Agent-based Models: Methodology, Calibration and Estimation
Part 2: Bayesian estimation with Gaussian regression surrogate

Sylvain Barde

School of Economics, University of Kent

July 3, 2023

CEF pre-conference workshop, Nice

Outline

Disclaimers - just a reminder

- Just as a reminder, this section is much more ‘open’ than the previous one
  - This is still very much an active research area.
  - I am presenting one approach that I believe works well.
  - But this is subject to (rapid) change. **Caveat emptor!**
- I am going to present an estimation methodology that is designed for the more **computationally-intensive** ABMs
- But I do want to flag some very similar approaches
  - The surrogate approach of Lamperti et al. (2018)
  - Platt (2022) - Actually very similar, just a different ML approach
  - Shiono (2021) Uses Bayesflow, a Bayesian deep-learning approach that is likelihood-free.
- As explained, this is more of a guide of the types of methods that are currently being developed than a prescription to use one or the other.

The problem of computationally demanding models

$$Y = M(X, \theta)$$

- A rule-of-thumb definition for the 'hardest' possible models:
  1. No closed-form expression for the inverse problem $\hat{\theta} = M^{-1}(x, y)$.
  2. No analytical expression for the likelihood $p(y \mid \theta)$ or posterior $p(\theta \mid y)$. Non-parametric approaches require *simulating* the model.
  3. There is a compute constraint on the number of times the model can be run, (say $\approx$ 1000).

- Why bother with such models in the first place?
  - They exist because research expands to fill existing capacity (Parkinson's law!)
  - Assuming a reasonable calibration $\hat{\theta}$, the compute budget can be used to generate a Monte-Carlo ensemble and do scenario analysis

- **Is it possible to estimate such models within the restricted compute budget?**

Bayesian Estimation with Gaussian process Regression Surrogate (BEGRS)

- Combine Grazzini et al. (2017); Kukacka and Barunik (2017) and Lamperti *et. al* (2018)
  - A full Bayesian estimation framework with a surrogate-based likelihood function.
- Key insight is to train a surrogate model $f(\dots)$ not only on the parameter space, but also on lagged values of the observables, creating a one-step-ahead predictor:

$$m(y) = f(\theta) + \varepsilon \quad \rightarrow \quad y_t = f(\theta, y_{t-1}) + \varepsilon_t$$

- Benefits of the approach:
  - Large increase in the number of training points ($N \times T$ vs. N)
  - This allows the generation of a conditional likelihood function!

$$p(y \mid \theta) = \sum_t p(Y_t = y_t \mid \theta, y_{t-1})$$

Gaussian Process Regression Surrogate.

- Training the Gaussian process with simulated data x and estimating the model on y are separate phases.

$$x_t = \mathcal{GP}(\theta, x_{t-1}) + \varepsilon_t$$

$$Y_t \mid \theta, y_{t-1} \sim N(\mu_{x,y}, \Sigma_{x,y})$$

  - Where $\mu_{x,y}$ and $\Sigma_{x,y}$ are **functions** of the kernel covariances of x with y.

- Why use Gaussian Process Regression specifically?
  - GP with RBF covariance kernels are proven to be universal approximators. **So what?** So are NNs!
  - **Most importantly**: Bayesian learning of GP parameters protects against overfitting on the training data. Useful when training data is restricted!
  - The likelihood $L(\theta \mid y)$ has a Gaussian structure, giving a closed-form gradient $\nabla_\theta L(\theta \mid y)$. This allows the use of Hamiltonian Monte Carlo methods.

Some (non-technical) theoretical considerations on GPs

More technical details provided in my paper: Barde (2022)

Self penalising GP regression

- The objective function that is maximised in the GP training stage on the simulated data is:

$$\ln \hat{p}(\mathbf{Y}) = c - \frac{1}{2}\mathbf{Y}^T \left(\mathbf{K}^{\mathbf{x},\mathbf{x}} + \mathbf{\Sigma}^2\right)^{-1} \mathbf{Y} - \frac{1}{2} \ln \left|\mathbf{K}^{\mathbf{x},\mathbf{x}} + \mathbf{\Sigma}^2\right|$$

- This is equivalent to minimising the sum of the squared prediction error and a penalty term (called *regret*)
- If the Kernel is linear, it's actually equivalent to ridge regression on the training data!
- This protects against over-fitting on the training data - a major concern in ML!
- **Note**: This requires homoskedastic noise (or properly modelling the noise process separately).

Some (non-technical) theoretical considerations on GPs

LMC regression as a latent decomposition

- The surrogate is a Linear Model of Coregionalisation, using a linear combination of latent GP variables to model a multivariate system.
- This enables an efficiency gain if you use fewer latents than observables.
- More importantly, each latent GP kernel will have a different bandwidth, thus capturing correlations over different scales of the training data
- This allows very flexible modelling
- **Note**: The number of training observations required for the GP to converge is governed by the smallest bandwidth used.

Some (non-technical) theoretical considerations on GPs

Universal approximation without the design problems of Neural Networks

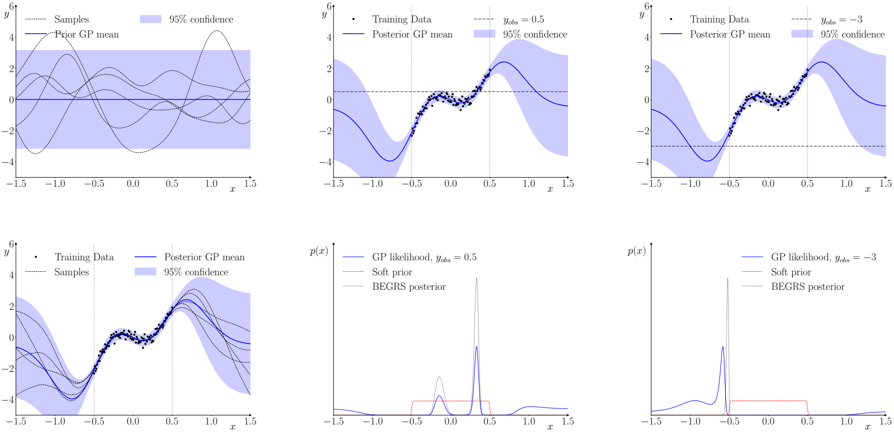
- GP theoretically of a single hidden layer NN of infinite width
- For NNs, the width is a hyper-parameter that needs tuning. GPs don't have that problem
- Cost: A GP is a single layer NN, so it is not as flexible as a Deep NN in terms of what it can learn. It will do badly on breaks/sharp transitions
- **Note**: Deep GP allows multiple GP 'layers' to be stacked, just like a NN, which limits the problem. BEGRS does not use this, as much more training data is required.

Minimal prior and identification

- BEGRS is **necessarily** a Bayesian estimation method. The surrogate likelihood is not enough
  - The GP prior is defined over the entire input space
  - The GP posterior converges on a compact subset (where there is training data.)
- You need a **minimal** prior to restrict the posterior to that subset
- It needs to be smooth at the boundaries so that gradients can be calculated
  - Advice is to use this following double logistic

$$p(\boldsymbol{\theta}) = \prod_i \frac{1}{1 + e^{-\alpha(\boldsymbol{\theta}_i - \underline{\boldsymbol{\theta}})}} \frac{1}{1 + e^{-\alpha\left(\overline{\boldsymbol{\theta}} - \boldsymbol{\theta}_i\right)}}$$

Pictures are worth a thousand words...



BEGRS pros

- Able to make the most of potentially limited computational resources.
- Self penalising training avoids overfitting the data
- Has the Universal approximation property.
- Closed-form expression for the suroggate likelihood gradient is beneficial for finding the mode of the posterior and sampling with HMC
- Separate training and empirical estimation phases means that the same surrogate can be reused over multiple data sources.

BEGRS cons

- Requires an external design (Sobol, Latin hypercube).
- Not as flexible as a Deep learning models, especially Neural Nets.
- Will struggle on models with sharp transitions, requiring a narrow Kernel bandwidth.
- This can (probably) be mitigated with a hybrid design (samples more from the transition zone), but would require sensitivity analysis.

Barde, Sylvain (2022) "Bayesian Estimation of Large-Scale Simulation Models with Gaussian Process Regression Surrogates," *School of Economics Discussion Papers*, Vol. 2203.

Grazzini, Jakob, Matteo G Richiardi, and Mike Tsionas (2017) "Bayesian estimation of agent-based models," *Journal of Economic Dynamics and Control*, Vol. 77, pp. 26–47.

Kukacka, Jiri and Jozef Barunik (2017) "Estimation of financial agent-based models with simulated maximum likelihood," *Journal of Economic Dynamics and Control*, Vol. 85, pp. 21–45.

Lamperti, Francesco, Andrea Roventini, and Amir Sani (2018) "Agent-based model calibration using machine learning surrogates," *Journal of Economic Dynamics and Control*, Vol. 90, pp. 366–389.

Platt, Donovan (2022) "Bayesian estimation of economic simulation models using neural networks," *Computational Economics*, Vol. 59, pp. 599–650.

Shiono, Takashi (2021) "Estimation of agent-based models using Bayesian deep learning approach of BayesFlow," *Journal of Economic Dynamics and Control*, Vol. 125, p. 104082.