MOCU for block-copolymer experimental design

Anthony DeGennaro

April 2019

General Problem Description

- We have a "real-life" physical system that maps feature inputs to outputs: $y = f_r(x)$. Because of either measurement noise and/or stochasticity in f_r (the latter of which could arise e.g. from physical processes in f_r that involve states/dynamics that we have not captured with x), the output of f_r given x may not be completely deterministic, and so we characterize the output of the real system with the conditional probability distribution $\rho_r(y|x)$.
- Because it is expensive to query this system, we have a computationally cheaper model that approximates its behavior: $\hat{y} = f_m(x, \theta)$, with $\theta \sim \rho(\theta)$. As before, if we wish to make this model non-deterministic, we may do so by constructing the probability distribution $\rho_m(y|x, \theta)$.
- The parameters θ capture any uncertainties in the model structure. If the model f_m is a physical model, then θ would consist of uncertain parameters appearing in the model dynamics. If instead the model is data-derived (e.g., POD-Galerkin, DMD/Koopman, spectral methods), then θ could simply capture statistical uncertainty in the weights/coefficients. As an example, if we had $f_m(\theta, x) = \sum_j \theta_j \phi_j(x)$ for some basis functions defined on x, then θ would simply represent the random coefficients of that expansion.
- We can think of θ as representing our ignorance in how the real system is related to the model system. That is, we assume that the real system *should* be described accurately by one of the candidate models represented by variations of θ, but we do not know which specific values of θ produce that agreement.
- Our goal w.r.t. operator design is to build a function $\psi_{IBR}(x, \theta) : X \times \Theta \mapsto Y$ from a family of functions Ψ that does the "best" job approximating the model f_m on average over the uncertainty θ. For example, we could use neural networks and think of Ψ as the space of neural networks with a certain structure and number of weights. We wish to find the optimally-robust mapping:

  $\psi_{IBR}(x) = \text{argmin}_{\psi \in \Psi} \mathbb{E}_\theta[C(\theta, \psi)]$

  where $C(\theta, \psi)$ is a cost function that quantifies the discrepancy between predictions made by ψ and f_m
- At the same time, we have model uncertainty *vis-a-vis* $\rho(\theta)$ that we wish to reduce by sampling the real system and updating our prior belief about θ to produce the posterior $\rho(\theta|D)$ through Bayesian inference. We don't want to select just any experiment though; we want to select that experiment $D = (x^*, y^*)$ that also reduces our cost. This leads (after "some algebra") to the MOCU framework for experiment selection:

  $x^* = \text{argmin}_{x \in X} \; \mathbb{E}_y[\mathbb{E}_{\theta|y}[C(\theta, \psi_{IBR}^{\Theta|x,y})]]$
- The nice thing about this framework is we are (1) designing experiments that respect an objective, (2) tuning a low-dimensional model to more accurately represent reality over the span of those objective-driven experiments, and (3) constructing a function that best represents the input/output mapping on average over all uncertainty, all in one shot.

Details Specific to Our Setting

1. **Ground Truth Source**
   - We should start with a computational model (Cahn-Hilliard) as our ground truth source. In the future, we will hopefully shift to considering actual experimental data for this purpose. However, for an initial proof-of-concept, we should start here.
   - The features $x \in X$ of Cahn-Hilliard are comprised of the material-specific parameters that appear in the dynamics. These parameters include the interface thickness parameter, the shape/form of the potential function, and other material constants. We will consult the materials-science literature in order to identify physically-meaningful ranges/distributions for these.
   - We may make the system non-deterministic by adding noise to the dynamics, i.e. $\dot{x} = \mathcal{C}(x) + \mathcal{N}$, where $N \sim \rho(\mathcal{N})$ is some noise profile and $\mathcal{C}(\cdot)$ simply denotes the (deterministic) CH dynamics.
   - We should begin by assuming known, fixed initial/boundary conditions, so as not to complicate things. If we want to consider a range of initial/boundary conditions, then probably we will have to incorporate these into the feature (experiment) space X via some parameterization.
   - W.r.t. numerics, we should probably use Danial Faghihi-Shahrestani's (UT) code. If we cannot do that, I (Anthony DeGennaro, BNL) have a 2-D spectral solver, although that would be non-ideal for a variety of reasons.
2. **Low-Dimensional Model**
   - The cheap model should be fitted prior to MOCU-based sampling using some k training data pairs $D_{train} = \{(x_1, \ldots, x_k), (y_1, \ldots, y_k)\}_{train}$, collected from Cahn-Hilliard. This model could be constructed in a variety of ways, depending on how we do things. POD/POD-Galerkin would be classical choices, and DMD/Koopman methods would be an interesting alternative. Karen Wilcox (UT) and Anthony DeGennaro (BNL) could investigate these and other approaches.
   - We should fit a "mean" model to the training data: $\hat{y} = f_m(x, \theta_{fit})$, where θ_{fit} represent some weights (or coefficients) associated with the model fit
   - To account for model imperfections etc., we can "fuzzify" the model with uncertainty and consider the parameterized class of models $\hat{y} = f_m(x, \theta + \mathcal{N})$ with $\theta \sim \rho(\theta)$ and $\mathcal{N} \sim \rho(\mathcal{N})$. θ accounts for uncertainty in the model structure; $\mathcal{N}$ is just non-deterministic noise that makes the system stochastic.
   - $\rho(\theta)$ should be based on our prior expectations. For example, the mean value should be at θ_{fit}. If we are using a POD-based or spectral type method, then we might also expect exponential decay in the variance of coefficients for higher-order modes.
3. **Intrinsically Bayesian Robust Operator**
   - We should use some sort of regressor for $y = \psi(x, \theta)$, e.g. a fully-connected neural network
   - The difference between $f_m(x, \theta)$ and $\psi(x, \theta)$ is that the computational model is a low-dimensional model that has been trained to approximate the physics, whereas ψ is just a function that maps (x, θ) to y. For example, if we use POD for f_m, then we have $\dot{\hat{y}} = \sum_j \theta_j \phi_j(x) + \mathcal{N}$ and we will still have to drive the approximate system dynamics to steady-state to get $\hat{y}$, whereas ψ just gives a direct mapping. Also note that in the MOCU machinery, we will need to compute $\psi(\Theta|(x, y))$, which is the optimal regressor that approximates f_m given (x, y), for all combinations of $(x, y) \in X \times Y$. This will result in a different robust operator for each pair of (x, y)
   - Obtaining ψ could be done in the usual way, e.g. training a neural net on a set of data generated by the ROM. For example, to approximate $\psi(\Theta|(x, y))$, we would train a neural network on a subset of k data points generated from the ROM using $(x, y; \theta_1 \ldots \theta_k)$
4. **MOCU Methodology**
   - Ed Dougherty and Guang Zhao (A&M) have recently done a derivation showing how the MOCU sampling formula reduces from the general form presented in these notes to something else by marginalizing over Θ, under mild assumptions about X, Y, Θ, Ψ. As far as I can tell, these assumptions are perfectly valid and I defer to their presentation/algorithm for specific details.