

Re item 2. Editor's report regarding 2025-26¹

1. Preamble

Gekko's steering group was established in 2009, in connection with Gekko being made open source. Since then, annual steering group meetings have been held, including the election of an "editor", etc. In practice, Thomas Thomsen has been the editor every year, and since January 2026, the editor has also been employed at both Statistics Denmark and the DREAM Group (working on both Gekko-related and non-Gekko-related tasks). Therefore, the editor feels that there is presently a high "physical" availability regarding ongoing Gekko development, features, requests, bugs, etc.

For these reasons, the editor regards the Gekko steering group meeting as nowadays having more the character of a user forum, where stakeholders can give the project an overall long-term direction, rather than a detailed planning session. Consequently, this year's meeting is a bit shorter than usual, with an emphasis on topics that have historically been addressed in part 1 of previous meetings (overall planning and prioritization).

The editor considers the sense of direction to be important, also because there are various underlying developments that could well impact the Gekko project and Gekko's viability. For example, the overall movement toward making software a "package" within environments like Python/R/Julia; developments regarding modeling software and solvers (such as GAMS/gamY/CONOPT); integration with databases, dataframes, and Parquet files; and perhaps questions regarding platforms (such as offering Gekko for .NET Core, and perhaps being able to run Gekko on a Linux server).

2. Developments since last meeting

A **PyGekko** (Gekko for Python) package has been released on [PyPI](#). The package makes it possible to call Gekko directly from inside Python. At the moment, the interface is mostly string-based, with syntax like the following:

```
import pygekko as pg
pg.run("read <gdx> makrobk;")
pg.run("prt <p> qBNP, vBNP/qBNP;")
```

It would be possible to integrate the syntax more, for instance making it possible to write syntax like `pg.prt(op = "p", "qBNP", "vBNP/qBNP")`. Such deeper integration would be quite easy to implement (also for Gekko users), but for now the string-based interface works quite well. Gekko windows like DECOMP, PLOT, etc. can be opened directly from Python. Note however that PyGekko will only work on Microsoft Windows.

PLOT has been augmented with the possibility of scaling, either as an option in the PLOT statement, or from the PLOT window. PLOT can now also transform the data for

¹ Links: Gekko main webpage: www.t-t.dk/gekko, organization: www.t-t.dk/gekko/organization. Gekko on GitHub: <https://github.com/thomsen67/GekkoTimeseries>.

consumption in another “plotter”, like for instance Plotly for Python (parquet-files are used to transfer the data).

Parquet-files import and export has been improved quite a lot, in collaboration with Danmarks Nationalbank (Central Bank). Danmarks Nationalbank can now perform a roundtrip of Gekko data from a Gekko databank, to a parquet file, to the Macrobond platform, back into a parquet file, and back into Gekko. See more [here](#). Parquet files are also expected to have a bright long-term future at Statistics Denmark, as backbone data containers in the so-called DUAL project (“Danmarks Statistik Ud Af Legacy”).

A new `balance()` function has been implemented, to aid **RAS/GRAS** adjustments of input-output cells. The function has been tested against the official GRAS algorithm used by Statistics Denmark (117 grouping), and provides identical results on test data. Fixing/exogenizing cells is easy using `balance()`, which uses an arrays-series interface. Using entropy optimization is ongoing.

The `laspchain()` function (**Laspeyres chain indexes**) has been improved as regards missing values in the beginning of, or in the middle of input timeseries. In that sense, the function is now more missings-tolerant.

DECOMP now has an [Infl.] button (“influences”) to quickly see which other variables a given variable affects. Related to this, it would be really nice to implement “negative depths” in the flowgraph window, to the same effect.

Gekko has produced a **html equation browser** for MAKRO, made from the GAMS scalar model representation of the MAKRO model (browser online [here](#)). An equation browser has been produced for the GREU model, too (of the GrønREFORM family).

Some work has been done regarding **data versioning**. The idea is to implement features in Gekko that makes it produce small “pointer” text files (called `.dlink` for “data link”) akin to the text files used by [Git LFS](#). Interfaces regarding so-called Git hooks have been implemented, too. If data files are not allowed to be put into Git (sensitive data), this system could help out.

Quite a lot of **new functions and features** have been implemented as a result of the editor contributing to the so-called “source project” at Statistics Denmark (constructing the MAKRO databank directly from data sources). Also, a MAKRO course identified a number of bugs regarding DECOMP etc. that have been fixed.

Regarding Gekko command files (`.gcm`), there are now online **syntax coloring** available for both VS Code and Cursor (just search for “Gekko” under extensions).

Technical change: The **inner representation** of model variables has been reworked from using strings like “`x[a,b,2025]`” into a more abstract array-like representation with variable name plus any number of dimensions, where one dimension may be designated “timelike” (and also represent for instance lags/leads or fixed periods). The rewrite was quite heavy and did cost a bit of performance when loading GAMS scalar models for DECOMP, but the

code behind DECOMP, html browser etc. is now much more clean and manageable, which is expected to pay off long-term.

Technical change: When Gekko executes a Gekko command file (.gcm) or a Gekko model (.frm), these files are translated and compiled into machine code. The dynamically generated machine code was technically stored as a .dll on the user's computer, which interacted badly with the so-called WDAC (Windows Defender Application Control), which is going to be enforced at Danmarks Nationalbank. To fix that problem, Gekko now uses a **newer compiler**, and .gcm and .frm files are now compiled in RAM only.