Does BOAT perform well in the real world?

1 Is SBO actually needed?

2 Priors' quality on performance?

3 Can we trust it?

What does the developer actually write?

01 A prior

The allocation rate is unknown: somewhere between 0 and 5000 MB/s.

02 A trend

The developer's belief: GC rate falls as the eden space grows.

03 Inference, done for you

observe() fits every unknown to the GC logs; predict() returns distributions.

```
struct GCRateModel : SemiParametricModel {
  GCRateModel() {
    // 1. Prior on the unknown
    alloc_rate = uniform(0, 5000)(gen);
  }
  // 2. Parametric trend
  double parametric(double eden) {
    return alloc_rate / eden;
  }
};  // 3. A GP learns what's left
```

Based on Listing 1 of the paper (simplified)

Is it more than a GP?

	Plain GP	BOAT	General language (Pyro, Stan)
Model shape	One GP, input to output	Formula + GP per node, fixed DAG	Any code you write
Inference	Closed-form GP formula	Each node fitted separately	General engine, e.g. sampling
Uses the logs	No	Yes, if observed	Yes

> BOAT sits closer to a GP with a hand-written mean than to a general language.

Is the modelling effort worth it?

2 vs 16

Iterations to find a good GC configuration when compared to Spearmint and OpenTuner.

2×

Faster neural network schedules when compared to generic tuners which were found after 30 iterations

What did writing the model cost?

Creating the prior model can take time and experience which the paper does not measure

BOAT paper, Sections 6.1 and 6.2

Could a cheaper prior do as well?

	πBO (ICLR 2022)	BOAT
What you give	A belief about where the optimum is	A program of how the system works
Effort	Low: one distribution	High: write and debug a model
Uses the GC logs	No	Yes
If the knowledge is wrong	Its influence fades; still converges	No guarantee

How much knowledge do developers actually have: hunches or models?

Hvarfner et al., πBO, ICLR 2022 (not probabilistic programming)

Can we trust a wrong model?

01

A bad prior can block a region

If the model predicts poor performance in a region, the optimiser may never look and find the actual optimum.

02

Missing baselines

No comparison with REMBO (2013) or additive GPs (2015), which handle many parameters without a developer model.

Only compared to Spearmint and Generic Tuners

03

Later work adds the guarantee

Function Networks (2021) keeps the structure and proves convergence.

Is fast but unguaranteed OK for production tuning?

To discuss

Is BOAT's real contribution the abstraction, not the algorithm?

Probabilistic programs make system knowledge readable: the whole GC model is about 15 lines of C++. Underneath, it's still Bayesian optimisation over GPs.