Technical
Memo

ECMWF
European Centre for Medium-Range
Weather Forecasts

934

Bayesian optimisation of parameters in the ECMWF IFS

Annelize van Niekerk, Birgit Sützl, Nina Raoult,
Martin Janoušek and Ivan Bastak-Duran

December 11, 2025

Series: ECMWF Technical Memoranda

A full list of ECMWF Publications can be found on our web site under:
http://www.ecmwf.int/en/publications/

Contact: library@ecmwf.int

European Centre for Medium Range Weather Forecasts, Shinfield Park, Reading, RG2 9AX, UK

Abstract

Calibrating uncertain parameters is a necessary step in the development of comprehensive numerical weather prediction models such as the ECMWF Integrated Forecasting System (IFS), particularly when scientific changes have been made. We present a Bayesian optimisation tool and workflow that semi-automates parameter calibration within the fully coupled IFS using perturbed parameter ensemble forecasts and Gaussian Process (GP) emulators. The workflow uses the GP emulator to map from parameter values to a combined forecast-error metric. The tool then proposes new parameter values to sample with the IFS based on minimising the error metric using the GP emulator predictions, following a Bayesian optimisation approach. By iteratively sampling parameter combinations using the IFS, minimisation of the GP emulator yields several plausible optimal parameter sets. This helps avoid local minima and exposes parameter interdependencies.

We demonstrate the approach by calibrating five orographic drag parameters, after changes to the parametrisation scheme led to initially degraded large-scale scores. We show that the method is able to identify parameter sets that deliver neutral to improved upper-air skill and significant improvements in near-surface variables. The workflow is user configurable and applicable to many other parameters in the IFS. While choices of metrics and priors remain subjective and compensating errors can mask regional trade-offs, Bayesian optimisation offers a practical, systematic alternative to manual calibration and accelerates the path from scientific change to operationally acceptable performance.

1 Introduction

Due to the complex nature of the atmospheric system and the difficulties in observing its phenomena, numerical weather prediction and climate models contain many uncertain parameters. The European Centre for Medium-Range Weather Forecasts Integrated Forecasting System (ECMWF IFS) is no exception. Particularly in the physics parametrisation schemes, which are simplified approximations of unresolved or under-resolved processes based on theory or empirical measurements, there are large uncertainties and limitations in what is modelled. As a result, these schemes contain parameters with fixed values that in fact have a range of values that may be physically valid. What is more, there can be a strong interaction between parameters and processes that mean changing the value of one parameter should or could be compensated through a change in another parameter. While some of this uncertainty is accounted for through the Stochastically Perturbed Parametrisations (SPP) scheme in the ECMWF IFS probabilistic ensemble forecasting system (Lang *et al.*, 2021; Leutbecher, 2024), not all parameters or processes are perturbed and those that are still rely on a mean control value about which a distribution is defined for the perturbations. As a result, optimal deterministic values of the parameters must be sought through parameter calibration with observations of measurable quantities like wind and temperature - a process which is currently very time consuming and non-exhaustive.

In the development of the ECMWF IFS, as with many other models, calibration of uncertain parameters is routinely done (Mauritsen *et al.*, 2012). There are typically two situations in which this is desirable or necessary. Firstly, when biases relative to observations have been identified, one may be able to calibrate uncertain parameters in order to reduce these biases without making changes to the underlying physical model. Secondly, and perhaps a more complex case, to calibrate new parameters when new physics has been introduced into the model to add missing processes or improve physical realism. In the first case, some authors have shown that models are relatively well tuned for the mean climate (Ollinaho *et al.*, 2013b,a), suggesting that remaining biases are due to structural model errors (McNeall *et al.*, 2016) - these are errors that cannot be reduced through parameter calibration, or explained by observational uncertainty alone. In order to address structural errors, changes to model physics should be made to improve the model's fidelity. However, large changes to the model physics often lead to degradation of large-scale error scores, even when the new physics is more consistent with observations of a particular process and improve the representation of local processes (see for example the discussion in van Niekerk and Vosper (2021)). This degradation may be due to compensating biases from other processes, which can be difficult to disentangle in a complex model that has been tuned over many cycles. The typical approach to this problem is to select a handful of parameters that interact strongly with the particular physics changes that have been made, tune these parameters in a trial-and-error manner and hope that one has arrived at an optimally balanced new state. While this can work for simple problems with only a few degrees of freedom, it can be very time-consuming, since one must wait for the results of the first trial before deciding on the next set of parameters, and it is likely to be non-exhaustive in terms of exploring the parameter space. A more systematic and automated approach to parameter calibration is desirable (Raoult *et al.*, 2025).

In recent years, there has been a growing body of work aimed at making the calibration of NWP and climate models more systematic and automated, driven by increased computational capacity and advances in statistical emulation and machine learning. These methods include the use of emulator-based approaches to efficiently explore and constrain high-dimensional parameter spaces (Williamson *et al.*, 2013, 2015; Hourdin *et al.*, 2023; Raoult *et al.*, 2024). Emulator-based calibration has also been applied to models of different complexities and components of the Earth system, including climate models (Edwards *et al.*, 2011; Williamson *et al.*, 2013, 2015; Hourdin *et al.*, 2023), ocean models (Williamson *et al.*, 2017; Lguensat *et al.*, 2023), atmospheric models (Couvreux *et al.*, 2021; Hourdin *et al.*, 2021; Villefranque *et al.*, 2021), and ice sheet models (McNeall *et al.*, 2013). Similar developments have been applied to land surface models, where emulators and Bayesian model emulation techniques have been used to accelerate parameter estimation and improve model fidelity (Fer *et al.*, 2018; Dagon *et al.*, 2020). Together, these studies demonstrate a broader shift toward more efficient, data-informed calibration strategies that complement traditional expert-driven tuning.

The aim of the methodology presented here is to improve the efficiency and efficacy of the current method of calibrating uncertain parameters in the development of the ECMWF IFS, particularly when model changes are made. In this work, we use Bayesian optimisation (Rasmussen and Williams, 2006) to find optimal parameters that minimise the model error in fully coupled (land-atmosphere-ocean-waves) forecasts. The tools and workflow developed here are transferable to any global parameters and any number of parameters can be optimised simultaneously, although constraints on sample size and convergence should be considered. We demonstrate the approach by calibrating the parameters from the orographic drag schemes, namely the turbulent orographic form drag (TOFD) (Beljaars *et al.*, 2004), orographic flow-blocking drag and gravity-wave drag (Lott and Miller, 1997). This recalibration is motivated by the fact that the mean orography and subgrid fields used within these schemes have recently been updated to use a new source dataset and refined interpolation methods, and an error in the original code has been corrected (Beljaars *et al.*, 2022). As a result of these changes, there is a degradation of the large scale scores relative to observations, which can be seen in Figure C1 in the Appendix. In particular, the root mean square error (RMSE) of the 10m winds (10ff) and the 2m temperatures (2t) are degraded by as much as 3% over Asia and 1% over the entire Northern Hemisphere (NH). While these percentages may seem small, on the order of 1%, they are locally very large and would be regarded as unacceptable for operational implementation. The reason for this degradation is that the balance between the different drag components (flow blocking, gravity-wave drag, TOFD and resolved orography) has changed and the optimal parameters have shifted far away from the current control values - making it difficult to determine a priori how to begin calibration by hand. Orographic drag is a good candidate for multi-parameter calibration, with a manageable number (approximately 5) of strongly interacting uncertain parameters.

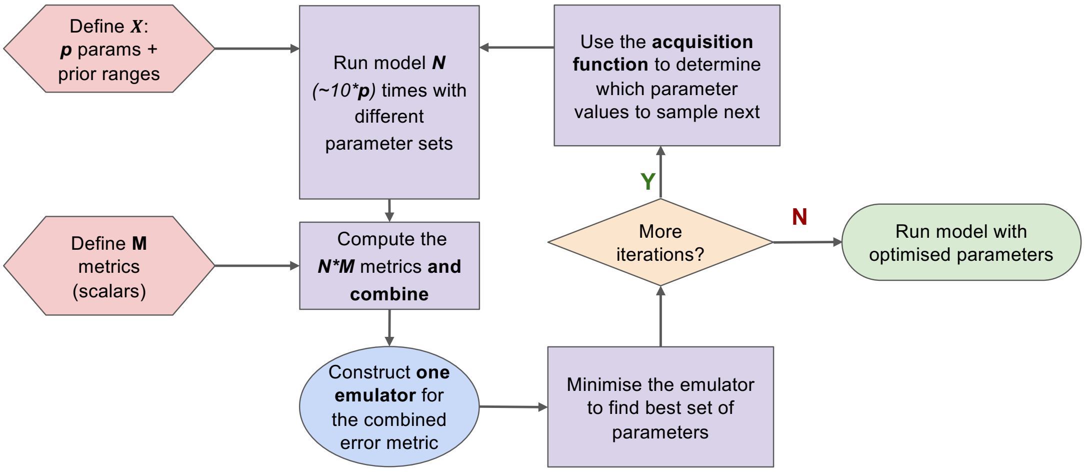

Define X: p params + prior ranges
Run model N (~10*p) times with different parameter sets
Use the acquisition function to determine which parameter values to sample next
Y
N
More iterations?
Run model with optimised parameters
Define M metrics (scalars)
Compute the N*M metrics and combine
Construct one emulator for the combined error metric
Minimise the emulator to find best set of parameters

Figure 1: Bayesian Optimisation workflow

The paper is laid out as follows. We begin with an overview of the Bayesian optimisation framework and workflow in Section 2.1. Sections 2.2 to 2.6 provide a more detailed description of its implementation using the IFS. Results from the optimisation of orographic drag parameters are presented in Section 3, split into single and multilple parameter optimisation examples in Sections 3.3 and 3.4, respectively. Section 4 summarises key findings, discusses shortcomings of the methodology and suggests extensions.

2 Methodology

2.1 Overview

The goal of Bayesian optimisation is to find the set of model parameters that lead to the smallest model error. In other words, we need to find the minimum of a function relating our parameters to the model error:

$$\mathbf{y} = f(\mathbf{X}) + \varepsilon \tag{1}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_i, ...\mathbf{x}_N]^T$ are the possible parameter input combinations (N), with $\mathbf{x_i} = [x_1, x_2, ..., x_p]$ being an array with length p, the number of parameters being optimised. $\mathbf{y} = [y_1, y_2, ..., y_i, ..., y_N]^T$ is a measure of the model error for each input, with some estimate of noise ε. The optimisation is then tasked with finding $\mathbf{x}^* = arg\min_{\mathbf{x}} f(\mathbf{X})$, where $\mathbf{x}^*$ is the set of optimal parameter values that minimise the model error. However, the function $f(\mathbf{X})$ does not have an analytic expression: the relationship between the parameters (p) and the model error ($\mathbf{y}$) is not known without running the model to evaluate y_i. Thus, $f(\mathbf{X})$ is treated as a 'blackbox' function.

In order to emulate this relationship between the parameters and the model error, we use what is sometimes referred to as a surrogate model. Specifically, a Gaussian process emulator (Section 2.4) is used to emulate the relationship between the parameters and the model error. The benefit of using a surrogate model is that it provides us with an analytic expression, in this case is a distribution that can be minimised. The Gaussian process emulator assumes that the function $f(\mathbf{X})$ takes a probabilistic form for each parameter combination modelled by a normal distribution, such that:

$$f(\mathbf{X}) \sim \mathscr{N}(\mu(\mathbf{X}), k(\mathbf{X}, \mathbf{X}')) \tag{2}$$

where $\mu(\mathbf{X})$ is the mean of the distribution and $k(\mathbf{X}, \mathbf{X}')$ is the covariance between the errors at the current parameter combinations, $\mathbf{X}$, and different locations in the parameter space, $\mathbf{X}'$. This function $f(\mathbf{X})$ is now a probability distribution to be sampled from and, since it can be sampled at any point in the parameter space, standard minimisation techniques can be used to find the optimal parameters.

To perform the training of the Gaussian process emulator, samples of the model's error must be mapped onto different parameter-value combinations. One could perform a very large number of deterministic model experiments with different global parameter values, varied about some distribution, and use this to train the emulator once. However, this would be very expensive and may mean densely sampling regions of the parameter space that are nowhere near the optimal values, making them redundant. What is more, the number of initial samples required for training and the degree of convergence of the Gaussian process emulator would be unknown. This is where Bayesian optimisation is a useful method for determining how to sequentially sample the parameter space of the full model, without over-sampling. In the Bayesian optimisation framework, outputs from the full model are iteratively added to the training of our Gaussian process emulator only where they are needed to improve the emulator's predictions in finding the minimal forecast error. The theory used in Gaussian processes emulation and sequential optimisation are documented in many papers, including Marchant *et al.* (2014) and Rasmussen and Williams (2006). In this work and other optimisation studies, the deterministic model experiments in which the parameters are varied about some distribution are referred to as a Perturbed Parameter Ensemble (PPE). The use of the word 'ensemble' here is not to be conflated with the probabilistic ensemble that is operationally performed at ECMWF, which has perturbed initial conditions and uses stochastically perturbed physics (Lang *et al.*, 2021).

The steps in our Bayesian optimisation approach are depicted in Figure 1 and are as follows:

1. Perform an initial Perturbed Parameter Ensemble (PPE) of model experiments in which the parameters are perturbed simultaneously within a given range (Section 2.2)

2. Calculate a model-error metric for each member of the PPE (Section 2.3)

3. Train the Gaussian processes emulator using the PPE members by mapping the parameter values to the errors (Section 2.4)

4. Find the minimum of the Gaussian process emulator and the corresponding input parameters (Section 2.5)

5. Use an acquisition function to determine which parameter values to sample next (Section 2.6)

6. Extend the PPE using these new parameter values (Section 2.2) and then return to step 2, iterating until convergence of the emulator has been achieved.

To semi-automate these steps, a user configurable Bayesian optimisation workflow and python package has been developed in-house. For an illustration of this workflow see Appendix A. The scikit-learn (Pedregosa, 2011) and scikit-optimize (Head, 2018) python packages (both based on NumPy and SciPy) are used to perform the Gaussian processes emulation, Bayesian optimisation and minimisation. The python package does the following: initialises the perturbed parameter values; sets up the error metric choices; trains the Gaussian process emulator; suggests the next parameter values to sample with the IFS; produces diagnostic plots; and outputs the optimal parameter values. This is integrated into a fully coupled suite that performs the PPE forecasts and computes individual forecast-error metrics for each member necessary for training of the Gaussian process emulator. This purpose-built suite treats each member of the ensemble as individual deterministic forecast, with the different parameter values as the only source of model perturbation.

Each step in the optimisation has several subjective choices that must be made, which have been carefully considered and tested. A brief description of the steps and choices are given in the following sections. While we aim for the choices to be as applicable and generalisable to as many different parameter sets as possible, optimal choices will ultimately vary based on the parametrisations, error metric and application considered. To this end, the specific example of optimisation of orographic parameters is documented in Section 3.

2.2 Generating and running the perturbed parameter ensemble

The first step is to identify the parameters for optimisation and to determine a suitable range for the parameter values. This step typically involves expert elicitation, by which previous experience and knowledge is harnessed to more confidently determine these subjective choices. The selection may also be complemented by sensitivity analysis using a wider set of parameters, see Appendix B for an example of sensitivity analysis. The initial PPE forecasts, that are used to train iteration 0 of the Gaussian process emulator, are generated by sampling the parameter space using a random uniform distribution. This generates a PPE, in which each member has different values of the parameters being optimised. The initial sampling of parameter values is generated using the purpose-built python package, which allows the user to easily define the parameters and their ranges and the number of initial samples. The purpose-built suite then reads in these parameter values to generate the PPE of deterministic forecasts and computes the forecast error metrics for each member. When using these types of methods, a common rule of thumb is to perform a PPE with a sample size that is roughly ten times the number of parameters (Loeppky *et al.*, 2009). Further iterations are run using parameter values suggested by the acquisition function, see section 2.6.

2.3 Computing a combined error metric

The Gaussian process emulator can handle several input parameters simultaneously but only predicts one output to minimise. The output should be some measure of the model error relative to analysis or observations for a given set of input parameters. Our workflow combines several error metrics from the deterministic PPE. Errors are computed for each variable (v) (such as wind and temperature) at a particular forecast lead time (l), altitude (h) and over different regional domains (d) (see for example those in Figure 2) for each PPE member (m). The error metrics are configurable by the user through the python package and are based on those typically used for forecast verification when cycle development is undertaken. The individual error metrics are then averaged over all the initialisation dates, producing a time-averaged metric denoted by $v_{l,h,d}(m)$. For example, the metric could be the root mean square error (RMSE) of the zonal wind relative to radiosonde observations ($v = RMSE(u)$) at 250hPa ($l = 250hPa$) 24 hours into the forecast ($l = 24$) averaged over the Northern Hemisphere ($d = NH$) for the first member ($m = 1$). The average metrics are subsequently combined into a single metric for each PPE member to train the Gaussian process emulator. The process of combining the error metrics aims to automate the human task of analysing scorecards such as those shown in Figure C1, mentally combining the different metrics in some subjective way and then deciding which experiments are the best. Our workflow also allows for optimisation of single metrics, if desired.

To fairly combine the error metrics, normalisation is necessary. This is because the variables being used in the error metrics may have different magnitudes, and normalising them allows them to have the same scale so that they do not dominate the combined error metric. For example mean sea level pressure may be thousands of hPa while winds may be tens of ms^{-1}. In standard verification of model changes, scorecards show the normalised difference of the RMSE compared with some benchmark forecast, such that the values represent a relative fractional change in the RMSE. For example, in Figure C1, the benchmark forecast is the current candidate for cycle 50r1 and the errors show the relative increase in the RMSE in the unoptimised experiment. To mimic this in the combined error metric, one option for normalisation is to use the relative error:

$$\tilde{v}_{l,h,d}(m) = \frac{v_{l,h,d}(m) - v_{l,h,d}(b)}{v_{l,h,d}(b)} \tag{3}$$

where $\tilde{v}_{l,h,d}(m)$ is the normalised error metric, $v_{l,h,d}(m)$ is the error in a given perturbed parameter ensemble member and $v_{l,h,d}(b)$ is the error in the benchmark experiment. A benefit of this normalisation is that it removes any error metrics that are not affected by the model changes or the parameter values. While this normalisation provides a good measure of the relative change in the errors, another normalisation may be more suited to different applications. We find that the normalisation matters for the final results and, as may be expected, this relative normalisation produces the best scorecards.

Another option is to normalise each error metric by its range:

$$\tilde{v}_{l,h,d}(m) = \frac{v_{l,h,d}(m)}{\max\left(v_{l,h,d}(m)\right) - \min\left(v_{l,h,d}(m)\right)} \tag{4}$$

This more equally weights each error metric, but may lead to over-emphasis of particular regions, lead times or heights where the error is already very large.

Each variable then has a weighting attached to it (w_v), as do the lead times (w_l), height levels (w_h) and domains (w_d). The user is able to configure the relative importance of each metric by changing these weights. The error metrics are then combined through a weighted averaging as follows:

$$\text{Error}(m) = \frac{\sum\limits_{v,l,h,d} \delta_{\{p_i<p\}}\, w_{v,l,h,d}\, \tilde{v}_{l,h,d}(m)}{\sum\limits_{v,l,h,d} w_{v,l,h,d}} \tag{5}$$

where the weights, $w_{v,l,h,d} := w_v \times w_l \times w_h \times w_d$, should each be set to values between 0 and 1, with lower values meaning that the metric is considered less important. Additionally, the impact of too-small sample size or large outliers on the combined error metric can be reduced by only retaining statistically significant values at some level p based on a paired t-test, denoted by $\delta_{\{p_i<p\}}$. Equation 5 produces a single error metric for each ensemble member, which can then be used to train the Gaussian process emulator.

2.4 Gaussian processes emulator

The probability distribution of the forecast error is defined using equation 2, our Gaussian process emulator. This requires modelling the mean and the variance of the distribution using some function, known as a kernel function. The kernel relates the mean and covariance at the current $\mathbf{X}$ point to the other points in the parameter space $\mathbf{X}'$. This allows for information from sampled points in $f(\mathbf{x}_i)$ to be propagated throughout the parameter space (see Rasmussen and Williams (2006) for a detailed description).

In this work, the kernel function is given the form:

$$k(\mathbf{X},\mathbf{X}') = \exp\left(\frac{-||\mathbf{X}-\mathbf{X}'||^2}{2L^2}\right) + \sigma_n^2 \tag{6}$$

also known as the radial basis function with White noise (σ_n^2) added. Here, $||\mathbf{X}-\mathbf{X}'||$ is the Euclidean distance between $\mathbf{X}$ and $\mathbf{X}'$, points within our parameter space. The hyperparameters σ_n^2 and L control the behaviour of the emulator, where L is a characteristic lengthscale that determines the correlation between two points. Smaller values of L mean that only points that are close together are considered similar. Increasing the value of L leads to smooth functions with small variances at large distances from the sampled points. Decreasing the value of L leads to more variability of the function over small distances in $\mathbf{X}$ and large variances (uncertainty) even at points close to those sampled.

While the lengthscale L and the noise σ_n^2 have a user defined starting value and range, the final values of the kernel's hyperparameters are optimised during fitting to the sampled values. The fitting uses an initial value for L and σ_n as a starting point and then randomly samples from a given range a number of times to find the best values. The choice was made to allow this fitting to choose different values of L for each input parameter, such that L is an array of length p (the number of input parameters). This led to much improved fitting of the Gaussian process emulator to the training data, evidently by a smaller RMSE of the emulator's predictions of forecast error for a small testing sample. This is likely due to the fact that the relationship between a given input parameter and the forecast error is different for each parameter, as will be seen, meaning that the correlation lengthscale should also vary for each parameter.

A prior distribution for $f(\mathbf{X})$ with mean $\mu(\mathbf{X}) = 0$ and variance of $\sigma^2(\mathbf{X}) = \mathbf{I}k(\mathbf{X},\mathbf{X}') = 1+\sigma_n^2$ ($\mathbf{I}$ is the identity matrix) is assumed. Since the prior assumes $\mu(\mathbf{X}) = 0$, the values of the sampled combined error metric used to train the emulator are normalised by removing the mean and scaling to unit-variance before training.

The Gaussian process emulator is trained on the PPE samples of the IFS by maximizing the log-marginal-likelihood function (this is the loss function) using a limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm, a quasi-Newton gradient-based method, with bounds (L-BFGS-B; Byrd *et al.*, 1995). This is the default method used by scipy and scikit-optimize, and was found to give a good fit of the emulator to the test data. After training, we have an updated posterior distribution that is sequentially updated at every iteration of the optimisation loop as new samples from the IFS are added.

2.5 Finding the minima

The ultimate goal of the Bayesian optimisation is to find the parameter inputs that minimise forecast error, i.e. $\mathbf{x}^* = arg\min_{\mathbf{X}} f(\mathbf{X})$. However, there may be several minima in $f(\mathbf{x})$, particularly if there are many input parameters, and there are different methods of minimising the function. One way of finding the minimum is by simply assuming that it has already been sampled, and defining $\mathbf{X}^*$ as the set of parameters that have the smallest error within our ensemble of model forecasts. This method bypasses issues related to uncertainty in the accuracy of the Gaussian process emulator.

Another method, which utilises the emulator itself, is to minimise the surrogate model using well known minimisation algorithms. The minimisation used by scipy by default is L-BFGS-B, and we make use of this method. The algorithm starts with an initial estimate of the optimal value $f(\mathbf{x}^*)$ and proceeds to iteratively refine that estimate using the gradients of the function. Once a minimum is found, this is mapped onto the Gaussian process emulator's prediction of the corresponding input parameters $\mathbf{x}^*$. However, the method may lead to the identification of local rather than global minima of the function $f(\mathbf{X})$ and may only find one minimum. As a result, we use a stochastic approach to produce several minima. The method is known as a basin-hopping algorithm (see Olson *et al.* (2012)) and, as the name suggests, it prevents identification of local minima close to the initial estimate. The minimisation algorithm used in the basin-hopper is also the L-BFGS-B. The steps are as follows:

1. Use the initial guess of $\mathbf{x}^*$ (best sampled) as the starting point of the minimisation algorithm, and find the nearest local minimum of $f(\mathbf{X})$ using L-BFGS-B to get a new corresponding $\mathbf{x}^*$
2. The new $\mathbf{x}^*$ is randomly perturbed
3. A new local minimum of $f(\mathbf{X})$ using L-BFGS-B is found using the perturbed $\mathbf{x}^*$ as a starting point
4. If the new $f(\mathbf{x}^*)$ is lower than the previous one, this is the new global minimum
5. Steps 1. to 4. are repeated a number of times (we choose 100 iterations) and the found minima are stored throughout the iterations only if they are within some acceptance criterion (currently uses Metropolis criterion, see Gelman *et al.* (2013))

We use the Basin-hopping method in this work and only the best few minima are selected, the number of which can be specified by the user. We also shows the best-sampled minimum in our results, which is often close to the optimal value from the minimisation.

Another consideration, which was found to make a significant difference to the minimum values found, was the choice of the function that is being minimised. Since the emulator function $f(\mathbf{X})$ is actually a distribution, we first need to define a one-dimensional function to minimise, with single outputs for each input of $\mathbf{x}$. An original implementation of the minimisation algorithm used the mean $\mu(\mathbf{X})$ of the function $f(\mathbf{X})$ as the objective function. However, this approach led to optimal values being found at the edges of the parameter space, where the mean tended towards the prior (which is assumed to be zero) because there are fewer observations in this region. Moreover, this approach lead to very unstable optimal values between iterations of the optimisation. To improve this implementation, we chose to make the objective function $f(\mathbf{X}) = \mu(\mathbf{X}) + \sigma(\mathbf{X})$, i.e. the mean with the standard deviation added on. This approach led to significantly more stable results between iterations and also much more reliable estimates of the optimal parameters. However, a risk of this approach is that we may not select the true minimum if the emulator has a large uncertainty. This drawback can hopefully be balanced by having enough members in our ensemble of forecasts.

2.6 Acquisition function

Once the posterior distribution has been updated using the PPE from the ECMWF IFS, a new set of suggested input parameters are required to run the next iteration of the model in order to update the Gaussian process emulator. Rather than randomly sampling from the range of the input parameters again, the use of an acquisition function allows us to determine which input parameters will best help us to sample the parameter space in regions where the emulator still has a large uncertainty or where we are close to the current minimum of the emulator function. The trade off between sampling in regions of large uncertainty versus regions close to the current minimum is known as exploration versus exploitation, respectively.

The acquisition function used in this work is the Expected Improvement (Jones *et al.*, 1998):

$$EI(\mathbf{X}) = (f(\mathbf{x}^*) - \mu(\mathbf{X}) - \xi)\Phi\left(\frac{f(\mathbf{x}^*) - \mu(\mathbf{X}) - \xi}{\sigma(\mathbf{X})}\right) + \sigma(\mathbf{X})\phi\left(\frac{f(\mathbf{x}^*) - \mu(\mathbf{X}) - \xi}{\sigma(\mathbf{X})}\right) \quad (7)$$

where $\mathbf{x}^*$ is the current best sampled set of input parameter values with error $f(\mathbf{x}^*)$, $\mu(\mathbf{X})$ and $\sigma(\mathbf{X})$ are from the Gaussian process emulator. Here, $\phi(X)$ is the probability density function of the Gaussian process emulator and Φ is the corresponding cumulative distribution function. The parameter ξ controls how much exploration versus exploitation of the parameter space one desires. The name ‘Expected Improvement’ is given to the function because it tells us how much we can expect to improve on our current best estimate of the parameters $\mathbf{x}^*$.

One can see that the Expected Improvement becomes larger where the emulator predicts values that are smaller than the current best estimate, i.e. $f(\mathbf{x}^*) > \mu(\mathbf{X})$. The Expected Improvement is also large where the uncertainty is large because this increases the width of the probability density function ϕ and the value of $\sigma(\mathbf{X})$. The first term in (7) is the exploiting term and the second term is the exploring term. The exploiting term is negative in regions where the mean is larger than the current best ($f(\mathbf{x}^*) < \mu(\mathbf{X})$), whereas, the exploring term is positive definite. As ξ is increased, the exploiting term is reduced and vice versa when ξ is reduced. For very large values of ξ the Expected Improvement becomes close to constant across the parameter space, meaning that the values chosen for the next set of samples may be spread evenly across the parameter space.

The next set of parameter values to sample using the IFS are chosen where the Expected Improvement is maximised. Since the Expected Improvement will only have one maximum, the ‘constant liar strategy’ is used to get multiple next samples of the parameter space ($\mathbf{X} = [\mathbf{X}_i, ..., \mathbf{X}_{ns}]^T$, where ns is the number of points we want to sample in our next iteration). This strategy has the following steps (Ginsbourger *et al.*, 2010):

1. Compute the Expected Improvement by randomly sampling the parameter space of the trained emulator $N = 10000$ times

2. Find the set of parameter inputs that maximise the Expected Improvement, $\mathbf{x}_i = arg\max_{\mathbf{X}}(EI(\mathbf{X}))$. This set of parameters is one of the next parameter combinations.

3. Make a copy of the Gaussian process emulator

4. Train the copy of the Gaussian Process emulator using $\mathbf{x}_i$, from the second step above, as the current sampled minimum of $f(\mathbf{x}^*)$, i.e. ”lie” to the maximisation algorithm. This makes the maximisation algorithm believe that the emulator has already sampled this point, has little to no

uncertainty there and that the expression $f(\mathbf{x}^*) - \mu(\mathbf{x}_i) = 0$ at that point. This prevents the parameter values $\mathbf{x}_i$ from being chosen for sampling again, since they would no longer maximise the Expected Improvement.

5. Repeat the steps 1. to 4. *ns* times using the Gaussian emulator that has been trained on a series of lies.

The result of this iterative process is that the suggested next samples should be distributed in parameter space. Once we have our set of next suggested parameter values, the PPE can be extended and used to update the Gaussian process emulator until satisfactory convergence of optimal parameters has been reached.

3 Results

3.1 Optimisation of orographic drag parameters

Table 1: List of orographic parameters being optimised, the ranges within which they are perturbed, their current values and a brief description of their role within the schemes.

Parameter	**Range**	**Current value**	**Description**
RTOFDALPHA α_{TOFD}	1.0 - 40.0	24.0	Proportional to TOFD
GKWAKE C_d	0.2 - 15.0	3.0	Proportional to flow-blocking drag
GKDRAG G	0.01 - 1.0	0.15	Proportional to gravity wave drag
GFRCRIT Fr_{crit}	0.1 - $\pi/2$	0.4	Determines partitioning between flow-blocking and gravity wave drag. Larger values increase h_{eff} and decrease z_{blk}
GRCRIT Ri_{crit}	0.1 - 1.0	0.25	Determines altitude of gravity waves breaking. Larger values lead to lower wave breaking

The Bayesian optimisation framework and tools can be used to optimise almost any set of global parameters within the ECMWF IFS. In this work, we estimate optimal parameters within the orographic drag schemes, namely: the turbulent orographic form drag (TOFD, Beljaars *et al.*, 2004), orographic flow-blocking drag and gravity-wave drag (Lott and Miller, 1997). This is motivated by the fact that the mean orography and subgrid fields used within these schemes have recently been updated and a fix has been implemented to address an error in the values of a_2 in equation 8, which was also present in the original manuscript of Beljaars *et al.* (2004).

The parameters being optimised are listed in Table 1, along with their ranges, current values and a description of their role. As an illustration of the optimisation we first only optimise a single parameter RTOFDALPHA from the turbulent orographic form drag scheme in Section 3.3. We then proceed to perform multiple-parameter optimisation on all those parameters listed in Table 1 in Section 3.4.

For context, we give a brief introduction to the orographic drag expressions and how the parameters being optimised fit within these. The expression for the vertical profile of the TOFD is:

$$\frac{\partial \overrightarrow{U}}{\partial t} = \frac{1}{\rho}\frac{\partial}{\partial z}\overrightarrow{\tau}_{TOFD} = -\alpha_{TOFD}\beta_{TOFD}C_{md}C_{corr}|\overrightarrow{U}|\overrightarrow{U}2.109e^{(z/1500)^{1.5}}a_2 z^{-1.2} \tag{8}$$

where α_{TOFD} is the parameter being optimised, β_{TOFD}, C_{md}, C_{corr} and a_2 are empirically derived, $\overrightarrow{U} = (U,V)$ is the wind vector and z is the height above ground. It is the empirical coefficient $a_2 = \sigma^2_{flt}(I_H k^{n_1}_{flt})^{-1}k_1^{n_1-n_2}$ that has been corrected to use updated values of I_H and k_{flt}, making a_2 approximately 4.8 times larger.

The expression for the orographic flow blocking drag vertical profile is:

$$\frac{\partial \overrightarrow{U}}{\partial t} = -C_d max(2-\frac{1}{r})\rho\frac{\alpha}{2\sigma}\left(\frac{z_{blk}-z}{z+\sigma}\right)^{\frac{1}{2}}(B\cos^2\psi + C\sin^2\psi)\frac{|\overrightarrow{U}|\overrightarrow{U}}{2} \tag{9}$$

where C_d is the flow blocking drag coefficient, r represents the aspect ratio as seen by the incoming flow, $\psi = \theta - \phi$ is the angle between the incoming flow (given by $U,V = \cos\phi, \sin\phi$) and the orientation of the subgrid orography (θ), γ is the anisotropy (or aspect ratio) of the subgrid orography, α is the slope of the subgrid orography, σ is the standard deviation of the subgrid orography, z_{blk} is the flow blocking depth, z is the height above the first model level, $B = 1 - 0.18\gamma - 0.04\gamma^2$ and $C = 0.48\gamma + 0.3\gamma^2$ account for anisotropy of the mountains.

The gravity wave surface stress parallel ($||$) and perpendicular ($\perp$) to the surface winds are given by:

$$\tau_{||}, \tau_{\perp} = G\rho_H U_H N_H \frac{h^2_{eff}}{4}\frac{\alpha}{\sigma}(B\cos^2\psi_H + C\sin^2\psi_H, (B-C)\sin\psi_H\cos\psi_H) \tag{10}$$

where G is a tuning coefficient, h_{eff} is the effective mountain height and subscript H indicates that the variable has been averaged over a height of σ to 2σ. The drag is partitioned into orographic flow blocking drag and gravity wave drag through the division of the mountain height into a blocked layer (z_{blk}) and an effective mountain height (h_{eff}). The blocking depth z_{blk} is given by satisfying the inequality:

$$\int_{z_{blk}}^{3\sigma}\frac{N(z)}{U(z)}dz \geq Fr_{crit}. \tag{11}$$

The effective mountain height is then given by $h_{eff} = 3\sigma - z_{blk}$. This means that these two forms of drag are intimately connected through the parameter Fr_{crit} and the standard deviation of the subgrid orography σ.

The gravity-wave amplitude $\eta(z)$, initially set to $h_{eff} = \eta(z_0)$ near the surface (z_0), grows with altitude z depending on the winds, stability and density of the background flow. The wave is then assumed to break and deposit momentum (drag) to the mean flow when the wave modified local Richardson number ($\tilde{Ri}$) falls below the critical Richardson number (Ri_{crit}):

$$\tilde{Ri} = Ri\frac{1-\frac{N\eta}{|U|}}{(1+Ri^{\frac{1}{2}}\frac{N\eta}{|U|})^2} < Ri_{crit} \tag{12}$$

where $Ri = N^2/(dU/dz)^2$ is the local gradient Richardson number. The wave amplitude is then reduced such that $\tilde{Ri} = Ri_{crit}$ and the wave can continue propagating in the vertical. The gravity-wave stress is reduced such that it is proportional to the squared amplitude at each height $|\tau| \propto \eta^2$, and the gravity-wave drag is given by:

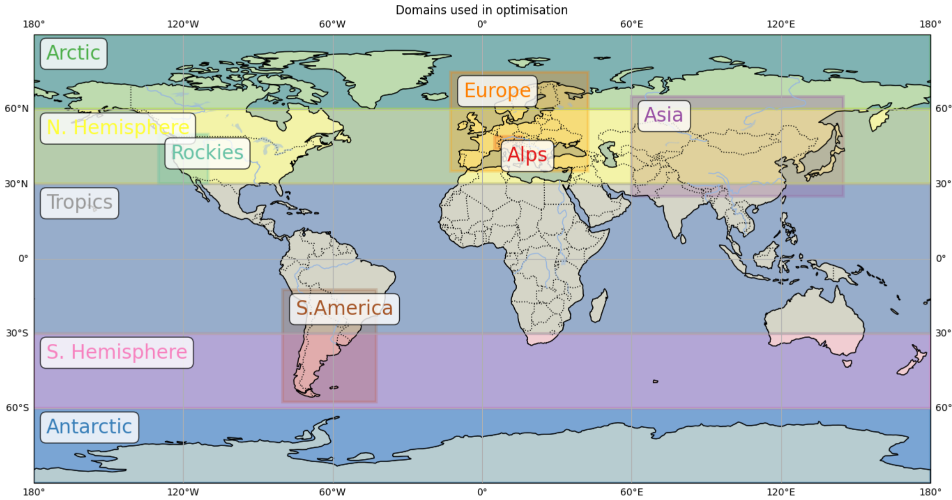


Figure 2: Domains over which the error metrics are calculated

$$\frac{\partial \vec{U}}{\partial t} = \frac{1}{\rho}\frac{\partial}{\partial z}\vec{\tau}_{GWD} \tag{13}$$

where $\vec{\tau}_{GWD}$ is the orographic gravity-wave stress vector projected into the zonal and meridional wind components. It is clear that there is likely to be a strong interaction between the different parameters, since they all impact the winds over orography. There is also a non-linear interaction between the flow blocking and gravity wave drag through Fr_{crit}, which impact h_{eff} and Ri_{crit} through its dependence on the wave amplitude. This makes the orographic parameters an excellent candidate for multiple parameter optimisation, which allows us to capture these interactions.

3.2 Perturbed parameter model setup

Each member of the PPE is run for 5 days, initialised at 00UTC. The forecasts are performed using ECMWF IFS cycle 50r1 at a resolution of Tco399 (28 km) and are initialised from an analysis at the same resolution and cycle. When optimising only the single parameter RTOFDALPHA, the forecasts are run from 20211201 to 20211229, with the forecast initialisation dates separated by 4 days, resulting in 8 initial dates. When optimising multiple parameters, the forecasts are run over two seasons, June to August (JJA) 2022 and December to February (DJF) 2022-2023, with the forecast initialisation dates separated by 48 hours (2 days). This allows us to sample enough synoptic and large scale variability, so that we do not over-fit to a certain regime or flow condition. This results in 86 initial dates (43 for JJA and 43 for DJF).

The purpose-built suite that generates the PPE forecasts is configured to compute mean absolute error (MAE) and root mean square error (RMSE) relative to analysis and observations, which are used by the emulator in the combined error metric. The RMSE and MAE of a particular variable relative to analysis, averaged over a particular domain (e.g. Figure 2) is computed as follows:

Table 2: Error metrics used for single parameter optimisation. Variables are described in Table C1.

Metrics for single-parameter optimisation				
Variable	**Error Metric**	**Heights**	**Domains**	**Lead times**
msl	MAE against analysis	Surface	NH, SH Tropics	+72–120 h 12-hourly
u,v, vw	MAE against analysis	850, 700, 500, 250, 200, 100, 50, 30, 10 hPa	NH, SH Tropics	+72–120 h 12-hourly
t	MAE against analysis	850, 700, 500, 250, 200, 100, 50, 30, 10 hPa	NH, SH Tropics	+72–120 h 12-hourly
z	MAE against analysis	500 hPa	NH, SH	+72–120 h 12-hourly

$$RMSE_d = \sqrt{\frac{\sum_{i=1}^{N_{pts}} \cos(\phi_i)(F_i - A_i)^2}{\sum_{i=1}^{N_{pts}} \cos(\phi_i)}}, \tag{14a}$$

$$MAE_d = \frac{\sum_{i=1}^{N_{pts}} \cos(\phi_i)|F_i - A_i|}{\sum_{i=1}^{N_{pts}} \cos(\phi_i)} \tag{14b}$$

where N_{pts} is the number of 1.5 by 1.5 degree grid-points in the considered domain, ϕ_i is the latitude at that point, F_i is the variable in the forecast (at a particular height, time and point) and A_i is the same variable from some reference analysis.

The RMSE and MAE of a particular variable relative to observations averaged over a particular domain (d) is computed using:

$$RMSE_d = \sqrt{\frac{\sum_{i=1}^{N_{obs}} (F_i - O_i)^2}{N_{obs}}}, \tag{15a}$$

$$MAE_d = \frac{\sum_{i=1}^{N_{obs}} |F_i - O_i|}{N_{obs}} \tag{15b}$$

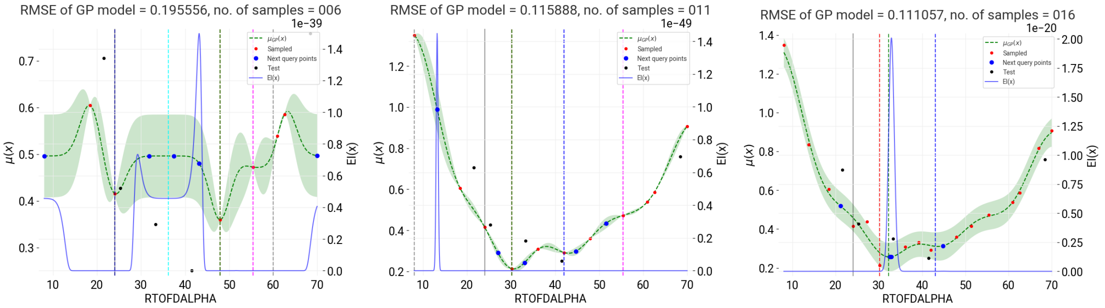
where F_i is the forecast variable at the nearest grid-point to the observation point, O_i is the observation at some location and N_{obs} is the number of observations contained within the particular domain. The observations used are from SYNOP (surface synoptic observations) stations for surface observations and from vertical soundings, both of which are over land. This makes the observational metrics unsuitable for large changes over ocean, for which other observations can and should be used. For orographic changes, however, these observations are ideal.

3.3 Single parameter

Since the optimisation of several parameters can be both difficult to visualise and requires several iterations with many members of the model, we initially demonstrate the optimisation method outlined above using only one drag parameter. The parameter RTOFDALPHA (α_{TOFD}) was chosen because it was found to have a relatively simple relationship with the combined error metric. The error metrics are mean absolute error (MAE) of the variables listed in Table 2 against analysis, with all weights set to 1 in equation 5. We do not compute the relative error but, instead, normalise each error metric by its range as in equation 4. The initial sample size was chosen to be 11, including the control forecast with the current value of $\alpha_{TOFD} = 24$. Four members of the ensemble were held out for testing, leaving six members for training in the 0th iteration. At consecutive iterations, five new samples were added from the full model and used to update the Gaussian process emulator. In order to demonstrate convergence over a few iterations, we choose to use a lenghtscale $L = 0.1$ and range $1e-4 < L < 0.1$. This purposefully makes the relationship between the parameters and the error less smooth, since a smooth function led to rapid convergence towards an optimal value. Since this is for illustrative purposes, we wanted to make the convergence more difficult. As with the multi-parameter optimisation, the noise level (σ_n^2) is set to the variance across the enseble of forecasts σ_n^2. In this example we have not changed the orographic fields and have not corrected the bug in the parameters within the TOFD scheme (see section 3.1), such that the final optimal values are not comparable to those in the next section.

Figure 3 shows the prediction from the Gaussian process emulator at consecutive iterations of the optimisation steps. One sees from the 0th iteration that, in regions far from sampled points, the Gaussian process emulator prediction tends towards the prior, with a mean equal to the mean across the ensemble and a standard deviation of $\sqrt{1+\sigma_n^2}$. We see that the optimal minima (given by the vertical lines) are found to be close to points that have been sampled, because we include the uncertainty of the emulator in the function that we are minimising. Figure 3 also shows the Expected Improvement (EI) acquisition function for each iteration and the next points to sample. Where the Expected Improvement is large, the acquisition function suggests the next points to be sampled. Since the value of $\xi = 1.0$ for the 0th iteration is a relatively large value, the acquisition function maximises in regions where the uncertainty is the largest and therefore 'explores' the parameter space. Upon sampling these next points, the next iteration (iteration 1) shows a smaller uncertainty in the Gaussian process predictions and the found minima are different compared with the previous iteration. What is more, the acquisition function in the 1st iteration now samples closer to the minima of the function. By the 2nd iteration, the final iteration in this case, the found minima are relatively close to the best sampled experiment. Since the noise level, i.e. the uncertainty, in the Gaussian process emulator is tuned throughout the training process, we find that the uncertainty bands actually increases a little between iterations 1 and 2, to account for the variability across the ensemble members that are relatively close to each other in parameter space. This increased uncertainty is a beneficial impact since the variability may be coming from internal variability of the model.

Since this optimisation is performed using the model without physics changes, we do not expect the parameter RTOFDALPHA to change greatly, seeing as the model has previously been tuned. Nonetheless, we do see that some improvement could be gained from increasing this parameter from 24.0 to 30.16. It is interesting to see the relationship between the RTOFDALPHA parameter and the error emerging as a quadratic, with too-large values or too-small values of RTOFDALPHA leading to larger errors. By the 2nd iteration, the emulator seems close to convergence with only 16 samples. This rapid convergence is because we are only learning the relationship with one parameter. Optimisation of multiple parameters at the same time is likely to lead to more complex relationships, as they can interact nonlinearly and are interdependent, which require many more ensemble members.



Optimal values for interation 0

	RTOFDALPHA
control	24.0
Best sampled	47.942
1	47.938
2	55.422
3	60.822

Optimal values for interation 1

	RTOFDALPHA
control	24.0
Best sampled	30.16
1	30.161
2	24.019
3	47.94

Optimal values for interation 2

	RTOFDALPHA
control	24.0
Best sampled	30.16
1	30.758

Figure 3: The Gaussian process emulator prediction of the mean $\mu(\mathbf{X})$ and standard deviation (shaded region) of the combined error for a range of $RTOFDALPHA$ values for (from left to right) iteration 0, 1 and 2 of the optimisation are shown in green. Sampled values, test points and parameters to be sampled in the next ensemble of model experiments (Next query points) are indicated by the legend. The Expected Improvement ($EI(x)$) acquisition function is shown in a blue solid line with values on the right-hand-side axis. The vertical dashed lines are the current parameter values (black) and the optimal values produced from the minimisation. The vertical lines correspond in colour to the values presented in the tables. The 'Best sampled' is the parameter values of the overall best member of the ensemble. The optimal values in the table are ordered from smallest to largest predicted error.

Table 3: Error metrics used for multiple parameter optimisation. Variables are described in Table C1.

Metrics for multiple parameter optimisation				
Variable	**Error Metric**	**Heights**	**Domains**	**Lead times**
u, v, vw	RMSE against observations	850, 700, 500, 250, 200, 100, 50, 30, 10 hPa	See Figure 2	+12–120 h 12-hourly
t	RMSE against observations	850, 700, 500, 250, 200, 100, 50, 30, 10 hPa	See Figure 2	+12–120 h 12-hourly
10ff	RMSE against observations	10 m above surface	See Figure 2	+12–120 h 12-hourly
2t	RMSE against observations	2 m above surface	See Figure 2	+12–120 h 12-hourly
z	RMSE against observations	500 hPa	See Figure 2, excluding Tropics	+12–120 h 12-hourly

3.4 Multiple parameters

In this section we discuss the results from optimising all five parameters listed in Table 1. The error metrics are listed in Table 3 and the domains over which they are computed are shown in Figure 2. The error metrics are normalised relative to the benchmark experiment, which is the default CY50R1 without changes to the orography, using equation 3. Note that each domain is normalised relative to its own error. Additionally, we only retain statistically significant values at the 95% ($p < 0.05$) level when combining the metrics and set all weights to 1 in equation 5.

The initial ensemble contains 100 PPE members, with 5 samples held out for testing, and subsequent iterations contain 50 members. We choose to use a large initial PPE sample size to improve coverage of the parameter space and better account for complex interactions between parameters. After 4 iterations we have a total of 300 members, although, as shall be seen, fewer than this was required to achieve convergence. Initial values of the emulator's characteristic length scale $L = 0.2$ and noise term $\sigma_n = \sigma_m$, where σ_m is the standard deviation of the error across the members, and ranges of $1e-4 \leq L \leq 1$ and $1e-5 \leq \sigma_n \leq 1$ were found to perform well in the multiple parameter optimisation. However, it should be noted that optimal values of these hyperparameters could vary between applications, parameter ranges and optimisation metrics. We gradually decrease the value of ξ in equation 7 at each iteration, in order to exploit the parameter space and sample more closely to the minimum, but only once we have enough samples to trust the estimate from the emulator.

A more creative approach to presenting the parameter-error relationships and the convergence with iterations is required with many parameters. Figure 4 shows the partial dependence of the error metric between the five parameters for the first (0th) and last (4th) iteration of the optimisation. The scatter points show where the model has been sampled. The contours and line dependence plots are produced from predictions of the combined error metric by the trained Gaussian process emulator, through varying the parameters. Each contour plot shows the predicted relationship between two parameters. These parameter relations are produced by sampling the Gaussian process emulator at 100 evenly spaced points along each parameter dimension, while randomly varying all other parameters 100 times for each parameter combination. This evaluation produces a distribution at each parameter pair, of which the average is taken. The line plots are produced in the same way but by only sampling along one parameter dimension, while randomly varying all other parameters. Similarly, we can see the uncertainty in the emulator by looking at the standard deviation (as opposed to the mean) of the predicted combined error metric across

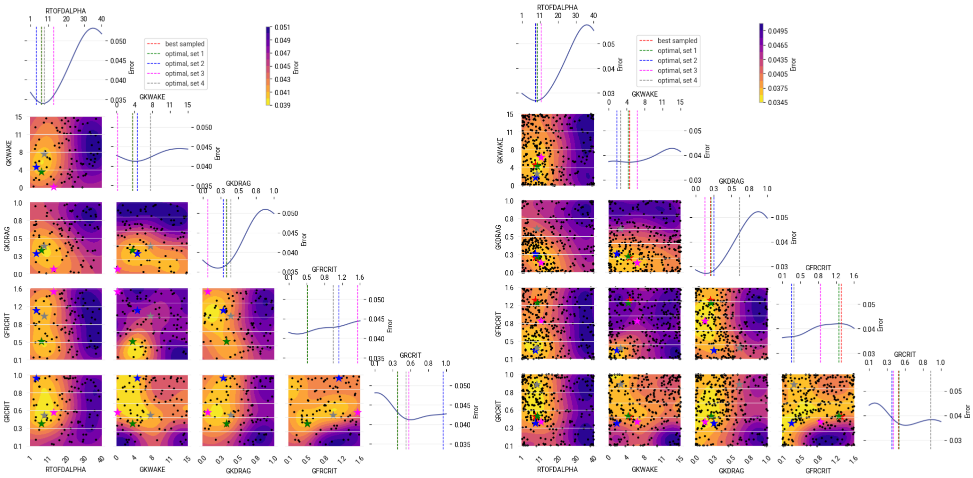


Figure 4: Partial dependence plots of the predicted mean of the combined error metric ($\mu_{GP}(\mathbf{X})$), as predicted by the Gaussian process emulator. The relationship for each parameter is shown in line plots and the relationship between the parameters themselves are shown in contours, for (left) iteration 0 and (right) iteration 4. The black scatter points show the sampled members.

the parameter space, shown in Figure 5.

Figure 4 shows that there is a relatively strong dependence of the forecast error on RTOFDALPHA and GKDRAG, which is evident already from iteration 0. From the line plots showing the dependence of each parameter individually, the form of the relationship with the combined error seems very similar for these two parameters. Both show an increase in the error at large values and a slight increase in error at very small parameter values. Their similarity is not surprising given that both are a linear multiplier of the drag that is exerted on the winds. However, the RTOFDALPHA acts near the surface while GKDRAG, being the multiplier of the gravity-wave drag, acts predominantly in the stratosphere. Since the error metric contains observations of near surface winds and winds from soundings, both of these impacts are captured in the error metric. What is surprising is that such a clear relationship does not emerge for GKWAKE, since this parameter is also a linear multiplier of the flow-blocking drag near the surface. It is possible that there is a less clear relationship emerging for GKWAKE because the error is dominated by the response to RTOFDALPHA and GKDRAG, and so averaging over randomly sampled values leads to a dilution of the signal. The parameter GFRCRIT also shows a less significant relationship with the error, although there is an indication that larger values lead to a larger error. This makes physical sense since larger values would increase the gravity-wave drag. Increasing GRCRIT leads to a smaller combined error but shows little impact on the error beyond a certain value of ∼0.5.

It is also clear that there are some relationships emerging between the parameters (contour plots in Figure 4). For example, there is a relationship between RTOFDALPHA and GKDRAG such that neither should be very large. Conversely, larger values of GRCRIT should be balanced by smaller values of RTOFDALPHA. This balance likely is required because increases to GRCRIT lead to lower wave breaking, which can more readily affect the error nearer the surface where the TOFD acts. The relationship between GFRCRIT and GRCRIT shows that small values of GFRCRIT should be balanced by a moderate increase in GRCRIT. This balance emerges because decreasing GFRCRIT decreases the gravity-wave amplitude and smaller amplitude waves would break higher up. In order to get the correct wave-breaking altitude, the critical Richardson number should be increased to encourage wave breaking at low enough altitudes.

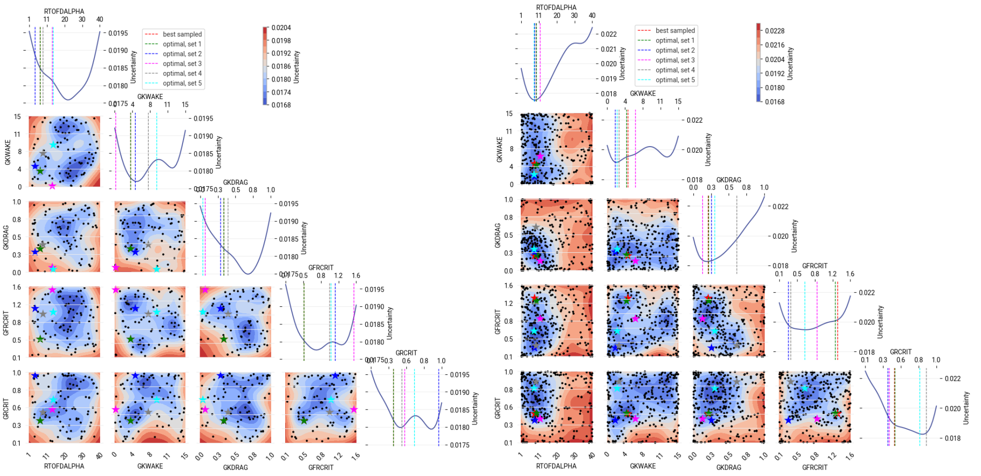


Figure 5: Partial dependence plots of the predicted standard deviation of the combined error metric ($\sigma_{GP}(\mathbf{X})$), as predicted by the Gaussian process emulator. The relationship for each parameter is shown in line plots and the relationship between the parameters themselves are shown in contours, for (left) iteration 0 and (right) iteration 4.

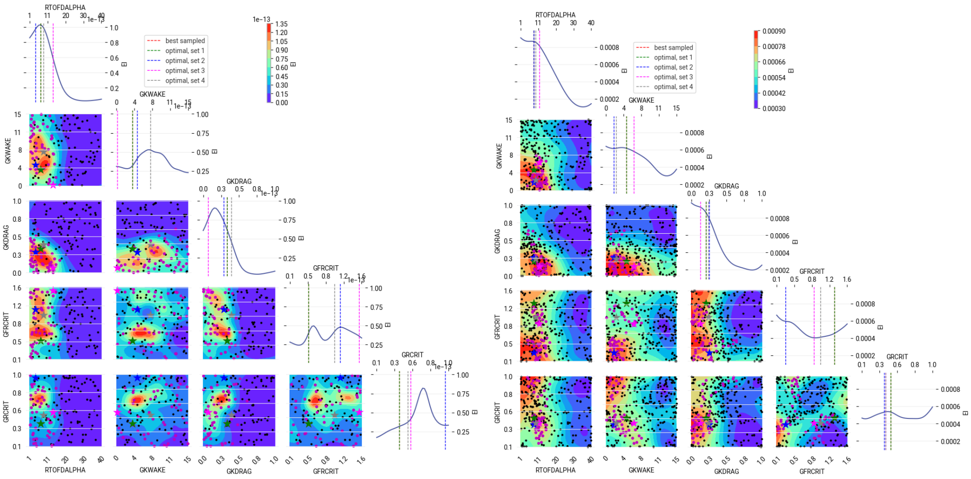


Figure 6: Partial dependence plots of the expected improvement acquisition function, as given by equation 7. The relationship for each parameter is shown in line plots and the relationship between the parameters themselves are shown in contours, for (left) iteration 0 and (right) iteration 3.

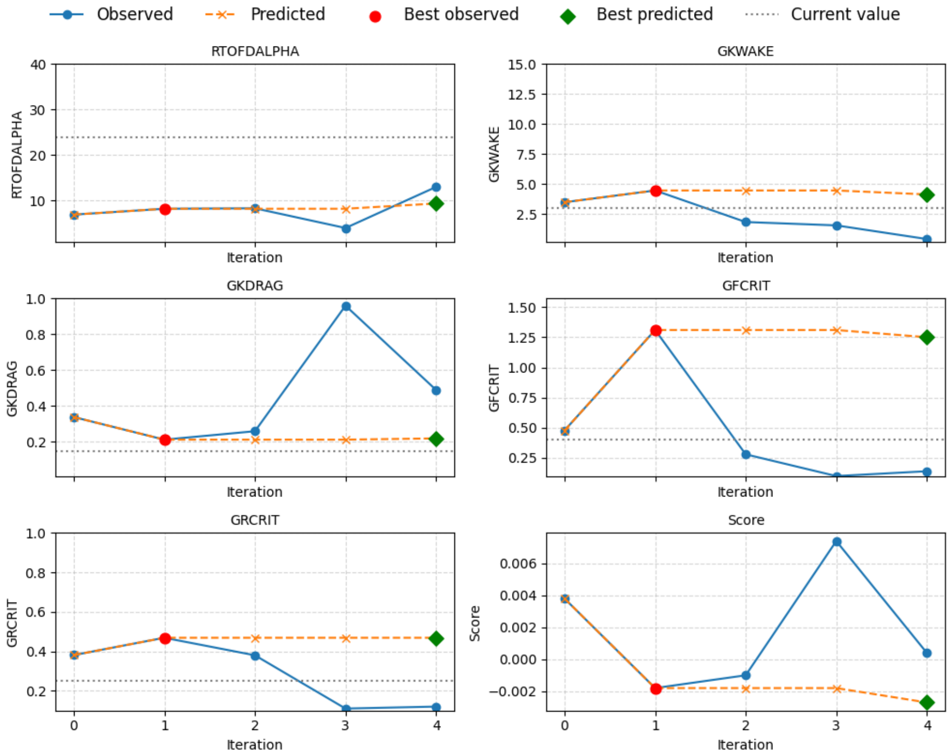


Figure 7: Values of the parameters and scores from the best sampled member (solid blue) and the best predicted values from the Gaussian process emulator (dashed orange), from each iteration. The current parameter values are shown in the dashed grey line. The overall best sampled values are shown in a red dot and the overall best predicted from the emulator are shown in a green diamond.

The partial dependence of the error metric and the parameters should be understood in conjunction with the uncertainty of the emulator, Figure 5. At the edges of the parameter space, the emulator is not well sampled, leading to large uncertainty. The parameter dependence of the mean may therefore be untrustworthy there. We see from the partial dependency of the uncertainty that between iteration 0 to iteration 4 the uncertainty has not reduced significantly in regions where the errors are large. The emulator's uncertainty has not decreased here because the acquisition function, the expected improvement, is small in these regions and few observing samples were suggested there.

The partial dependence of the acquisition function is shown in Figure 6 for iteration 0 and iteration 3 (the penultimate iteration). Indeed, the expected improvement is relatively small where the errors are large. At iteration 3 (the penultimate iteration) $\xi = 0$ such that the acquisition function is largest at the minimum of the combined error metric and the parameter space is exploited. From the magenta scatter points in Figure 6 at iteration 3, one can see that the acquisition function has led to dense sampling in regions close to the expected minimum with smallest forecast errors and little additional sampling in the regions of very large forecast errors. This efficient sampling of the parameter space is exactly why the Bayesian optimisation is a useful method of sampling, so that convergence towards the optimal parameters can be achieved without wasting computing resources.

Clearly, the parameter relationships are non-linear and complex but there is an indication that the Gaussian process emulator is able to capture these. Further evidence of the accuracy of the GP emulator is its ability to predict the five unseen 'test' samples of the model, these are members that were held out of training. At iteration 0, 1, 2, 3 and 4 the RMSE of the emulator relative to the test data was 0.010096, 0.007902, 0.007599, 0.005913 and 0.005021, respectively. While we see a clear improvement in the predictions made by the emulator, it is worth stating that the the fidelity of the Gaussian process emulator is not the goal of the optimisation. The predictions by the emulator may have large errors where the sampling is less dense, and the sampling may be sparse in regions where the acquisition function is small.

The goal of the optimisation is, ultimately, to find the combination of parameters that minimise the forecast-error metric. In Figures 4 and 5 the best-sampled parameters, i.e. the best member of the PPE, and the best 4 minima of the Gaussian process emulator, as retrieved using the basin-hopping algorithm, are shown in vertical dashed lines. At iteration 0, there is already an indication that the optimal values for RTOFDALPHA and GKDRAG have converged, since the lines are very close to each other. As mentioned before, this early convergence is likely because these two parameters dominate the error across the perturbed parameter ensemble. The optimal values for GKWAKE, GFRCRIT and GRCRIT, however, are still spread out in the parameter space at iteration 0, indicating that further sampling is required. By iteration 4, there is further convergence in the optimal values, particularly GRCRIT and GKWAKE. Since GFRCRIT does not have a very strong relationship with the forecast error, the optimal values for this parameter do not fully converge even by the 4th iteration. The fact that our method allows us to output several minima is a real advantage of this optimisation approach in this situation over other approaches, such as those using Kalman filters (Annan *et al.*, 2005), that converge only to one minimum. It allows for several minima with very different values of GFRCRIT.

In order to identify convergence of the optimisation, we output the parameter values of the best members in each iteration, giving that member a score, i.e. the value of the combined error metric. We also output the parameters that provide the global minimum of the Gaussian process emulator (i.e. the best parameters), along with their predicted score, at each iteration. A larger negative value of the combined error metric (score) means a smaller RMSE relative to the benchmark experiment. This information is presented in Figure 7, with the red and green points indicating the overall best sampled and predicted, respectively. One sees that by iteration 1, we have actually already converged on the best sampled optimal parameters, since the additional model samples provide training data for the Gaussian process emulator but none have a score lower than -0.0018. The optimal parameters predicted from the emulator also show some convergence by iteration 1, remaining constant up to iteration 4. At iteration 4 there is a small changes in the optimal values of GKWAKE, GKDRAG, GFRCRIT and GRCRIT, accompanied by a moderate change of RTOFDALPHA which leads to a small reduction in the predicted score. We can, therefore, conclude that we have essentially converged on the optimal parameters at iteration 1, i.e. with 150 perturbed parameter members, for our chosen error metric.

3.5 Validation

An additional step in this optimisation process is the validation of the optimisation's best estimate of the parameters. This step is not strictly part of the optimisation loop but provides a useful check of convergence and efficacy of the optimisation process. In order to demonstrate the robustness of our optimisation, we further perform analysis type experiments, in which the final optimised parameters are used in the background model with which the 4-dimensional variational data assimilation is run. This means that, in regions where the observations are sparse, the analysis will be closer to the model with the optimised parameters and so will be initialised in a consistent way. Additionally, this allows us to verify the model against its own analysis which has ingested many more observational types than those used to compute the RMSE against observations in Table 3. The robustness is further verified in these experiments by increasing the number of samples and initialising the model twice-daily at 00UTC and 12UTC each day of JJA 2022 and DJF 2022 to 2023 and running the forecast into the medium range up to 10 days. The extended forecast range provides a measure of the applicability of optimising at lead times of 12-120 hours for longer lead times.

Figure 8 shows the percentage change in the RMSE against observations, relative to the benchmark simulations (i.e. the current model version without changes to the orographic fields or recalibration of parameters) for key surface variables over certain regions. The unoptimised experiment and experiment with the optimal parameters are shown. The unoptimised forecast clearly see a degradation in many surface variables, with a particularly large degradation of the 10m winds over the NH, Asia and Europe and, similarly, a large degradation of 2m temperatures over the NH and the Rockies. After optimisation, both the 2m temperatures and 10m winds are improved relative to the benchmark experiment over most regions, excluding the tropics. While the 2m temperature change remains neutral over the Tropics, the winds are degraded by $\sim$1%. This degradation is understood to be due to a region over East Asia where the wind speeds are too strong due to other model errors and, as a result of the increased wind speeds over orography, the mean error is increased there. It is interesting to note that the 2m temperatures over the SH and Europe, although the variability is large, begin to degrade at longer lead times, suggesting that the optimisation at lead times of 12-120 is not necessarily suitable for those regions. Many regions do, however, see an improvement at longer leadtimes. Nonetheless, the optimisation has resulted in an overall improvement of both near surface variables and upper air variables, which can be seen in the full scorecard showing the change in the normalised RMSE of the optimised experiment relative to the unoptimised experiment in Figure 9. For completeness, the individual scorecards of the unoptimised and optimised experiments relative to the benchmark experiment are shown in Figure C1 and Figure C2, respectively. The changes to the orographic fields and code are now suitable for implementation in an operational forecasting model and the optimisation has allowed us to achieve this.

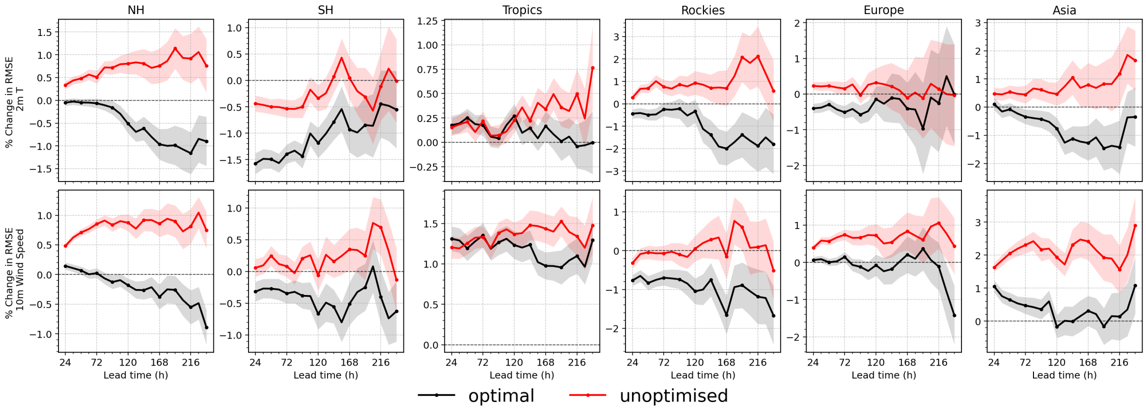


Figure 8: The % change in RMSE against observations, relative to the benchmark experiment, of the (top) 2m temperatures and (bottom) 10m winds as a functon of lead time in hours. The analysis experiments with unoptimised orographic parameters (red) and final optimal parameters (black) are shown. Results are combined from the JJA 2022 and DJF 2022/23 analysis experiment.

4 Discussion and Conclusions

We have shown that a more automated and systematic approach to calibration of parameters in a complex system, such as the ECMWF IFS, is possible through Bayesian optimisation using Gaussian process emulators. To this end, we have developed an in-house optimisation tool that can be used on any set of parameters, whether they be in the land-surface, atmosphere, ocean or wave components of the IFS. We have described and rationalised the methodology behind the tool, while demonstrating its efficacy using a set of orographic drag parameters. Before optimisation, the large-scale error metrics were significantly degraded when the orographic fields and code were updated. After optimisation, neutral scores were achieved in most upper-air variables and improvements were possible in near-surface variables. The hope is that others will adopt this approach to calibrate parameters within the ECMWF IFS, particularly when scientific changes have been made that lead to initial degradation of large-scale scores. The optimisation tool is relatively easy to adapt for other sets of parameters and runs efficiently, but some shortcomings and areas of future improvement to the optimisation tool are discussed here.

While this tool provides a more robust and streamlined approach compared to manual calibration, a level of subjectiveness remains. Choices such as the most appropriate metrics, forecast lead times, or geographical domains to consider are not always clear at the outset of an experiment. In practice, substantial hands-on involvement is still required to design experiments, interpret results, and decide whether proposed parameter updates are physically meaningful. This restriction means that the optimisation tool cannot be used as a ‘black-box’; it still requires a level of physical understanding of the processes.

Similarly, the choice of parameters strongly shapes the outcome. As with all types of calibration that rely on Bayesian principles, the solutions found only exist in the parameter space defined by the chosen parameters and their prior ranges (Williamson *et al.*, 2013). Changing these can lead to different outcomes, driven by new parameter interactions and the inherent non-linearity of the atmospheric system. In the example used for this study, the choice of five parameters was relatively straightforward since the parameters are linked to self-contained equations and their relationship is relatively well understood, albeit not trivial. The parameters can therefore be constrained by expert knowledge. However, when calibrating complex parts of the system, for example those parameters that have very non-linear relationships, the interactions across the system become less intuitive. This makes it essential to carry out comprehensive

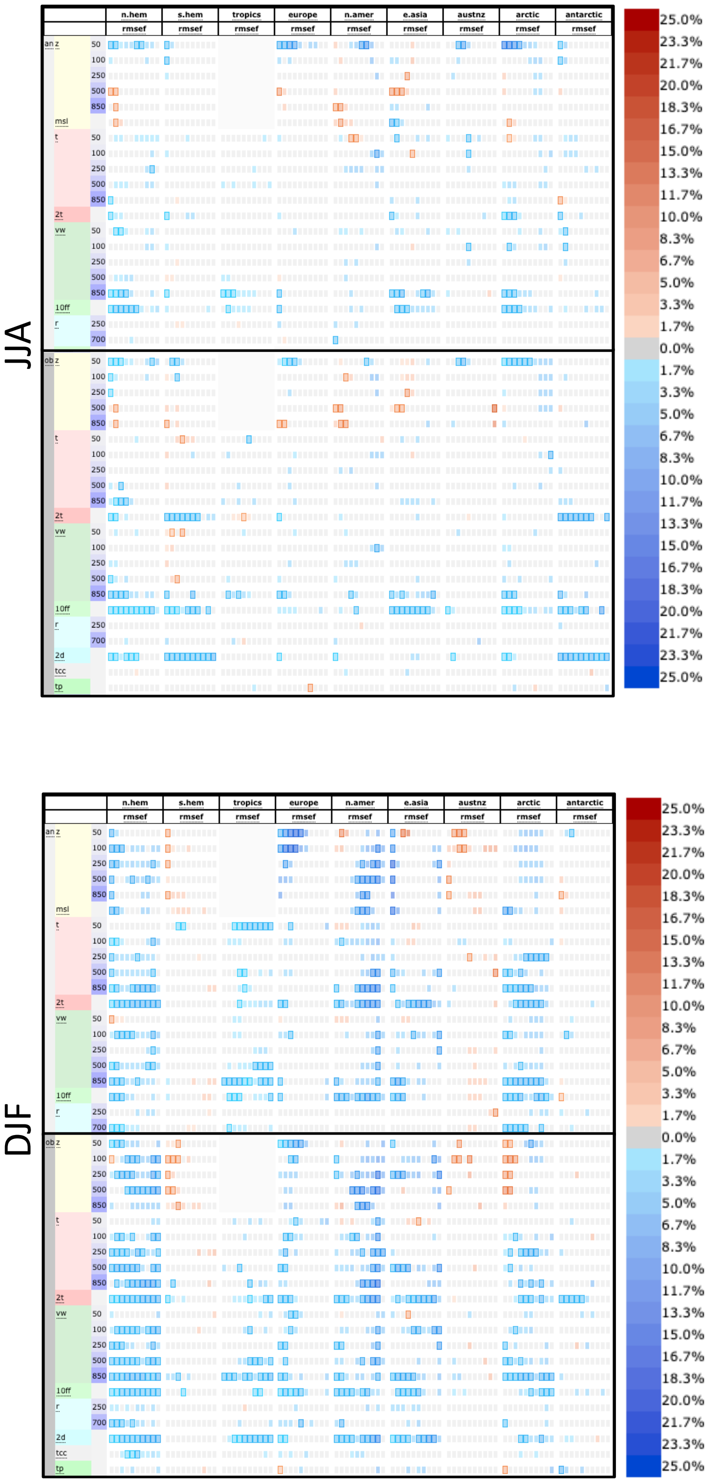


Figure 9: Scorecards showing the % changed of RMSE in the optimised analysis experiment compared with the unoptimised experiment for (top) June, July and August 2022 and (bottom) December, January and February 2022/23. Verification is shown against own analysis (an) and observations (ob). Each column is a different domain, each row is a different variable and each rectangle, from left to right, is a 24 hour forecast step up to 240 hours. Only values significant at the 95% level are shown and the colorbar indicates the % changed in RMSE.

sensitivity analyses to understand which parameters exert the strongest influence on forecasts and how uncertainties propagate through the model. An additional advantage of the tool developed at ECMWF is that the perturbed parameter ensemble it generates can be used in combination with sensitivity methods, such as the Morris method (Morris, 1991), to guide parameter selection in a physically consistent manner. As an example, a Morris sensitivity analysis of orographic parameters is shown in Figure B1 of the Appendix.

One of the key drawbacks to the Bayesian optimisation approach described in this paper is that the use of the combined error metric can lead to cancellation of large degradations, positive changes in RMSE, with large improvements, large negative changes in RMSE. For example, large improvements over the NH may be compensated by degradations in the SH. This compensation effect means it can be difficult to find parameters that are optimal for all regions, which makes the Bayesian optimisation not necessarily the best approach for global modelling and indicates further structural errors in the ECMWF IFS. An alternative approach that may help to mitigate these issues is the use of a history matching approach (Williamson *et al.*, 2013; Hourdin *et al.*, 2023), which adopts a more conservative strategy, constructing one Gaussian process emulator per error metric instead of one for them combined. Rather than seeking a single "optimal" parameter set, history matching identifies and rules out regions of parameter space that are implausible given observations. This strategy allows for a broader characterisation of parameter uncertainty, making it less sensitive to compensating errors across regions. However, history matching does not provide a single set of operational parameters; instead, it defines a space of plausible parameters, requiring further analysis or decision-making to select values for practical implementation.

Another limitation may arise from the use of Gaussian process emulators. GP emulators are most effective when the parameter space is low-dimensional and the relationships are reasonably smooth. For very high-dimensional problems, alternatives like sparse GPs, dimensionality reduction, or other surrogate models (e.g., neural networks, random forests) are often needed.

Despite these limitations, the approach offers several clear advantages. Beyond identifying parameter settings that improve forecast skill, it provides a deeper understanding of the model itself. The systematic exploration of parameter space can uncover new relationships between processes and highlight cases where the model may not be behaving as expected, thereby helping to identify structural errors and uncertainties. While the most costly aspect of this work is generating perturbed parameter ensembles through model runs, once these are available, it becomes straightforward to test different metrics, assess sensitivities, and evaluate trade-offs in calibration strategies. Currently, the Bayesian optimisation tool can only be applied to global parameters, but next steps include extending it to two-dimensional fields, which would greatly expand its potential for systematic model improvement.

A Appendix

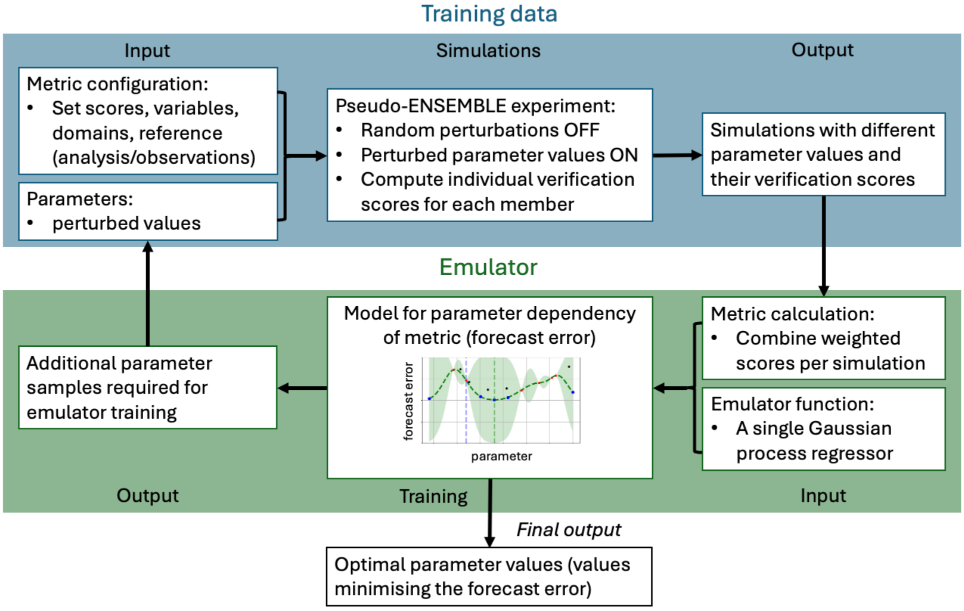

Training data
Input
Simulations
Output
Metric configuration:
• Set scores, variables, domains, reference (analysis/observations)
Parameters:
• perturbed values
Pseudo-ENSEMBLE experiment:
• Random perturbations OFF
• Perturbed parameter values ON
• Compute individual verification scores for each member
Simulations with different parameter values and their verification scores
Emulator
Model for parameter dependency of metric (forecast error)
forecast error
parameter
Additional parameter samples required for emulator training
Metric calculation:
• Combine weighted scores per simulation
Emulator function:
• A single Gaussian process regressor
Output
Training
Input
Final output
Optimal parameter values (values minimising the forecast error)

Figure A1: Iterative workflow of the optimisation in generating training data with the purpose-built forecasting suite (blue) and training of the Gaussian process emulator with the in-house python package (green).

B Appendix

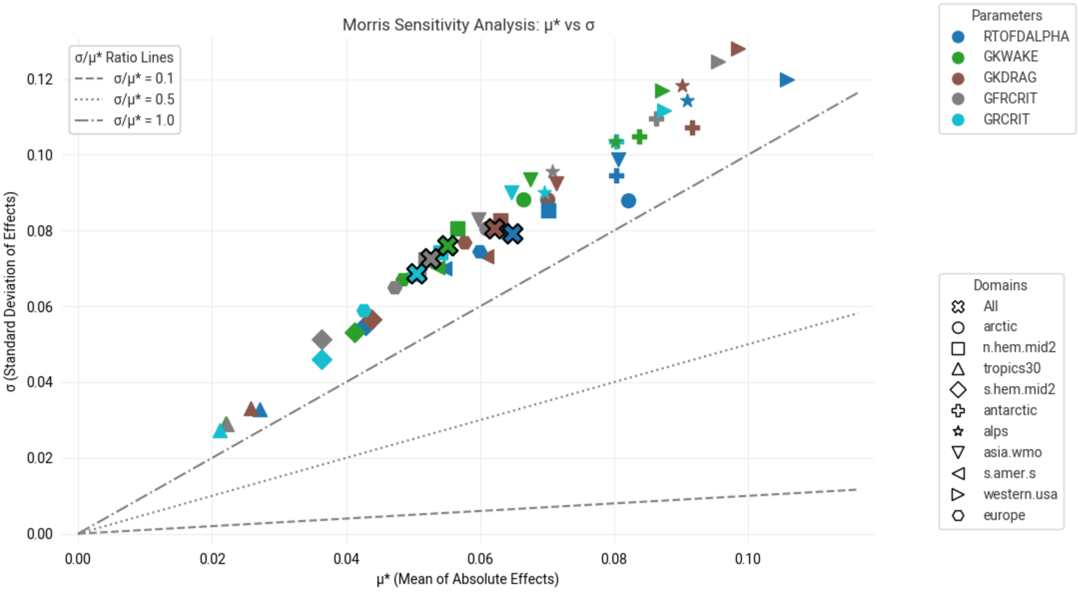


Figure B1: Values of the mean of the absolute effect (μ) and the standard deviation of the effects (σ) from Morris sensitivity analysis. The sensitivity of each parameter over different domains, given by the legend, are shown. Lines of constant $\sigma/\mu^* = 0.1, 0.5, 1.0$ are plotted for reference.

Morris sensitivity analysis (see Morris (1991)) was carried out on our 300 member perturbed parameter ensemble used in our multi-parameter optimisation, as a means of illustrating the concept. The analysis uses the same combined error metric described in Section 2.3 using the variables in Table 3 to determine the sensitivity of the parameters. The sensitivity analysis in our optimization tool computed the following scores, which as summarized in Figure B1:

- μ: the mean elementary effect: the average change in the output due to changes in a parameter.
- μ^*: the mean of the absolute values of the elementary effects. This captures the overall importance of a parameter, regardless of the direction of the effect. A higher μ^* value indicates that the model is more sensitive to changes in that parameter.
- σ: the standard deviation of the elementary effects: reflects the variability in a parameter's effect due to non-linearities or interactions with other parameters. A higher σ value suggests either non-linear effects (i.e., the parameter has a non-linear relationship with the output), or interactions with other parameters. These two effects cannot be disentangled from σ alone.
- μ^*_{conf}: the bootstrapped confidence interval for μ^*, indicates the statistical reliability of the μ^* estimate.
- σ/μ^*: An important diagnostic metric, which helps characterise the nature of a parameter's influence. The interpretation of this metric is given in Table B1.

σ/μ^* **value**	**Interpretation**
≈ 0	The effect of the input is **almost purely linear and additive**. Little to no interaction.
< 0.5	Mostly **linear or additive** behaviour. Low interaction.
$0.5 - 1$	**Moderate non-linearity or interaction.**
> 1	**Strong non-linearity and/or interaction** with other inputs. Effect is highly variable depending on input configuration.

Table B1: Interpretation of σ/μ^* Morris sensitivity values shown in Figure B1.

Figure B1 shows that, for all domains and all parameters, the values of σ/μ^* are larger than 1.0, indicating that they have is a strong non-linearity and/or interaction with other parameters. This confirms that the parameters should be optimised together, as has been done here, to avoid missing interactions. Looking at the different parameters clustered by the domains over which the combined error metric is computed, it is evident that the parameter RTOFDALPHA generally has the largest importance in determining the error (largest values of μ^*). The other parameters tend to have slightly larger values of σ/μ^*, indicating a slightly stronger non-linearity, although most points lie on roughly the same slope.

C Appendix

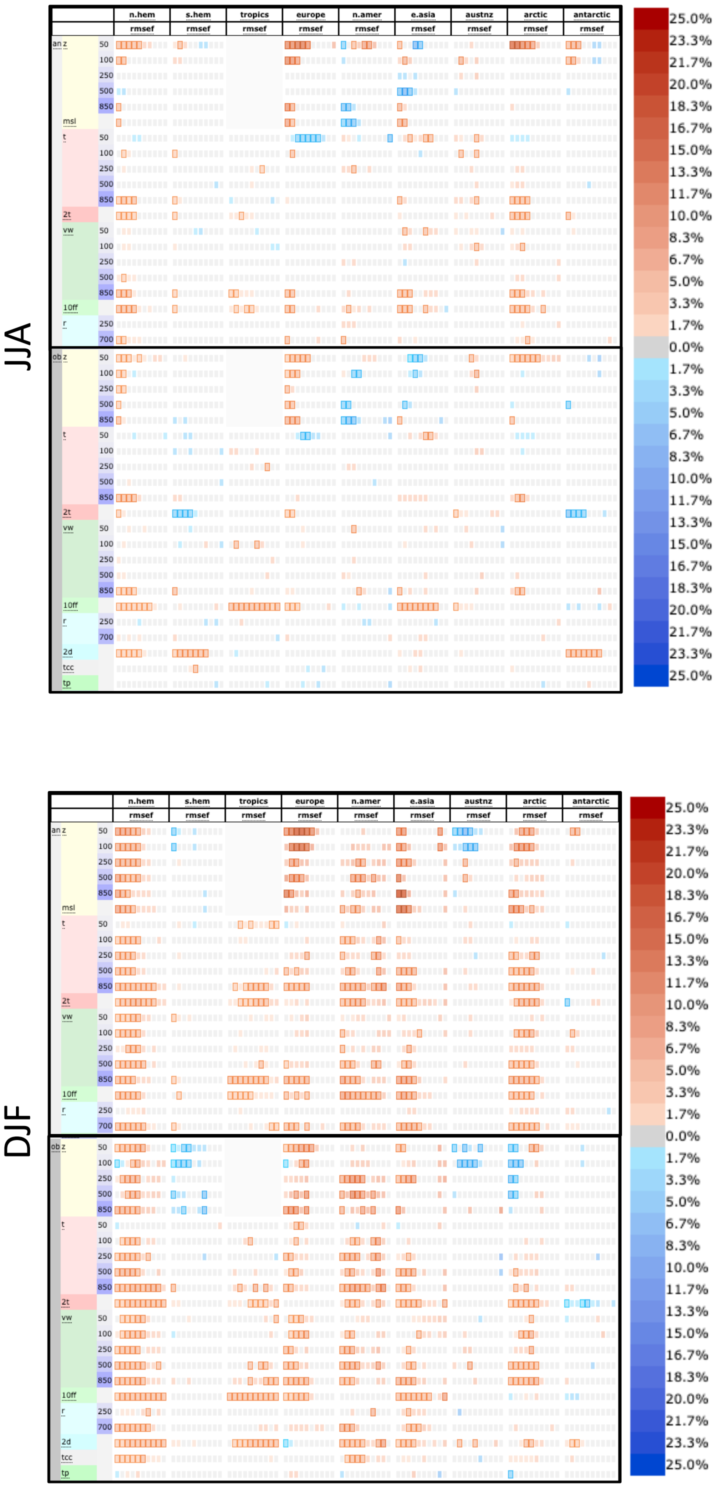


Figure C1: Scorecards showing the % changed of RMSE in the unoptimised analysis experiment compared with to the benchmark experiment for (top) June, July and August 2022 and (bottom) December, January and February 2022/23. Verification is shown against own analysis (an) and observations (ob). Each column is a different domain, each row is a different variable and each rectangle, from left to right, is a 24 hour forecast step up to 240 hours. Only values significant at the 95% level are shown and the colorbar indicates the % changed in RMSE.

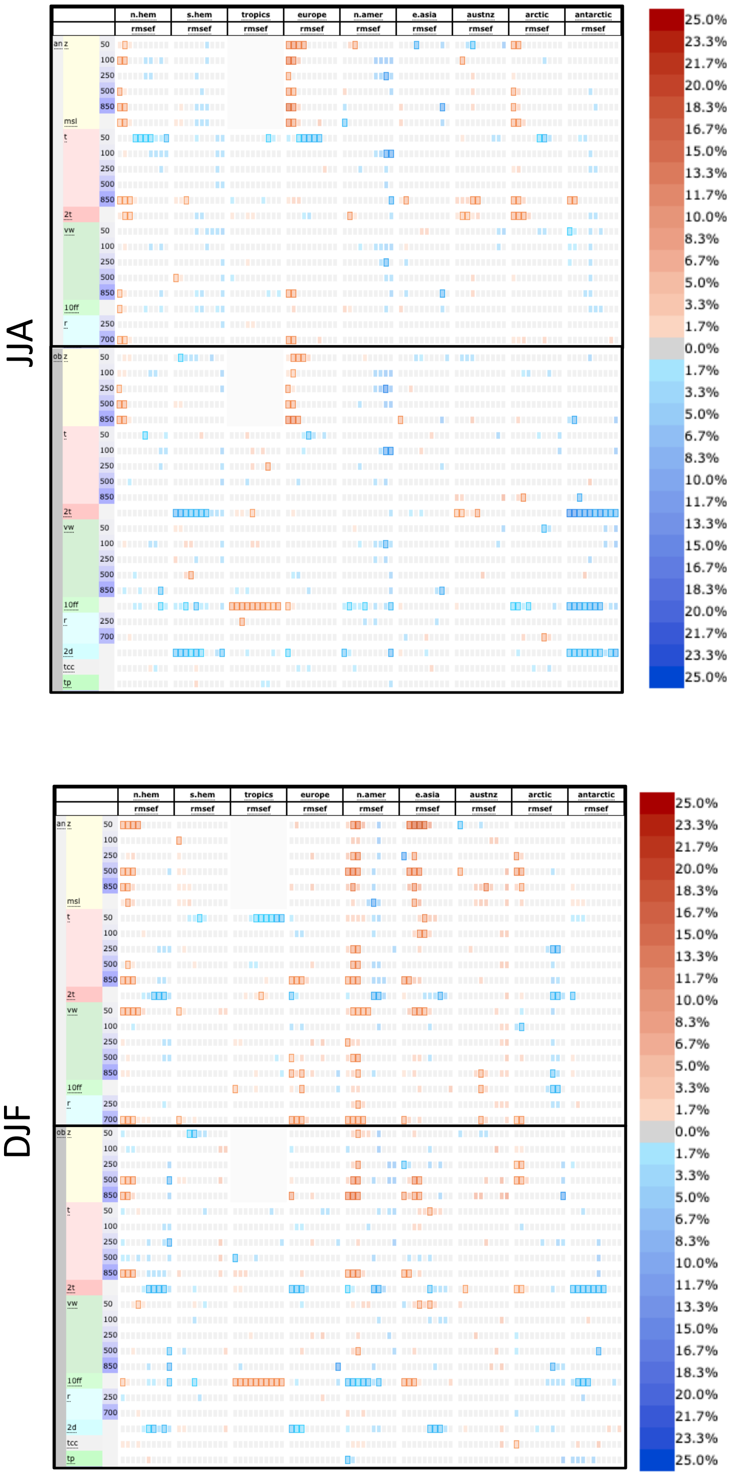


Figure C2: As in Figure C1 but for the analysis experiment with optimised orography parameters.

Variable	Definition	Units
z	Geopotential	m
msl	Mean sea level pressure	hPa
t	Temperature	K
2t	Temperature at 2m	K
vw	Vector wind	ms^{-1}
10ff	10m wind speed	ms^{-1}
r	Relative humidity	%
2d	2m dew point temperature	K
tcc	Total cloud cover	%
tp	Total precipitation	mm

Table C1: Description and units of variables that are shown in scorecards of Figures C1 and C2.

Acknowledgements

Thank you to Matthew Chantry for guidance on the use of Bayesian optimisation and to Aristofanis Tsiringakis for help with the development of the purpose-built PPE suite.

References

Annan, J. D., Lunt, D. J., Hargreaves, J. C. and Valdes, P. J. (2005). Parameter estimation in an atmospheric gcm using the ensemble kalman filter. *Nonlinear Processes in Geophysics*, **12**(3), 363–371, doi:10.5194/npg-12-363-2005, URL https://npg.copernicus.org/articles/12/363/2005/.

Beljaars, A., Boussetta, S., Kanehama, T., Deconinck, W. and Maciel, P. (2022). Future processing of subgrid orography fields. *ECMWF Research Department Memorandum*.

Beljaars, A. C. M., Brown, A. R. and Wood, N. (2004). A new parametrization of turbulent orographic form drag. *Quarterly Journal of the Royal Meteorological Society*, **130**(599), 1327–1347, doi:https://doi.org/10.1256/qj.03.73, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1256/qj.03.73.

Byrd, R. H., Lu, P., Nocedal, J. and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing*, **16**(5), 1190–1208.

Couvreux, F., Hourdin, F., Williamson, D., Roehrig, R., Volodina, V., Villefranque, N., Rio, C., Audouin, O., Salter, J., Bazile, E. *et al.* (2021). Process-based climate model development harnessing machine learning: I. a calibration tool for parameterization improvement. *Journal of Advances in Modeling Earth Systems*, **13**(3), e2020MS002217.

Dagon, K., Sanderson, B. M., Fisher, R. A. and Lawrence, D. M. (2020). A machine learning approach to emulation and biophysical parameter estimation with the community land model, version 5. *Advances in Statistical Climatology, Meteorology and Oceanography*, **6**(2), 223–244.

Edwards, N. R., Cameron, D. and Rougier, J. (2011). Precalibrating an intermediate complexity climate model. *Climate dynamics*, **37**(7), 1469–1482.

Fer, I., Kelly, R., Moorcroft, P. R., Richardson, A. D., Cowdery, E. M. and Dietze, M. C. (2018). Linking big models to big data: efficient ecosystem model calibration through bayesian model emulation. *Biogeosciences*, **15**(19), 5801–5830.

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A. and Rubin, D. (2013). *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science, Taylor & Francis, ISBN 9781439840955, URL https://books.google.co.uk/books?id=ZXL6AQAAQBAJ.

Ginsbourger, D., Le Riche, R. and Carraro, L. (2010). Kriging is well-suited to parallelize optimization. In *Computational Intelligence in Expensive Optimization Problems*, pp. 131–162, Springer, doi:10.1007/978-3-642-10701-6_6.

Head, K. M. N. H.-L. G. . S. I., T. (2018). scikit-optimize/scikit-optimize: v0.5.2. *Zenodo*, doi:https://doi.org/10.5281/zenodo.1207017.

Hourdin, F., Ferster, B., Deshayes, J., Mignot, J., Musat, I. and Williamson, D. (2023). Toward machine-assisted tuning avoiding the underestimation of uncertainty in climate change projections. *Science Advances*, **9**(29), eadf2758, doi:10.1126/sciadv.adf2758, URL https://www.science.org/doi/abs/10.1126/sciadv.adf2758.

Hourdin, F., Williamson, D., Rio, C., Couvreux, F., Roehrig, R., Villefranque, N., Musat, I., Fairhead, L., Diallo, F. B. and Volodina, V. (2021). Process-based climate model development harnessing machine learning: Ii. model calibration from single column to global. *Journal of Advances in Modeling Earth Systems*, **13**(6), e2020MS002225.

Jones, D. R., Schonlau, M. and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, **13**(4), 455–492, doi:10.1023/A:1008306431147.

Lang, S. T. K., Lock, S.-J., Leutbecher, M., Bechtold, P. and Forbes, R. M. (2021). Revision of the stochastically perturbed parametrisations model uncertainty scheme in the integrated forecasting system. *Quarterly Journal of the Royal Meteorological Society*, **147**(735), 1364–1381, doi:https://doi.org/10.1002/qj.3978, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3978.

Leutbecher, L. S. L. S.-J. R. C. D. T. A., M. (2024). Improving the physical consistency of ensemble forecasts by using spp in the ifs. *ECMWF Newsletter, Autumn 2024*, **181**.

Lguensat, R., Deshayes, J., Durand, H. and Balaji, V. (2023). Semi-automatic tuning of coupled climate models with multiple intrinsic timescales: Lessons learned from the lorenz96 model. *Journal of Advances in Modeling Earth Systems*, **15**(5), e2022MS003367.

Loeppky, J. L., Sacks, J. and Welch, W. J. (2009). Choosing the sample size of a computer experiment: A practical guide. *Technometrics*, **51**(4), 366–376, doi:10.1198/TECH.2009.08040.

Lott, F. and Miller, M. J. (1997). A new subgrid-scale orographic drag parametrization: Its formulation and testing. *Quarterly Journal of the Royal Meteorological Society*, **123**(537), 101–127, doi:https://doi.org/10.1002/qj.49712353704, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.49712353704.

Marchant, R., Ramos, F. and Sanner, S. (2014). Sequential bayesian optimisation for spatial-temporal monitoring. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, p. 553–562, UAI'14, AUAI Press, Arlington, Virginia, USA, ISBN 9780974903910.

Mauritsen, T., Stevens, B., Roeckner, E., Crueger, T., Esch, M., Giorgetta, M., Haak, H., Jungclaus, J., Klocke, D., Matei, D., Mikolajewicz, U., Notz, D., Pincus, R., Schmidt, H. and Tomassini, L. (2012). Tuning the climate of a global model. *Journal of Advances in Modeling Earth Systems*, **4**(3), doi:https://doi.org/10.1029/2012MS000154, URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2012MS000154.

McNeall, D., Williams, J., Booth, B., Betts, R., Challenor, P., Wiltshire, A. and Sexton, D. (2016). The impact of structural error on parameter constraint in a climate model. *Earth System Dynamics*, **7**(4), 917–935, doi:10.5194/esd-7-917-2016, URL https://esd.copernicus.org/articles/7/917/2016/.

McNeall, D. J., Challenor, P. G., Gattiker, J. and Stone, E. J. (2013). The potential of an observational data set for calibration of a computationally expensive computer model. *Geoscientific Model Development Discussions*, **6**(2), 2369–2401.

Morris, M. D. (1991). Factorial sampling plans for preliminary computational experiments. *Technometrics*, **33**(2), 161–174.

Ollinaho, P., Bechtold, P., Leutbecher, M., Laine, M., Solonen, A., Haario, H. and Järvinen, H. (2013a). Parameter variations in prediction skill optimization at ecmwf. *Nonlinear Processes in Geophysics*, **20**(6), 1001–1010, doi:10.5194/npg-20-1001-2013, URL https://npg.copernicus.org/articles/20/1001/2013/.

Ollinaho, P., Laine, M., Solonen, A., Haario, H. and Järvinen, H. (2013b). Nwp model forecast skill optimization via closure parameter variations. *Quarterly Journal of the Royal Meteorological Society*, **139**(675), 1520–1532, doi:https://doi.org/10.1002/qj.2044, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.2044.

Olson, B., Hashmi, I., Molloy, K. and Shehu, A. (2012). Basin hopping as a general and versatile optimization framework for the characterization of biological macromolecules. *Advances in Artificial Intelligence*, **2012**(1), 674832, doi:https://doi.org/10.1155/2012/674832, URL https://onlinelibrary.wiley.com/doi/abs/10.1155/2012/674832.

Pedregosa, V. G. G. A.-M. V. T. B. G. O. . . D. , F. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, **12**, 2825–2830.

Raoult, N., Beylat, S., Salter, J. M., Hourdin, F., Bastrikov, V., Ottlé, C. and Peylin, P. (2024). Exploring the potential of history matching for land surface model calibration. *Geoscientific Model Development*, **17**(15), 5779–5801.

Raoult, N., Douglas, N., MacBean, N., Kolassa, J., Quaife, T., Roberts, A. G., Fisher, R., Fer, I., Bacour, C., Dagon, K., Hawkins, L., Carvalhais, N., Cooper, E., Dietze, M. C., Gentine, P., Kaminski, T., Kennedy, D., Liddy, H. M., Moore, D. J. P., Peylin, P., Pinnington, E., Sanderson, B., Scholze, M., Seiler, C., Smallman, T. L., Vergopolan, N., Viskari, T., Williams, M. and Zobitz, J. (2025). Parameter estimation in land surface models: Challenges and opportunities with data assimilation and machine learning. *Journal of Advances in Modeling Earth Systems*, **17**(11), e2024MS004733, doi: https://doi.org/10.1029/2024MS004733, URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2024MS004733, e2024MS004733 2024MS004733.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

van Niekerk, A. and Vosper, S. (2021). Towards a more “scale-aware” orographic gravity wave drag parametrization: Description and initial testing. *Quarterly Journal of the Royal Meteorological Society*, **147**(739), 3243–3262, doi:https://doi.org/10.1002/qj.4126, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.4126.

Villefranque, N., Blanco, S., Couvreux, F., Fournier, R., Gautrais, J., Hogan, R. J., Hourdin, F., Volodina, V. and Williamson, D. (2021). Process-based climate model development harnessing machine learning: Iii. the representation of cumulus geometry and their 3d radiative effects. *Journal of Advances in Modeling Earth Systems*, **13**(4), e2020MS002423.

Williamson, D., Blaker, A. T., Hampton, C. and Salter, J. (2015). Identifying and removing structural biases in climate models with history matching. *Climate dynamics*, **45**(5), 1299–1324.

Williamson, D., Goldstein, M., Allison, L., Blaker, A., Challenor, P., Jackson, L. and Yamazaki, K. (2013). History matching for exploring and reducing climate model parameter space using observations and a large perturbed physics ensemble. *Climate dynamics*, **41**(7), 1703–1729.

Williamson, D. B., Blaker, A. T. and Sinha, B. (2017). Tuning without over-tuning: parametric uncertainty quantification for the nemo ocean model. *Geoscientific Model Development*, **10**(4), 1789–1816.