised by what you want to do.

Pollis is for anyone who has data and a question. You pick a method from a menu, fill in a few boxes, and Pollis gives you working code that produces the result, together with advice on what a careful analyst does next. You do not need to program to start.

Behind the menus are the Julia language and 289 open-source Julia packages. Pollis looks and works like a familiar code editor, with five extra, curated menus: **Explore**, **Model**, **Simulate**, **Optimise** and **Compose**.

Explore

What do my data look like?

Bring in data from finance, economics and AI libraries, Excel, CSV or SQL. Thirty-five plot types, from scatter plots to Sankey diagrams and maps. Descriptive statistics, outliers, missing data.

Model

What explains it?

Regression with robust inference, generalised linear models, time series, survival analysis, Bayesian inference, machine learning, neural networks and text analysis.

Simulate

What could happen?

Monte Carlo and Markov chain Monte Carlo, bootstrap, stochastic differential equations, agent-based models and network models.

Optimise

What is the best decision?

Linear, convex and nonlinear programming, global and Bayesian optimisation, optimal control, Markov decision processes and reinforcement learning.

Compose

How do I put it all together?

Workflows that replicate classic papers, reports, learning activities, new methods as packages, collaboration, coding agents with their skills, and local reasoning models.

5 curated menus

203 method panels

656 procedures

289 open-source Julia packages

Ideas you could test this afternoon

- **Can mild preferences split a whole city?** Neighbours who only want a few people like them nearby. *Agent-based Schelling model.*
- **Does a rumour spread like a virus?** Run an epidemic model on a social network. *Network SIR simulation.*
- **How fast is an outbreak growing right now?** Estimate the reproduction number from daily case counts. *Bayesian renewal model.*
- **Did the interest-rate rise work?** Trace a policy shock through inflation and output. *VAR impulse responses.*
- **Can a computer teach itself to balance a pole?** Watch it learn by trial and error. *Reinforcement learning.*

Who is it for?

Students encountering a method for the first time, who want to see it in action and explore what happens when assumptions change.

Lecturers who want to teach with real, runnable code and the literature behind it.

Researchers who want classical statistics, machine learning, simulation and optimisation without switching tools.

Authors of books and reports who want to illustrate and enhance their work with Pollis, using customised Views and panels: for example, a View that follows the text, chapter by chapter.

Toolboxes add whole fields to the menus, one click to install.

Julia packages do the computing. They are the open-source libraries such as GLM.jl or Turing.jl. Every panel's code calls them, and Pollis installs them into Julia when a panel needs them.

Toolbox extensions are Pollis add-ons. Each one adds a menu of ready-made panels for a field, installed with one click from the Extensions pane. Those panels in turn run on Julia packages. You choose which panels to add, and where.

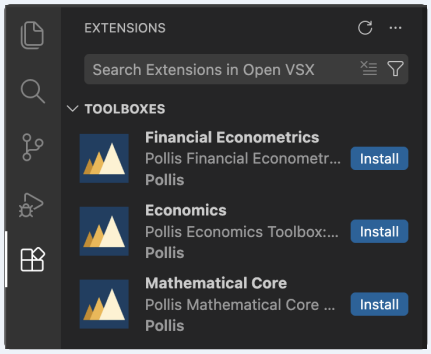
Three are ready from the start:

Financial Econometrics, including ARIMAX, VAR/VECM, GARCH, regime switching and advanced time-series forecasting.

Economics, including DSGE, heterogeneous-agent models, continuous-time macro-finance and game theory.

Mathematical Core, including symbolic maths, linear algebra, differential equations and solvers.

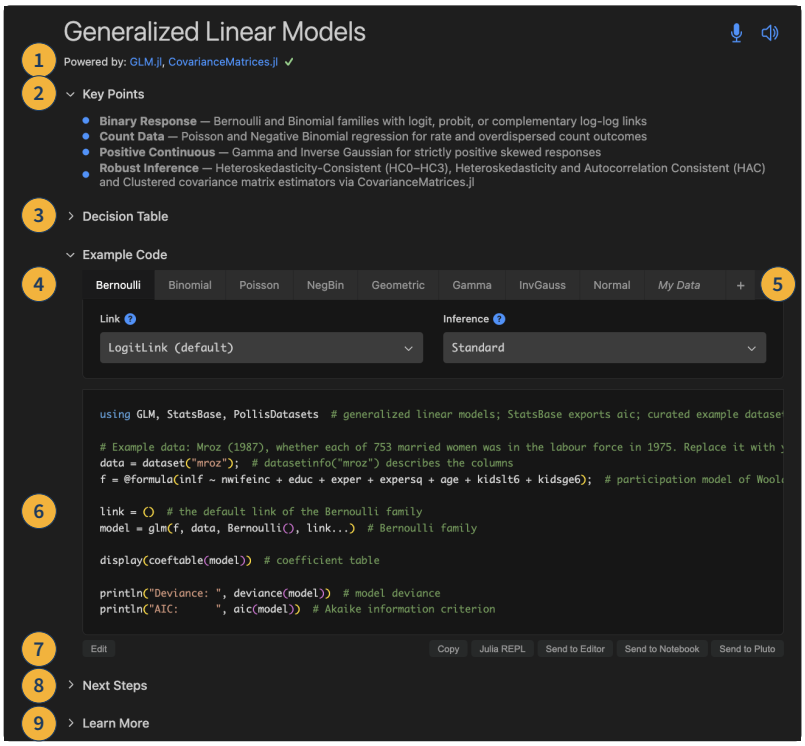
Many more are on the way. Creating a new toolbox is quick and easy: just package existing procedure panels into a ready-to-use collection. Toolboxes make it simple to share your work, extend Pollis, and make your methods available to the wider community.



INSIDE A PANEL

Start by clicking, end with code you own.

Each of the 203 panels brings together one or more methods and their procedures, with everything you need around them.



The Generalised Linear Models panel. Choosing a family, a link or a kind of inference (Standard, HC0–HC3, HAC or Clustered) rewrites the example code.

1 **Powered by.** The open-source Julia packages behind the method (here: GLM.jl and CovarianceMatrices.jl) working together, with links. A tick when they are installed, or a one-click *Install*.

2 **Key Points.** The method in a few lines.

3 **Decision Table.** Which variant to use, and when.

4 **Your choices.** Family, link function and kind of inference: tabs and drop-down lists.

5 **Your own tab.** Click + to add a tab of your own, with your own code.

6 **Example code.** Real Julia code that follows your choices, with comments.

7 **Run it your way.** Edit or copy the code, or send it directly to the Julia REPL, the Editor, or a Jupyter or Pluto notebook.

8 **Next Steps.** Ready-made code to Visualise, Diagnose, Predict, Compare and Interpret the fitted model.

9 **Learn More.** Local wikis, notebook tutorials, video tutorials and references.

A tab of your own. Click +, name the tab and write any Julia code in it; it is kept with the panel, ready for next time. Its × removes it.

Do I need to know how to program?

No, not to start. Fill in the boxes and run the example. Reading the code that Pollis writes is the easiest way to learn; changing it is the next step. Julia installs from the Welcome page. When a panel needs a package, Pollis installs it for you, with your agreement, in one click.

Every example runs as shipped

Each example comes with its own data: simulated, or a well-known real dataset from the companion package PollisDatasets.jl. Press *Julia REPL* and it runs. Then replace the example data with yours.

Not a black box

Pollis is not a point-and-click statistics package that hides what it does. Every result comes with the code that produced it. You can read it, change it, rerun it and share it, so a colleague, a referee or a student can check exactly what was done. The path from menus to programming is a short one.

TEACHING BUILT IN

More than tools: it teaches good practice.

Pollis helps you choose a method, check it and understand it.

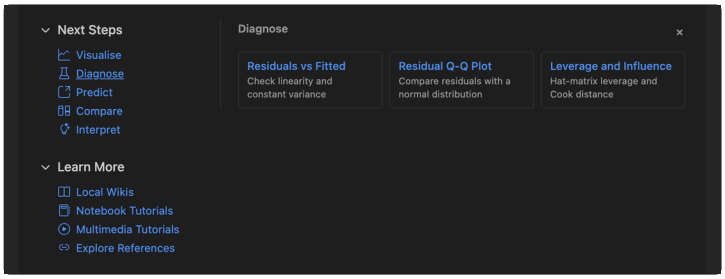
1 Choose: which method, and when?

Decision Table

Model	Data Type	Use when
LOESS	Exploring the shape of a trend without committing to a model; no forecast beyond a short extrapolation	Locally weighted polynomial regression on time: a smooth trend whose flexibility is set by the span
Local Level / Trend	A series that wanders with no seasonality: a level (random walk plus noise) or a level with a drifting slope	Separates signal from noise with the Kalman filter; the signal-to-noise ratio decides how quickly the level adapts
Seasonal Model	A trending series with a stable seasonal pattern of known period	Basic structural model: stochastic level, slope and dummy seasonal, each with its own variance
ETS Framework	Operational forecasting of many series; a fast, reliable benchmark	Exponential smoothing in state space form: weighted averages of the past with optional (damped) trend and seasonality
ARIMA	Series whose autocorrelation structure, after differencing, is short and regular	Differenced autoregressive moving-average model; the order is picked from the ACF and PACF or by AICc
Unobserved Components	When trend, seasonal and cycle should be estimated and read as separate components	Harvey's structural model: choose a trend type and stochastic or deterministic seasonality, all estimated jointly

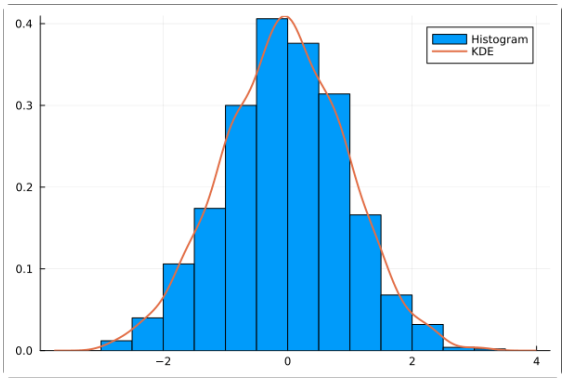
The Decision Table of the Time-Series Forecasting panel: what kind of data each model suits, and when to use it, in plain words.

2 Check: what does a careful analyst do next?



Next Steps → Diagnose, after a linear regression: residuals against fitted values, a residual Q–Q plot, leverage and Cook's distance.

After fitting a model, **Next Steps** offers ready-made code for five things a careful analyst does: **Visualise** the fit, **Diagnose** its assumptions, **Predict** new values, **Compare** it with other models and **Interpret** the results. One click sends the code to the Julia REPL, and the plot appears.



Next Steps → Visualise → Density Overlay: a smooth KDE curve on top of a histogram.

3 Learn: where can I discover more?

Learn More is not merely documentation, but a place for discovery and intellectual inspiration. Go beyond the standard information and discover new knowledge, ideas and perspectives. Explore inspiring cases and practical examples, and let them inspire your own analyses. Learn More brings together:

- **Local wikis** that go beyond the method, to the ideas, fields and questions around it;
- **Notebook tutorials** that tell a story, with comments, images and code you can run and change at your leisure: a rich teaching tool;
- **Video tutorials**, from high-performance computing in Julia to MIT's *Introduction to Computational Thinking*: many topics, for many audiences;
- **References**: the papers and books behind the method and its packages.

Method, code, interpretation, literature

Most software runs the method you ask for. Pollis also connects it to the choice that came before (the Decision Table), the checks that come after (Next Steps) and the knowledge, ideas and literature around it (Learn More). That is the teaching a good supervisor does, built into every panel.

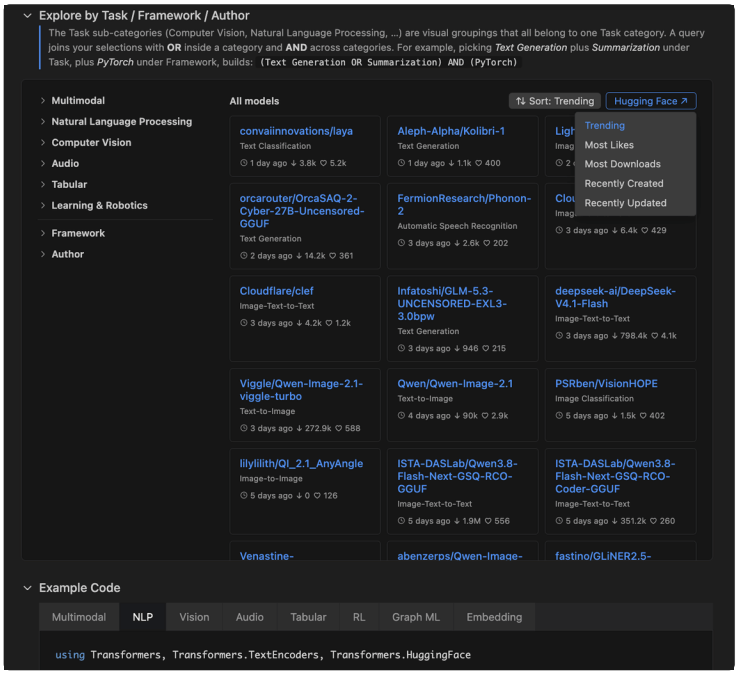
COVERAGE

The whole job in one place, at the size you need.

From classical statistics to machine learning, simulation and operations research.

Explore 49	Model 69	Simulate 26	Optimise 52
Data libraries Excel, CSV and SQL 35 plot types, from box plots to Sankey diagrams, choropleth maps and voxel plots Descriptive statistics Anomaly and novelty detection Missing data, interpolation	Linear models with robust inference, 2SLS, GMM Generalised and mixed models Time-series forecasting, state space Survival analysis Bayesian analysis Trees, ensembles, clustering Neural networks, transformers Text: topic models, BERT, LLMs	Random and quasi-random numbers Markov chain, sequential and multilevel Monte Carlo Bootstrap Probabilistic programming Stochastic differential equations Agent-based models Network models	Linear, quadratic, conic and nonlinear programming Global, Bayesian and evolutionary optimisation Multi-objective, robust and stochastic optimisation Optimal control Reinforcement learning, including AlphaZero

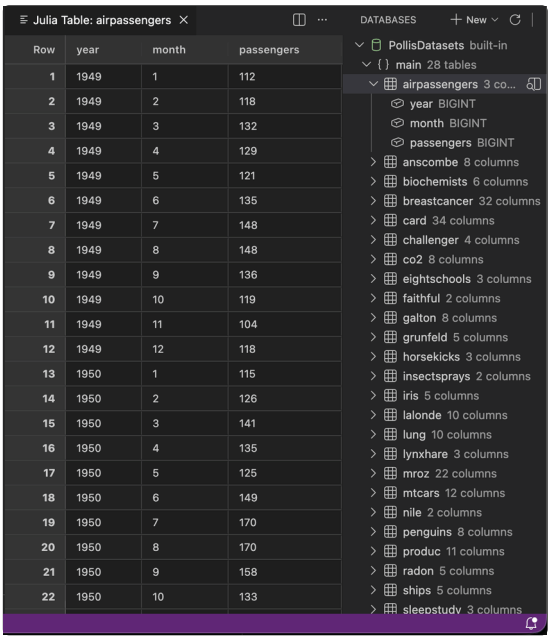
Data at hand. Panels read data from **Yahoo Finance**, **Alpha Vantage**, **US SEC EDGAR**, **FRED**, the **ECB**, the **IMF**, the **World Bank**, **DBnomics**, the **Kenneth French Data Library**, **Hugging Face** and **Kaggle**, as well as **Excel**, **CSV** and **SQL**. A built-in database and a Databases pane cover DuckDB, PostgreSQL and SQLite.



Browse Hugging Face models by task, framework and author, sorted by trend, likes, downloads or date, without leaving Pollis.

Models, not just data

Compose → Local Reasoning Models opens Hugging Face and Kaggle models, and the runners that serve them on this computer, such as Ollama. Example code covers multimodal, language, vision, audio, tabular, reinforcement-learning, graph and embedding models.



The built-in database: 28 real data sets, ready to query and to browse as a table.

Private by design

No telemetry, no cloud: Pollis collects no usage data and runs no cloud service of its own, so your data and models stay on your computer. **Your keys stay safe:** keys for Hugging Face, Kaggle, FRED and Alpha Vantage, and the contact e-mail EDGAR asks for, are kept in your system's secure store (Keychain on macOS, Credential Manager on Windows). They are passed only to the running Julia session and never written into your code, notebooks or files, so a script you share never leaks a key.

CUSTOMISATION

Make it yours.

Shape Pollis around a course, a book, a research group or an institution.

Most analysis software gives everyone the same menus. Pollis lets the people who teach, write and research decide what their students and readers see, and still hands them real code.

1 Views

Choose what to show.

Select the panels you need and reshape the menu structure around them. Save the result as a named View: a course, a talk, a book.

No programming needed

2 Your panels

Change the examples.

Edit a panel's example code to use your data, your variable names and your notation, and keep it, or add tabs of your own. The panel becomes your version of the method.

Reading and editing Julia

3 VS Code Extensions

Add your own menus and panels.

VS Code extensions of your own add new menu items, panels and commands. They can use any licence, from MIT to closed source.

Extension development

Who does what

A student

A View for the course or the thesis, and panels whose examples now use their own data: a personal toolkit that grows with each project.

A lecturer

A View with the ten panels of the course, and examples edited to match the lecture notes. Students see only what they need; the rest of Pollis is still there when they are ready.

An author

A View that follows the book chapter by chapter, with panels that carry the book's own examples, so readers can rerun every figure and table.

A research group

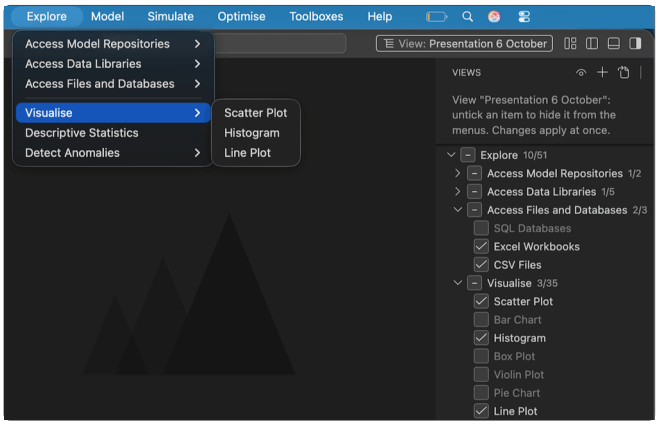
The group's own methods as a VS Code extension, next to the standard panels, so new members start from shared, documented code.

An institution or company

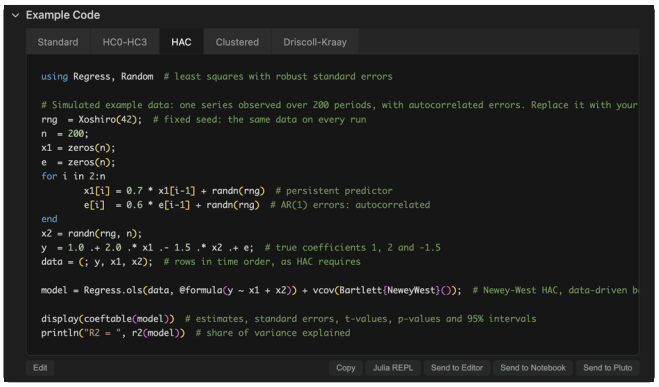
Internal tools and data sources in a private extension, under its own licence, inside the same familiar environment.

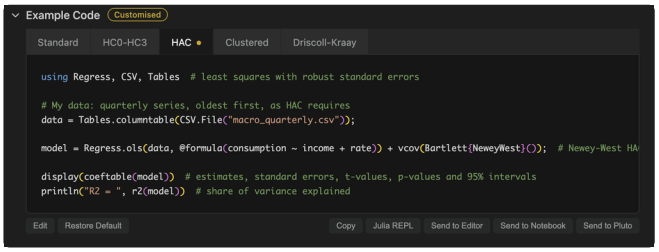
A speaker

A View with the six panels of a talk, ready to run live in front of the audience.



1 A View for a talk: only the panels it needs. The menus (left) show what is ticked in the Views panel (right). Visualise keeps just 3 of its 35 plot types; the other 32 are one tick away.





2 A panel's example as it ships (top), and edited to use your own data (bottom). Save keeps an edit, Cancel drops it, and Restore Default brings back the original.

EXTENSIONS AND WORKFLOWS

Add a field. Replay a classic.

Toolboxes bring whole fields to the menus; workflows replicate famous analyses step by step.

Pollis Toolbox Extensions

A toolbox adds a field to Pollis: a menu of ready-made panels, installed with one click from the **Toolboxes** section of the Extensions pane, and removed as easily. A toolbox describes its panels and holds no other program, so it is quick to make. Your coding agent can package one from the panels you choose (**Compose → New Panels and Toolboxes → Package a Toolbox**), to share with colleagues or with everyone. Each toolbox picks its menu; those with no natural home, such as Mathematical Core, gather in an **Extend** menu.

VS Code Extensions

For more than panels: a VS Code extension adds its own menus, panels and commands, such as a research group's methods or a company's internal data sources (page 6). It comes from the Open VSX Registry or as a private file, and may use any licence, from MIT to closed source. Pollis also runs the themes of the Open VSX Registry.

Three toolboxes, ready from the start

21 entries

Mathematical Core

Symbolic Math
Linear Algebra
Nonlinear Systems
Numerical Differentiation
Numerical Integration
Ordinary Differential Equations
Partial Differential Equations

Economics

Business Cycle Filters
Stochastic Volatility Macroeconomic Models
Dynamic Stochastic General Equilibrium (DSGE)
Heterogeneous-Agent Models
Continuous-Time Macro-Finance
Game Theory

Financial Econometrics

ARIMAX Models
VAR/VECM Systems
GARCH-Type Models
Regime Switching Models
Advanced Time-Series Forecasting
Neural Network Time-Series Forecasting
Automatic Time-Series Forecasting
Bayesian Linear Stochastic Discount Factor
Fama–MacBeth Regression

Workflows: learn from the classics

Workflows replicate famous analyses from papers and textbooks. They go step by step and print each result beside the published one, so you can see your work rests on solid foundations. A workflow is a notebook: each step is a short Julia function you can reuse, and the dataset is an input, so the same workflow runs again on your own data. A step can test what it finds and take a different path. Students follow a guide beside the panels; experienced users run the whole workflow at once. Find them in **Compose → Workflows → Browse Analytical Workflows**. Operational Workflows turn a notebook into a report or slides.

Panel data

Baltagi (2005)'s panel regressions, on Grunfeld's investment data, OECD gasoline demand and Munnell's public capital in the US states: pooled, fixed and random effects, table by table.

Volatility

Bollerslev and Ghysels (1996): a GARCH model with a calendar effect in the variance, on daily Deutschmark–pound returns from 1984 to 1991.

Survival

Dickson, Grambsch, Fleming, Fisher and Langworthy (1989): the Mayo Clinic model for primary biliary cirrhosis, a Cox model with seven years of predicted survival.

Risk

Dalal, Fowlkes and Hoadley (1989): the Challenger O-rings. At 30 °F the paper expected "about (1, 6)" damaged O-rings; the workflow computes (1.27, 5.95).

New methods, with their proof attached

Nine times, a paper needed a method that no Julia package had. Each is now a documented Julia package, tested against the paper's tables or its authors' code, with a vignette, in **Compose → New Methods**, ready to use on your own data.

REPORTS

From notebook to PDF.

LaTeX reports and beamer slides from any notebook, with or without an agent.

Reports: from notebook to PDF

Turn any notebook into a LaTeX report or a set of beamer slides with one click, from the Export button on the notebook's toolbar or from **Compose → Reports**. Pollis runs the notebook and writes its text, results and figures into a document, then compiles the PDF. No agent is needed. A workflow can bring its own report template, laid out like the paper it replicates. When the data change, export again: the numbers are refreshed and your edits are kept. If you like, your coding agent can then polish the draft into a finished document, with sections, tables, captions and citations, using the numbers already computed.

Report

Export Notebook as LaTeX.

- The notebook's text, in a LaTeX article.
- What each cell prints, and each figure, in its place; the code stays in the notebook.
- Compiled to PDF, shown beside the source.

Slides

Make Slides from a Notebook.

- Beamer slides from the same notebook.
- A frame for the text under each heading, and one for each result and each figure.
- Ready to present, or to polish.

Polish

optional

Polish with Agent.

- Your coding agent rewrites the draft: a title, an abstract, sections, tables and captions.
- It never retypes a number: every value comes from what Pollis computed.
- The PDF reloads beside you as it works.

How a report is made

1. **The notebook runs.** Its code cells run in order, and what each prints and draws is saved beside the document; a workflow's template writes the tables and figures of its paper instead.
2. **The draft is written once.** The `.tex` file is written only when there is none, so it is yours to edit from then on.
3. **Export again, at any time.** New data or a changed step: the results and figures are refreshed, and your text stays as you left it.
4. **The PDF is compiled in Pollis,** with Tectonic, through the **PollisTeX.jl** package: no separate LaTeX installation is needed.

Published values, side by side

When a notebook replicates a paper, the polished report shows each published value beside the one Julia computed, rounded to the published digits, with the page or table it comes from and how many decimals they share. Where they differ, the report says so.

Every reference, ready to cite

Pollis keeps one library of every work its panels and notebooks refer to, 679 in all, as a single BibTeX file. Reports cite from it as Author (year), with every author written out and a DOI link, with nothing to copy.

In one line

Run the notebook, get the PDF, keep your edits: a report or slides in one click, polished by your agent if you like.

PRACTICE

Learn by doing, with every answer checked.

Self-study, tutoring and assessment, computed in Julia.

Practice: self-study, tutoring and assessment

Anyone can study with Pollis: a student revising, a professional learning a new field, a teacher exploring a topic of their own. Choose a topic and practise. Pollis computes every answer in Julia, so each one is checked, and each exercise draws its own data from a seed, so every attempt is new. An AI tutor gives hints and explains, and you see your progress through practice or through timed exams marked automatically. On a computer, much more than knowledge can be scored: the skills and competencies of real analysis, from choosing a method to checking and reading its results. Studying on your own does not mean studying alone: with Collaboration, and the simulation games to come, you learn with others, even across oceans and language barriers. Teachers can also write exercises, by hand or with their coding agent and in any language, create exams with a slightly different version for each student, and export them to learning platforms such as Canvas, Blackboard and Moodle.

Learn

On your own, at your pace.

- Choose a topic and practise on its exercises, from the first ideas to full analyses.
- Every attempt brings new data, so you practise the method, not the answer.
- Ask the AI tutor for a hint, or for an explanation once you have answered.
- See your progress, topic by topic.

Teach

Write your own exercises.

- Write an exercise by hand: the question, the data and the Julia code that computes the answer.
- Or ask your coding agent to draft a set; Pollis runs each one and checks it before you accept it.
- Write in your own language and in your own field.
- Share your exercises with colleagues and students.

Assess

Exams marked automatically.

- Timed exams, marked as soon as they are handed in.
- A slightly different version for each student, from the same exercises.
- Printed exams as PDF, each with its own answer key.
- Export to Canvas, Blackboard and Moodle.

How an exercise works

1. **A question, with its own data.** Each exercise draws its data from a seed: the same seed gives the same data, so a version can always be rebuilt, and a new seed gives a new exercise.
2. **The answer is computed, never typed.** The exercise carries the Julia code that solves it, so its key is computed on its data, for every version.
3. **Your answer, checked.** A number, a choice or a few lines of code; numbers are checked to the precision the exercise asks for, with a decimal point or a decimal comma.
4. **Then the why.** The tutor explains the result and points to the panel, wiki or notebook where the method is explained in full.

More than knowledge

A written test checks what you remember. In Pollis the exercise is the analysis itself: choose the method, run it, check its assumptions and read its results, on data you have not seen before. These are the skills and competencies an analyst needs at work, and they can now be practised and scored like any other question.

In one line

Practise on fresh data, with every answer computed in Julia: for students, tutors and teachers.

COLLABORATION

Work together, from anywhere.

Side by side or on the same page, with an agent if you like, and one day from your phone.

Collaboration

Work together from wherever you are: two colleagues on different continents, or a tutor helping a student. Start a session, send the link and chat, with images; everything sent is enciphered. Then work in one of two ways, depending on how you like to work together. An agent can be invited to join the session too, and look at the code you share.

Side by Side

Pass the code. Share and compare.

- Each of you keeps your own editor or notebook: any code file, or a Jupyter or Pluto notebook.
- You share it when you are ready. The others see your code next to their own, compare the two line by line, and copy yours with one click.
- Nobody can change anyone else's code.
- You share one editor at a time, when you choose.

Same Page

next release

Edit together. Live editing.

- Everyone works on the same notebook or code at the same time.
- You see the others' changes as they type.
- The host opens a whole project folder to the others, who can browse and edit its files.
- Natural between close colleagues, and a bigger step of trust than Side by Side.

Who works together

Two people: two colleagues, or a student and a tutor.
A small group: a research team, everyone sharing with everyone.
A classroom: students share with the tutor, and the tutor shares with the whole class.
Breakout groups: a class split into teams, with the tutor visiting each one.
A game: teams and roles, private negotiations, and a facilitator who runs the simulation.

An agent can be invited to join any of them. The first version is for two people and small groups of up to five.

Voice Link: collaborate on the move

Coming later: your phone joins a session too. Driving to work, you think of a better way to run an analysis, and you just say it. Your phone sends your words, enciphered, to your own computer, where your agent takes the idea further with Pollis: it runs the panels, workflows and Julia code it needs, and saves a dated note of what it found in your Pollis Workspace. Your phone reads a short summary aloud: hands-free, with your eyes on the road. A colleague at another computer can join and add ideas too: you on the road, your agent and your colleague, on one idea from three places. When you arrive, it is all waiting on your computer.

Above all, your agent knows your work. Panels, Workflows and Views you selected and edited. The analyses, notes and reports you created. So the answer you hear is about your work, not a general one, and better than anything a chat without Pollis could give.

FREE AND OPEN

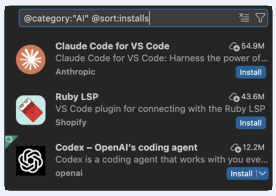
Free, open and yours.

No licence fees, no lock-in, and code you can take anywhere.

- **Free and open source.** Pollis is licensed under the AGPL 3.0. The Julia packages it uses are open source too, most of them (259 of 289) under the permissive MIT licence; the others use BSD, Apache, Mozilla, GPL or AGPL licences.
- **Runs everywhere.** Windows, macOS and Linux.
- **A familiar editor.** Pollis is built on Code - OSS, the open-source core of VS Code, so it looks and works like a code editor many people already know.
- **Extensible.** It runs VS Code extensions and themes, from the Open VSX Registry or your own, and extensions can add new menus and panels (page 6). Extensions may use any licence.

What about AI?

AI is **off by default**, and Pollis works fully without it. See **Pollis AI+** for Pollis and AI working together.



Coming soon: voice

Speak commands, **dictate**, and **listen** to any panel read aloud, with short texts written to be heard. Built on the latest VS Code voice features.

Why Julia?

Julia is free and open source, and it was designed for exactly this kind of work: statistics, simulation and optimisation. Its code reads like the mathematics on the page and runs fast without being rewritten in another language, so the code a panel gives you is the code an expert would write. Its packages work together: a model from one can be estimated, simulated or optimised with another. Python is larger and better known, and Julia can call Python and R libraries when they are needed.[1]

[1] The first time you use a package, Julia prepares it for use. This may take a few seconds for small packages and longer for larger ones; subsequent starts are much faster.

Get started in three steps

1. **Download** Pollis for your system: github.com/saragga/pollis/releases/latest
2. **Install Julia** from the Welcome page: click *Install Julia*.
3. **Open a panel** from any menu, click *Install* if it asks for packages, then press *Julia REPL* and watch the example run.

Download Pollis

Work in progress

Pollis is new. It works today, and its panels are being reviewed one by one, with the key points of each panel being rewritten. Found a problem, or missing a method? Use **Help → Report Issue** or **Search Feature Requests**.

The name

Pollis echoes the Greek *polis*, the community of citizens and the rules it freely adopts. Like a polis, Pollis is shared: free, open source, built on open-source packages, and open to change by those who use it. Its double *l* recalls the Greek *pollá*, "many things": many fields and many methods in one place, intertwined in a code that turns knowledge into action.

Pollis

Thanks. Pollis stands on the work of the Julia language, Code - OSS and the authors of 289 open-source Julia packages.

Antonio Saragga Seabra
contact@pollis.org

Everything in Pollis

5 curated menus · 203 method panels · 656 procedures · 289 open-source Julia packages

Items marked › open a submenu; one is shown open.

Explore

49 entries

Access Data Libraries ›
Access Files and Databases ›
Visualise ›
Descriptive Statistics
Detect Anomalies ›
Detect Novelties
Handle Missing Data
Interpolate Data

Model

69 entries

Model Evaluation ›
Distributions ›
Copulas ›
Hypothesis Testing
Linear Models ›
Linear Mixed Models
Generalised Linear Models
Generalised Linear Mixed Models
Nonlinear Regression
Penalised and Projection Regression
Robust and Quantile Regression
Time-Series Forecasting
State-Space Inference Methods
System Identification Methods
ODE Inference Methods
Nonlinear Dynamics ›
Point Processes
Survival Analysis ›
Bayesian Analysis ›
Decision Trees
K-Nearest Neighbours
Rule-Based Methods
Neural Network Architectures ›
Ensemble Learning Methods
Clustering ›
Dimensionality Reduction Methods ›
Natural Language Processing ›

Simulate

26 entries

Sampling Foundations ›
Markov Chain Monte Carlo
Sequential Monte Carlo
Multilevel Monte Carlo
Bootstrap Resampling
Probabilistic Programming Languages ›
Uncertainty Propagation
Point Processes
State Space Models
Continuous Dynamical Systems
Discrete Maps
Stochastic Differential Equations ›
Agent-Based Simulation ›
Network Models ›

Reinforcement Learning

Bandit Problems
Q-Learning
Policy Gradient Methods
Actor-Critic Methods
Multi-Agent Reinforcement Learning
AlphaZero

Optimise

51 entries

Gradient-Free Optimisation ›
Gradient-Based Optimisation ›
Stochastic and Learning-Based Optimisation ›
Global Optimisation ›
Bayesian Optimisation
Surrogate-Based Optimisation
Evolutionary Computation ›
Swarm Intelligence
Optimisation on Riemannian Manifolds
Discrete and Combinatorial Optimisation
Network and Graph Optimisation
Multi-Objective Optimisation ›
Linear Programming
Quadratic Programming
Polynomial Programming
Convex, Conic and Semidefinite Programming ›
Nonlinear Programming
Hierarchical Optimisation
Complementarity and Variational Problems ›
Uncertainty-Set Robust Optimisation
Chance-Constrained and Distributionally Robust
Stochastic Programming
Dynamic Programming
Optimal Control ›
Markov Decision Processes ›
Reinforcement Learning ⌄
Infinite-Dimensional Optimisation

Compose

13 entries

Local Reasoning Models ›
Coding Agents
Agent Skills
Orchestration ›
New Panels and Toolboxes ›
New Methods ›
Workflows ›
Reports ›
Correctness Tests ›
Profiling ›
Compilation Lab ›
Collaboration ›
Practice ›

Compose goes beyond single panels: a coding agent and its skills, local reasoning models, new methods, analytical and operational workflows, reports, tests, collaboration and practice. Toolboxes add their own menus (page 7).

Bright Insights through Statistical Learning, Simulation and Optimisation

Pollis · free and open source · Windows, macOS, Linux · github.com/saragga/pollis/releases/latest · contact@pollis.org

A Tour of Pollis, dated 15 October 2026