GAMS Models
in Agricultural Economics

Wolfgang Britz
Institute for Food And Resource Economics
University Bonn
• Simulation models in agricultural economics
• CAPRI as an example: what, why and how
• Why GAMS: features, costs, user community
• Limitations of GAMS
• CAPRI GUI: wrapping GAMS
• Summary
• Long tradition:
  – 1950ties: LP based farm level optimization, today mostly MIP, fully dynamic, stochastic
  – 1970ties: application to „typical“ or „regional“ farm for policy analysis
  – Branch dried out for a while due to typical problems of LPs (jumpy behaviour, over-specialization) and hunger for data not available from statistics
• 1980ties, **price endogenous market models:**
  
  – **Multi-Commodity models:**
    • Global simultaneous equilibrium for many agricultural products and many countries/country blocks
    • Prices, supply, demand and trade endogenous
    • Non-linear CNS or NCP

  – **Computable General Equilibrium Models:**
    • Simultaneous equilibrium for all markets in an economy (input, outputs, primary factors)
    • Non-linear CNS or MCP
• Today:
  – Revival of programming models
    • introduction of (econometrically estimated) non-linearities eases calibration and avoids over-specialization
    • increasing focus on agri-environmental interactions (bio-economic modelling)
  – All model types grow in resolution (products, space, agents) and complexity (policy instrument covered, functional forms, stochastics ..)
  – Combination of different type of models into tools for impact assessment
• **Common Agricultural Policy Regionalized Impact:**
  
  – Tool for *impact assessment* of policies related to the agricultural sector
  
  – Clients: mainly *EU commission*, but also national government and industry
  
  – Combines economic models, matching data bases, IT infrastructure and institutional setting
  
  – Financed mainly by EU research framework programs and tenders
CAPRI as an example

- Two modules, links by sequential calibration
  - About 2000 *farm type models* with
    (each ~300 vars x ~200 equations, NLPs, independently solved as grid)
  - Global, spatial trade model for agricultural products
    (CNS, in parts highly non linear, 40,000x40,000)
  - One run solves each module ~20 times, ~20-30 minutes,
    generates ~15 Mio non zeros, in parts post-model calculated economic and environmental indicators
• Simulates impacts of policy instruments such as tariffs, subsidies, agri-environmental programs
• On various indicators such as
  – Crop areas, yields, herd sizes, production, feed use
  – Human consumption, industrial processing, trade flows
  – Farm income, EU budget costs, consumer welfare
  – GHG emissions, nutrient balances
• Developed end of the nineties, since then continuously expanded, improved and applied
• Open source concept, network based
• Complete work flow in GAMS
  – Data base generation and model parameterization:
    • Mapping/aggregation/dis-aggregation of raw official data
    • Outlier detection and correction
    • Bayesian estimators to consilidate raw statistical data
    • Estimators for behavioural parameters
  – Generation of „market outlook“ (currently 2020)
  – Model calibration to outlook results
  – Simulation runs
  – Post model processing, including statistical down-scaling to about 200,000 1x1 km clusters
  – Results are passed between worksteps via GDX
Why GAMS?

• Developed originally from economists for economists:
  – Notation comes close to scientific papers
  – Relatively easy to learn
  – Compact notation for element-wise operations, avoids explicit loop and if statements
  – Transparent interface to high performance solvers for different problem types: LP, NLP, MIP, CNS, MCP etc.
• Basic **IDE**, sufficient for medium sized projects

• **Proprietary binary data format** (GDX):
  – Fast, safes disk space, tailored for large sparse matrices, link e.g. to EXCEL
  – API support for high-level programming languages, provides bridge to DBMS or tailored GUIs

• Continous development e.g. support for new solvers
Why GAMS?

• Relatively restricted language features
  – no functions/sub-routines
  – limited number of object classes (set, parameter, equation, variable, model, file ..)
  – all numbers are double precisions (not distinctions between int, long, float, double etc.)
  – no explicit string handling
  – every symbol has global scope
Why GAMS?

• **Consequences** of restricted language features:
  – no formal education in IT required – fits typical profile of (agr) economists
  – debugging is typical easy
  – low learning costs
  – dis-advantages for experienced programmers and large-scale projects
Why GAMS?

- **Trial version** (often sufficient for didactic purposes) can be downloaded for free
- Relatively **modest license fees** for degree granting institutions
- Good **manuals**, active e-mail list
- Long list of **example models**
- **Portable** between platforms, can be used without GUI/IDE
Why GAMS?

- Features explain wide spread use in community as a kind of "lingua franca":
  - Taught in courses at master level based on "toy models", e.g. already in Germany by various ag-econ departments
  - Allows researchers to switch relatively easy between tools and institutions
  - Eases tool linkage
• Many other examples of policy relevant ag-econ models realized in GAMS, e.g.
  – Uni Hohenheim: ESIM
  – Univ. Wageningen, NL: farm type models
  – Texas A&M: FASOM
  – Penn state: PEATSIM
  – Purdue: GTAP in GAMS
  – ...
Limitations

• Modular design and exchange of code between projects not easy:
  – Global scope of all symbols, only rudimentary support for functions/subroutines
    ⇔ namespace conflicts
    ⇔ no encapsulation
    ⇔ documentation is tedious
  – IDE not ideal for large-scale projects compared e.g. with Eclipse

Spaghetti code

Each symbol global
Limitations

- So far **no parallel execution in the base module** and in some widely solvers such as CONOPT:
  - Multiple cores not used
  - Pre- and post model processing increasingly CPU bottleneck
- Parallel model solves supported, but I/O intensive
  - Little gain for a suite of small model instances
  - „hot“ updates in solvers in early stages, applicability not yet clear
• GAMS alone not appropriate to host complex tools, CAPRI e.g. should allow
  – users to easily define, start and exploit own scenarios (if possible without touching GAMS directly)
  – administrators to steer supporting work steps (such as data base and outlook generation, calibration), best without coding
  – efficiently analysis of huge data sets
• CAPRI responded by developing its own GUI
  – Realized in Java
  – Steers GAMS work steps by generating „code snippets“ and spawning GAMS
  – comprises powerful exploitation tools (reports as tables, graphs, maps) to analyze results
  – access to GAMS generated data via API bridge to GDX
What to do (step & task)

Switch modules on/off

Post-model reporting options

Start GAMS
Work step is running … (GUI has spawned GAMS)

GAMS output redirected to GUI
After the run:
Exploit results ...

Chose several scenarios
CAPRI GUI

Pre-defined reports (tables, graphs, maps)

Allows for pivot, selections, comparisons etc. …
• Further GAMS related GUI features:
  – Generation of HTML based documentation of GAMS files à la Javadoc
  – Batch processing mode (check of all work steps after code changes, running packages of scenarios)
  – Meta data generation/handling (are stored along with numerical results in GDX files)
• GAMS has currently high market share for agricultural economic simulation models

• Probable reasons
  – Ease of use and convincing basic concept
  – Transparent link to solvers
  – Relatively low cost
  – Existing market penetration

• Larger projects wrap GAMS in applications
• Future market share of GAMS might depend on:
  – Further performance gains (parallel execution, „hot“ solver updates, ...)
  – Combining the appealing simple concept with more advanced features such as scoping, object oriented design, functions/subroutines ... but how?
  – Adding functionalities such as automated generation of project code documentation or link to SVN
  – Development of alternatives such as libraries in Object Oriented Languages (Java, C#), already now applied for some newer methodologies such as Agent Based Modelling
Thanks for your attention

more on CAPRI

www.capri-model.org

contact author

Wolfgang.Britz@ilr.uni-bonn.de