### Background

Model fitting $\Rightarrow$ optimization problem

- Models in science (e.g., comput. neuroscience)
  - Moderately costly evaluation (0.1 $\sim$ 10 s)
  - Typical budget $10^4$ – $10^6$ func. evaluations.
- Noise or roughness of the likelihood landscape
- Gradients often unavailable

What about Bayesian optimization (BO)? [1]

- Great at optimizing very costly black-box functions.
- Relatively unused in scientific fields
  - Too slow, typical budget < 100 func. evaluations.
  - Unknown performance vs. other optimizers [2]
  - Might require additional technical knowledge

### Contributions of this paper

- A novel hybrid BO algorithm, Bayesian Adaptive Direct Search (BADS) [3]
- Tested BADS & 16 optimizers on both artificial functions and real datasets and models
- Released BADS as a free MATLAB toolbox
  - What about Python? See Conclusions

### Key Ideas

- BADS follows mesh adaptive direct search [4,5]
  - Alternates POLL and SEARCH stages
- Keeps local Gaussian process (GP) approximation of the objective function
- POLL: ‘dumb’ near model-free direct search
  - Rescale poll vectors by GP length scales
- SEARCH: ‘smart’ local search via BO
  - Fast local optimization of the acquisition function via CMA-ES inspired search
  - Multiple local search covariance matrices (chosen based on performance via heuristics algorithm)
  - Aggressive: SEARCH goes on until BO fails to find a better optimum several times
- POLL provides failsafe method when SEARCH via BO stops working (e.g., due to bad GP model)

### Optimization algorithms

- BADS $+$ 16 other optimizers in MATLAB
  - Popular methods, e.g. fminsearch, fmincon
  - Competitive optimizers, e.g. mcs and CMA-ES
  - Vanilla Bayesian optimization (bayesopt)
  - As [6] but w/ hyperparameter optimization
  - Noise-aware snobfit and noisy CMA-ES
- For all methods, default settings (no fine-tuning)
- 50 runs of each algorithm on each test function

### Problems

- Artificial functions. (BB8089; not shown here, see [3])
  - Both deterministic and noisy functions.
- Total 288 test functions with $D \in [3, 6, 10, 15, 20]$
- Six studies in comput. neuroscience (ccn17)
  - Six real datasets (subjects or neurons) per study
  - Optimize log likelihood for given models
- Total 36 test functions with $D \in [6, 9, 10, 12, 13]$

### Conclusions

- BADS generally outperforms other optimizers
- Second best: CMA-ES or fmincon, it depends
- Vanilla Bayesian Optimization fails miserably

When should BADS be used?

- Problems up to $D \sim 15$
- Noisy or jagged func. landscape
- Model evaluation $\gtrsim 0.1$ s

Future directions

- Port BADS to other languages (e.g., Python)
  - Interested in helping? Let’s talk!
- Check alternatives to GPs
- Combine with smart multi-start method
- Support for variable and tunable precision [7]
- Go from heuristics to approximate inference

### References