# Appendices

These appendices are structured as follows:

- Appendix A contains an analysis of the adaptive trapezium method. In particular we investigate casting the method into the Bayesian framework by performing an average case analysis. We present Proposition A.1 which demonstrates the assumptions that underlay the use of the adaptive trapezium method do not carry through to the Bayesian setting. In addition, we provide an average-case analysis of the number of integrand evaluations required by `AdapTrap` (in Proposition A.4 and Corollary A.5).

- Appendix B contains the `AdapBC` algorithm, the idealised version of the `E-AdapBC` algorithm that we presented in the main text where $\theta$ is marginalised rather than optimised.

- Full details for the stochastic process model used in our experiments are contained in Appendix C.

- Aspects of the implementation of all algorithms considered are addressed in Appendix D. These include details for the marginalisation of $\theta$ in `AdapBC` and for the optimisation over $\theta$ in `E-AdapBC`.

- Appendix E completes a full description of the experiments that were carried out and reported in the main text. In addition, the impact of the choice of $\phi$ and $\ell$ is empirically assessed in Appendix E.3, while the `AdapBC` and `E-AdapBC` methods are compared in Appendix E.4.

- Finally, for completeness Appendix F recalls standard mathematical definitions that are used in the arguments of Appendix A.

## A Average Cases Analysis of `AdapTrap`

The aim of this section is to discuss how one might naively attempt to create a direct Bayesian analogue of `AdapTrap`. To this end we recall the approach of Diaconis (1988), who took a classical cubature rule of the form (3) and asked "for what prior does (3) arise as the mean of the posterior marginal distribution of the integral?".[11] Thus, in the context of creating an analogue of `AdapTrap`, we can follow Diaconis and seek a prior such that the mean of the posterior marginal for the integral is `Trap` in (4). Thus we must consider stochastic processes for which the conditional mean is the piecewise linear interpolant (over the range of $x_1, \dots, x_n$) of the data $\mathcal{D}_n$ on which it is conditioned.

Let $C([a,b])$ denote the set of continuous real-valued functions on $[a,b]$ and consider the subset $F_{\rho,m,k,\tau} \subset C([a,b])$ of integrands $f^*$ for which $\texttt{AdapTrap}_{\rho,m,k}$ fails to achieve its stated error tolerance $\tau$ upon termination, or for which $\texttt{AdapTrap}_{\rho,m,k}$ fails to terminate at all (this set is non-empty; e.g. Clancy et al., 2014). From an inferential perspective, the decision to employ $\texttt{AdapTrap}_{\rho,m,k}$ can be regarded as a belief that $f^* \notin F_{\rho,m,k,\tau}$. Proposition A.1, presented next, suggests that stochastic process models giving rise to piecewise linear interpolants are incompatible with the use of $\texttt{AdapTrap}_{\rho,m,k}$, due to assigning non-zero probability mass to $F_{\rho,m,k,\tau}$ whenever $\tau > 0$. This result, whose proof is straight-forward and contained in the supplement, can be interpreted as an average-case analysis of `AdapTrap` (Ritter, 2000). Denote the error function $\mathrm{erf}(x) := \frac{1}{\sqrt{\pi}} \int_{-x}^{x} e^{-t^2} \,\mathrm{d}t$.

**Proposition A.1.** Fix $a < b$, $\rho > 0$, $m \in \mathbb{N}$ and $k$ a positive even integer. Let $f^*$ be sampled from a centred Gaussian process on $C([a,b])$, whose law is denoted $\mathbb{P}^*$, such that the conditional mean $f^*|\mathcal{D}_n$ is the piecewise linear interpolant (over the range of $x_1, \dots, x_n$) of the data $\mathcal{D}_n$ on which it is conditioned. If `AdapTrap` terminates, denote its error $\epsilon_{\rho,m,k,\tau}(f^*) := I(f^*) - \texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$, otherwise set $\epsilon_{\rho,m,k,\tau}(f^*) := \infty$. Then for every $\tau > 0$,

$$\mathbb{P}^*(|\epsilon_{\rho,m,k,\tau}| > \tau) \ > \ \mathrm{erf}(c\tau) \left[1 - \mathrm{erf}(\sqrt{3}c\tau)\right],$$

where $c > 0$ is a $\mathbb{P}^*$-dependent constant.

[11] Paraphrased. Conversely, Cor. 2.10 of Karvonen et al. (2018) showed that *all* non-adaptive cubature rules of the form (3) arise as the posterior mean of some stochastic process model.

Though the probability mass assigned to $F_{\rho,m,k,\tau}$ can be made small, the fact that it is non-zero for all $\tau > 0$ calls into doubt whether direct Bayesian analogues of classical adaptive methods can exist, in contrast to the situation for non-adaptive methods (Karvonen et al., 2018). In Appendix A.2, further average-case analysis is provided, showing that for mis-specified $\rho$ the expected number of steps of `AdapTrap` can be unbounded. Taken together, our analyses suggest that classical adaptive methods cannot be directly replicated in BC and a different strategy is needed. In Section 3 of the main text we put forward a *de novo* BC method, which achieves adaptivity through a flexible non-stationary stochastic process model.

In Appendix A.1 we introduce our notation, then in Appendix A.2 we provide a detailed average-case analysis of the expected number of evaluations of the integrand required by the `AdapTrap` method. Finally, in Appendix A.3 we prove Proposition A.1. The arguments that we present in this appendix exploit definitions and basic results about full $k$-ary trees. For completeness, the required background knowledge is set out in Appendix F.

### A.1 Notation and Set-Up

In what follows we let $C([a,b])$ denote the set of continuous functions $g : [a,b] \to \mathbb{R}$. The set $C([a,b])$ can be endowed with the structure of a measurable space using the Borel $\sigma$-field generated from the topology induced by the supremum norm $\|g\|_\infty := \sup_{a \leq x \leq b} |g(x)|$. The stochastic processes considered in this work are all Gaussian measures on the measurable space $C([a,b])$; we refer the reader to Bogachev (1998) for full mathematical background.

In the main text we followed the usual convention in numerical analysis that the error of a quadrature method $Q_n(\cdot)$ is defined as $\epsilon = |I(f^*) - Q_n(f^*)|$, i.e. as the absolute value of the difference between the quadrature rule and the true integral. However, when it comes to performing an average-case analysis, it is more natural (and convenient) to consider the signed error instead. Therefore we now re-instantiate our notation as per the statement of Proposition A.1, namely we use the signed error $\epsilon := I(f^*) - Q_n(f^*)$ in the sequel.

Following the discussion of Appendix A, we are interested in Gaussian measures on $C([a,b])$ whose conditional mean function $f|\mathcal{D}_n$ is the piecewise linear interpolant (in the range of $x_1, \dots, x_n$) of the data $\mathcal{D}_n$. Diaconis (1988) noted that the only non-trivial Gaussian measures with this property are based on the covariance function $k(x,y) = \lambda \min(x,y) + \gamma$, where $\gamma > -a$ controls the initial starting point of the process and $\lambda > 0$ is the amplitude parameter with mean function $m(x) = 0$. In other words, the only processes satisfying the preconditions of Proposition A.1 are shifted and scaled Wiener processes. In the following we therefore consider an integrand $f^*$ that is drawn at random from the Gaussian process on $C([a,b])$ with mean $m(x) = 0$ and covariance $k(x,y) = \lambda \min(x,y) + \gamma$ where $\lambda > 0$ and $\gamma > -a$. The law of this process will be denoted $\mathbb{P}^*$ and we use $\mathbb{E}^*$, $\mathbb{V}^*$ and $\mathbb{C}^*$ to denote expectation, variance and covariance with respect to $\mathbb{P}^*$.

Recall that the algorithm `AdapTrap` was presented as Algorithm 1 in the main text. Note that if we want to ensure we use previous evaluations of $f^*$ at each level of recursion then we only require that $m$ is an integer multiple of $k$. This allows computational speed up by memoising the previous iteration's function calls. A termination of $\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$ can be represented as a full $k$-ary tree.

In what follows let $\mathcal{T}^k$ be the set of full $k$-ary trees. Full background is provided in Appendix F but for illustration we provide an example of a full 3-ary tree:

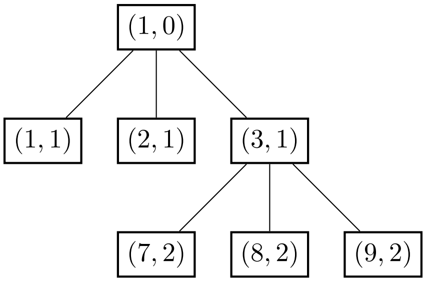


Figure 4: Example of a full 3-ary tree, with levels 0,1,2. Level 1 has the maximum of 3 nodes, while level 2 has 3 of a maximum 9 nodes present.

A full $k$-ary tree $T$ is characterised by its nodes, and the $p$th possible node at depth $q$ will be represented as the vector $(p, q)$; c.f. Appendix F. The points $x_i$ at which $f^*$ is evaluated in $\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$ can be represented as the nodes of a full $k$-ary tree and we denote this tree by $A_{\rho,m,k,\tau}(f^*)$. That is, each node $(p, q)$ in $A_{\rho,m,k,\tau}(f^*)$ corresponds to a recursive step in the running of $\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$, namely the step

$$\texttt{AdapTrap}_{\rho,m,k}\left(f^*, a + \frac{(b-a)(p-1)}{k^q}, a + \frac{(b-a)p}{k^q}, \tau\rho^q\right). \tag{10}$$

Formally, the full $k$-ary tree representation defines a map $A_{\rho,m,k,\tau} : C([a, b]) \to \mathcal{T}^k$ and, with $C([a, b])$ endowed with the measure $\mathbb{P}^*$, then $A_{\rho,m,k,\tau}$ can be considered as a random variable on $\mathcal{T}^k$. Our aim in the remainder is to study the random variable $A_{\rho,m,k,\tau}$ and, in doing so, we shall establish Proposition A.1 from the main text.

The notation $\tilde{\epsilon}^{(p,q)} = Q_2^{(p,q)} - Q_1^{(p,q)}$ will be used to denote the local error estimate computed in the recursive step in (10), corresponding to node $(p, q)$ of $A_{\rho,m,k,\tau}(f^*)$. From the definition we have that, letting $X_{Q_1}^{(p,q)} := \{x_i\}_{i=1}^{m+1}$ denote the set of ordered $(x_1 < \ldots < x_{m+1})$ abscissae used in the calculation of $Q_1^{(p,q)}$ and $x_{\text{mid}}^{(i)} := \frac{x_i + x_{i+1}}{2}$,

$$\begin{aligned}\tilde{\epsilon}^{(p,q)} &= \frac{b-a}{2mk^q}\sum_{i=1}^{m} f^*(x_{\text{mid}}^{(i)}) - \frac{f^*(x_i) + f^*(x_{i+1})}{2} \qquad (11)\\ &= \frac{b-a}{4mk^q}\sum_{i=1}^{m}\left[\left(f^*(x_{\text{mid}}^{(i)}) - f^*(x_i)\right) - \left(f^*(x_{i+1}) - f^*(x_{\text{mid}}^{(i)})\right)\right].\end{aligned}$$

Thus, with $f^* \sim \mathbb{P}^*$, $\tilde{\epsilon}^{(p,q)}$ is a random variable on $\mathbb{R}$. By the independent increment property of the Wiener process, each of the random variables $f^*(x_{\text{mid}}^{(i)}) - f^*(x_i)$ and $f^*(x_{i+1}) - f^*(x_{\text{mid}}^{(i)})$ are independent. By the Gaussian increment property of the Wiener process, $f^*(x_{\text{mid}}^{(i)}) - f^*(x_i) \overset{d}{=} f^*(x_{i+1}) - f^*(x_{\text{mid}}^{(i)}) \sim \mathcal{N}\left(0, \frac{\lambda(b-a)}{2mk^q}\right)$, where $\overset{d}{=}$ is equality in distribution. Thus,

$$\tilde{\epsilon}^{(p,q)} \sim \frac{b-a}{4mk^q}\mathcal{N}\left(0, \frac{\lambda(b-a)}{k^q}\right) = \mathcal{N}\left(0, \frac{\lambda(b-a)^3}{(4m)^2k^{3q}}\right). \tag{12}$$

Before addressing Proposition A.1, we will establish results on the expected number of steps of $\texttt{AdapTrap}$ next.

### A.2 Expected number of steps of AdapTrap

The first of these intermediate results is an elementary property of the local error random variables $\tilde{\epsilon}^{(p,q)}$. To present this, let $I^{(p,q)}$ be the closed interval over which $\tilde{\epsilon}^{(p,q)}$ is computed and recall that $X_{Q_1}^{(p,q)}$ is the set of ordered abscissae used in the computation of $Q_1^{(p,q)}$; for instance, with $m = 1$ and $a = 0, b = 1$ the tree in Figure 4 has $I^{(3,1)} = [2/3, 1]$ and $X_{Q_1}^{(3,1)} = \{2/3, 1\}$. Then we have the following independence result:

**Lemma A.2.** Under $\mathbb{P}^*$, the local error estimate random variable $\tilde{\epsilon}^{(p,q)}$ is independent of the random variable $f^*(x)$ for all $x \in (\mathbb{R} \setminus I^{(p,q)}) \cup X_{Q_1}^{(p,q)}$.

*Proof.* From joint Gaussianity of the random variables, it is sufficient to show that $\mathbb{C}^*\left(\tilde{\epsilon}^{(p,q)}, f^*(x)\right) = 0$ for any $x \in \mathbb{R} \setminus I^{(p,q)}$ or $x \in X_{Q_1}^{(p,q)}$. In fact, since $b - a > 0$, from bilinearity of $\mathbb{C}^*(\cdot, \cdot)$ it is sufficient to consider $\mathbb{C}^*\left(\bar{\epsilon}^{(p,q)}, f^*(x)\right) = 0$ where $\bar{\epsilon}^{(p,q)} := \frac{4mk^q}{b-a}\tilde{\epsilon}^{(p,q)}$. Note that $I^{(p,q)} = [\alpha, \beta]$ for some $\alpha < \beta$. Consider the three cases in turn, using (11):

$$\begin{aligned}
\text{For } x < \alpha\text{:}\quad \mathbb{C}^*\left(\overline{\epsilon}^{(p,q)}, f(x)\right) &= \sum_{i=1}^{m} 2\mathbb{C}^*[f^*(x_{\text{mid}}^{(i)}), f^*(x)] - \mathbb{C}^*[f(x_i), f(x)] - \mathbb{C}^*[f(x_{i+1}), f(x)] \\
&= \sum_{i=1}^{m} 2(\lambda x + \gamma) - (\lambda x + \gamma) - (\lambda x + \gamma) \;=\; 0. \\
\text{For } x > \beta\text{:}\quad \mathbb{C}^*\left(\overline{\epsilon}^{(p,q)}, f(x)\right) &= \sum_{i=1}^{m} 2\mathbb{C}^*[f^*(x_{\text{mid}}^{(i)}), f^*(x)] - \mathbb{C}^*[f^*(x_i), f^*(x)] - \mathbb{C}^*[f^*(x_{i+1}), f^*(x)] \\
&= \sum_{i=1}^{m} [\lambda(x_{i+1} + x_i) + 2\gamma] - (\lambda x_i + \gamma) - (\lambda x_{i+1} + \gamma) \;=\; 0. \\
\text{For } x = x_j \in X_{Q_1}^{(p,q)}\text{:}\quad \mathbb{C}^*\left(\overline{\epsilon}^{(p,q)}, f(x)\right) &= \sum_{i=1}^{j-1} \mathbb{C}^*[2f^*(x_{\text{mid}}^{(i)}) - f^*(x_i) - f^*(x_{i+1}), f^*(x_j)] \\
&\qquad + \sum_{i=j}^{m} \mathbb{C}^*[2f^*(x_{\text{mid}}^{(i)}) - f^*(x_i) - f^*(x_{i+1}), f^*(x_j)] \\
&= \sum_{i=1}^{j-1} \lambda(x_{i+1} + x_i) + 2\gamma - (\lambda x_i + \gamma) - (\lambda x_{i+1} + \gamma) \\
&\qquad + \sum_{i=j}^{m} 2(\lambda x_j + \gamma) - (\lambda x_j + \gamma) - (\lambda x_j + \gamma) \;=\; 0.
\end{aligned}$$

This completes the proof. □

Our next intermediate result concerns the probability of obtaining any given full $k$-ary tree $T$ as the value of the random variable $A_{\rho,m,k,\tau}(f^*)$:

**Proposition A.3** (Probability of $A_{\rho,m,k,\tau} = T$)**.** Let $k$ be an even positive integer and let $T \in \mathcal{T}^k$ be finite. Denote by $D$, $L_i$ and $V_i$ the height of $T$, the number of leaves of $T$ at depth $i$ and the number of inner nodes of $T$ at depth $i$, respectively (recall that these definitions are reserved for Appendix F). Then

$$\mathbb{P}^*(A_{\rho,m,k,\tau} = T) = \prod_{i=0}^{D} \alpha_i^{L_i} (1 - \alpha_i)^{V_i},$$

where $\alpha_i = \mathrm{Prob}_{Z \sim \mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau(k^{3/2}\rho)^i}{\sqrt{\lambda(b-a)^3}}\right)$.

*Proof.* Let $T \in \mathcal{T}^k$ be finite, so that we seek to compute

$$\mathbb{P}^*(A_{\rho,m,k,\tau} = T) = \int \mathbb{1}[f^* \in A_{\rho,m,k,\tau}^{-1}(T)] \, \mathrm{d}\mathbb{P}^*(f^*).$$

Note that from a given full $k$-ary tree $T$ we know the local error tolerance intervals $I_\tau^{(p,q)} := [-\tau\rho^q, \tau\rho^q]$ and whether the local error estimates $\tilde{\epsilon}^{(p,q)} \in I_\tau^{(p,q)}$ or $\tilde{\epsilon}^{(p,q)} \notin I_\tau^{(p,q)}$ for each node $(p,q) \in T$. That is, if $(p,q)$ is a leaf then $\tilde{\epsilon}^{(p,q)} \in I_\tau^{(p,q)}$ and if $(p,q)$ is an inner node then $\tilde{\epsilon}^{(p,q)} \notin I_\tau^{(p,q)}$. Define

$$S^{(p,q)} := \begin{cases} I_\tau^{(p,q)}, & \text{if } (p,q) \text{ is a leaf in } T, \\ \mathbb{R} \setminus I_\tau^{(p,q)}, & \text{if } (p,q) \text{ is an inner node in } T. \end{cases}$$

Further, let $\langle (p_i, q_i) \rangle_{i=1}^{N}$ be the preorder traversal of $T$ (see Definition F.4 in Appendix F). For notational convenience in the following we will denote $v_i := (p_i, q_i)$, $\tilde{\epsilon}_i := \tilde{\epsilon}^{v_i}$ and $S_i := S^{v_i}$. Thus, returning to our original problem, we have

$$\begin{aligned}\mathbb{P}^*(A_{\rho,m,k,\tau} = T) &= \int \prod_{(p,q)\in T} \mathbb{1}[\tilde{\epsilon}^{(p,q)} \in S^{(p,q)}]\,\mathrm{d}\mathbb{P}^*(f^*) \\ &= \int_{S_N} \cdots \int_{S_1} p(\tilde{\epsilon}_1, \dots, \tilde{\epsilon}_N)\,\mathrm{d}\tilde{\epsilon}_1 \dots \mathrm{d}\tilde{\epsilon}_N,\end{aligned}$$

where $p(\tilde{\epsilon}_1, \dots, \tilde{\epsilon}_N)$ is the joint density function of $\tilde{\epsilon}_1, \dots, \tilde{\epsilon}_N$.

Now, motivated by the factorisation

$$p(\tilde{\epsilon}_1, \dots, \tilde{\epsilon}_N) = \prod_{i=0}^{N-1} p(\tilde{\epsilon}_{N-i} \,|\, \tilde{\epsilon}_{N-i-1}, \dots, \tilde{\epsilon}_1),$$

we make the following claim (whose proof is provided immediately after the present proof):

**Claim:** For $i \in \{0, \dots, N-1\}$ and $k$ an even positive integer we have $p(\tilde{\epsilon}_{N-i} \,|\, \tilde{\epsilon}_{N-i-1}, \dots, \tilde{\epsilon}_1) = p(\tilde{\epsilon}_{N-i})$.

Using the claim, we have that

$$\mathbb{P}^*(A_{\rho,m,k,\tau} = T) = \int_{S_N} \cdots \int_{S_1} \prod_{i=1}^{N} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_1 \dots \mathrm{d}\tilde{\epsilon}_N \;=\; \prod_{i=1}^{N} \int_{S_i} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_i.$$

Recalling that $n_i$ is the depth of node $v_i$, the integrals in the final product can be expressed as

$$\int_{S_i} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_i = \begin{cases} \int_{-\tau\rho^{n_i}}^{\tau\rho^{n_i}} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_i, & \text{if } v_i \text{ is a leaf in } T, \\ 1 - \int_{-\tau\rho^{n_i}}^{\tau\rho^{n_i}} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_i, & \text{if } v_i \text{ is a inner node in } T. \end{cases}$$

By (12) we have,

$$\int_{-\tau\rho^{n_i}}^{\tau\rho^{n_i}} p(\tilde{\epsilon}_i)\,\mathrm{d}\tilde{\epsilon}_i = \int_{-L}^{L} p_Z(z)\,\mathrm{d}z,$$

where $p(z)$ is the density function of $Z \sim \mathcal{N}(0,1)$ and $L = \frac{4m\tau(k^{3/2}\rho)^{n_i}}{\sqrt{\lambda(b-a)^3}}$. Letting $\bar{\alpha}_i = \mathrm{Prob}_{Z\sim\mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau(k^{3/2}\rho)^{n_i}}{\sqrt{\lambda(b-a)^3}}\right)$ we have,

$$\mathbb{P}^*(A_{\rho,m,k,\tau} = T) = \prod_{i=1}^{N} \bar{\alpha}_i^{\mathbb{1}(v_i \in L(T))}(1-\bar{\alpha}_i)^{1-\mathbb{1}(v_i \in L(T))},$$

where $L(T)$ is the set of leaves in $T$. By noting that $\bar{\alpha}_i$ only depends on the depth $n_i$ of each node $v_i$, we can rearrange this product by multiplying by depth instead of by the preorder traversal. Thus,

$$\mathbb{P}^*(A_{\rho,m,k,\tau} = T) = \prod_{i=0}^{D} \alpha_i^{L_i}(1-\alpha_i)^{V_i},$$

where $\alpha_i = \mathrm{Prob}_{Z\sim\mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau(k^{3/2}\rho)^{i}}{\sqrt{\lambda(b-a)^3}}\right)$, $L_i$ and $V_i$ are the number of leaves and inner nodes at depth $i$ respectively and $D$ is the height of $T$. □

The claim used in the above proof is established as follows:

*Proof of Claim.* Let $X_{\tilde{\epsilon}}^{i}$ be the abscissae used in the computation of $\tilde{\epsilon}_i$ and let $N_i = \{v_1, \dots, v_{i-1}\}$. By Lemma A.2 and bilinearity of $\mathbb{C}^*(\cdot,\cdot)$, if $X_{\tilde{\epsilon}}^{i} \cap X_{\tilde{\epsilon}}^{j} = \emptyset$ for $i \neq j$, then $\tilde{\epsilon}_i$ is independent of $\tilde{\epsilon}_j$. This immediately implies that $\tilde{\epsilon}_i$

is conditionally independent of all $\tilde{\epsilon}_{N_i \setminus \mathrm{asc}(v_i)}$ given $\tilde{\epsilon}_{\mathrm{asc}(v_i)}$, where $\mathrm{asc}(v_i)$ are the nodes in $T$ that are ascendants of $v_i$. Thus we are left to prove that $\tilde{\epsilon}_i$ is independent of any $\tilde{\epsilon}_j$ with $j \in \mathrm{asc}(v_i)$.

Let $p$ be the parent node of $v_i$ and let $d$ be the depth of $v_i$. We will prove that $X^i_{\tilde{\epsilon}} \cap X^p_{\tilde{\epsilon}} \subseteq X^{v_i}_{Q_1}$ and by induction and the application of Lemma A.2 the result will be established.

Note that $X^i_{\tilde{\epsilon}}$ is an affine transformation of $X^p_{\tilde{\epsilon}} = \{x + \frac{i}{2k^{d-1}m}\}_{i=0}^{2m}$, where $x$ is the left-hand end point of the subinterval, of the form[12] $X^i_{\tilde{\epsilon}} = \frac{1}{k}(X^p_{\tilde{\epsilon}} - x) + x + nk^{-d}$ for some $n = 0, \dots, k-1$. Furthermore, both $X^i_{\tilde{\epsilon}}$ and $X^p_{\tilde{\epsilon}}$ are affine transformations of the set $\{\frac{i}{2m}\}_{i=0}^{2m}$. Thus it is enough to prove that $\{\frac{i}{2m}\}_{i=0}^{2m} \cap \{\frac{i}{2km}\}_{i=0}^{2m} \subseteq \{\frac{i}{km}\}_{i=0}^{m} = \{\frac{2i}{2km}\}_{i=0}^{m}$.

Let $2m = ak + b$ where $0 \leq b < k$. Then $\{\frac{ik}{2km}\}_{i=0}^{2m} \cap \{\frac{i}{2km}\}_{i=0}^{2m} = \{\frac{ik}{2km}\}_{i=0}^{a}$ and so, if $k$ is even then $\{k, 2k, \dots, ak\} \subseteq \{2i\}_{i=0}^{m}$ and we are finished. By the definition of a preorder traversal we have, for each $i$, $\mathrm{asc}(v_i) \subseteq N_i$ and further $N_i \cap \mathrm{desc}(v_i) = \emptyset$, where $\mathrm{desc}(v_i)$ are the descendants of $v_i$. Thus we have $p(\tilde{\epsilon}_{N-i} \mid \tilde{\epsilon}_{N-i-1}, \dots, \tilde{\epsilon}_1) = p(\tilde{\epsilon}_{N-i})$ as required. □

The main result in this section shows that there are settings (albeit not the standard setting of $\rho = k^{-1}$) for `AdapTrap` for which the expected number of steps is unbounded:

**Proposition A.4** (Expected number of steps of `AdapTrap`)**.** Let $a < b$ and let $f^*$ be drawn at random from any centred Gaussian process on $D = [a, b]$ whose whose conditional mean function $f | \mathcal{D}_n$ is the piecewise linear interpolant (in the range of $x_1, \dots, x_n$) of the data $\mathcal{D}_n$. Let $\mathbb{E}^*$ denote expectation with respect to this random integrand. Let $N_{\rho,m,k}(f^*, \tau)$ be the total number of integrand evaluations incurred in the running of $\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$. Then for every $k \in \mathbb{N}$ and $k$ a positive even integer, there exists $C > 0$ such that for every $\tau \leq C$ and any $\rho \leq k^{-3/2}$ we have $\mathbb{E}^*[N_{\rho,m,k}(f^*, \tau)] = \infty$.

*Proof.* Let $V_n$ be the number of inner nodes of $T$ at depth $n$. Then, by Proposition A.3,

$$\mathbb{E}^*[V_n \mid V_{n-1}] = k(1 - \alpha_{n-1})V_{n-1}.$$

By the law of total expectation, induction and noting that $\mathbb{E}^*[V_0] = 1 - \alpha_0$, we have

$$\mathbb{E}^*[V_n] \; = \; \mathbb{E}^*[\mathbb{E}^*[V_n \mid V_{n-1}]] \; = \; k(1 - \alpha_{n-1})\mathbb{E}^*[V_{n-1}] \; = \; \prod_{i=0}^{n-1} k(1 - \alpha_i).$$

Note that if we have $\mathbb{E}^*[V_n] \nrightarrow 0$ as $n \to \infty$ then this implies $\mathbb{E}^*[N_{\rho,m,k}(f^*, \tau)] = \infty$. Thus studying the convergence properties of the infinite product $\prod_{i=0}^{\infty} k(1 - \alpha_i)$ with varying $\rho, k, \tau$ and $\lambda$ is sufficient to prove the result.

For $\rho = \frac{1}{k^{3/2}}$ we have $\alpha_i = \alpha \coloneqq \mathrm{Prob}_{Z \sim \mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau}{\sqrt{\lambda(b-a)^3}}\right)$. Then the product simplifies to $\mathbb{E}^*[V_n] = k^n(1 - \alpha)^n$. This implies that if $\tau$ and $\lambda$ are selected such that $\alpha \leq \frac{k-1}{k}$, then $\mathbb{E}^*[X_n] \nrightarrow 0$. This is the case if and only if

$$\tau \leq -\frac{\Phi^{-1}\left(\frac{1}{2m}\right)\sqrt{\lambda(b-a)^3}}{4m}. \tag{13}$$

Note further that for $\rho \leq \frac{1}{k^{3/2}}$ we have $\alpha_i \leq \alpha$. Thus, for $\rho \leq \frac{1}{k^{3/2}}$ we have,

$$k(1 - \alpha_i) \geq k(1 - \alpha) \implies \prod_{i=0}^{n-1} k(1 - \alpha_i) \geq k^n(1 - \alpha)^n.$$

So, if $\tau$ satisfies (13) then for any $\rho \leq \frac{1}{k^{3/2}}$ we have $\mathbb{E}[V_n] \nrightarrow 0$. This completes the proof. □

Our final contribution is to provide a closed form for the probability of non-termination in the case $k = 2$:

[12] Here we are using the standard notation $aX + b \coloneqq \{ax + b \mid x \in X\}$.

**Corollary A.5** (Probability of non-termination for $k = 2$)**.** Let $T_\infty$ be the set of full $k$-ary trees with infinite depth. For $k = 2$, $\rho = \frac{1}{k^{3/2}}$ and $\tau$ satisfying (13) then the probability of non-termination is

$$\mathbb{P}^*(A_{\rho,m,k,\tau} \in T_\infty) = \frac{1-2\alpha}{1-\alpha},$$

where $\alpha = \mathrm{Prob}_{Z\sim\mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau}{\sqrt{\lambda(b-a)^3}}\right)$. Further, for $\rho < \frac{1}{k^{3/2}}$ we have

$$\mathbb{P}^*(A_{\rho,m,k,\tau} \in T_\infty) > \frac{1-2\alpha}{1-\alpha}, \tag{14}$$

*Proof.* Assume that $\rho = \frac{1}{k^{3/2}}$ and that $\tau$ satisfies (13). The probability of an outcome being a full $k$-ary tree with $kn+1$ (for $n \in \mathbb{N}_0$) nodes is

$$\mathbb{P}^*(|A_{\rho,m,k,\tau}| = kn+1) = C_n^{(k)}\alpha^{n(k-1)+1}(1-\alpha)^n,$$

where $C_n^{(k)} = \frac{1}{(k-1)n+1}\binom{nk}{n}$ is the number of $k$-ary trees with $n$ nodes (see Theorem F.2). Define the probability of termination function

$$P_k(\alpha) = \sum_{i=0}^{\infty} \mathbb{P}^*(|A_{\rho,m,k,\tau}| = ki+1).$$

Recall the generating function of the standard Catalan numbers (18),

$$C_2(x) := \sum_{i=0}^{\infty} C_i^{(2)} x^i = \frac{1-\sqrt{1-4x}}{2x}.$$

Thus, by noting that

$$P_2(\alpha) = \alpha \sum_{i=0}^{\infty} C_i^{(2)}[\alpha(1-\alpha)]^i,$$

we have

$$\begin{aligned} P_2(\alpha) &= \alpha C_2(\alpha(1-\alpha)) \\ &= \alpha\frac{1-\sqrt{1-4\alpha(1-\alpha)}}{2\alpha(1-\alpha)} = \frac{1-\sqrt{(2\alpha-1)^2}}{2(1-\alpha)} = \frac{1-|2\alpha-1|}{2(1-\alpha)} = \begin{cases} \frac{\alpha}{1-\alpha}, & \text{for } \alpha \in [0, 0.5), \\ 1, & \text{for } \alpha \in [0.5, 1]. \end{cases} \end{aligned}$$

The inequality (14) can be derived by noting that for $\rho < \frac{1}{k^{3/2}}$ and for every $i$ we have $\alpha_i < \alpha$. Thus,

$$\mathbb{P}^*(|A_{\rho,m,2,\tau}| = 2n+1) > C_n^{(2)}\alpha^{n+1}(1-\alpha)^n,$$

since $g(x) = x^n(1-x)^{n-1}$ is monotonically increasing[13] for $0 < x < \frac{n}{2n-1}$. □

As a final remark, note that using the same approach we can show that, for $k \geq 5$, the probability of termination $1 - P_k(\alpha)$ does not have a closed form. Note that

$$P_k(\alpha) = \alpha C_k(\alpha(1-\alpha)),$$

where $C_k$ is the generating function of the $k$-Catalan numbers. $C_k$ obeys the following functional equation

$$C_k(x) = 1 + x[C_k(x)]^k.$$

Thus expressing $C_k$ as a function of $x$ in closed form is equivalent to solving a degree $k$ trinomial. This has no algebraic solution for $k \geq 5$ with general $x$ and so one cannot express $P_k(\alpha)$ in closed form for $k \geq 5$.

[13] This can be shown by noting $g'(x) = nx^{n-1}(1-x)^{n-1} - (n-1)x^n(1-x)^{n-2} = x^{n-1}(1-x)^{n-2}(n + x(1-2n))$.

### A.3 Proof of Proposition A.1

This section contains the proof of Proposition A.1 from the main text. Recall that we aim to perform an average-case analysis of the `AdapTrap` method which is simply the composite trapezoidal rule on a non-uniform grid of abscissae under the aforementioned prior measure $\mathbb{P}^*$. We have,

$$I(f^*) \coloneqq \int_a^b f^*(x)\,\mathrm{d}x \approx \texttt{Trap}(f^*, a, b, X) \coloneqq \frac{1}{2}\sum_{i=1}^{n}[f^*(x_{i+1}) + f^*(x_i)][x_{i+1} - x_i]$$

with $X = \{x_i\}_{i=1}^{n+1}$ a given set of $n+1$ ordered abscissae such that $a = x_1 < \ldots < x_n = b$. Then the error of the trapezoidal rule is $\epsilon_X^{\texttt{Trap}}(f^*) = I(f^*) - \texttt{Trap}(f^*, a, b, X)$. Under $\mathbb{P}^*$, the error $\epsilon_X^{\texttt{Trap}}$ can now be considered a random variable. Let $f^*(X) = (f^*(x_i))_{i=1}^{n+1}$.

Recall that, due to Diaconis (1988), the mean of $f^*|\mathcal{D}_n$ is the piecewise linear interpolant of the data $\mathcal{D}_n$. Thus, by (7), $\mathbb{E}^*[I(f^*) \mid f^*(X)] = \texttt{Trap}(f^*, a, b, X)$ and by Gaussianity of $\mathbb{P}^*$ we have $\epsilon_X^{\texttt{Trap}} \mid f^*(X) \sim \mathcal{N}(0, \sigma^2)$ for some $\sigma^2 > 0$. By (8) note that under $\mathbb{P}^*$, $\sigma^2$ is only dependent on the set of abscissae $X$. Before directly proving Proposition A.1 we derive the variance of $\epsilon_X^{\texttt{Trap}} \mid f^*(X)$. In the following we use the notation $f^*_{\mathcal{D}} \coloneqq f^* \mid f^*(X)$ and $f^*_{\mathcal{D}} \sim \mathcal{GP}(m_{\mathcal{D}}, k_{\mathcal{D}})$.

**Proposition A.6.** We have

$$\epsilon_X^{\texttt{Trap}} \mid f^*(X) \sim \mathcal{N}\left(0, \sum_{i=1}^{n} \sigma_i^2\right),$$

where $\sigma_i^2 = \frac{\lambda}{12}(x_{i+1} - x_i)^3$ and with $X = \{x_i\}_{i=1}^{n+1}$ a given set of $n+1$ ordered abscissae such that $a = x_1 < \ldots < x_n = b$.

*Proof.* Define $\epsilon_i \coloneqq \int_{x_i}^{x_{i+1}} f^*_{\mathcal{D}}(x)\,\mathrm{d}x - \frac{1}{2}[f^*_{\mathcal{D}}(x_{i+1}) + f^*_{\mathcal{D}}(x_i)][x_{i+1} - x_i]$. Then $\epsilon_X^{\texttt{Trap}} \mid f^*(X) = \sum_{i=1}^{n} \epsilon_i$. Note that by the Markov property of $f^*$ for $x \in [x_i, x_{i+1}]$, $f^*_{\mathcal{D}}(x) \overset{D}{=} f^*(x) \mid f^*(x_{i+1}), f^*(x_i)$ and so $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$, where $\sigma_i^2 = \mathbb{V}^*(\epsilon_i)$.

For $x, y \in [x_i, x_{i+1}]$ we have $f^*(x) \mid f^*(x_{i+1}), f^*(x_i) \sim \mathcal{GP}(m_i(x), k_i(x,y))$ where $m_i(x)$ is the linear interpolant between $(x_i, f^*(x_i))$ and $(x_{i+1}, f^*(x_{i+1}))$ and, for $x < y$,

$$\begin{aligned}
k_i(x,y) &= k(x,y) - [k(x_i, x), k(x_{i+1}, x)] \begin{pmatrix} k(x_i, x_i) & k(x_i, x_{i+1}) \\ k(x_i, x_{i+1}) & k(x_{i+1}, x_{i+1}) \end{pmatrix}^{-1} [k(x_i, y), k(x_{i+1}, y)]^\top \\
&= \lambda x + \gamma - \frac{1}{\lambda(x_{i+1} - x_i)}\left[(\lambda x_{i+1} - \lambda x)(\lambda x_i + \gamma) + (\lambda x - \lambda x_i)(\lambda y + \gamma)\right] \\
&= \lambda x - \frac{\lambda}{x_{i+1} - x_i}\left[(x_{i+1} - x)x_i + (x - x_i)y\right] \\
&= \lambda \frac{x x_{i+1} - x x_i - x_i x_{i+1} + x x_i - xy + x_i y}{x_{i+1} - x_i} \;=\; \lambda \frac{(x_{i+1} - x)(y - x_i)}{x_{i+1} - x_i}.
\end{aligned}$$

Thus we have,

$$\begin{aligned}
\sigma_i^2 &= \mathbb{V}^*(\epsilon_i) \\
&= \int_{x_i}^{x_{i+1}} \int_{x_i}^{x_{i+1}} k_i(x,y)\,\mathrm{d}x\,\mathrm{d}y \\
&= \int_{x_i}^{x_{i+1}} \int_{x_i}^{x} \lambda \frac{(x_{i+1} - x)(y - x_i)}{x_{i+1} - x_i}\,\mathrm{d}y\,\mathrm{d}x + \int_{x_i}^{x_{i+1}} \int_{x_i}^{y} \lambda \frac{(x_{i+1} - y)(x - x_i)}{x_{i+1} - x_i}\,\mathrm{d}x\,\mathrm{d}y \\
&= 2\lambda \int_{x_i}^{x_{i+1}} \int_{x_i}^{x} \frac{(x_{i+1} - x)(y - x_i)}{x_{i+1} - x_i}\,\mathrm{d}y\,\mathrm{d}x \\
&= \frac{\lambda}{x_{i+1} - x_i} \int_{x_i}^{x_{i+1}} -x^3 + x^2(x_{i+1} + 2x_i) + x(-2x_{i+1}x_i - x_i^2) + x_{i+1}x_i^2\,\mathrm{d}x \\
&= \frac{\lambda}{12(x_{i+1} - x_i)}[x_{i+1}^4 - 4x_{i+1}^3 x_i + 6x_{i+1}^2 x_i^2 - 4x_{i+1}x_i^3 + x_i^4] \;=\; \frac{\lambda(x_{i+1} - x_i)^3}{12}.
\end{aligned}$$

The final part to prove is that for $i \neq j$ we have $\mathbb{C}^*(\epsilon_i, \epsilon_j) = 0$. Since $\mathbb{E}^*[\epsilon_i] = \mathbb{E}^*[\epsilon_j] = 0$, we have

$$\begin{aligned}\mathbb{C}^*(\epsilon_i, \epsilon_j) &= \mathbb{E}^*[\epsilon_i \epsilon_j] \\ &= \mathbb{E}^*\left[\int_{x_i}^{x_{i+1}} f^*_{\mathcal{D}}(x) - m_i(x)\,\mathrm{d}x \int_{x_j}^{x_{j+1}} f^*_{\mathcal{D}}(x) - m_j(x)\,\mathrm{d}x\right] \\ &= \mathbb{E}^*\left[\int_{x_j}^{x_{j+1}} \int_{x_i}^{x_{i+1}} [f^*_{\mathcal{D}}(x) - m_i(x)][f^*_{\mathcal{D}}(y) - m_j(y)]\,\mathrm{d}x\,\mathrm{d}y\right].\end{aligned}$$

By Fubini's theorem we can interchange the expectation and the integral. We obtain,

$$\mathbb{C}^*(\epsilon_i, \epsilon_j) = \int_{x_j}^{x_{j+1}} \int_{x_i}^{x_{i+1}} k_{\mathcal{D}}(x, y)\,\mathrm{d}x\,\mathrm{d}y.$$

By the Markov property of the Wiener process we have $k_{\mathcal{D}}(x, y)$ for $x \in [x_i, x_{i+1}]$ and $y \in [x_j, x_{j+1}]$. Thus the $\epsilon_i$ are independent and our results follows. □

Recall that we defined the error distribution at termination of the `AdapTrap` algorithm as $\epsilon_{\rho,m,k,\tau}(f^*) := I(f^*) - \texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$. From now on we will denote the error of `AdapTrap` as $\epsilon := \epsilon_{\rho,m,k,\tau}$. Thus, letting $X = \{x_i\}_{i=1}^M$ be the set of $M$ ordered abscissae used in the computation of $\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau)$ and $f^*(X) = (f^*(x_i))_{i=1}^M$, we have the following result.

**Proposition A.7.** Let $T \in \mathcal{T}^k$ be finite. Then for any $f^*$ drawn at random from any centred Gaussian process on $D = [a, b]$ whose conditional mean $f^*|\mathcal{D}_n$ is the piecewise linear interpolant (in the range of $x_1, \ldots, x_n$) of the data $\mathcal{D}_n$ such that $A_{\rho,m,k,\tau}(f^*) = T$, we have $\epsilon \,|\, T \overset{d}{=} \epsilon_X^{\texttt{Trap}} \,|\, f^*(X)$.

*Proof.* A termination $T$ of `AdapTrap` corresponds to a set $S \subseteq \mathbb{R}^M$ such that $f^*(X) \in S$. Note that for any $f^* \in C([a, b])$ such that $f^*(X) \in S$ we have

$$\texttt{AdapTrap}_{\rho,m,k}(f^*, a, b, \tau) = \texttt{Trap}(f^*, a, b, X) \Rightarrow \epsilon(f^*) = \epsilon_X^{\texttt{Trap}}(f^*).$$

In the following we identify $f_i^* = f^*(x_i)$. Thus[14],

$$p(\epsilon \,|\, T) = \frac{1}{\mathbb{P}^*(T)} \int_S p(\epsilon \,|\, f_1^*, \ldots, f_M^*) p(f_1^*, \ldots, f_M^*)\,\mathrm{d}\mathbf{f}^*,$$

where $\mathbf{f}^* = (f_1^*, \ldots, f_M^*)$. Since, for any $f^*(X) \in S$, $\epsilon \,|\, f_1^*, \ldots, f_M^* \overset{d}{=} \epsilon_X^{\texttt{Trap}} \,|\, f_1^*, \ldots, f_M^*$ and $p(\epsilon_X^{\texttt{Trap}} \,|\, f_1^*, \ldots, f_M^*)$ is only a function of $X$, we have $p(\epsilon \,|\, f_1^*, \ldots, f_M^*) = g(\epsilon, X)$. Thus,

$$\begin{aligned} p(\epsilon \,|\, T) &= g(\epsilon, X) \frac{1}{\mathbb{P}^*(T)} \int_S p(f_1^*, \ldots, f_M^*)\,\mathrm{d}\mathbf{f}^* \\ &= g(\epsilon, X) \frac{\mathbb{P}^*(T)}{\mathbb{P}^*(T)} \\ &= g(\epsilon, X). \end{aligned}$$

For any $f^*(X) \in S$ we have $g(\epsilon, X) = p(\epsilon \,|\, f^*(X))$ which implies that $\epsilon \,|\, T \overset{d}{=} \epsilon_X^{\texttt{Trap}} \,|\, f^*(X)$. □

Finally we turn our attention to the proof of Proposition A.1. The distribution of the error of `AdapTrap` can be computed as

$$p(\epsilon) = \sum_{T \in \mathcal{T}^k \setminus T_\infty} p(\epsilon \,|\, T) \mathbb{P}^*(T) + \delta(\infty) \mathbb{P}^*(A_{\rho,m,k,\tau} \in T_\infty),$$

where we have formally defined the event of non-termination as having infinite error (i.e. for $T \in T_\infty$). We can now directly prove Proposition A.1:

[14] Let $X, Y$ be real random vectors and let $S_X, S_Y$ be events of $X$ and $Y$ respectively. Note that $P(X \in S_X | Y \in S_Y) = \frac{P(X \in S_X, Y \in S_Y)}{P(Y \in S_Y)} = \frac{1}{P(Y \in S_Y)} \int_{S_X} \int_{S_Y} p(x \,|\, y) p(y)\,\mathrm{d}x\,\mathrm{d}y$.

*Proof of Proposition A.1.* For any $\rho, m, \tau$ and $k$ an even integer we have

$$p(\epsilon) = \sum_{T \in \mathcal{T}^k \backslash T_\infty} p(\epsilon \,|\, T)\mathbb{P}^*(T) + \delta(\infty)\mathbb{P}^*(A_{\rho,m,k,\tau} \in T_\infty),$$

then for any finite $T \in \mathcal{T}^k$ we have $\mathbb{P}^*(|\epsilon| > \tau) > \mathbb{P}^*(|\epsilon| > \tau \,|\, T)\mathbb{P}^*(T)$. Let $T_1$ be the full $k$-ary tree with 1 node. Then, by Proposition A.3 we have $\mathbb{P}^*(T_1) = \alpha_0$ where $\alpha_0 = \text{Prob}_{Z \sim \mathcal{N}(0,1)}\left(|Z| < \frac{4m\tau}{\sqrt{\lambda(b-a)^3}}\right)$ and further by Proposition A.7, we have $\epsilon \,|\, T_1 \sim \mathcal{N}(0, \sigma_1^2)$ and so

$$\mathbb{P}^*(|\epsilon| > \tau) > \mathbb{P}^*(|\epsilon| > \tau \,|\, T_1)\mathbb{P}^*(T_1)$$

By Proposition A.6 we have

$$\sigma_1^2 = \frac{\lambda}{12}\sum_{i=1}^{2m} \frac{(b-a)^3}{(2m)^3} = \frac{\lambda(b-a)^3}{48m^2}.$$

Thus we have

$$\begin{aligned}\mathbb{P}^*(|\epsilon| > \tau) &> \mathbb{P}^*(|\epsilon| > \tau \,|\, T_1)\mathbb{P}^*(T_1) \\ &= \left[1 - \text{erf}\left(\frac{2\sqrt{6}m\tau}{\sqrt{\lambda(b-a)^3}}\right)\right] \text{erf}\left(\frac{2\sqrt{2}m\tau}{\sqrt{\lambda(b-a)^3}}\right).\end{aligned}$$

where $\text{erf}(x) := \frac{1}{\sqrt{\pi}}\int_{-x}^{x} e^{-t^2}\,\mathrm{d}t$ is the error function. This completes the proof, with $\mathbb{P}^*$-dependent constant $c := 2\sqrt{2}m\lambda^{-1/2}(b-a)^{-3/2}$. □

It is clear that the $T_1$-based bound employed in the proof of Proposition A.1 can be improved by taking into account a larger number of terms; however we were unable to find an elegant bound when proceeding in this manner and therefore we present only the simplest bound.

# B The AdapBC Algorithm

The AdapBC algorithm, in which $\theta = (c, \sigma, \ell(\cdot))$ is marginalised instead of being optimised, is displayed in Algorithm 3.

Lines 4 and 8 each require MCMC to be used. As such, AdapBC demands that the user carefully monitors the convergence of a Markov chain and, in turn, requires more technical knowledge on the part of the user compared to E-AdapBC.

Here $M$ is the number of samples of $f \,|\, \mathcal{D}_{n-1}$ and for each $m = 1, \dots, M$ and each $x \in D_n$, $K$ is the number of samples of $\theta \,|\, \mathcal{D}_{n-1} \cup \{x, f_m(x)\}$. Note that to estimate $\mathbb{V}[I(f)|\tilde{\mathcal{D}}_n]$ we used the law of total variance, that is

$$\begin{aligned}\mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n] &= \mathbb{E}[\mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta]] + \mathbb{V}[\mathbb{E}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta]] \\ &\approx \frac{1}{K}\sum_{k=1}^{K} \mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta_k] + s.v.\left(\{\mathbb{E}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta_k]\}_{k=1}^{m}\right),\end{aligned}$$

where $s.v.(X)$ is the sample variance of the set $X$.

# C Details on the Non-Stationary Model

In this section we provide full details of the non-stationary stochastic process model that our algorithms employed for the experimental assessment. In particular, we employed a hierarchical Gaussian process model $f|\theta \sim \mathcal{GP}(m_\theta, k_\theta)$ on $[0,1]^d \subset \mathbb{R}^d$ with

$$m_\theta(x) \;=\; c, \qquad k_\theta(x, y) \;=\; \sigma^2 \prod_{i=1}^{d} k_i(x_i, y_i), \tag{15}$$

**Algorithm 3** Adaptive Bayesian Cubature

```
procedure AdapBC($f^*, \tau$)
  $n \leftarrow 1$, $\tilde{\epsilon} \leftarrow \infty$
  while $\tilde{\epsilon} \geq \tau$ do
    Sample $(f_m)_{m=1}^M \sim f \,|\, \mathcal{D}_{n-1}$                                   ▷ $M \gg 1$
    for each $x$ in $D_n$ do
      for $m = 1, \ldots, M$ do
        $\tilde{\mathcal{D}}_n \leftarrow \mathcal{D}_{n-1} \cup \{(x, f_m(x))\}$
        Sample $(\theta_k)_{k=1}^K \sim \theta \,|\, \tilde{\mathcal{D}}_n$                        ▷ $K \gg 1$
        $V_m^k \leftarrow \mathbb{V}[I(f)|\tilde{\mathcal{D}}_n, \theta_k]$
        $E_m^k \leftarrow \mathbb{E}[I(f)|\tilde{\mathcal{D}}_n, \theta_k]$
        $\bar{V}_m \leftarrow \frac{1}{K}\sum_{k=1}^K V_m^k$
        $\bar{E}_m \leftarrow \frac{1}{K}\sum_{k=1}^K E_m^k$
        $\hat{V}_m(x) \leftarrow \hat{V}_m + \frac{1}{K}\sum_{k=1}^K (E_m^k - \bar{E}_m)^2$
      $\hat{E}(x) \leftarrow \frac{1}{M}\sum_{m=1}^M \hat{V}_m(x)$
    Pick $x_n \in \arg\min_{x \in D_n} \hat{E}(x)$
    $\mathcal{D}_n \leftarrow \mathcal{D}_{n-1} \cup \{(x_n, f^*(x_n))\}$
    $n \leftarrow n + 1$, $\tilde{\epsilon} \leftarrow \mathbb{V}[I(f)|\mathcal{D}_n]^{\frac{1}{2}}$
  return $I(f)|\mathcal{D}_n$
```

where $x = (x_1, \ldots, x_d), y = (y_1, \ldots, y_d)$ and the $k_i(x_i, y_i)$ are symmetric positive definite functions defined over $[0, 1]$ of the form

$$k_i(x_i, y_i) = \frac{\sqrt{\ell_i(x_i)\ell_i(y_i)}}{\sqrt{\ell_i(x_i)^2 + \ell_i(y_i)^2}} \phi\left(\frac{|x_i - y_i|}{\sqrt{\ell_i(x_i)^2 + \ell_i(y_i)^2}}\right), \tag{16}$$

where $\phi : [0, \infty) \to \mathbb{R}$ is a symmetric positive definite radial basis function and $\ell_i : [0, 1] \to (0, \infty)$ is a length scale function. Thus the parameters to be inferred are $\theta = \{c, \sigma, \ell_1(\cdot), \ldots, \ell_d(\cdot)\}$.

**Radial Basis function:** In the computational experiments detailed in the paper the choice of radial basis function $\phi$ was the standard Matérn radial basis function with smoothness parameter $\nu = 3/2$. Recall that the Matérn radial basis function for $\nu = a + 1/2$ for some $a \in \mathbb{Z}^+$ is of the form

$$\phi^{\nu}_{\text{Mat}}(d) = \exp\left(-d\sqrt{2a+1}\right) \frac{a!}{(2a)!} \sum_{i=0}^{a} \frac{(a+i)!}{i!(a-i)!} \left(2d\sqrt{2a+1}\right)^{a-i}. \tag{17}$$

For fixed $\theta$, the kernel $k_\theta$ reproduces a Sobolev space of dominating mixed smoothness; see e.g. Dick and Pillichshammer (2010). The impact of this choice is explored in Appendix E.2.

**Lengthscale Field:** The lengthscale field can be parameterised in arbitrarily complex ways. In particular, we highlight the recent work of Roininen et al. (2019) who focussed on performing computation with a hierarchical parametrisation of a Matérn kernel. In that paper, sophisticated MCMC samplers were proposed, along with an acknowledgement of the difficulty of the computational task. Since sampling methods are not the focus of our work, for computational tractability we specified a simple and transparent parameterisation for each $i = 1, \ldots, d$,

$$\ell_{\theta_i}(x_i) = \sum_{j=1}^{n-1} \frac{\beta_{i,j+1} - \beta_{i,j}}{\bar{x}_{i,j+1} - \bar{x}_{i,j}} x_i - \frac{\beta_{i,j+1} - \beta_{i,j}}{\bar{x}_{i,j+1} - \bar{x}_{i,j}} \bar{x}_{i,j} + \beta_{i,j}.$$

Thus $\ell_i(\cdot) := \ell_{\theta_i}(\cdot)$ is the piecewise linear interpolant of a finite number of fixed reference points $(\bar{x}_{i,1}, \beta_{i,1}), \ldots, (\bar{x}_{i,n}, \beta_{i,n})$ with $\bar{x}_{i,1} = 0$ and $\bar{x}_{i,n} = 1$ and thus the parameters to be inferred are $\theta_i = (\beta_{i,1}, \ldots, \beta_{i,n})$. This is computationally tractable since the number of parameters can be controlled and both the $\ell_i(\cdot)$ and $\ell_i(\cdot)^{-1}$ have closed form integrals (which we used in the regularisation of `E-AdapBC` in Appendix D.4). Positivity of $\ell_i(x_i)$ is ensured by taking $\beta_{i,j} = \exp(\alpha_{i,j})$ and inferring the $\alpha_{i,j} \in \mathbb{R}$. In all of our experiments we re-parametrise the domain to be $D = [0, 1]^d$ and we took $n = 11$ and $\bar{x}_{i,j} = \frac{j-1}{n-1}$, which allowed for sufficient expressiveness of the associated stochastic process model whilst controlling the complexity of the auxiliary computational task of estimating the $\alpha_{i,j}$. The total number of parameters associated with the lengthscale field $\ell(\cdot)$ is therefore $11d$. The impact of using this parametrisation of the lengthscale field was investigated in Appendix E.3.

## D Computational Details

It still remains to provide full computation details for `AdapBC` (Algorithm 3) and `E-AdapBC` (Algorithm 2) in each of the experiments performed. In this section the generic aspects of these details are provided. However, we note that certain details are particular to one or more of the experiments and these remaining experiment-specific details are clarified in full in Appendix E, where the experiments are described.

### D.1 Generic Aspects of `AdapBC` and `E-AdapBC`

First we discuss the computational details that both `AdapBC` and `E-AdapBC` have in common before discussing their differing aspects individually.

**Initial Data:** The set $\mathcal{D}_0$ of points on which our integrand $f^*$ is *a priori* evaluated must be specified. In this work we avoided the "obvious" choice $\mathcal{D}_0 = \emptyset$ since it is unreasonable to expect any inferential approach to provide well-calibrated uncertainty assessment at such low values as $n = 2, 3$ etc. Therefore, we took $\mathcal{D}_0$ to be an experiment-specific small set of mesh points in $D$. The specific choices are reported in Appendix E.

**Point Set Selection:** The point set $D_n$ is the set over which we optimise the objective function $x \mapsto E(x)$ (for `E-AdapBC`) or $x \mapsto \hat{E}(x)$ (for `AdapBC`). These objectives are non-convex in general and thus a global optimisation method must be employed. Since this auxiliary computation is assumed negligible with respect to evaluation of the integrand, we employed brute force grid search with $D_n$ used to define the grid.

In one dimension, $D_n$ was taken to be the following: Let $\{x_i\}_{i=1}^K$ be the set of abscissae on which $f^*$ has been evaluated after iteration $n$ of the algorithm has completed. Then we set

$$D_n := \{(x_i + x_{i+1})/2 \,|\, i = 1, \dots, K-1\}.$$

Although the "natural" generalisation of this approach to dimension $d > 1$ is a Voronoi point set, we instead preferred to endow $D_n$ with a structure commensurate with the tensor product form of the kernel $k_\theta$ in (15). Thus, in dimensions $d > 1$, $D_n$ was taken to be a randomly sampled subset of cardinality $K_n := K + 1 - n$ for some $K \in \mathbb{N}$ of a uniform grid of points on $D$. The computational convenience of the grid structure is explained in further detail in Appendix D.2.

More precisely, let $U = \{u_1, \dots, u_k\} \subset [0, 1]$ be a uniform grid of points on $[0, 1]$ and define $\bar{D}_1 = U^d \setminus D_0$ and $\bar{D}_{n+1} = \bar{D}_n \setminus \{x_n\}$, where $x_n$ is the point selected at step $n$ of the integration method. Then $D_n$ was taken to be a random sample without replacement from $\bar{D}_n$ such that $|D_n| = K_n$.

### D.2 Consequences of the Tensor Product Set-Up

Note that, at iteration $n$, the evaluation of the objective functions $E(x)$ (for `E-AdapBC`) and $\hat{E}(x)$ (for `AdapBC`) requires the computation of integrals of the conditional mean and covariance of $f|\theta, \mathcal{D}_n$ to be performed. In the discussion that follows we focus on `E-AdapBC` for simplicity, where in principle $K$ separate $d$-dimensional integrals are required to evaluate $E(x)$. Further, the approximate computation of $\arg\min_{x \in D_n} E(x)$ that we perform requires the computation of $K \times |D_n|$ of these $d$-dimensional integrals. However, since the kernel $k_\theta$ in (15) is a tensor product, then at most $dK \times |D_n|$ univariate integrals are necessary for computation of $\arg\min_{x \in D_n} E(x)$. Furthermore, if the chosen point set $D_n$ is some subset of a uniform grid $\{u_1, \dots, u_k\}^d \subset [0, 1]^d$, we can perform memoisation of the univariate integrals at each $u_i$. This reduces the computation of $\arg\min_{x \in D_n} E(x)$ to only require $dk$ univariate integrals. If the chosen univariate kernels are of the form in (16), then the integrals are of the form

$$\int_0^1 \frac{\sqrt{\ell_{\theta_i}(x)\ell_{\theta_i}(u_k)}}{\sqrt{\ell_{\theta_i}(x)^2 + \ell_{\theta_i}(u_k)^2}} \phi\left(\frac{|x - u_k|}{\sqrt{\ell_{\theta_i}(x)^2 + \ell_{\theta_i}(u_k)^2}}\right) \mathrm{d}x.$$

If the length scale function is piecewise linear, this integrand is piecewise as smooth as the choice of $\phi$ and further has no closed form integral. Thus to integrate these functions we integrated each piece separately using a standard Python quadrature[15] function in scipy. In the cases where the integral of the kernel was available in closed form then this was used instead.

To return $I(f)|\mathcal{D}_n$ we need to compute the mean and variance of the integral of the posterior process (see equations (7) and (8) in the main text). The computation of these terms requires computing $|\mathcal{D}_n|$ $d$-dimensional integrals and $|\mathcal{D}_n|$ $2d$-dimensional integrals. The univariate integrals were computed in the same way as before. For similar reasons to those outlined in the previous paragraph, the use of the tensor product reduces this requirement to $d|\mathcal{D}_n|$ bivariate integrals. If the chosen univariate kernels are of the form in (16), then the integrals are of the form

$$\int_0^1 \int_0^1 \frac{\sqrt{\ell_{\theta_i}(x)\ell_{\theta_i}(y)}}{\sqrt{\ell_{\theta_i}(x)^2 + \ell_{\theta_i}(y)^2}} \phi \left( \frac{|x-y|}{\sqrt{\ell_{\theta_i}(x)^2 + \ell_{\theta_i}(y)^2}} \right) \mathrm{d}x \, \mathrm{d}y.$$

This integrand is smooth over square subregions of $[0,1]^2$ and so is computed by integrating over each of these subregions separately using the standard double quadrature function in scipy. Again, if this integral was available in closed form then this was used instead.

### D.3 Details Specific to AdapBC

It remains to explain how MCMC was used to facilitate the computation on lines 4 and 8 of Algorithm 3 describing the AdapBC method. These details are now provided.

**Sampling from $\theta \,|\, \tilde{\mathcal{D}}_n$:** Due to the difficulty in directly sampling from $\theta \,|\, \mathcal{D}_n$ we used a Metropolis-Hastings algorithm. Note that

$$p(\theta \,|\, \mathcal{D}_n) \propto p(\theta) p(\mathcal{D}_n \,|\, \theta),$$

where $p(\theta)$ is the prior density of $\theta$ (yet to be specified) and we have $\mathcal{D}_n \,|\, \theta \sim \mathcal{N}(c\mathbf{1}, k_{\theta,X,X})$. Define $q(\theta) := p(\theta)p(\mathcal{D}_n \,|\, \theta)$, then our Metropolis algorithm is as follows:

**Algorithm 4** Metropolis-Hastings Algorithm

```
procedure METROPOLIS($\theta_0, n, s$)
    $\theta \leftarrow \theta_0$
    for $i = 1, \ldots, n$ do
        Sample $\theta^* \leftarrow \theta_{i-1} + \mathcal{N}(\mathbf{0}, s^2 I)$
        Sample $u \sim U(0,1)$
        if $\log u < \log q(\theta^*) - \log q(\theta_{i-1})$ then
            $\theta_i \leftarrow \theta^*$
        else
            $\theta_i \leftarrow \theta_{i-1}$
    return $(\theta_i)_{i=1}^n$
```

The proposal distribution here is thus $\mathcal{N}(\mathbf{0}, s^2 I)$. Figure 5 contains typical trace plots of Metropolis output.

**Sampling from $f \,|\, \mathcal{D}_{n-1}$:** In order to obtain a sample $\tilde{f}$ from the posterior marginal $f \,|\, \mathcal{D}_{n-1}$ we used *ancestral sampling*; i.e. we first sample $\tilde{\theta}$ from $\theta \,|\, \mathcal{D}_{n-1}$ and then we sample $\tilde{f}$ from $f \,|\, \mathcal{D}_{n-1}, \tilde{\theta}$. To obtain the sample $\tilde{\theta}$ we used the aforementioned Metropolis algorithm.

**Computing $\mathbb{E}[I(f)|\theta_k, \tilde{\mathcal{D}}_n]$ and $\mathbb{V}[I(f)|\theta_k, \tilde{\mathcal{D}}_n]$:** To compute $\mathbb{E}[I(f)|\theta_k, \tilde{\mathcal{D}}_n]$ we used the $1d$ integration methodology discussed in Appendix D.2. In order to compute $\mathbb{V}[I(f)|\theta_k, \tilde{\mathcal{D}}_n]$ we approximated the $2d$ integral in (8) with

$$\int_0^1 \int_0^1 k_{\theta_k}(x,y) \, \mathrm{d}x \, \mathrm{d}y \approx \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N k_{\theta_k}(x_i, x_j),$$

[15] The function being scipy.integrate.quad which, depending on input, calls a QUADPACK routine. In our case it calls QAGS, an adaptive quadrature based on 21-point Gauss-Kronrod quadrature within each subinterval. See Piessens (1983).

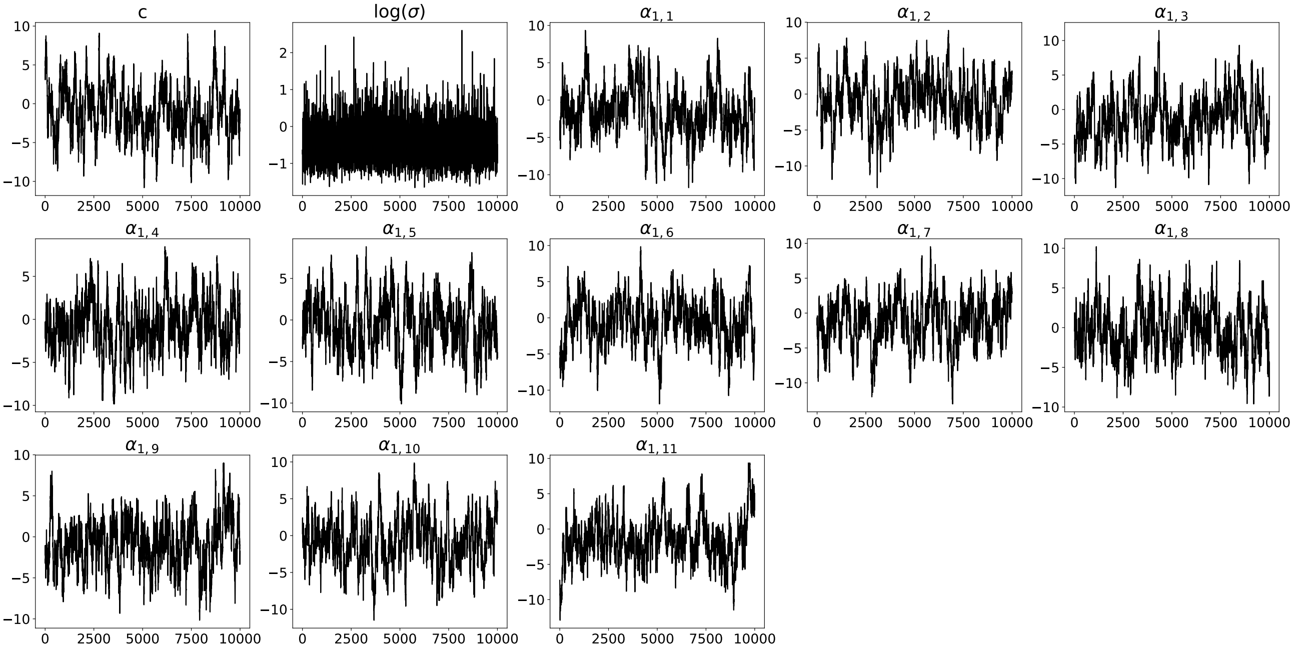


Figure 5: Trace plots for components of the parameter $\theta$ obtained using `Metropolis` under the prior $\theta \sim \mathcal{N}(-\mathbf{1}, 2I)$ with data $\mathcal{D}_0 = \{(i/5, f^*(i/5))\}_{i=0}^5$ with $f^*$ as in Figure 10.

where $x_i = \frac{i-1}{N-1}$ for some $N \in \mathbb{Z}^+$.

**Returning $I(f) \,|\, \mathcal{D}_n$:** To compute $\mathbb{E}[I(f) \,|\, \mathcal{D}_n]$ we use the following approximation, by the law of total expectation,

$$\begin{aligned}\mathbb{E}[I(f) \,|\, \mathcal{D}_n] &= \mathbb{E}[\mathbb{E}[I(f) \,|\, \mathcal{D}_n, \theta]] \\ &\approx \frac{1}{J}\sum_{i=1}^{J} \mathbb{E}[I(f) \,|\, \mathcal{D}_n, \theta_j],\end{aligned}$$

where $(\theta_j)_{j=1}^J$ is sampled from `Metropolis` and the expectation is computed using the methodology in Appendix D.2. To compute $\mathbb{V}[I(f) \,|\, \mathcal{D}_n]$ we use the following approximation, again using the law of total variance,

$$\begin{aligned}\mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n] &= \mathbb{E}[\mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta]] + \mathbb{V}[\mathbb{E}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta]] \\ &\approx \frac{1}{J}\sum_{j=1}^{J} \mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta_j] + s.v.\left(\{\mathbb{E}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta_j]\}_{j=1}^J\right),\end{aligned}$$

where $s.v.(X)$ is the sample variance of the set $X$. To compute the double integral in the computation of $\mathbb{V}[I(f) \,|\, \tilde{\mathcal{D}}_n, \theta]$ we used the $2d$ integration methodology discussed in Appendix D.2.

### D.4 Details Specific to `E-AdapBC`

It remains to be explained how the marginal likelihood $p(\mathcal{D}_{n-1}|\theta)$ was penalised to facilitate line 4 of Algorithm 2 describing the `E-AdapBC` method.

Line 4 of Algorithm 2 relates to computing the maximum of the (penalised) marginal likelihood $\theta \mapsto p(\mathcal{D}_{n-1} \,|\, \theta) - r(\theta)$. The likelihood function itself is derived from the Gaussian finite dimensional distribution of $f$ under the stochastic process model:

$$\log p(\mathcal{D}_{n-1} \,|\, \theta) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log[\det(k_{\theta,X,X})] - \frac{[f_X^* - c\mathbf{1}]^\top k_{\theta,X,X}^{-1}[f_X^* - c\mathbf{1}]}{2},$$

where $X$ is the abscissae of $\mathcal{D}_{n-1}$ and $k_{\theta,X,X}$ is the matrix $k_{X,X}$ based on the kernel $k = k_\theta$.

It was demonstrated in Briol et al. (2019) that the empirical Bayesian approach to kernel parameters can lead to over-confident uncertainty quantification at small values of $n$ in the context of a standard BC method. The issue is more pronounced in `E-AdapBC` due to the increased dimension of the kernel parameter $\theta$ compared to `StdBC`. For this reason we included a penalty term $r(\theta)$ on line 2 to regularise the non-asymptotic regime (only) and to try to avoid over-confident estimation under the proposed `E-AdapBC` method. The regularisation term we used in $d$-dimensions was the following

$$r(\theta) = \prod_{i=1}^{d} (\lambda_1 \|\ell_{\theta_i}(\cdot)\|_1 + \lambda_2 \|1/\ell_{\theta_i}(\cdot)\|_1)$$

where $\|g\|_1 := \int_D |g(x)| \mathrm{d}\pi(x)$. The specific form of regularisation was heuristically motivated (only) and many other choices are possible - to limit scope these were not explored. The regularisation term includes two parameters, $\lambda_1$ and $\lambda_2$, which are used respectively to ensure the length scale doesn't get too large or small when the number $n$ of data is small. Specific values of $\lambda_1$ and $\lambda_2$ are reported in Appendix E. To optimise the (logarithm of the) penalised marginal likelihood the standard BFGS method was used.

# E Details for the Experimental Assessment

In this section all remaining experiment-specific details are provided.

## E.1 Illustration of Adaptation

In this section we detail the integration problem and how it was solved by `AdapTrap`, `StdBC` and `E-AdapBC` in the production of Figure 1.

The integrand in Figure 1 was randomly sampled according to the procedure in Appendix E.2 with parameters (to 3 s.f.) $C = 0.554, R = 0.0726, H = 1.64, F = 2.65$ and $P = 1$.

**`AdapTrap` parameters:** For `AdapTrap` we used $\rho = 0.5, m = 5, k = 2$ and with global error tolerances (from left to right) $\tau = 0.06, 0.04, 0.02$.

**`StdBC` setup:** In our `StdBC` arrangement we used the following Gaussian process model: Using the same Matérn ($\nu = 3/2$) radial basis function as we used for `E-AdapBC`, we took $f \mid c, \sigma, \ell \sim \mathcal{GP}(c, k_{\sigma,\ell}(x, y))$, where

$$k_{\sigma,\ell}(x, y) = \sigma^2 \phi^{\nu}_{\mathrm{Mat}} \left( \frac{|x - y|}{\ell} \right).$$

In our implementation of `StdBC` we used the `E-AdapBC` algorithm with this stationary Gaussian process with $\theta = (c, \sigma, \ell)$, $r(\theta) = 0$, with initial data $\mathcal{D}_0 = \{(\frac{i}{10}, f^*\left(\frac{i}{10}\right))\}_{i=0}^{10}$ and the point selection algorithm as detailed in Appendix D.1.

**`E-AdapBC` setup:** In our `E-AdapBC` implementation we used the non-stationary model detailed in Appendix C, where $\ell_1$ is a piecewise linear function defined on $n = 11$ uniform knots. Our regularisation term $r(\theta)$ was detailed in Appendix D.4, we took $\lambda_1 = 30$ and $\lambda_2 = 1$. We further used the initial data $\mathcal{D}_0 = \{(\frac{i}{10}, f^*\left(\frac{i}{10}\right))\}_{i=0}^{10}$ and the point selection algorithm as detailed in Appendix D.1.

## E.2 Synthetic Assessment

In this section we detail how our results in Section 4.2 were created.

**Synthetic Integrand Generation:** Our synthetic integrands in $d$ dimensions are generated as follows: First, we sample

1. $C = (C_1, \dots, C_d) \sim U(0.1, 0.9)^d$,
2. $R = (R_1, \dots, R_d) \sim \mathrm{Beta}(5, 2)^d$,

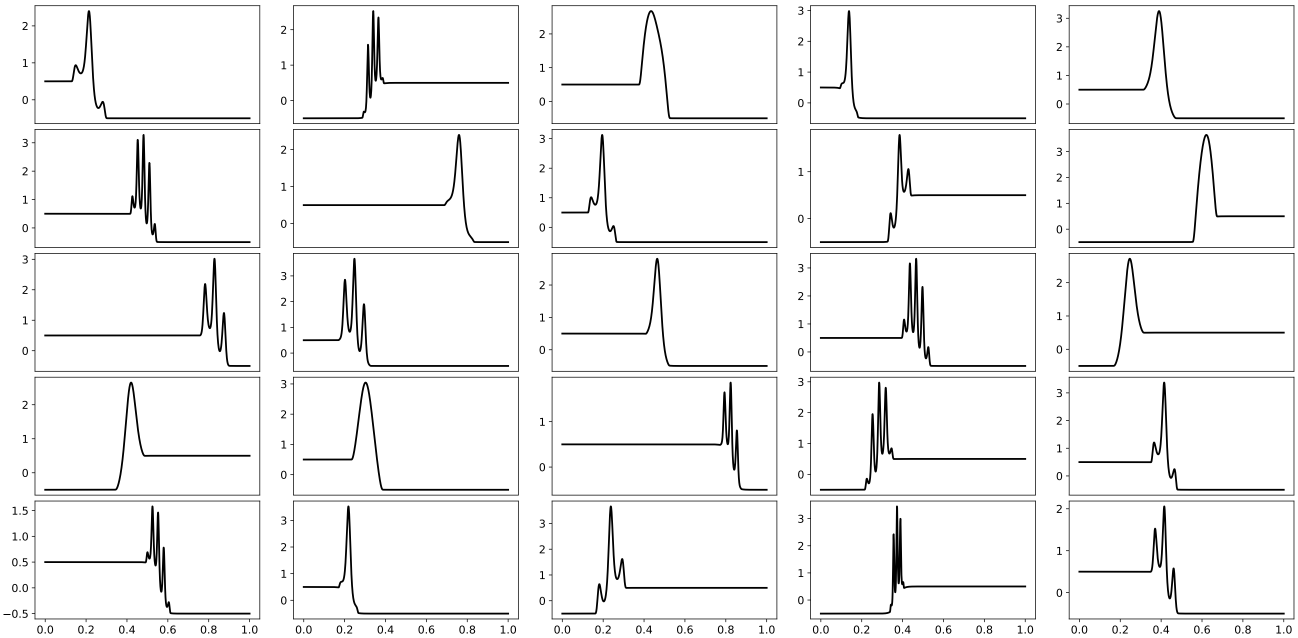


Figure 6: 25 randomly generated synthetic functions in $1d$.

3. $H = (H_1, \dots, H_d) \sim U(0.5e, 1.5e)^d$,
4. $F = (F_1, \dots, F_d) \sim U(0, 5)^d$,
5. $P = (P_1, \dots, P_d) \sim \text{Bernoulli}(0.5)^d$.

Then we let

$$h(x) = \frac{1}{1 + \exp(-80x)}, \qquad g_F(x) = \begin{cases} 0, & |x| \geq 1, \\ \exp\left\{\frac{-1}{1-|x|^2} + \cos(F\pi|x|)\right\}, & |x| < 1. \end{cases}$$

Our synthetic integrand is then

$$f^*(x) = \prod_{i=1}^{d} H_i g_{F_i}\left(\frac{1}{R_i}[x_i - C_i]\right) + (-1)^{P_i}[1/2 - h(x_i - C_i)],$$

with $x = (x_1 \dots, x_d)$. In Figure 6 we plot 25 randomly sampled synthetic integrands. To obtain the true integrals of these synthetic integrands we compute

$$\int_{[0,1]^d} f^*(x)\,\mathrm{d}x = \prod_{i=1}^{d} \int_0^1 H_i g_{F_i}\left(\frac{1}{R_i}[x_i - C_i]\right) + (-1)^{P_i}[1/2 - h(x_i - C_i)]\,\mathrm{d}x_i$$

and for each of the $1d$ integration problems we integrate each term separately using `scipy.integrate.quad` with its absolute error and relative error parameters taken as $10^{-10}$.

**Experiments in $1d$:** For our experiments in $1d$ we sampled 100 integrands according to our synthetic integrand generation procedure discussed in the previous paragraph and used the same implementations of `StdBC` and `E-AdapBC` discussed in Appendix E.1.

**Experiments in $3d$:** For our experiments in $3d$ we sampled 100 integrands according to our synthetic integrand generation procedure discussed in the previous paragraph. For both our implementations of `StdBC` and `E-AdapBC` we used the `E-AdapBC` algorithm with slight variations with each implementation. For both `StdBC` and `E-AdapBC` we took $\mathcal{D}_0 = \{(x, f^*(x))\}_{x \in G}$ where $G = \{0, 1/5, 2/5, 3/5, 4/5, 1\}^3$ and we used the point set selection algorithm discussed in Appendix D.1 with $U = \{i/40\}_{i=0}^{40}$ and $K_1 = 8000$.

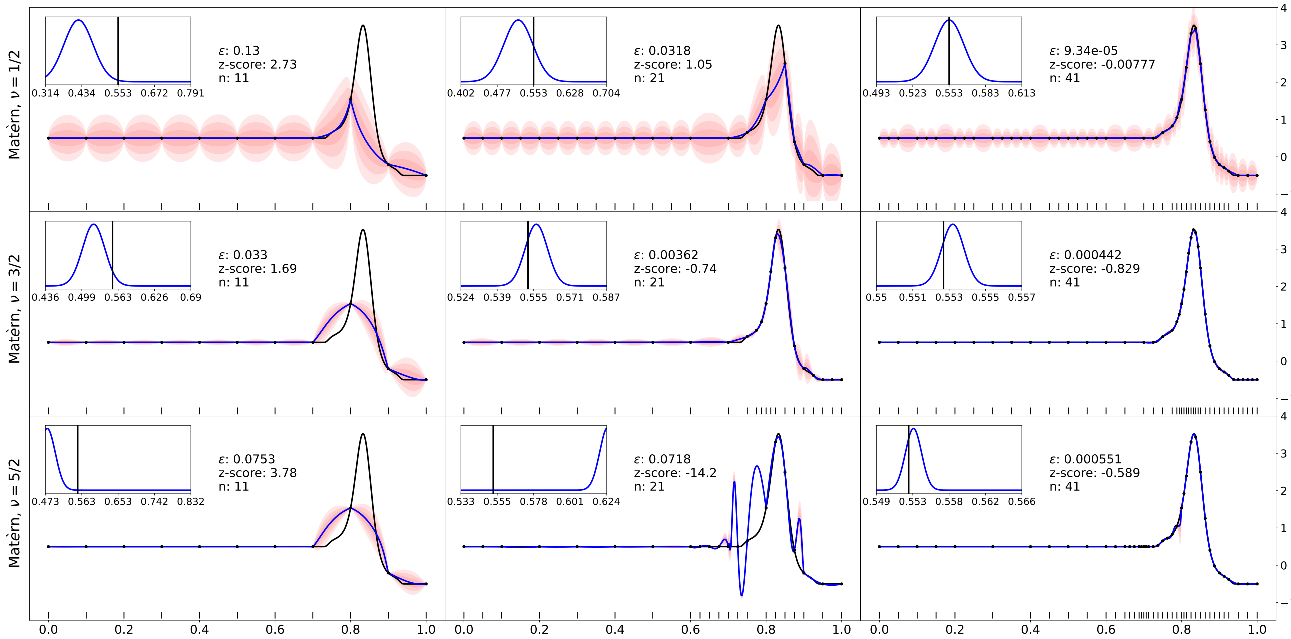


Figure 7: Each row corresponds to a different radial basis function with E-AdapBC run on the same integrand. The integrand was generated randomly from our synthetic integrand generation procedure with parameters (to 3 s.f.) $C = 0.835, R = 0.111, H = 3.50, F = 1.63, P = 0$. [Here — represents the true integrand $f^*$, — represents the mean of the conditional process $f|\mathcal{D}_n$ and 🟥 represents pointwise credible intervals. The tick marks | ||||| | indicate where the integrand was evaluated. For each radial basis function the error $\epsilon := |\mu_n(f^*) - I(f^*)|$, the z-score $[\mu_n(f^*) - I(f^*)]/\sigma_n(f^*)$ and the number of integrand evaluations $n$ are reported. Inset panels compare the true value $I(f^*) \approx 0.156$ to the distribution $I(f)|\mathcal{D}_n$.]

For our implementation of E-AdapBC the underlying Gaussian process follows what we detailed in Appendix C and the regularisation term follows what was detailed in Appendix D.4 with $\lambda_1 = 9, \lambda_2 = 0.9$.

For our implementation of StdBC the underlying Gaussian process was $f \mid c, \sigma, \ell \sim \mathcal{GP}(c, k_{\sigma,\ell}(x, y))$ where,

$$k_{\sigma,\ell}(x, y) = \sigma^2 \prod_{i=1}^{3} \phi^{\nu}_{\text{Mat}} \left( \frac{|x_i - y_i|}{\ell_i} \right).$$

where $\ell = (\ell_1, \ell_2, \ell_3), x = (x_1, x_2, x_3), y = (y_1, y_2, y_3)$ and $\nu = 3/2$. We further took $r(\theta) = 2(|\ell_1| + |\ell_2| + |\ell_3|)$, where $\theta = (c, \sigma, \ell)$.

### E.3 Variations of the Non-Stationary Model

In this section we explore variations in our non-stationary model specification in the use of the Algorithm 3.

**Different Choice of Radial Basis Function:** As discussed in Appendix C the radial basis function $\phi$ was taken to be the standard Matérn radial basis function with smoothness parameter $\nu = 3/2$ in all the experiments in the main text. In Figure 7 and Figure 9 we explore the robustness of E-AdapBC under different choices of radial basis function in our non-stationary model. In these experiments all other settings used in our non-stationary model (detailed in Appendix C) were kept the same. The radial basis functions that we chose were the Matérn radial basis function (17) with smoothness parameters $\nu = 1/2, 3/2, 5/2$.

**Different Lengthscale Fields:** In the following we explore the behaviour of E-AdapBC for different choices of lengthscale function in our non-stationary model. In these experiments all other settings used in our non-stationary model (detailed in Appendix C) were kept the same. The lengthscale functions that we compared

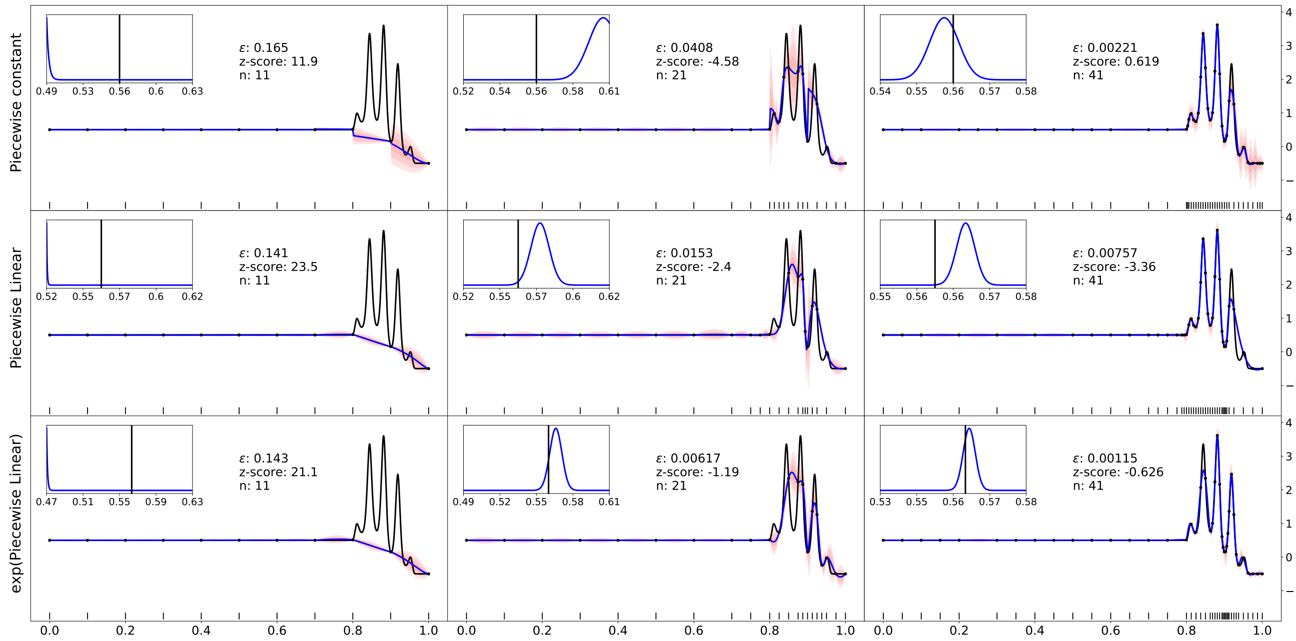


Figure 8: Each row corresponds to a different length scale function with `E-AdapBC` run on the same integrand. The integrand was generated randomly from our synthetic integrand generation procedure with parameters (to 3 s.f.) $C = 0.882, R = 0.0892, H = 3.61, F = 4.73, P = 0$. [Here — represents the true integrand $f^*$, — represents the mean of the conditional process $f|\mathcal{D}_n$ and 🟥 represents pointwise credible intervals. The tick marks | |||| | indicate where the integrand was evaluated. For each length scale function the error $\epsilon := |\mu_n(f^*) - I(f^*)|$, the z-score $[\mu_n(f^*) - I(f^*)]/\sigma_n(f^*)$ and the number of integrand evaluations $n$ are reported. Inset panels compare the true value $I(f^*) \approx 0.156$ to the distribution $I(f)|\mathcal{D}_n$.]

were piecewise linear, piecewise constant

$$\ell_{\theta_i}^{\text{const.}}(x_i) = \sum_{j=1}^{n} \beta_{i,j} \mathbb{1}_{[(j-1)/n, j/n)}(x_i),$$

where $\mathbb{1}_A(x)$ is the indicator function, and the exponential of piecewise linear

$$\ell_{\theta_i}^{\exp}(x_i) = \exp\left\{ \sum_{j=1}^{n-1} \frac{\beta_{i,j+1} - \beta_{i,j}}{\bar{x}_{i,j+1} - \bar{x}_{i,j}} x_i - \frac{\beta_{i,j+1} - \beta_{i,j}}{\bar{x}_{i,j+1} - \bar{x}_{i,j}} \bar{x}_{i,j} + \beta_{i,j} \right\}.$$

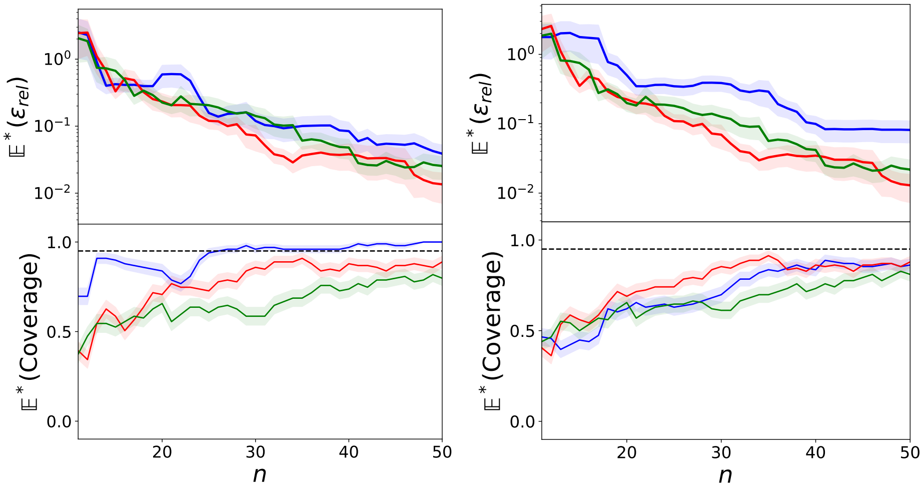
For the piecewise constant lengthscale, to ensure positivity we took $\beta_{i,j} = \exp(\alpha_{i,j})$ and inferred the $\alpha_{i,j} \in \mathbb{R}$. For the piecewise constant lengthscale we took $n = 10$ and for the other lengthscale functions we took $n = 11$. See Figure 8 and Figure 9 for our results.

### E.4 Full Bayes vs Empirical Bayes

In the following we test the differences in behaviour between `AdapBC` (Algorithm 3) and `E-AdapBC` (Algorithm 2). For this test we ran both methods on the same integrand which was generated randomly from our synthetic integrand generation procedure detailed in Appendix E.2. Results are shown in Figure 10s.

For our implementation of `E-AdapBC` we used the same settings as detailed in Appendix E.1.

For our implementation of `AdapBC` we used the same initial data $\mathcal{D}_0 = \{(\frac{i}{10}, f^*\left(\frac{i}{10}\right))\}_{i=0}^{10}$, the same point set selection algorithm and the same non-stationary Gaussian process as our implementation of `E-AdapBC` used. Our choice of prior was $\theta \sim \mathcal{N}(-\mathbf{1}, 2I)$. When sampling the $\theta \,|\, \tilde{\mathcal{D}}_n$ and the $\theta \,|\, \mathcal{D}_{n-1}$, to ensure a tolerable acceptance rate in the output from `Metropolis`, at each step of `AdapBC` we set $s = 0.3 - 0.07n$ for $n = 0, \ldots, 30$, so our proposal distribution used in `Metropolis` at step $n$ was $\mathcal{N}\left(\mathbf{0}, (0.3 - 0.07n)^2 I\right)$. In our approximation of



(a)

(b)

Figure 9: Synthetic assessment in $d = 1$ of E-AdapBC with (a) different choices of radial basis function and (b) different choices of lengthscale function. Plot (a) Matérn $\nu = 1/2$ (—), Matérn $\nu = 3/2$ (—) and Matérn $\nu = 5/2$ (—), where 100 integrands were randomly generated. Plot (b) piecewise constant (—), piecewise linear (—) and exponential of piecewise linear (—), where 100 integrands were randomly generated. Top row: the mean relative error against the number of evaluations $n$. Bottom row: the coverage frequencies for 95% credible intervals for each method. The notional coverage (- - -) is indicated. [Standard errors displayed.]

$\mathbb{V}[I(f)\,|\,\theta_k, \tilde{\mathcal{D}}_n]$ we set $N = 101$. For our parameters $M$ and $K$ that control the number of samples of $f\,|\,\tilde{\mathcal{D}}_{n-1}$ and $\theta\,|\,\tilde{\mathcal{D}}_n$ in AdapBC respectively, we took $M = K = 8$. All the output obtained from Metropolis was preceded by a length 1000 burn in and was thinned by 5. The $\theta_0$ in each run of Metropolis was taken as the last sample from the previous Metropolis output and at step 0 was taken to be the mean of the prior on $\theta$. In outputting $I(f)\,|\,\mathcal{D}_n$ we took $J = 50$.

Figure 10 suggests that AdapBC provides locally adaptive behaviour similar to E-AdapBC, but that AdapBC has better-calibrated uncertainty (in line with the previously documented over-confidence of Empirical Bayes in this context; Briol et al., 2019). However, the auxiliary computational cost associated with AdapBC is substantial - to produce Figure 10 the AdapBC method required 24 hours of CPU time whereas E-AdapBC required approximately one minute of CPU time. In addition, the need to carefully control the MCMC algorithm within AdapBC makes this method less attractive compared to E-AdapBC.

### E.5 Autonomous Robot Assessment

In this section we detail our autonomous robot experiment. The autonomous robot that we studied is due to Chrono (2019b).

**Details of Robot:** In the following we provide the necessary details on the robot and how the actuators give motion. The robot was simulated in the open source physics engine Chrono (2019a). The robot has 6 legs with each leg consisting of 3 actuators which control the walking motion of the robot; see Figure 11. Each leg has 3 associated actuators that are depicted in Figure 12. Each actuator has a predefined loop. For each period $T = [\alpha, \beta]$ (lasting $\beta - \alpha = 2$ seconds) of the loop the actuators are controlled as follows:

- Legs 1, 3 and 5:

  (a) Actuator A:

$$f_A(x) = 0.2 \sin\left(\pi(x - \alpha)\right).$$

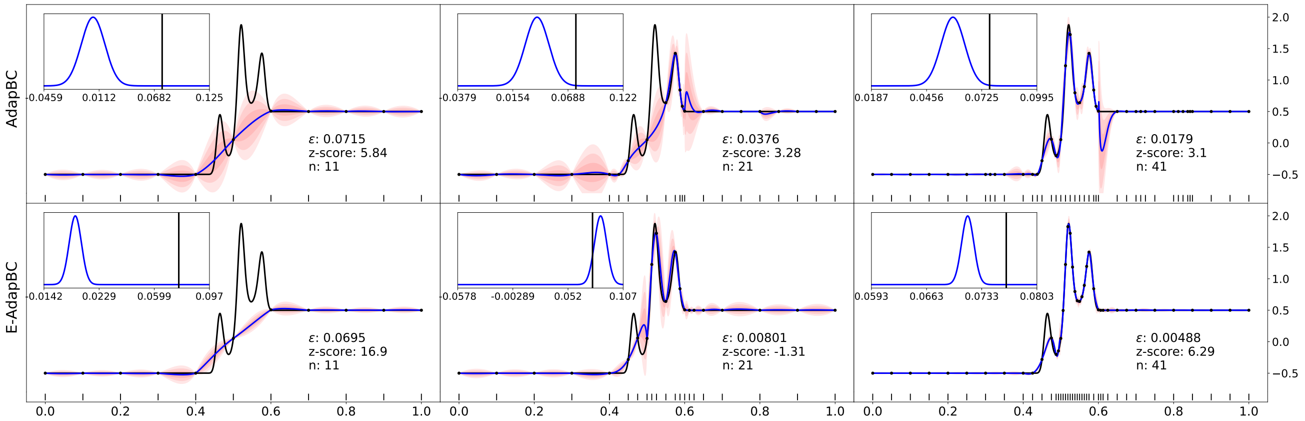


Figure 10: The upper and lower rows correspond to `AdapBC` and `E-AdapBC` respectively, run on the same integrand. The integrand was generated randomly from our synthetic integrand generation procedure with parameters (to 3 s.f.) $C = 0.520, R = 0.0897, H = 1.87, F = 3.04, P = 1$. [Here — represents the true integrand $f^*$, — represents the mean of the conditional process $f|\mathcal{D}_n$ and 🟥 represents pointwise credible intervals. The tick marks | |||| | indicate where the integrand was evaluated. For both methods the error $\epsilon := |\mu_n(f^*) - I(f^*)|$, the z-score $[\mu_n(f^*) - I(f^*)]/\sigma_n(f^*)$ and the number of integrand evaluations $n$ are reported. Inset panels compare the true value $I(f^*) \approx 0.0764$ to the distribution $I(f)|\mathcal{D}_n$.]

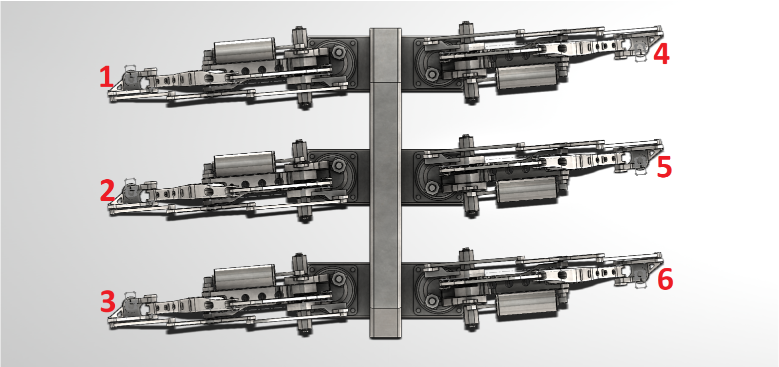


Figure 11: Elevated view of the robot, with each leg annotated.

(b) Actuator B:

$$f_B(x) = \begin{cases} -0.2\,\mathrm{Sig}(x-\alpha), & x \in [\alpha, \alpha+0.5], \\ -0.2, & x \in [\alpha+0.5, \alpha+1.5], \\ 0.2\,\mathrm{Sig}(x-\alpha-1.5), & x \in [\alpha+1.5, \beta]. \end{cases}$$

(c) Actuator C:

$$f_C(x) = 0.$$

- Legs 2, 4 and 6:

(a) Actuator A:

$$g_A(x) = -0.2\sin\left(\pi(x-\alpha)\right)$$

(b) Actuator B:

$$g_B(x) = \begin{cases} -0.2, & x \in [\alpha, \alpha+0.5], \\ 0.2\,\mathrm{Sig}(x-\alpha-0.5), & x \in [\alpha+0.5, \alpha+1], \\ -0.2\,\mathrm{Sig}(x-\alpha-1), & x \in [\alpha+1, \alpha+1.5], \\ -0.2, & x \in [\alpha+1.5, \beta]. \end{cases}$$

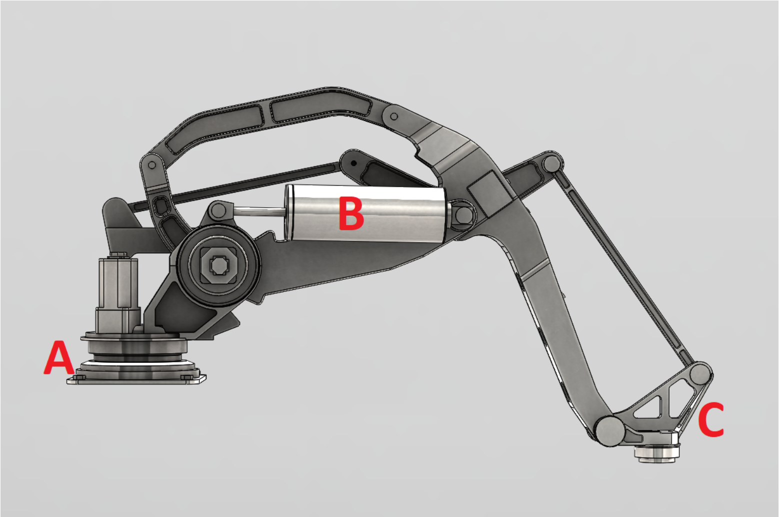


Figure 12: Detailed view of robot leg with each actuator labelled. Actuator A controls the rotation of the leg in the horizontal plane. Actuator B controls the up/down retraction of the leg. Actuator C controls the left/right extension of the leg.

(c) Actuator C:

$$g_C(x) = 0.$$

Here $\text{Sig}(x)$ is a polynomial smooth ramp such that $\text{Sig}(0) = 0$ and $\text{Sig}(0.5) = 1$.

**Robot Experimental Details:** In our robot experiment we investigated the distribution of spatial location of the robot after a prescribed time of movement under uncertainty in the parameterisation of the functions that control the actuators in leg 1 of the robot. Our functions that controlled the actuators in leg 1 subject to our parameterisation are as follows, for each period $T = [\alpha, \beta]$:

$$\begin{aligned} f_A(x) &= \sin(\pi(x-\alpha)), \\ f_B(x) = f_B(x) &= \begin{cases} -(0.2+p_1)\,\text{Sig}(x-\alpha), & x \in [\alpha, \alpha+(1-p_2)/2], \\ -(0.2+p_1), & x \in [\alpha+(1-p_2)/2, \alpha+(3+p_2)/2], \\ (0.2+p_1)\,\text{Sig}(x-\alpha-1.5), & x \in [\alpha+(3+p_2)/2, \beta]. \end{cases} \\ f_C(x) &= p_3, \end{aligned}$$

Thus $p_1$ controls the how far the leg travels up and down in each period, $p_2$ controls how long the leg is down for in each period and $p_3$ controls the extension of the leg. In our experiment we took $(p_1, p_2, p_3) \sim \mathcal{N}(0, \frac{1}{10} I_{3\times 3})$ and so after reparameterisation we have $x = (x_1, x_2, x_3) \sim \mathcal{N}(0, I_{3\times 3})$ such that each $x_i = \frac{1}{\sqrt{10}} p_i$. In our experiment $z_1(x)$ and $z_2(x)$ were the spatial coordinates of the robot after 10 seconds of movement. In our implementation we used `Chrono`'s inbuilt Barzila-Borwein solver with a discretisation time step of 0.005s.

For both our implementations of `StdBC` and `E-AdapBC` we used the `E-AdapBC` algorithm with slight variations with each implementation. For both `StdBC` and `E-AdapBC` we took $\mathcal{D}_0 = \{(x, f^*(x))\}_{x\in G}$ where $G = \{1/5, 2/5, 3/5, 4/5\}^3$ and we used the point set selection algorithm discussed in Appendix D.1 with $U = \{i/40\}_{i=1}^{39}$ and $K_1 = 8000$. For each integrand we ran both methods to evaluate the integrand 200 times and thus at termination we were using 264 points.

For our implementation of `E-AdapBC` the underlying Gaussian process follows what we detailed in Appendix C and the regularisation term follows what was detailed in Appendix D.4 with $\lambda_1 = 10, \lambda_2 = 0.8$.

For our implementation of `StdBC` the underlying Gaussian process was $f \mid c, \sigma, \ell \sim \mathcal{GP}(c, k_{\sigma,\ell}(x, y))$ where,

$$k_{\sigma,\ell}(x, y) = \sigma^2 \prod_{i=1}^{3} \phi_{\text{Mat}}^{\nu}\left(\frac{|x_i - y_i|}{\ell_i}\right).$$

| $f^*$ | StdBC | E-AdapBC |
|---|---|---|
| $z_1$ | $\mu_n = 0.1095, \sigma_n = 0.02129$ | $\mu_n = 0.06451, \sigma_n = 0.008535$ |
| $z_2$ | $\mu_n = -5.760, \sigma_n = 0.03812$ | $\mu_n = -5.4913, \sigma_n = 0.02373$ |
| $z_1^2$ | $\mu_n = 0.1643, \sigma_n = 0.01897$ | $\mu_n = 0.1252, \sigma_n = 0.007559$ |
| $z_2^2$ | $\mu_n = 32.93, \sigma_n = 0.3969$ | $\mu_n = 32.43, \sigma_n = 0.1562$ |

Figure 13: Autonomous robot experiment output to 4 s.f.

where $\ell = (\ell_1, \ell_2, \ell_3), x = (x_1, x_2, x_3), y = (y_1, y_2, y_3)$ and $\nu = 3/2$. We further took $r(\theta) = 2(|\ell_1| + |\ell_2| + |\ell_3|)$, where $\theta = (c, \sigma, \ell)$. The output of the experiments can be seen in Figure 13.

## F Full $k$-ary Trees

This section provides supporting material on the combinatorial results used in the average case analysis of the adaptive trapezoidal rule in Appendix A. In addition to basic definitions, it contains Theorem F.2 which was used in the proof of Corollary A.5.

**Definition F.1** (Rooted tree)**.** A *rooted tree* is a (possibly infinite) tree where one node is specified to be the root.

The *depth* $d(v)$ of a node $v$ in a rooted tree is the length of the path from the root to $v$. A node $v$ is a *child* of a node $u$ if $u$ and $v$ are connected by an edge and the depth of $v$ is 1 greater than the depth of $u$. A *leaf* of a rooted tree is a node with degree 1. An *inner node* of a rooted tree is a node with degree greater than 1. The *height* of a rooted tree $T$ is $\sup_{v \in T} d(v)$.

**Definition F.2** ($k$-ary tree)**.** A *$k$-ary* tree is a rooted tree such that every node has at most $k$ children.

A *full $k$-ary tree* is a $k$-ary tree where every node has exactly $k$ children or 0 children. We define the null tree to be a $k$-ary tree but not a full $k$-ary tree. Note that a tree with a single node is both a $k$-ary tree and a full $k$-ary tree. The set of all full $k$-ary trees is denoted $\mathcal{T}^k$. One can always create a full $k$-ary tree from a $k$-ary tree:

**Definition F.3** (Extension of a $k$-ary tree)**.** Let $S$ be a non-null $k$-ary tree. The *extension* of $S$ is the full $k$-ary tree $\overline{S}$ obtained by adding leaf nodes to $S$ such that every node in the original tree $S \subseteq \overline{S}$ has precisely $k$ children. The extension of the null $k$-ary tree is taken to be the single node full $k$-ary tree.

Note that this extension function $S \mapsto \overline{S}$ forms a bijection from the set of $k$-ary trees to the set of full $k$-ary trees.

**Theorem F.1** (Full $k$-ary tree theorem)**.** Let $S$ be a $k$-ary tree with $n$ nodes and let $\overline{S}$ be its extension. Then $\overline{S}$ has $nk + 1$ nodes.

*Proof.* The proof is by induction. The base case is trivial: Consider the null tree with 0 nodes, the extension of this tree has 1 node. Assume now that every $k$-ary tree with $n$ nodes has, in its extension, $nk + 1$ nodes. Note that any $k$-ary tree with $n + 1$ nodes can be formed by adding an additional node and edge to a $k$-ary tree with $n$ nodes. We can only add this extra node and edge to a node of degree at most $k$. In any of these cases the number of extra nodes added in this new tree's extension is $k$. That is, in this new tree of $n + 1$ nodes, the number of nodes in its extension is $nk + 1 + k = (n + 1)k + 1$. □

Thus a full $k$-ary tree with $n$ nodes has $\frac{n-1}{k}$ inner nodes and $\frac{(k-1)n+1}{k}$ leaves.

Next we consider the problem of counting the number of $k$-ary trees with a given number of nodes. Let $C_n^{(k)}$ be the number of $k$-ary trees with $n$ nodes with corresponding generating function $C_k(x) := \sum_{i=0}^{\infty} C_i^{(k)} x^i$. From (Graham et al., 1994), the $C_n^{(k)}$ follow the recurrence relation

$$C_{n+1}^{(k)} = \sum_{n_1 + n_2 + \ldots + n_k = n} C_{n_1}^{(k)} C_{n_2}^{(k)} \ldots C_{n_k}^{(k)}.$$

This recurrence relation yields the following functional equation,

$$C_k(x) = 1 + x[C_k(x)]^k.$$

For $k = 2$ this has the solution

$$C_2(x) = \frac{1-\sqrt{1-4x}}{2x}. \tag{18}$$

**Theorem F.2** (Number of $k$-ary trees with $n$ nodes)**.** The total number of $k$-ary trees with $n$ nodes is

$$C_n^{(k)} = \frac{1}{(k-1)n+1}\binom{nk}{n},$$

where $C_n^{(k)}$ is the $n$th $k$-Catalan number. Note that for $k = 2$, we get the standard Catalan numbers.

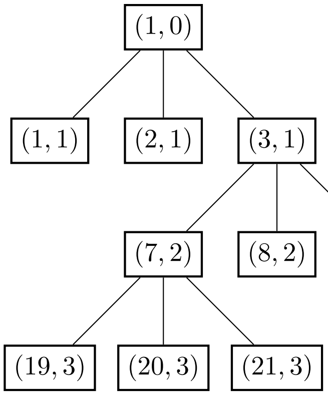
*Proof.* Use the Lagrange inversion theorem on the generating function's functional equation. See (Graham et al., 1994). □

Since the extension function defines a bijection from the set of $k$-ary trees to the set of full $k$-ary trees, the above result also counts the total number of full $k$-ary trees with $nk + 1$ nodes as $C_n^{(k)}$.

**Definition F.4** (Preorder traversal)**.** Let $T \in \mathcal{T}^k$ be finite. A preorder traversal of $T$ is a sequence of nodes $\langle v_i \rangle_{i=1}^N$ that is defined by the following steps:

1. Visit the root.
2. For $i = 1, \dots, k$, traverse the $i$th subtree from the left.

For example, consider the following full 3-ary tree:



The preorder traversal of this tree is the sequence $(1, 0), (1, 1), (2, 1), (3, 1), (7, 2), (19, 3), (20, 3), (21, 3), (8, 2), (9, 2)$.