

A Gaussian Process Enhancement to Linear Parameter Varying Models

Stefan Schuet^{*} Carlos Malpica[†] Jeremy Aires[‡]

NASA Ames Research Center, Moffett Field, CA 94035

Simulation and analysis for modern engineering systems now routinely requires the merging of multiple disciplines, physical-domains, time-scales, and data sets — all at ever increasing levels. These capabilities are especially needed in the domain of Advanced Air Mobility, where rapidly emerging vehicle designs are significantly more complex, while having to be both cost-effective and safe. To meet these engineering challenges, machine learning methods are an attractive option for merging models and data across multiple areas while providing uncertainty quantification and maintaining computational efficiency. This paper examines the use of Gaussian process machine learning to generalize and enhance the commonly used class of quasi-Linear Parameter Varying models for fast full-envelope simulation while also supporting control system design and analysis with model uncertainty. Gaussian process machine learning is selected because it: can fuse multiple data sets, enables an easy trade-off between data fitting and smoothing, provides model uncertainty quantification, scales well with increasing complexity, and does not generally require starting from a large training data set. To demonstrate the benefits of the approach, a robust stability analysis with Gaussian process uncertainty is shown for a NASA reference design of an electric quad-rotor air-taxi concept vehicle with motor parameter uncertainty.

Nomenclature

GP	Gaussian Process
LPV	Linear Parameter Varying
qLPV	quasi-Linear Parameter Varying
RPM	Rotations Per Minute
$E[\cdot], \text{Var}[\cdot]$	Expected value, variance
x, u	Dynamical system state and input vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively
$A(\nu), B(\nu)$	qLPV model state and control matrices evaluated at ν
$x_t(\nu), u_t(\nu)$	qLPV model trim state and input vectors evaluated at ν
M	Number of observed anchor points in the training data set
ν	qLPV model look-up parameter, same as x in this paper
$\mathbf{x}$	Gaussian process input vector in $\mathbb{R}^d$
$\mathbf{z}$	Vector of observed noisy data in $\mathbb{R}^M$
ϵ	Observation noise random variable
θ_{hp}	Vector of GP hyper-parameters
ℓ, σ_f, σ_m	Kernel length-scale parameter, prior variance, observation noise variance
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$	Multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ
$m(\mathbf{x})$	Mean of GP output at $\mathbf{x}$
$k(\mathbf{x}, \mathbf{x}')$	Kernel of GP, sets covariance between outputs at $\mathbf{x}$ and $\mathbf{x}'$
$\sim$	Denotes “is distributed according to”
$\mathbf{u} \mathbf{v}$	Random vector $\mathbf{u}$ given $\mathbf{v}$

^{*}Computer Engineer, Intelligent Systems Division, Mail Stop 269-3, AIAA Member, stefan.r.schuet@nasa.gov.

[†]Aerospace Engineer, Aeromechanics Office, Mail Stop 243-11, AIAA Member.

[‡]Computer Engineer, Intelligent Systems Division, Mail Stop 269-1.

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

I. Introduction

Emerging air vehicle designs, especially for Advanced Air Mobility (AAM) applications, are becoming increasingly complex, typically requiring high fidelity simulation and analysis tools to process complex interactions between aerodynamic and propulsive forces, flexible structures, and vehicle configuration parameters. Since these higher-fidelity simulation tools do not run in real-time, the current prevailing methodology for developing fast nonlinear full-flight envelope models is to stitch together linearized dynamics models at predefined anchor points.^{1–5} Such models are referred to as Linear Parameter Varying (LPV) when the parameter vector is independent of the state dynamics, and quasi-LPV (qLPV) otherwise. In the basic approach, these models require an exponentially increasing number of anchor points over a mesh-grid of the parameters, which may include, airspeed, altitude, temperature, flap settings, nacelle angle, and motor efficiency with multiplicity for each rotor in the design. This could make the standard qLPV approach cumbersome to extend to the new, significantly more complex AAM vehicle designs that industry is proposing. More importantly, stitched conceptual design models typically do not include an accuracy or uncertainty assessment of the approximation relative to an underlying true vehicle model (that may not be knowable), or even to what the higher-fidelity tools would predict. This makes it more difficult to ensure the control system design robustly meets the design requirements. This paper explores the use of Gaussian process machine learning to manage these emerging issues.

The use of Gaussian processes for machine learning is described by Williams and Rasmussen,⁶ and a more extensive literature review of this area is included in their 2006 book.⁷ There they describe use of Gaussian processes in: Astronomy as far back as 1880, time series analysis by renowned researchers Wiener and Kolmogorov in the 1940's, meteorology starting in the 1950's, Geostatistics in the 1970's (where the same method is also known as *kriging*), and more general regression type problems in the 1970's and 1980's [7, pg. 29]. More recently, in the aerospace related fields, Gaussian processes are used for the discovery of Partial Differential Equations (PDE),⁸ surrogate modeling from experimental and computational aerodynamic data sets,⁹ multi-fidelity data fusion with application to the study of blended-wing body aircraft,¹⁰ nonlinear time-series prediction of complex rotorcraft dynamics,¹¹ and disturbance model learning for use in reachability based safety frameworks.^{12,13} A common theme, especially in the more recent work, is the use of Gaussian process models for fusing data from multiple information sources including experiment, flight-test, and physics-based models.

The focus in this paper is on developing a Gaussian process model that generalizes the qLPV approach to provide smooth interpolation between a finite collection of linearized perturbation models in parameter space. Overall the Gaussian process model is nonlinear, like its qLPV counterpart. Unlike a typical qLPV model, however, the Gaussian process model is able to manage known errors in the observed input data, and it outputs the model prediction uncertainty as a smooth function of the look-up parameters. This typically includes uncertainty on the model evaluation at points that are “far” from the observed data — where far is quantified relative to a length-scale parameter of the Gaussian process. Furthermore, the approach is expected to scale well as the dimension of the look-up parameter vector increases to moderate sizes. While it is well recognized that Gaussian processes become cumbersome with larger observation data sets, they can be augmented with an auto-relevance detection capability for removing irrelevant data that does not improve model prediction.¹⁴ The uncertainty quantification capability of Gaussian processes is also useful for managing both epistemic and aleatory model uncertainty for robust control analysis and design. The Gaussian process models proposed here are also readily converted back into qLPV form for use with existing software for qLPV models.

We proceed as follows. In Section II, a practical introduction to data fitting with Gaussian processes is provided. We hope this will serve as a useful tutorial for engineers looking to get started quickly from a prerequisite understanding of multi-variate Gaussian random vectors and linear algebra. Section II then discusses the use of Gaussian processes for enhancing the quasi-Linear Parameter Varying model, in the context of developing full-envelope simulation and control models for air-vehicle conceptual designs. The usefulness of the approach for control system analysis is then demonstrated in Section IV, before concluding in V.

II. Gaussian Process Modeling

At the end-of-the-day, computation with Gaussian processes is performed using little more than random sampling from multivariate normal distributions and linear algebra. This section provides a quick (non-rigorous) review of Gaussian processes, sufficient for a more detailed understanding of how they are used in this paper.

A. Multivariate Gaussian Distribution

A Gaussian random vector is a finite collection of jointly correlated random variables

$$\mathbf{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_M \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma) \quad (1)$$

where “ $\sim$ ” denotes “is distributed according to”, $\mathcal{N}$ denotes the multi-variate Normal distribution (a.k.a, Gaussian distribution), where $\boldsymbol{\mu} \in \mathbb{R}^M$ is the mean with elements $\mu_k = \mathbb{E}[v_k]$, and $\Sigma \in \mathbb{R}^{M \times M}$ is the covariance matrix with elements $\Sigma_{jk} = \mathbb{E}[(v_j - \mu_j)(v_k - \mu_k)]$. Here $\mathbb{E}[\cdot]$ denotes the expected value.

Now suppose $\mathbf{v}$ is separated into two sub-component vectors, i.e.,

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{21}^T \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right), \quad (2)$$

and that $\mathbf{v}_2$ is known. Since $\mathbf{v}_1$ and $\mathbf{v}_2$ are correlated, one can infer some information about $\mathbf{v}_1$ given $\mathbf{v}_2$. This fact is expressed mathematically by stating that

$$\mathbf{v}_1 | \mathbf{v}_2 \sim \mathcal{N}(\boldsymbol{\mu}_1 + \Sigma_{21}^T \Sigma_{22}^{-1}(\mathbf{v}_2 - \boldsymbol{\mu}_2), \Sigma_{11} - \Sigma_{21}^T \Sigma_{22}^{-1} \Sigma_{21}), \quad (3)$$

where the random variable $\mathbf{v}_1 | \mathbf{v}_2$ is read as $\mathbf{v}_1$ “given” $\mathbf{v}_2$. Another frequently used property of Gaussian random vectors is that any linear transformation of a Gaussian random vector produces another Gaussian random vector, in particular,

$$\mathbf{u} = A\mathbf{v} \sim \mathcal{N}(A\boldsymbol{\mu}, A\Sigma A^T). \quad (4)$$

Finally, the marginal distribution specifies the probability of one random variable (or vector) without reference to any other. By definition, it is computed by integrating the probability distribution over the un-referenced variables. For a Gaussian random vector, this is accomplished simply by taking the corresponding terms from the mean and covariance matrix.^a For example, the marginal distribution for $\mathbf{v}_1$ in Equation (2) is $\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_{11})$.

B. Gaussian Process

A Gaussian Process (GP) is a (usually infinite) collection of random variables, where any finite sub-collection is governed by a joint Gaussian distribution.⁷ It is a stochastic process because it is an indexed collection of random variables. A useful way to think about GPs is as a model that generates random functions $f : \mathbb{R}^d \rightarrow \mathbb{R}$ in much the same way that one imagines sampling a random vector from a probability distribution. A GP is typically specified as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')),$$

where $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ is the mean function, and $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\{[f(\mathbf{x}) - m(\mathbf{x})][f(\mathbf{x}') - m(\mathbf{x}')]\}$ is the covariance function.⁷ The function $k(\mathbf{x}, \mathbf{x}')$ is also known as the *kernel* function of the Gaussian process.

With this definition it follows that sampling from $\mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, on a set of discrete inputs $\mathbf{x}_1, \dots, \mathbf{x}_M$ amounts to the ordinary sampling of a random vector from a multi-variate Gaussian distribution, i.e., sampling

$$\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_M) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \quad (5)$$

^aThis is easily proven with property (4).

where $\boldsymbol{\mu} = [m(\mathbf{x}_1), \dots, m(\mathbf{x}_M)]^T$ and Σ is a positive semidefinite matrix in $\mathbb{R}^{M \times M}$ with elements $\Sigma_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, for $i, j = 1, \dots, M$. This operation is readily accomplished by sampling a vector $\mathbf{v}$ from $\mathcal{N}(0, I)$, where I is the identity matrix in $\mathbb{R}^{M \times M}$, and setting $\mathbf{u} = \Sigma^{1/2}\mathbf{v} + \boldsymbol{\mu}$.^b With property (4) one can easily verify that $\mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, as desired.

In a supervised machine learning problem, one seeks a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ such that $f(\mathbf{x}_k) \approx y_k$ for all k in a set of training data, expressed as $\{(y_k, \mathbf{x}_k) \text{ for } k = 1, \dots, M\}$. The function $f(\mathbf{x})$ must predict an output at any input $\mathbf{x}$, and a quantification of the uncertainty of any such prediction should be provided.

In the simple case, one seeks an interpolation function $f(\mathbf{x})$ that exactly matches the training data, i.e.,

$$f(\mathbf{x}_k) = y_k \text{ for } k = 1, \dots, M. \quad (6)$$

This corresponds to the case where the training data is provided without error. The Gaussian process $\mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ characterizes an entire solution space to this problem. To see this, use the relation (5) to build the random vector

$$\begin{bmatrix} f(\mathbf{x}) \\ f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_M) \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m(\mathbf{x}) \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \sigma_f^2 & k_{\mathbf{x}}(\mathbf{x})^T \\ k_{\mathbf{x}}(\mathbf{x}) & \Sigma \end{bmatrix} \right), \quad (7)$$

where,

$$k_{\mathbf{x}}^T(\mathbf{x}) = \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) & \dots & k(\mathbf{x}, \mathbf{x}_M) \end{bmatrix}, \quad (8)$$

$$\boldsymbol{\mu} = \begin{bmatrix} m(\mathbf{x}_1) & \dots & m(\mathbf{x}_M) \end{bmatrix}, \quad (9)$$

and $\sigma_f^2 = k(\mathbf{x}, \mathbf{x})$. The vector $\mathbf{x}$ in this context is an arbitrary test point used to evaluate a prediction. From (3) it follows that

$$f(\mathbf{x})|\mathbf{y} \sim \mathcal{N}(m(\mathbf{x}) + k_{\mathbf{x}}^T \Sigma^{-1}(\mathbf{y} - \boldsymbol{\mu}), \sigma_f^2 - k_{\mathbf{x}}^T \Sigma^{-1} k_{\mathbf{x}}), \quad (10)$$

where $\mathbf{y} = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_M)]^T$. With a weights vector defined as

$$\mathbf{w} = \Sigma^{-1}(\mathbf{y} - \boldsymbol{\mu}), \quad (11)$$

the mean of $f(\mathbf{x})|\mathbf{y}$ is expressed as

$$\mathbb{E}[f(\mathbf{x})|\mathbf{y}] = m(\mathbf{x}) + \mathbf{w}^T k_{\mathbf{x}} = m(\mathbf{x}) + \sum_{i=1}^N w_i k(\mathbf{x}_i, \mathbf{x}). \quad (12)$$

Clearly, this function is differentiable with respect to $\mathbf{x}$ when the mean and kernel functions are differentiable with respect to $\mathbf{x}$. In this event, the mean of the GP given the observed data is a smooth function of $\mathbf{x}$. One can also show that $\mathbb{E}[f(\mathbf{x}_k)|\mathbf{y}] = y_k$, and that the marginal variance of $f(\mathbf{x}_k)|\mathbf{y}$ is

$$\sigma_f^2 - k_{\mathbf{x}}^T \Sigma^{-1} k_{\mathbf{x}} = 0$$

when $\mathbf{x} = \mathbf{x}_k$ for any $k = 1, \dots, M$. It follows in this case that $\mathbb{E}[f(\mathbf{x})|\mathbf{y}]$ smoothly interpolates the input data.

In typical applications, $f(\mathbf{x}_k)$ is not directly observed. Instead, an observation is better modeled as

$$z_k = f(\mathbf{x}_k) + \epsilon_k, \quad (13)$$

^bFor positive definite Σ , it is computationally more efficient to use the Cholesky decomposition $\Sigma = LL^T$ and $\mathbf{u} = L\mathbf{v} + \boldsymbol{\mu}$.

where ϵ_k is a normally distributed, zero mean, random variable that admits uncertainty in the observation z_k . With the assumption,

$$\begin{bmatrix} f(\mathbf{x}) \\ f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_M) \\ \epsilon_1 \\ \vdots \\ \epsilon_M \end{bmatrix} = \begin{bmatrix} f(\mathbf{x}) \\ \mathbf{y} \\ \boldsymbol{\epsilon} \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m(\mathbf{x}) \\ \boldsymbol{\mu} \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_f^2 & k_{\mathbf{x}}^T & 0 \\ k_{\mathbf{x}} & \Sigma & 0 \\ 0 & 0 & D \end{bmatrix} \right) \quad (14)$$

it follows with $\mathbf{z} = \mathbf{y} + \boldsymbol{\epsilon}$ that

$$\begin{bmatrix} f(\mathbf{x}) \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} f(\mathbf{x}) \\ \mathbf{y} \\ \boldsymbol{\epsilon} \end{bmatrix}. \quad (15)$$

From the property (4) it then follows that

$$\begin{bmatrix} f(\mathbf{x}) \\ \mathbf{z} \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m(\mathbf{x}) \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \sigma_f^2 & k_{\mathbf{x}}^T \\ k_{\mathbf{x}} & \Sigma + D \end{bmatrix} \right), \quad (16)$$

where D is the, typically diagonal, covariance matrix of the random vector $\boldsymbol{\epsilon}$. Then making use of (3) we obtain

$$\mathbb{E}[f(\mathbf{x})|\mathbf{z}] = m(\mathbf{x}) + k_{\mathbf{x}}(\mathbf{x})^T (\Sigma + D)^{-1} (\mathbf{z} - \boldsymbol{\mu}) \quad (17)$$

$$\text{Var}[f(\mathbf{x})|\mathbf{z}] = \sigma_f^2 - k_{\mathbf{x}}(\mathbf{x})^T (\Sigma + D)^{-1} k_{\mathbf{x}}(\mathbf{x}), \quad (18)$$

where $\mathbb{E}[f(\mathbf{x})|\mathbf{z}]$ is used to predict the output of the model $f(\mathbf{x})$ at any input $\mathbf{x}$, with marginal uncertainty characterized by the variance function $\text{Var}[f(\mathbf{x})|\mathbf{z}]$. Both of these are smooth functions of $\mathbf{x}$ whenever $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ are smooth functions of $\mathbf{x}$.

Critically, the variance in Equation (18) captures the ability to estimate a true underlying model $f(\mathbf{x})$, which tends to diminish with additional training data, and therefore represents a reducible (or epistemic) uncertainty. It is however, not the variance that captures how well the model will predict new data according to the observation process (13). This case is captured by the random variable

$$z(\mathbf{x})|\mathbf{z} \sim \mathcal{N}(\mathbb{E}[f(\mathbf{x})|\mathbf{z}], \text{Var}[f(\mathbf{x})|\mathbf{z}] + \sigma_{\epsilon}^2), \quad (19)$$

where as before, $\mathbf{z} \in \mathbb{R}^M$ is the vector of observed noisy training data, σ_{ϵ}^2 is the variance of ϵ_k in (13) for the new independent observation, and $z(\mathbf{x}) \in \mathbb{R}$ here models a noisy observation at a new input $\mathbf{x}$. Notice that the predictive variance $\text{Var}[f(\mathbf{x})|\mathbf{z}] + \sigma_{\epsilon}^2$ includes the irreducible variance σ_{ϵ}^2 representative of aleatory uncertainty in the model. Also, the use of D for the covariance of $\boldsymbol{\epsilon}$ in (14) permits the assignment of a separate noise variance to each observation in the training data set, which may itself be aggregated from multiple data sources. Furthermore, σ_{ϵ}^2 need not match the variance for any previous observation since it corresponds to a new observation that one may know in advance to have a different precision.

C. Computing Derivatives

The derivative of a GP is again a GP, because differentiation is a linear operation. To see this, let

$$f_h(\mathbf{x}) = \frac{1}{h} \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} f(\mathbf{x}) \\ f(\mathbf{x} + h\mathbf{e}_k) \end{bmatrix},$$

where $\mathbf{e}_k$ is the k -th natural basis vector. By definition

$$\lim_{h \rightarrow 0} f_h(\mathbf{x}) = \frac{\partial f}{\partial x_k}.$$

With application of property (4) one then finds that

$$\frac{\partial f}{\partial x_k} \sim \mathcal{GP} \left(\frac{\partial m(\mathbf{x})}{\partial x_k}, \frac{\partial^2 k(\mathbf{x}, \mathbf{x}')}{\partial x_k \partial x'_k} \right).$$

D. Kernel Selection

Looking at (12), we observe that the estimate is a weighted sum of basis functions $k(\mathbf{x}_i, \mathbf{x})$, for $i = 1, \dots, M$. In this sense the GP estimate $E[f(\mathbf{x})|\mathbf{z}]$ inherits properties from the selected kernel function. A common starting choice, and the one used in this paper, is the *square-error* kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp \left\{ -\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2} \right\}, \quad (20)$$

where $\|\cdot\|$ is the euclidean vector norm, σ_f^2 sets the marginal variance of a sample from the GP at any fixed input, and ℓ is a length-scale parameter of the distribution. The sampled functions from a GP defined with this kernel function have mean-square derivatives of all orders and are therefore very smooth [7, pg. 83]. The length-scale parameter characterizes how quickly the function output can change. Notice that the square error kernel function is stationary, i.e., its length-scale does not change as a function of $\mathbf{x}$. This property can be a detriment for systems that change rapidly in localized regions, and such systems may be more naturally treated with a Neural Network based approach.

The weighted square-error kernel

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp \left\{ -\frac{1}{2} \sum_{i=1}^d \gamma_i^2 (x_i - x'_i)^2 \right\},$$

is an important extension of the square-error kernel, where γ_i is an inverse length-scale parameter assigned to each of the input dimensions. This kernel function is often a better choice when the input vector $\mathbf{x}$ has dimension greater than one, as it provides flexibility to fit the data at length-scales appropriate for each input dimension. Use of an inverse length-scale parameter is also typically preferred, as it permits use of prior distributions that are concentrated near zero in the hyper-parameter selection process. This biases the inverse length-scale towards smaller values in the absence of data. The Gamma distribution is one such popular choice in Bayesian data analysis.¹⁵

There are many types of kernel functions available and each defines a function space with useful properties. One is the Matérn class, which has parameters to control the differentiability of the kernel. Another is a piecewise polynomial class of functions with compact support. See [7, Ch. 4] for details.

E. Hyper-Parameter Selection

Training the GP model requires finding the weights vector w defined by (11) given the observed data, and usually several other hyper-parameters involved in the problem formulation. For the square-error kernel function, the hyper-parameters are σ_f and ℓ . There are also typically hyper-parameters associated with the specification of D in (16). For example, in the basic case $D = \sigma_m^2 I$, where σ_m^2 is a parameter that is not necessarily known upfront.

To develop a sense for the effect of the hyper-parameters, Figure 1 shows two different results on trimmed rotor-rate data for the quad-rotor model discussed in Section IV below. The left-hand plot contains the result with $\ell = 20$ knots (twice the airspeed sampling resolution), and $\sigma_m = 0.2$ relative to normalized rotor-rate data for which $\sigma_f = 1$ is assumed (see Subsection G below). For all plots, the uncertainty band shown covers $\pm 3\sqrt{\text{Var}[f(\mathbf{x})|\mathbf{z}]}$, where $\text{Var}[f(\mathbf{x})|\mathbf{z}]$ is computed from Equation (18). For the plot on the left, this band appears inflated with respect to how well the mean of the GP is able to predict the observed data. The right-hand plot shows the change, primarily to the uncertainty bands, when both ℓ and σ_m are reduced. In this case the GP model reflects the state-of-knowledge where the observed data is assumed accurate with more variation allowed to occur in-between the observed points. Absent more information, either of these cases may be a fair representation of what is known about the system. Notice too, that in both cases the GP uncertainty increases rapidly (relative to the length-scale) as predictions are extrapolated beyond the region where the anchor points are defined.

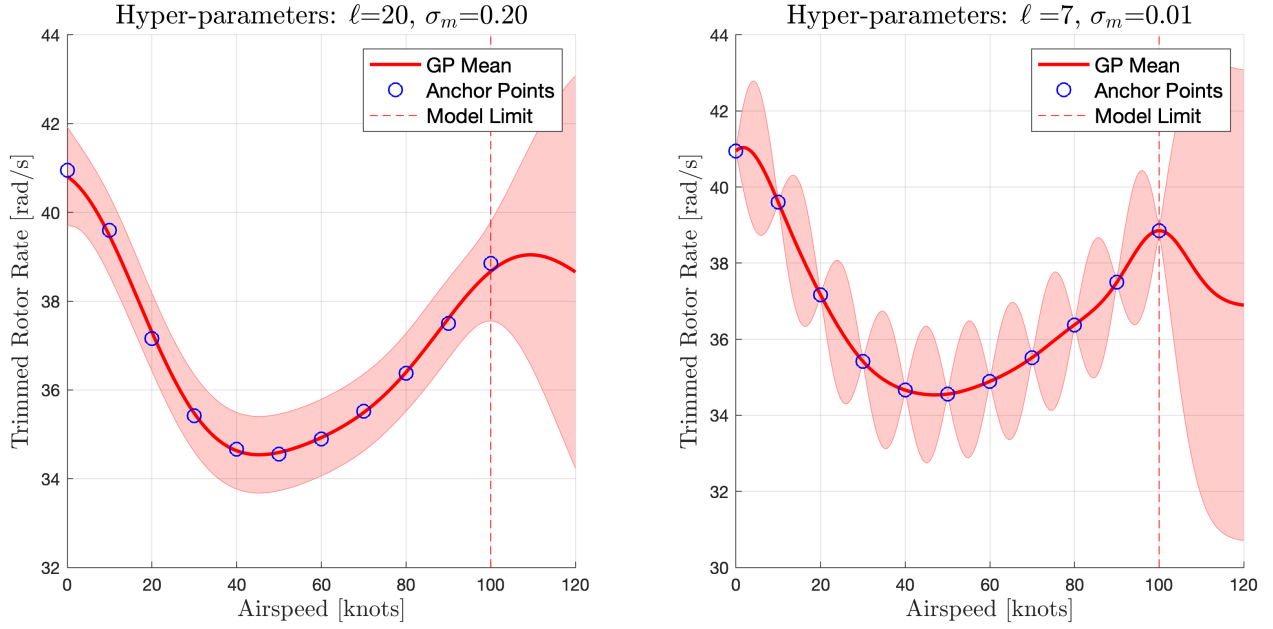


Figure 1: Example GP fits to qLPV model anchor points with uncertainty.

In many cases, one wants to select the hyper-parameters automatically. The frequently used Bayesian approach is discussed here. Let $\theta_{\text{hp}} \in \mathbb{R}^r$ represent the vector of hyper-parameters. Looking back at (16), the marginal distribution for the observed data $\mathbf{z}$ depends on knowing θ_{hp} , i.e.,

$$\mathbf{z}|\theta_{\text{hp}} \sim \mathcal{N}(\boldsymbol{\mu}(\theta_{\text{hp}}), \Sigma(\theta_{\text{hp}}) + D(\theta_{\text{hp}})).$$

The probability density function for $\mathbf{z}|\theta_{\text{hp}}$, denoted as $p(\mathbf{z}|\theta_{\text{hp}})$, tells us the likelihood of having observed $\mathbf{z}$ as a function of θ_{hp} . With a given prior distribution $p(\theta_{\text{hp}})$ and use of Bayes' Theorem one finds that the posterior distribution $p(\theta_{\text{hp}}|\mathbf{z})$ is proportional to $p(\mathbf{z}|\theta_{\text{hp}})p(\theta_{\text{hp}})$. One sensible way to select θ_{hp} is then to choose the one that maximizes this posterior distribution, also known as the maximum a posteriori (MAP) estimate. This is often accomplished by finding θ_{hp} that maximizes

$$\begin{aligned} \log p(\mathbf{z}|\theta_{\text{hp}})p(\theta_{\text{hp}}) = & -\frac{1}{2}(\mathbf{z} - \boldsymbol{\mu}(\theta_{\text{hp}}))^T (\Sigma(\theta_{\text{hp}}) + D(\theta_{\text{hp}}))^{-1} (\mathbf{z} - \boldsymbol{\mu}(\theta_{\text{hp}})) \\ & - \frac{1}{2} \log \det (\Sigma(\theta_{\text{hp}}) + D(\theta_{\text{hp}})) - \frac{M}{2} \log 2\pi + \log p(\theta_{\text{hp}}). \end{aligned} \quad (21)$$

Numerical optimization methods are typically used to find a *local* maxima with respect to θ_{hp} , which relies on having a good starting guess for θ_{hp} . With the solution θ_{hp}^* , the weights are then found by solving

$$\mathbf{w} = [\Sigma(\theta_{\text{hp}}^*) + D(\theta_{\text{hp}}^*)]^{-1} (\mathbf{z} - \boldsymbol{\mu}).$$

An advantage of the Gaussian process approach (over neural networks) is that once θ_{hp} is set, the weights vector $\mathbf{w}$ produces the globally optimal Minimum Mean Square Error estimate of $f(\mathbf{x})|\mathbf{z}$. When used with the square-error kernel, this approach produces a GP model that interpolates across uncertain data points with an underlying assumption of smoothness.

The left-hand plot in Figure 2 shows the resulting GP model fit, at a value θ_{hp}^* that locally maximized the posterior distribution for the same anchor points used in Figure 1. An exponential prior (in the family of Gamma distributions)^c was used in the formulation of Equation (21). The right-hand plot shows the validation error, determined by computing the sample standard deviation of the difference between the GP mean estimate and validation data not included in the training data set. This is of course a better indicator of prediction capability and it may be a preferred method for selecting θ_{hp} when good validation data is

^cSpecifically, $p(\theta_{\text{hp}}) = \beta_1 e^{-\beta_1 \theta_1} \beta_2 e^{-\beta_2 \theta_2}$, where $\theta_{\text{hp}} = (\theta_1, \theta_2)$ with $\theta_1 = \ell^{-2}$, $\theta_2 = \sigma_m^{-2}$, $\beta_1 = 20^{-2}$ and $\beta_2 = 0.02^{-2}$.

available, as the right-hand plot suggests further reducing σ_m to 0.01 to achieve even better predictive performance. However, the Bayesian hyper-parameter selection approach scales well for higher-dimensional θ_{hp} , and with the Gamma prior tends to safely overestimate uncertainty in the absence of data. Overall, the GP model accurately interpolates the trim rotor-rates in the validation data to within an error standard deviation of 0.023 rad/s (or 0.06%), and provides appropriate uncertainty quantification for the prediction at any input. However, this is not a model that represents uncertainty across multiple vehicles with differing performance characteristics, nor is it suitable for even a single vehicle with uncertain characteristics that vary over time. An example tackling such issues is presented in Section IV within the context of (stationary) GP estimation with fixed hyper-parameters.

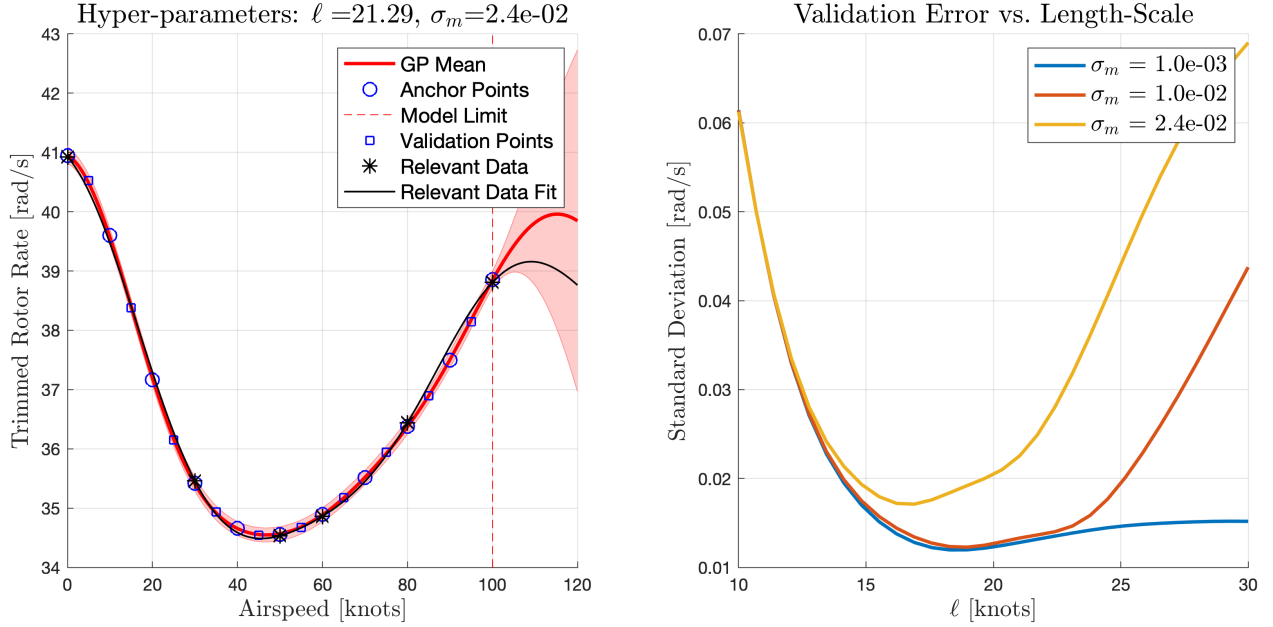


Figure 2: Left: Optimized GP fit to qLPV model anchor points with uncertainty. Right: Validation error standard deviation for varying length-scale (ℓ) and normalized observation noise (σ_m) hyper-parameters.

F. Auto-Relevance Detection

Gaussian processes are often motivated in a Bayesian framework for linear regression as follows. Let

$$f(\mathbf{x}) = m(\mathbf{x}) + \sum_{i=1}^M v_i \phi(\mathbf{x}, \mathbf{x}_i) \quad (22)$$

$$z = f(\mathbf{x}) + \epsilon, \quad (23)$$

where $\phi(\mathbf{x}, \mathbf{x}_k)$ is a basis function assignment (implicitly depending on θ_{hp}) for each $\mathbf{x}_i$ in the training data set, and $v = [v_1, \dots, v_M]^T$. Here, v is a set of weights different from, but related to, the weights w in Equation (11). With $\epsilon \sim \mathcal{N}(0, \sigma_m^2)$ independent and identically distributed for each input $\mathbf{x}_i$, one obtains an expression for the likelihood distribution of the observed training data $\mathbf{z}$ given v and θ_{hp} , denoted $p(\mathbf{z}|v, \theta_{\text{hp}})$. With the additional prior assumption that $v \sim \mathcal{N}(0, \Sigma_p)$, where Σ_p is a given covariance matrix, one ultimately finds that the marginal distribution for $f(\mathbf{x})|\mathbf{z}$ is Gaussian with mean and covariance given by Equations (17) and (18), respectively. For the details of this derivation see [7, Ch. 2].

In his 2001 paper on sparse Bayesian learning, Tipping investigates the use of hyper-parameters $\alpha = [\alpha_1, \dots, \alpha_M]^T$, one for each weight in v along with a prior distribution

$$v_i | \alpha_i \sim \mathcal{N}(0, \alpha_i^{-1}), \quad (24)$$

independent for each $i = 1, \dots, M$, instead of $v \sim \mathcal{N}(0, \Sigma_p)$.¹⁴ A Gamma prior distribution for α is then also assumed, and the marginal likelihood $p(\mathbf{z}|\alpha, \theta_{\text{hp}})$ is maximized with respect to α . The maximization process is then shown to produce large values of α_k for v_k associated with observations that are irrelevant to the predictive performance of $f(\mathbf{x})$. Overall, this establishes an auto-relevance detection capability for simplifying Gaussian process models by removing uninformative data. Figure 2 shows the relevant data and fit estimated with this procedure for the previous example. The initial result is quite satisfying, as about half of the anchor points are eliminated in rough correspondence with the estimated length-scale of about 20 knots. However, the validation error standard deviation increases to 0.12 rad/s (or 0.3%) on the fit with the reduced data set.

G. Practical Matters

When working with GPs for machine learning, it is typically good practice to use the normalized z -score data, i.e., the observations are normalized by subtracting the sample mean and dividing the result by the sample standard deviation. With the data normalized in this way, it is often reasonable to assume $m(\mathbf{x}) = 0$ for all $\mathbf{x}$, and $\sigma_f = 1$. These assumptions are common because they produce a simpler approach that often works well with fewer hyper-parameters.

It sometimes happens in practice that a series of observations are all constant, either exactly or approximately. This could indicate that the value *is* effectively constant, in which case it should be eliminated from the problem. However, it may also indicate that the inputs did not excite a response. The safer assumption in this case, is to assume a worst-case value for the normalizing standard deviation based on the range of physical values the variable might be expected to take.

The mean prediction (17) and variance (18) depend on inverting the matrix $\Sigma + D$. Standard matrix inversion at this stage is $\mathcal{O}(M^3)$, where M is the number of observed data points. This is the reason Gaussian process regression does not scale to large datasets, since the size of Σ grows with M . The matrix $\Sigma + D$, is at least always positive definite when D is positive definite, and in that case it is non-singular and invertible (by Cholesky decomposition). The covariance matrix Σ by itself, may only be semidefinite but the meaning in this case is perfectly well defined. When Σ is semidefinite and $D = 0$, it indicates certainty, i.e., that certain linear combinations of the elements of the random vector have zero variance.

More interestingly, Equation (18) is independent of the observed data once the hyper-parameters are fixed. Therefore, the uncertainty of the model prediction depends only on where the data is observed, not what data is observed. Also, $\Sigma + D$ depends on the observed data only indirectly through the training data that is used to select the hyper-parameters, and with the hyper-parameters fixed, the factorization of $\Sigma + D$ can be stored up front along with the weights vector w . The computational cost for computing the mean prediction and variance at any input $\mathbf{x}$ is then $\mathcal{O}(M)$ and $\mathcal{O}(M^2)$, respectively.

The selection of the hyper-parameters, typically requires use of an unconstrained optimization routine, such as Matlab's `fminsearch`, or `fminunc` in the optimization toolbox. Such routines are generically defined to optimize over an input vector with elements that take on both positive and negative values. However, the objective function (21) is typically ill-defined for negative elements of θ_{hp} . This issue is overcome by optimizing $f_0(e^\xi)$ with respect to ξ , where f_0 is the objective function (21) and $\xi = \log(\theta_{\text{hp}})$.

Finally, with a normalized data set, it is possible to evaluate multiple GPs with a single matrix multiply in the following sense. Let

$$Z = \begin{bmatrix} \mathbf{z}_1 & \dots & \mathbf{z}_N \end{bmatrix}$$

where $\mathbf{z}_k \in \mathbb{R}^M$ is a normalized output observation vector corresponding to its own GP for each $k = 1, \dots, N$. We further assume that each GP has $m(\mathbf{x}) = 0$ for all $\mathbf{x}$, that each is observed on the same set of inputs $\mathbf{x}_1, \dots, \mathbf{x}_M$, and that they all share a common θ_{hp}^* . With the weights matrix

$$W = [\Sigma(\theta_{\text{hp}}^*) + D(\theta_{\text{hp}}^*)]^{-1} Z$$

stored up front, one then evaluates the mean prediction for all the GPs with one vector-matrix multiply, i.e.,

$$\mathbb{E}[f_1(\mathbf{x}), \dots, f_N(\mathbf{x})|Z] = \mathbf{k}_{\mathbf{x}}(\mathbf{x})^T W. \quad (25)$$

An analogous calculation holds for the variance. After the normalized predictions are computed they are unnormalized, individually, by using the stored values for the sample standard-deviations and means. Under these assumptions, the computational cost of evaluating multiple GPs is $\mathcal{O}(NM)$.

III. From qLPV to GPLPV

The standard Linear Parameter Varying (LPV) model is specified as follows

$$\dot{x} = f(x, u, \nu) = A(\nu)[x - x_t(\nu)] + B(\nu)[u - u_t(\nu)], \quad (26)$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the control input vector, and $\nu \in \mathbb{R}^d$ is a model parameter vector. The remaining terms are the state-derivatives matrix $A(\nu) \in \mathbb{R}^{n \times n}$, the control-derivatives matrix $B(\nu) \in \mathbb{R}^{n \times m}$, the trim states $x_t(\nu) \in \mathbb{R}^n$, and the corresponding trim control inputs $u_t(\nu) \in \mathbb{R}^m$. In the quasi-LPV (qLPV) case the look-up parameter vector ν has elements that depend on x . Hereafter, ν is used instead of $\mathbf{x}$ for the GP input vector to avoid confusion with the state variable x .

For a series of parameter anchor points $\nu_1, \dots, \nu_M$, a physics-based numerical code (or system identification from flight-test data)¹⁶ is used to compute the trim state, trim input, and derivatives matrices, i.e., the A , B , x_t and u_t elements of the qLPV model. Many research organizations have preferred in-house codes for accomplishing this step. For rotorcraft conceptual design, the National Aeronautics and Space Administration (NASA) has developed several such tools in collaboration with the U.S. Army Aviation Development Directorate.¹⁷ In particular, the NASA Design and Analysis of Rotorcraft (NDARC)¹⁸ performs mission-based air vehicle design and optimization analysis. NDARC is then used in conjunction with the Simplified Flight Dynamics for Conceptual Design (SIMPLI-FLYD)¹⁷ framework to compute the needed trim state, input, and derivatives matrices. The bare airframe and propulsion models can then be linked with the CONTROL Designer's Unified InTerface (CONDUIT) for control system design and analysis.¹⁹ There are also commercial comprehensive analysis codes that provide similar and higher-fidelity capabilities such as Johnson Aeronautics' CAMRAD II, Advanced Rotorcraft Technologies's FLIGHTLAB, and Continuum Dynamics's CHARM tool.

In a standard implementation of the full-envelope qLPV model, the state accelerations $\dot{x}$ are evaluated at the current system state (x, u, ν) by first linearly interpolating for the elements of $A(\nu)$, $B(\nu)$, $x_t(\nu)$ and $u_t(\nu)$ with respect to ν , and then using the interpolated values to evaluate (26). With GPs the qLPV model is enhanced by replacing the linear interpolation step with a GP predictor for the elements of $A(\nu)$, $B(\nu)$, $x_t(\nu)$ and $u_t(\nu)$. GPs are configurable to include linear (or spline) interpolation, so this is really just a generalization of the standard implementation. Also, GP predictors can be used for all or a selected subset of the elements, many of which are known a priori, and (25) is used for efficient computation where applicable. Leveraging the techniques presented above, this approach provides smooth interpolation between the individual linearized models, uncertainty quantification, auto-relevance detection for data reduction, and the ability to fuse multiple data sets and noisy observations. The GP based approach also gracefully handles higher dimensional parameter vectors, as it does not require any particular input grid or ordering of the parameter data. Also, the marginal variance function (18) quantifies the model uncertainty at any input and this can be used to identify locations where new higher fidelity or experiment data is needed to refine the model. Finally, by using GPs within the qLPV framework, one also preserves the underlying usefulness of the qLPV form for controller design and safety analysis.

A. Rigid-Body Dynamics

In models typical for flight dynamics, the above qLPV model (26) primarily accounts for the aerodynamic and propulsive forces on the airframe. It also subtly includes the body-axis accelerations from gravity at the trim state, essentially, because Equation (26) evaluates to zero (inclusive of gravity) when $x = x_t(\nu)$ and $u = u_t(\nu)$. With the current state x , input u , and parameter vector ν given, it remains to account for the nonlinear gravitational and rigid-body motion effects. Adding these to Equation (26) produces the total model

$$\dot{x} = f(x, u, \nu) - g(x_t(\nu)) + g(x) + h(x), \quad (27)$$

where the state vector x must at least include the body-axis velocities (u, v, w) and rotational rates (p, q, r) needed for the rigid-body equations of motion, along with the Euler angles ϕ and θ needed to evaluate the gravity vector in the body frame. For rotorcraft applications, x may also include rotor and wake states. Under a rigid-body assumption, $f(x, u, \nu)$ provides the accelerations from the qLPV model (26), $g(x)$ computes acceleration terms due to gravity (and potentially other trim forces or moments), and $h(x)$ accounts for the remaining accelerations proscribed by the rigid-body equations of motion, such that $h(x_t(\nu))=0$. Note

that $\dot{x} = 0$ when $x = x_t(\nu)$ and $u = u_t(\nu)$ as required. A more detailed derivation with similar notation is presented in our earlier paper,⁵ and in previous work validated for specific rotorcraft by other authors.¹⁻⁴

B. Linear Approximation

Fundamental control system analysis and design techniques require the use of linear model approximations to (27) at any desired *arbitrary* operating point (x_0, u_0) . The linear approximation is obtained through the standard Taylor series approximation

$$\delta\dot{x} = \mathbf{J}_x\delta x + \mathbf{J}_u\delta u + \mathbf{f}(x_0, u_0, \nu), \quad (28)$$

where $\delta x = x - x_0$, $\delta u = u - u_0$, and $\mathbf{f}(x_0, u_0, \nu)$ is the total model (27). The matrix $\mathbf{J}_x \in \mathbb{R}^{n \times n}$ is the total Jacobian matrix of $\mathbf{f}$ with respect to state x , e.g., if ν is independent of x , then $\mathbf{J}_x = J_x^f + J_x^g + J_x^h$, where J_x^f , J_x^g , and J_x^h are the respective Jacobian matrices of f , g , and h with respect to x . The matrix $\mathbf{J}_u = J_u^f$, because the Jacobian matrix for g and h with respect to u is zero. The J_x^g and J_x^h matrices are regarded here as standard and well known for the rigid-body mechanics assumption.

It holds that $J_x^f = A(\nu)$ and $J_u^f = B(\nu)$ for the LPV case where ν is independent of x . The qLPV case is trickier. In this case ν is a function of x , and after applying the chain-rule to

$$f(x, u, \nu(x)) - g(x_t(\nu(x)))$$

it follows that

$$[J_x^f]_{ij} = \frac{\partial f_i}{\partial x_j} + \sum_{k=1}^d \frac{\partial f_i}{\partial \nu_k} \frac{\partial \nu_k}{\partial x_j} - \sum_{l=1}^n \sum_{k=1}^d \frac{\partial g_l}{\partial x_{tl}} \frac{\partial x_{tl}}{\partial \nu_k} \frac{\partial \nu_k}{\partial x_j} \quad (29)$$

where $[J_x^f]_{ij}$ denotes the i, j -th element of the Jacobian matrix J_x^f . In matrix form, the same result is expressed as

$$J_x^f = A(\nu) + J_\nu^f J_x^\nu - J_{x_t}^g J_\nu^{x_t} J_x^\nu.$$

Though tedious, this computation is usually tractable. With the GP approach, one may also intentionally neglect the parasitic term $J_\nu^f J_x^\nu - J_{x_t}^g J_\nu^{x_t} J_x^\nu$, and treat it as a contributor to the uncertainty in $A(\nu)$.

IV. Demonstration

This section explores use of the proposed methodology to develop a model with motor parameter uncertainty for a six passenger electric quad-rotor air-taxi, specifically designed for this study. This vehicle concept is shown in Figure 3. This model is a rotor-speed controlled variant of the collective controlled, electric propulsion, six-passenger (1,200-lb payload) NASA reference concept quadcopter designed to the same common sizing mission profile.²⁰ A summary of design parameters for the rotor-speed and collective controlled aircraft is given in Table 1.

Bare-airframe linear stability and control derivative models of the quad-rotor configuration of this study were generated following the SIMPLI-FLYD framework,¹⁷ based on aircraft designs and performance maps



Figure 3: Quad-rotor electric propulsion reference design.

Table 1: Basic characteristics of NASA quad-rotor configurations.

Configuration	Collective	RPM Control
Design Gross Weight (lb)	6,480	5,716
- Payload	1,200	1,200
- Weight Empty	5,269	4,507
- Operating Weight	5,279	4,517
Capacity (passengers + crew)	6	6
Number of rotors	4	4
Disk loading (lb/ft ²)	3.0	3.0
Number of blades	3	3
Rotor radius (ft)	13.1	12.3
Solidity, thrust-weighted	0.055	0.055
Hover tip-speed (ft/s)	550	550
Rotational speed (rad/s)	42.0	44.7
Flapping frequency (/rev)	1.030	1.030
Lock number	5.16	4.95
Moments of inertia (slug ft ²)		
- I_{xx}	15,540	12,094
- I_{yy}	16,963	13,201
- I_{zz}	20,525	15,973
- I_{xy}	0	0
- I_{yz}	0	0
- I_{xz}	0	0
Rotor moment of inertia (slug ft ²)	265.8	202.6
Propulsion group	Centralized	Distributed
Number of motors	4	4
Sea Level Static power available per motor (hp)	168.0	130.5
Specification engine speed (rpm)	8,000	8,000
Engine power limit (hp)	319.9	254.3
Drive system torque limit (lb-ft)	8,389	1,566
Battery energy capacity (MJ)	1,331	1,196

obtained using the rotorcraft design tool NDARC^{21,22} for discrete level flight points within the operating envelope at an altitude of 6k/ISA. Specifically, a blade element model using quasi-steady airfoil lift and drag coefficient polynomial representations and 3-state dynamic inflow (Peters-HaQuang) with wake curvature for modeling of the rotor aerodynamics was used.²³ The rotor model geometry included the span-wise variation of blade chord and twist. The flapping dynamics of the articulated rotors were modeled assuming rigid blades with a flapping hinge offset. The rotor blade mass and inertia properties were represented by lumped parameters for the total blade. Rotational speed dynamics were implemented by explicitly modeling the rotor shaft torque reactions on the rotors and the airframe, allowing for prescribed shaft torque inputs into the system. This later development allowed for the separate modeling and integration of the electric propulsion systems with the bare-airframe flight mechanics equations of motion. Simple fuselage aerodynamic contributions were limited to the drag, as defined by drag coefficient and surface wetted area parameters.

The order of the linear model at this stage was reduced with steady-state gain matching so as to preserve quasi-steady effects. Only the coupled airframe rigid-body, first-order flapping and rotor speed states were retained prior to their implementation into the qLPV framework, such that the individual point models are described by 6 rigid-body states (u, v, w, p, q and r) and 3 rotor states ($\beta_{1c_i}, \beta_{1s_i}$ and Ω_i) per rotor. For controller design purposes, the unobservable flapping states ($\beta_{1c_i}, \beta_{1s_i}$) were also eliminated with steady-state gain matching, and a voltage-controlled DC motor model was incorporated into the overall system using a previously developed method.⁵ At this stage, the stability and control derivatives represent only linear perturbation aerodynamic and propulsion effects. Rigid-body dynamics, including the non-linear Coriolis and gravitational effects were added downstream to the qLPV model as discussed in III.A above. The state and input vectors for the final model were

$$x = \begin{bmatrix} \Omega_i & \text{angular rate of rotor } i = 1, \dots, 4 \\ \phi & \text{roll angle} \\ \theta & \text{pitch angle} \\ \psi & \text{yaw angle} \\ h & \text{altitude} \\ u & x\text{-body-axis velocity} \\ v & y\text{-body-axis velocity} \\ w & z\text{-body-axis velocity} \\ p & \text{roll rate around } x\text{-body-axis} \\ q & \text{pitch rate around } y\text{-body-axis} \\ r & \text{yaw rate around } z\text{-body-axis} \end{bmatrix} \quad u = \begin{bmatrix} V_1 & \text{voltage input for motor 1} \\ V_2 & \text{voltage input for motor 2} \\ V_3 & \text{voltage input for motor 3} \\ V_4 & \text{voltage input for motor 4} \end{bmatrix}.$$

The qLPV parameter ν was the true airspeed in level flight with zero side-velocity component ($v_t = 0$, and $\phi_t = 0$), i.e., $\nu = \sqrt{u_t^2 + w_t^2}$, where (u_t, v_t, w_t) are the trim speed components in the body-axes. With a given parameter ν and function for the trimmed pitch $\theta_t(\nu)$, the trim airspeeds in the body-axes are determined as $u_t = \nu \cos \theta_t$, $v_t = 0$, and $w_t = \nu \sin \theta_t$ for $\nu_k = 10k$ knots with $k = 0, \dots, 10$. With this setup, $d = 1$, $n = 14$, $m = 4$, and $M = 11$. A GP for each of the elements in $x_t(\nu)$, $u_t(\nu)$, $A(\nu)$ and $B(\nu)$ would amount to $N = n + m + n^2 + nm = 270$ component GP models. However, typically many of these components are known. For example, by definition the body-axis angular rates (p_t, q_t, r_t) are zero in trim. Furthermore, v_t and ϕ_t are zero in trim, and $u_t = \nu \cos \theta_t$ and $w_t = \nu \sin \theta_t$. Many elements of the $A(\nu)$ and $B(\nu)$ matrices are also typically known. In short, not every element of the LPV model requires a GP, and one can even select particular elements of interest to the application. In any case, even when all elements are estimated, the overall simulation from the GP model with the use of Equation (25) has roughly the same computational cost as the basic qLPV model.

Figure 4 plots the GP model for representative elements of $A(\nu)$, $B(\nu)$, and $x_t(\nu)$. The specific elements were selected to highlight the main caveats of the approach. The anchor point data was z -scored and used to train the GP with a square-error covariance kernel with length-scale parameter ℓ and constant training observation variance σ_m^2 as discussed in Section II.E. A single normalized prediction variance parameter σ_ϵ^2 was then manually selected so that the GP variance estimate from Equation (19) captured the worst-case uncertainty across all state and control derivatives. This uncertainty was produced by adding normally distributed random noise, with standard deviation at 1% of the nominal value, to the motor parameters for operating efficiency, armature resistance, and back-EMF constant. Each element in a set of ten sampled

motor parameters was then used to generate new model data depicted by the red x's in the plots. The top plots in the figure show examples of the affected rotor-rate and control derivatives for the voltage controlled system, where the rotor-shaft torques depend on motor currents, which are determined by the voltage inputs, motor parameters, and rotor-rates.⁵

The motor parameter uncertainty, however, does not affect other variables as one should expect. For example, the x -body-axis velocity derivative ($\partial \dot{u}/\partial u$) and the trimmed pitch angle (θ) are determined from aerodynamic properties (e.g., drag) that are independent of the motor parameters. These are shown on the bottom of Figure 4, with an inflated uncertainty relative to the prediction error. The uncertainty is inflated due to the use of a single worst-case σ_ϵ that is forced on all the (normalized) LPV data in $A(\nu)$, $B(\nu)$, $x_t(\nu)$, $u_t(\nu)$. Since the intent is to use the uncertainty in a robustness analysis, the inflated values are not too offensive, and may actually represent implicit uncertainties in the system. For example, uncertainty in the vehicle's mass would produce uncertainty in $\partial \dot{u}/\partial u$. Biased tail or headwind measurements would cause uncertainty in the trim θ .

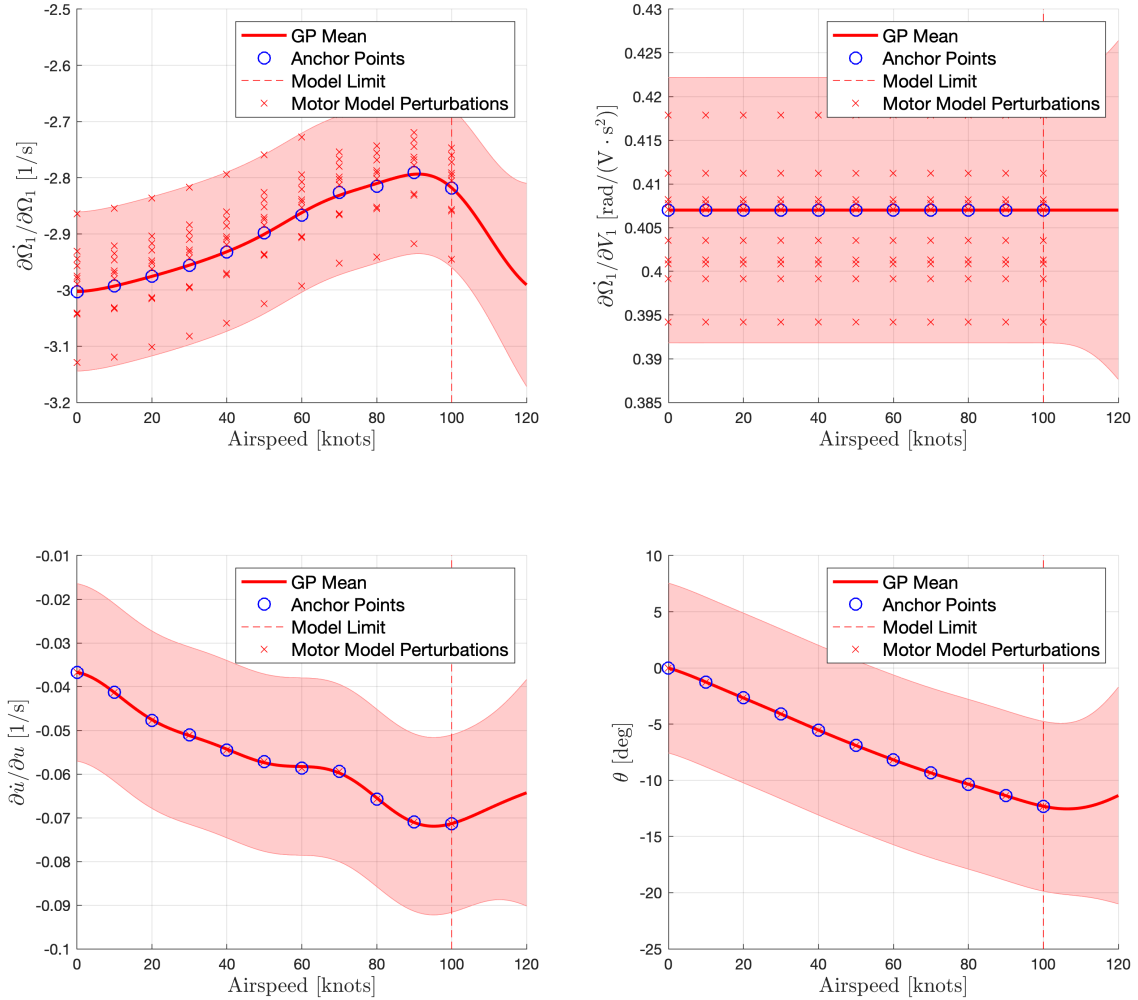


Figure 4: Example quad-rotor GP model with uncertain motor parameters.

Overall, this GP models a simulated scenario where one wants to design a controller robust to uncertainty across multiple vehicles with manufacturing and operating uncertainty in the motor parameters. To demonstrate this capability, a full-state feedback Linear-Quadratic-Regulator (LQR) controller was designed based on the GP mean model evaluated at a particular airspeed. The nominal closed-loop system met disturbance rejection peak and bandwidth requirements for rotorcraft from [24, Table 1, pg. 38]. With the given linearized model and controller, both stability and performance with uncertainty can be ensured by μ -analysis,

see for example [25, Ch. 8]. One such analysis is shown here using the `robstab` function in MATLAB's Robust Control Toolbox, with uncertain system and control matrices defined as

$$A_{\text{unc}} = \bar{A}(\nu) + k_A \Delta A(\nu) \quad (30)$$

$$B_{\text{unc}} = \bar{B}(\nu) + k_B \Delta B(\nu), \quad (31)$$

where $\bar{A}(\nu)$ and $\bar{B}(\nu)$ are the state and control derivatives from the qLPV model (26) inclusive of known derivative terms for the rigid-body dynamics. The terms $\Delta A(\nu)$ and $\Delta B(\nu)$ are the element-wise GP observation standard deviations produced from the square-root of the variance in (19). The variables k_A and k_B were configured as the uncertain parameters in the analysis, such that the uncertainty bounds shown in Figure 4 correspond to $k_A = k_B = 3$. The `robstab` function is then run on the closed-loop system for A_{unc} and B_{unc} with the fixed constant gain feedback matrix designed for the nominal system. Figure 5 shows results from a sweep across a dense grid of airspeeds with 0.5-knot spacing. The uncertainty tolerance is the lower-bound on the stability margin estimated by `robstab`, and values greater than one indicate the design is stable for all modeled uncertainty. For airspeeds up to the modeled limit of 100 knots, an LQR controller exists that is quite stable relative to the GP uncertainty, though, mildly diminishing uncertainty tolerance is observed with increasing airspeed. The analysis also identifies the minimum k_A and k_B needed to destabilize the system. This is just one example showing how the Gaussian process qLPV model uncertainty can be connected to important tools from the field of robust control.

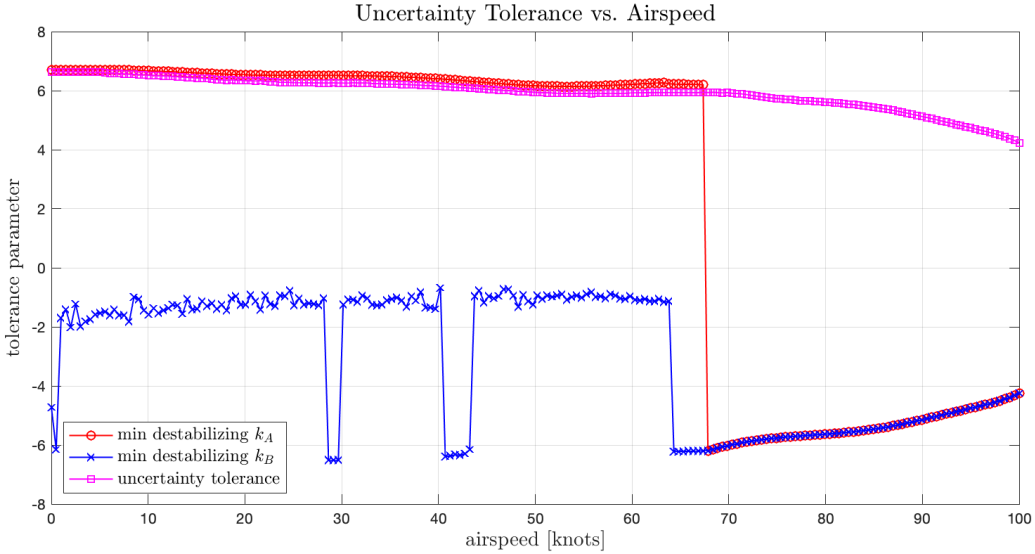


Figure 5: Example robust stability analysis vs. airspeed.

V. Conclusions

Gaussian process machine learning was developed to enhance the commonly used qLPV model. It is recognized that overall the qLPV model is non-linear, but when the parameter vector is fixed it takes a linear form with respect to the other variables. While such model forms are not necessarily true to the underlying-physics, they are computationally efficient and have proven very useful in rotorcraft applications. Through the use of Gaussian process regression, the qLPV model is imbued with smooth non-linear interpolation through parameter space, with individual uncertainty quantification on the constituent model elements (state and control derivatives, etc.) while maintaining computational efficiency. The approach is significant because it contributes to methodologies for model estimation and uncertainty quantification that can manage multiple-data sets, while scaling with the emerging complexity of new rotor-craft designs and maintaining a model form compatible with existing tools for robust controls analysis.

Acknowledgements

This work was supported by NASA’s Revolutionary Vertical Lift Technology project.

References

- ¹Lior Zivan and Mark B. Tischler. Development of a full flight envelope helicopter simulation using system identification. *Journal of the American Helicopter Society*, 55, 2010. DOI: 10.4050/JAHS.55.022003.
- ²Ben Lawrence, Carlos A. Malpica, and Colin R. Theodore. The development of a large civil tiltrotor simulation for hover and low-speed handling qualities investigations. In *36th European Rotorcraft Forum*, September 2010.
- ³Eric L. Tobias, Frank C. Sanders, and Mark B. Tischler. Full-envelope stitched simulation model of a quadcopter using stitch. In *AHS International 74th Annual Forum & Technology Display*, May 2018.
- ⁴NATO STO AVT-296 Research Task Group. Rotorcraft flight simulation model fidelity improvement and assessment. Technical Report TR-AVT-296-UU, North Atlantic Treaty Organization, May 2021.
- ⁵Stefan Schuet, Carlos Malpica, Thomas Lombaerts, John Kaneshige, Shannah Withrow, Gordon Hardy, and Jeremy Aires. A modeling approach for handling qualities and controls safety analysis of electric air taxi vehicles. In *AIAA AVIATION FORUM*, June 2020. DOI: 10.2514/6.2020-3188.
- ⁶Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 514–520. MIT Press, 1996.
- ⁷C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- ⁸Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125 – 141, 2018. DOI: 10.1016/j.jcp.2017.11.039.
- ⁹Yuichi Kuya, Kenji Takeda, Xin Zhang, and Alexander I. J. Forrester. Multifidelity surrogate modeling of experimental and computational aerodynamic data sets. *AIAA Journal*, 49(2):289–298, 2011. DOI: 10.2514/1.J050384.
- ¹⁰Alex Feldstein, David Lazzara, Norman Princen, and Karen Willcox. Multifidelity data fusion: Application to blended-wing-body multidisciplinary analysis under uncertainty. *AIAA Journal*, 58(2):889–906, 2020. DOI: 10.2514/1.J058388.
- ¹¹Ryan D. Jackson, Michael Jump, and Peter L Green. Towards gaussian process models of complex rotorcraft dynamics. In *AHS International 74th Annual Forum & Inn Technology Display*, May 2018.
- ¹²Jaime F. Fisac, Anayo K. Akametalu, Melanie N. Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2019. DOI: 10.1109/TAC.2018.2876389.
- ¹³Sylvia Herbert, Jason J. Choi, Suvansh Sanjeev, Marsalis Gibson, Koushil Sreenath, and Claire J. Tomlin. Scalable learning of safety guarantees for autonomous systems using hamilton-jacobi reachability, 2021. arXiv: 2101.05916.
- ¹⁴Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.
- ¹⁵A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 2nd edition, 2004.
- ¹⁶M. B. Tischler. *Aircraft and Rotorcraft System Identification: Engineering Methods with Flight Test Examples*. American Institute of Aeronautics and Astronautics, Inc., Reston, VA, 2nd edition, 2012.
- ¹⁷B. Lawrence, C. R. Theodore, W. Johnson, and T. Berger. A handling qualities analysis tool for rotorcraft conceptual designs. *The Aeronautical Journal*, 122(1252):960–987, June 2018.
- ¹⁸W. Johnson. NDARC, NASA design and analysis of rotorcraft. Technical Report TP 2009-215402, National Aeronautics and Space Administration, 2009.
- ¹⁹M.B. Tischler, J. Colbourne, M. Morel, D. Biezad, K. Cheung, W. Levine, and V. Moldoveanu. A multidisciplinary flight control development environment and its application to a helicopter. *IEEE Control Systems Magazine*, 19(4):22–33, August 1999.
- ²⁰Christopher Silva, Wayne R. Johnson, Eduardo Solis, Michael D. Patterson, and Kevin R. Antcliff. VTOL urban air mobility concept vehicles for technology development. In *2018 Aviation Technology, Integration, and Operations Conference*, Dallas, TX, June 2018. DOI: 10.2514/6.2018-3847.
- ²¹W. Johnson. NDARC-NASA design and analysis of rotorcraft theoretical basis and architecture. In *American Helicopter Society Specialists’ Conference on Aeromechanics*, January 2010.
- ²²W. Johnson. NDARC-NASA design and analysis of rotorcraft validation and demonstration. In *American Helicopter Society Specialists’ Conference on Aeromechanics*, January 2010.
- ²³D. A. Peters and N. HaQuang. Dynamic inflow for practical applications. *Journal of the American Helicopter Society*, 33(4):64–68, 1988.
- ²⁴Chris L. Blanken, Mark B. Tischler, Jeff A. Lusardi, and Tom Berger. Proposed revisions to aeronautical design standard - 33E (ADS-33E-PRF) toward ADS-33F-PRF. Special Report SR-FCDD-AMV-19-01, U.S. Army Aviation Development Directorate, September 2019.
- ²⁵Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. John Wiley & Sons, 2nd edition, 2001.