Derivatives of GPs

1 Premise

This example comes from Brynjarsdóttir and O'Hagan (2014). It is intended to demonstrate the importance of incorporating model discrepancy when attempting to learn about model parameters through calibration.

A real-life process is given by the equation

$$\zeta(x) = \frac{\theta x}{1 + \frac{x}{a}}, \tag{1}$$

with $\theta = 0.65$ and $a = 20$, and x a one-dimensional *control input*.

2 Parameter estimation using observations (calibration)

In attempting to model the real life process given in Equation (1), we build a simulator, denoted $\eta(x, \theta)$, of the form

$$\eta(x_i, \theta) = \theta x_i. \tag{2}$$

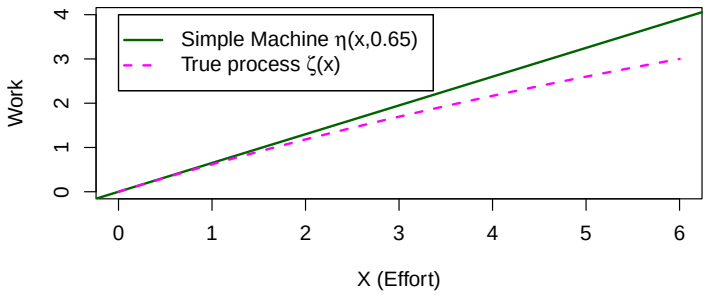


Figure 1: The real-life process and our simulator for it, plotted over control inputs from 0.00 to 6.00.

This model, with $\hat{\theta}$ set equal to the true value of θ, is plotted alongisde the equation representing the real-life process in Figure 1. A word on this straight-line model: Figure 1 shows us that in order for this model (green) to fit well any data produced by the real-life process (pink), the estimated value of θ, which corresponds to the gradient of the green line would need to be lower that the true value of 0.65.

Usually we would take observations of the real-life process $\zeta(x)$ in order to allow us to estimate θ, but in this demonstrative example we simulate observations instead. We simulate three samples of observations at, respectively, 11, 31 and 61 values of x spaced evenly over the interval $[0.2, 4]$. When taking measurements in practice, you will incur a measurement error, and for each of our n simulated observations we generate a

measurement error ϵ from a $N(0, 0.01^2)$ distribution and add this on. In other words, the ith observation, z_i, $i = 1, \ldots, n$, is generated by

$$z_i = \frac{0.65x_i}{1 + \frac{x_i}{20}} + \epsilon_i, \qquad \epsilon_i \sim N(0, 0.01^2).$$

The observations produced are shown in Figure 2.

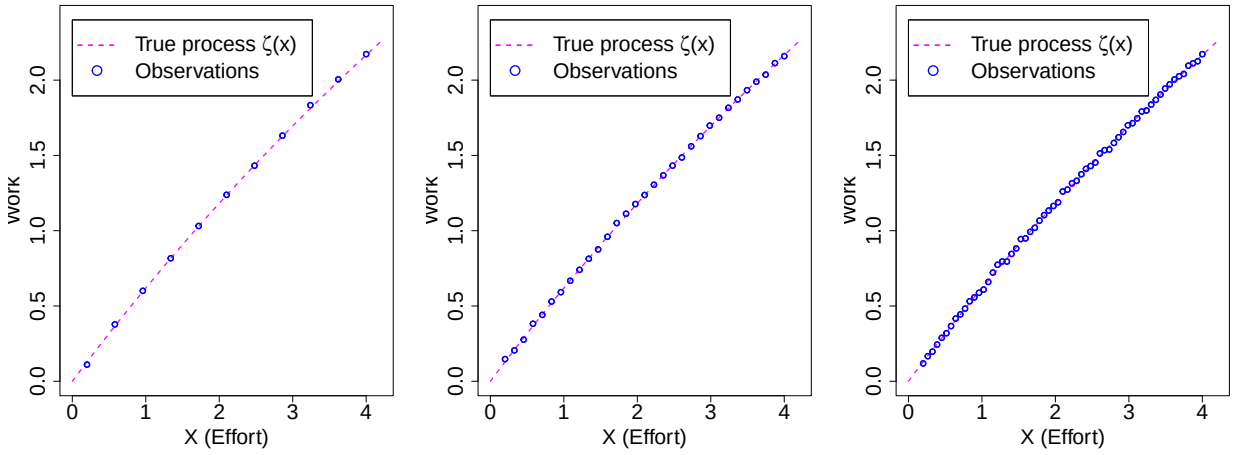


Figure 2: The three samples of observations, of sizes 11, 31 and 61.

We now attempt calibration using the observations to estimate the unknown parameter θ, first without considering model discrepancy and secondly with.

2.1 Not incorporating model discrepancy

By not taking into account model discrepancy, we are assuming that $\eta = \zeta$, in essence that our model is correct and that the only error involved is the measurement error. Thus we assume that the ith observation, z_i, is generated by

$$z_i = \theta x_i + \epsilon_i, \qquad i = 1, \ldots, n,$$

with $\epsilon_i \sim N(0, \sigma_e^2)$, σ_e^2 unknown. Therefore,

$$z_i \mid \theta, \sigma_e^2 \sim N(\theta x_i, \sigma_e^2)$$

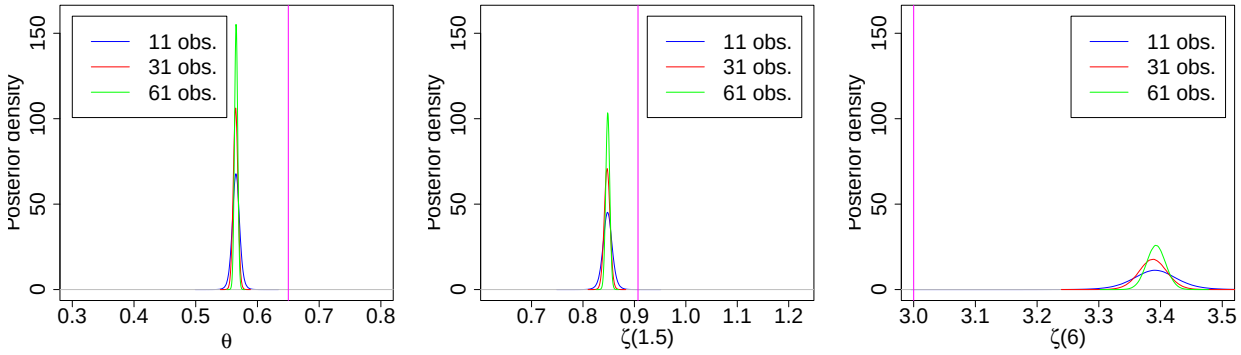
(where θ and σ_e^2 need to be stated 'given', since these are treated as random / unknown, but the x_is don't need such a statement, since we're not assuming these to be treated as random, and are instead known/set from the outset), and the likelihood for the data is

$$\begin{aligned}
\mathcal{L}(\theta, \sigma_e^2 \mid \mathbf{z}) &= \prod_{i=1}^{n} z_i \\
&= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_e} \exp\left\{ -\frac{1}{2\sigma_e^2} \left(z_i - \theta x_i\right)^2 \right\} \\
&\propto \frac{1}{\sigma_e} \exp\left\{ -\frac{1}{2\sigma_e^2} \sum_{i=1}^{n} \left(z_i - \theta x_i\right)^2 \right\}. \qquad (3)
\end{aligned}$$

A joint improper prior for θ and σ_ϵ^2 of $p(\theta, \sigma_\epsilon^2) \propto \sigma_\epsilon^{-2}$ results in the analytically-derivable (marginal) posterior for θ

$$\mathrm{St}_{n-p-1}\left(\theta \,\middle|\, \hat{\theta} = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{z},\ S^2 = \frac{1}{n-p-1}\frac{(\mathbf{z}-\mathbf{x}\hat{\theta})^T(\mathbf{z}-\mathbf{x}\hat{\theta})}{(\mathbf{x}^T\mathbf{x})},\ \nu = n-p-1\right) \tag{4}$$

where $\mathbf{x}$ and $\mathbf{z}$ are column vectors of length n containing, respectively, the control inputs and simulated observations, and $p = 0$. For the three different samples, posterior distributions[1] are shown in Figure 3(a).



(a) (b) (c)

Figure 3: Posterior densities, for each of the three samples, of: (a) θ, (b) $\zeta(1.5)$ (interpolation with the simulator), and (c) $\zeta(6)$ (extrapolation). True values are indicated by vertical lines.

90% posterior credible intervals for θ for the three samples are:

- 11 observations: 0.565 (0.555, 0.576), (0.021)
- 31 observations: 0.565 (0.558, 0.571), (0.013)
- 61 observations: 0.565 (0.561, 0.57), (0.009).

The small value chosen for the variance of the measurement errors ($\sigma_\epsilon^2 = 0.01$) results in very narrow posterior intervals, but this is not the reason that they all fail to capture the true value of the parameter. As mentioned earlier, the form of the model results in an underestimation of θ, as we attempt to fit, to the data, a straight line model through the origin (Equation (2)) with gradient given by the posterior parameter point estimate for θ (0.565, 0.565 and 0.565 for the samples of size 11, 31 and 61 respectively) – see Figure 4.

[1] See https://stats.stackexchange.com/questions/567944/how-can-i-sample-from-a-shifted-and-scaled-student-t-distribution-with-a-specifi.

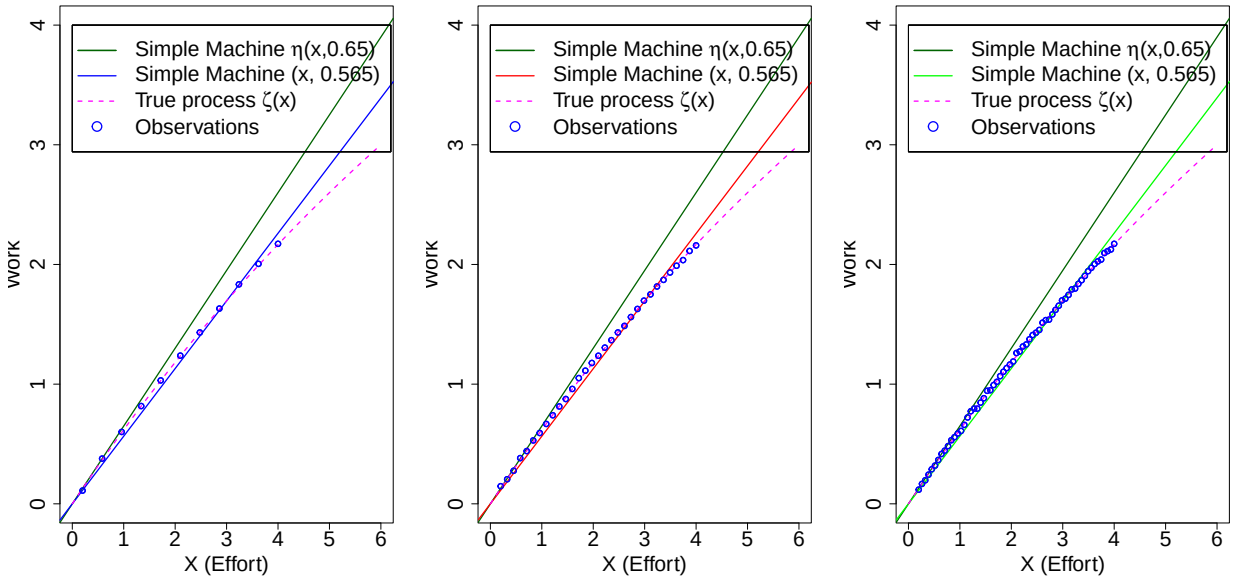


Figure 4: Our simulator, given by Equation (2), assumes a straight line model passing through the origin. When fitting this to the data we get the blue, red and light green lines respectively for the samples of size 11, 31 and 61. In each case, we underestimate θ – the dark green line shows this model but using the true value of θ.

Allowing for a larger variance of measurement errors will lead to wider posterior credible intervals for θ, but taking more observations narrows them again – no matter the size of the measure error variance, there will be a value of n above which the posterior credible interval no longer captures θ. In other words, increasing n narrows the posterior credible interval around a wrong value of θ – this bias cannot be corrected by taking more observations, and is due to a source of uncertainty we have not factored in: **model discrepancy**.

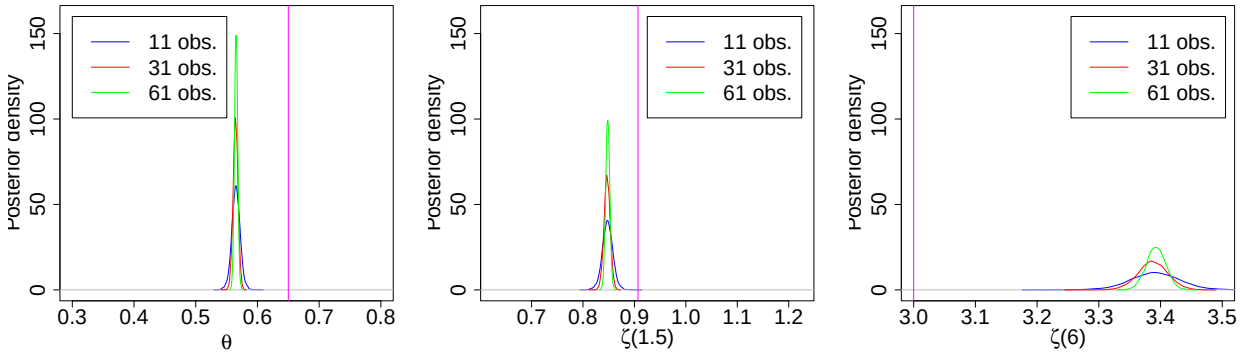
When we use the model in Equation (2) to predict the value of the real life process at $x_0 = 1.5$ (interpolation) and at $x_0 = 6$ (extrapolation), we see Figures 3 (b) and (c). Once again, the more data we collect, the more sure we become about the wrong value, with 90% posterior credible intervals failing to capture the true value for all three samples.

Figures 3 (a) – (c), achieved by using the analytically derived posterior for θ, is replicated using `rstan` instead[2] – see the code below and Figure 5.

```
data {
  int n ;
  vector[n] z ; // observations
  vector[n] x ; // control input
}
parameters {
  real theta;
  real log_sigma;
}
transformed parameters {
  real sigma = exp(log_sigma);
}
```

[2]Using https://stats.stackexchange.com/questions/535955/how-do-i-write-the-jeffreys-prior-for-error-variances-in-stan-p-mu-sigma-1.

```
model {
  z ~ normal(theta * x, sigma);
}
generated quantities {
  real ModInt ;
  real ModExt ;
  ModInt = theta * 1.5 ;
  ModExt = theta * 6 ;
}
```



(a) (b) (c)

Figure 5: Posterior densities, arrived at using rstan, for each of the three samples, of: (a) θ, (b) $\zeta(1.5)$ (interpolation with the simulator), and (c) $\zeta(6)$ (extrapolation). True values are indicated by vertical lines.

2.2 Incorporating model discrepancy using a zero-mean GP

We now acknowledge that $\eta \neq \zeta$ but instead $\eta = \zeta + \delta$, where δ is the model discrepancy. We are thus now assuming that the ith observation, z_i, is generated by

$$z_i = \theta x_i + \delta(x_i) + \epsilon_i, \qquad i = 1, \ldots, n,$$

where

- θx_i is not a r.v. given θ and x_i
- $\delta(x_i)$ is a r.v. given x_i and σ^2, with $\delta() \sim \text{GP}\left(m(), \sigma^2 c(.,.)\right)$, where
  - $m() = 0$
  - $c(x, x') = \exp\left\{-\frac{1}{2\Psi}(x - x')^2\right\}$
- ϵ_i is a r.v. given σ_e^2, with $\epsilon_i \overset{iid}{\sim} N(0, \sigma_\epsilon^2)$.

We therefore have, for $\mathbf{z} = (z_1, \ldots, z_n)^T$,

$$\mathbf{z} \mid \theta, \sigma_e^2, \sigma^2, \Psi \sim \text{MVN}\left(\mathbf{m} = \begin{bmatrix} \theta x_1 \\ \vdots \\ \theta x_n \end{bmatrix}, A = \begin{bmatrix} \sigma^2 + \sigma_e^2 & \ldots & \sigma^2 c(x_n, x_1) \\ \vdots & \ddots & \vdots \\ \sigma^2 c(x_1, x_n) & \ldots & \sigma^2 + \sigma_e^2 \end{bmatrix}\right).$$

Since the p.d.f. of this MVN distribution is

$$f(\mathbf{z}) = \frac{1}{(2\pi)^{\frac{n}{2}}} |A|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{z} - \mathbf{m})^T A^{-1} (\mathbf{z} - \mathbf{m})\right\},$$

we have the following likelihood:

$$\mathcal{L}(\theta, \sigma_e^2, \sigma^2, \Psi \mid \mathbf{z}) \propto |A|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{z} - \mathbf{m})^T A^{-1} (\mathbf{z} - \mathbf{m})\right\}.$$

Zero-mean Gaussian processes

This uses the approach from Kennedy and O'Hagan (2001), whereby the model discrepancy term $\delta(x)$ is modelled as a zero-mean Gaussian process (GP), a probability distribution for a function with form:

$$\delta(\,\cdot\,) \sim \mathrm{GP}\left(0, \sigma^2 c(\,\cdot\,,\,\cdot\,|\Psi)\right),$$

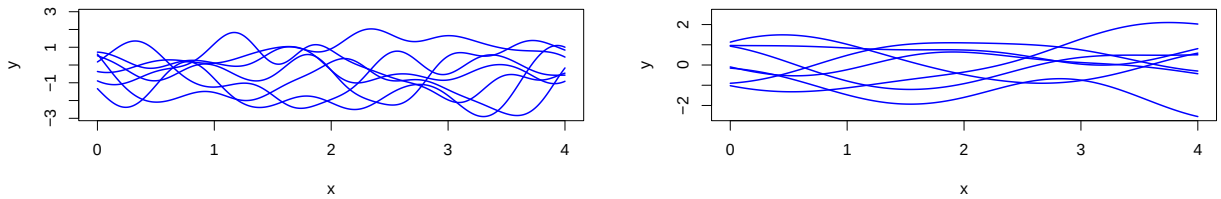
where c is the squared exponential (or Gaussian) correlation function and is given by:

$$c(x_1, x_2|\Psi) = \exp\left(-\left(\frac{x_1 - x_2}{\Psi}\right)^2\right).$$

How does a zero-mean GP representation allow for formulation of prior knowledge about the discrepancy function δ? At any point x the prior probability distribution of $\delta(x)$ is, by Equation (6), normal with:

- mean zero - i.e. we do not have a prior expectation that $\delta(x)$ is more likely to be positive or more likely to be negative.
- $\sigma^2 c(x, x \mid \Psi)$:
  - $\sigma^2 c(x, x \mid \Psi) = \sigma^2 \exp\left(-\left(\frac{x-x}{\Psi}\right)^2\right) = \sigma^2$, so the variance is the same for all x - i.e. we do not have a prior expectation that $|\delta(x)|$ is likely to take larger values for some x values than for others.
  - σ^2 expresses a prior belief that $\delta(x)$ is not likely to be outside the range $\pm 2\sigma$, so it measures the strength of prior information about $\delta(x)$.
- $\sigma^2 c(x_1, x_2 \mid \Psi)$:
  - The correlation function (8) expresses a prior belief that $\delta(x)$ will be a smooth function, with the value of $\delta(x_1)$ being close to that of $\delta(x_2)$ if x_1 is close to x_2. Decreasing Ψ decreases the variance of δ for two points x_1 and x_2 – as δ moves around $\mathbf{x}$ it needs to perform tight turns, since it is not allowed to vary much as it moves away from x_1 and towards x_2.

Figure 6 shows seven draws from the GP prior for δ, with $\sigma^2 = 1$ and (a) $\Psi = 0.3$, (b) $\Psi = 1$, without using `rstan`[3]:



(a) (b)

Figure 6: Seven draws from the zero-mean GP prior for $\delta(x)$, with (a) $\Psi = 0.3$ and (b) $\Psi = 1$.

with the following code:

[3]Using https://www.r-bloggers.com/2019/07/sampling-paths-from-a-gaussian-process.

```r
library(MASS)

x <- seq(0, 4, length.out = 401)  # x-coordinates
N <- 7  # no. of draws

se_kernel <- function(x, y, sigma = 1, length = 1) {
    sigma^2 * exp(- ((x - y) / (sqrt(2) * length))^2)
}

# generate covariance matrix for points in `x` using given kernel function
cov_matrix <- function(x, kernel_fn, ...) {
    outer(x, x, function(a, b) kernel_fn(a, b, ...))
}

# given x coordinates, take N draws from kernel function at those points
draw_samples <- function(x, N, kernel_fn, ...) {
    Y <- matrix(NA, nrow = length(x), ncol = N)
    for (n in 1:N) {
        K <- cov_matrix(x, kernel_fn, ...)
        Y[, n] <- mvrnorm(1, mu = rep(0, times = length(x)), Sigma = * K)
    }
    Y
}
```

If we instead do this in `rstan`[4], we must first run the following `stan` block. Note that we add a small multiple of the identity matrix to the covariance matrix (here set to a value of 0.001) for computational stability when inverting the covariance matrix, particularly a problem if two or more of the input values are "close together" in input space[5] – see Rasmussen and Williams (2006) and Figure 7 below from Jeremy.

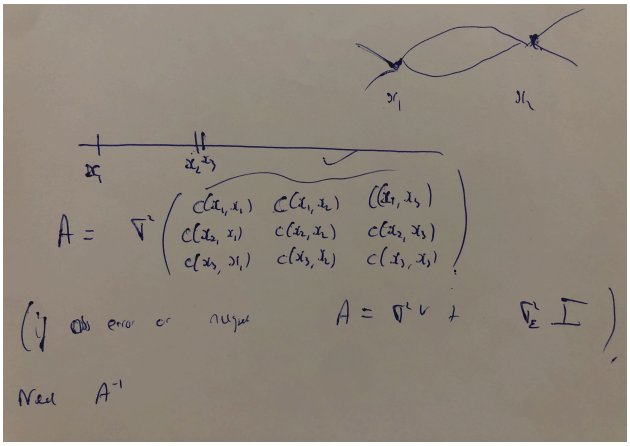


Figure 7: Why we need to add a small multiple of the identity matrix to the covaraince matrix (JO).

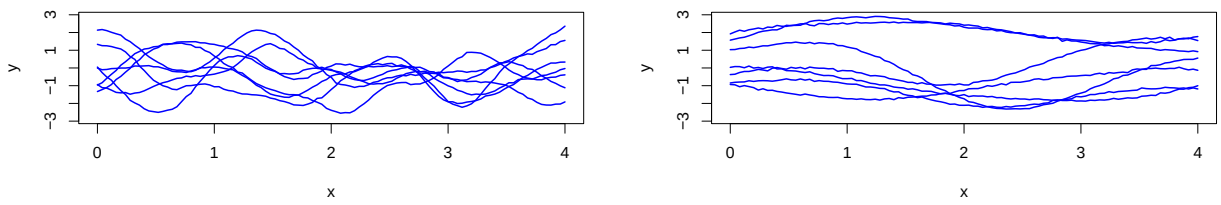
[4]Using https://mc-stan.org/docs/2_19/stan-users-guide/simulating-from-a-gaussian-process.html.

[5]Note that a small value is sometimes added to the diagonal of a correlation matrix when building an emulator but for a different reason, and in this case is referred to as the *nugget term*: it represents added noise, allowing for variability at the training data points. If you have a deterministic model, when you repeat a simulation you should get the same output and so have no noise at the training data. However, with some models (for example, a cloud simulation model) they can have a stochastic element to them, and running at the same settings can give slightly different outputs. Adding the nugget term means that the mean function of the fitted posterior emulator doesn't have to go directly through the training points, and this can lead to a smoother fit.

```
data {
  int<lower=1> n;
  array[n] real x;
  real l;
  real ident_mult;
}
transformed data {
  matrix[n, n] K;
  vector[n] mu = rep_vector(0, n);
  for (i in 1:(n - 1)) {
    K[i, i] = 1 + ident_mult;
    for (j in (i + 1):n) {
      K[i, j] = exp(-0.5 * square((x[i] - x[j]) / l));
      K[j, i] = K[i, j];
    }
  }
  K[n, n] = 1 + ident_mult;
}
parameters {
  vector[n] y;
}
model {
  y ~ multi_normal(mu, K);
}
```

```
xs <- 2
x <- seq(0, 4, length.out = xs)  # x-coordinates
k <- matrix(NA, nrow=n, ncol=n)
for (i in 1:(n - 1)) {
    k[i, i] = 1;
    for (j in (i + 1):n) {
      k[i, j] = exp(-0.5 * ((x[i] - x[j])^2 / l));
      k[j, i] = k[i, j]
    }
  }
k[n, n] = 1
k
#mvrnorm(7, rep(0, xs), k)
det(k)
```

then produce posterior draws for $\delta(x)$ at a given number of equally-spaced points within $x \in [0, 4]$. Given 10,000 iterations, we take the final seven (to match with the plots in Figure 6) and produce line plots. In order to make the plots smooth, the number of equally-spaced points is set to 101 (see Figure 7).



(a) (b)

Figure 8: Seven draws from the zero-mean GP prior, using rstan, for $\delta(x)$, with (a) $\Psi = 0.3$ and (b) $\Psi = 1$.

Priors for $\theta, \sigma_e^2, \sigma^2$ and Ψ are as follows:

- $p(\theta) \propto 1$,
- $p(\Psi) \sim \text{Ga}(5, 5)$,
- $p(\sigma_e^2) \sim \text{IGa}(9.526, 0.0008526)$,
- $p(\sigma^2) \sim \text{IGa}(2.6, 0.144)$,

with the prior distribution for Ψ truncated at $\Psi = 4$ due to numerical problems in the MCMC algorithm. The following `rstan` code chunk assigns these priors and defines the model:

```
data {
  int<lower=1> n;
  vector[n] x;
  vector[n] z;
  real x_array[n];
}
parameters {
  real<lower=0, upper=4> Psi;
  real<lower=0> sq_sigma;
  real<lower=0> sq_sigma_e;
  real theta;
}
model {
  matrix[n, n] K = cov_exp_quad(x_array, sqrt(sq_sigma), Psi);

  // diagonal elements
  for (i in 1:n) {
    K[i, i] = K[i, i] + sq_sigma_e;
  }

  Psi ~ gamma(5, 5);
  sq_sigma_e ~ inv_gamma(9.526, 0.0008526);
  sq_sigma ~ inv_gamma(2.6, 0.144);

  z ~ multi_normal(multiply(theta, x), K);
}
generated quantities {
  real ModInt ;
  real ModExt ;
```

```
  ModInt = theta * 1.5 ;
  ModExt = theta * 6 ;
}
```

```
fit_md
```

```
## Inference for Stan model: anon_model.
## 2 chains, each with iter=10000; warmup=5000; thin=1;
## post-warmup draws per chain=5000, total post-warmup draws=10000.
##
##              mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## Psi          1.65    0.01 0.48   0.81   1.32   1.59   1.93   2.73  6387    1
## sq_sigma     0.05    0.00 0.04   0.02   0.03   0.04   0.06   0.14  4522    1
## sq_sigma_e   0.00    0.00 0.00   0.00   0.00   0.00   0.00   0.00  6738    1
## theta        0.55    0.00 0.05   0.45   0.52   0.55   0.58   0.65  6058    1
## ModInt       0.82    0.00 0.07   0.68   0.78   0.82   0.87   0.97  6058    1
## ModExt       3.29    0.00 0.29   2.72   3.10   3.28   3.47   3.88  6058    1
## lp__       107.19    0.02 1.53 103.42 106.40 107.50 108.33 109.17  4142    1
##
## Samples were drawn using NUTS(diag_e) at Thu Jul 04 10:28:46 2024.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Posterior means and 90% posterior credible intervals, along with the width of these intervals, for θ are:

- 11 observations: 0.548 (0.47, 0.629), (0.16),
- 31 observations: 0.535 (0.461, 0.608), (0.15),
- 61 observations: 0.535 (0.462, 0.608), (0.15).

Repeating this a few times shows agreement with Brynjarsdóttir and O'Hagan (2014) in two ways:

- the intervals are centred in approximately the same place and with approximately the same width,
- the interval width does not tend to zero as the number of observations increases.

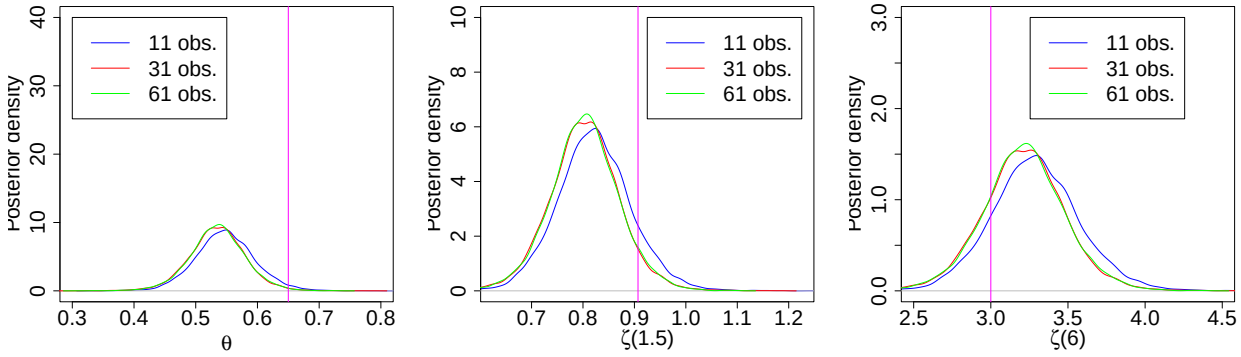
The three intervals have centres below those obtained when not incorporating model discrepancy, which were already biased downwards and, despite being wider, still fail to capture the true value (0.65).

Posterior means and 90% posterior credible intervals for the three other parameters are shown in Table 1.

Table 1: Posterior means and 90% credible intervals for Psi , sigma and sigma_e for different cases.

	Psi	sigma	simga_e
11 obs.	1.65 (0.93, 2.54)	0.22 (0.14, 0.34)	0.0109 (0.0085, 0.0139)
31 obs.	1.95 (1.24, 2.75)	0.21 (0.14, 0.31)	0.011 (0.0092, 0.0131)
61 obs.	1.96 (1.33, 2.8)	0.21 (0.14, 0.32)	0.0099 (0.0086, 0.0113)

Posterior plots for θ, $\zeta(1.5)$ and $\zeta(6)$ are shown in Figure 8 but only the first of these matches Brynjarsdóttir and O'Hagan (2014).



(a) (b) (c)

Figure 9: Posterior densities, arrived at using rstan, for each of the three samples, of: (a) θ, (b) $\zeta(1.5)$ (interpolation with the simulator), and (c) $\zeta(6)$ (extrapolation). True values are indicated by vertical lines.

Citations

Brynjarsdóttir, Jenný, and Anthony O'Hagan. 2014. "Learning about Physical Parameters: The Importance of Model Discrepancy." *Inverse Problems* 30 (11): 114007. https://doi.org/10.1088/0266-5611/30/11/114007.

Kennedy, Marc C., and Anthony O'Hagan. 2001. "Bayesian Calibration of Computer Models." *Journal of the Royal Statistical Society Series B: Statistical Methodology* 63 (3): 425–64. https://doi.org/10.1111/1467-9868.00294.

Rasmussen, Carl E., and Christopher K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press.