Constrained Bayesian Optimisation with Knowledge Gradient

Juan Ungredda

University of Warwick

Constrained KG

Non-constrained KG:

$$KG(x) = \mathbb{E}[\max_{x'}\{\mu^{n+1}(x')\}|x^{n+1} = x]$$

Constrained KG with Probability of Feasibility (pf):

$$cKG(x) = \mathbb{E}[\max_{x'}\{pf^{n+1}(x')\mu^{n+1}(x')\}|x^{n+1} = x]$$

Constrained KG corrected*:

$$cKG(x) = \mathbb{E}[\max_{x'}\{pf^{n+1}(x')\mu^{n+1}(x') + (1 - pf^{n+1}(x'))M\}|x^{n+1} = x]$$

where $M \in \mathbb{R}$ is the penalisation for sampling points in an infeasible region. It's commonly assumed to be zero.

Constrained KG

Benefits

- Takes into account that constraints change for each possible x^{n+1} considered.
- Assuming $M = 0$ may give "benefit" to infeasible regions. A more general approach avoids that problem.

Limitations

- Computationally expensive compared to constrained Expected Improvement. However, it's possible to make an efficient implementation by using gradients.
- M may be need to chosen by the decision maker.

Results

benchmark Method:

- Constrained Expected Improvement. i.e. Expected Improvement times Probability of feasibility.

Test Functions:

- New Branin, test function 2, and Mistery function. Non-noisy and Noisy objective functions with unknown constraints.

Important Remark:

- In the last evaluation of our approach, the function is sampled according to Expected Improvement times the probability of feasibility.

Performance Evaluation

Given sampled data $D^N = \{(x_0, y_0), \ldots, (x_N, y_N)\}$. The recommended design is given by the max value of the Gaussian Process approximation μ^N.

$$x_r = \max_{D^N} \mu^N(x)$$

Performance is measured as difference in true value y_{true} (if it's noisy) between best true design x^* and recommended sampled design (x_r).

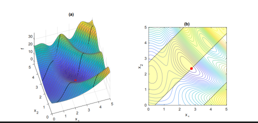
$$Performance(x_r) = y_{true}(x_r) - y_{true}(x_r)$$

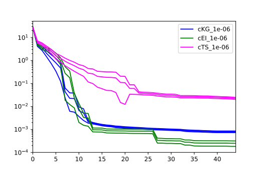
For real experiments performance was measured as

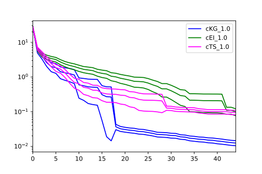
$$Performance(x_r) = y_{true}(x_r)$$

PLOTS

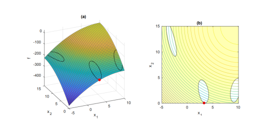
Mistery

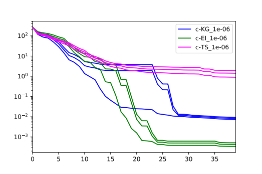


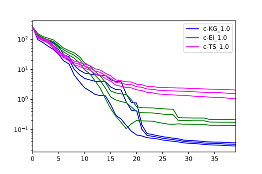




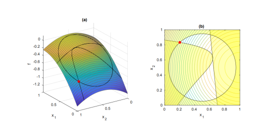
New Branin

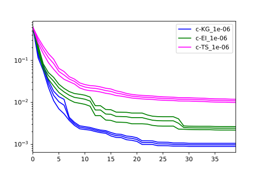


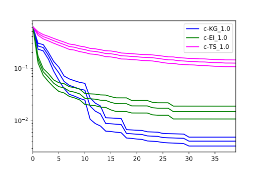




Test function 2

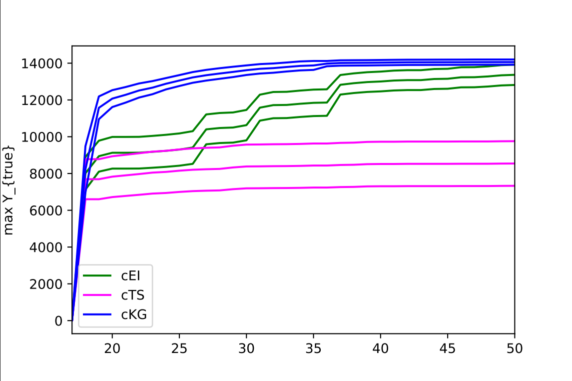






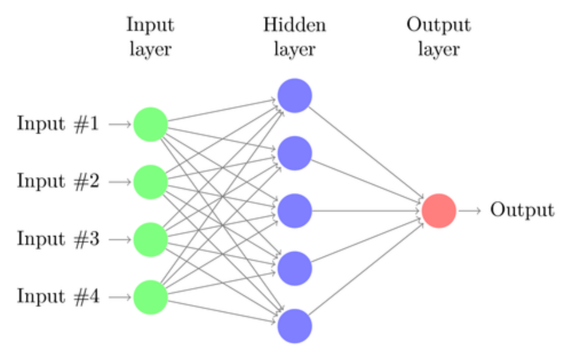
Real Experiments (Multi-Stage Revenue Management with Inter-Temporal Dependence)

A businessman chooses to buy b ¿ 0 units of capacity, paying c ¿ 0 dollars per unit of capacity at $t = 0$.During stage $t(t = 1, \ldots, T)$ he observes demand D_t for units at price p_t, at which point, he must choose to sell x_t units ($0 \leq xt \leq D_t$), provided that the total number of units sold (accross all past periods) does not exceed b.



Real Experiments (Tuning a Fully Connected Neural Network)

I tune the hyperparamters of a two-hidden-layer neural network subject to the constraint that the prediction time must not exceed 0.01s. The search space consists of 4 parameters: 2 dropout parameters, and the number of hidden units in each layer.



Gradient Information

- Numerical methods to optimise KG rely on ∇KG. Quantity can be obtained either by Finite Difference or Analytically.

$$\nabla cKG(x) = \nabla \mathbb{E}[\max_{x'}\{pf^{n+1}(x')\mu^{n+1}(x')\}|x^{n+1} = x]$$

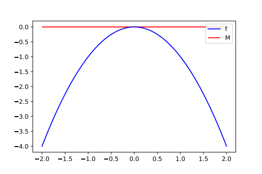
Once solved, the inner optimisation problem,

$$\nabla cKG(x) = \mathbb{E}[\nabla\{pf^{n+1}(x^*; x^{n+1})\mu^{n+1}(x^*; x^{n+1})\}|x^{n+1} = x]$$

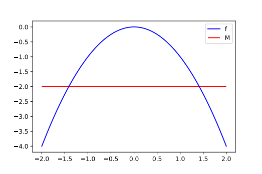
Potential issue with Finite Differences:

- Since the expectations is calculated only over a few values slight deviations of x^* affect greatly the outer gradient estimation.

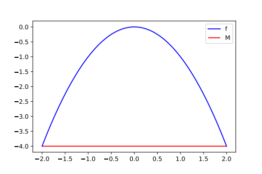
Dynamically change of M penalization.



(a)



(b)



(c)

Dynamically change of M penalization (Formulation)

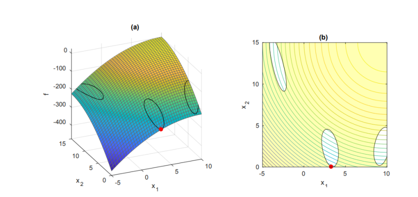
$$cKG(x) = \mathbb{E}[\max_{x'}\{pf^{n+1}(x')\mu^{n+1}(x') + (1 - pf^{n+1}(x'))M\}|x^{n+1} = x]$$

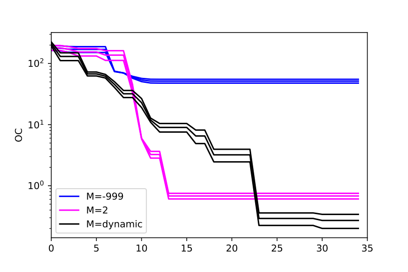
$$M = (\mu(x) - \mu^*)\sum_i \mu_{c_i}(x)$$

- $(\mu(x) - \mu^*)$ penalises values with higher negative values $\mu(x)$ far from estimated optimum μ^*)
- $\sum_i \mu_{c_i}(x)$ penalises regions far from feasible area.

Result plots

Constrained Brannin function using dynamic change of penalisation on EGO. Comparison against highly penalised infeasibility (M=-999), and encouraged infeasibility (M=2).





Multi-Objective Constrained Optimisation

- Multi-Objective formulation using linear scalarisation.

$$KG(x';\theta) = \mathbb{E}_y[\max_{x'} \theta\mu^{n+1}(x')|x^{n+1} = x', \theta]$$

where the policy is,

$$maKG(x') = \mathbb{E}_\theta[KG(x';\theta)]$$

- Multi-Objective constrained formulation

$$KG(x';\theta) = \mathbb{E}_y[\max_{x'} \theta\mu^{n+1}(x')pf^{n+1}(x')|x^{n+1} = x', \theta]$$

where the policy is,

$$maKG(x') = \mathbb{E}_\theta[KG(x';\theta)]$$

Work to do

Main line of work

- Constrained Multi-Objective problem