Appendix

A. Primer on GP and BO

Gaussian processes We consider a GP surrogate model for a black-box function f which takes an input $\mathbf{z} = [\mathbf{h}, \mathbf{x}]$ and returns an output $y = f(\mathbf{z}) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Here, the input includes a continuous variable $\mathbf{x}$ and a categorical variable $\mathbf{h}$. A GP defines a probability distribution over functions f under the assumption that any finite subset $\{(\mathbf{z}_i, f(\mathbf{z}_i)\}$ follows a normal distribution (Rasmussen, 2006). Formally, a GP is denoted as $f(\mathbf{z}) \sim \mathrm{GP}(m(\mathbf{z}), k(\mathbf{z}, \mathbf{z}'))$, where $m(\mathbf{z})$ and $k(\mathbf{z}, \mathbf{z}')$ are called the mean and covariance functions respectively, i.e. $m(\mathbf{z}) = \mathbb{E}[f(\mathbf{z})]$ and $k(\mathbf{z}, \mathbf{z}') = \mathbb{E}\left[(f(\mathbf{z}) - m(\mathbf{z}))(f(\mathbf{z}') - m(\mathbf{z}'))^T\right]$. The covariance function (kernel) $k(\mathbf{z}, \mathbf{z}')$ can be thought of as a similarity measure relating $f(\mathbf{z})$ and $f(\mathbf{z}')$. There have been various proposed kernels which encode different prior beliefs about the function $f(\mathbf{z})$, typically in the continuous space. Popular choices include the Square Exponential kernel, the Matérn kernel (Rasmussen, 2006).

Assume the zero mean prior $m(\mathbf{z}) = 0$, to predict $f_* = f(\mathbf{z}_*)$ at a new data point $\mathbf{z}_*$, we have,

$$\left[\begin{array}{c} \boldsymbol{f} \\ f_* \end{array}\right] \sim \mathcal{N}\left(0, \left[\begin{array}{cc} \boldsymbol{K} & \mathbf{k}_*^T \\ \mathbf{k}_* & k_{**} \end{array}\right]\right), \tag{5}$$

where $k_{**} = k(\mathbf{z}_*, \mathbf{z}_*)$, $\mathbf{k}_* = [k(\mathbf{z}_*, \mathbf{z}_i)]_{\forall i \leq N}$ and $\boldsymbol{K} = [k(\mathbf{z}_i, \mathbf{z}_j)]_{\forall i,j \leq N}$. Combining Eq. (5) with the fact that $p(f_* \mid \boldsymbol{f})$ follows a univariate Gaussian distribution $\mathcal{N}(\mu(\mathbf{z}_*), \sigma^2(\mathbf{z}_*))$, the GP posterior mean and variance can be computed as,

$$\begin{aligned} \mu(\mathbf{z}_*) &= \mathbf{k}_* \left[\mathbf{K} + \sigma^2 \mathbf{I}\right]^{-1} \mathbf{y}, \\ \sigma^2(\mathbf{z}_*) &= k_{**} - \mathbf{k}_* \left[\mathbf{K} + \sigma^2 \mathbf{I}\right]^{-1} \mathbf{k}_*^T. \end{aligned}$$

As GPs give full uncertainty information with any prediction, they provide a flexible nonparametric prior for Bayesian optimisation. We refer the interested readers to Rasmussen (2006) for further details on GPs.

Bayesian optimisation Bayesian optimisation is a powerful sequential approach to find the global optimum of an expensive black-box function $f(\mathbf{z})$ without making use of derivatives. First, a surrogate model is learned from all the current observed data $\mathcal{D}_t = \{\mathbf{z}_i, y_i\}_{i=1}^t$ to approximate the behavior of $f(\mathbf{z})$. Second, an acquisition function is derived from the surrogate model to select new data points that mostly inform about the global optimum. The process is conducted iteratively until the evaluation budget is depleted, and the global optimum is estimated based on all the sampled data. In-depth discussions about Bayesian optimisation beyond this brief overview can be found in recent surveys (Brochu et al., 2010; Shahriari et al., 2016; Frazier, 2018).

B. Additional Experimental Results

B.1. Running Time Comparison

In this section, we provide comparison of CASMOPOLITAN against some baselines in terms of wall-clock running time on a number of problems considered. However, since we conduct our experiments on a shared server, inevitably there are fluctuations in wall clock time depending on the server load, leading to (perhaps rather significant) amount of uncertainty over the computing time reported here and thus, the figures here are for ballpark reference only. CASMOPOLITAN scales $\mathcal{O}(N^3)$, where N here refers to the number of training data (note that in CASMOPOLITAN, this is not necessarily the total number of observations, but only the number of training samples of the GP surrogate of the current restart), which is the time complexity of any GP-BO method where the computational bottleneck is the inversion of the covariance matrix (Shahriari et al., 2016). Practically, due the implementation in Gpytorch which utilises Blackbox Matrix-matrix (BBMM) multiplication which reduces the cost of exact GP inference to $\mathcal{O}(N^2)$ (Gardner et al., 2018). Overall, the computing cost of CASMOPOLITAN is generally comparable to TURBO. On the other hand, previous methods generally scale worse. For example, in addition to the inherent $\mathcal{O}(N^3)$ complexity (or $\mathcal{O}(N^2)$ if BBMM is similarly exploited), COMBO additionally incurs the cost in the graph Fourier transform of $\mathcal{O}(\sum_{i=1}^{d_h} n_h^3)$ using the notations of our paper (Proposition 2.3.1 in Oh et al. (2019)). Furthermore, it also uses slice sampling for the approximate marginalisation of the posterior predictive distribution, which is arguably more expensive than simple optimisation of the log marginal-likelihood. A single iteration of BOCS incurs complexity of $\mathcal{O}(N^2 {d_h}^2)$ (Baptista and Poloczek, 2018), suggesting that the runtime of BOCS quadratically also with respect to the *dimensionality* of the problem. Furthermore, it is worth noting that the quadratic dependence of d_h stems from the second order *approximation* of their sparse Bayesian linear regression model. This term will become much more expensive if a higher order approximation is used, e.g. it becomes d_h^m if an m-th order approximation is used.

For the categorical problems, COMBO achieves comparable performance in terms of the function value at termination in 2 out of 3 problems in Fig. 3 and thus the analysis of computing cost against it is of our prime interest (other methods are either not competitive in terms of performance (e.g. TPE), or are *much* more expensive and/or more constrained in applicability (e.g. BOCS). The comparison against COMBO (and BOCS where applicable) is shown in Table 1, where it is evident that our method offers around 2-3 times speedup compared to COMBO, whereas BOCS is orders-of-magnitude more expensive.

For the mixed problems, we analyse the black-box attack

Table 1. Wall-clock time comparison of a single trial (mean $\pm$ standard deviation across 20 trials for COMBO and CASMOPOLITAN) on categorical problems on a shared Intel Xeon server.

Problem	Ours	COMBO	BOCS
Pest	**1130**$_{\pm 80}$s	3330$_{\pm 50}$s	$\sim$ 1d
Contamination	**2350**$_{\pm 180}$s	6630$_{\pm 400}$s	n.s.
MAXSAT-60	**12100**$_{\pm 3000}$s	34300$_{\pm 2000}$s	o.o.t

n.s: setup not supported
o.o.t: run out-of-time ($>$ 100 hours) and did not finish.

problem. On average, the time taken to attack an image (successful or not) is around 45 mins with MVRSM, which uses ReLU surrogate instead of GP, whereas TURBO and our CASMOPOLITAN run roughly $1.6\times$ and $2.0\times$ more expensive – TURBO run faster likely due to its more frequent restarts. However, in realistic setups suitable for BO where either the objective function evaluation time completely eclipses the algorithm running time (e.g. tuning of large-scale machine learning system) or where *sample efficiency*, as contrasted *wall-clock efficiency*, is otherwise more valued (e.g. the black-box attack setup discussed here), the larger cost of CASMOPOLITAN is likely justifies by its better performance. Overall, we believe that CASMOPOLITAN offers a sound balance between good performance and reasonable computing cost.

B.2. Additional Problems

Func3C The results are shown in Fig. 7. The results are broadly comparable to that of Func2C in Fig. 4(a), although in this case COCABO and vanilla-BO perform more strongly near the end.

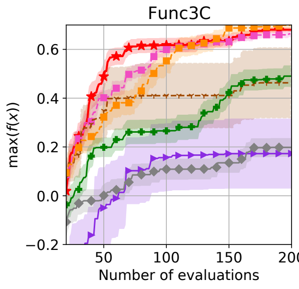


Figure 7. Results on Func3C

Noisy Contamination We conduct a further experiment on the Contamination problem but with an additional noise variance of 1×10^{-2}, and the results are shown in Fig. 8. In this case, we again see CASMOPOLITAN and COMBO outperforming the rest and CASMOPOLITAN again enjoys a faster convergence than the other methods. In this particular case, COMBO outperforms CASMOPOLITAN, albeit marginally, at the end.

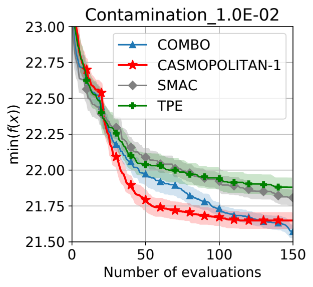


Figure 8. Results on Contamination problem with noise variance of 0.01.

Ordinal problems Sometimes we encounter ordinal problems, which are discrete variables that are similar to the categorical problems, but unlike categorical, there exists some kind of ordering between the different values that a variable can take. For example, in deep learning we often choose batch size as powers of 2, where possible batch sizes are $\{64, 128, 256, 512\}$. While current methods and popular packages (e.g. COCABO and the Bayesmark[5] package) often treat these as ordinary categorical variables by ignoring such ordinal structure, this practice might not be optimal. In this section, we describe an exemplary adaptation of CASMOPOLITAN in the ordinal setting that recognises and leverage such relations, and conduct a preliminary experiment to validate it as a demonstration of the versatility of our approach.

In our tailored kernel for the categorical variables (Eq. (1)), we use Kronecker delta function which only has two possible outcomes: 0 if the two values are different or 1 if the two values are the same. This is appropriate in the categorical setting because there exists no ordering amongst different choices a variable may take (e.g. consider choosing from {SGD, Adam, RMSPROP}: to SGD, Adam can be considered "as different as" RMSPROP. However, in ordinal-structured problems such as the batch size example above, the choice of 128 is certainly "more similar" to 64 than 256. To recognise this, we modify Eq. (1), reproduced below for convenience:

$$k_h(\mathbf{h}, \mathbf{h}') = \exp\Big(\frac{1}{d_n}\sum_{i=1}^{d_h} \ell_i \delta(h_i, h'_i)\Big).$$

For ordinal variables, we use the *ordinal kernel* k_o by replacing the Kronecker delta function with an appropriate distance metric. One possible formulation is:

$$k_o(\mathbf{h}, \mathbf{h}') = \exp\Big(\frac{1}{d_n}\sum_{i=1}^{d_h} \ell_i \big(1 - \frac{|h_i - h'_i|}{|h_i - h'_i|_{\max}}\big)\Big), \quad (6)$$

where $|h_i - h'_i|$ is the distance that is dependent on the problem-specific metric and $|h_i - h'_i|_{\max}$ is the maximum possible distance (in the context of the batch size problem,

[5] https://github.com/uber/bayesmark

this is $512-64$). Note that when no ordinal structure exists, $|h_i - h_i'|$ is either 0 or $|h_i - h_i'|_{\max}$ and Eq. (6) reduces to the categorical kernel in Eq. (1).

We further include a preliminary empirical validation on the 2D discretised Branin problem introduced in Oh et al. (2019), where each dimension of the Branin function in $[-1,1]^2$ is discretised into 51 equally spaced points – as such, the problem has 2 ordinal dimensions with 51 choices for each. Note that this is a rather extreme example due to the large number of choices relative to the number of variables, and the fact that the function landscape resembles much more to a continuous problem instead of a typical ordinal one, but we include it for the sake of illustration. We also do not use trust region for this example due to the low dimensionality and the fact that the point of this experiment is to compare categorical and ordinal kernels. We show the results in Fig. 9, where we also include the results for COMBO which is the strongest baseline shown to outperform other methods such as SMAC and TPE in Oh et al. (2019). It is worth emphasising that COMBO also explicitly accounts for the ordinal relations, so the comparison of it against CASMOPOLITAN with ordinal kernel is fair.

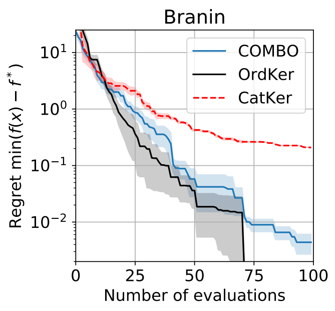


Figure 9. Results on Discretised Branin. Lines and shades denote mean and standard error across 20 trials. Note that since the optimum is known in this case ($f^* = 0.404$), in the y-axis we show the regret in log-scale.

It is clear that CASMOPOLITAN with ordinal kernel (`OrdKer`) outperforms both the ordinal-agnostic CASMOPOLITAN (`CatKer`) and the ordinal-aware COMBO in both convergence speed and final performance (`OrdKer` converges to f^* every single trial). To show why it is the case, we plot the GP posterior variance of CASMOPOLITAN with each kernel in Fig. 10: categorical kernel measures similarity via the Hamming distances only, and thus each observation $\mathbf{h}_i$ only reduces posterior variance on the points sharing at least one common dimension as $\mathbf{h}_i$. On the other hand, ordinal kernel further accounts for the similarity amongst different values an input may take, and thus each evaluation also reduces the variance in the vicinity of $\mathbf{h}_i$.

While we only consider a toy problem here, the fact that k_o is a simple modification over k_h means it is trivial to scale the approach to high dimensions with the local TR approaches described in the main text and/or to the mixed inputs, such as ordinal-continuous or even ordinal-categorical-continuous search space. We defer a thorough investigation to this even richer class of problems to a future work, which we believe would be an exciting extension to the present work.

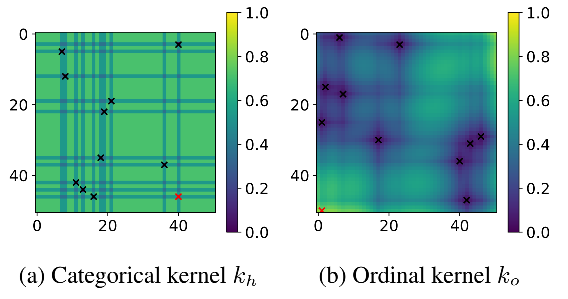


Figure 10. Posterior variance of GP with categorical and ordinal kernels after 10 random initial points on the discretised Branin problem. Black markers are the GP observations; red markers are the proposed locations for the next evaluations.

B.3. Parallel CASMOPOLITAN by number of objective function queries

Supplementary to Fig. 5 which shows the comparison of performances of CASMOPOLITAN of varying batch sizes by *number of batches*, here we compare the performance by *number of objective function queries* in Fig. 11. It is evident that increasing the number of batches, at least in the experiments we consider, does not lead to significant performance deterioration even though we may achieve near-linear reduction in wall-clock time if we have sufficient parallel computing resources.

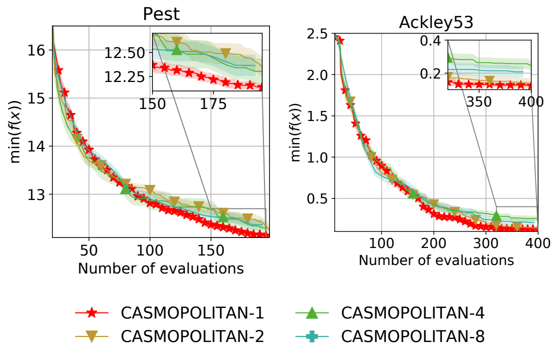


Figure 11. Parallel CASMOPOLITAN on representative categorical and mixed problems by *number of function queries*.

B.4. Additional Results on the Black-Box Attack Task

Supplementary to the main text, in Fig. 12 we show more examples of the adversarial examples generated by our method, where the diagonal images are the original, unperturbed images in the CIFAR-10 validation dataset that the CNN initially classifies correctly while the off-diagonal entries are the adversarial examples. From the 50 images attacked by us, we select an image and we compare the objective function value against number of queries in the 9 attack instances in Fig. 13. It is clear that our method achieves higher success rate within the highly limited budget (successful in $6/9$ instances, as opposed to 3 and 1 in TURBO and MVRSM), and even in cases where attack is unsuccessful within the budget, our method still increases the loss more and pushes it closer to the success boundary.

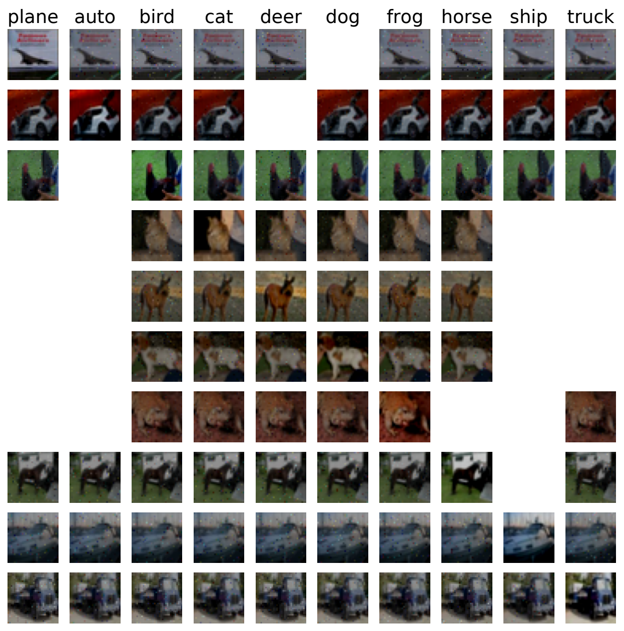


Figure 12. Some adversarial examples generated by our method.

B.5. Sensitivity Studies on the Additional Hyperparameters

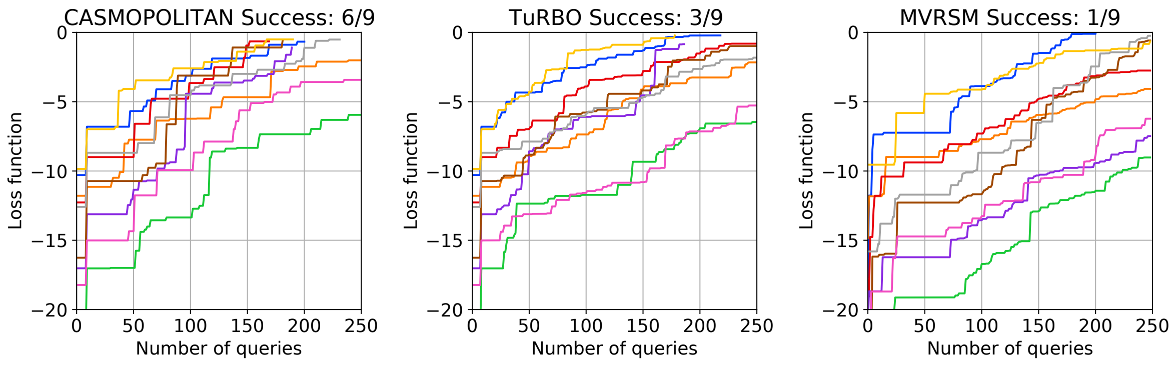
Similar to TURBO, our method introduces some additional hyperparameters related to the trust regions. In this section we examine the sensitivity of the performance of CASMOPOLITAN towards these hyperparameters on Pest and Ackley53 problems. Specifically, we test the sensitivity towards:

- Initial trust region length: unlike the hyperrectangular TRs for the continuous space where $L^x_{\min}$ and $L^x_{\max}$ are additional hyperparemeters, the Hamming distance-based TRs are constrained to be positive integers in $(0, d_h]$, relieving us from the need to tune $L^h_{\min}$ and $L^h_{\max}$. Nonetheless, the trust region length at the beginning of each restarts is still a free hyperparameter.
- Failure tolerance (`fail_tol`): the number of successive failures to shrink the trust region. An aggressive `fail_tol` setting (i.e. one that is very small) could lead to rapid trust region shrinking and possibly more frequent restarts. Note that it is generally rare to have a large number of consecutive successes in increasing the function value, and therefore we fix the success tolerance (`succ_tol`) to be 2.
- Shrinking rate of TRs (α_s): the multiplier when TR shrinking is triggered $L \leftarrow \alpha_s L$; a more aggressive value of α_s leads to more rapid shrinking and restart upon stagnation in improving $f(\mathbf{z})$. Note that α_s is always coupled with the expansion rate $\alpha_e = \frac{1}{\alpha_s}$ when TR expansion is triggered, and hence we do not further test the sensitivity to α_e. Also, we use the same expansion and shrinking rates in both continuous and categorical TRs.

We show the results in Fig. 14 where the default hyperparameter values are `fail_tol` $= 40$, initial trust region length 20 (for Pest Control with $d_h = 25$) or 40 (for Ackley-53 with $d_h = 50$) and α_s = 0.667 (and thus $\alpha_e = 1.5$). In each of the experiments presented in Fig. 14, we only tune the hyperparameter in interest, and leave all others at their default values. For the mixed problems, we do not tune the hyperparameters specific to the continuous TRs (e.g. the initial, min and max continuous TR lengths) and instead leave them at their default values in the official TURBO implementation. Furthermore, due to the large number of hyperparameter configurations, we only run each configuration once. The results show that the performance of CASMOPOLITAN is generally insensitive to the hyperparameter choice, as the vast majority of the results fall within 2 standard deviations of results in the main text *running the exactly the same configurations*, suggesting that, as a whole, the impact on performance due to different hyperparameter choices might not be more significant compared to the inherent randomness in initialisation in different trials. It is further worth noting that in all cases CASMOPOLITAN still outperforms the corresponding next best baseline – this suggests that the performance difference is mainly driven by the choice of *different algorithms*, as opposed to *different hyperparameters of the same algorithm.*

B.6. Comparison against ALEBO and REMBO

In this section we run a small comparison of our method against REMBO (Wang et al., 2013) and ALEBO (Letham et al., 2020), the representative methods of the class of high-dimensional BO methods. We compare against them in the Ackley-53 problem with setups identical to the description in Sec. 4 in the main text, and we show the results in Fig. 15 where for both algorithms, we run under their respective default hyperparameter settings. We observe that while both outperform COCABO, they are outperformed by CASMOPOLITAN and TURBO by a large margin.



(a) CASMOPOLITAN (b) TURBO (c) MVRSM

Figure 13. Attack loss function (described in Eq. (11) against the number of queries of an attacked image in CASMOPOLITAN, TURBO and MVRSM, three most competitive methods of the black-box adversarial attack problem. Lines on each image denote the targeted attack to the 9 target classes. Some lines terminate earlier than the full budget because the attack is successful before using up all the query budget.

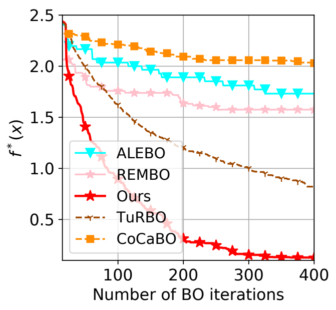


Figure 15. Comparison against REMBO and ALEBO on the Ackley-53 problem. The lines denote the mean performance across 10 random trials.

B.7. Empirical Comparison of UCB-based and Random Restarts

As discussed, the primary motivation of using UCB-based restarts of the trust regions is to theoretically driven, but in this section we investigate whether there exists any practical, finite-time benefits of using the UCB-based restarts.

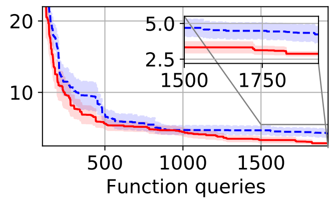


Figure 16. UCB vs random restarts in 20d Levy over 10 trials. t-test p-value between the two method is 0.048 at the final iteration. Shades denote ± 1 standard error.

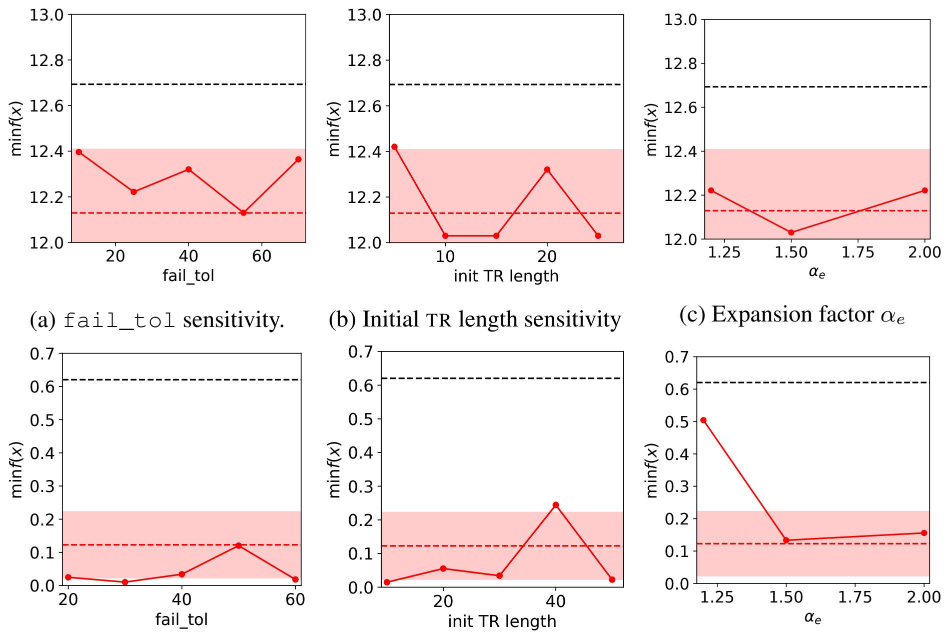
Practically, optimising the UCB on the auxiliary GP exactly is difficult. Instead, at each restart of the trust region, we simply sample a large number of points, compute their UCB scores based on the auxiliary GP, and select the top ones as the initialising points for the next restart. We emphasise that the auxiliary GP is not meant to fit well to the objective function, as otherwise we do not need trust regions to constrain the surrogate, but is to instead generate better initialising points than random selection. Based on this described procedure, we conduct an experiment comparing UCB vs random restarts on 20-dimensional Levy function, and we show the results in Fig. 16. It can be seen that using UCB-based restarts leads to small but statistically significant improvements over the randomly initialising baseline. Furthermore, in terms of running time, since the auxiliary GPs scale with the number of *restarts* instead of number of observations, we find the UCB variant to be only 1.9% slower in terms of running time. With these results, we expect that the proposed UCB criterion to lead to practical benefits even in modestly higher-dimensional problems given an extended query budget (such that we would typically observe a number of TR restarts for the effect of initialisation at the start of each restart to be significant).

C. Implementation Details

C.1. Description of the categorical problems

A table containing the details and other characteristic details of all the test problems are shown in Table 2.

Contamination Control Contamination Control is a binary optimisation problem in food supply chain (Hu et al., 2010): at each stage, we have the choice of whether to introduce contamination control, but early use of contamination control could inevitably lead to increase in cost and as such our objective is to minimise food contamination with the smallest monetary cost (hence a minimisation problem). It is worth noting that in this problem and the Pest Control



(a) `fail_tol` sensitivity. (b) Initial TR length sensitivity (c) Expansion factor α_e

(d) `fail_tol` sensitivity. (e) Initial TR length sensitivity (f) Expansion factor α_e

Figure 14. Sensitivity of CASMOPOLITAN performance towards various hyperparameters in Pest control (top row) and Ackley-53 (bottom row). The red lines/shades denote the mean $\pm$ 2 standard deviation of the baseline results in the main text, where as the dotted black line is the performance of the *next best-performing baseline* (COMBO and MVRSM respectively).

problem described below, the actions taken by the previous stage have implications on the following stages, thus leading to highly complicated interactions amongst the different variables. In this problem, we use the implementation used in Oh et al. (2019). However, it is worth noting that while Oh et al. (2019) consider a 21-stage (with a total of $2^{21} \approx 2.1 \times 10^6$ configurations) problem, we increase the total number of stages to 25 (with a total of $2^{25} \approx 3.4 \times 10^7$ configurations). In this experiment we limit the maximum number of evaluations of 150, as the running time of BOCS quickly increases beyond our computing budget if we set the it to a significantly higher value.

Pest Control We use the problem proposed in Oh et al. (2019) which expands the contamination control problem into a multi-categorical optimisation problem: at each stage, we now need not only to determine whether to take an action (to use pesticide or not), but also the type of the pesticide (4 choices in total). This thus gives rise to 5 potential choices for each stage. Similar to Contamination control, we again increase the total number of stages to 25 (as opposed to 21 in Oh et al. (2019)) to give an expanded and more complicated search space. In Ablation Studies of Sec. 4, we also include a variant named DifficultPest, where the total number of stages is further increased to 80.

Weighted Maximum Satisfiability Maximum satisfiability problem is a classical combinatorial optimisation problem that aims to determine the maximum number of clauses of a given Boolean formula in conjunctive normal form (CNF) that can be made true by an assignment of truth values to the variables. Similar to Oh et al. (2019), we take the same 60-variable benchmark from Maximum Satisfiability Competition 2018[6] (`frb-frb10-6-4.wcnf` problem from `https://maxsat-evaluations.github.io/2018/benchmarks.html`)

C.2. Description of the mixed problems

Func2C and Func3C These synthetic problems were first proposed in Ru et al. (2020a). In Func2C ($d_x = 2, d_h = 2$), the value of $\mathbf{h}$ determines the objective function value that is a linear combination of three benchmark functions, namely Beale, Six-Hump Camel and Rosenbrook (abbreviated as bea, cam and ros in Table 2); the function form of these 3 functions are:

$$\mathrm{bea}(\mathbf{x}) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2.$$

$$\mathrm{cam}(\mathbf{x}) = (5 - 2.1 x_1^2 + \frac{x_1^4}{3}) x_1^2 + x_1 x_2 + (-4 + 4 x_2^2) x_2^2.$$

$$\mathrm{ros}(\mathbf{x}) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2. \tag{7}$$

[6]`http://sat2018.azurewebsites.net/competitions/`

Func3C ($d_x = 2, d_h = 3$) is similar but has one extra categorical dimension to enable more complicated interactions.

XG-MNIST ($d_x = 5, d_h = 3$) This is a real hyperparameter tuning task of a machine learning model (XGBoost). The tunable continuous hyperparameters are maximum depth, minimum split loss, subsample, learning rate of the optimiser and the amount of regularisation. The categorical variables are the booster type, grow policies and training objective. We use the `xgboost` python package and adopt a train-test split of $7:3$ on the MNIST data. Note that this setup is identical to that used in Ru et al. (2020a).

Ackley-53 is a stylised version of the original 53-dimensional Ackley function, whose original form is given by:

$$f(\mathbf{z}) = -a \exp\Big(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} z_i^2)}\Big) - \exp\Big(\frac{1}{d}\sum_{i=1}^{d}\cos(cz_i)\Big) + a + \exp(1), \quad (8)$$

where in this case $a = 20, b = 0.2, c = 2\pi$ and $d = 53$ and we define $\mathbf{z} \in [-1,1]^{53}$. From this continuous form, the first 50 dimensions are modified to be *binary* variables that take the value of either 0 and 1, and the final 3 variables are continuous and limited in the range of $[-1,1]^3$. This adaptation is first proposed in Bliek et al. (2020). This function has a known global minimiser of $\mathbf{h}^* = [0,...,0]$ and $\mathbf{x}^* = [0,0,0]$ with $f^*(\mathbf{z}) = 0$.

Rosenbrock-200 is a stylised and scaled version of the classical Rosenbrock function. The Rosenbrock function is given by:

$$f(\mathbf{z}) = \frac{1}{50000}\Big(\sum_{i=1}^{d-1}\big(100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2\big)\Big), \quad (9)$$

where in this case $d = 200$. The first 100 dimensions are then converted to binary variables, while the final 100 dimensions are continuous in the range of $[-2,2]^{100}$.

Black-box adversarial attack We adapt black-box setup from Ru et al. (2020b), one of the first works that introduce BO in the image adversarial attack setting. Specifically, denoting $\mathcal{M}$ as the target model (or the victim model, in this case a CNN image classifier) from which we may query an image input I, the BO agent can only observe the prediction scores on all C classes (for CIFAR-10, $C = 10$): $\mathcal{M}(I) : \mathbb{R}_+^d \to [0,1]^C$ (thus a "black-box", since gradients, architecture and other information of the classifier itself are never revealed to the attack agent). Therefore, denoting I as the original, unperturbed image that $\mathcal{M}$ correctly gives its prediction as c, the targeted adversarial attack objective is to find some perturbation $\boldsymbol{\delta} \in \mathbb{R}^d$ to be superposed on the original image such that $\mathcal{M}$ now mis-classify the perturbed image to another target class t. In this work, we use the identical CNN models to the previous works (Ru et al., 2020b; Tu et al., 2019; Alzantot et al., 2019), which approximately gives 80% validation accuracy on the CIFAR-10 dataset. Ru et al. (2020b) further claim that the query efficiency of the BayesOpt attack strategy can be enhanced by searching the perturbation over a latent space $\tilde{\boldsymbol{\delta}} \in \mathbb{R}^{d_r}$ with reduced dimension $d_r \ll d$ and upsampling it back to the original high-resolution image space $\mathbb{R}^d$. This leads to a categorical variable which is the downsampling/upsampling technique, and in this work we have 3 options: bilinear, nearest and bicubic interpolations. In our attack on CIFAR10 images, we set $d = 32 \times 32 \times 3$ and $d_r = 14 \times 14 \times 3$ following Ru et al. (2020b).

In our work, we adopt a *sparse* setup where instead of perturbing all the pixels, we only perturb one pixel per row per colour in the latent space, allowing a total of $s = 14 \times 3 = 42$ pixels in the reduced space to take non-zero values. Such setup corresponds to add perturbation to some pixels of the original image only, which is more actionable in real life (for e.g., to evade real-life image classifiers this only requires one to carefully manipulate some parts of a printed image; this is contrasted to L_2 attack, another often studied setup where we perturb a small amount on *every* pixel of the image which is less feasible in real life). We additionally impose a constraint on the pixels ϵ to limit the maximum amount of perturbations. Mathematically, the goal is formulated as:

$$\arg\max_{j\in\{1,\dots,C\}} \mathcal{M}\big(I + \text{Upsample}(\tilde{\boldsymbol{\delta}})\big)_j = t.$$

$$\text{s.t. } ||\{\tilde{\delta}_i \mid \tilde{\delta}_i \neq 0\}|| \leq s \text{ and } ||\tilde{\boldsymbol{\delta}}||_\infty \leq \epsilon, \quad (10)$$

where the first $||\cdot||$ denote the cardinality of the set of non-zero elements of $\tilde{\boldsymbol{\delta}}$ and the second $||\cdot||$ is the L_∞ norm.

In summary, the variables $\mathbf{z} \in \mathbb{R}^{85}$ that we need to search over include 42 categorical variables deciding the positions of the pixels to be perturbed at each row (thus 14 choices for each variable), 1 categorical variable on the type of upsampling technique chosen (3 choices) and 42 continuous variables defining the amount of perturbation to be added to each chosen pixel. These setups conveniently cast the problem of finding adversarial perturbation as a mixed continuous-categorical optimisation problem for which CASMOPOLITAN is suitable. In this case, we follow Ru et al. (2020b) and select the following as the objective function f we aim to maximise:

$$f(\mathbf{z}) = \Big[\log \mathcal{M}\big(I + \boldsymbol{\delta}(\mathbf{z})\big)_t - \log \mathcal{M}\big(I + \boldsymbol{\delta}(\mathbf{z})\big)_c\Big], \quad (11)$$

where $\boldsymbol{\delta}(\mathbf{z})$ is the image perturbation $\boldsymbol{\delta}$ induced by our combined choices of the pixel locations and the corresponding amount of perturbations. Essentially, in Eq. (11), we aim to maximise the difference between the logit value of the target class t and the true class c, and trivially the attack succeeds if and when $f(\mathbf{z}) > 0$. Thus, we terminate each experiment either the attack succeeds or the maximum budget (250) is reached. It is finally worth noting that the combined dimension $\mathbf{z}$ is 85-dimensional whose one-hot transformed dimension amounts to 633, which is clearly beyond the common scope of usage of vanilla GP-BO that neither gives special treatments to the categorical dimensions nor is tailored for high-dimensional optimisation.

C.3. Experimental setup

We run all experiments on a shared Intel Xeon server with 256GB of RAM. For all categorical problems, we run 20 random trials with the exception of BOCS on Contamination Control, where we only run 5 trials due to the very long running time of BOCS and our computing constraints (reported in App. B.1). For the mixed problems, we follow Ru et al. (2020a), where we run 20 trials for the synthetic problems and 10 trials for the real-life problems. For black-box attack, we run attack once on all 450 attack instances on 50 images. We report mean and standard error in all cases.

CASMOPOLITAN Our algorithm introduces a number of additional hyperparameters relating to the initialisation, adjustment and restarting of the trust regions. In the categorical space, the distances (Hamming distance) are always integers, and the minimum ($L^h_{\min}$) and maximum ($L^h_{\max}$) trust region sizes are always set to 0 and the dimensionality of the problem (i.e. the diameter of the combinatorial graph), respectively. The failure tolerance, which is the number of successive failures in increasing the best objective function value before shrinking the trust region size (`fail_tol`), is set to 40 unless otherwise specified; the success tolerance (`succ_tol`), which is the number of successive successes in increasing the best function value before expanding the trust region, is set to 2. We investigate the sensitivity in performance of our algorithm to these hyperparameters in App. B3. The only other GP hyperparameter is the amount of noise (or jitter), where we constrain the noise variance in the interval of $[10^{-5}, 0.1]$ and this value is learnt as a hyperparameter during the log-marginal likelihood optimisation. We always start the experiments with 20 initial randomly sampled points.

In the mixed setting that involves continuous variables, unless specified otherwise we always use the Matern 5/2 kernel. In these continuous problems, we bound the lengthscale in the range of $[0.01, 0.5]$ and outputscale in $[0.5, 5]$ and in all cases, we normalise the continuous inputs $\mathbf{x}$ into hypercubes $[0, 1]^{d_x}$ and standardise the targets by their mean and standard deviation from the initially randomly sampled locations $\mathbf{y}$: $\tilde{\mathbf{y}} = \frac{y-\bar{y}}{\sigma(y)}$. When we compute the mixed kernel in Eq. (4), we set $\lambda = 0.5$ as it is empirically shown to perform the best in Ru et al. (2020a) that initially propose this kernel. On the hyperparameters specific to the continuous trust regions, since they are identical to those introduced in TURBO (Eriksson et al., 2019), we do not change these settings from their default values ($L^x_{\min} = 0.5^7, L^x_{\max} = 1.6, L^x_0 = 0.8$) with the exceptions of α_s (and hence α_e), `succ_tol` and `fail_tol` which all follow the settings of the categorical trust regions described above, instead of being independent hyperparameters. For the trade-off parameter β_i at each restart, we follow the common practice of setting β_i to a constant value (Berkenkamp et al., 2019). In our case, we set $\sqrt{\beta_i} = 1.96$ as it has been shown this value of β_i performs well for a variety of BO tasks (Berkenkamp et al., 2019).

During optimisation of the acquisition function, we use the local search strategy (for categorical optimisation; in Sec. 3.1) or the interleaved strategy (for mixed optimisation; in Sec. 3.2). In all cases, we initialise the search at the best location found so far, and we set the maximum number of local/interleaved search to be 100; for interleaved search in mixed space, one local search move + one gradient-based optimisation step count as one interleaved search step, we use Adam (Kingma and Ba, 2015) as the default optimiser for the log-marginal likelihood with learning rate 0.1 and maximum step 100, although we find the performance to be relatively stable at least for maximum step in the range of $[100, 300]$ and learning rate in the range of $[0.03, 0.3]$. By default, we optimise the log-marginal likelihood 3 times and select the point(s) with the largest acquisition function across the 3 runs, although we do not find optimising with just 1 restart to be significantly worse. In this work we use expected improvement (EI) as the acquisition function, although our work is compatible with any other common choice such as GP-UCB or Thompson sampling. When CASMOPOLITAN is run in the batch setting, we use the Kriging Believer strategy (Ginsbourger et al., 2010) to select b points simultaneously: specifically, given observation data $D_t = \{\mathbf{z}_i, y_i\}^t_{i=0}$ and a GP model, we first optimise the acquisition function as usual to propose the first out of the b points required:

$$\mathbf{z}^{(1)}_t = \arg\max \alpha(\mathbf{z} \mid D_t) \tag{12}$$

We then fully trust $\mu(\mathbf{z}^{(1)}_t)$, the predictive mean at $\mathbf{z}^{(1)}_t$, as a perfect proxy of the true objective function value $f(\mathbf{z}^{(1)}_t)$, and use this "hallucinated" input-output tuple $(\mathbf{z}^{(1)}_t, \mu(\mathbf{z}^{(1)}_t))$ to update the GP $D_{t-1} \leftarrow D_{t-1} \cup (\mathbf{z}^{(1)}_t, \mu((z^{(1)}_t))$. Conditioned on this GP with "hallucinated" inputs and observations, we then optimise the acquisition function value again to propose the second point $\mathbf{z}^{(2)}_t$ and this process is repeated until all b proposals are selected.

Other baselines Where possible and open-sourced, for the other baselines we use the implementation from their respective original authors:

- **TURBO** We use the official implementations made available by Eriksson et al. (2019) at `https://github.com/uber-research/TuRBO`. For the implementation of TURBO in categorical space or where categorical variables are involved, we use the modified implementation supplied by the organisers of the 2020 NeurIPS Black-box Optimisation Challenge which includes TURBO as a baseline that additionally supports one-hot transformation on the categorical variables (`https://github.com/rdturnermtl/bbo_challenge_starter_kit`). Note that for the vanilla one-hot GP-BO, we also adapt from the TURBO implementation but remove the additional features such as trust regions and restarting.
- **COCABO** We use the official implementation by Ru et al. (2020a) at `https://github.com/rubinxin/CoCaBO_code`. Note that in original COCABO, there is an option for the value of λ in Eq. (4) to be optimised as a hyperparameter within bounds of $[0, 1]$; in our work for fairness of comparison, we fix λ to 0.5 since it is the value used in our method. It is worth noting that $\lambda = 0.5$ is also shown to be performing overall the best in COCABO from the results reported in Ru et al. (2020a).
- **MVRSM** We use the official implementation by Bliek et al. (2020) at `https://github.com/lbliek/MVRSM`.
- **COMBO** We use the official implementation by Oh et al. (2019) at `https://github.com/QUVA-Lab/COMBO`.
- **BOCS** We use the official implementation by Baptista and Poloczek (2018) at `https://github.com/baptistar/BOCS`.
- **TPE** TPE (Tree Parzans Estimator) is implemented by the Hyperopt python package, available at `http://hyperopt.github.io/hyperopt/`.
- **SMAC** We use the implementation at `https://github.com/automl/SMAC3`.

D. Proofs and Further Theoretical Analysis

D.1. Lemma D.1

Lemma D.1. *The proposed categorical kernel in Eq. (1) and mixed kernel in Eq. (4) are valid kernels (i.e. positive semi-definite kernels).*

Proof. For the categorical kernel in Eq. (1), we have that exponential of a kernel is also a kernel, and since the categorical overlap kernel is a valid kernel (Ru et al., 2020a), its exponentiated version is also a valid kernel. For the mixed kernel in Eq. (4), since addition and multiplication between kernels result in valid kernels, and since both $k_x(.,.)$ and $k_h(.,.)$ are valid kernels, therefore, the mixed kernel in Eq. (4) is also a valid kernel. $\square$

D.2. Proof of Theorem 3.1

In this section, we derive the maximum information gain of the categorical kernel k_h (Section D.2.1) and the mixed kernel k (Section D.2.2).

D.2.1. Maximum information gain of the categorical kernel

We derive the maximum information gain of the categorical kernel k_h proposed in Eq. (1) by bounding $\gamma(T; k_h; \mathcal{H})$ directly. Let us first consider the case when the objective function f has only one categorical variable h with n distinct values (i.e. $h \in \{A_1, A_2, \dots, A_n\}$ where A_i is a categorical value and $A_i \neq A_j$ when $i \neq j$). Let us consider T data points $h_1, h_2, \dots, h_T$, then its corresponding covariance matrix K_T is $[k_h(h_i, h_j)]_{i,j=1}^T$. As the maximum information gain $\gamma(T; k_h; \mathcal{H})$ is equal to $\log|I_T + \sigma^{-2}K_T|$ where I_T is the identity matrix of size T,[7] thus, we will bound $\gamma(T; k_h; \mathcal{H})$ by bounding $\log|I_T + \sigma^{-2}K_T|$. Our general idea is to perform a decomposition of K_T, i.e. expressing $K_T = \Phi E \Psi^T$ where $\Phi \in \mathbb{R}^{T\times n}$, $\Psi \in \mathbb{R}^{T\times n}$, and $E \in \mathbb{R}^{n\times n}$, and then apply the Sylvester's determinant theory and the Hadamard's inequality to derive an upper bound for $\log|I_T + \sigma^{-2}K_T|$.

In the sequel, for ease of notation, we define the function q as a mapping from A_i to i. In particular, $q(A_i) = i,\ \forall i = 1, \dots, n$. With the categorical kernel $k_h(h, h') = \exp(l\delta(h, h'))$, in the following, we will prove that K_T can be decomposed as,[8]

$$K_T = \Phi E \Psi^T, \tag{13}$$

where $\Phi, \Psi \in \mathbb{R}^{T\times n}, E \in \mathbb{R}^{n\times n}$, and

$$\Phi = \begin{bmatrix} \phi(h_1) \\ \phi(h_2) \\ \dots \\ \phi(h_T) \end{bmatrix}, \Psi = \begin{bmatrix} \psi(h_1) \\ \psi(h_2) \\ \dots \\ \psi(h_T) \end{bmatrix},$$
$$E = \text{diag}(\exp(l) + n - 1, \exp(l) - 1, ..., \exp(l) - 1),$$

with $\phi(h_i)$ being an n-dimensional row vector with 1 at the 1st column, (-1) at the $q(h_i)$-th column, and 1 at the

[7] $|S|$ denotes the determinant of matrix S.

[8] When $T = n$, this decomposition is equivalent to the eigendecomposition. That is, the diagonal of matrix E consists of the eigenvalues of K_T and each column of Φ is an eigenvector of K_T.

$(q(h_i)+1)$-th column, i.e.,

$$\phi(h_i) = \begin{cases} [1\ 1\ 0\ ...\ 0\ 0], & \text{if } q(h_i)=1 \\ [1\ 0\ 0\ ...\ 0\ (-1)\ 1\ 0 \ldots 0], & \text{if } 1 < q(h_i) < n\ , \\ [1\ 0\ 0\ ...\ 0\ (-1)], & \text{if } q(h_i) = n \end{cases}$$

and $\psi(h_i)$ being an n-dimensional row vector with the following formula,

$$\psi(h_i) = \begin{cases} \frac{1}{n}[1\ (n-1)\ (n-2)\ ...\ 1], & \text{if } q(h_i)=1 \\ \frac{1}{n}[1\ (-1)\ ...\ -(q(h_i)-1)\ (n-q(h_i))\ ...\ 1], & \\ & \text{if } 1 < q(h_i) < n \\ \frac{1}{n}[1\ (-1)\ ...\ -(n-1)], & \text{if } q(h_i)=n. \end{cases}$$

To prove the decomposition in Eq. (13), we compute the element at the i-th row and j-th column of $\Phi E \Psi^T$, i.e. $[\Phi E \Psi^T]_{ij}$, and then prove that $[\Phi E \Psi^T]_{ij}$ is equal to $[K_T]_{ij}$. To compute $[\Phi E \Psi^T]_{ij}$, it can be directly seen that,

$$[\Phi E \Psi^T]_{ij} = \sum_{r=1}^{n} \phi_r(h_i) E_r \psi_r(h_j),$$

where $\phi_r(h_i)$ denotes the r-th element of $\phi(h_i)$, $\psi_r(h_j)$ denotes the r-th element of $\psi(h_j)$ and E_r denotes the r-th element on the diagonal of matrix E. We then consider the following three cases:

Case 1: $q(h_j) = q(h_i)$. First, let us consider $1 < q(h_i) < n$, then we have,

$$\begin{aligned}[\Phi E \Psi^T]_{ij} =& 1 \times (\exp(l)+n-1) \times \frac{1}{n} \\ &+ (-1) \times (\exp(l)-1) \times \frac{(-q(h_i)+1)}{n} \\ &+ 1 \times (\exp(l)-1) \times \frac{(n-q(h_i))}{n} \\ =& \exp(l).\end{aligned}$$

Note that when $q(h_j) = q(h_i)$, we will have $h_j = h_i$, thus, the element $[K_T]_{ij}$ is equal to $\exp(l)$. Similar arguments can be made when $q(h_i)=1$ or $q(h_i)=n$, that is, $[K_T]_{ij}$ is equal to $\exp(l)$. Therefore, $[\Phi E \Psi^T]_{ij} = [K_T]_{ij} = \exp(l)$.

Case 2: $q(h_j) \geq q(h_i)+1$. Let us first consider $1 < q(h_i)$, then,

$$\begin{aligned}[\Phi E \Psi^T]_{ij} =& 1 \times (\exp(l)+n-1) \times \frac{1}{n} \\ &+ (-1) \times (\exp(l)-1) \times \frac{(-q(h_i)+1)}{n} \\ &+ 1 \times (\exp(l)-1) \times \frac{(-q(h_i))}{n} \\ =& 1.\end{aligned}$$

In this case, with $q(h_j) \geq q(h_i)+1$, we will have $h_i \neq h_j$, hence, the element $[K_T]_{ij}$ is equal to 1. Similar arguments can be made when $q(h_i)=1$, that is, in this case, $[K_T]_{ij}$ is also equal to 1. Therefore, $[\Phi E \Psi^T]_{ij} = [K_T]_{ij} = 1$.

Case 3: $q(h_j) \leq q(h_i)-1$. Let us first consider $q(h_i) < n$, then,

$$\begin{aligned}[\Phi E \Psi^T]_{ij} =& 1 \times (\exp(l)+n-1) \times \frac{1}{n} \\ &+ (-1) \times (\exp(l)-1) \times \frac{(n-q(h_i))}{n} \\ &+ 1 \times (\exp(l)-1) \times \frac{(n-q(h_i)-1)}{n} \\ =& 1.\end{aligned}$$

Similar to *Case* 2, we also have $h_i \neq h_j$. Similar arguments can be made when $q(h_i)=n$, $[K_T]_{ij}$ is also equal to 1. Hence, $[\Phi E \Psi^T]_{ij} = [K_T]_{ij} = 1$.

Combining *Cases* 1, 2, 3, we proved the decomposition in Eq. (13). Now using this decomposition, we have,

$$\gamma(T; k_h; \mathcal{H}) = \log |I_T + \sigma^{-2} K_T| = \log |I_T + \sigma^{-2} \Phi E \Psi^T|.$$

By Sylvester's determinant theorem (Sylvester, 1851),

$$\gamma(T; k_h; \mathcal{H}) = \log |I_n + \sigma^{-2} \Psi^T \Phi E|. \quad (14)$$

Next, we prove the matrix $\Psi^T \Phi E$ is a positive semi-definite (p.s.d.) matrix, and the maximum element on the diagonal of $\Psi^T \Phi E$ is equal or less than $T(\exp(l)+n-1)$. Let us denote m_i as the number of times the categorical value A_i appears in T data points. It can be directly seen that,

$$[\Psi^T \Phi]_{ij} = \sum_{r=1}^{T} \psi_i(h_r)\phi_j(h_r) = \sum_{r=1}^{n} m_r \psi_i(A_r)\phi_j(A_r).$$

Hence, the matrix $\Psi^T \Phi$ can be written as,

$$\Psi^T \Phi = \Psi_A^T F \Phi_A, \qquad \Phi_A, \Psi_A, E \in \mathbb{R}^{n \times n}, \quad (15)$$

where

$$\Phi_A = \begin{bmatrix} \phi(A_1) \\ \phi(A_2) \\ \ldots \\ \phi(A_n) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & ... & 0 & 0 \\ 1 & -1 & 1 & 0 & \ldots & 0 & 0 \\ 1 & 0 & -1 & 1 & \ldots & 0 & 0 \\ & & & & \ldots & & \\ 1 & 0 & 0 & 0 & \ldots & -1 & 1 \\ 1 & 0 & 0 & 0 & \ldots & 0 & -1 \end{bmatrix},$$

$$\Psi_A = \begin{bmatrix}\psi(A_1)\\ \psi(A_2)\\ \dots\\ \psi(A_n)\end{bmatrix},$$
$$= \frac{1}{n}\begin{bmatrix}1 & n-1 & n-2 & \dots & 2 & 1\\ 1 & -1 & n-2 & \dots & 2 & 1\\ 1 & -1 & -2 & \dots & 2 & 1\\ & & & & \dots & \\ 1 & -1 & -2 & \dots & -(n-2) & 1\\ 1 & -1 & -2 & \dots & -(n-2) & -(n-1)\end{bmatrix},$$
$$F = \text{diag}(m_1, m_2, \dots, m_n).$$

It is straightforward that $\Psi_A^T\Phi_A = I_n$, thus, from Eq. (15), we can see that $\Psi_A^T F\Phi_A$ is an eigendecomposition of $\Psi^T\Phi$, and hence, the eigenvalues of $\Psi^T\Phi$ are $m_1, m_2, \dots, m_n$. As $m_i \geq 0, \forall i = 1, ..., n$, so $\Psi^T\Phi$ is a p.s.d. matrix, and therefore, $\Psi^T\Phi E$ is also a p.s.d. matrix. Besides, note that the r-th element on the diagonal of $\Psi^T\Phi E$ can be computed as $[\Psi^T\Phi]_{rr}E_r$ where $[\Psi^T\Phi]_{rr}$ and E_r are the r-th elements on the diagonal of $\Psi^T\Phi$ and E, respectively. Since $[\Psi^T\Phi]_{rr} \leq \sum_{i=1}^T (n-1)/n \times 1 \leq T$, and $E_r \leq (\exp(l) + n - 1)$, hence, $[\Psi^T\Phi]_{rr}E_r \leq T(\exp(l) + n - 1)$. This results that the maximum element on the diagonal of $\Psi^T\Phi E$ is equal or smaller than $T(\exp(l) + n - 1)$.

Combining Eq. (14) and the Hadamard's inequality (Mazya and Shaposhnikova, 1999) on the positive semi-definite matrix $\Psi^T\Phi E$, we have,

$$\gamma(T; k_h; \mathcal{H}) \leq \log|I_n + \sigma^{-2}W|,$$

where $W = \text{diag}(\text{diag}^{-1}(\Psi^T\Phi E))$. Since the maximum element on the diagonal of $\Psi^T\Phi E$ is equal or smaller than $T(\exp(l) + n - 1)$. Therefore, $\gamma(T; k_h; \mathcal{H}) = \mathcal{O}(n\log(1 + \sigma^{-2}T(\exp(l) + n - 1))) = \mathcal{O}(n\log T)$. $\square$

Now let consider the case when the objective function f has d_h categorical variables where each variable has n_j distinct values. This can be considered to be equivalent to the case when f has one variable with $\prod_{j=1}^{d_h} n_j$ distinct values. Thus, the same proof can be used, and we have $\gamma(T; k_h; \mathcal{H}) = \mathcal{O}\big((\prod_{j=1}^{d_h} n_j)\log T\big)$. $\square$

D.2.2. Maximum information gain of the mixed kernel

We make use of Theorems 2 and 3 in Krause and Ong (2011) to bound the maximum information gain of the mixed kernel k. In particular, Theorem 2 states that given two kernels: k_h on $\mathcal{H}$ and k_x on $\mathcal{X}$, and if k_h is a kernel on $\mathcal{H}$ with rank at most m, then $\gamma(T; k_h k_x; [\mathcal{H}, \mathcal{X}]) \leq m\gamma(T; k_x; \mathcal{X}) + m\log T$. On the other hand, Theorem 3 states that for any two kernels k_h on $\mathcal{H}$ and k_x on $\mathcal{X}$, then $\gamma(T; k_h + k_x; [\mathcal{H}, \mathcal{X}]) \leq \gamma(T; k_x; \mathcal{X}) + \gamma(T; k_h; \mathcal{X}) + 2\log T$.

As proven in Section D.2.1, the kernel k_h has at most rank $\tilde{N} = \prod_{j=1}^{d_h} n_j$ (based on the eigendecomposition). Thus, using Theorem 2 in Krause and Ong (2011), we have

$$\gamma(T; k_h k_x; [\mathcal{H}, \mathcal{X}]) \leq \tilde{N}\gamma(T; k_x; X) + \tilde{N}\log T. \quad (16)$$

Similarly, using Theorem 3 in Krause and Ong (2011), we obtain

$$\gamma(T; k_h + k_x; [\mathcal{H}, \mathcal{X}]) \leq \mathcal{O}\big(\gamma(T; k_x; X) + (\tilde{N} + 2)\log T\big). \quad (17)$$

We have the mixed kernel k defined as $\lambda(k_x k_h) + (1-\lambda)(k_h + k_x)$ where $\lambda \in [0, 1]$ is a trade-off parameter. By combining Eqs. (16) and (17), we have,

$$\begin{aligned}\gamma(T; k; [\mathcal{H}, \mathcal{X}] &\leq \lambda\mathcal{O}\big(\tilde{N}\gamma(T; k_x; X) + \tilde{N}\log T\big)\\ &\quad + (1-\lambda)(\gamma(T; k_x; X) + (\tilde{N} + 2)\log T)\\ &\leq \mathcal{O}\big((\tilde{N}\lambda + 1 - \lambda)\gamma(T; k_x; X)\\ &\quad + (\tilde{N} + 2 - 2\lambda)\log T\big). \quad \square\end{aligned}$$

D.3. Proof of Theorem 3.2

We prove that under Assumptions 3.1 & 3.2, after a restart, (1) if CASMOPOLITAN terminates after a finite number of iterations, then it converges to a local maxima of f, or, (2) if CASMOPOLITAN does not terminate after a finite number of iterations, then it converges to the global maximum of f. We prove this property by contradiction.

First, let us assume after a restart, case (2) occurs, i.e. CASMOPOLITAN does not terminate after a finite number of iterations. This means when the iteration t goes to infinity, the TR length L^h is not shrunk below L^h_{min} in the categorical setting, or, both L^h and L^x are not shrunk below L^h_{min} and L^x_{min} in the mixed space setting. From the algorithm description, the TR is shrunk after `fail_tol` consecutive failures. Thus, if after $N_{\text{min}} =$ `fail_tol` $\times\, m$ iterations where $m = \lceil\log_{\alpha_e}(L^h_0/L^h_{\text{min}})\rceil$[9] in the categorical setting and $m = \max(\lceil\log_{\alpha_e}(L^h_0/L^h_{\text{min}})\rceil, \lceil\log_{\alpha_e}(L^x_0/L^x_{\text{min}})\rceil)$ in the mixed space setting, there is no success, CASMOPOLITAN terminates. This means, in order for case (2) to occur, CASMOPOLITAN needs to have at least one improvement per N_{min} iterations. Let consider the series $\{f(\mathbf{z}^k)\}_{k=1}^{\infty}$ where $f(\mathbf{z}^k) = \max_{i=(k-1)N_{\text{min}}+1,\dots,kN_{\text{min}}}\{f(\mathbf{z}_i)\}$ and $f(\mathbf{z}_i)$ is the function value at iteration i. This series is strictly increasing and the objective function $f(\mathbf{z})$ is bounded (Assumption 3.1). Thus, using the monotone convergence theorem (Bibby, 1974), this series converges to the global maximum of the objective function f.

Second, let consider case (1) occurs, i.e. CASMOPOLITAN terminates after a finite number of iterations. We will prove that in this case, CASMOPOLITAN converges to a local maxima of $f(\mathbf{z})$ given Assumption 3.2. For simplicity,

[9]The operator $\lceil . \rceil$ denotes the ceiling function

let us consider the categorical setting first. Let us denote L_s as the largest TR length that after being shrunk, the algorithm terminates. By the definition of L_s, we have $\lfloor \alpha_s L_s \rfloor \leq L^h_{\min}$.[10] Due to $\lfloor \alpha_s L_s \rfloor \leq \alpha_s L_s < \lfloor \alpha_s L_s \rfloor + 1$, we have $L_s < (L^h_{\min} + 1)/\alpha_s$. And because L_s is an integer, we finally have $L_s \leq \lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1$. By choosing $L_s = \lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1$, we have that $\forall L > L_s, \alpha_s L \geq \alpha_s \lceil (L^h_{\min} + 1)/\alpha_s \rceil > L^h_{\min}$. This says that for all TR with length $L > L_s$, after being shrunk one time, the algorithm doesn't terminate yet. Therefore, $L_s = \lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1$ is the largest TR length that after being shrunk, the algorithm terminates. This tells us when the TR length first becomes smaller or equal than L_s, CASMOPOLITAN does not terminate yet (*Conclusion* 1). In addition, since GP can fit f accurately within a TR with length L_s (Assumption 3.2), for any TR with length $L \leq L_s$, the solution of BO is a success (*Conclusion* 2). Combining *Conclusions* 1 & 2, we have that when TR length first becomes smaller or equal than L_s, if the current TR center is not a local maxima, CASMOPOLITAN can find a new data point whose function value larger than the function value of current TR center. Thus, in the next iteration, the TR still keeps the same length whilst having center as the new found data point. This process occurs iteratively until a local maxima is reached (i.e. when CASMOPOLITAN fails to improve from the current center), and CASMOPOLITAN terminates.

Similar arguments can be made for the mixed space setting. Let us remind that for the mixed space setting, CASMOPOLITAN terminates when either the continuous TR length $\leq L^x_{\min}$ or the categorical TR length $\leq L^h_{\min}$. Now let us consider two cases. Case (i): when the continuous TR reaches $L^x_{\min}/\alpha_s$, the corresponding length of the categorical TR is $\lceil L^h_0 L^x_{\min}/(\alpha_s L^x_0) \rceil$. Case (ii): when the categorical TR length reaches $\lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1$, the corresponding length of the continuous TR is $L^x_0(\lceil (L^h_{\min}+1)/\alpha_s \rceil - 1)/L^h_0$. Based on Assumption 3.2, GP can fit accurately a TR with continuous length $L^x \leq \max\left(L^x_{\min}/\alpha_s, L^x_0(\lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1)/L^h_0\right)$ and $L^h \leq \max\left(\lceil (L^h_{\min} + 1)/\alpha_s \rceil - 1, \lceil L^h_0 L^x_{\min}/(\alpha_s L^x_0) \rceil\right)$, then when Case (i) or Case (ii) occurs, the GP approximates accurately the objective function f within the corresponding TR, and thus similar argument as in the categorical setting can be made. That is, if the current TR center is not a local maxima, then CASMOPOLITAN can find a new data point whose function value larger than the function value of current TR center. And this process occurs iteratively until a local maxima is reached, and CASMOPOLITAN terminates. □

D.4. Proof of Theorem 3.3

Let us first remind our restart strategy in the categorical setting. At the i-th restart, we first fit an auxiliary global GP model $GP(0, k_h)$ on a subset of data $D^*_{i-1} = \{\mathbf{h}^*_j, f(\mathbf{h}^*_j)\}^{i-1}_{j=1}$, where $\mathbf{h}^*_j$ is the local maxima found after the j-th restart, or, a random data point, if the found local maxima after the j-th restart is same as one of previous restart. Let us also denote $\mu_{gl}(\mathbf{h}; D^*_{i-1})$ and $\sigma^2_{gl}(\mathbf{h}; D^*_{i-1})$ as the posterior mean and variance of the global GP learned from D^*_{i-1}. Then, at the i-th restart, we select the following location $\mathbf{h}^{(0)}_i$ as the initial centre of the new TR:

$$\mathbf{h}^{(0)}_i = \arg\max_{\mathbf{h}\in\mathcal{H}} \mu_{gl}(\mathbf{h}; D^*_{i-1}) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{h}; D^*_{i-1}),$$

where β_i is the trade-off parameter in GP-UCB (Srinivas et al., 2010).

To prove the convergence property of CASMOPOLITAN, apart from Assumptions 3.1 & 3.2, let us also assume that at the i-th restart, there exists a function $g_i(\mathbf{h})$ that: (a) is a sample from the global $GP(0, k_h)$, (b) shares the same global maximum $\mathbf{h}^*$ with f, and, (c) passes through the all the local maxima of f and any data point $\mathbf{h}'$ in $\mathcal{D}^*_{i-1} \cup \{\mathbf{h}^{(0)}_i\}$ that are not local maxima (i.e. $g_i(\mathbf{h}') = f(\mathbf{h}')\ \forall \mathbf{h}' \in D^*_{i-1} \cup \{\mathbf{h}^{(0)}_i\}$). In layman's terms, the function $g_i(\mathbf{h})$ is a function that passes through all the maxima of f and is a sample from the auxiliary global $GP(0, k_h)$. It is worth noting that our assumption is more relaxed than the assumption in Srinivas et al. (2010) where it is assumed that the objective function f must be sampled from the global GP$(0, k_h)$. Specifically, it can be seen that if the assumption in Srinivas et al. (2010) holds, our assumption also holds because if $f(\mathbf{h})$ is a sample from GP$(0, k_h)$, then a choice for $g_i(\mathbf{h})$ is $f(\mathbf{h})$, thus, our assumption holds.

Using Lemmas 5.1 and 5.2 in Srinivas et al. (2010) for the function g_i, when $\beta_i = 2\log(|\mathcal{H}|i^2\pi^2/6\zeta)$, for all i, with probability $1-\zeta$, we have,

$$\begin{aligned}&\mu_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1})\\ &\quad \geq \mu_{gl}(\mathbf{h}^*; D^*_{i-1}) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^*; D^*_{i-1})\\ &\quad \geq g_i(\mathbf{h}^*).\end{aligned}$$

Thus, with probability $1-\zeta$,

$$\begin{aligned}&g_i(\mathbf{h}^*) - g_i(\mathbf{h}^{(0)}_i)\\ &\leq \mu_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}) - g_i(\mathbf{h}^{(0)}_i)\\ &\leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}).\end{aligned}$$

Combining this inequality with the fact that $g_i(\mathbf{h}^{(0)}_i) = f(\mathbf{h}^{(0)}_i)$, and $g_i(\mathbf{h}^*_i) = f(\mathbf{h}^*_i)$, we have, with probability $1-\zeta$,

$$f(\mathbf{h}^*) - f(\mathbf{h}^{(0)}_i) \leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}).$$

Let us denote $\mathbf{h}^*_i$ as the local maxima found by CASMOPOLITAN at the i-th restart. As $f(\mathbf{h}^{(0)}_i) \leq f(\mathbf{h}^*_i)$, therefore,

$$f(\mathbf{h}^*) - f(\mathbf{h}^*_i) \leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{h}^{(0)}_i; D^*_{i-1}).$$

[10]The operator $\lfloor . \rfloor$ denotes the floor function

This results that, with probability $1-\zeta$,

$$R_I = \sum_{i=1}^{I}(f(\mathbf{h}^*) - f(\mathbf{h}_i^*)) \leq \sum_{i=1}^{I} 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{h}_i^{(0)}; D_{i-1}^*).$$

Finally, using Lemmas 5.3 and 5.4 in Srinivas et al. (2010), we can bound R_I as $R_I \leq \sqrt{IC_1\beta_I\gamma(I;k_h,\mathcal{H})}$ with $C_1 = 8/\log(1+\sigma^{-2})$ and $\gamma(I;k_h,\mathcal{H})$ being the maximum information gain for the categorical kernel derived in Theorem 3.1. □

D.5. Proof of Theorem 3.4

Similar to the proof for categorical setting in Section D.4, let us first remind our restart strategy in the mixed space setting. Suppose we are restarting the i-th time, we first fit the global GP model on a subset of data $D_{i-1}^* = \{\mathbf{z}_j^*, f(\mathbf{z}_j^*)\}_{j=1}^{i-1}$, where $\mathbf{z}_j^*$ is the local maxima found after the j-th restart, or, a random data point, if the found local maxima after the j-th restart is same as one of previous restart. Let us also denote $\mu_{gl}(\mathbf{z}; D_{i-1}^*)$ and $\sigma_{gl}^2(\mathbf{z}; D_{i-1}^*)$ as the posterior mean and variance of the global GP learned from D_{i-1}^*. Then, at the i-th restart, we select the following location $\mathbf{z}_i^{(0)}$ as the initial centre of the new TR:

$$\mathbf{z}_i^{(0)} = \arg\max_{\mathbf{z}\in[\mathcal{H},\mathcal{X}]} \mu_{gl}(\mathbf{z}; D_{i-1}^*) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{z}; D_{i-1}^*),$$

where β_i is the trade-off parameter in GP-UCB (Srinivas et al., 2010).

To prove the convergent property of CASMOPOLITAN in the mixed space setting, apart from Assumptions 3.1 & 3.2, let us also assume that at the i-th restart, there exists a function $g_i(\mathbf{z})$: (a) lies in the RKHS $\mathcal{G}_k([\mathcal{H},\mathcal{X}])$ and $\|g_i\|_k^2 \leq B$, (b) shares the same global maximum $\mathbf{z}^*$ with f, and, (c) passes through all the local maxima of f and any data point $\mathbf{z}'$ in $\mathcal{D}_{i-1}^* \cup \{\mathbf{z}_i^{(0)}\}$ which are not local maxima (i.e. $g_i(\mathbf{z}') = f(\mathbf{z}')\ \forall \mathbf{z}' \in D_{i-1}^* \cup \{\mathbf{z}_i^{(0)}\}$). In layman's terms, the function $g_i(\mathbf{z})$ is a function that passes through the maxima of f whilst lying in the RKHS $\mathcal{G}_k([\mathcal{H},\mathcal{X}])$ and satisfying $\|g_i\|_k^2 \leq B$. Our assumption is more relaxed than Srinivas et al. (2010) which assumed that the objective function f lies in the RKHS $\mathcal{G}_k([\mathcal{H},\mathcal{X}])$. Specifically, it can be seen that if the assumption in Srinivas et al. (2010) holds, our assumption also holds because if $f(\mathbf{z})$ lies in the RKHS $\mathcal{G}_k([\mathcal{H},\mathcal{X}])$, then a choice for $g_i(\mathbf{z})$ is $f(\mathbf{z})$, thus, our assumption holds.

Using Theorem 6 in Srinivas et al. (2010) for function g_i, when $\beta_i = 2\|g_i\|_k^2 + 300\gamma_i\log(i/\zeta)^3$, $\forall i, \forall z \in [\mathcal{H},\mathcal{X}]$, we have,

$$\Pr\{|\mu_{gl}(\mathbf{z}; D_{i-1}^*) - g_i(\mathbf{z})| \leq \sqrt{\beta_i}\sigma_{gl}(\mathbf{z}; D_{i-1}^*)|\} \geq 1-\zeta. \tag{18}$$

Note since $\|g_i\|_k^2 \leq B$, Eq. (18) is also correct using $\beta_i = 2B + 300\gamma_i\log(i/\zeta)^3$. By using the inequality in Eq. (18), the proof technique is similar to that in Section D.4. In particular, with probability $1-\zeta$, we have that,

$$\begin{aligned}&\mu_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*)\\ &\quad \geq \mu_{gl}(\mathbf{h}^*; D_{i-1}^*) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{z}^*; D_{i-1}^*) \geq g_i(\mathbf{z}^*).\end{aligned} \tag{19}$$

Thus, with probability $1-\zeta$, we have

$$\begin{aligned}&g_i(\mathbf{z}^*) - g_i(\mathbf{z}_i^{(0)})\\ &\leq \mu_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*) + \sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*) - g_i(\mathbf{z}_i^{(0)})\\ &\leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*).\end{aligned}$$

Since $g_i(\mathbf{z}_i^{(0)}) = f(\mathbf{z}_i^{(0)})$, and $g_i(\mathbf{z}_i^*) = f(\mathbf{z}_i^*)$, hence, $f(\mathbf{z}^*) - f(\mathbf{z}_i^{(0)}) \leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*)$ with probability $1-\zeta$. With $\mathbf{z}_i^*$ as the local maxima found by CASMOPOLITAN at the i-th restart. As $f(\mathbf{z}_i^{(0)}) \leq f(\mathbf{z}_i^*)$, therefore,

$$f(\mathbf{z}^*) - f(\mathbf{z}_i^*) \leq 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*).$$

This results, with probability $1-\zeta$,

$$R_I = \sum_{i=1}^{I}(f(\mathbf{z}^*) - f(\mathbf{z}_i^*)) \leq \sum_{i=1}^{I} 2\sqrt{\beta_i}\sigma_{gl}(\mathbf{z}_i^{(0)}; D_{i-1}^*).$$

Finally, using Lemmas 5.3 and 5.4 in Srinivas et al. (2010), we can bound R_I as $R_I \leq \sqrt{IC_1\beta_I\gamma(I;k,[\mathcal{H},\mathcal{X}])}$ with $C_1 = 8/\log(1+\sigma^{-2})$ and $\gamma(I;k,[\mathcal{H},\mathcal{X}])$ is the maximum information gain for the mixed kernel derived in Theorem 3.1. □

Discussion It is worth emphasizing that the assumption of the existence of such a function $g_i(\mathbf{z})$ (at the i-th restart) that satisfies our requirements generally does not need to hold when $i \to \infty$. In fact, if this assumption needs to satisfy $\forall i$ when $i \to \infty$ then it will be same as the assumption in Srinivas et al. (2010). We will show that generally this assumption only needs to hold for a finite number of restarts. In particular, it is common that for the objective function f, there exists a local maxima $\tilde{\mathbf{z}}^*$ which is larger than all other local maxima and only smaller than the global maximum. Then as $\lim_{I\to\infty} R_I/I = 0$, there exists a finite number I_0 that the function value of the TR center at the I_0-th restart will be larger than $f(\tilde{\mathbf{z}}^*)$, and thus the 'local maxima' found after the I_0-th restart is actually the global maximum, and CASMOPOLITAN converges. Therefore, our assumption regarding the existence of $g_i(\mathbf{z})$ only needs to hold until the I_0-th restart. This discussion is applicable for both categorical and mixed space settings.

Table 2. Configurations of the test problems.

Objective f	Type	Inputs
Contamination (Hu et al., 2010)	real, cat, min, 25-dim	Choices on whether to use control at each stage $\mathbf{h} \in \{\text{True}, \text{False}\}^{25}$
Pest (Oh et al., 2019)	real, cat, min, 25-dim	Pesticide choice at each stage (or use no pesticide) $\mathbf{h} \in \{\text{No pesticide}, 1, 2, 3, 4\}^{25}$
DifficultPest	real, cat, min, 80-dim	Pesticide choice at each stage (or use no pesticide) $\mathbf{h} \in \{\text{No pesticide}, 1, 2, 3, 4\}^{80}$
MAXSAT	real, cat, min, 60-dim	$\mathbf{h} \in \{0, 1\}^{60}$
Func2C (Ru et al., 2020a)	synthetic, mixed, max, 4-dim	$\mathbf{x} \in [-1, 1]^2$ $h_1 = \{\text{ros}(\mathbf{x}), \text{cam}(\mathbf{x}), \text{bea}(\mathbf{x})\}$ $h_2 = \{+\text{ros}(\mathbf{x}), +\text{cam}(\mathbf{x}), +\text{bea}(\mathbf{x}), +\text{bea}(\mathbf{x}), +\text{bea}(\mathbf{x})\}$
Func3C (Ru et al., 2020a)	synthetic, mixed, max, 5-dim	$\mathbf{x} \in [-1, 1]^2$ $h_1 = \{\text{ros}(\mathbf{x}), \text{cam}(\mathbf{x}), \text{bea}(\mathbf{x})\}$ $h_2 = \{+\text{ros}(\mathbf{x}), +\text{cam}(\mathbf{x}), +\text{bea}(\mathbf{x}), +\text{bea}(\mathbf{x}), +\text{bea}(\mathbf{x})\}$ $h_3 = \{+5 \times \text{ros}(\mathbf{x}), +2 \times \text{cam}(\mathbf{x}), +2 \times \text{bea}(\mathbf{x}), +3 \times \text{bea}(\mathbf{x})\}$
XG-MNIST	real, mixed, max, 8-dim	h_1 (booster type type) $\in$ {gbtree, dart} h_2 (grow policies) $\in$ {depthwise, loss} h_3 (training objective) $\in$ {softmax, softprob} x_1 (learning rate) $\in [0, 1]$ x_2 (max depth) $\in [1, 10]$ x_3 (minimum split loss) $\in [0, 10$ x_4 (subsample) $\in [0.001, 1]$ x_5 (amount of regularisation) $\in [0, 5]$
Ackley-53 (Bliek et al., 2020)	synthetic, mixed, min, 53-dim	$\mathbf{h} \in \{0, 1\}^{50}$ $\mathbf{x} \in [-1, 1]^3$
Rosen-200	synthetic, mixed, min, 200-dim	$\mathbf{h} \in \{0, 1\}^{100}$ $\mathbf{x} \in [-2, 2]^{100}$
Black-box adversarial attack	real, mixed, max, 85-dim	Choice on the location of the pixel $h_i \in \{0, 1, ..., 13\}\ \forall i \in [1, 42], i \in \mathbb{Z}$ Upsampling technique $h_{43} \in$ {bilinear, nearest, bicubic} Amount of perturbation $\mathbf{x} \in [-1, 1]^{42}$

Note: real/synthetic: whether the problem is/simulates a real-life task or whether it is a standard benchmark function.
cat/mixed: categorical or mixed categorical-continuous problem.
max/min: maximisation or minimisation problem. We flip the sign of the objective function values where appropriate.