Hi Dev,

Thank you so much for the kind words and your thoughtful question — I really appreciate it.

I completely agree that the idea of a two-stage architecture offers a compelling and practical solution, especially when deploying these models in a real-time or production environment. In our experience, the need for such a setup depends largely on the flexibility of the model with respect to market parameters.

When the model is trained to take all key parameters — such as interest rate and volatility — as input features, it gains the ability to generalize across a wide range of market conditions. In these cases, a single, well-trained model can often accommodate daily fluctuations, making a second stage unnecessary.

However, not all methods support such parameterized training. Some solvers still require the PDE to be defined with fixed parameter values. For these cases, your suggested two-stage setup becomes particularly effective. One practical approach we've explored is to use the model trained on yesterday’s market parameters as a warm start. Since market conditions typically evolve gradually, the difference between yesterday’s and today’s inputs is often small. This allows us to fine-tune the pretrained model using today’s parameters, enabling much faster convergence than training from scratch.

When the market changes more significantly — for instance, a large shift in interest rate or implied volatility — we can instead pretrain a library of models across a grid of plausible parameter values. At runtime, we can select the closest match to current conditions as the base model and retrain from there. This hybrid strategy balances accuracy and efficiency, making it well-suited for intraday updates or frequent recalibrations.

I truly appreciate your question — it reflects exactly the kind of practical integration we’ve been thinking about.

Warm regards,  
Annie Zhang