Gekko steering committee meeting, June 17, 2021 (virtually on Zoom)[1]

Proposed agenda

Part 1

1. Welcome + choice of minute taker + approval of last year's minutes.
2. Editor's report regarding 2020-21.
3. Status regarding Gekko 2.4 and 3.0
4. Feedback regarding Gekko 2.4 and 3.0
5. Prioritization of main lines regarding 2021-22
6. Organization and choice of editor for the next period
7. Date of the next meeting + any other business

Part 2

8. Commentaries regarding detailed checklist + prioritization of this
9. Other potential users
10. Status regarding documentation, help systems, etc.

[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.

Re item 5. Main lines

Note that Gekko version 2.4 generally means 2.5.2, and Gekko version 3.0 generally means 3.1.12. Unless otherwise stated, the points deal with the Gekko 3.0 version.

1. Finish work on daily frequency, and implement weekly frequency. Use the ideas in the frequency blueprint.
2. Make new DECOMP and pivot of array-series available for users. This should include a comprehensive description in the user manual, and a guided tour of it in the user guide.
3. Better error messages in Gekko 3.0. This is overdue.
4. Discuss the Gekko model blueprint among interested users (meeting/workshop). Decide in how to represent models/submodels/equations as objects, and implement it.
5. Solver: more means than goals (interdependence between goals) via addition of a special discardable model block describing the interdependence. A prototype for doing this has been proposed. More goals than means (goals contribute partly to means). Could perhaps be implemented in Gekko 2.4, too.
6. Regarding better INTERPOLATE (lower frequency to higher frequency): implement a convenient interface for the R package tempdisagg? This package implements Denton, Chow-Lin and many others. If some of the components of this package become popular, they can probably be translated into pure Gekko C# code, eliminating the dependency upon R here.
7. Seasonal adjustment could perhaps be done via the R package RJDemetra. This package handles all the tedious interface stuff in relation to JDemetra+, and Apache Arrow files could be used for data interchange. Using R would make the project more manageable, but the package still requires Java to be able to run. It seems people live with X12A, but a timerseries-oriented program ought to have access to state of the art seasonal adjustment.
8. Metadata, more possibilities? For instance a ”calc” field in timeseries, where calculations (actual series statements) are stored, instead of polluting the ”source” field with this. The calc field will be in form of a list, so that the history of calculation statements can be shown. Perhaps this will be described in a short blueprint before it is implemented.
9. Introductory guides, examples collections, exercises, etc. Is more material needed?

Re item 8. More detailed check list

1. User-developed procedures/functions could possibly be hosted on Gekko's website, in the form of downloadable library packages (cf. the new LIBRARY command). An “authorized” official gekko.zip library package could be made available, containing general helper functions/procedures for general use and inspiration. Or maybe GitHub could be considered for hosting Gekko libraries?
2. Better translator from Gekko 2.4 to 3.0 syntax? The ADAM group and ADAM model users still use Gekko 2.4, and a better translator could help regarding the translation of (parts of) their systems, including databank systems. See more on concrete syntax differences between Gekko 2.4 and 3.0, and the possibility of a simplified Gekko 3.0 syntax in this syntax discussion.
3. Autocomplete of variable names in the Gekko window? Other autocomplete or highlighting?
4. Improved PLOT window? A draft of this exists, inspired by RStudio.
5. Improving the interface to StatBank Denmark? Interface to other online databases like for instance Jobindsats?
6. Showing Gekko options in a clickable tree structure with explanations? Because of some architecture changes in the Gekko source code, this would be easier to do now.
7. Redesign of the inner workings of the Gekko tabelling system. The inner workings are old, and a modernization would not be too hard to do. As a spin-off, users would be able to construct their own tables manually cell by cell by using a simple nested Gekko list (which can then be converted to text or html format).
8. In the longer run, implementing dataframes in Gekko proper as a new variable type could be useful, also for handling Apache Arrow files and interfacing more easily with R and Python (the new DECOMP could use dataframes objects, too).
9. Think about migrating Gekko to .NET Core at some point (and migrate the main Gekko window to WPF, too). This ought to be considered in a timely fashion, before the existing .NET Framework is phased out.