Data-Centric Engineering (2024), 5: e48
doi:10.1017/dce.2024.43

RESEARCH ARTICLE

Meta-models for transfer learning in source localization

Lawrence A. Bull[1,2], Matthew R. Jones[3], Elizabeth J. Cross[3], Andrew Duncan[4,5] and Mark Girolami[2,5]

[1]School of Mathematics and Statistics, University of Glasgow, Glasgow, G12 8TA, UK
[2]Department of Engineering, University of Cambridge, Cambridge, CB3 0FA, UK
[3]Department of Engineering, University of Sheffield, Sheffield, S1 3JD, UK
[4]Department of Mathematics, Imperial College London, London, SW7 2AZ, UK
[5]The Alan Turing Institute, The British Library, London, NW1 2DB, UK
Corresponding author: Lawrence A. Bull; Email: lawrence.bull@glasgow.ac.uk

Received: 16 May 2023; **Revised:** 10 August 2024; **Accepted:** 26 September 2024

Keywords: damage localization; deep Gaussian processes; meta-models; multilevel models; transfer learning

Abstract

In practice, nondestructive testing (NDT) procedures tend to consider experiments (and their respective models) as distinct, conducted in isolation, and associated with independent data. In contrast, this work looks to capture the interdependencies between acoustic emission (AE) experiments (as meta-models) and then use the resulting functions to predict the model hyperparameters for previously unobserved systems. We utilize a Bayesian multilevel approach (similar to deep Gaussian Processes) where a higher-level *meta-model* captures the inter-task relationships. Our key contribution is how knowledge of the experimental campaign can be encoded *between* tasks as well as within tasks. We present an example of AE time-of-arrival mapping for source localization, to illustrate how multilevel models naturally lend themselves to representing aggregate systems in engineering. We constrain the meta-model based on domain knowledge, then use the inter-task functions for transfer learning, predicting hyperparameters for models of previously unobserved experiments (for a specific design).

Impact Statement

Data-centric engineering considers telemetry data from populations of systems with increasingly complex interdependencies. This work introduces a Bayesian approach to represent the relationships between aggregate systems, via multilevel models, similar to deep Gaussian processes. Considering the population as a whole, the value of data is extended and knowledge of underlying physics can be incorporated within models and between models. By leveraging information between domains, the combined model is used for transfer learning and predicts the characteristics (hyperparameters) of systems that were previously unobserved.

1. Introduction: Multilevel models for meta-modelling

Increasingly, engineering systems are equipped with sensors, often providing streams of telemetry data. As the number of instrumented systems grows, *population* data become available [1]. While machine populations clearly differ from the natural examples in ecology, epidemiology, and behavioural science, the same statistical methods can be used to represent artificial systems [2].

In this article, we consider *multilevel* (or hierarchical) models [3] which are particularly suited to population data, as they naturally exhibit a hierarchical structure. As an engineering example, consider turbines of the same specification in a wind farm. Each individual is unique, with its own wind-power relationship, depending on the local environment. On the other hand, some parameters are shared between operating subgroups (e.g. maximum power) while others are global (e.g. the minimum rpm to generate power).

A multilevel model can represent interdependent population data via *partial pooling* [3] where parameter hierarchies are learnt with machine-specific and shared parameters (or functions). As such, models share (i.e. pool) information, to extend the value of data. Multilevel models are often termed *multitask learners*, as multiple related tasks f_k are learnt simultaneously to improve inference [4]. Figure 1 visualises how multilevel models can learn multiple tasks.

With interpretable models, one can inspect how parameters θ_k vary between tasks. These variations are termed *intertask relationships*, and each function that approximates them (shown as g in Figure 1) can be considered as a *meta-model* (a *model of models*). If desired, these intertask functions can be built to vary with respect to higher-level (*macro*) explanatory variables. A simple example is the varying coefficients model, typically demonstrated with the 8-schools data [5]. Intertask relationships are useful in engineering practice, as they allow alternative insights to be extracted from collected population data.

1.1. Source localisation

This work considers a population where members are represented by a sequence of 28 Acoustic Emission (AE) experiments, originally from [6]. In each experiment, the arrival time of waves propagating through a complex plate geometry is recorded. For each dataset, a 2D map of the arrival times is learnt through regression. Despite distinct experiment designs, the metal plate remains consistent (the medium through which waves propagate) suggesting that their models share certain parameters. By analysing these datasets collectively, as a population, additional insights are extracted from the test campaign. We capture how *characteristics* of the 2D map vary, allowing predictions of hyperparameters for the response surface of experimental designs that were absent from the training data. In other words, one can extrapolate or interpolate in the *model space* by using the intertask functions as meta-models, to predict hyperparameters. Because of the plate's complexity, we encode weak constraints on the model given domain expertise of the experimental campaign, rather than specific physics-based laws (e.g. via differential equations).

These insights have significant implications since the meta-models can be used to predict the characteristics of the response (hyperparameters) for new, previously unobserved experimental designs. Such *model* predictions alleviate the requirement of extensive training data in new tasks, by sharing information from similar experiments. The result is an interpretable and explainable approach to transfer learning for engineering experiments.

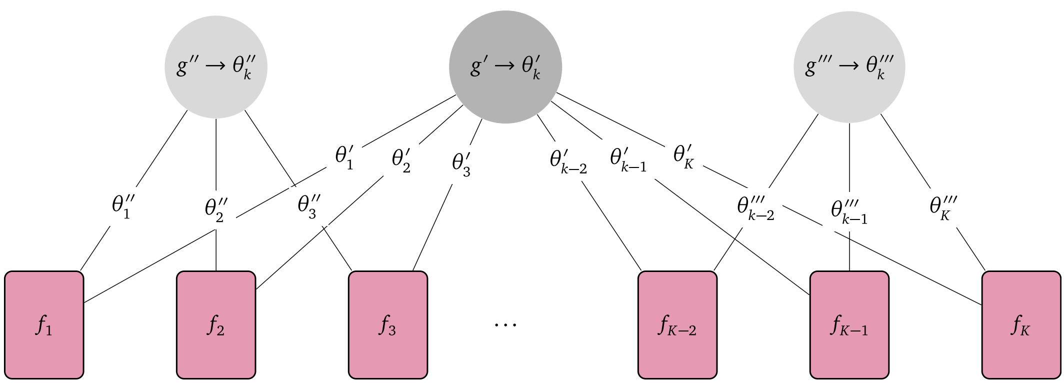

Figure 1. *A visual example of multilevel models for multitask learning: predictive tasks f_k are parametrised by $\theta_k = \{\theta'_k, \theta''_k\}$ or $\theta_k = \{\theta'_k, \theta'''_k\}$; in turn, θ_k is generated by shared intertask functions $\{g', g''\}$ or $\{g', g'''\}$. The notation $g \to \theta$ shows a function g that predicts a parameter θ.*

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

1.2. Layout

Section 2 introduces the AE data, the experimental campaign, and their multilevel interpretation. Section 3 uses Gaussian Process regression for a single AE experiment. Section 4 extends the model to represent data from the full test campaign in a joint inference, investigating different model assumptions and the associated intertask relationships. Section 5 assesses the predictive performance and utilises the model for transfer learning. Section 6 offers concluding remarks.

1.3. Related Work & Contribution

To improve interpretability and ensure meaningful outputs from data-driven models, the inclusion of physical insight within machine learning methods is becoming increasingly popular, with overviews found in [7, 8]. These approaches are often referred to as physics-informed machine learning, since they utilise both data and explanatory physics to improve modelling when compared to either approach independently. Some examples include physics-guided loss functions [9], vector field constraints [10], and the inclusion of governing differential equations [11]. Given the complexity of the plate geometry in this work, the physics-based constraints are *soft*, since any predictive functions can deviate from the suggested structure. To achieve this, a Bayesian approach is used, where the constraints are placed on a (hierarchical) prior distribution [3], which offers a natural trade-off between prior knowledge and data. While this trade-off is possible with a (weighted) physics-guided loss function, we favour a hierarchical statistical approach for uncertainty quantification and interpretability when multitask learning.

Contribution In the context of AE (time of arrival) mapping, previous work of (coauthors) Jones et al. [12] considered how domain knowledge can be encoded in Gaussian Process (GP) models via boundary condition constraints, applied to the kernel function for a single experiment. Here, we extend the work to represent multiple experiments, allowing us to encode domain expertise at the *systems level* and infer intertask relationships for a series of tests[1]. Our key contribution is the ability to encode physics-based knowledge *between* tasks, as well as within tasks. Rather than a single experiment, the resultant model represents variations over collected tests in an experimental campaign: this allows for simulation and (hyper) parameter prediction at previously unobserved experimental designs (in this case, sensor separation).

A multilevel modeling approach is adopted, which is increasingly utilized in the engineering literature. An early monitoring example is presented by Huang et al. [13] and Huang and Beck [14], where multiple correlated regression tasks are utilized for modal analysis. A shared sparsity profile is inferred for tasks relating to measurement channels to improve damage detection by considering the correlation between damage scenarios or adjacent sensors. More recent applications include Di Francesco et al. [15] who use multilevel models to represent corrosion progression given evidence from multiple locations, and Papadimas and Dodwell [16], where the results from materials tests (that is coupon samples) are combined to inform the estimation of material properties. Similar to Papadimas and Dodwell [16] our work considers an experimental campaign, but rather than infer the higher-level representation as a global estimate of a single experiment, we introduce inter-task explanatory variables. In turn, the model represents task variations as functions, rather than uni-modal sampling distributions.

On a related theme, Hughes et al. [17] recognize structures and their populations as nested hierarchies, proposing a convenient formulation for decision analyses. Sedehi et al. [18] also present work to encode physics into hierarchical GPs, where time-history measurements are partitioned into multiple segments to create longitudinal data, accounting for temporal variability and addressing the non-stationarity of the measured responses for a single structure.

2. The AE experiments

AE are ultrasonic signals released within a material as its internal structure undergoes some irreversible change. The driving mechanisms often relate to the initiation and growth of damage, so monitoring AE

[1] In this work, we use the term *systems level* to refer to the combined tasks, which allow parameters or functions to be learnt between multiple experiments; that is the inter-task functions.

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

signals can serve to assess the condition of materials and structures [19]. As emissions propagate through material from the point of origin, differences in the time of arrival at separate sensors (in an array) can be used for *triangulation* (or *trilateration*) to enable source localization [20]. From a damage monitoring perspective, source localization provides an operator with more insight to make better maintenance and planning decisions.

One strategy for localizing AE signals involves learning the map of the arrival times across the surface of interest [21]. That is, the forward mapping from AE source localization $\mathbf{x}_i$ to the measured difference in time-of-arrival (ΔToA) for a given sensor pair,

$$y_i = f(\mathbf{x}_i) + \epsilon_i \tag{2.1}$$

where y_i is some *noisy* observation of ΔToA with additive observation noise ϵ_i. In this paper, we consider data from experiments by Hensman et al. [6] concerning an aluminum plate-like structure shown in Figure 2.

Artificial AEs were excited by thermoelastic expansion generated with an incident laser pulse, with the signals captured at eight piezoceramic (Sonox P5) sensors mounted to the surface of the plate, visible in Figure 2 and shown by the black markers in Figure 3, numbered 1–8 (left). The sensors operate by converting surface displacements resulting from the AE stress waves into electrical energy, allowing the ultrasonic signals to be captured digitally. There are a total of $N = 2{,}227$ possible source locations, shown by blue markers in Figure 3 (left). The time of arrival for each of the eight sensors was extracted following standard practice, using an autoregressive form of the Akaike Information Criterion (AIC) [6].

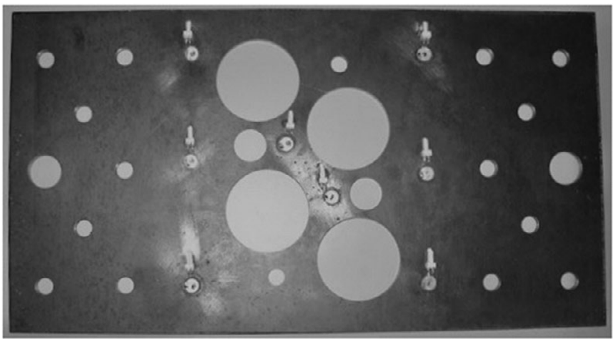

Figure 2. *Image of plate used in the AE experiments.*

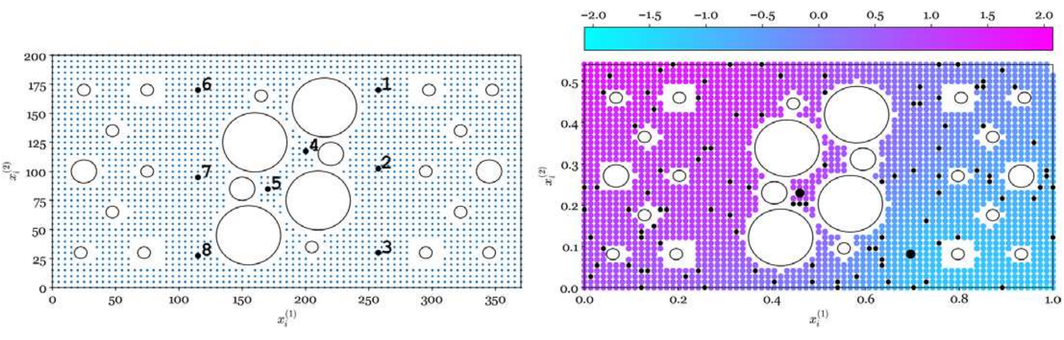

Figure 3. *Left: all source locations (blue •) and sensor locations (black •). Right: heatmap of the ΔToA (response y_i) given locations (inputs $\mathbf{x}_i = \left\{x_i^{(1)}, x_i^{(2)}\right\}$) for the experiment $k = 15$, sensor pair (3, 5).*

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

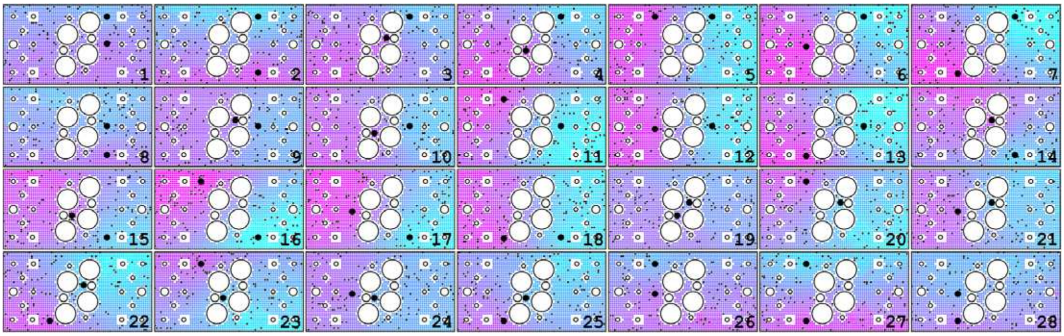

Figure 4. *Heatmaps of the measured ΔToA (y_i) with respect to locations ($\mathbf{x}_i$) for all sensor combinations. Large black markers plot sensor pairs, while small black markers plot training observations. The remaining data are used to test out-of-sample performance. The number in the bottom right is the experiment index $k \in \{1, 2, \ldots 28\}$.*

2.1. Difference in time-of-arrival (ΔToA)

Let the arrival time of AE i at sensor j be denoted A_{ij} $\forall j \in \{1, 2, \ldots, 8\}$. The difference in time-of-arrival (ΔToA) is then the difference between any two sensors. Since there are eight sensors, there are 28 pairwise combinations and associated maps (8C2). Each pair generates a different f (distinguished with notation later). For example, the pair (3, 5) would present the scalar output $y_i \in \mathbb{R}$, $y_i = A_{i3} - A_{i5}$. The input vectors $\mathbf{x}_i \in \mathbb{R}^2$ are the locations (length versus width), $\mathbf{x}_i = \left\{ x_i^{(1)}, x_i^{(2)} \right\}$where $\{0, 0\}$ is the bottom left corner of the plate. Figure 3 (right) shows the data associated with the sensor pair (3, 5) while Figure 4 shows all pairs. Each combination is labelled with an experiment index $k \in \{1, 2 \ldots, 28\}$ listed in Table 2 Appendix B.

2.2. A note on normalization

Figure 4 plots $N = 100$ training data for each sensor pair combination, sampled uniformly from each task. We then normalize inputs with respect to the longest edge, such that the plate's length is unity. This scaling maintains approximately the same relative smoothness of the map in each direction, that is one length scale for both dimensions. Rather than normalize the response for each plate independently, it is z-score normalized with respect to all 28 sensor pairings. This maintains the relative structure between experiments (sensor pairs) in the combined data. More specifically, a *global* normalization of the outputs is essential to prevent meaningful differences (between tests) from being scaled out of the data.

2.3. Why multilevel?

In the context of this work, each map and sensor pair corresponds to a distinct but related *environment*: each referred to as an *experiment*, with an associated predictive task learned as a regression. These collected environments are visualized in Figure 4. If one learns these maps collectively, in a combined inference, the model should capture variations between each sensor pair. These relationships capture further insights from the experimental campaign, which emerge at the systems level.

3. Representing AE maps with Gaussian Processes Regression

GPs are used to represent each of these experiments since they offer a flexible tool for regression with natural mechanisms to encode engineering knowledge and domain expertise. First, we consider one task at a time (Section 4 extends the model to consider all experiments in a combined inference). The additive noise ϵ_i is assumed to be normally distributed,

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \qquad \epsilon_i \sim N(0, \sigma_i) \tag{3.1}$$

in practice, ϵ_i will represent more than observation noise, since it captures the combined uncertainty of the ΔToA extraction, from the raw time series data. Still, as with previous work [21, 22, 12, 6], we found this representation works well in practice. Rather than suggest a parametrization of f we assume that it has a nonparametric GP prior distribution[2],

$$\boldsymbol{\theta} \sim p(\boldsymbol{\phi}) \tag{3.2}$$

$$f \sim \mathrm{GP}\left(m_f(\cdot;\boldsymbol{\theta}), k_f(\cdot,\cdot;\boldsymbol{\theta})\right) \tag{3.3}$$

The prior over f is specified by its mean m_f and covariance k_f functions, where $\boldsymbol{\theta}$ is the set of collected hyperparameters. These hyperparameters are sampled from the higher level distribution $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$ whose *prior* is parametrised by constants in $\boldsymbol{\phi}$. The mean and covariance of the prior offer natural mechanisms to encode knowledge of the expected functions given domain expertise, before any data are observed. The covariance determines the expected correlation between outputs—influencing process variance, and smoothness—while the mean represents the prior expectation of the structure of the functions. A function sample from the GP, denoted $\mathbf{f}$, is multivariate normal for any finite set of N observations,

$$\mathbf{f} \sim \mathrm{N}(\mathbf{m_f}, \mathbf{K_f}) \tag{3.4}$$

$$\mathbf{y} \sim \mathrm{N}(\mathbf{f}, \boldsymbol{\sigma}) \tag{3.5}$$

$$\mathbf{K_f}[i,j] \triangleq k_f\left(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta}\right) \tag{3.6}$$

$$\mathbf{m_f}[i] \triangleq m_f(\mathbf{x}_i) \tag{3.7}$$

Since the *observation* model is assumed Gaussian, we can avoid sampling $\mathbf{f}$ entirely by combining the GP kernel $\mathbf{K_f}$ with the additive noise vector σ, to describe the likelihood function,

$$\mathbf{y} \sim \mathrm{N}\left(\mathbf{m_f}, \mathbf{K_f} + \mathrm{diag}\left(\boldsymbol{\sigma}^2\right)\right) \tag{3.8}$$

Following [21] we use a zero-mean and Matérn $3/2$ kernel function k_f for the covariance function,

$$\mathbf{m_f}[i] = 0 \tag{3.9}$$

$$\mathbf{K_f}[i,j] = \alpha^2 \left(1 + \frac{\sqrt{3}|\mathbf{x}_i - \mathbf{x}_j|}{l}\right) \exp\left(-\frac{\sqrt{3}|\mathbf{x}_i - \mathbf{x}_j|}{l}\right) \tag{3.10}$$

The zero-mean function is justified since the complex plate geometry prevents the specification of a parametrized mean. In the absence of this information, a zero mean is usually sufficient practice, since the GP alone is flexible enough to model arbitrary trends, given enough training data. Two hyperparameters are introduced via the kernel k_f (3.10) such that $\boldsymbol{\theta} = \{\alpha, l\}$. The process variance α encodes the magnitude of function variations around the expected mean. The length scale l encodes how much influence a training observation has on its neighboring inputs, it represents the *smoothness* of the approximating family of functions.

3.1. Heteroscedastic noise

In the experiments, the scale of the additive noise σ_i is expected to increase at the extremities of the plate [22]. As such, the magnitude of the noise is *input-dependent*, and it is not statistically identical over the

[2] Note, we use $p(\mathbf{a}|\mathbf{b})$ and $\mathbf{a} \sim p(\mathbf{b})$ to denote the same probability distribution.

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

whole input (that is it is not homoscedastic). Such input-dependent noise can be represented with a heteroscedastic regression. Heteroscedasticity is implemented with another GP in a combined model, mapping the inputs $\mathbf{x}_i$ to the scale of the additive noise σ_i. This introduces the *noise process* r, where the prior utilizes a constant mean function[3] $m_r(\mathbf{x}_i) = m_r$ and another Matérn $3/2$ kernel function (with its own hyperparameters). Therefore, a finite sample from the noise process is distributed as follows,

$$\mathbf{r} \sim \mathrm{N}(\mathbf{m_r}, \mathbf{K_r}) \tag{3.11}$$

$$\sigma = \exp(\mathbf{r}) \tag{3.12}$$

$$\begin{aligned} \mathbf{m_r}[i] &= m_r \\ \mathbf{K_r}[i,j] &= \alpha_r^2 \left(1 + \frac{\sqrt{3}|\mathbf{x}_i - \mathbf{x}_j|}{l_r}\right) \exp\left(-\frac{\sqrt{3}|\mathbf{x}_i - \mathbf{x}_j|}{l_r}\right) \end{aligned} \tag{3.13}$$

Samples from r are exponentiated to define $\boldsymbol{\sigma}$ since the noise variance must be strictly positive (note that, untransformed, a GP will map to any real number). The exponential transformation requires a constant mean function, otherwise, the prior of the expected noise variance is too large, as a zero mean function would lead to $\exp(0) = 1$. The additional hyperparameters from the noise process are now included in the total set $\boldsymbol{\theta}$,

$$\{m_r, \alpha_r, l_r\} \in \boldsymbol{\theta}$$

3.2. Prior formulations

One should encode *prior knowledge* of the AE maps as prior distributions over the higher-level variables $\boldsymbol{\theta} \sim p(\boldsymbol{\phi})$. The hyperparameters of the GP are sampled from these distributions, which characterize the expected variation, smoothness, and noise of the response. The following $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$ structure is adopted,

$$\boldsymbol{\theta} = \{l, \alpha, m_r, \alpha_r, l_r\}, \quad \boldsymbol{\theta} \sim p(\boldsymbol{\phi})$$

$$l \sim \mathrm{Gamma}(\mathrm{shape} = 2, \mathrm{rate} = 1) \tag{3.14}$$

$$\alpha \sim \text{Half-Normal}^{+}(\mathrm{scale} = 1) \tag{3.15}$$

$$m_r \sim \mathrm{N}(\mathrm{mean} = -0.9, \mathrm{scale} = 1) \tag{3.16}$$

$$l_r \sim \mathrm{Gamma}(\mathrm{shape} = 2, \mathrm{rate} = 1) \tag{3.17}$$

$$\alpha_r \sim \text{Half-Normal}^{+}(\mathrm{scale} = 1) \tag{3.18}$$

where Gamma and Half-Normal^{+} are the Gamma and positive Half-Normal probability distribution functions respectively. The Half-Normal distribution is centered at zero, with support for positive values only. The constants in (3.14)–(3.18) correspond to $\boldsymbol{\phi}$, and they are specified given the combined normalized space as weakly informative prior distributions, explained below.

Recall that the inputs are normalized between [0, 1] with respect to the longest side. We expect a smooth ΔToA map (f) and noise process (r), so the Gamma distributed length scales l (3.14) and l_r (3.17) have their mode at $1 = (\texttt{shape} - 1)/\texttt{rate}$ [3]. The output is z-score normalized, so we expect the

[3] A constant GP mean function does not imply the posterior (predictive) distribution of r is constant, only the mean of its prior distribution.

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

process variance α should be less than unity (shrunk further by outliers). This is reflected by a half-normal distribution with a unit scale, with an expected value $E[\alpha] = \frac{1\sqrt{2}}{\sqrt{\pi}} = 0.8$.

The expectation of the noise process r is set to 0.08, to encode prior belief of a signal-to-noise ratio of 10, in terms of expected scale,

$$\frac{\text{expected process noise}}{\text{expected additive noise}} = \frac{\mathrm{E}[\alpha]}{\mathrm{E}[\sigma]} = \frac{\mathrm{E}[\alpha]}{\mathrm{E}[\exp(r)]} = \frac{\mathrm{E}[\alpha]}{\exp(\mathrm{E}[\mathbf{m_r}])} \tag{3.19}$$

$$= \frac{1\sqrt{2}\times\pi^{-0.5}}{\exp(-2.5)} = \frac{0.8}{0.08} \tag{3.20}$$

A vague prior is defined as α_r, which indicates high variance in the noise process r, to reflect weak a priori knowledge,

$$\mathrm{E}[\alpha_r] = \frac{2\sqrt{2}}{\sqrt{\pi}} = 1.6 \tag{3.21}$$

A high scale for the noise process is assumed, since it is known that measurement noise increases dramatically at the extremities of the plate, far from the centroid of sensor pairs [22].

3.3. Inference and prediction

To identify the model and make predictions, one can infer the posterior distribution for latent variables $p(\boldsymbol{\Theta}|\mathbf{y})$ by conditioning the joint distribution (which encodes domain expertise via the model and the prior specification) on the training data $\mathbf{y}$. We use $\boldsymbol{\Theta}$ to generically collect all (unobservable) latent variables, including functions, parameters, and hyperparameters. The joint distribution is written as the product of two densities, referred to as the *likelihood* $p(\mathbf{y}|\boldsymbol{\Theta})$ (or the data distribution) and the *prior* $p(\boldsymbol{\Theta})$,

$$p(\mathbf{y},\boldsymbol{\Theta}) = p(\mathbf{y}|\boldsymbol{\Theta})p(\boldsymbol{\Theta};\boldsymbol{\phi}) \tag{3.22}$$

During model design, we have specified the likelihood with (3.8) and the prior throughout (3.14)–(3.18). Applying the property of conditional probability to (3.22) we arrive at Baye's rule and an expression for the posterior distribution,

$$p(\boldsymbol{\Theta}|\mathbf{y}) = \frac{p(\mathbf{y}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})}{p(\mathbf{y})} \tag{3.23}$$

While (3.23) is a straightforward application of conditioning [4], in practice, the evaluation of the denominator (that is the *marginal likelihood* or *evidence*) is nontrivial. It is specified by the following integral, which is intractable for most prior-likelihood combinations,

$$\text{evidence}: \quad p(\mathbf{y}) = \int p(\mathbf{y},\boldsymbol{\Theta})\,d\boldsymbol{\Theta} = \int p(\mathbf{y}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})\,d\boldsymbol{\Theta} \tag{3.24}$$

The integral (3.24) is feasible for a subset of likelihood-prior distributions, known as conjugate pairs [3]. In many practical applications, however, it becomes increasingly hard to justify the model and prior choices that lead to conjugacy. In our case, the prior formulation $p(\boldsymbol{\theta}|\boldsymbol{\phi})$ required for a multilevel representation leads to an intractable eq. (3.24).

A number of approximate Bayesian methods are used with non-conjugate models. Here, we utilize a sampling-based solution, inferring the parameters using MCMC and the no U-turn implementation of Hamiltonian Monte Carlo [23]. The models are implemented in the probabilistic programming language `Stan` [24].

When predicting new (previously unobserved) data $\tilde{\mathbf{y}}$ one (empirically) integrates out $\boldsymbol{\Theta}$ from the following product,

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

$$p(\tilde{\mathbf{y}}|\mathbf{y}) = \int p(\tilde{\mathbf{y}}|\boldsymbol{\Theta})p(\boldsymbol{\Theta}|\mathbf{y})d\boldsymbol{\Theta}$$

For GP variables $\{f, r\}$, the distribution used to predict new inputs has an analytical solution when the hyperparameters $\boldsymbol{\theta}$ are fixed. For the f-process, this would be $p(\tilde{\mathbf{y}}|\mathbf{y}, \boldsymbol{\theta}_s)$ where $\boldsymbol{\theta}_s$ represents samples from the approximated posterior distribution. The analytical solution is then defined by conditioning a joint Gaussian, e.g. $p(\tilde{\mathbf{y}}, \mathbf{y}|\boldsymbol{\theta}_s)$, on the training variables for all samples from the approximated posterior distribution. The relevant identity is provided in Appendix A.

For the (3,5) sensor pair, a random sample of $N = 100$ observations is used for training, and the rest are set aside as test data. Following inference and prediction, Figure 5 plots the mean of the posterior predictive distribution for the two-dimensional map over the plate. To visualize the predictive variance and its heteroscedastic nature, a random slice is taken along the length of the map and also plotted in Figure 5.

4. Multilevel representations

To conceptualize the hierarchical structure of the experimental data, independent GPs are learned for all 28 tests (rather than sensor pair (3, 5) only). Recall that each pairwise map is distinguished with an index $k \in \{1, 2, \ldots, 28\}$ listed in Table 2, Appendix B. The set of all maps is given by,

$$\{f_k\}_{k=1}^{K}$$

Throughout, we use the same test-train split of $N = 100$ random samples from each task, visualized in Figure 4. Figure 6 shows samples from the posterior distributions $p(\boldsymbol{\theta}|\mathbf{y}_k)\forall k$. Since the data are not shared between the experiments, each task is learned independently, and these models are considered single-task learners (STL) [4]. Figure 6 is typical longitudinal or panel data [3] where the task-specific $\boldsymbol{\theta}$ are similar, with random variations. The experiment-specific models can be viewed as perturbations around an average (higher-level model). An intuitive concept, since all experiments concern the same plate, despite the varying experimental design (sensor placement).

To summarize, each task represents a distinct but related environment. In turn, MTL is used to learn predictive maps in a combined inference, to capture the inter-task functions (with quantified uncertainty) relating to differences between tests in the experimental campaign.

4.1. (Method A) K GPs, one prior

In a combined inference, one simple way to share information between tests assumes that the smoothness l and process variance α^2 of all ΔToA maps $\{f_k\}$ are consistent. In other words, the GPs for each experiment are sampled from a shared *prior*,

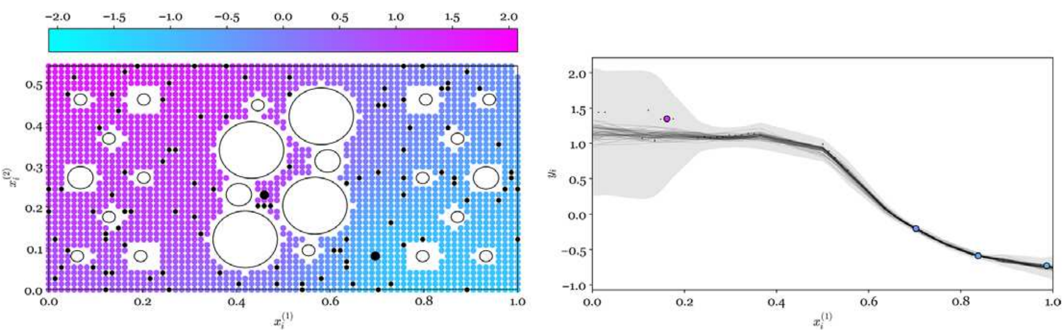

Figure 5. *The inferred sensor pair (3,5) for the experiment k = 15 (left) and a length-wise slice to visualize the heteroscedastic noise (right).*

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

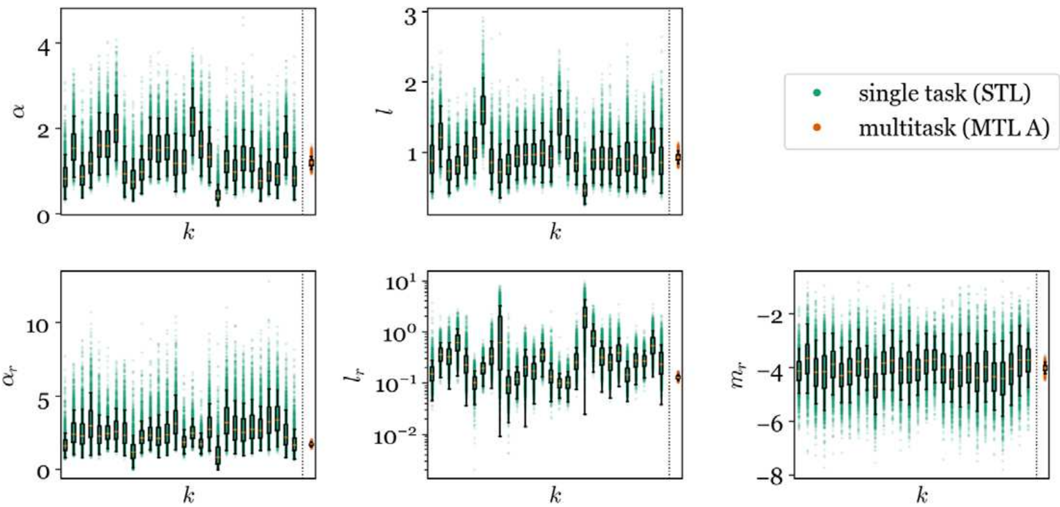

Figure 6. *Posterior distribution of hyperparameters for independent GPs of each experiment (STL, green), compared to sharing hyperparameters (the prior) between all experiments (MTL-A, orange). Top row: f-process hyperparameters (AE map). Bottom row: r-process hyperparameters (heteroscedastic noise).*

$$\{\mathbf{f}_k\}_{k=1}^{K} \sim \mathrm{N}(\mathbf{m_f}, \mathbf{K_f})$$

Same for the noise process r_k,

$$\{\mathbf{r}_k\}_{k=1}^{K} \sim \mathrm{N}(\mathbf{m_r}, \mathbf{K_r})$$

Shared hyperparameters across all K GPs is one form of partial pooling [3] and it will likely improve inference compared to independent learners. However, Figure 6 suggests the assumption of tied hyperparameters is too simple. Furthermore, process understanding supports this concern: it is unlikely that the independent posterior samples correspond to the same hyperparameter since there are differences in the design of each experiment. The question is whether the assumption provides a sufficient model. Considering the above, distinct maps for each experiment $\{f_k\}$ are sampled from GP priors with shared hyperparameters distributed by $p(\boldsymbol{\theta})$. This approach is presented as a benchmark—multitask learning method A (MTL-A). The resultant hyperparameter posterior distributions are shown by orange markers in Figure 6. Their reduced variance is typical for pooled estimates because a single distribution is inferred from all K datasets (rather than one for each task). Caution is required, this assumption misrepresents the population variance if the model form is misspecified.

4.2. (Method B) Hyperparameter modelling

A more intuitive generating process considers that certain hyperparameters are conditioned on the experimental setup, rather consistent, to better represent the differences between each test. We extend the notation of (3.2) with a k-index to reflect this,

$$\begin{aligned} \boldsymbol{\theta}_k &\sim p(\boldsymbol{\phi}) \\ \boldsymbol{\theta}_k &\triangleq \{\alpha_k, l_k\} \end{aligned} \tag{4.1}$$

Distinct variables are considered for the $\mathbf{K_f}$ kernel only (α_k and l_k) since these are easier to interpret and relate to domain knowledge, especially if the plate had represented a simple geometry—this choice also helps to constrain the model design. Having a distinct set $\boldsymbol{\theta}_k$ allows the characteristics of the map to vary between each experiment: this should be expected since they are different experimental designs. The higher-level sampling distribution $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$ remains shared between all experiments (and learned from pooled data to share information). We now consider modifications to the prior model $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$ to encode knowledge and domain expertise of the intertask relationships, that is between the experiments.

4.2.1. Beyond exchangeable experiments

When specifying a new prior model $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$ it is important to consider whether the experiments are exchangeable [3]. Here, they would be considered exchangeable if no information is available to distinguish the $\boldsymbol{\theta}_k$'s from one another. In other words, the set $\{\boldsymbol{\theta}_k\}_{k=1}^{k}$ (presented in Figure 6) cannot be reordered such that patterns are revealed in the latent variables.

However, since we know about the design of the AE experiments, there are a few possibilities. One simple description is sensor separation, to order the models and reveal structure (that is patterns) in the hyperparameter estimates. Once structure appears, the experiments are no longer exchangeable. Sensor separation δS_k is defined as the Euclidean distance between any two of the eight $\binom{8}{2}$ possible sensor locations $\widehat{\mathbf{S}} = \{\widehat{\mathbf{s}}_1, \widehat{\mathbf{s}}_2, \dots \widehat{\mathbf{s}}_8\}$,

$$\delta S_k = |\widehat{\mathbf{s}}_g - \widehat{\mathbf{s}}_h| \quad \forall \text{ pairs from} \widehat{\mathbf{S}} \tag{4.2}$$

where the vector $\delta\mathbf{S}$ will be length $K = \binom{8}{2} = 28$. The values for sensor separation from each experiment are also presented in Table 2, Appendix B. The simple pattern we expect is that process variance α_k will be greater for sensors that are further apart; that is the variation in the ΔToA signal will be greater since the AE signals have the potential to travel further.

Figure 7 plots the posterior distributions $\boldsymbol{\theta}_k$ (STL) with respect to δS_k for the independent models from Figure 6. These samples indicate parameter relationships that we hope to learn with the higher-level model $p(\boldsymbol{\theta}_k|\boldsymbol{\phi})$. (Note that in Figure 7, samples correspond to latent variables and not observations.)

As expected, the process variance α_k increases with sensor separation. The length scale l_k presents a similar trend: we believe the increasing length scale appears for smaller sensor spacing as the noise floor (observation noise as well as parameter uncertainty) is larger compared to the magnitude of the response variations (α_k). In turn, the influence of each training observation on its neighbors is reduced.

Figure 7 also highlights the hold-out experiments $k \in \{4,7,16,22\}$ represented by black markers (rather than green). Herein, hold-out experiments are not used when training (as with held-out data) to test interpolation and extrapolation in the hyperparameter space for the multilevel models.

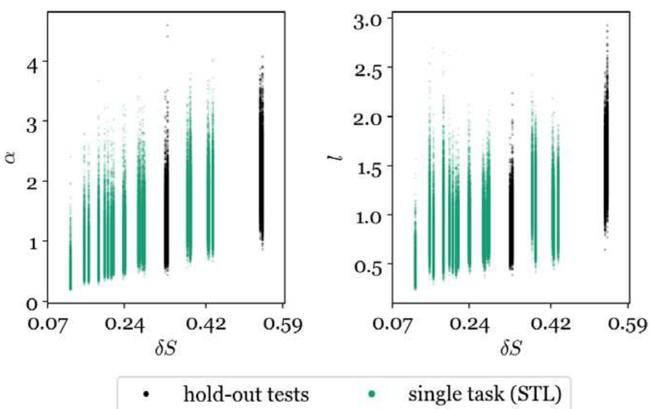

Figure 7. *Ordering the hyperparameters (conditional posterior distribution) from the STL experiments θ_k with respect to sensor separation δS_k. Black markers correspond to models of the hold-out tests $k \in \{4,7,16,22\}$ which do not contribute training data for MTL.*

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

4.2.2. Intertask GPs

Like the input-dependent noise process (r) the variation of the hyperparameters (of $\mathbf{K_f}$) can be represented with another function. The output of these functions is *task-dependent* (that is experiment-dependent). That is, the hyperparameters of the AE map f are sampled from higher-level functions, denoted $\{g, h\}$,

$$\boldsymbol{\theta} \sim p(\boldsymbol{\phi}) \tag{4.3}$$

$$(\alpha) \quad \text{g-process}: \quad g \sim \mathrm{GP}\left(m_g(\cdot;\boldsymbol{\theta}), k_g(\cdot,\cdot;\boldsymbol{\theta})\right) \tag{4.4}$$

$$(l) \quad \text{h-process}: \quad h \sim \mathrm{GP}(m_h(\cdot;\boldsymbol{\theta}), k_h(\cdot,\cdot;\boldsymbol{\theta})) \tag{4.5}$$

where hyperparameter functions vary (between tasks) with respect to sensor separation δS_k for the 24 training experiments,

$$\mathbf{g} \sim \mathrm{N}\left(\mathbf{m_g}, \mathbf{K_g}\right) \tag{4.6}$$

$$\alpha_k = \mathrm{softplus}(g_k) \quad \forall k \in \mathrm{train} \tag{4.7}$$

$$\mathbf{h} \sim \mathrm{N}(\mathbf{m_h}, \mathbf{K_h}) \tag{4.8}$$

$$l_k = \mathrm{softplus}(h_k) \quad \forall k \in \mathrm{train} \tag{4.9}$$

$$\boldsymbol{\theta}_k \triangleq \{\alpha_k, l_k\}$$

Each GP is transformed to be strictly positive, as the hyperparameters they predict must be greater than zero. While the exponential function was used for this purpose with the noise process r (3.12) here softplus transformation [25] is used since it can be specified to approximate the identity function ($f(a) = a$) over the inputs of interest, aiding the interpretability of the hyperparameter models $\{g, h\}$. The exponentiated transformation is maintained for the lower-level GPs f_k since it prevented divergences during inference via HMC, and aided convergence of the MCMC chains.

4.2.3. Priors

The noise process prior distributions are set as before (3.16)–(3.18) while the AE map priors are sampled from GPs (4.7)–(4.9) whose hyperparameters are added to θ. We use the same Matérn 3/2 kernel for the higher-level GPs (k_g and k_h); however, we use a linear mean function (with slope β and gradient γ),

$$\begin{aligned}
\mathbf{K_g}[k,l] &\triangleq k_g(\delta S_k, \delta S_l; \boldsymbol{\theta}) \\
\mathbf{K_h}[k,l] &\triangleq k_h(\delta S_k, \delta S_l; \boldsymbol{\theta}) \\
\mathbf{m_g}[k] &\triangleq \beta_g \delta S_k + \gamma_g \\
\mathbf{m_h}[k] &\triangleq \beta_h \delta S_k + \gamma_h
\end{aligned}$$

The linear mean allows us to encode domain knowledge of a positive gradient (as a soft constraint) for the expected intertask functions via the GP prior. The conditional posterior predictive distributions of $\{g, h\}$ then provide intertask relationships learned from the collected data (while only observing data as inputs on the lowest level f). The additional hyperparameters for the shared (global) sampling distributions are, $\left\{\alpha_g, l_g, \alpha_h, l_h, \beta_g, \gamma_g, \beta_h, \gamma_h\right\} \in \boldsymbol{\theta}$where β and γ are the slope and intercepts of the linear mean function for the intertask relationships. The priors for each of these should reflect the difficulty in making specific statements around hyperparameter values, due to their limited interpretability,

$$\left\{\alpha_g, l_g, \alpha_h, l_h\right\} \overset{\text{i.i.d.}}{\sim} \mathrm{Gamma}(2,1) \tag{4.10}$$

$$\{\beta_g, \beta_h\} \overset{i.i.d.}{\sim} \text{Uniform}(0,10), \quad \{\gamma_g, \gamma_h\} \overset{\text{i.i.d.}}{\sim} \text{Uniform}(-1,1) \tag{4.11}$$

Given the normalized space, these priors encode weak knowledge and constrain the mean function to positive gradients. The upper bound (10) of the uniform distribution was set to avoid divergences of the HMC samples and ensure convergence of MCMC. The descriptive multilevel model, with GPs representing hyperparameter variations, is referred to as multitask learning method B (MTL-B).

4.3. MTL comparison

Table 1 is provided to compare the parameter hierarchies between each method:

- Independent learners (STL)
- Shared GP prior (MTL-A)
- Hyperparameter modeling (MTL-B)

STL has distinct GPs $\{f, r\}$ and associated hyperparameters $\{\alpha, l, m_r, \alpha_r, l_r\}$ for each task. MTL-A has distinct task GPs $\{f, r\}$ but shares hyperparameters $\{\alpha, l, m_r, \alpha_r, l_r\}$ between all tasks. Lastly, MTL-B has distinct GPs $\{f, r\}$ and hyperparameters of the response $\{\alpha, l\}$ which are themselves predicted by intertask GPs $\{g, h\}$, with (shared) hyperparameters $\{\alpha_g, l_g, \alpha_h, l_h\}$.

For visual comparison, Figure 8 plots the resultant hyperparameter posterior distributions for STL (green), MTL-A (orange), and MTL-B (purple). The (purple) functions $\{g, h\}$ appear to capture the hyperparameter relationships, compared to the trends presented by the posterior distributions of the STL models. The variance of the functions is also reduced, compared to independent models (green), without the assumption of one consistent hyperparameter (orange). Critically, while each hyperparameter model makes sense given their respective assumptions, the GP model (purple) is more expressive: it represents the variations of the map between experiments with respect to an explanatory variable (sensor separation).

To summarize, both the process variance α_k and length scale l_k increase with sensor separation δS. For process variance α_k, this is because variation in the difference in time of arrival will be greater for sensors that are further apart. For the length scale $l_{k,}$ we believe lower amplitude signals have a lower signal-to-noise ratio, therefore the inferred functions are less smooth (that is lower length scale).

4.3.1. Modelling hyperparameters, not parameters

Why not include δS_k as an explanatory variable on the lower level—within a product kernel, for example? This is because changes in the response are not smooth with respect to δS_k. Instead, the response surface switches discontinuously between experiments, see Figure 4. However, the response *characteristics* (smoothness, process variance) show smooth relationships with respect to δS_k (observed in Figures 7 and 8). For this reason, it is more appropriate to model variations at the hyperparameter level, rather than the map itself. The expected smoothness of intertask functions should be a primary consideration when building multilevel models.

4.3.2. Post-selection inference

By investigating (independent) model behavior with plots of the posterior distributions, we use the training data twice: (i) for inference and (ii) to inform model design. Specifically, Figure 7 was used to

Table 1. *Comparison of methods based on the hierarchy of latent variables*

Method	STL	MTL-A	MTL-B
Task-specific	$\{f, r\}$ $\{\alpha, l\}$ $\{m_r, \alpha_r, l_r\}$	$\{f, r\}$	$\{f, r\}\{\alpha, l\}$
Shared	N/A	$\{\alpha, l\}$ $\{m_r, \alpha_r, l_r\}$	$\{g, h\}$ $\{\alpha_g, l_g\}$ $\{\alpha_h, l_h\}$ $\{m_r, \alpha_r, l_r\}$

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

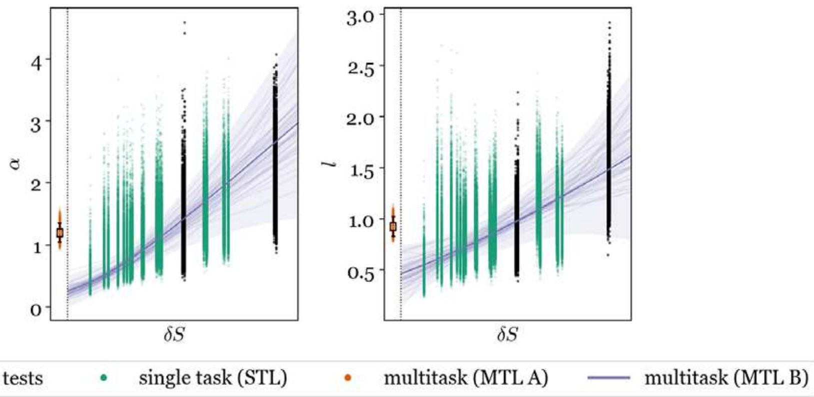

Figure 8. *Posterior distributions $p(\boldsymbol{\theta}_k|\mathbf{y})$ following each method of learning from the series of AE experiments. Independent learners (STL, green); Shared GP prior (MTL-A, orange); Hyperparameter modeling (MTL-B, purple). Black markers correspond to STL models of the hold-out tests $k \in \{4,7,16,22\}$ which do not contribute training data for MTL.*

inform the structure of the multilevel model, so we are guilty of post-selection inference [26]. With reference to Gelman et al. [3], when the number of candidate models is small, the bias resulting from data reuse is also small. In this case study, the plots informed only certain aspects of the higher-level GP design. Post-selection inference should be treated cautiously, however, as when the number of candidate models grows, the risk of overfitting the data increases.

5. Results and discussion

To test each method of representing the experimental data, the ground truth (or target) out-of-sample data $\{\bar{\mathbf{y}}_k\}_{k=1}^{k}$ are compared to the posterior predictive distribution. The predictive log-likelihood is used as a probabilistic assessment of performance,

$$lpY_k = \sum_{i=1}^{N} \log\left(\frac{1}{S}\sum_{s=1}^{S} p(\bar{y}_{ik}|\boldsymbol{\Theta}_s)\right) \tag{5.1}$$

where $\{\boldsymbol{\Theta}_s\}_{s=1}^{S}$ are the S samples from the full approximated posterior distribution and the combined likelihood of all models is $lpY = \sum_{k=1}^{K} lpY_k$. In other words, (5.1) quantifies the (log) likelihood that test data were generated by the model inferred from the training data. A higher value indicates that the model has a better approximation of the underlying data-generating process and indicates good generalization.

Figure 9 presents lpY for all benchmarks. There is a small improvement in the combined predictive likelihood for both MTL methods. Rather than motivate MTL, the predictive likelihoods show that hierarchical GPs maintain the predictive performance of the STL models. Insights are enabled in Figure 8, where the inferred intertask functions are presented with uncertainty quantification—e.g. these are used later for transfer learning in Section 5.1. For task-specific performance (lpY_k) independent STL performs better for a subset of tasks $k \in \{3,8,10,11,13\}$. This is typical, however, since MTL considers the joint distribution of the whole data. If some experiments have data with fewer outliers and less noise, these might be negatively impacted by experiments with higher noise and sparse data. For example, in tasks with low sensor separation, the signal-to-noise ratio is lower, which might represent *weak* data.

Both MTL-A and B extend data for training by partial pooling, but B has a more descriptive multilevel structure, allowing us to encode domain expertise for intertask (between experiment) learning. In turn, hyperparameter relationships are captured over the experimental campaign (rather than their marginal distribution). Modeling prior variations in MTL-B represents changes in the tasks with respect to experimental design parameters (in this case δS_k). To summarise:

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

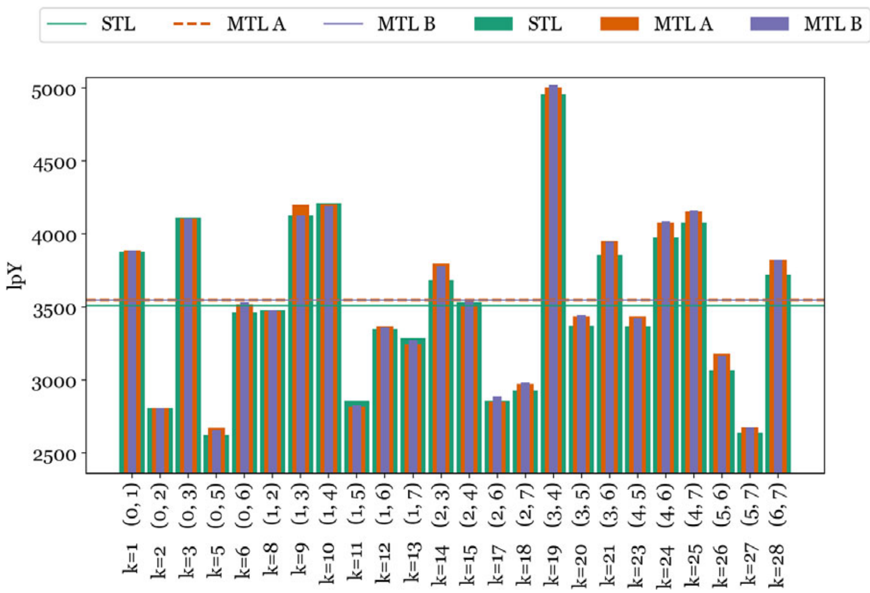

Figure 9. *Predictive log-likelihood for the out-of-sample test data. Excluding hold-out experiments* $k \in \{4,7,16,22\}$ *which are modeled using transfer learning in s:transfer. Lines plot the average across all tasks, the combined log-likelihood (summed) for each method is: (STL, 84217), (MTL A, 85149), and (MTL B, 85204).*

- (A & B) can simulate the hyperparameters for hypothetical/unobserved experimental setups
- (B) has the potential to encode physics/domain expertise of behavior between sensor pairs (mean functions, constraints)
- (A & B) can use the intertask relationships, learned from similar experiments, to predict hyperparameters for tasks with sparse data (this can be viewed as a form of transfer learning)

5.1. Using the multilevel model for transfer learning

To demonstrate transfer learning, the intertask functions are used for meta-modeling, to interpolate and extrapolate in the model space. The results intend to show how multilevel models can capture overarching insights from the experimental campaign (at the systems level) as well as task-specific insights. The expected hyperparameter values $E[\boldsymbol{\theta}_k]$ from MTL-A and B can be visualized in Figure 8 with the solid lines. These point estimates are used when conditioning on data from the hold-out tests[4]. In turn, hyperparameter inference can be avoided for new (previously unobserved) experiments. Instead, we predict their value given the other, similar experiments: where MTL-A assumes one hyperparameter set for all tests, and MTL-B learns how these vary with respect to sensor separation. By predicting hyperparameters, informed by data-rich tasks, the predictive performance should be improved for new experiments with sparse data.

The expected hyperparameter values are used to condition new GPs for an increasing training budget ($N = 5 - 100$) for the hold-out experiments. Figure 10 shows that both forms of MTL consistently improve the predictive performance, especially when extrapolating in the model space ($\delta S = 0.54$). These improvements are intuitive since the extrapolated parameters are associated with higher uncertainty for conventional STL (refer to Figure 8). Another (more natural) way to share information would be to retrain

[4] Ideally, the full predictive distribution should be used, rather than the expectation alone. A point estimate is used here, however, for computational reasons—there are 100 repeats of these experiments.

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

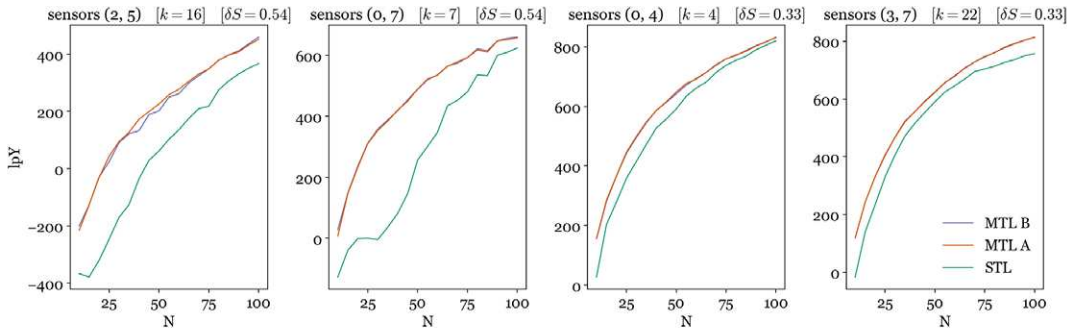

Figure 10. *Predictive log-likelihood for an increasing training budget for the hold-out experiments* $k \in \{4,7,16,22\}$. *Conventional single-task learning (STL) compared to multitask learning (MTL) which predicts hyperparameter values from similar experiments. Averaged for 100 repeats.*

the MTL models and include held-out experiments while increasing the training data. We favor the method presented here for computational reasons, referring again to the footnote[4].

These results demonstrate transfer learning since the information from the training experiments (source tasks) is used to improve prediction for held-out experiments (target tasks). However, both MTL-A and B provide (effectively) the same performance increase. This raises the question: is the more descriptive model for $p(\boldsymbol{\theta}|\phi)$ worth it, given that A provides the same predictive performance? We argue that this depends on the purpose of the model. If the purpose is purely prediction, the assumptions of A will likely be sufficient (in engineering, however, prediction is rarely the only motivation). On the other hand, B more closely resembles our understanding and domain expertise of the experimental campaign: it provides more insights regarding variations of the AE map between each experiment. In future work, stronger constraints could be applied to the parameters of the model directly (rather than hyperparameters). This structure would allow the meta-model to simulate the response $\mathbf{y}_k$ of unobserved experiments directly, and the inter-task functions would have more influence on predictive performance. However, these constraints would be more restrictive and require more specific domain knowledge, which was not available for these experiments.

6. Concluding remarks

In this work, we demonstrated how multilevel Gaussian Processes (GPs) can be used as meta-models to represent an experimental campaign for nondestructive testing (NDT). Two model formulations were used to represent a series of experiments concerning source localization with AE data for a complex plate geometry. While the same plate was used throughout the test campaign, the experimental design was varied (re. sensor placement). By learning all tasks in a joint inference, the representation captures how characteristics of the AE map (that is hyperparameters) vary between the experiments. The model can also share information between tasks, to extend the (effective) number of training data and their value. We presented the intertask relations and explained how they inform insights into systems-level behavior, allowing domain expertise to be encoded between experiments, relating to the effects of design variables on the outcome of each test. The intertask functions were used as meta-models to predict hyperparameter values of similar (previously unobserved) experiments and enhance inference in new tasks by transfer learning. Looking forward, more specific physics-based constraints could be encoded into multilevel representations (via the kernel function) to describe intertask variations with physics-based models, that is differential equations. Instead, this article encoded general domain expertise of the underlying process (via the GP mean function) such that data simulated from the model reflected our understanding of the environment and experiments.

Data availability statement. While the experimental data in this work are not currently available to share, the stan code is hosted on https://github.com/labull/EngineeringPatternRecognition.

Acknowledgments. LAB and MG acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) through the ROSEHIPS project (Grant EP/W005816/1). AD is supported by Wave 1 of The UKRI Strategic Priorities Fund under the EPSRC Grant EP/T001569/1 and EPSRC Grant EP/W006022/1, particularly the *Ecosystems of Digital Twins* theme within those grants & The Alan Turing Institute. EJC and MRJ are supported by grant reference numbers EP/S001565/1 and EP/R004900/1.

Author contribution. Conceptualization: LAB, MG, and MRJ.; Formal analysis: LAB and MRJ; Funding acquisition: EJC, AD, and MG; Investigation: LAB and MRJ; Methodology: LAB, MRJ, and AD; Software: LAB; Supervision: EJC, AD, and MG; Validation: MRJ; Writing, original draft: LAB; Writing, review & editing: MRJ and AD.

Funding statement. The support of Keith Worden during the completion of this work is gratefully acknowledged. Thanks are offered to James Hensman, Mark Eaton, Robin Mills, and Gareth Pierce for their work in generating the data used throughout this paper. For the purposes of open access, the authors have applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript version arising.

Competing interest. None.

Ethical standard. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

References

Gardner P, **Bull LA**, **Gosliga J**, **Dervilis N**, **Cross EJ**, **Papatheou E and Worden K** (2022) *Population-Based Structural Health Monitoring*, pages 413–435. NYC, USA: Springer International Publishing.

Rahwan I, **Cebrian M**, **Obradovich N**, **Bongard J**, **Bonnefon J-F**, **Breazeal**, **Crandall JW**, **Christakis NA**, **Couzin ID**, **Jackson MO**, **Jennings NR**, **Kamar E**, **Kloumann IM**, **Larochelle H**, **Lazer D**, **McElreath R**, **Mislove A**, **Parkes DC**, **Pentland AS**, **Roberts ME**, **Shariff A**, **Tenenbaum JB and Wellman M** (2019) Machine behaviour. *Nature*, *568*(7753):477–486.

Gelman A, **Carlin JB**, **Stern HS**, **Dunson DB**, **Vehtari A, and Rubin DB** (2013) *Bayesian Data Analysis*. CRC press, third edition.

Murphy KP (2012) *Machine Learning: A Probabilistic Perspective*. MIT press.

Kreft IG and De Leeuw J (1998) *Introducing Multilevel Modeling*. Sage.

Hensman J, **Mills R**, **Pierce S**, **Worden K, and Eaton M** (2010) Locating acoustic emission sources in complex structures using Gaussian processes. *Mechanical Systems and Signal Processing*, *24*(1):211–223.

Karniadakis GE, **Kevrekidis IG**, **Lu L**, **Perdikaris P**, **Wang S and Yang L.** (2021) Physics-informed machine learning. *Nature Reviews Physics*, *3*(6):422–440.

Willard J, **Jia X**, **Xu S**, **Steinbach M and Kumar V** (2020) Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, *1*(1):1–34.

Daw A, **Karpatne A**, **Watkins W**, **Read J and Kumar V** (2021) Physics-guided neural networks (pgnn): An application in lake temperature modeling. arXiv,.

Wahlström N, **Kok M**, **Schön TB and Gustafsson F** (*2013*) Modeling magnetic fields using Gaussian processes. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3522–3526. IEEE.

Alvarez M, **Luengo D and Lawrence ND** (2009). Latent force models. In *Artificial Intelligence and Statistics*, pages 9–16. PMLR.

Jones R, **Rogers TJ and Cross EJ** (2023). Constraining Gaussian processes for physics-informed acoustic emission mapping. *Mechanical Systems and Signal Processing*, *188*:109984.

Huang Y, **Beck JL, and Li H** (2019). Multitask sparse Bayesian learning with applications in structural health monitoring. *Computer-Aided Civil and Infrastructure Engineering*, *34*(9):732–754.

Huang Y and Beck JL (2015). Hierarchical sparse Bayesian learning for strucutral health monitoring with incomplete modal data. *International Journal for Uncertainty Quantification*, *5*(2).

Di Francesco D, **Chryssanthopoulos M**, **Faber MH and Bharadwaj U** (2021). Decision-theoretic inspection planning using imperfect and incomplete data. *Data-Centric Engineering*, *2*.

Papadimas N and Dodwell T (2021). A hierarchical Bayesian approach for calibration of stochastic material models. *Data-Centric Engineering*, *2*.

Hughes AJ, **Gardner P and Worden K** (2023). Towards risk-informed pbshm: Populations as hierarchical systems. *arXiv preprint arXiv:2303.13533.*

Sedehi O, **Kosikova AM**, **Papadimitriou C and Katafygiotis LS** (2023). On the integration of physics-based machine learning with hierarchical bayesian modeling techniques. arXiv *preprint arXiv:2303.00187*.

Shull PJ. (2002) *Nondestructive evaluation: theory, techniques, and applications*. USA: CRC press.

Tobias A. (1976) Acoustic-emission source location in two dimensions by an array of three sensors. *Non-destructive testing*, *9*(1): 9–12.

Jones MR, **Rogers TJ**, **Worden K and Cross EJ** (2022) A bayesian methodology for localising acoustic emission sources in complex structures. *Mechanical Systems and Signal Processing*, *163*:108143.

https://doi.org/10.1017/dce.2024.43 Published online by Cambridge University Press

Jones MR, **Rogers TJ**, **Worden K and Cross EJ**.(2022) Heteroscedastic Gaussian processes for localising acoustic emission. In *Data Science in Engineering, Volume 9: Proceedings of the 39th IMAC, A Conference and Exposition on Structural Dynamics 2021*, pages 185–197. Springer.
Hoffman MD, **Gelman A**, *et al.* (2014) The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, *15*(1):1593–1623.
Carpenter B, **Gelman A**, **Hoffman MD**, **Lee D**, **Goodrich B**, **Betancourt M**, **Brubaker M**, **Guo J**, **Li P and Riddell A** (2017) Stan: A probabilistic programming language. *Journal of statistical software*, *76*(1).
Wiemann PF, **Kneib T and Hambuckers J**.(2021) Using the softplus function to construct alternative link functions in generalized linear models and beyond. arXiv *preprint arXiv:*2111.14207.
Lee JD, **Sun DL**, **Sun Y, and Taylor JE** (2016). Exact post-selection inference, with application to the lasso. *The Annals of Statistics*, *44*(3): 907–927.
Rasmussen CE, **Williams CK**, *et al.* (2006) *Gaussian Processes for Machine Learning*, *1*. Springer.

Appendix

A. A Gaussian identity for GP prediction

Let $\mathbf{x}$ and $\mathbf{y}$ be jointly distributed Gaussian random vectors [27],

$$\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} \sim N\left(\begin{bmatrix}\mu_x\\ \mu_y\end{bmatrix}, \begin{bmatrix}A & C\\ C^\top & B\end{bmatrix}\right) = N\left(\begin{bmatrix}\mu_x\\ \mu_y\end{bmatrix}, \begin{bmatrix}\tilde{A} & \tilde{C}\\ \tilde{C}^\top & \tilde{B}\end{bmatrix}^{-1}\right)$$

The conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ is

$$\mathbf{x}|\mathbf{y} \sim N\left(\mu_x + CB^{-1}(\mathbf{y}-\mu_y), A - CB^{-1}C^\top\right)$$

This conditional is used in GP predictive equations, for a given/fixed set (that is sample) of hyperparameters. For further details, refer to Rasmussen et al. [27].

B. Pair label indices

Table 2. *Experiment indices k, sensor pairs, and their separation δS (Euclidean distance in normalized space)*

experiment index (k)	sensor pair	sensor separation (δS)
1	(1, 2)	0.18
2	(1, 3)	0.38
3	(1, 4)	0.21
4	(1, 5)	0.33
5	(1, 6)	0.39
6	(1, 7)	0.44
7	(1, 8)	0.54
8	(2, 3)	0.2
9	(2, 4)	0.16
10	(2, 5)	0.24
11	(2, 6)	0.43
12	(2, 7)	0.39
13	(2, 8)	0.44
14	(3, 4)	0.28
15	(3, 5)	0.28
16	(3, 6)	0.54

Continued

experiment index (k)	sensor pair	sensor separation (δS)
17	(3, 7)	0.42
18	(3, 8)	0.39
19	(4, 5)	0.12
20	(4, 6)	0.27
21	(4, 7)	0.24
22	(4, 8)	0.33
23	(5, 6)	0.27
24	(5, 7)	0.15
25	(5, 8)	0.22
26	(6, 7)	0.2
27	(6, 8)	0.39
28	(7, 8)	0.18

Cite this article: Bull LA, Jones MR, Cross EJ, Duncan A and Girolami M (2024). Meta-models for transfer learning in source localization. *Data-Centric Engineering*, 5, e48. doi:10.1017/dce.2024.43