Supplementary Appendix

Contents:

.1 Calculus for the base kernel k

All experiments were performed using the base kernel

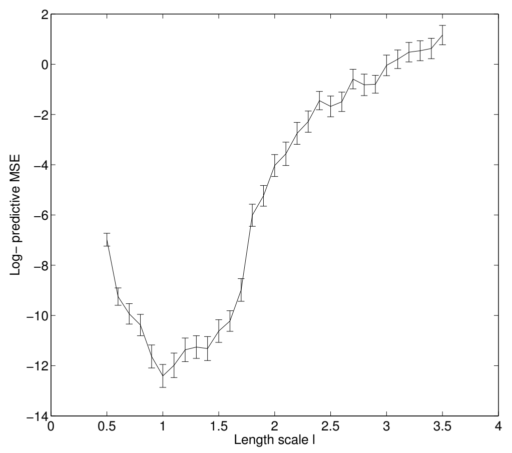
$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{(1+\alpha_1\|\boldsymbol{x}\|_2^2)(1+\alpha_1\|\boldsymbol{x}'\|_2^2)} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2\alpha_2^2}\right).$$

Below we provide formulae for the derivatives of the base kernel that are required in our method. Here $\boldsymbol{x} \in \mathbb{R}^{1\times d}$ is a row vector.

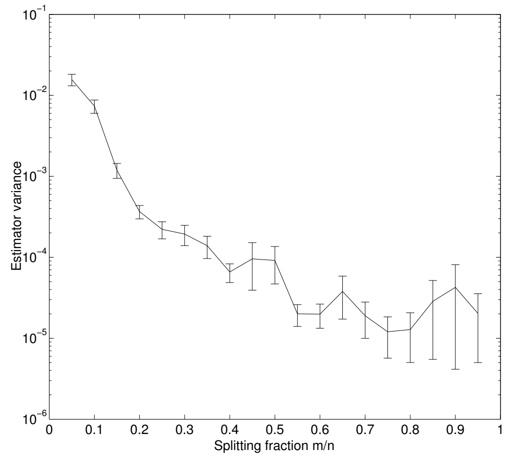
$$\begin{aligned}
\nabla_{\boldsymbol{x}} k(\boldsymbol{x}, \boldsymbol{x}') &= \left[\frac{-2\alpha_1 \boldsymbol{x}}{(1+\alpha_1\|\boldsymbol{x}\|_2^2)} - \frac{\boldsymbol{x}-\boldsymbol{x}'}{\alpha_2^2}\right] k(\boldsymbol{x}, \boldsymbol{x}') \\
\nabla_{\boldsymbol{x}'} k(\boldsymbol{x}, \boldsymbol{x}') &= \left[\frac{-2\alpha_1 \boldsymbol{x}'}{(1+\alpha_1\|\boldsymbol{x}'\|_2^2)} + \frac{\boldsymbol{x}-\boldsymbol{x}'}{\alpha_2^2}\right] k(\boldsymbol{x}, \boldsymbol{x}') \\
\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{x}'} k(\boldsymbol{x}, \boldsymbol{x}') &= \left[\frac{4\alpha_1^2 \boldsymbol{x}^T \boldsymbol{x}'}{(1+\alpha_1\|\boldsymbol{x}\|_2^2)(1+\alpha_1\|\boldsymbol{x}'\|_2^2)} + \frac{2\alpha_1(\boldsymbol{x}-\boldsymbol{x}')^T \boldsymbol{x}'}{\alpha_2^2(1+\alpha_1\|\boldsymbol{x}'\|_2^2)} \right. \\
&\quad \left. - \frac{2\alpha_1 \boldsymbol{x}^T(\boldsymbol{x}-\boldsymbol{x}')}{\alpha_2^2(1+\alpha_1\|\boldsymbol{x}\|_2^2)} + \frac{\boldsymbol{I}}{\alpha_2^2} - \frac{(\boldsymbol{x}-\boldsymbol{x}')^T(\boldsymbol{x}-\boldsymbol{x}')}{\alpha_2^4}\right] k(\boldsymbol{x}, \boldsymbol{x}')
\end{aligned}$$

The final term, $\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{x}'} k(\boldsymbol{x}, \boldsymbol{x}')$ is a $d \times d$ matrix with (i, j)th element $(\partial/\partial x_i)\nabla_{x'_j} k(\boldsymbol{x}, \boldsymbol{x}')$.

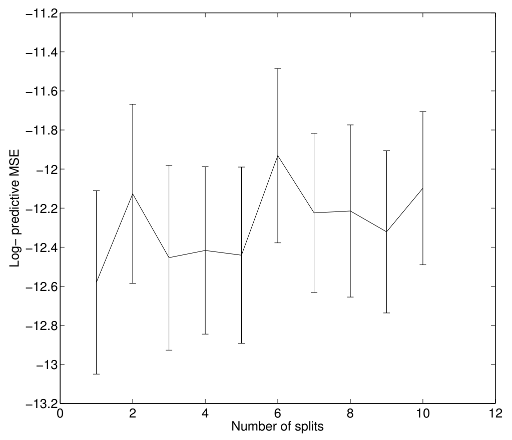
.2 Diagnostics for the synthetic illustration



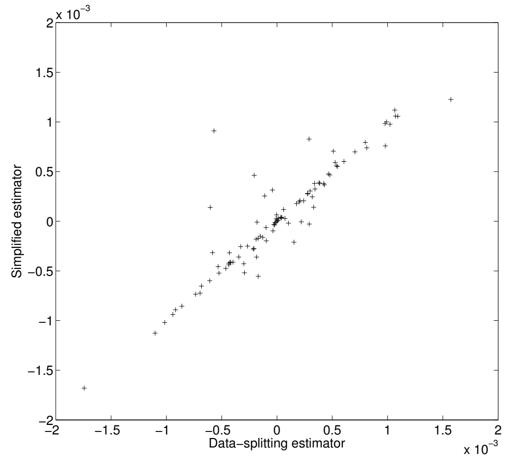
(a)



(b)



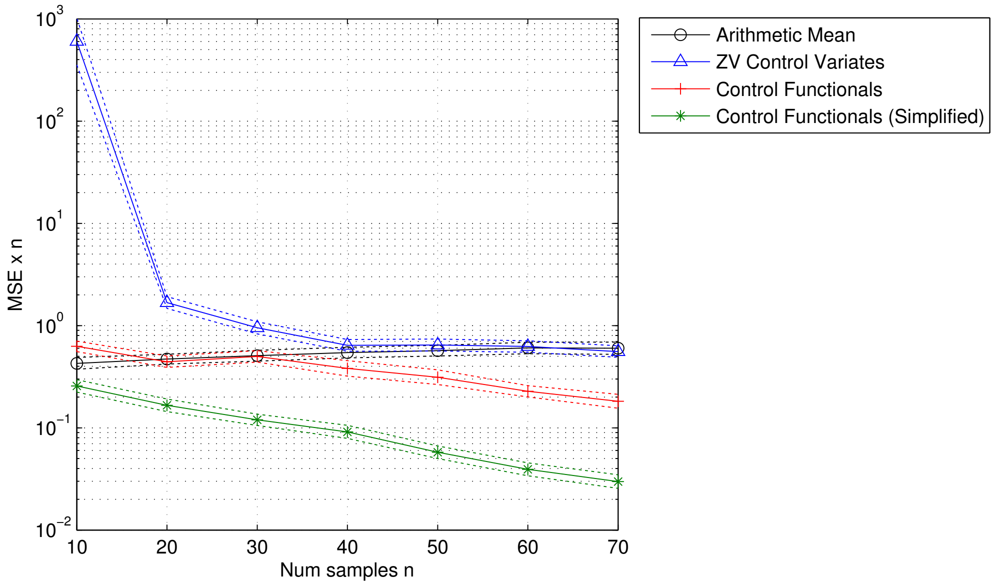
(c)



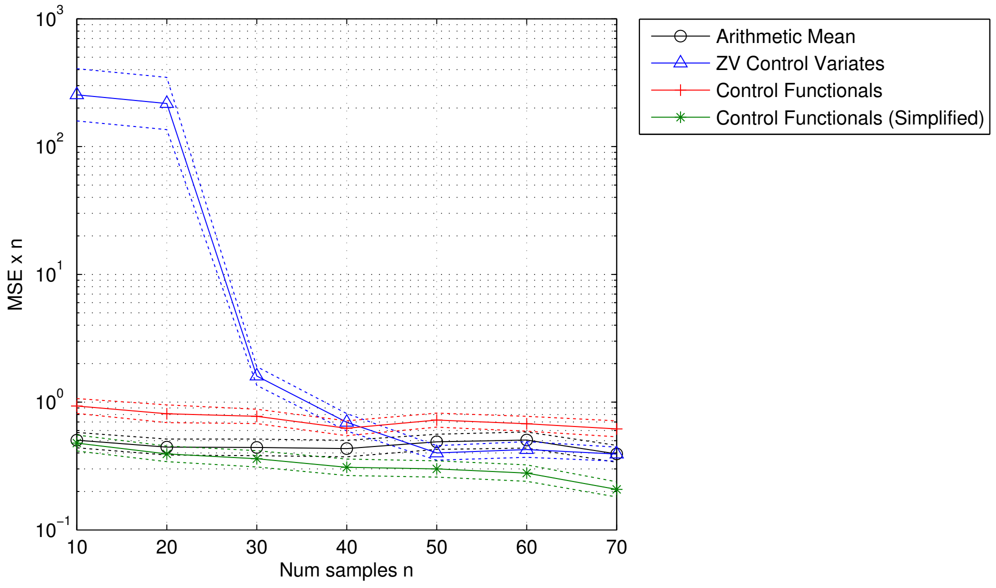
(d)

Figure S1: Illustration on a synthetic integration problem. (a) Determining the kernel length-scale hyper-parameter $\alpha_2 = 1$ by minimising the predictive mean square error (MSE). ($\alpha_1 = 0.1$ was selected similarly, not shown here.) (b) Determining an appropriate ratio m/n of training to test samples, again by consideration of the predictive MSE. (c) Examining the effect of using multiple, randomly selected sample splits. (d) Comparing the sample-splitting estimator with the simplified estimator. [Here the sample-splitting estimator uses just one split of the samples.] In each of (a-d) we considered 100 independent realisations of the sampling process. The number of samples was taken to be $n = 50$ throughout.

.3 Performance in higher dimensions



(a) Relative variance of estimators ($d = 3$)



(b) Relative variance of estimators ($d = 5$)

Figure S2: Illustration on a synthetic integration problem. (continued). Empirical assessment of asymptotic properties. Here we consider examples with dimension (a) $d = 3$, (b) $d = 5$. [Riemann sums were not considered for dimensions $d > 1$ due to the substantial increase in methodological complexity here.]

.4 A larger collection of examples

	Problem		Mean Square Error				Notes
	$f(\boldsymbol{x})$	π	Arithmetic Mean	Riemann Sums	ZV Control Variates	Control Functionals (Simplified)	
CF ✓	$\sin(\pi x)$	$N(0,1)$	0.0098 ± 0.0013	0.0018 ± 0.00043	0.01 ± 0.0013	**$6.9e-07 \pm 3.4e-07$**	
	$\sin((x_1+x_2+x_3)\pi/3)$	$N(\mathbf{0}_{3\times1}, \boldsymbol{I}_{3\times3})$	0.011 ± 0.0016	-	0.015 ± 0.0027	**0.0013 ± 0.00016**	
	$\sin((x_1+\cdots+x_5)\pi/5)$	$N(\mathbf{0}_{5\times1}, \boldsymbol{I}_{5\times5})$	0.0088 ± 0.0015	-	0.01 ± 0.0017	**0.0057 ± 0.00089**	
	$\sin((x_1+\cdots+x_{10})\pi/10)$	$N(\mathbf{0}_{10\times1}, \boldsymbol{I}_{10\times10})$	**0.0081 ± 0.0010**	-	5.2 ± 2.6	**0.0078 ± 0.00099**	(a)
	$\exp(-x_1^2-\cdots-x_{10}^2)$	$N(\mathbf{0}_{10\times1}, \boldsymbol{I}_{10\times10})$	$5.8e-06 \pm 1.3e-06$	-	0.49 ± 0.35	**$3.8e-06 \pm 4.5e-07$**	
	x	$N(0,1)$	0.018 ± 0.0024	0.0054 ± 0.00088	**$2.1e-33 \pm 7.4e-34$**	$1.6e-06 \pm 3.8e-07$	(b)
	x_1	$N(\mathbf{0}_{3\times1}, \boldsymbol{I}_{3\times3})$	0.022 ± 0.0037	-	**$4.8e-33 \pm 1.1e-33$**	0.0036 ± 0.00063	
	x^2	$N(0,1)$	0.059 ± 0.0093	0.047 ± 0.0035	**$3.6e-33 \pm 1.2e-33$**	$5.5e-05 \pm 6.3e-06$	
	x	$\Gamma(5,1)$	0.11 ± 0.016	0.038 ± 0.0055	**$7.1e-32 \pm 2.3e-32$**	0.016 ± 0.0019	
	$\exp(x)$	$N(0,1)$	0.1 ± 0.013	0.04 ± 0.005	0.015 ± 0.0056	**$0.00019 \pm 2.7e-05$**	(c)
	$x^2\sin(\pi x)$	$N(0,1)$	0.028 ± 0.0035	0.012 ± 0.002	0.029 ± 0.0035	**0.00047 ± 0.00013**	
	x	$\beta(2,2)$	0.00076 ± 0.0001	$0.00014 \pm 2.5e-05$	0.00086 ± 0.00016	**$7.2e-11 \pm 2.8e-11$**	
	$\sin(2\pi x)$	$\beta(2,2)$	0.0092 ± 0.001	0.001 ± 0.0002	0.017 ± 0.0036	**$2.4e-06 \pm 1e-06$**	
	x	$0.5N(-1,1)+0.5N(1,1)$	0.043 ± 0.0055	0.0097 ± 0.0014	0.0048 ± 0.00065	**$6.9e-05 \pm 1.1e-05$**	
	x^2	$0.5N(-1,1)+0.5N(1,1)$	0.15 ± 0.026	0.12 ± 0.01	0.0034 ± 0.00053	**$0.00039 \pm 7.7e-05$**	
CF ×	$\sin(\pi x)$	Cauchy$(0,1)$	**0.0095 ± 0.0013**	0.021 ± 0.0032	**0.0093 ± 0.0013**	0.31 ± 0.035	(d)
	x	Exp(1)	0.023 ± 0.0028	**0.0087 ± 0.0013**	0.014 ± 0.0021	5.5 ± 0.42	(e)
	x	Non-differentiable	0.00094 ± 0.00012	$0.00012 \pm 1.7e-05$	0.0083 ± 0.0011	**$5.7e-05 \pm 1.5e-05$**	(f)
	$1_{x>0}$	$N(0,1)$	0.0035 ± 0.00042	**0.0014 ± 0.00019**	0.0022 ± 0.00028	$8e+03 \pm 2.1e+03$	(g)

Table S1: A broad range of examples, some of which are compatible with the CF method (CF ✓) and some of which are not (CF ×). In each case the true mean $\mu(f)$ is known analytically and we report the mean square error computed over 100 independent realisations. [Here the number of samples was fixed at $n = 50$. For all examples we employed (simplified) CF with the hyper-parameters $\alpha_1 = 0.1$, $\alpha_2 = 1$. Riemann sums were not used in the multi-dimensional problems due to their non-trivial implementation.] Notes: (a) ZV control variates perform poorly since their implementation here requires us to estimate a 10×10 covariance matrix from only $n = 50$ samples. On the other hand, CF offers stability here that is comparable with the usual arithmetic mean. (b) For these low-dimensional polynomial examples, ZV control variates are essentially exact. (c) In each of these examples CF offers the most precise estimates. (d) Here $f \notin \mathcal{L}^2(\pi)$, violating our basic assumption. (e) Here $\pi(0)\phi(0)$ is not forced equal to zero, violating (A1). (f) Here π is not differentiable, violating the assumption that the gradient function exists. This "non-differentiable" example uses a triangular distribution on $[0, 1]$. (g) Here f is not differentiable, so in particular $f \notin \mathcal{H}_+$, violating (A5).

.5 A comparison with extensions of ZV control variates

In this section we consider "zero variance" (ZV) control variates in greater detail and ask whether they can be extended, by analogy with our control functional method. Firstly we re-interpret ZV control variates within our general framework: Consider degree J polynomials $\phi_i(\boldsymbol{x})$ of the form

$$\phi_i(\boldsymbol{x}) = a_i + \sum_{j=1}^{d} b_{i,j} x_j + \sum_{j,k=1}^{d} c_{i,j,k} x_j x_k + \sum_{j,k,l=1}^{d} d_{i,j,k,l} x_j x_k x_l + \dots \tag{1}$$

with coefficients $\boldsymbol{\theta}_i = \{a_i, b_{i,j}, c_{i,j,k}, \dots\}$. The polynomial is degree J, so that only the product of at most J of the components x_j can appear on the right hand side of Eqn. 1. The associated control functional is

$$\begin{aligned}
\psi_{\boldsymbol{\phi}}(\boldsymbol{x}) &= \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\phi}(\boldsymbol{x}) \cdot \boldsymbol{u}(\boldsymbol{x}) \\
&= \sum_{i=1}^{d} \Bigg[b_{i,i} + \sum_{j=1}^{d} c_{i,j,i} x_j + \sum_{k=1}^{d} c_{i,i,k} x_k + \sum_{j,k=1}^{d} d_{i,j,k,i} x_j x_k \\
&\qquad + \sum_{j,l=1}^{d} d_{i,j,i,l} x_j x_l + \sum_{k,l=1}^{d} d_{i,i,k,l} x_k x_l + \dots \Bigg] \\
&\quad + \sum_{i=1}^{d} \Bigg[a_i + \sum_{j=1}^{d} b_{i,j} x_j + \sum_{j,k=1}^{d} c_{i,j,k} x_j x_k + \sum_{j,l,k=1}^{d} d_{i,j,k,l,} x_j x_k x_l + \dots \Bigg] u_i(\boldsymbol{x}). \qquad (2)
\end{aligned}$$

This can in turn be re-written as $\boldsymbol{\theta}^T \boldsymbol{m}(\boldsymbol{x})$ where the components $\{a_i, b_{i,j}, c_{i,j,k}, \dots\}$ of $\boldsymbol{\theta}$ and $\boldsymbol{m}(\boldsymbol{x})$ are identified in the inner product as

$$\begin{aligned}
a_i &\leftrightarrow u_i & (3)\\
b_{i,i} &\leftrightarrow 1 + x_i u_i & (4)\\
b_{i,j} &\leftrightarrow x_j u_i \quad (i \neq j) & (5)\\
c_{i,i,i} &\leftrightarrow 2x_i + x_i^2 u_i & (6)\\
c_{i,i,k} &\leftrightarrow x_k + x_i x_k u_i \quad (k \neq i) & (7)\\
c_{i,j,i} &\leftrightarrow x_j + x_i x_j u_i \quad (j \neq i) & (8)\\
c_{i,j,k} &\leftrightarrow x_j x_k u_i \quad (j,k \neq i) & (9)\\
&\vdots
\end{aligned}$$

Thus, for *fixed* polynomial degree J, the ZV estimator converges at super-root-n if and only if $f(\boldsymbol{x}) = c + \psi_{\boldsymbol{\phi}}(\boldsymbol{x})$ for some $c \in \mathbb{R}$ and some $\psi_{\boldsymbol{\phi}}$ of the form in Eqn. 2. Except for very special combinations of f and π, e.g. as detailed in Papamarkou *et al.* (2014), this condition will not be satisfied. For example, if $X \sim N(0,1)$ then the ZV estimator converges at super-root-n if and only if $f(x)$ is a polynomial of degree at most J. As such, super-root-n convergence is not achieved even for "elementary" functions, such as $f(x) = \sin(\pi x)$.

Rescuing ZV control variates: An extension of the ZV method, *not previously proposed in the literature*, would be to take the control functional perspective and let the degree J of the polynomial increase to infinity as the number of samples $n \to \infty$. In this case super-root-n convergence can be achieved if and only if $f(\boldsymbol{x})$ can be approximated by a *sequence* of functions from the class $\{c + \psi_{\boldsymbol{\phi}}(\boldsymbol{x}) : c \in \mathbb{R},\ \psi_{\boldsymbol{\phi}} \text{ as in Eqn. 2},\ J \in \mathbb{N}\}$. Under reasonable regularity assumptions, it follows from the Stone-Weierstrass theorem that this class of functions will be dense in $C^0(\Omega, \mathbb{R})$, so that super-root-n convergence will be realised for all $f \in C^0(\Omega, \mathbb{R})$. In this way the ZV method can in principle be rescued. Below we explore how such an approach could be implemented and compare its performance against the proposed fully non-parametric control functional method.

Choosing the polynomial degree J: We work in the same setting as the main text, starting with a dichotomy $\mathcal{D}_0 \cup \mathcal{D}_1$ of the samples. Based on $\mathcal{D}_0$, our challenge is to select an appropriate polynomial degree $J \equiv J(m)$ and then, conditional on J, to select appropriate values for the polynomial coefficients $\boldsymbol{\theta}$. Selection of the coefficients $\boldsymbol{\theta}$ is standard and recapped in the section below. Given that it is *a priori* unclear how $J(m)$ should scale with m, the number of training samples, the most straight-forward solution is to proceed retrospectively, fitting polynomials of each degree $J \in \mathbb{N}$ and selecting the polynomial that minimises a cross-validation error (computed by partitioning $\mathcal{D}_0$ into $\mathcal{D}_{0,0} \cup \mathcal{D}_{0,1}$, as described in the main text). Since it is impossible to consider all polynomial degrees, a simple strategy is to consider an increasing sequence $J = 1, 2, 3, \dots$ and stop at the point where the cross-validation error for $J = l + 1$ is greater than that for $J = l$; i.e. stop when we begin to "over-fit", in order to maintain low predictive error.

Choosing the polynomial coefficients $\boldsymbol{\theta}$: Following Mira *et al.* (2013) we note that optimal polynomial coefficients $\boldsymbol{\theta}$ are given by

$$\boldsymbol{\theta} = -\mathbb{V}_{\boldsymbol{X}}[\boldsymbol{m}(\boldsymbol{X})]^{-1}\mathbb{E}_{\boldsymbol{X}}[f(\boldsymbol{X})\boldsymbol{m}(\boldsymbol{X})]$$

and so we approximate these coefficients by plugging in the sample estimate

$$\frac{1}{m}\sum_{i=1}^{m} f(\boldsymbol{x}_i)\boldsymbol{m}(\boldsymbol{x}_i)$$

for the expectation $\mathbb{E}_{\boldsymbol{X}}[f(\boldsymbol{X})\boldsymbol{m}(\boldsymbol{X})]$ and

$$\frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{m}(\boldsymbol{x}_i) - \overline{\boldsymbol{m}})(\boldsymbol{m}(\boldsymbol{x}_i) - \overline{\boldsymbol{m}})^T, \qquad \overline{\boldsymbol{m}} = \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{m}(\boldsymbol{x}_j)$$

for the variance $\mathbb{V}_{\boldsymbol{X}}[\boldsymbol{m}(\boldsymbol{X})]$.

Comments on the extended ZV approach: There are several ways in which the extension of ZV control variates described above is less elegant compared to the proposed control functionals: (i) The use of cross-validation to elicit J is *essential* to ensure super-root-n convergence, whereas for control functionals it was merely an optional tool that can be used

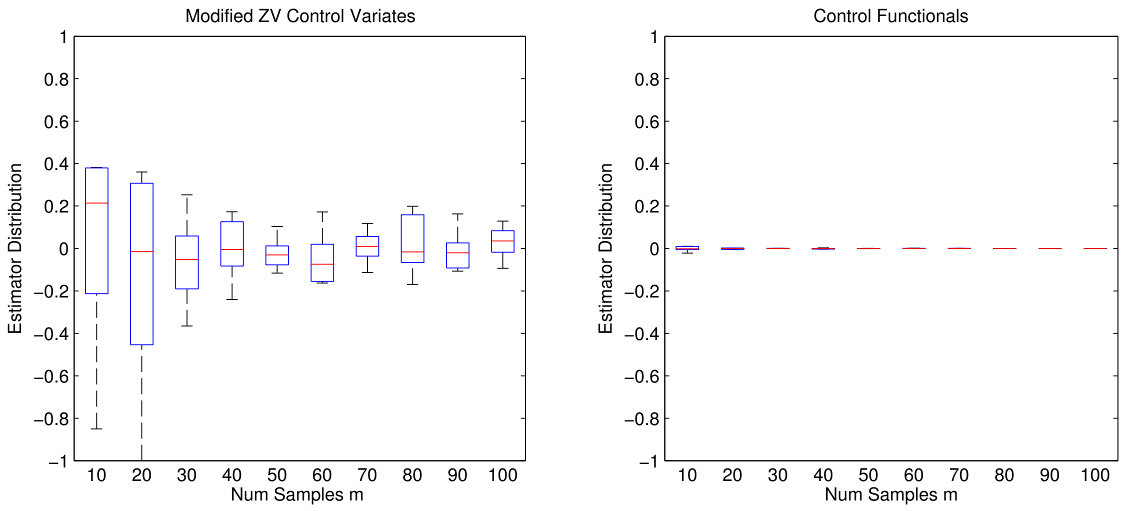


Figure S3: A comparison between modified ZV and control functionals, based on a synthetic example. Here we display the empirical sampling distribution of Monte Carlo estimators, based on n samples and 100 independent realisations.

to improve on a default rate that is already super-root-n. (ii) We are required to fit many different polynomial approximations, indexed by $J \in \mathbb{N}$, whereas control functionals can be implemented based on just one functional approximation. (Computational time here is typically not the bottle-neck in applications, but there is nevertheless a cost associated with increased coding effort. Indeed, coding Eqns. 3-9 in the general case is non-trivial.) (iii) Control functionals are flexible, in the sense that the kernel function k_0 can be chosen to enforce a pre-specified level of smoothness and, in this way, approximation error rates can be optimised. In contrast, ZV control variates are only based on polynomials cannot be made "more or less smooth". This will result in sub-optimal rates for functional approximation in the case of general f, for example when f is very rough then polynomial approximations will converge slowly.

For the remainder we present an empirical comparison of the two approaches. It will be shown that the performance of this extended ZV scheme is far inferior to the control functionals that we are proposing.

Empirical comparison: We implemented the extended ZV control variates as described above and compared their performance to our proposed control functional approach. Our study focuses on the illustrative synthetic example from the main text in dimension $d = 1$; i.e.

$$f(X) = \sin(\pi X), \qquad X \sim N(0, 1),$$

based on $n = 2m$ independent samples from π, split as $\mathcal{D}_0 = \{x_i\}_{i=1}^m$, $\mathcal{D}_1 = \{x_i\}_{i=m+1}^{2m}$. (We chose dimension $d = 1$ to reduce the coding effort relative to the case of general d in Eqns. 3-9; for control functionals there is very little coding effort required for the case of general d.) For experiments we considered $m \in \{10, 20, \ldots, 100\}$. For ZV we performed 10 rounds of

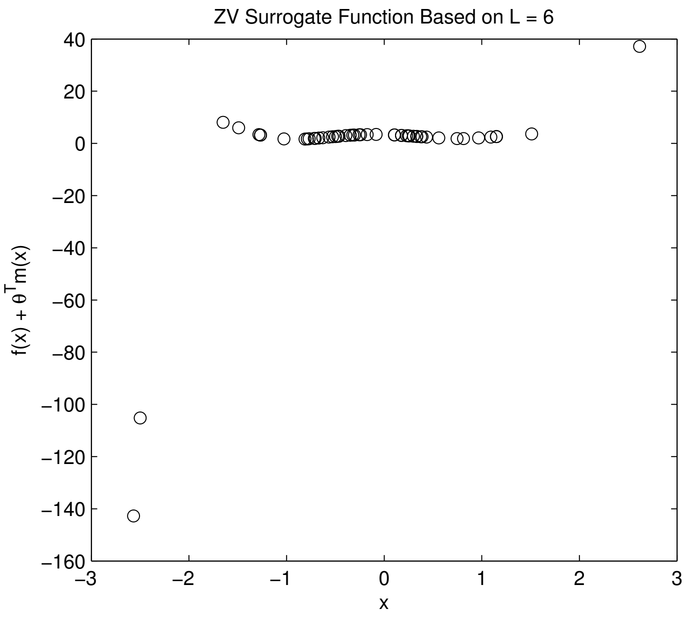


Figure S4: Synthetic example; extrapolation from the fitted ZV model based on a degree $J = 6$ polynomial. Here we evaluate the fitted model at locations sampled from π, minicking the evaluation of the actual estimator.

cross-validation in order to select an appropriate polynomial degree J, in each case splitting the training samples $\mathcal{D}_0$ into two subsets $\mathcal{D}_{0,0}$, $\mathcal{D}_{0,1}$ of equal size. For control functionals we used the single split of the samples defined by $\mathcal{D}_0 \cup \mathcal{D}_1$; this ensures fairness of comparison. All experiments were repeated 10 times and the same datasets were used to assess each estimator.

Results are presented in Fig. S3. Firstly, we note that performance does not seem to improve over standard ZV control variates (cf. results in the main text). Closer investigation revealed that low polynomial degree (e.g. $J = 1, 2, 3$) almost always achieved the best performance in the cross-validation sense and was automatically chosen for estimation. Since the main text results are based on $J = 2$, this explains the observed similarity between the sets of results. Secondly, it is interesting to ask why low-degree polynomials are optimal in this objective, cross-validated sense. Fig. S4 plots the ZV surrogate function $\tilde{f}(x) = f(x) + \boldsymbol{\theta}^T \boldsymbol{m}(x)$, for a higher-degree polynomial $J = 6$, at the values $\mathcal{D}_1$ that will be used for computing the estimate. It is clearly seen that polynomial approximations extrapolate extremely poorly as the degree J of the polynomial increases; in Fig. S4 we see that the surrogate function $\tilde{f}$ "blows up" at values of $x \in \mathcal{D}_1$ that were outside $[-2, 2]$.

We conclude that, whilst polynomial approximations have been shown to be useful in many practical applications (e.g. Mira *et al.*, 2013; Friel *et al.*, 2015; Oates *et al.*, 2016), they are not well-suited to the development of control functionals with super-root-n convergence. The kernel version that we propose is much more stable when it comes to extrapolation of the surrogate function.

.6 Marginalisation in hierarchical models: RQMC

We investigated whether control functionals can confer gains when employed alongside (R)QMC techniques. Since (R)QMC is a sampling method, it is in a sense orthogonal to *post-hoc* techniques and it is interesting to investigate whether a combined approach achieves superior estimation performance. Note, however, that samples from (R)QMC are not independent and thus the theoretical framework that we described does not directly apply - this motivates the empirical investigation below. Specifically, we focus on a RQMC Halton scheme with scrambling that has been shown to achieve $O(n^{-3/2+\epsilon})$ root mean square error for any $\epsilon > 0$ (Owen, 1997). Results in Fig. S5 show that the combined approach of CF + RQMC achieves lower-variance estimation compared to RQMC alone.

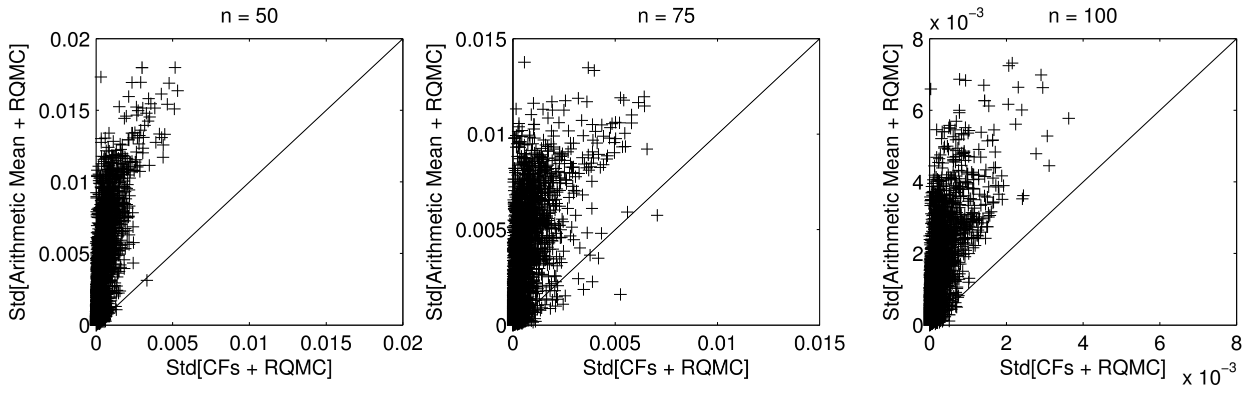


Figure S5: Exploring the use of randomised quasi Monte Carlo (RQMC) with control functionals. [Here we display the sampling standard deviation of Monte Carlo estimators for the posterior predictive mean $\mathbb{E}[Y_*|\boldsymbol{y}]$ in the SARCOS robot arm example, computed over 10 independent realisations. Each point, representing one Monte Carlo integration problem, is represented by a cross.

.7 Non-linear ODE models

We consider first-order non-linear dynamical systems of the form

$$\frac{d\boldsymbol{x}}{ds} = \boldsymbol{F}(\boldsymbol{x}, s; \boldsymbol{\theta}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{10}$$

where $\boldsymbol{x}_0 \in \mathbb{R}^p$ is assumed known, $p < \infty$, and $\boldsymbol{F} : \mathbb{R}^p \times [0, \infty) \to \mathbb{R}^p$ is non-linear. We assume that only a subset of the state variables are observed under noise, so that $\boldsymbol{x} = [\boldsymbol{x}_a, \boldsymbol{x}_b]$ and $\boldsymbol{y}$ is a $d \times n$ matrix of observations of the coordinates $\boldsymbol{x}_a$. Write $s_1 < s_2 < \cdots < s_n$ for the times at which observations are obtained, such that $\boldsymbol{y}(s_j) = \boldsymbol{y}_{\bullet,j}$ where $\boldsymbol{y}_{\bullet,j}$ is the jth column of the data matrix $\boldsymbol{y}$. We consider a Gaussian observation process with likelihood

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{x}_0, \sigma) = \prod_{j=1}^{n} \mathcal{N}(\boldsymbol{y}(s_j)|\boldsymbol{x}_a(s_j; \boldsymbol{\theta}, \boldsymbol{x}_0), \sigma^2 \boldsymbol{I}_{d\times d})$$

where $\boldsymbol{x}_a(s_j; \boldsymbol{\theta}, \boldsymbol{x}_0)$ denotes the deterministic solution of the system in Eqn. 10 and $\sigma > 0$ is assumed known. The gradient function in this case is

$$u_i(\boldsymbol{\theta}) = \nabla_{\theta_i} \log p(\boldsymbol{\theta}) + \frac{t}{\sigma^2} \sum_{j=1}^{n} \boldsymbol{S}^i_{j,1:d}(\boldsymbol{y}(s_j) - \boldsymbol{x}_a(s_j; \boldsymbol{\theta}, \boldsymbol{x}_0)) \tag{11}$$

where $\boldsymbol{S}^i$ is a matrix of sensitivities with entries $S^i_{j,k} = \frac{\partial x_k}{\partial \theta_i}(s_j)$. Note that in Eqn. 11, $\boldsymbol{S}^k_{j,k}$ ranges over indices $1 \leq k \leq d$ corresponding only to the observed variables. In general the sensitivities $\boldsymbol{S}^i$ will be unavailable in closed form, but may be computed numerically by either employing adjoint schemes or by augmenting the system of ordinary differential equations, as

$$\dot{S}^i_{j,k} = \frac{\partial F_k}{\partial \theta_i} + \sum_{l=1}^{p} \frac{\partial F_k}{\partial x_l} S^i_{j,l}$$

where $\frac{\partial x_k}{\partial \theta_i} = 0$ at $s = 0$. Indeed, these sensitivities are already computed when differential-geometric sampling schemes are employed, so that the evaluation of Eqn. 11 incurs negligible additional computational cost (Girolami and Calderhead, 2011; Papamarkou *et al.*, 2014).

The method is illustrated on the van der Pol oscillator (van der Pol, 1926), a non-conservative oscillator with non-linear damping that has classical modelling applications in fields ranging from neuroscience (FitzHugh, 1961) to seismology (Cartwright *et al.*, 1999). Here a position $x(s) \in \mathbb{R}$ evolves in time s according to the second order differential equation

$$\frac{d^2x}{dt^2} - \theta(1 - x^2)\frac{dx}{dt} + x = 0$$

where $\theta \in \mathbb{R}$ is an unknown parameter indicating the non-linearity and the strength of the damping. Letting $x_1 := x$ and $x_2 := dx/dt$ we can formulate the oscillator as the first-order system

$$\boldsymbol{F}(\boldsymbol{x}, s; \theta) = \begin{cases} x_2 \\ \theta(1 - x_1^2)x_2 - x_1 \end{cases}$$

where only the first component x_1 is observed. This system was solved numerically using $\theta = 1$, $\boldsymbol{x}_0 = [0, 2]$. Observations were made once every time unit, up to 10 units, and Gaussian measurement noise of standard deviation $\sigma = 0.1$ was added. A log-normal prior was placed on θ such that $\log(\theta) \sim N(0, 0.25)$.

.8 Pseudocode for control functionals

As a convenient shorthand, write $\boldsymbol{u}_i = \boldsymbol{u}(\boldsymbol{x}_i) = \nabla_{\boldsymbol{x}} \log \pi(\boldsymbol{x}_i)$ and $(\boldsymbol{f})_i = f(\boldsymbol{x}_i)$ for the cached values of the gradient $\boldsymbol{u}(\boldsymbol{x})$ and the integrand $f(\boldsymbol{x})$ respectively. The first pseudocode description that we present is for the simplest formulation of our estimator, identical to BMC based in the space $\mathcal{H}_+$:

Algorithm 1 CF estimator (simplified)

```
Require: $\{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^{d\times 1}$, $\{\boldsymbol{u}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1}$, $\boldsymbol{f} \in \mathbb{R}^{n\times 1}$, $\lambda > 0$
    $(\boldsymbol{K})_{i,j} \leftarrow k_0(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, n$
    $\hat{\mu} \leftarrow (1 + \mathbf{1}^T(\boldsymbol{K} + \lambda n \boldsymbol{I})^{-1}\mathbf{1})^{-1}\mathbf{1}^T(\boldsymbol{K} + \lambda n \boldsymbol{I})^{-1}\boldsymbol{f}$
```

In this simple version of the sample splitting estimator that we consider below, one split S of the samples $\mathcal{D}$ is used, corresponding to $\mathcal{D}_0 = \{\boldsymbol{x}_i\}_{i=1}^m$, $\mathcal{D}_1 = \{\boldsymbol{x}_i\}_{i=m+1}^n$:

Algorithm 2 CF estimator

```
Require: $\{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^{d\times 1}$, $\{\boldsymbol{u}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1}$, $\boldsymbol{f} \in \mathbb{R}^{n\times 1}$, $\lambda > 0$
    $(\boldsymbol{f}_0)_i \leftarrow (\boldsymbol{f})_i$ for $i = 1, \ldots, m$
    $(\boldsymbol{f}_1)_i \leftarrow (\boldsymbol{f})_{m+i}$ for $i = 1, \ldots, n - m$
    $(\boldsymbol{K}_0)_{i,j} \leftarrow k_0(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $i = 1, \ldots, m$ and $j = 1, \ldots, m$
    $(\boldsymbol{K}_{1,0})_{i,j} \leftarrow k_0(\boldsymbol{x}_{m+i}, \boldsymbol{x}_j)$ for $i = 1, \ldots, n - m$ and $j = 1, \ldots, m$
    $\hat{\boldsymbol{f}}_1 \leftarrow \boldsymbol{K}_{1,0}(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\boldsymbol{f}_0 + (1 + \mathbf{1}^T(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\mathbf{1})^{-1}(\mathbf{1} - \boldsymbol{K}_{1,0}(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\mathbf{1})(\mathbf{1}^T(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\boldsymbol{f}_0)$
    $\hat{\mu} \leftarrow (n - m)^{-1}\mathbf{1}^T(\boldsymbol{f}_1 - \hat{\boldsymbol{f}}_1) + (1 + \mathbf{1}^T(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\mathbf{1})^{-1}\mathbf{1}^T(\boldsymbol{K}_0 + \lambda m \boldsymbol{I})^{-1}\boldsymbol{f}_0$
```

Now in this multi-splitting version we consider R different splits S of the samples $\mathcal{D}$, corresponding to $\mathcal{D}_0 = \{\boldsymbol{x}_{S(i)}\}_{i=1}^m$, $\mathcal{D}_1 = \{\boldsymbol{x}_i\}_{i=S^c(i)}^{n-m}$. Here a split S is a subset of $\{1, 2, \ldots, n\}$ containing m distinct elements. The members of S are themselves indexed and we denote this by $S = \{S(1), \ldots, S(m)\}$. The complement S^c is defined to be $\{1, 2, \ldots, n\} \setminus S$ and we again index $S^c = \{S^c(1), \ldots, S^c(n-m)\}$. In addition, $r \in (0, 1)$ is a user-defined constant that we often take to be $r = 1/2$ in practice.

Algorithm 3 CF estimator + multi-splitting

```
Require: $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^{d\times 1}$, $\{\boldsymbol{u}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1}$, $\boldsymbol{f} \in \mathbb{R}^{n\times 1}$, $\lambda > 0$
  $\hat{\mu} \leftarrow 0$
  for r = 1,...,R do
    $S \leftarrow$ random subset of $\{1, \dots, n\}$ of size $m \approx rn$.
    $(\boldsymbol{f}_0)_i \leftarrow (\boldsymbol{f})_{S(i)}$ for $i = 1, \dots, m'$
    $(\boldsymbol{f}_1)_i \leftarrow (\boldsymbol{f})_{S^c(i)}$ for $i = 1, \dots, m - m'$
    $(\boldsymbol{K}_0)_{i,j} \leftarrow k_0(\boldsymbol{x}_{S(i)}, \boldsymbol{x}_{S(j)})$ for $i = 1, \dots, m$ and $j = 1, \dots, m$
    $(\boldsymbol{K}_{1,0})_{i,j} \leftarrow k_0(\boldsymbol{x}_{S^c(i)}, \boldsymbol{x}_{S(j)})$ for $i = 1, \dots, n - m$ and $j = 1, \dots, m$
    $\hat{\boldsymbol{f}}_1 \leftarrow \boldsymbol{K}_{1,0}(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0+(1+\boldsymbol{1}^T(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\boldsymbol{1})^{-1}(\boldsymbol{1}-\boldsymbol{K}_{1,0}(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\boldsymbol{1})(\boldsymbol{1}^T(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0)$
    $\hat{\mu} \leftarrow \hat{\mu} + R^{-1}[(n-m)^{-1}\boldsymbol{1}^T(\boldsymbol{f}_1 - \hat{\boldsymbol{f}}_1) + (1 + \boldsymbol{1}^T(\boldsymbol{K}_0 + \lambda m\boldsymbol{I})^{-1}\boldsymbol{1})^{-1}\boldsymbol{1}^T(\boldsymbol{K}_0 + \lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0]$
  end for
```

Finally, in this most general version of the algorithm, samples $\mathcal{D}$ are partitioned *a priori* as $\mathcal{D}_0 \cup \mathcal{D}_1$ and we are interested in optimising over parameters $\boldsymbol{\theta}_i \in \Theta$ that specify the base kernel $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$. We consider T different candidate values for these parameters and use cross-validation to select an optimal set, indexed by $t^* \in \{1, 2, \dots, T\}$:

Algorithm 4 CF estimator + multi-splitting + cross-validation

```
Require: $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^{d\times 1}$, $\{\boldsymbol{u}_i\}_{i=1}^n \in \mathbb{R}^{d\times 1}$, $\boldsymbol{f}_0 \in \mathbb{R}^{m\times 1}$, $\boldsymbol{f}_1 \in \mathbb{R}^{n-m\times 1}$, $\{\boldsymbol{\theta}_i\}_{i=1}^T \subset \Theta$,
  $\lambda > 0$
  $\hat{\mu} \leftarrow 0$
  for r = 1,...,R do
    $S_0 \leftarrow$ random subset of $\{1,\dots,n\}$ of size $m \approx rn$.
    $(\boldsymbol{f}_0)_i \leftarrow (\boldsymbol{f})_{S_0(i)}$ for $i = 1,\dots,m'$
    $(\boldsymbol{f}_1)_i \leftarrow (\boldsymbol{f})_{S_0^c(i)}$ for $i = 1,\dots,m-m'$
    for t = 1,...,T do
      $S \leftarrow$ random subset of $\{1,\dots,m\}$ of size $m' \approx rm$.
      $(\boldsymbol{f}_{0,0})_i \leftarrow (\boldsymbol{f}_0)_{S(i)}$ for $i = 1,\dots,m'$
      $(\boldsymbol{f}_{0,1})_i \leftarrow (\boldsymbol{f}_0)_{S^c(i)}$ for $i = 1,\dots,m-m'$
      $(\boldsymbol{K}_{0,0})_{i,j} \leftarrow k_0(\boldsymbol{x}_{S(i)}, \boldsymbol{x}_{S(j)}; \boldsymbol{\theta}_t)$ for $i = 1,\dots,m'$ and $j = 1,\dots,m'$
      $(\boldsymbol{K}_{0,1,0})_{i,j} \leftarrow k_0(\boldsymbol{x}_{S^c(i)}, \boldsymbol{x}_{S(j)}; \boldsymbol{\theta}_t)$ for $i = 1,\dots,m-m'$ and $j = 1,\dots,m'$
      $\hat{\boldsymbol{f}}_{0,1} \leftarrow \boldsymbol{K}_{0,1,0}(\boldsymbol{K}_{0,0} + \lambda m'\boldsymbol{I})^{-1}\boldsymbol{f}_{0,0} + (1 + \mathbf{1}^T(\boldsymbol{K}_{0,0} + \lambda m'\boldsymbol{I})^{-1}\mathbf{1})^{-1}(\mathbf{1} - \boldsymbol{K}_{0,1,0}(\boldsymbol{K}_{0,0} +$
      $\lambda m'\boldsymbol{I})^{-1}\mathbf{1})(\mathbf{1}^T(\boldsymbol{K}_{0,0} + \lambda m'\boldsymbol{I})^{-1}\boldsymbol{f}_{0,0})$
      error$(t) \leftarrow$ error$(t)$ $+\|\boldsymbol{f}_{0,1} - \hat{\boldsymbol{f}}_{0,1}\|_2$
    end for
    $t^* \leftarrow \arg\min_{t=1,\dots,T}$ error$(t)$
    $(\boldsymbol{K}_0)_{i,j} \leftarrow k_0(\boldsymbol{x}_{S_0(i)}, \boldsymbol{x}_{S_0(j)}; \boldsymbol{\theta}_{t^*})$ for $i = 1,\dots,m$ and $j = 1,\dots,m$
    $(\boldsymbol{K}_{1,0})_{i,j} \leftarrow k_0(\boldsymbol{x}_{S_0^c(i)}, \boldsymbol{x}_{S_0(j)}; \boldsymbol{\theta}_{t^*})$ for $i = 1,\dots,n-m$ and $j = 1,\dots,m$
    $\hat{\boldsymbol{f}}_1 \leftarrow \boldsymbol{K}_{1,0}(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0+(1+\mathbf{1}^T(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\mathbf{1})^{-1}(\mathbf{1}-\boldsymbol{K}_{1,0}(\boldsymbol{K}_0+\lambda m\boldsymbol{I})^{-1}\mathbf{1})(\mathbf{1}^T(\boldsymbol{K}_0+$
    $\lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0)$
    $\hat{\mu} \leftarrow \hat{\mu} + R^{-1}[(n-m)^{-1}\mathbf{1}^T(\boldsymbol{f}_1 - \hat{\boldsymbol{f}}_1) + (1 + \mathbf{1}^T(\boldsymbol{K}_0 + \lambda m\boldsymbol{I})^{-1}\mathbf{1})^{-1}\mathbf{1}^T(\boldsymbol{K}_0 + \lambda m\boldsymbol{I})^{-1}\boldsymbol{f}_0]$
  end for
```

References

Cartwright, J., Eguiluz, V., Hernandez-Garcia, E. and Piro, O. (1999) Dynamics of elastic excitable media. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, **9**, 2197-2202.

FitzHugh, R. (1961) Impulses and physiological states in theoretical models of nerve membranes. *Biophysics J.*, **1**, 445-466.

Friel, N., Mira, A. and Oates, C. J. (2015) Exploiting Multi-Core Architectures for Reduced-Variance Estimation with Intractable Likelihoods. *Baysian Anal.*, **11**(1), 215-245.

Girolami, M. and Calderhead, B. (2011) Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *J. R. Statist. Soc. B*, **73**, 1-37.

Mira, A., Solgi, R. and Imparato, D. (2013) Zero Variance Markov Chain Monte Carlo for Bayesian Estimators. *Stat. Comput.*, **23**, 653-662.

Oates, C. J., Papamarkou, T. and Girolami, M. (2016) The Controlled Thermodynamic Integral for Bayesian Model Evidence Evaluation. *J. Am. Stat. Assoc.*, to appear.

Owen, A. B. (1997) Scramble net variance for integrals of smooth functions. *Ann. Stat.*, **25**, 1541-1562.

Papamarkou, T., Mira, A. and Girolami, M. (2014) Zero Variance Differential Geometric Markov Chain Monte Carlo Algorithms. *Bayesian Anal.*, **9**, 97-128.

van der Pol, B. (1926) On relaxation-oscillations. *The London, Edinburgh and Dublin Phil. Mag. & J. of Sci.*, **2**, 978-992.