Standards & Interoperability in Economic Modelling
Some lessons from Dynare

Sébastien Villemot

Dynare

CEF 2026 Pre-conference Meeting
Venice, 28 June 2026

Outline

1. What is Dynare?
2. The Dynare DSL: strengths and weaknesses
3. Standardization vs interoperability
4. Standardization is much more than a good specification
5. Lessons for the Standards Agenda

Dynare at a Glance

Mission statement

"Write your model almost as you would on paper and Dynare will take care of the rest!"

Dynare in practice:

- A **domain-specific language** (DSL) for describing economic models
- A **preprocessor** that translates the DSL into computational code input
- A **collection of computational routines** for simulation, estimation, post-processing...
- An **open-source project** (GPL-licensed)
- A **community** of users and contributors worldwide

Current capabilities (version 7):

- DSGE (incl. RBC, RANK, TANK)
- Heterogeneous agent models (incl. HANK)
- Semi-structural models (incl. PAC)
- Simulation: rational expectations (local perturbation), perfect foresight
- Estimation: full-information Bayesian or frequentist, SMM, GMM, IRF matching...
- Optimal policy: Ramsey, discretion
- Platforms: MATLAB, GNU Octave, Julia

Aggregate models, specified as FOC (1/3)

Perfect foresight case

$$F_\theta(y_{t+1}, y_t, y_{t-1}, x_t) = 0$$

- y_t (resp. x_t): vector of endogenous (resp. exogenous) variables
- F: multivariate homogeneous equilibrium conditions

2-equation neoclassical growth model

$$C_t^{-\sigma} = \beta\left(\alpha Z_{t+1} K_t^{\alpha-1} + 1 - \delta\right) C_{t+1}^{-\sigma}$$
$$C_t + K_t - (1-\delta)K_{t-1} = Z_t K_{t-1}^{\alpha}$$

```
var c k;
varexo z;
parameters alpha beta delta sigma;

model;
  c^(-sigma) = beta*(alpha*z(+1)*k^(alpha-1)
    + 1-delta)*c(+1)^(-sigma);
  c + k = z*k(-1)^alpha + (1-delta)*k(-1);
end;
...
steady;
perfect_foresight_setup(periods=50);
perfect_foresight_solver;
```

Aggregate models, specified as FOC (2/3)

Rational expectations case

$$\mathbb{E}_t F_\theta(y_{t+1}, y_t, y_{t-1}, x_t) = 0$$

- Expectation operator implicit in Dynare DSL: no change compared to perfect foresight
- The selection between the two cases is done by the choice of computational commands

2-equation neoclassical growth model

$$C_t^{-\sigma} = \beta\,\mathbb{E}_t\left(\left(\alpha Z_{t+1} K_t^{\alpha-1} + 1 - \delta\right) C_{t+1}^{-\sigma}\right)$$
$$C_t + K_t - (1-\delta)K_{t-1} = Z_t K_{t-1}^{\alpha}$$

```
var c k;
varexo z;
parameters alpha beta delta sigma;

model;
  c^(-sigma) = beta*(alpha*z(+1)*k^(alpha-1)
    + 1-delta)*c(+1)^(-sigma);
  c + k = z*k(-1)^alpha + (1-delta)*k(-1);
end;
...
steady;
stoch_simul;
```

Being user-friendly

- Lead and lag management
  - The solvers operate on models with at most one lead and one lag on endogenous (and no lead/lag on exogenous)
  - However that constraint is not imposed in the DSL: the preprocessor makes the relevant symbolic transformations (with auxiliary variables and equations)
- Timing convention
  - By default, timing of variables must reflect *when they are decided*
  - For stocks, this means end-of-period timing convention
  - Advantage: no need to split variables into two groups (state vs control)
  - Flexibility: the `predetermined_variables` declaration enables an alternative timing convention on selected variables

Aggregate models, specified as FOC (3/3)

Regime switching

- Generalization as mixed-complementarity problem (MCP)

$$F_\theta(y_{t+1}, y_t, y_{t-1}, x_t) = 0 \ \perp \ \ell \leq y_t \leq u$$

```
model;
  (c^theta*(1-L)^(1-theta))^(1-tau)/c - mu = beta*((c(+1)^theta*(1-L(+1))^(1-theta))
    ^(1-tau)/c(+1)*(alpha*(y(+1)/k)^(1-psi)+1-delta)-mu(+1)*(1-delta));
  mu = 0 ⊥ i > 0;
...
```

- More flexible switching, with equations differing across regimes, using equation tags

```
model;
  [name='Notional rate Taylor rule']
  i_not = rho*i_not(-1)+rho*(phi_pi*pie+phi_y*y)+zeps_i;
  [name='Observed interest rate', relax='elb']
  i = i_not;
  [name='Observed interest rate', bind='elb']
  i = i_elb;
...
```

Aggregate model, specified as maximization problem

$$\max_{i_t} \; E_0 \sum_{t=0}^{\infty} \beta^t u(y_t)$$

$$\text{subject to } F_\theta(y_{t+1}, y_t, y_{t-1}, x_t) = 0$$

where instruments i_t are a subset of endogenous y_t

```
model;
  ... // Constraints
end;

planner_objective log(C) - chi*N^(1+phi)/(1+phi);

ramsey_model(instruments=(i), planner_discount=beta);
stoch_simul;
```

Social planner FOCs derived automatically by Dynare

Heterogenous-agent models (1/2)

FOC of individual problem in Krussel & Smith model

$$\lambda_{j,t} = R_t c_{j,t}^{-\sigma} \qquad \beta \mathbb{E}_t \lambda_{j,t+1} + \xi_{j,t} = \frac{\lambda_{j,t}}{R_t} \qquad \xi_{j,t} = 0 \perp k_{j,t} > 0$$

```
heterogeneity_dimension households;
var(heterogeneity=households) c k lambda xi e;   // Idiosyncratic endogenous
varexo(heterogeneity=households) eps_e;          // Idiosyncratic exogenous
var R W C Y K;                                   // Aggregate endogenous
varexo Z;                                        // Aggregate exogenous

model(heterogeneity=households);                 // Idiosyncratic equations
  c = (lambda/R)^(-1/sigma);
  beta*lambda(+1) + xi = lambda/R;
  xi = 0 ⊥ k > 0;
  ...
end;

model;                                           // Aggregate equations
  K = SUM(k);
  ...
```

Heterogeneous-agent models (2/2)

$$\mathbb{E}_t F_\theta(x_{t+1}, y_t, y_{t-1}, x_t, \int \tilde{y}_{j,t} \mathrm{d}j) = 0$$
$$\mathbb{E}_t G_\theta(\tilde{y}_{j,t+1}, \tilde{y}_{j,t}, \tilde{y}_{j,t-1}, \tilde{x}_{j,t}, y_{t+1}, y_t, y_{t-1}, x_t) = 0$$

where j indexes the continuum of agents, $\tilde{y}_{j,t}$ (resp. $\tilde{x}_{j,t}$) are idiosyncratic endogenous (resp. exogenous) variables, G_θ are idiosyncratic FOCs

DSL allows immediate extensions for:

- Multiple heterogeneous dimensions
- Cross-variables ($y_{j,k,t}$ with j and k heterogeneity dimensions)

Unifying the Interface: the HANK example

Two major approaches to HANK

- Auclert, Bardóczy, Rognlie & Straub (2021) — **Sequence-space** Jacobian (SSJ)
- Bhandari, Bourany, Evans & Golosov (2023) — **State-space** perturbation
- Mathematical objects, terminologies and algorithms very different between the two communities

Dynare's approach: unify the interface, not necessarily the algorithms

- A single Dynare heterogeneous-agent syntax
- Under the hood, different solvers can be called — without the user rewriting the model
- Analogue of what Dynare has always done for DSGE models: one `model` block, multiple estimation/solution methods

Takeaway for the standardization agenda: A good DSL standard separates *model description* from *computational method* — these should be orthogonal concerns.

Model classes covered by DSL, and possible extensions

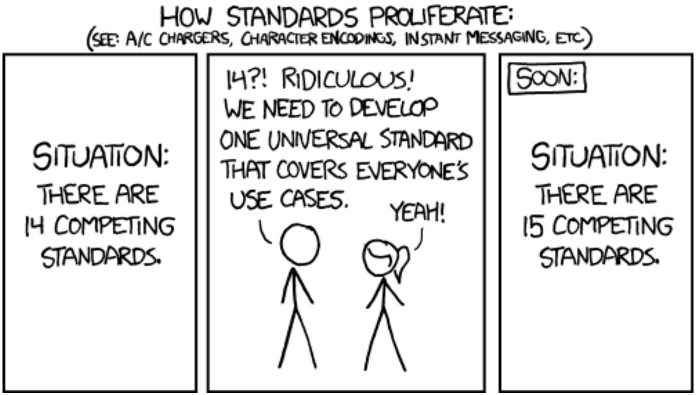
Aggregate structural models, specified as FOC	✓
Aggregate structural models, specified as FOC, with regime-switching	✓
Aggregate structural models, specified as maximization under constraints	✓
Heterogenous-agent structural models, specified as FOC	✓
Semi-structural models, specified as dynamic equilibrium conditions	✓
Heterogenous-agent structural models, specified as maximization under constraints	easy
Heterogenous-agent structural models, with regime-switching	easy
Continuous time	medium
Sequential and/or strategic decision making (ABM-like)	hard

Philosophy of the Dynare DSL

- **Closeness** to concepts and objects manipulated by economists
  - Equations written as in a paper
  - Vocabulary mostly from economics, not from mathematics or computer science
- **Declarative**, not imperative: primarily describes *what* should be done, not *how*
- **Separation** of model description from computational tasks
- **Simplicity**
  - Raw text file as input
  - Keywords with parenthesized name-value arguments
  - Semi-colon separator
- **Extensibility**: design decisions made with the future in mind
- **Stability**: backward-incompatible changes kept to a minimum

Disclaimer: those are objectives, not fully met by actual DSL; historical mistakes hard to fix (conflicts with stability objective)

Standardization vs interoperability



Source: `https://xkcd.com/927/` (CC BY-NC 2.5)

- Unification around a single DSL is unrealistic, for both technical and social reasons
- Communication between projects (as done today) is essential
- Interoperability as a less ambitious but more pragmatic route to unification

What Dynare offers for interoperability

- **Dynare DSL → other tools**
  Preprocessor outputs:
  - JSON file with declarations and computational commands
  - Model residual and derivatives in MATLAB/Octave, Julia, Python, JSON
- **Other tools → Dynare computational routines**
  Load commands (through CSV or MAT-files) for:
  - steady state
  - initial values
  - scenarios
  - data files
- LLMs good at translating languages
  - Dynare documentation and forum provide lots of online LLM training material
  - Language server protocol (LSP) for Dynare DSL: a third-party one already available, an official one coming soon

The Platform Question

- DSL vs platform
  - In principle the DSL could be a complete programming language, but too costly in practice
  - Hence the underlying language/platform is exposed to users, and matters for standardization and interoperability
  - The Dynare DSL partially abstracts the platform, since it covers most high-level computational tasks, easing platform transitions
- Move from Gauss to **MATLAB** in the early 2000s was the right call at the right time: MATLAB became the *lingua franca* of quantitative economics
- **GNU Octave** support around 2007 provided a free/open-source path — crucial for accessibility in developing countries and teaching
- Recent **Julia** (unfinished) port: Julia is modern and fast, but adoption remains limited
- **Python**: the future? Very popular now, though JIT is not first-class, and less elegant for numerical code

Industrial-quality development process

- Continuous integration, test suite, stable release management — not a prototype that "usually works"
- **Maintenance** over decades
  - Handling bug reports, implementing and publishing fixes
  - Adapt to constantly-evolving technological environment (platform, operating systems, language versions, library dependencies, compilers, build system, development forge...)
- Structured change management: backward-compatibility guarantees, deprecation warnings
- A **team**, not a single-person project: sustainability is essential for institutional adoption
- **Documentation** effort: tutorials, reference manuals, example bases
- **Open source**: enables independent verification and reproducibility

Governance and community

- Thousands of posts on the Dynare Forum over two decades — patient, detailed replies, culture of mutual assistance
- Dynare Summer Schools / trainings run every year in Paris and across the world
- Workshops for advanced users at institutions
- Open-source process
  - Channels for user and institutional feedback
  - Community contributions welcomed
  - Transparent roadmap
- Funding strategy
  - Institutional hosting
  - Sponsorship and grants
  - Service contracts

The Digital Commons perspective

Cherrier, Saïdi & Sergi (JEDC, 2026): Dynare as a "digital commons"

- Following Ostrom: a *commons* is a shared resource governed by community institutions
- Dynare is not just code — it is a set of *governance institutions* that ensure its sustainability
- Prevents "tragedy of the commons": without governance, the tool would have fragmented or become obsolete
- Michel Juillard's contribution is simultaneously **intellectual**, **technical**, and **institutional**

Three pillars of success

1. **Tractability**
   Reliable algorithms, scalable to policy-relevant models
2. **Portability**
   Models can be shared, taught, replicated across institutions and countries
3. **Community governance**
   Open, sustained, institutionally anchored coordination

Adoption as a path-dependent endogenous process

Some endogenous or unpredictable factors

- Early institutional movers (*e.g.* IMF) critical for adoption and credibility
- Timing matters: *e.g.* Bayesian estimation implemented when academia and policy worlds were intellectually ready for it
- Open-source licensing enabled unrestricted sharing and reuse
- Once a critical mass reached, switching costs reinforce the position

The burden of the past

- Some early design decisions are now locked in, even if we would make different choices today
- Fixing design mistakes easier on implementation side than in DSL (less user-visible impact)
- Lesson: *invest heavily in language design upfront*

Lessons from Dynare for the Standards Agenda

1. **Separate model description from computation.**
   A good DSL lets users write models once; solvers can be swapped underneath.
2. **Design the language carefully — you will live with it for decades.**
   Path dependence is real. Invest in language design upfront; technical debt compounds.
3. **Standards emerge endogenously, but can be catalysed.**
   Institutional adopters and community building are the most powerful levers.
4. **Open source is essential.**
   Scalability and reproducibility require that the standard and its reference implementation be freely available.
5. **The Dynare DSL as a base for a standard?**
   We are certainly open to it, but do not underestimate the technical and social challenges.

Thank you!

Sébastien Villemot

sebastien@dynare.org