

Technical Appendix for: Black-box Mixed-Variable Optimisation using a Surrogate Model that Satisfies Integer Constraints

Laurens Blik

l.blik@tue.nl

Eindhoven University of Technology
Eindhoven, Netherlands

Sicco Verwer

S.E.Verwer@tudelft.nl

Delft University of Technology
Delft, Netherlands

Arthur Guijt

arthur.guijt@cwi.nl

Centrum Wiskunde & Informatica
Amsterdam, Netherlands

Mathijs de Weerd

M.M.deWeerd@tudelft.nl

Delft University of Technology
Delft, Netherlands

ACM Reference Format:

Laurens Blik, Arthur Guijt, Sicco Verwer, and Mathijs de Weerd. 2021. Technical Appendix for: Black-box Mixed-Variable Optimisation using a Surrogate Model that Satisfies Integer Constraints. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion)*, July 10–14, 2021, Lille, France. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3449726.3463136>

A DETAILS FOR GENERATING MIXED BASIS FUNCTIONS

In this section we show how to choose p_ω and p_b from Definition 2 in such a way that the mixed z-functions are never completely outside the domain $X_c \times X_d$. We recommend to choose p_ω to be a uniform distribution over $[-\frac{1}{\gamma_c + \gamma_d}, \frac{1}{\gamma_c + \gamma_d}]^{\gamma_c + \gamma_d}$. This way, the term $\mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d$ will not take on large values, which might cause numerical problems.

After sampling $\omega_k = \begin{bmatrix} \mathbf{v}_k \\ \mathbf{w}_k \end{bmatrix}$ from p_ω , we look for two corner-points $\mathbf{q}_1, \mathbf{q}_2$ of the space $X_c \times X_d$. For every dimension i , the i -th element of corner points $\mathbf{q}_1, \mathbf{q}_2$ is determined by

$$\mathbf{q}_{1i} = \begin{cases} l_i, & \omega_{ki} \geq 0, \\ u_i, & \omega_{ki} < 0, \end{cases} \quad (1)$$

$$\mathbf{q}_{2i} = \begin{cases} u_i, & \omega_{ki} \geq 0, \\ l_i, & \omega_{ki} < 0. \end{cases} \quad (2)$$

Here, l_i and u_i are the lower and upper bounds of the i -th variable respectively, so this gives

$$\omega_k^T \mathbf{q}_1 \leq \mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d \leq \omega_k^T \mathbf{q}_2 \quad \forall \mathbf{x}_c \in X_c, \mathbf{x}_d \in X_d. \quad (3)$$

Now we calculate the distance from the hyperplane generated by ω_k to these corner points, which can be done with the inner product:

$$\beta_1 = \omega_k^T \mathbf{q}_1, \quad \beta_2 = \omega_k^T \mathbf{q}_2. \quad (4)$$

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

GECCO '21 Companion, July 10–14, 2021, Lille, France

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8351-6/21/07.

<https://doi.org/10.1145/3449726.3463136>

By the way β_1 and β_2 are constructed and because $l_i < u_i$, we now have $\beta_1 < \beta_2$. We choose p_b equal to the uniform distribution over $[-\beta_2, -\beta_1]$.

Next we prove that this choice of p_b prevents the hyperplane $z_k(\mathbf{x}_c, \mathbf{x}_d) = 0$ from being completely outside the set $X_c \times X_d$.

THEOREM 1. Let $\omega_k = \begin{bmatrix} \mathbf{v}_k \\ \mathbf{w}_k \end{bmatrix}$ be sampled from any continuous probability distribution p_ω and let b_k be sampled from the uniform distribution over $[-\beta_2, -\beta_1]$, with β_1, β_2 as in (4). Let $z_k(\mathbf{x}_c, \mathbf{x}_d) = \mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d + b_k$. Then, there exists a $(\mathbf{x}_c, \mathbf{x}_d) \in X_c \times X_d$ such that $z_k(\mathbf{x}_c, \mathbf{x}_d) = 0$.

PROOF. Suppose that $(\mathbf{x}_c, \mathbf{x}_d) \notin X_c \times X_d$ for all $(\mathbf{x}_c, \mathbf{x}_d)$ for which $z_k(\mathbf{x}_c, \mathbf{x}_d) = 0$. Then from (3), at least one of the following inequalities holds:

$$\mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d > \omega_k^T \mathbf{q}_2, \quad (5)$$

$$\mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d < \omega_k^T \mathbf{q}_1. \quad (6)$$

Because $z_k(\mathbf{x}_c, \mathbf{x}_d) = 0$, we have $b_k = -\mathbf{v}_k^T \mathbf{x}_c - \mathbf{w}_k^T \mathbf{x}_d$. Because b_k is sampled from p_b , from (4) we also have $-\omega_k^T \mathbf{q}_2 \leq b_k \leq -\omega_k^T \mathbf{q}_1$. This gives $\omega_k^T \mathbf{q}_1 \leq \mathbf{v}_k^T \mathbf{x}_c + \mathbf{w}_k^T \mathbf{x}_d \leq \omega_k^T \mathbf{q}_2$, which is in conflict with (5)-(6). By contradiction, there has to exist a $(\mathbf{x}_c, \mathbf{x}_d) \in X_c \times X_d$ with $z_k(\mathbf{x}_c, \mathbf{x}_d) = 0$. \square

B DETAILS ON THE EXPLORATION STEP FOR INTEGER VARIABLES

This section gives more details on the last step of the MVRSM algorithm, the exploration step. For the integer variables \mathbf{x}_d^* , the exploration step consists of determining a random perturbation $\delta_d \in \mathbb{Z}^{\gamma_d}$ that is added to the solution. Our approach is similar to the one in [3, Sec. 3.4], except that we allow perturbations that are larger than 1. We determine δ_d according to Algorithm 1.

For the continuous variables, we use the procedure from [2], adding a random variable $\delta_c \in \mathbb{R}^{\gamma_c}$ to \mathbf{x}_c^* . For each continuous variable $\mathbf{x}_c[i]$, δ_c is zero-mean normally distributed with a standard deviation of $0.1|\mathbf{x}_c[i]|/\sqrt{\gamma_c + \gamma_d}$. The exploration step for both integer and continuous variables is done in such a way that the solution stays within the bounds X_c, X_d .

Algorithm 1 Determining δ_d

Input Domain X_d , current solution \mathbf{x}^*
Output $\delta_d \in \mathbb{Z}^{Y_d}$

```

for  $i = 1, \dots, Y_d$  do
   $r_1 \sim \text{Uniform}[0, 1]$ 
   $r_2 \sim \text{Uniform}[0, 1]$   $\triangleright$  Whether to increase or decrease  $x_i$ , the
   $i$ -th element of  $\mathbf{x}_d^*$ 
   $p = 1/(\gamma_c + \gamma_d)$ 
  while  $r_1 < p$  do
    if  $x_i = l_i$  then  $x_i \leftarrow x_i + 1$ 
    else if  $x_i = u_i$  then  $x_i \leftarrow x_i - 1$ 
    else
      if  $r_2 < 0.5$  then  $x_i \leftarrow x_i + 1$ 
      else  $x_i \leftarrow x_i - 1$ 
   $r_1 \leftarrow 2r_1$ 

```

C DETAILS ON THE INITIALISATION OF COEFFICIENTS

In this section we would like to provide a short motivation for the choice of initial weights of the surrogate model used by MVRSM (Sec. 4.2). By initializing the coefficients corresponding to the integer z -functions as 1, the surrogate model has a convex shape with a global minimum around the center of the discrete part of the search space, making it easier for the optimiser. For the mixed z -functions we cannot guarantee this particular shape due to the randomness involved, hence we initialise those coefficients as 0. All of this can also be interpreted as putting a convex prior on the surrogate model.

D ADDITIONAL EXPERIMENTS ON SYNTHETIC BENCHMARK FUNCTIONS

In this section we show the results on some additional synthetic benchmarks, ordered by problem dimension.

Func3C. This benchmark was taken from [5, Sec. 5.1]. It has 3 categorical and 2 continuous variables.

Figure 1 shows the results of 200 iterations averaged over 100 runs. We have managed to reproduce the results from [5, Fig. 6(b)] for both HO (also called TPE) and CoCaBO. Our result of SMAC is better here due to not using the default setting. As this benchmark has categorical variables and was one of CoCaBO's benchmarks, we expect CoCaBO to perform best, which it does, though it uses more computation time than the other methods.

Rosenbrock10. The Rosenbrock function¹ is a standard benchmark in continuous optimisation that can be scaled to any dimension. For any dimension, the function has its global minimum in the point $(1, 1, 1, \dots, 1)$, where it achieves the value 0. This benchmark has a dimension of 10, but 3 of the variables were adapted to integers in $X_d = \{-2, -1, 0, 1, 2\}^3$. The 7 remaining continuous variables were limited to $X_c = [-2, 2]^7$. The function was scaled with a factor $1/300$, and uniform noise in $[0, 10^{-6}]$ was added to every function evaluation. This problem is of the same scale as the problem of gradient boosting hyperparameter tuning [4, Sec. 4(a)].

¹Details available at <https://www.sfu.ca/~ssurjano/optimization.html>

Figure 2 shows the results of 100 iterations averaged over 100 runs. Surprisingly, BO has the best performance, though it is much slower than MVRSM. This method is typically used on continuous problems and widely assumed to be inadequate for discrete or mixed problems. Here, we have experimentally shown that this is a false assumption. MVRSM and CoCaBO get similar results as BO on this problem, with MVRSM being the most efficient.

MiVaBO synthetic function. We also compare with one of the randomly generated synthetic test functions from [4, Appendix C.1, Gaussian weights variant]. This problem has 16 variables of which 8 integer and 8 continuous. No bounds were reported so we set them to $X_d = \{0, 1, 2, 3\}^8$ for the integer variables and $X_c = [0, 3]^8$ for the continuous variables. We generated 8 of these random functions and ran all algorithms 16 times on each of them for 100 iterations.

Figure 3 shows the average over all 128 runs. Again, the standard BO algorithm performs best, which is a result that was not concluded in [4]. However, as will be seen in the remainder of this appendix and as is often mentioned in literature, when the number of variables becomes larger than 20, the performance of BO quickly deteriorates.

Cvxnonsep_psig20 synthetic function. We also investigated a publicly available mixed-variable benchmark function, namely *cvxnonsep_psig20* from the MINLP library². This problem contains 10 integer and 10 continuous variables.

Figure 4 shows the distance to the known global optimum, averaged over 7 runs with 1000 iterations each. Not only does MVRSM get closest to the global optimum, it does so in a much faster time than the other surrogate-based methods. BO is the second-best in terms of accuracy, but has a significantly larger computation time.

Cvxnonsep_psig30 synthetic function. This problem is similar to the previous example, but contains 15 integer and 15 continuous variables³. Again, MVRSM outperforms the other methods, while BO's performance is further decreased.

Rosenbrock238 synthetic function. We scaled the Rosenbrock function up to a dimension of 239, with the first 119 variables adapted to integers in $X_d = \{-2, -1, 0, 1, 2\}^{119}$, and 119 continuous variables limited to $X_c = [-2, 2]^{119}$. The function was scaled with a factor $1/50000$. This problem is of the same scale as the problem of feed-forward classification model hyperparameter tuning [1], except that the ratio between continuous and integer variables is chosen to be 1 : 1. Uniform noise in $[0, 10^{-6}]$ was added to each function evaluation.

Figure 6 shows the average over 7 runs with 2000 iterations each. The BO and CoCaBO methods were not tried due to their prohibitively large computation time. MVRSM performs especially well on this application, possibly due to the structure of this synthetic benchmark: it contains many interactions between subsequent variables, just like the discrete basis function of MVRSM's surrogate model.

²https://www.minlplib.org/cvxnonsep_psig20.html

³https://www.minlplib.org/cvxnonsep_psig30.html

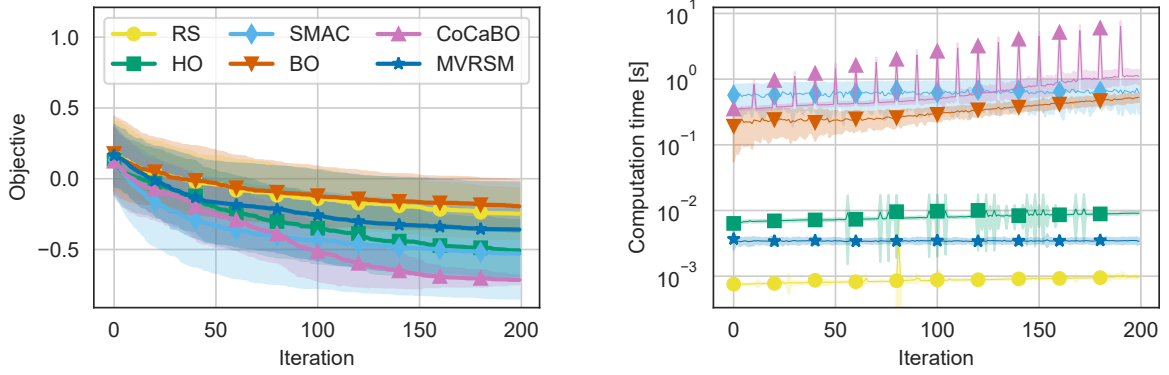


Figure 1: Results on the func3C [5, Sec. 5.1] benchmark (3 categorical, 2 continuous), averaged over 100 runs. The compared methods are random search (RS), HyperOpt (HO), SMAC, Bayesian optimisation (BO), CoCaBO and MVRSM.

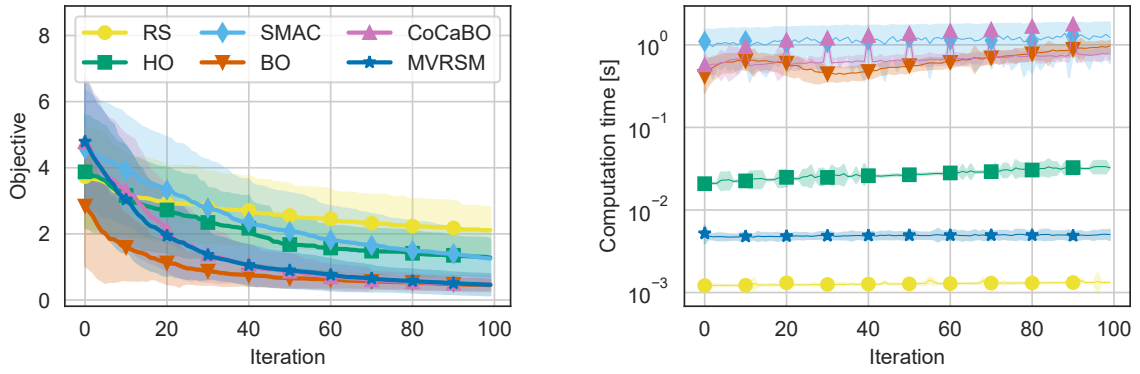


Figure 2: Results on the Rosenbrock10 benchmark (3 integer, 7 continuous), averaged over 100 runs. This problem is of a similar scale as gradient boosting hyperparameter tuning [4, Sec. 4(a)].

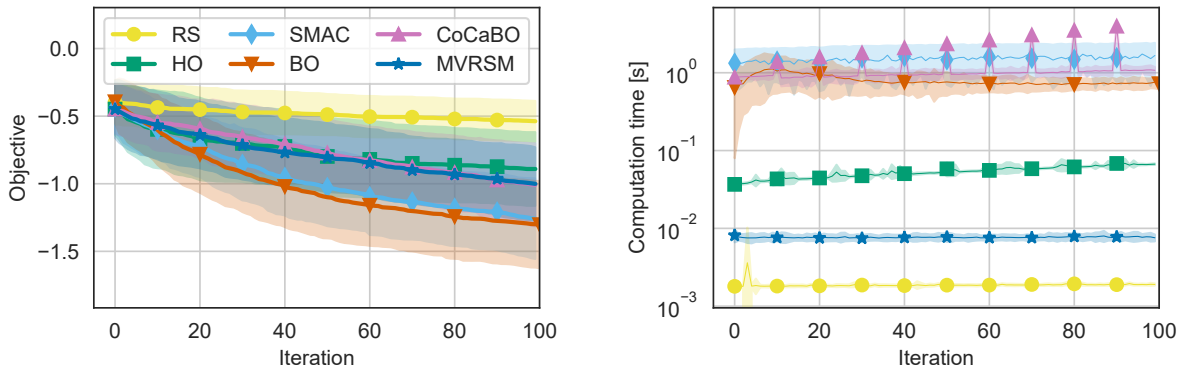


Figure 3: Results on 8 randomly generated MiVaBO synthetic benchmarks [4, Appendix C.1, Gaussian weights variant] (8 integer, 8 continuous), averaged over 16 runs and over the 8 different benchmarks.

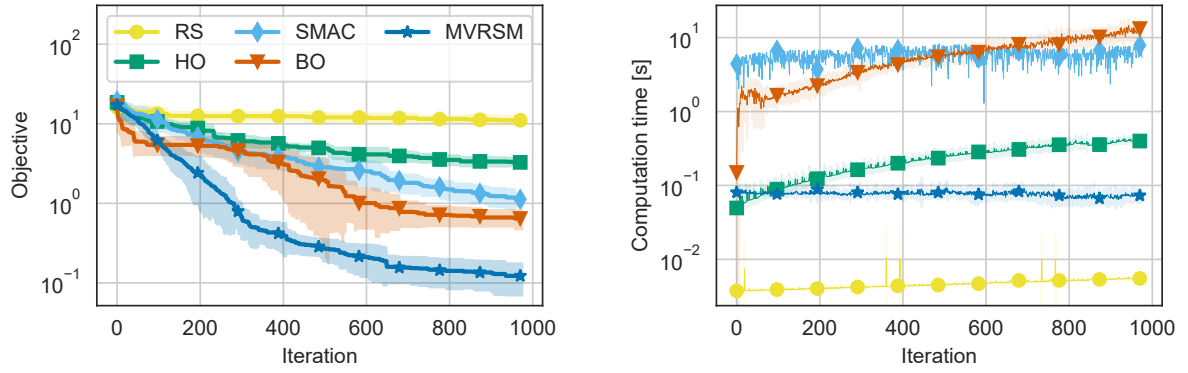


Figure 4: Results on the `cvxnonsep_psig20` benchmark (10 integer, 10 continuous), averaged over 7 runs. The distance to the known global optimum is shown on a logarithmic scale.

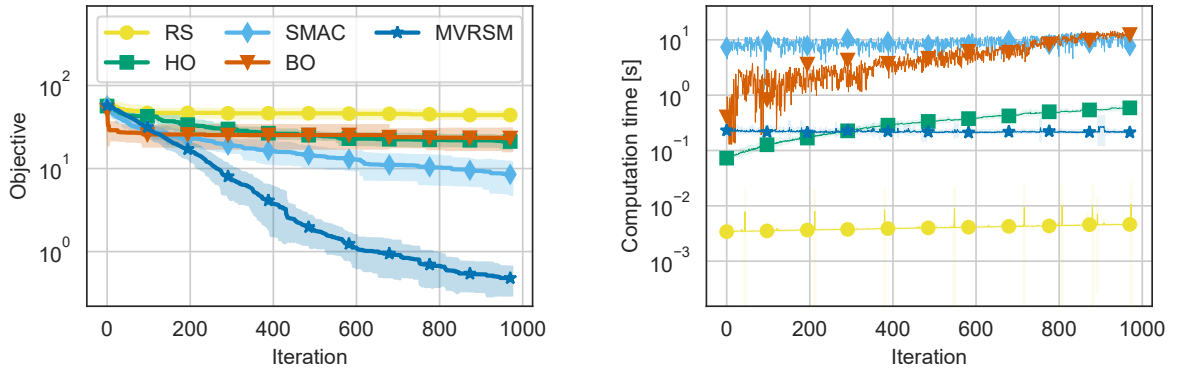


Figure 5: Results on the `cvxnonsep_psig30` benchmark (15 integer, 15 continuous), averaged over 7 runs. The distance to the known global optimum is shown on a logarithmic scale.

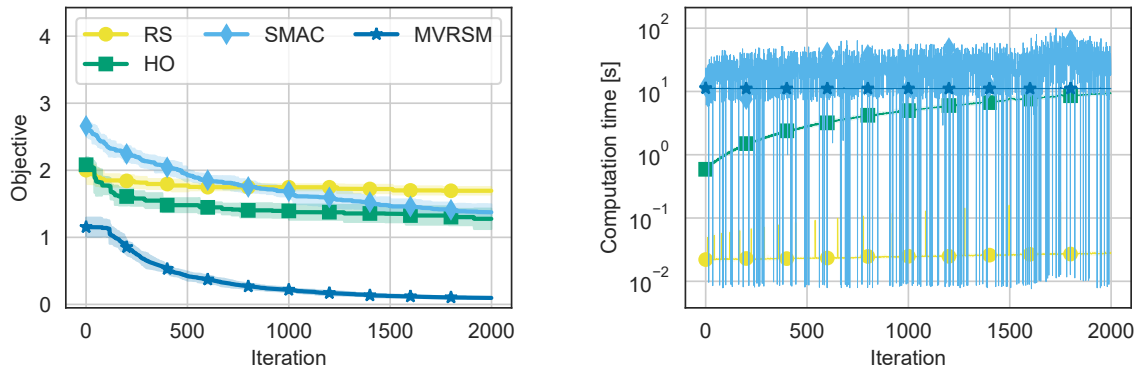


Figure 6: Results on the `Rosenbrock238` benchmark (119 integer, 119 continuous), averaged over 7 runs. BO and CoCaBO were not evaluated for this benchmark due to the large computation time. This problem is of a similar scale as feed-forward classification model hyperparameter tuning [1].

REFERENCES

- [1] J. Bergstra, D. Yamins, and D. Cox. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In *ICML - Volume 28*, pages I–115, 2013.
- [2] L. Bliet, H. R. Verstraete, M. Verhaegen, and S. Wahls. Online optimization with costly and noisy measurements using random Fourier expansions. *IEEE Transactions on Neural Networks and Learning Systems*, 29(1):167–182, Jan 2018.
- [3] L. Bliet, S. Verwer, and M. de Weerd. Black-box combinatorial optimization using models with integer-valued minima. *Annals of Mathematics and Artificial Intelligence*, pages 1–15, 2020.
- [4] E. Daxberger, A. Makarova, M. Turchetta, and A. Krause. Mixed-variable Bayesian optimization. In *IJCAI*, pages 2633–2639, 2020.
- [5] B. Ru, A. Alvi, V. Nguyen, M. A. Osborne, and S. Roberts. Bayesian optimisation over multiple continuous and categorical inputs. In *ICML*, pages 8276–8285, 2020.