Studies in Hyperparameter Tuning, Design Selection and Optimization

Samuel Onyambu
Advisor: Hongquan Xu

Department of Statistics
UCLA

November 19, 2024

Table of Contents

Table of Contents

Introduction

- Optimization algorithms play a pivotal role in tackling complex real-world problems.
- Identifying optimal solutions requires efficient exploration of large and often high-dimensional search spaces.
- Generation of Optimal Uniform Projection Designs falls under optimization problem
- Issues?
    - Discrete Search Space
    - Non-convex Optimization
- Methods?
    - Modified Differential Evolution
    - Bayesian Optimization

Why DE

- DE has simplistic nature
- Strong global search capabilities
- Robust performance
- Flexibility in a range of applications
- High convergence speed for certain problems
- parallelizable

Table of Contents

Why UniPro

- When only a subset of the input variables are active, uniformity of projected designs in low-dimensional spaces is important.
- UniPro have good space-filling properties not only in two dimensions, but also in all dimensions. (Sun, Wang, and Xu 2019)
- Uniform projection designs are robust and perform well under other design criteria eg the maxPro criteria. (Sun, Wang, and Xu 2019)
- No optimal construction method(s) available.

DE Algorithm

- **Genetic Representation** Let $\pi_1, \cdots, \pi_N$ be the initial population where each agent $\pi_i = (\pi_{i1}, \cdots, \pi_{im})$ is randomly chosen from Ω
- **Mutation** Produce a potential donor.

$$\nu_i = \pi_a + \omega(\pi_b - \pi_c)$$

- **Crossover** Blend the current generation of agents with the population potential donors.

$$\mu_{ij} = \begin{cases} \nu_{ij} & \text{With probability pCR or if } j = j_0 \\ \pi_{ij} & \text{otherwise} \end{cases}$$

where $j = 1, \cdots, m$

- **Selection** Adopt the trial agent if it leads to an improvement.

$$\pi_i = \begin{cases} \mu_i & \text{if } h(\mu_i) > h(\pi_i) \\ \pi_i & \text{otherwise} \end{cases}$$

- **Repeat** Repeat steps 2-4 over many generations.

DE Algorithm

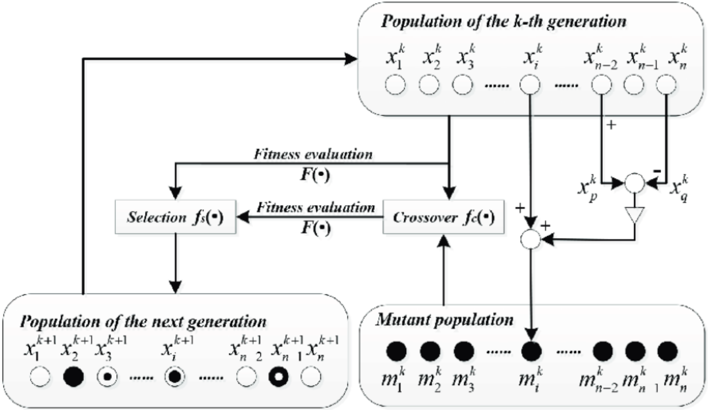


Figure: outline of DE algorithm

Modified DE

- DE is effective for continuous tasks but needs modification for discrete problems.
- Stokes, Wong, and Xu (2024) Proposes modification in the mutation step as:
  1. with probability pGBest, choose between the global best design and a random design
  2. Generate mutant population by perturbing the columns of the chosen design:

$$\nu_i = \text{Perturb}(\pi_a), \tag{1}$$

where

$$\text{Perturb}(\pi_a) : \text{randomly swap 2 elements in } \pi_{aj} \text{ with probability pMut}$$

Modified DE

Hyperparameters

- **Population Size** - *NP* The size of the population to chose from, with domain chosen as [10, 100].
- **Itermax** The maximum number of iterations. Take the values between [500, 1500].
- **Probability of CrossOver** - *pCR* A value between 0.05 and 0.95 that determines whether to use a variable from the v_i or form π_i.
- **Probability of mutation** - *pMut* A value between 0.05 and 0.95 which determines as to whether a given element in the trial is swapped with another within the same variable.
- **Probability of using global best** - *pGBest* A value between 0.05 and 0.95 that determines whether to use a random agent or the global optimal agent as the trial agent.

Objective

- Objective: Understand the DE hyperparameter surface.
- Why? DE performance depends on the hyperparameter settings
- How? Tuning the DE hyperparameters Using the UniPro Criterion as the objective function to be minimized.
- The UniPro criterion:

$$\phi(D) = \frac{2}{m(m-1)} \sum_{|u|=2} CD(D_u)$$

where CD is the centered L_2-discrepancy and defined as:

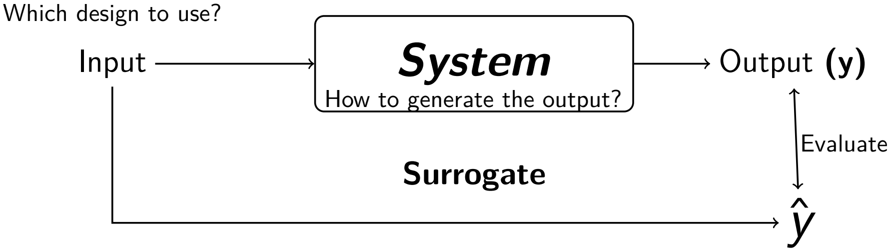
$$\begin{aligned}\mathrm{CD}(D) =& \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \prod_{k=1}^{m} \left(1 + \frac{1}{2}\left|z_{ik}\right| + \frac{1}{2}\left|z_{jk}\right| - \frac{1}{2}\left|z_{ik} - z_{jk}\right|\right) \\ & - \frac{2}{n} \sum_{i=1}^{n} \prod_{k=1}^{m} \left(1 + \frac{1}{2}\left|z_{ik}\right| - \frac{1}{2}\left|z_{ik}\right|^2\right) + \left(\frac{13}{12}\right)^m,\end{aligned}$$

where $z_{ik} = (2x_{ik} - s + 1)/(2s)$

How to Tune

Issues

- Which design to use? ie $\boldsymbol{X}$
- How to collect data? ie $\boldsymbol{y}$
- How do we model the data?
- How do we evaluate the data?



Data Generation

Obtain $\boldsymbol{X}$: Use the designs to set the parameters.

- Train Data:
  - Factorial Designs – Physical Experiments
    - 43-run Central Composite Design (CCD)
    - 50-run Orthogonal Array Composite Design (OACD)
  - Space Filling Designs – Computer Experiments
    - 50-run Latin Hypercube Design (LHD)
    - 50-run Maximin Distance Design
    - 50-run Maximum Projection Design (MaxPro)
- Test Data:
  - 3^5 & 4^5 Full Factorial Design
  - 243 & 1024 random Latin Hypercube Design

Obtain $\boldsymbol{y}$: Set the UniPro target size to be considered.

```
UniPro(n, m, levels, NP, itermax, pMut, pCR, pGBest, replicates)
```

Modeling and Evaluation

- Second order linear model: Mainly for physical experiments

$$\boldsymbol{y} = \beta_0 + \sum_{i=1}^{n} \beta_i \boldsymbol{x}_i + \sum_{i=1}^{n} \beta_{ii} \boldsymbol{x}_i^2 + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \beta_{ij} \boldsymbol{x}_i \boldsymbol{x}_j + \epsilon$$

- Kriging Model: Mainly for computer experiments (deterministic)

$$\boldsymbol{y}(\boldsymbol{x}) = f(\boldsymbol{x}) + Z(\boldsymbol{x})$$

  where $f(\boldsymbol{x})$ is a deterministic trend and $Z(\boldsymbol{x})$ is a GP with mean zero and stationary covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$

- Heterogeneous Gaussian Process: Computer experiments with error – need replicates

$$\boldsymbol{y}(\boldsymbol{x}) = f(\boldsymbol{x}) + Z(\boldsymbol{x}) + \varepsilon_i$$

  where $\varepsilon_i \sim \mathcal{N}(0, r(\boldsymbol{x}_i))$.

- **Evaluation:** Use RMSE to evaluate the designs.

Figure: Comparison of RMSE with target size 30×3

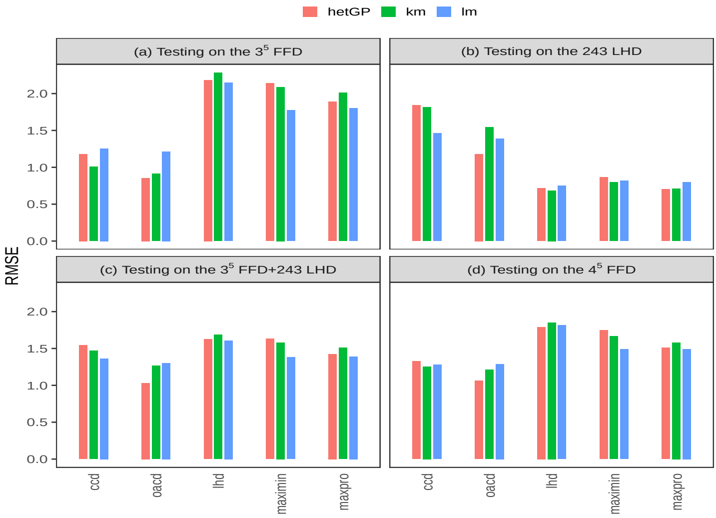


Figure: Comparison of RMSE with target size 70×7

Results

- The performance of the training data set depends on the nature of the testing data set.
- The composite designs, CCD and OACD, seem to be better when tested on the 3^5 FFD.
- The space filling designs (random LHD, maximin LHD, and maxpro LHD) did better when tested on the 243-run random LHD.
- OACD robust over CCD.
- Thus: Use OACD for hyperparameter initialization.
- Models? There seems to be no striking observation to be made as to whether one fitting method performs better than the other two, with exception for one 30×3 case when the kriging model fitting to the CCD training data had a much higher RMSE value than the other cases.

Factor Importance

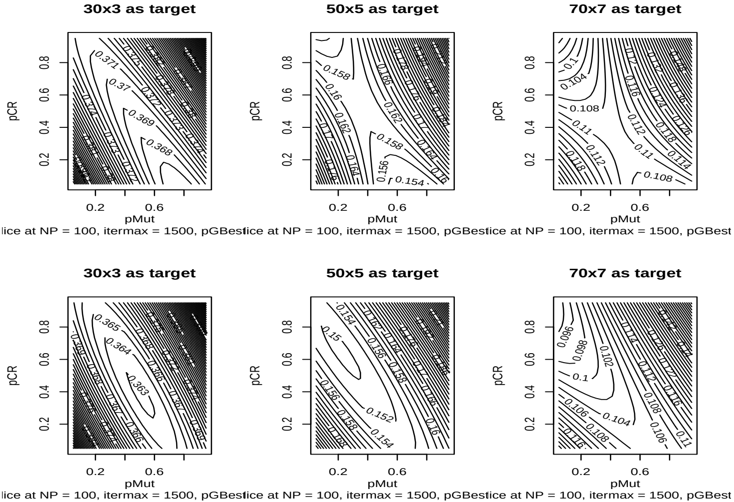
	ccd3_43	oacd3_50	maximin_50	maxpro_50
(Intercept)	0.3815***	0.3876***	0.3878***	0.3814***
NP	−0.0134***	−0.0143***	−0.0042***	−0.0055*
pMut	0.0020	0.0028	0.0043***	0.0045*
pGBest	−0.0204***	−0.0224***	−0.0047***	−0.0082***
pCR	−0.0022	−0.0030	0.0004	0.0007
itermax	−0.0146***	−0.0152***	−0.0046***	−0.0021
itermax_q	0.0090	−0.0081	0.0011	0.0057
NP_q	0.0119	0.0080	0.0022	−0.0004
pCR_q	0.0071	0.0074	−0.0008	0.0049
pGBest_q	0.0083	0.0192*	0.0035	0.0159***
pMut_q	0.0150	0.0174*	0.0083***	0.0104*
NP:pMut	0.0058	0.0064*	−0.0002	−0.0037
NP:pGBest	−0.0041	−0.0038	0.0017	0.0006
NP:pCR	0.0002	−0.0007	−0.0006	−0.0002
NP:itermax	0.0049	0.0033	0.0023	0.0052
pMut:pGBest	−0.0080*	−0.0093***	−0.0089***	−0.0139**
pMut:pCR	0.0203***	0.0197***	0.0042**	0.0018
pMut:itermax	0.0032	0.0025	−0.0061***	0.0004
pGBest:pCR	0.0034	0.0036	0.0000	0.0051
pGBest:itermax	0.0057	0.0068**	0.0076***	0.0058
pCR:itermax	0.0037	0.0021	0.0018	0.0040
R^2	0.9034	0.9235	0.8970	0.7297
Adj. R^2	0.8157	0.8708	0.8260	0.5433
Num. obs.	43	50	50	50

*** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$

Results

- The model obtained from using the maxpro LHD training data is the worst performing.
- Does not capture important main effects.
- The model obtained by using OACD performs the best. It has an adjusted R^2 of 0.87. Captures important main effects and interactions.
- CCD might be a little worse than OACD because of the fewer number of points (43) used for training compared to the other models which used 50 points.
- The model from the CCD does not identify any of the quadratic effects to be significant while the other modes do.
- The main effects of three hyperparameters (NP, itermax and pGBest) are very significant. Should be set at the maximum level

Contours



width

Figure: Contour plots of pMut and pCR while fixing other hyperparameters at high levels. Top row uses CCD as the training data; bottom row uses OACD as training data.

Table of Contents

- Objective: Which space-filling designs offer greater efficiency and robustness for prediction and Optimization
- Designs Considered?
    - Maximin Design
    - MaxPro Design
    - Latin Hypercube Design
    - Uniform Projection Design
    - Uniform Design (Only for Prediction)

Data Generation, Modeling and Evaluation

- Train Data: 64×15, 128×31 designs scaled to $[0 - 1]$
- Test Data: 10,000-run LHD.
- Modeling: Kriging Model using Matérn kernel function.
- Evaluation: Using Normalized RMSE:

$$\text{Normalized RMSE} = \left[\frac{N^{-1} \sum_{i=1}^{N} \{\hat{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)\}^2}{N^{-1} \sum_{i=1}^{N} \{\bar{y}_{train} - y(\boldsymbol{x}_i)\}^2} \right]^{1/2},$$

- N $= 10000$
- 200 Replications

Figure: Borehole - 8D

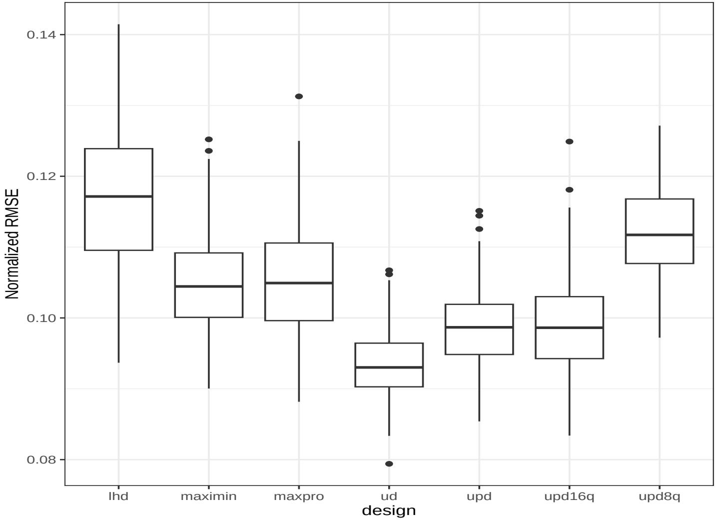


Figure: Gfunction - 9D

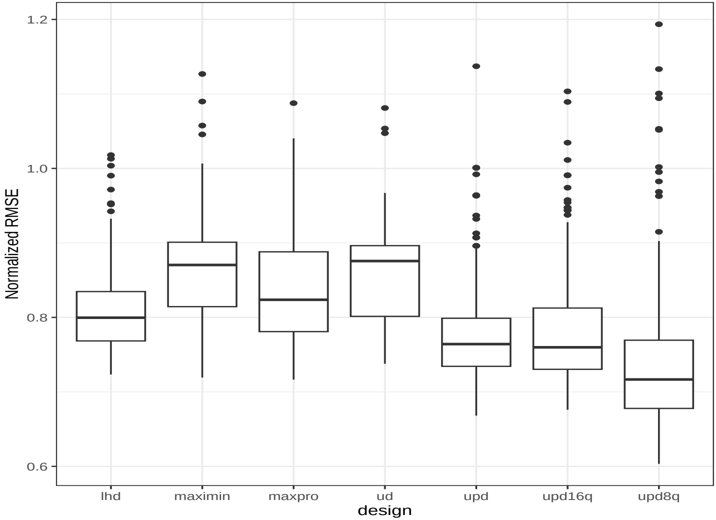
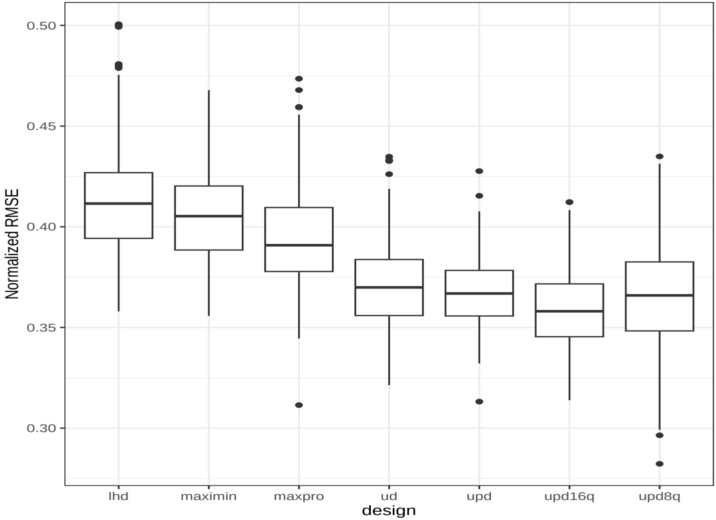


Figure: 4D Rosenbrock



Expected Improvement

- Use the Expected Improvement (EI) acquisition function formally given as

$$\mathrm{E}[I(\mathbf{x})] \equiv \mathrm{E}\left[\max\left(f_{\min} - Y, 0\right)\right]$$

- When $Y \sim N(\hat{y}, s^2)$ then closed formed is given as

$$\mathrm{E}[I(\mathbf{x})] = \underbrace{(f_{\min} - \hat{y})\,\Phi\left(\frac{f_{\min} - \hat{y}}{s}\right)}_{exploitation} + \underbrace{s\phi\left(\frac{f_{\min} - \hat{y}}{s}\right)}_{exploration}$$

  where $\hat{y}$ is the predictor and s is the standard error at point $\mathbf{x}$, and $\phi(\cdot)$ and $\Phi(\cdot)$ are normal pdf and cdf respectively. (Jones, Schonlau, and Welch 1998)

- Gives rise to the efficient global Optimization algorithm (EGO)

Algorithm EGO algorithm

Require: $\boldsymbol{X}$, $f =$ function to be minimized, n_{new}=number of points to add

```
Evaluate f at the design points X; y = f(X)
Build a kriging model based on X and y
for i in 1 to n_new do
    Find x* ← arg max_x E[I(x)]
    Evaluate y* ← f(x*)
    X = X ∪ x* and y = y ∪ y*
    Update the Kriging Model
end for
Return X, y
```

EGO 1d Elaboration

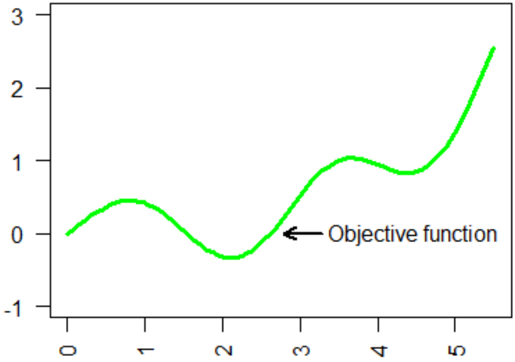
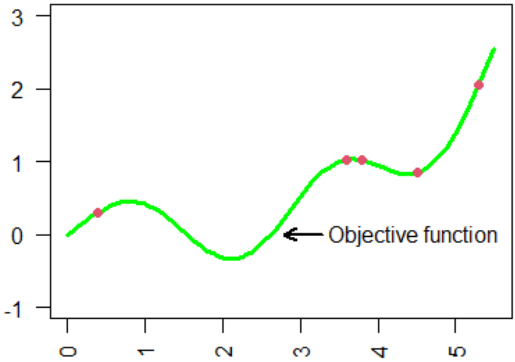
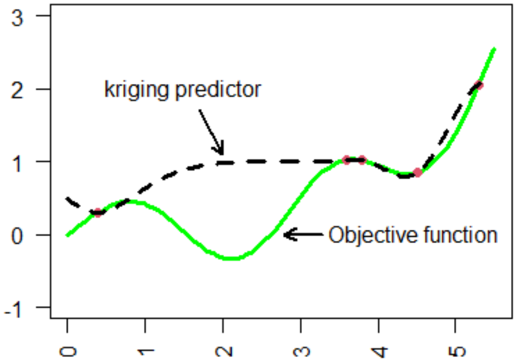


Figure: $0.01x^{3.2} + 0.9cos(x^{1.1})sin(x^{1.1})$

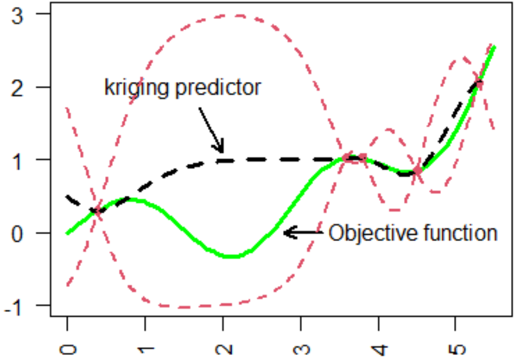
EGO 1d Elaboration



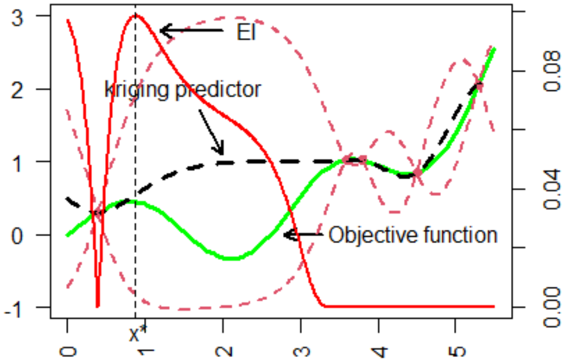
EGO 1d Elaboration



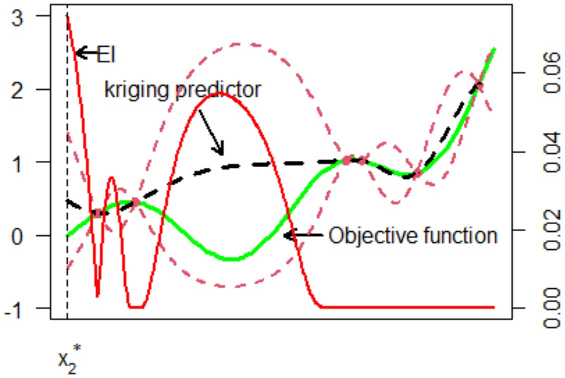
EGO 1d Elaboration



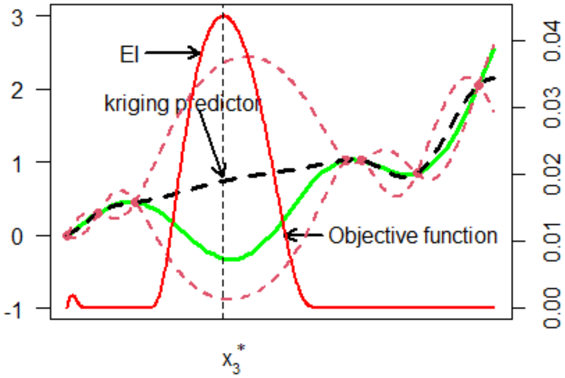
EGO 1d Elaboration



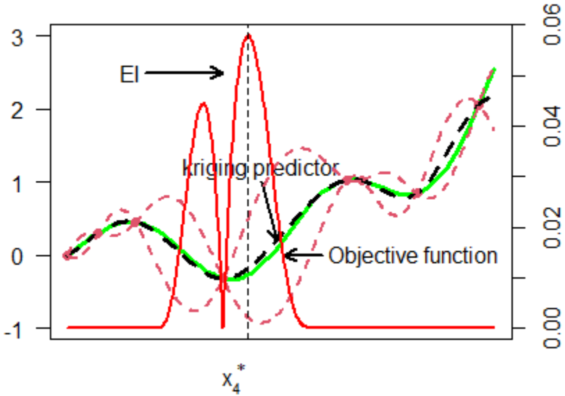
EGO 1d Elaboration



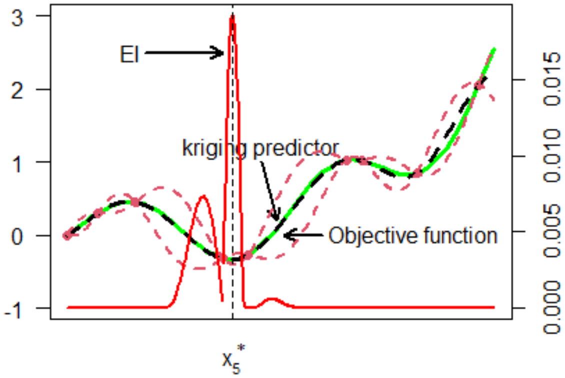
EGO 1d Elaboration



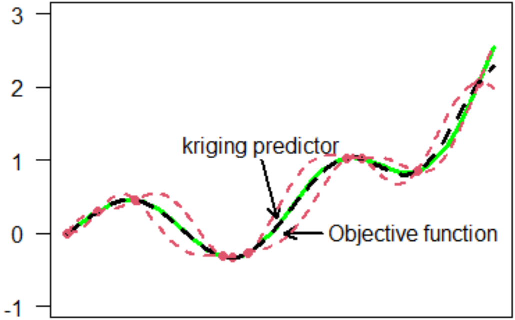
EGO 1d Elaboration



EGO 1d Elaboration



EGO 1d Elaboration



Initial Design Minimization Path

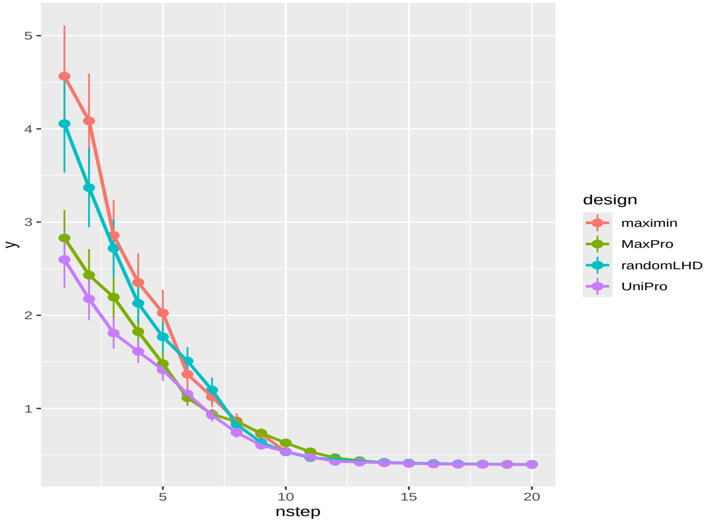


Figure: Branin (2D)

Initial Design Minimization Path

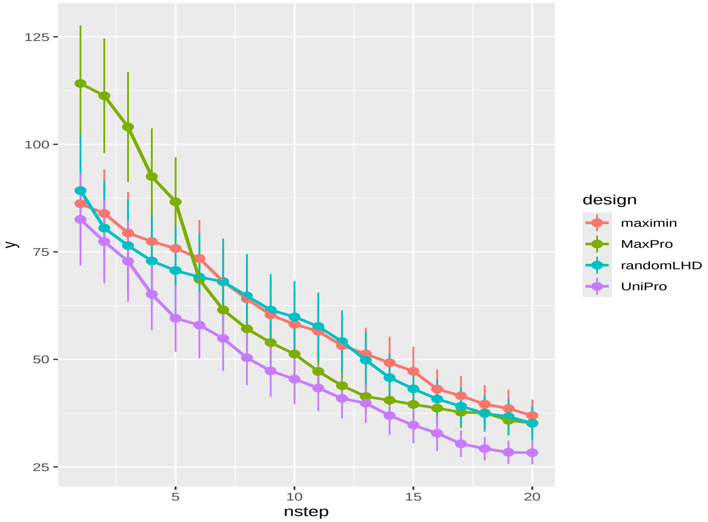
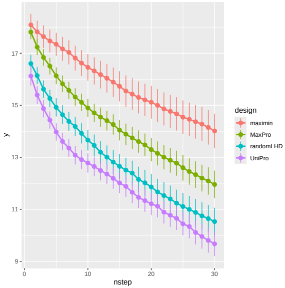
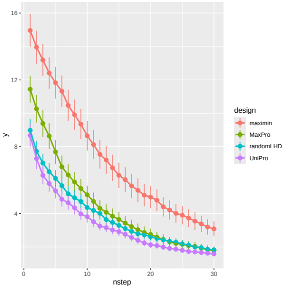


Figure: Goldstein-Price (2D)

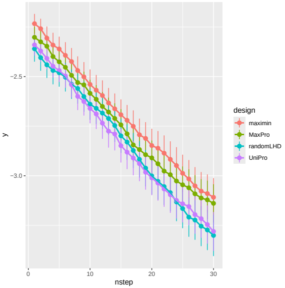
Initial Design Minimization Path



6D Ackley

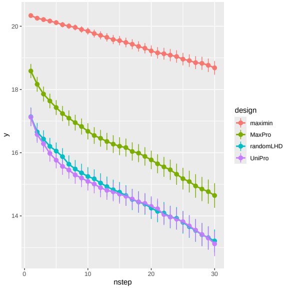


6D Levy

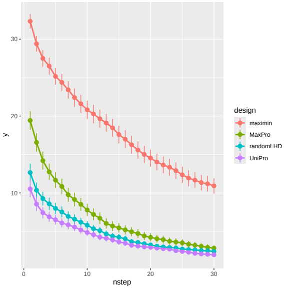


6D Michalewicz

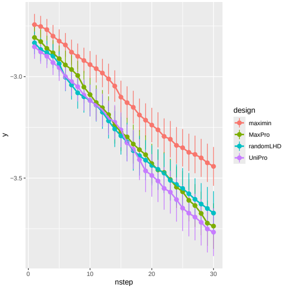
Initial Design Minimization Path



8D Ackley



8D Levy



8D Michalewicz

Table of Contents

Efficient Global Optimization – Limitations

- Slow Convergence as each iteration is over the entire search space
- Struggles to scale with dimension

Objective

Propose a method that focuses on a specific region of interest within the entire domain space in each iteration. Use the proposed method to optimize the UniPro construction.

Kriging based Region shrinkage for HPO

Algorithm Sequential Region Shrinkage Method

```
Require: X, f, Ω = domain, n_new, ρ
Evaluate f at the design points X; y ← f(X)
Build a Kriging model based on X and y
Set the region of interest (ROI) Ω' ← Ω (initial domain)
while stopping criteria not met do
    Run EGO within the domain Ω' to obtain the next n_new points
    Update X and y with the new points
    Update the Kriging model
    Determine the new ROI Ω' ← ROI(X, y, ρ)
    if small or no improvement (unsuccessful iteration) then
        Restore the original domain: Ω' ← Ω
    end if
end while
Return X, y
```

Algorithm Determine Region of Interest (ROI)

Require: $\boldsymbol{X}$, $\Omega =$ domain, $\boldsymbol{y}$, ρ.

1: $\tau \leftarrow$ indices of the top $100\rho\%$ of $\boldsymbol{y}$.
2: Select points in $\boldsymbol{X}$ associated with τ.
3: Find the best point $\boldsymbol{x}^*$ that minimizes the objective function
4: Determine the lower and upper bounds for points associated with τ; set $L_j := \min_{i \in \tau} x_{ij}$ and $U_j := \max_{i \in \tau} x_{ij}$ for each $j = 1, \ldots, d$
5: Set $\mathbf{D} := (\mathbf{U} - \mathbf{L})/2$, where $\mathbf{U} = (U_1, \ldots, U_d)$ and $\mathbf{L} = (L_1, \ldots, L_d)$.
6: Set $\Omega' = \{\boldsymbol{x}^* - \mathbf{D}, \boldsymbol{x}^* + \mathbf{D}\} \cap \Omega$.
7: **return** Ω'.

Example of ROI Determination Using Branin

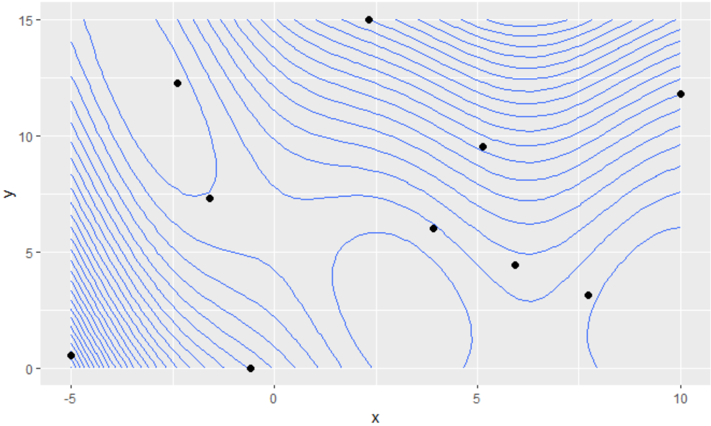


Figure: Initial design points

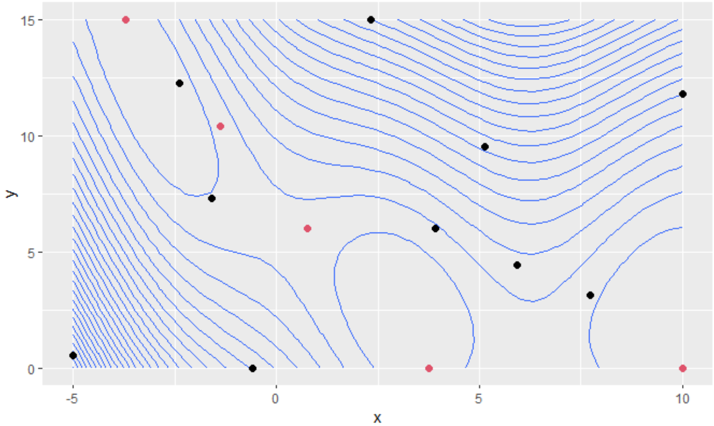


Figure: 5 points added using EGO

Example of ROI Determination Using Branin

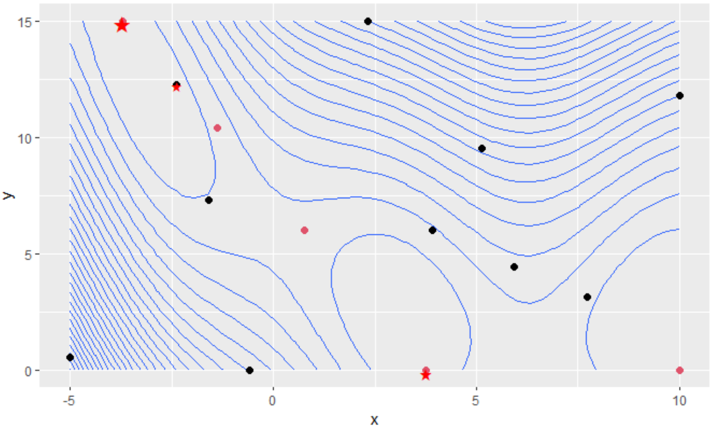


Figure: The best 3 points ie $\rho = 20\%$

Example of ROI Determination Using Branin

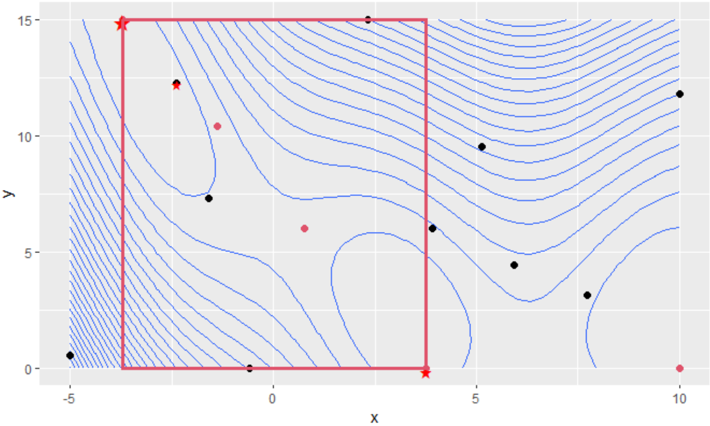


Figure: Region containing the best 20% of the points

Example of ROI Determination Using Branin

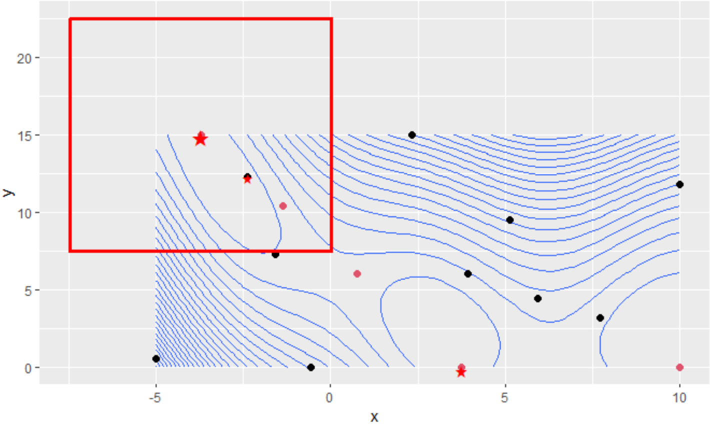


Figure: ROI centered at the current best point

Example of ROI Determination Using Branin

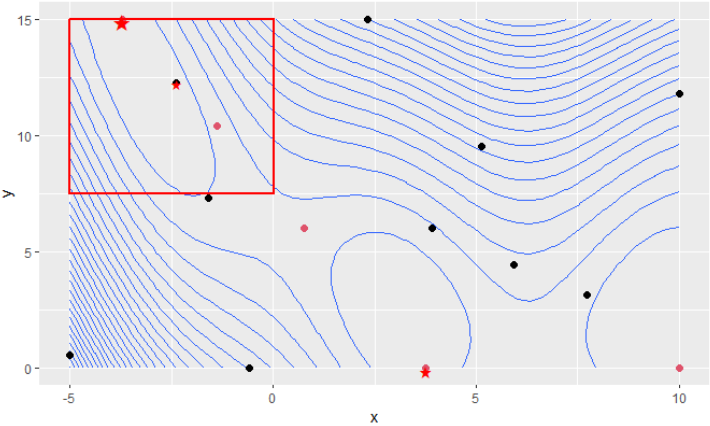
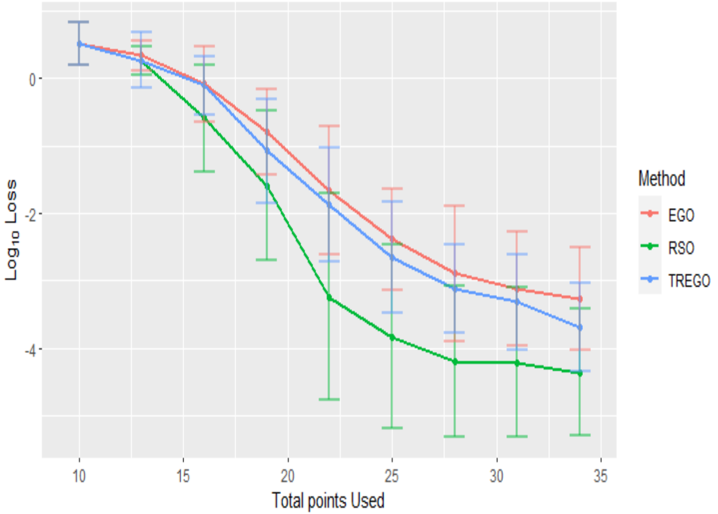


Figure: ROI constrained to be within the function domain

Why Restore the Region?

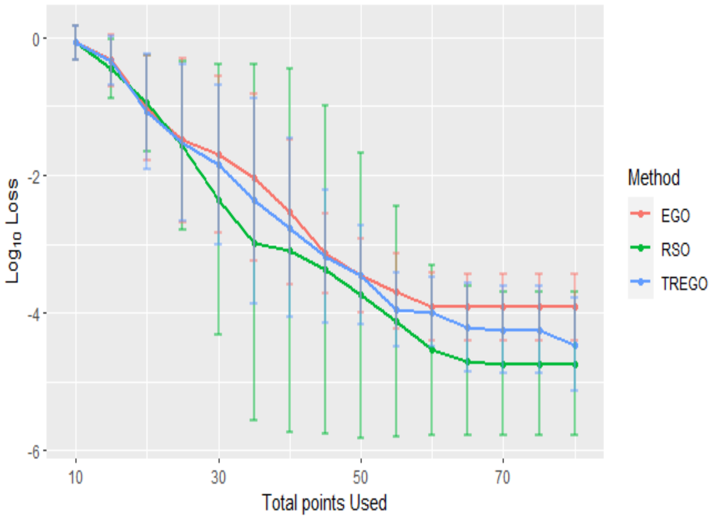
- Once converged, the current optimum is optimal only for the ROI
- We need to check as to whether its optimal for the entire search space
- Restore the ROI to the entire function domain and carry out EGO
- Check as to whether you converge to the current optimal item
  Continue with the optimization if you obtain a better point than the current point

Results – Branin (2D)



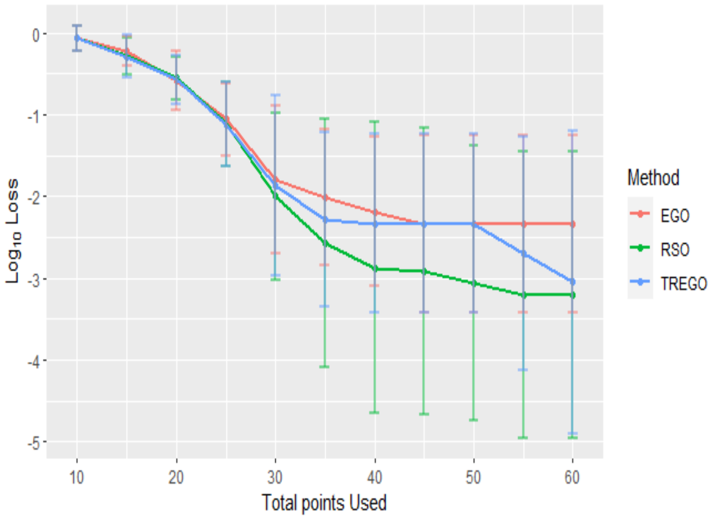
Branin

Results – Hartmann4 (4D)



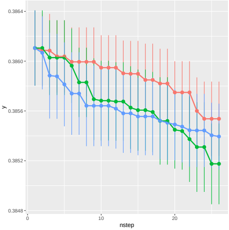
Hartmann4

Results – Hartmann6 (6D)

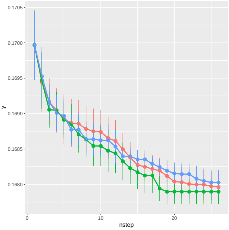


Hartmann6

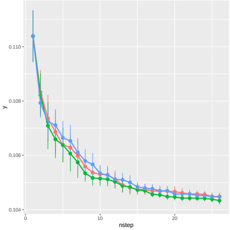
Methods Minimization Path



30×3

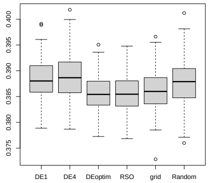


50×5

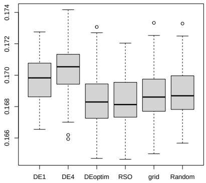


70×7

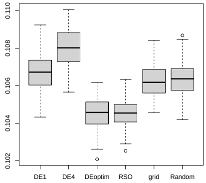
Methods Comparison Boxplots



30×3 as target



50×5 as target



70×7 as target

Contributions

- Study showing efficiency of UniPro
- Novel algorithm for generalized optimization
- Package to generate efficient UniPro designs

THANK YOU!