CEAS-GNC-2022-063

CEAS EuroGNC 2022

"Conference on Guidance, Navigation and Control"

3-5 May 2022 @ Technische Universität Berlin, Germany

Method to Account for Estimator-Induced Previewed Information Losses – Application to Synthesis of Lidar-Based Gust Load Alleviation Functions

Davide Cavaliere PhD Researcher, DLR (German Aerospace Center), Institute of Flight Systems, 38108 Braunschweig, Germany. davide.cavaliere@dlr.de

Nicolas Fezans Scientific Advisor, DLR (German Aerospace Center), Institute of Flight Systems, 38108 Braunschweig, Germany. nicolas.fezans@dlr.de

Daniel Kiehn Research Scientist, DLR (German Aerospace Center), Institute of Flight Systems, 38108 Braunschweig, Germany. daniel.kiehn@dlr.de

ABSTRACT

Preview control using wind estimates derived from Doppler wind lidar measurements is a promising technique for designing active gust load alleviation functions. Due to high noise levels in the lidar measurements, the associated estimator must use some type of smoothing to obtain a reasonable estimate, which entails a loss of some information. Taking this loss into account during control synthesis should help ease the tuning procedure and improve the performance and robustness of the resulting controller. This paper proposes a method to consistently design a linear filter which closely approximates the behavior of the wind estimator and which can be integrated into a linear robust control framework. Its characteristics are shown to closely match the real estimator over a range of types of turbulence using a standard set of system parameters, and in a control synthesis example, it demonstrates a significant improvement in load alleviation performance.

Keywords: Preview control; Lidar-based gust load alleviation; Robust control; Estimation losses

Nomenclature

$\overrightarrow{\delta}_{act}$	=	Commanded control surface deflections
$\delta_{noise,i}$	=	Measurement noise of measurement i
$D(z)$	=	Tapped delay line
η_{apert}	=	Lidar sensor aperture angle
f_{scan}, ϕ_{scan}	=	Lidar sensor scanning rotation rate and angle
γ_1, γ_2	=	Tikhonov regularization weighting parameters
GLA	=	Gust Load Alleviation
$G(z)$	=	Aircraft model
H	=	Discrete gust half-length
J	=	Jacobian of the wind field estimation algorithm
$\widetilde{J}$	=	Jacobian augmented with regularization terms
K_{interp}	=	Interpolation matrix which converts $\overrightarrow{w}_{filt,est}$ to $\overrightarrow{w}_{filt,ctrl}$
K_{WFE}	=	Filter matrix approximating the wind field estimation algorithm

LOS	=	Line Of Sight
LTI	=	Linear Time Invariant
$n_{meas,LOS}$	=	Number of measurement points along the lidar LOS
n_{meas}	=	Total number of measurements available for wind field estimation
n_{nodes}	=	Number of nodes in the estimated wind field
PRF	=	Pulse Repetition Frequency (of the lidar)
PAP	=	Power Aperture Product
PSD	=	Power Spectral Density
$\tilde{r}, r$	=	Residual vector with and without regularization terms, respectively
R_i	=	LOS range of a lidar measurement
R_{min}	=	Minimum measurement range
ΔR	=	Range resolution
σ_i	=	Standard deviation of measurements (due to measurement noise)
θ	=	Vector of estimated wind field parameters (i.e. vertical wind speeds)
$\theta_{p,i}, \theta_{p+1,i}$	=	Wind field nodes adjacent to measurement i
τ_{lead}, τ_{lag}	=	Fore and aft limits (in relative time) of the estimation window
Δt_θ	=	Relative time between adjacent estimation wind field nodes
Δt_{ctrl}	=	Controller sampling time
$\overrightarrow{u}_{meas}$	=	Relative wind speed measurements collected by the lidar sensor
V_{TAS}	=	True airspeed
$\overrightarrow{V_K}$	=	Aircraft inertial velocity vector
$\overrightarrow{w}_{ctrl}$	=	Controller wind field
$\overrightarrow{w}_{est}$	=	Estimated wind field
$\overrightarrow{w}_{filt,ctrl}$	=	Filtered vertical wind speeds at the controller wind field nodes' position
$\overrightarrow{w}_{filt,est}$	=	Filtered vertical wind speeds within the estimation window
$\overrightarrow{w}_{turb}$	=	Vertical wind speed caused by atmospheric turbulence
$\overrightarrow{w}_{turb,ctrl}$	=	True vertical wind speeds at the controller wind field nodes' positions
$\overrightarrow{w}_{turb,est}$	=	True vertical wind speeds within the estimation window
$w_{turb,lead}$	=	True vertical wind speed at the forwardmost point in the estimation window
$w_{turb,ctrl,lead}$	=	True vertical wind speed at the forwardmost node of the controller wind field
$\overrightarrow{w}_{turb,AC}$	=	Vertical wind speeds impacting the aircraft
$w_{turb,0}$	=	True vertical wind speed at the aircraft nose
$\overrightarrow{x_k}$	=	Flight path direction
x_{lead}, x_{lag}	=	Fore and aft bounds of the wind field estimation window
Δx_θ	=	Distance between adjacent estimation wind field nodes
Δx_{ctrl}	=	Distance corresponding to one controller time step at constant airspeed
$\Delta x_{a,i}$	=	Distance between measurement i and $\theta_{p,i}$
$\Delta x_{b,i}$	=	Distance between measurement i and $\theta_{p+1,i}$
y_i	=	Estimated vertical wind speed at the location of measurement i
z_i	=	Measured vertical wind speed component of measurement i

1 Introduction

Modern aircraft development places a strong emphasis on improved efficiency and reduced emissions. As part of these efforts, active control of loads in flexible structures not only offers the opportunity to improve flying qualities and reduce structural weight, but also enables otherwise impractical design features, such as high aspect ratio wings. Gust load alleviation (GLA) is particularly challenging due to the inherent unpredictability of atmospheric turbulence and to the limitations of flight control systems, which together make it difficult to detect and react to disturbances in a timely manner.

Unlike more traditional feedback-based GLA systems [1], feedforward GLA functions using directly-sensed wind information can compensate for both system delays and aircraft dynamics to reduce the turbulence loads. Direct sensing methods for GLA have included angle-of-attack vanes [2] and differential pressure sensors [3], however the lead-time that can be obtained this way is limited, especially at high speeds.

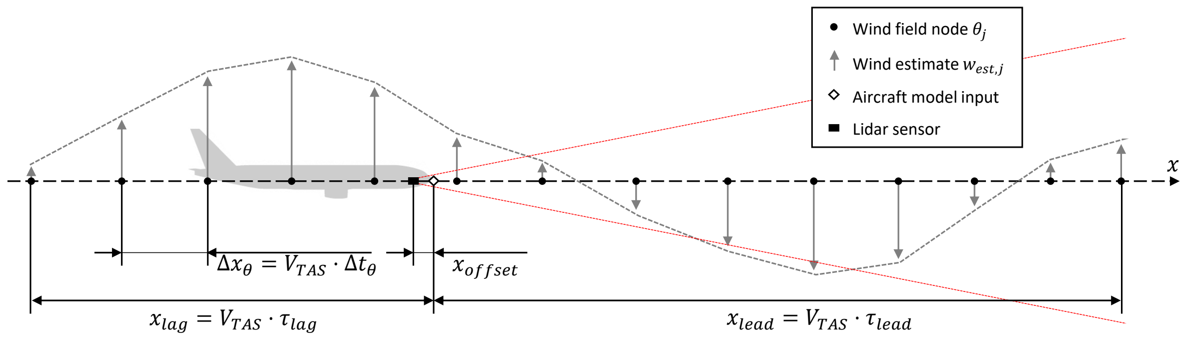
Starting principally with the AWIATOR project [4], lidar systems capable of detecting the relative wind vector well ahead of the aircraft have also been developed. With an effective range up to a few hundred meters ahead of the aircraft, this technology is particularly promising for gust load alleviation [5]. For example, it enables the adoption of a pitching strategy to alleviate aerodynamic loads by controlling the aircraft's overall angle of attack, which would otherwise be difficult or impossible due to slow short-period dynamics and late gust detection. On the other hand, like any feedforward control, lidar-based GLA functions are sensitive to model uncertainties, especially those inherent to the lidar system. Given that the airframe's structural integrity may potentially rely on such systems, they must be designed to be either adaptive or robust with respect to such uncertainties. Several different approaches to lidar-based controllers, including MPC [6], adaptive control [7], and robust control [8] have been developed. Preview control using a wind field estimated from a set of lidar wind measurements has proven to be an effective method to design such functions [9–11], and serves as the principal control framework for this paper.

In the interest of improving the control design process for lidar-based GLA as well as the resulting controllers, the present work proposes a linear filter which models the characteristics of the true lidar wind estimation system. The following section (Sec. 1.1) explains this concept in greater detail. In Section 2, the design and assembly of the proposed filter are described, as well as the key assumptions and simplifications upon which it rests. In Section 3, the characteristics and performance of the filter are evaluated. Finally, in Section 4, the filter is applied to a control design example and evaluated.

1.1 Problem definition

The lidar-based gust detection system considered here is the same as that described in [12], which establishes a public GLA benchmark based on the Common Research Model, and which also describes a benchmark control challenge and a default controller. The lidar system consists of a lidar sensor and a wind field estimation algorithm which fits a linear, free-form model of the vertical wind speed along a section of the aircraft's flight path to the set of lidar measurements stored in memory [9]. Note that only the vertical wind speed is considered here because vertical gusts are more critical in terms of loads and load alleviation, however lateral and longitudinal wind speeds could just as easily be estimated from the same measurements.

The wind field model, shown in Fig. 1, consists of a set of n_{nodes} evenly spaced nodes aligned with the aircraft's flight path direction $\overrightarrow{x_k}$. The space between the first and last nodes is referred to as the *estimation window* and is defined relative to the aircraft itself. Its parameter vector θ has n_{nodes} elements, each of which represents the value of the estimated vertical wind speed at the x-positions of each wind field node $w_{est,j}$. This assumes that the vertical wind speed is uniform in the vertical and spanwise directions, so the vertical wind field profile varies only along $\overrightarrow{x_k}$. Between nodes, the estimated vertical wind speed ($y(\theta)$) is the linear interpolation of the values at the two adjacent nodes. The nodes are numbered from aft to fore, such that θ_1 is at position $x_{lag} = -\tau_{lag} V_{TAS}$ while $\theta_{n_{nodes}}$ is at position $x_{lead} = +\tau_{lead} V_{TAS}$.



● Wind field node θ_j
Wind estimate $w_{est,j}$
◇ Aircraft model input
■ Lidar sensor
x
$\Delta x_\theta = V_{TAS} \cdot \Delta t_\theta$
x_{offset}
$x_{lag} = V_{TAS} \cdot \tau_{lag}$
$x_{lead} = V_{TAS} \cdot \tau_{lead}$

Fig. 1 Wind field model

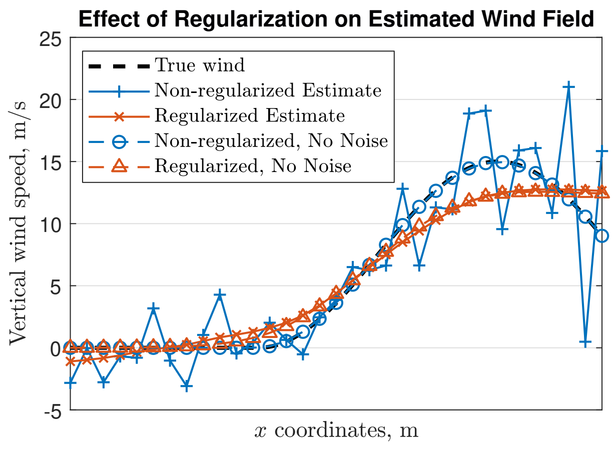


Fig. 2 Comparison between regularized and non-regularized vertical wind field estimates ($\vec{w}_{est}$), both with and without measurement noise.

Due to the fairly high level of noise in the individual measurements, Tikhonov regularization terms are introduced into the estimation problem to obtain a reasonably smooth estimate. Figure 2 compares wind field estimates with and without the regularization terms, both with and without measurement noise. Without noise, the non-regularized estimate matches the true wind almost perfectly, however it is ultimately very sensitive to noise. The regularization effect largely filters out higher-frequency noise, but it invariably also results in the loss of a significant part of the actual gust information, even if no actual noise is present in the measurements. This effect is especially strong at locations far ahead of the aircraft (right edge of the figure).

A schematic overview of the full lidar-based gust load alleviation control system (as found in e.g. the 'hybrid' simulation in [12]) is shown in Fig. 3. $\vec{w}_{turb,AC}$ represents the part of the wind field which directly impacts the aircraft and generates gust loads. The lidar sensor produces measurements $\vec{u}_{meas}$ of the relative airspeed along its line of sight (LOS), which includes components of the true vertical wind field $\vec{w}_{turb}$ and of the aircraft's inertial velocity vector $\vec{V_K}$ (in this case equivalent to the true airspeed), as well as their associated metadata, i.e. Earth-relative position, LOS direction, measurement range, expected measurement error, etc [5]. These measurements are collected at a high rate (on the order of 500 Hz) and stored in a database. The wind field estimation algorithm described above processes the contents of the measurement database at a significantly lower rate (10 Hz [10]), producing an estimated vertical wind field $\vec{w}_{est}$, shown already in Fig. 2. The controller K uses the controller wind field $\vec{w}_{ctrl}$ to calculate actuator deflection commands $\vec{\delta}_{act}$. Generally speaking, it does not operate at the same sampling rate as the wind field estimation, nor does it use the same wind field node coordinates. In fact, the controller usually has a much faster sampling rate (on the order of 100 Hz) and more closely spaced wind field nodes, so $\vec{w}_{ctrl}$ needs to be resampled from $\vec{w}_{est}$.

The linear preview control problem used in previous papers (e.g. [11, 12]) is schematically illustrated in Fig. 4. The entire system is modeled and tuned in discrete time with a sampling time equivalent to the desired sampling time of the final controller. The aircraft model $G(z)$ accepts as inputs the true vertical

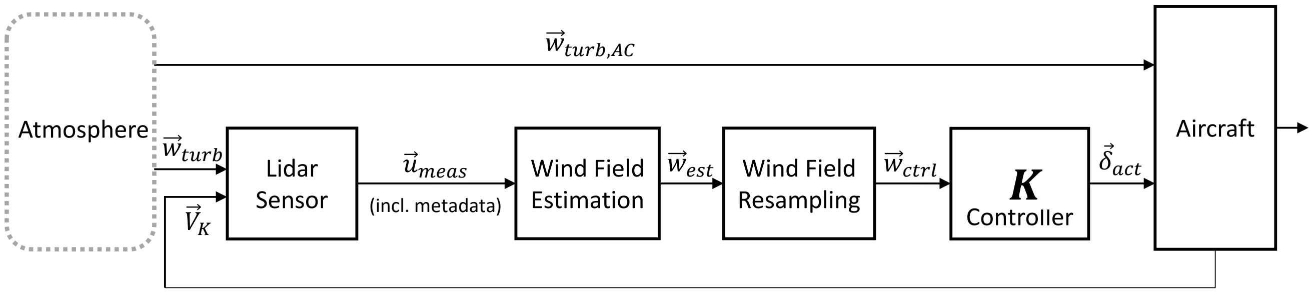


Fig. 3 Schematic view of the lidar-based feedforward gust load alleviation system.

wind speed at the nose $w_{turb,0}$ as well as a set of commanded actuator deflections. The tapped delay line $D(z)$ consists of a series of unit delays which delay the true vertical wind speed at the forwardmost node of the controller wind field $w_{turb,ctrl,lead}$ until it reaches the aftmost node. 'Taps' between the individual unit delays extract the intermediate wind speeds. The number and position of these taps is chosen so as to produce the full controller wind field profile $\vec{w}_{turb,ctrl}$, as well as $w_{turb,0}$.

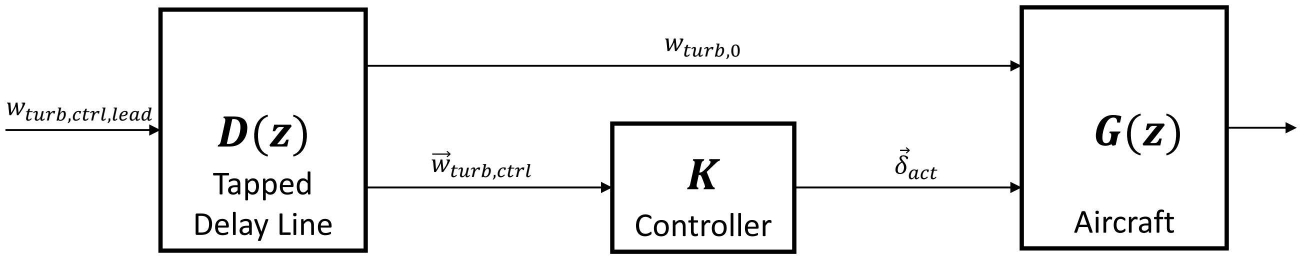


Fig. 4 Schematic view of the legacy control design problem.

Unlike $\vec{w}_{ctrl}$ in Fig. 3, $\vec{w}_{turb,ctrl}$ neglects the effects of the sensing, estimation, and resampling process, so the controller is designed as if it had access to the true wind field. The implication is that, in terms of control synthesis, it receives the full spectrum of wind information up to the Nyquist frequency of the controller, and depending on how the controller and the control design requirements are defined, may try to control the aircraft up to that frequency.

It is clear that such precise wind information is never actually available, and coupling such a controller to a realistic lidar system could result in a significant loss of performance. In fact, control design using the problem in Fig. 4 (see e.g. [11, 12]) typically requires several iterations in which the synthesized controller is evaluated in a higher-fidelity simulation environment including the lidar and wind field estimation. Based on this evaluation, the designer adjusts the control specifications to indirectly compensate for the characteristics of the lidar system. This lengthy and laborious process is necessary because of the nonlinear elements 'Lidar Sensor' and 'Wind Field Estimation.' Attempting to include them in the control problem would prohibit the use of control design tools for linear time invariant (LTI) systems.

To reduce the required time and effort and help ensure the robustness and performance of the controller, here it is proposed to model the nonlinear lidar sensor and wind field estimation as a single-input multiple-output (SIMO) LTI filter which approximates the smoothing effects of the wind field estimation algorithm. Figure 5 shows the control problem including the proposed filter contained in the dashed box. $w_{turb,lead}$ corresponds to the forwardmost node of the estimated wind field and $D(z)$ now produces the true wind field profile within the estimation window $\vec{w}_{turb,est}$. The block K_{WFE} applies the smoothing effect of the wind field estimation algorithm to $\vec{w}_{turb,est}$, yielding the filtered wind field profile $\vec{w}_{filt,est}$. Finally, K_{interp} approximates the wind field resampling process to calculate the controller wind field $\vec{w}_{filt,ctrl}$.

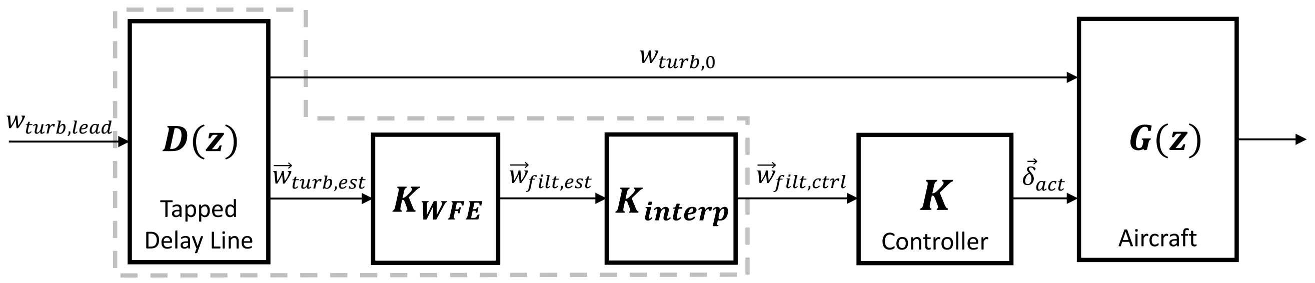


Fig. 5 Schematic view of control problem including the proposed linear filter, contained within the dashed box.

In [13], a way to characterize the low-pass behavior of the lidar wind field estimation based on long ($\approx$1000 s) continuous turbulence simulations and spectral analysis was presented. The obtained spectrum could serve as reference to identify a linear filter matching the true frequency-domain characteristics of the system. While this method would certainly deliver suitable linear filters, it would be fairly computationally expensive. Furthermore, any change in system parameters would require performing another long simulation and reidentifying the filter.

Instead, the present work aims to propose a method to directly calculate such a filter using the parameters of the lidar system and of the wind estimation algorithm. This should result in an efficient, consistent, and potentially even more precise[1] approach.

2 Construction of the linear filter

2.1 Concept and overview

The construction of the linear filter and in particular K_{WFE} is closely tied to the wind field estimation algorithm. This section briefly reviews the contents of the algorithm before outlining the steps needed to derive the full filter. For a more detailed description of the wind field estimation algorithm, see [9].

As described in [9], the wind field estimation algorithm optimizes a regularized least-squares problem which is defined as:

$$\widehat{\theta} = \underset{\theta}{\arg\min} \left(\theta \mapsto \sum_{i=1}^{n_{meas}} \frac{(z_i - y_i(\theta))^2}{\sigma_i^2} + \gamma_1 \, ||\Gamma_1 \, \theta||^2 + \gamma_2 \, ||\Gamma_2 \, \theta||^2 \right). \tag{1}$$

in which n_{meas} is the number of available measurements within the estimation window, z_i is an individual measurement, y_i is the estimated value at the coordinates of a given measurement, and σ_i is the expected standard deviation of a given measurement. σ_i is used to weight the measurements according to their accuracy: measurements with a higher standard deviation are less accurate, and so will have a smaller impact relative to other measurements. Γ_1, Γ_2 and γ_1, γ_2 are the Tikhonov regularization matrices and weights, respectively, and are expressed as:

[1]The continuous turbulence simulation involves random number generation; extremely long simulations would be required to really obtain the best average linear filter.

$$\Gamma_1 = \begin{bmatrix} -1 & +1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & -1 & +1 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & -1 & +1 & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & +1 \end{bmatrix}, \quad \Gamma_2 = \begin{bmatrix} -1 & +2 & -1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & -1 & +2 & -1 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & -1 & +2 & -1 & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & +2 & -1 \end{bmatrix} \tag{2}$$

Γ_1 and Γ_2 represent the finite-difference approximations of the first and second (spatial) derivatives[2] of θ, respectively, such that γ_1 and γ_2 can be used as weights to penalize them. These terms are responsible for creating the smoothing effect shown in Fig. 2.

The optimization problem is solved using the Gauss-Newton algorithm, whose standard expression (with Tikhonov regularization) is:

$$\theta^{[k+1]} = \theta^{[k]} + \left(\widetilde{J}\left(\theta^{[k]}\right)^T \widetilde{J}\left(\theta^{[k]}\right)\right)^{-1} \widetilde{J}\left(\theta^{[k]}\right)^T \tilde{r}\left(\theta^{[k]}\right) \tag{3}$$

in which $\theta^{[k+1]}$ and $\theta^{[k]}$ are the current and previous iterations of the estimate, respectively. $\widetilde{J}(\theta)$ is the augmented Jacobian matrix, and $\tilde{r}(\theta)$ is the augmented residual vector. They are written:

$$\tilde{r}\left(\theta^{[k]}\right) = \begin{bmatrix} r\left(\theta^{[k]}\right) \\ \Gamma\theta^{[k]} \end{bmatrix}, \quad \forall i \in [\![1 \; n_{meas}]\!], \quad r_i\left(\theta^{[k]}\right) = \frac{z_i - y_i\left(\theta^{[k]}\right)}{\sigma_i} \tag{4}$$

$$\widetilde{J}\left(\theta^{[k]}\right) = \begin{bmatrix} J\left(\theta^{[k]}\right) \\ \Gamma \end{bmatrix}, \quad \forall i \in [\![1 \; n_{meas}]\!], \forall j \in [\![1 \; n_{nodes}]\!], \quad J\left(\theta^{[k]}\right)_{(i,j)} = \frac{\partial r_i\left(\theta^{[k]}\right)}{\partial \theta_j} \tag{5}$$

$$\Gamma = \begin{bmatrix} \sqrt{\gamma_1}\,\Gamma_1 \\ \sqrt{\gamma_2}\,\Gamma_2 \end{bmatrix} \tag{6}$$

in which $J(\theta)$ and $r(\theta)$ are the (unaugmented) Jacobian and residual vector, respectively.

Thanks to how the wind field model is defined, the estimation problem has a few convenient properties. Firstly, y (and therefore r) is linear in parameters, so the Jacobian does not depend on θ. As a result, the Gauss-Newton algorithm converges in its first iteration, regardless of the value of $\theta^{[0]}$, so it is safe to impose $\theta^{[0]} = 0$ [9]. Furthermore, looking at the expressions for $\widetilde{J}$ and $\tilde{r}$ in Eqs. 5 and 4, once $\theta^{[0]} = 0$ the contributions of θ and Γ disappear in the product $\widetilde{J}^T\tilde{r}$, leaving $\widetilde{J}^T\tilde{r} = J^T r$. In other words, the smoothing effect of the regularization is wholly contained in the product $\left(\widetilde{J}^T\widetilde{J}\right)^{-1}$. The expression thus reads:

$$\theta^{[1]} = \left(\widetilde{J}^T\widetilde{J}\right)^{-1} J^T r \tag{7}$$

[2]The products $\Gamma_1\theta$ and $\Gamma_2\theta$ are not, of course, the precise derivatives of θ. Rather, they are proportional to the derivatives, which is sufficient in combination with their weights γ_1 and γ_2.

Recall (from Sec. 1.1) that the desired filter is an LTI representation of the average behavior of the wind estimation algorithm, formulated in such a way that it can be efficiently applied to control synthesis. The 'wind field estimation' filter matrix K_{WFE} should take as input the 'true' vertical wind speeds at the estimated wind field node positions $\overrightarrow{w}_{turb,est}$ and produce a 'filtered' estimated wind field $\overrightarrow{w}_{filt,est}$. Essentially, K_{WFE} must geometrically transform $\overrightarrow{w}_{turb,est}$ to 'reconstruct' the measurements in the residual r, from which it then performs an estimation. Based on Eq. 7, K_{WFE} must thus be found such that:

$$\overrightarrow{w}_{filt,est} = K_{WFE}\overrightarrow{w}_{turb,est} = \left(\widetilde{J}^T\widetilde{J}\right)^{-1} J^T r \tag{8}$$

$\overrightarrow{w}_{turb,est}$ is determined by simply delaying the forwardmost measurement $w_{turb,lead}$ n_{nodes} times ($D(z)$ in Fig. 5). This establishes the dynamic part of the filter.

In the real estimation process, J and r are recalculated at each estimation step based on the measurements available within the estimation window. The number, orientation, and relative position of available measurements varies slightly between estimation windows and the aircraft motion can induce differences in the spatial distribution of the measurements and their line-of-sight directions, so the size and composition of J and r vary as well.

K_{WFE} must be constant, so the first task is to create a 'reference' measurement database which is representative of the quantity, quality, and distribution of the average set of measurements (Sec. 2.2). From this, the reference Jacobian J can be constructed (Sec. 2.3) and K_{WFE} can be assembled (Sec. 2.4). Next, $D(z)$ must be calculated and K_{WFE} must be adapted to be compatible with the discrete-time system shown in Fig. 5 (Sec. 2.5), and K_{interp} must be calculated accordingly (Sec. 2.6).

2.2 Reference measurement database

The considered lidar sensor collects measurements of the relative wind speed by scanning the surface of a cone ahead of the aircraft, as illustrated in Fig. 6. With each laser 'shot', a set of $n_{meas,LOS}$ measurements, starting from R_{min} and spaced evenly along the lidar line-of-sight (LOS) is taken. The range resolution ΔR defines the spacing between individual measurements. The LOS rotates (approximately) around the principal longitudinal axis of the aircraft at the scan rate f_{scan}, and measurements are taken at the Pulse Repetition Frequency (PRF).

Looking at Fig. 6, the LOS direction is determined by two angles: the aperture angle η_{apert} and the scan angle ϕ_{scan}. Assuming that only vertical turbulence w_{turb} is present (i.e. no lateral or longitudinal disturbances), and that the steady-state angle of attack α is negligible (such that $\cos\alpha \approx 1$), the relative wind speed detected by the lidar is:

$$\forall i \in [\![1\ n_{meas}]\!], \quad u_{meas,i} = V_{TAS}\cos\eta_{apert} + w_{turb,i}\sin\eta_{apert}\sin\phi_{scan,i} + \delta_{noise,i} \tag{9}$$

Note that at scan angles of 0 and 180 deg, the vertical wind speed is perpendicular to the LOS and there is no component of the vertical wind speed in the measurement. Indeed, for the purposes of vertical wind estimation, the 'signal-to-noise ratio' of measurements taken around these angles is so small as to make them worthless.

The considered measurement noise $\delta_{noise,i}$ is characterized by a Gaussian distribution centered at 0 (i.e. $E(\delta_{noise}) = 0$) with an expected standard deviation σ_i which depends on atmospheric conditions, lidar technology, and several design parameters [12]. In reality, the noise is not necessarily centered, however it is assumed that the lidar is correctly calibrated and that the potential bias has already been compensated. As only one flight condition is considered in the present work, the atmospheric conditions

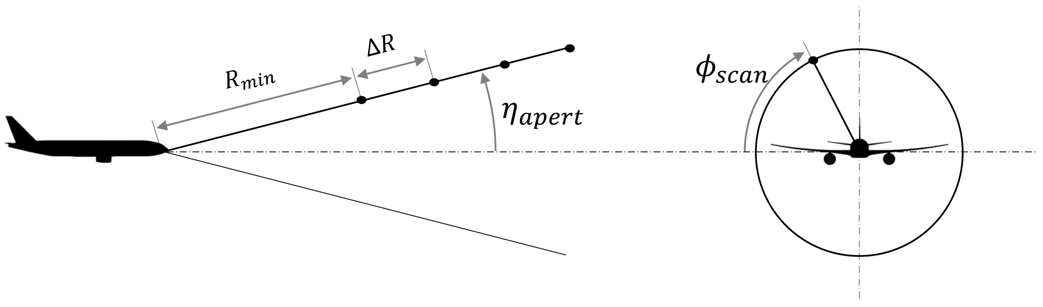


Fig. 6 Lidar system measurement geometry

are constant and the parameters driving the noise are the measurement range R_i, the range resolution ΔR, and the Power Aperture Product PAP:[3]

$$\forall i \in [\![1\ n_{meas}]\!], \quad \sigma_i \propto R_i \cdot \sqrt{\frac{1}{\Delta R}} \cdot \sqrt{\frac{1}{PAP}} \tag{10}$$

PAP and ΔR are generally fixed for a given lidar configuration, whereas R_i increases along the LOS. Each measurement point along the LOS therefore has a fixed σ_i.

As they are collected, measurements are stored inside a database along with a set of metadata, i.e. their LOS directions, expected standard deviations σ_i, and Earth-relative positions. All measurements whose positions fall within the estimation window can then be extracted from the database and used for estimation. For the purposes of this paper, the actual measured values are not needed: Eq. 8 shows that $\overrightarrow{w}_{turb,est}$ provides the values of the vertical wind. As will be demonstrated in the following sections, only the position (i.e. x-coordinate relative to the aircraft), scan angle $\phi_{scan,i}$, aperture angle η_{apert}, and expected standard deviation σ_i of each measurement are needed.

Accordingly, given the parameters of the system, a measurement database containing only the aforementioned elements of the measurement metadata can be built up. Sets of LOS measurements are evenly spaced (along $\overrightarrow{x_k}$) with V_{TAS}/PRF, so the x-coordinates can be easily calculated. As mentioned already, spanwise and vertical variations in the wind field are neglected, so the y- and z-coordinates are not needed. Only the measurement points relevant to the estimation are needed, so those which fall outside the estimation window are ignored. Each measurement has the same aperture angle, and the expected standard deviation can be calculated from the LOS range of the measurement R_i.

Assigning ϕ_{scan} values to the measurement points is more delicate: as discussed above and in Eq. 9, the vertical wind component of a measurement is proportional to $\sin\phi_{scan}$, so the distribution of ϕ_{scan} among the measurements can locally affect the quality of the estimate. For example, a local concentration of measurements taken near $\phi_{scan} = 0$ or 180 deg would result in a poorer estimate in its vicinity. In the real system, the 'leading' scan angle of each estimation window (i.e. the one corresponding to the current LOS) changes with each estimation unless the scan rate and PRF are both integer multiples of the estimation rate, and if the scan rate is too slow or the estimation window too small, the 'average' value of $\sin\phi_{scan}$ may vary significantly between windows.

Like K_{WFE}, the value of ϕ_{scan} attributed to each measurement point in the reference measurement database must remain constant across estimation windows. The goal is for the filter to be representative of the average estimation quality, so instead of arbitrarily choosing a 'leading' scan angle and extrapolating ϕ_{scan} for all measurement points based on the scan rate, here a single 'expected' value of $\sin\phi_{scan}$ is

[3]The Power Aperture Product is the product of the lidar's transmitted power and the area of the receiving telescope's objective, and serves as a general indicator of the sensor's performance and sizing.

assigned to all measurements. Seeing as the wind estimation algorithm is based on a least-squares optimization (Eq. 1), taking the root mean square of $\sin\phi_{scan}$ as the 'expected' value provides an excellent approximation:

$$\forall i \in [\![1\ n_{meas}]\!], \quad \sin\phi_{scan,i} = E(\sin\phi_{scan}) = (\sin\phi_{scan})_{RMS} = \frac{1}{\sqrt{2}} \tag{11}$$

Figure 7 shows an example of the positions and standard deviations contained in a reference measurement database built up in this way. In this example, a laser PRF of 500 Hz is used. Each pulse provides measurements at $n_{meas,LOS} = 9$ uniformly spaced points ranging from 60 m to 180 m along the laser's LOS. The expected standard deviation σ_i for each measurement (i.e. the expected standard deviation of $\delta_{noise,i}$ from Eq. 9) is plotted against its relative position along the flight path, and is represented by a blue dot. Each new laser pulse adds new measurements on the red line at the measurement positions (red dots; the standard deviations increase linearly with the distance, cf. Eq. 10). Older sets of measurements are separated by $\Delta x = V_{TAS}/PRF = 0.52$ m. Due to this small Δx, the old measurements (blue dots) appear as solid lines. The dashed lines indicate the bounds of the estimation window, which in this case is defined by $\tau_{lag} = 0.3$ s and $\tau_{lead} = 0.55$ s, and which at an airspeed of 260 m/s results in $x_{lag} = -78$ m and $x_{lead} = 143$ m. In this case, the real database would be filled completely within about one second. It is the time that the forwardmost measurement at $x = 180$ m takes to reach the rear end of the estimation window at $x = -78$ m, i.e. 258 m / 260 m/s.

Figure 8 shows the same information as Fig. 7 in the form of a histogram whose bins are defined as the space between adjacent nodes of a wind field with $n_{nodes} = 33$. The value of σ_i is denoted here by color: dark blue indicates the best measurements (i.e. with the lowest σ_i) and yellow the worst. Note how the density of measurements at a given position is lowest at $x = x_{lead} \approx 145$ m, increases until the aftmost LOS measurement point at $x \approx 55$ m, and then remains approximately constant until the end of the estimation window. Furthermore, note the significantly lower average quality of measurements in the bins further ahead of the aircraft. Both effects combined result in a significantly poorer quality of the estimated wind far ahead of the aircraft, see e.g. the non-regularized estimate in Fig. 2.

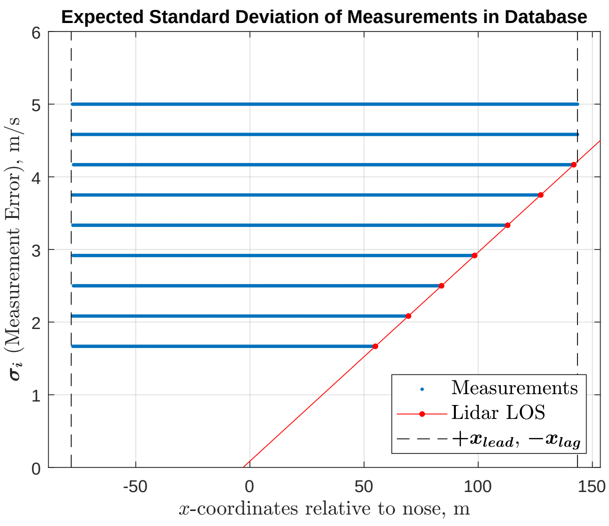


Fig. 7 Expected measurement error of measurements contained in the reference measurement database plotted against its positions.

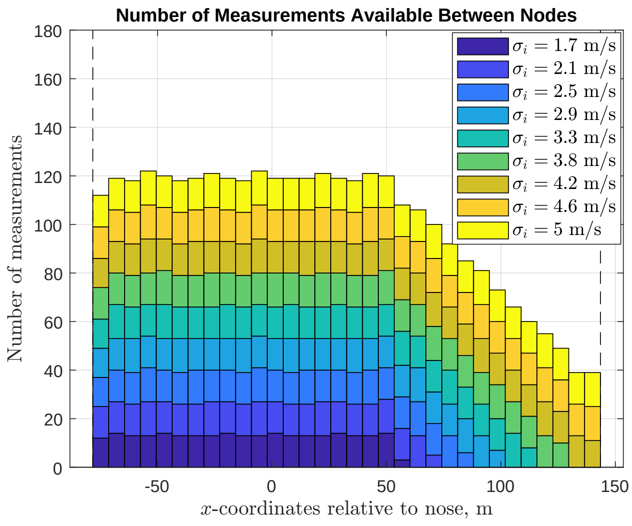


Fig. 8 Histogram of measurements in the reference database separated according to their standard deviations. Each bin occupies the space between adjacent nodes in a wind field with 33 nodes.

2.3 Construction of the reference Jacobian

The reference Jacobian can now be constructed using the reference measurement database. The first step is to identify $y_i(\theta)$ from Eq. 4, as it is the only value in $r_i(\theta)$ which depends on θ. Each measurement $u_{meas,i}$ is taken in the lidar sensor's LOS. V_{TAS} is measured separately, so its contribution can be removed from $u_{meas,i}$, leaving z_i as the projection of the vertical wind speed on the LOS:

$$\forall i \in [\![1\ n_{meas}]\!], \quad z_i = u_{meas,i} - V_{TAS}\cos\eta_{apert} = w_{turb,i}\ \sin\eta_{apert}\ \sin\phi_{scan,i} + \delta_{noise,i} \tag{12}$$

$y_i(\theta)$ is essentially the linear interpolation of the θ values of the two adjacent nodes, as illustrated in Fig. 9. To remain consistent with z_i, it must also include the trigonometric term $\sin\eta_{apert}\sin\phi_{scan,i}$. This gives:

$$\forall i \in [\![1\ n_{meas}]\!], \quad y_i(\theta) = \left[\frac{\Delta x_{b,i}}{\Delta x_{a,i}+\Delta x_{b,i}}\theta_{p,i} + \frac{\Delta x_{a,i}}{\Delta x_{a,i}+\Delta x_{b,i}}\theta_{p+1,i}\right]\sin\eta_{apert}\ \sin\phi_{scan,i} \tag{13}$$

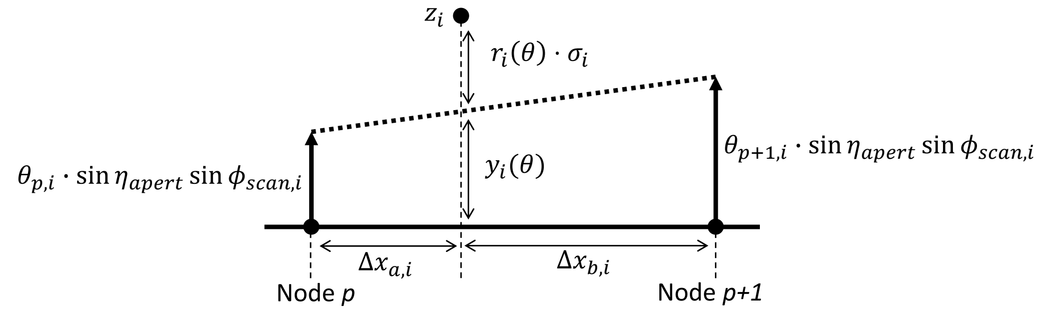

z_i
$r_i(\theta)\cdot\sigma_i$
$\theta_{p+1,i}\cdot\sin\eta_{apert}\sin\phi_{scan,i}$
$\theta_{p,i}\cdot\sin\eta_{apert}\sin\phi_{scan,i}$
$y_i(\theta)$
$\Delta x_{a,i}$
$\Delta x_{b,i}$
Node *p*
Node *p+1*

Fig. 9 Geometrical relation between z_i, y_i, and θ.

noting that here nodes p and $p+1$ are the nodes adjacent (along flight path direction $\overrightarrow{x_k}$) to measurement i, and $\Delta x_{a,i}$, $\Delta x_{b,i}$ are the distances along $\overrightarrow{x_k}$ between measurement i and nodes p and $p+1$, respectively. They can both be easily calculated from the set of x-coordinates (from the metadata associated to each measurement) in the reference measurement database. Taking the partial derivative as in Eq. 5:

$$\forall i \in [\![1\ n_{meas}]\!], \quad \frac{\partial r_i(\theta)}{\partial\theta_{p,i}} = \frac{\sin\eta_{apert}\ \sin\phi_{scan,i}}{\sigma_i}\frac{\Delta x_{b,i}}{\Delta x_{a,i}+\Delta x_{b,i}}, \quad \frac{\partial r_i(\theta)}{\partial\theta_{p+1,i}} = \frac{\sin\eta_{apert}\ \sin\phi_{scan,i}}{\sigma_i}\frac{\Delta x_{a,i}}{\Delta x_{a,i}+\Delta x_{b,i}} \tag{14}$$

The Jacobian J therefore has as many rows as there are measurements in the database (n_{meas}) and as many columns as there are nodes (n_{nodes}). Each row J_i contains only zeros, except for columns p and $p+1$:

$$\forall i \in [\![1\ n_{meas}]\!], \quad J_i = \left[0,\ \cdots\ 0,\ \frac{\partial r_i(\theta)}{\partial\theta_{p,i}},\ \frac{\partial r_i(\theta)}{\partial\theta_{p+1,i}},\ 0,\ \cdots\ 0\right] \tag{15}$$

Note that due to the fact that expected values are used for $\sin\phi_{scan}$ (since Eq. 11), there are no terms in the reference Jacobian less than 0.

2.4 Construction of K_{WFE}

The product from Eq. 8 can be divided into two parts:

$$\left(\widetilde{J}^T\widetilde{J}\right)^{-1} J^T r = K_{WFE}\overrightarrow{w}_{turb,est} = K_{inv}\ K_{weight}\ \overrightarrow{w}_{turb,est} \tag{16}$$

$K_{inv} = \left(\widetilde{J}^T\widetilde{J}\right)^{-1}$ is trivial to obtain once the reference Jacobian has been calculated.

The 'weighting gain' K_{weight} is somewhat more difficult. The product $J^T r$ results in a vector with n_{nodes} elements. Conceptually speaking, this product could be thought of as a kind of weighted sum of the measurements at each node, with the weights determined by the distance of the measurements from their adjacent nodes, their standard deviations, and their scan angle. The regularization terms are constant across the wind field, so their effect at more highly weighted nodes is reduced relative to the measurements. In other words, higher values indicate that more is known about the wind at that location, so the estimate has greater confidence in the measured values.

If the measurement noise δ_{noise} is neglected, and recalling that $y_i(\theta^{[0]}) = 0$, each residual term is:

$$\forall i \in [\![1\ n_{meas}]\!], \quad r_i = w_{turb,i}\frac{\sin\eta_{apert}\ \sin\phi_{scan,i}}{\sigma_i} \tag{17}$$

If w_{turb} is then assumed to only vary linearly between estimation nodes, not unlike the schematic in Fig. 9, $w_{turb,i}$ can be expressed as the linear interpolation of the wind speed at the adjacent nodes:

$$\forall i \in [\![1\ n_{meas}]\!], \quad w_{turb,i} = \frac{\Delta x_{b,i}}{\Delta x_{a,i}+\Delta x_{b,i}}w_{turb,p,i} + \frac{\Delta x_{a,i}}{\Delta x_{a,i}+\Delta x_{b,i}}w_{turb,p+1,i} \tag{18}$$

where $w_{turb,p,i}$ is the true vertical wind speed at the position of node p. Combining Eqs. 17 and 18 and recalling section 2.3, it becomes clear that $K_{weight} = J^T J$.

To give an indication of how the weighting is distributed across the nodes, an example of the values along the main diagonal of K_{weight} is shown in Fig. 10. Comparing it to Figs. 7 and 8, note how the value of K_{weight} increases as more and better measurements are available at each node. Similarly, because the minimum measurement range corresponds to node 19, no new measurements are added between nodes 1 and 18, so K_{weight} remains constant over those nodes. The value at node 1 is substantially lower than neighboring nodes due to the fact that while most nodes are weighted by measurements from both adjacent bins (i.e. one forward and one aft of the node), the nodes at the boundary of the estimation window (in this case nodes 1 and 33) only have access to one. This effect can be traced directly to the construction of the Jacobian in Eqs. 14 and 15. Node 1 only uses the measurements from the leftmost bin in Fig. 8, which means that it has slightly less than half as many available measurements as node 2.

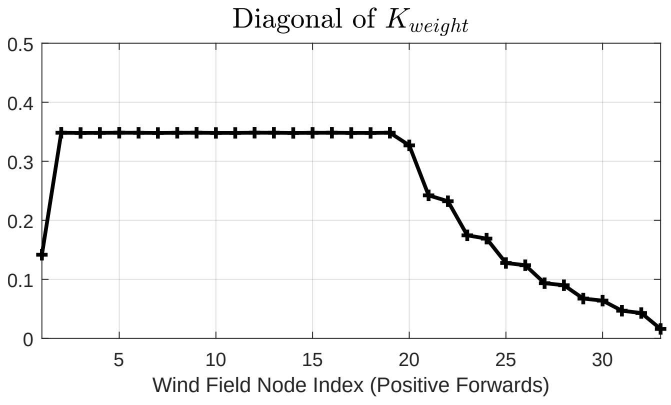


Fig. 10 Diagonal elements of weighting gain K_{weight}

2.5 Discrete-time implementation

$K_{WFE} = K_{inv}K_{weight}$ is simply a static gain matrix. As mentioned in the introduction (see Fig. 5), it must be coupled with a tapped delay line $D(z)$ which takes as input the true wind corresponding to the foremost node $\theta_{n_{nodes}}$, i.e. at $x = x_{lead}$ (see Fig. 1), and outputs the true vertical wind across the estimation window $\overrightarrow{w}_{turb,est}$. Note that this assumes that the wind field does not change over the time it takes the aircraft to traverse it. However, considering the speed of the aircraft and the time scales at which atmospheric phenomena change, this is not a particularly strong assumption.

The configuration of $D(z)$ depends mainly on the required discrete sampling time. K_{WFE}, as derived in the sections above, is based on the original parameter vector θ, which has n_{nodes} elements. An initial choice might be to keep K_{WFE} as it is and to define $\overrightarrow{w}_{turb,est}$ in accordance with the estimated wind field node positions. The sample time must then be:

$$\Delta t_{\theta} = \frac{\tau_{lag} + \tau_{lead}}{n_{nodes} - 1} \tag{19}$$

This is referred to hereafter as the *estimation-native* sampling time. This choice has the advantage of being simple and relatively low-order, as the filter only has n_{nodes} states. Unfortunately, Δt_{θ} is typically not equal to the controller sampling time Δt_{ctrl}, so it can't be applied directly to control synthesis. One solution is to then resample the entire filter to the controller's sampling time,[4] denoted hereafter as the *resampled estimation-native* implementation.

The alternative is to use the *controller-native* sampling time. This involves essentially replacing the original wind field nodes with 'new' nodes spaced with $\Delta x_{ctrl} = \Delta t_{ctrl} \cdot V_{TAS}$. The number of nodes is then:

$$n_{nodes,c\text{-}n} = \frac{\tau_{lag} + \tau_{lead}}{\Delta t_{ctrl}} + 1 \tag{20}$$

$D(z)$ must be recalculated with sampling time Δt_{ctrl} and $(n_{nodes,c\text{-}n} - 1)$ delays, and $\overrightarrow{w}_{turb,est}$ is consequently redefined as having $n_{nodes,c\text{-}n}$ elements. The new nodal coordinates are then used as the basis for a 2-D linear interpolation of K_{WFE}. The sum of the elements of each row of K_{WFE} is always 1 (equivalent to saying the filter has a static gain of 1), so the newly interpolated matrix must be renormalized by dividing each element by the sum of its row. Typically, $\Delta t_{\theta} > \Delta t_{ctrl}$, so this will result in an increase in resolution at the cost of a significant increase in states.

The practical differences between these methods are examined below, in Section 3.3.

2.6 Controller wind field interpolation

As already mentioned in the introduction, the controller has neither the same sampling rate as the wind field estimation nor the same wind field node coordinates. In fact, the controller usually has a much faster sampling rate and more closely spaced wind field nodes, although it could also use sparsely or irregularly spaced nodes. To produce the necessary controller wind field $\overrightarrow{w}_{ctrl}$ at the correct rate, the estimated wind field must be resampled.

In [12], for example, this resampling is done by linearly reinterpolating the estimated wind at the controller sampling rate. For example, the 'real' wind field estimation algorithm usually runs at 10 Hz and a typical controller runs at 100 Hz, so the controller wind field will be resampled from the estimated wind field 10 times for each estimate produced by the algorithm. The controller wind field coordinates take the aircraft's motion into account, so they 'slide' forward along the estimated wind field between

[4]The example shown in this paper was resampled using the MATLAB function *d2d* with Tustin's method.

estimation steps. For the nodes positioned in the region ahead of the aircraft, where the maximum measurement density is not reached ($x > 50$ m in Fig. 8), the wind estimate becomes progressively worse with each resampling due to the poorer quality of the estimate further ahead.

This can be reproduced in the filter by introducing an interpolation gain matrix K_{interp} which linearly interpolates the estimated wind speed at any given point within the wind field. This is done by applying the expression from Eq. 18 and building up the matrix in a similar way to the Jacobian in Eq. 15. In the resulting matrix, each column would therefore correspond to an estimated wind field node and each row to a controller wind field node. Note, however, that the relative coordinates of the controller wind field cannot change over time, so the effect of the 'sliding' mentioned above cannot be reproduced.

2.7 Simplifications and limitations

Several assumptions have been made above to obtain the desired results, so it is important to discuss them and their implications for the filter's accuracy.

First and foremost, this model does not include the effect of the real measurement noise on the estimate. The noise δ_{noise} is assumed to have a centered probability density function, i.e. $E(\delta_{noise}) = 0$, so it does not introduce a bias into the estimate. Under this assumption, it is dropped starting in Eq. 17. To be clear, the standard deviation σ_i found, for example, in Eq. 4, represents the *expected* measurement noise, and is used to correctly weight the measurements according to the degree of confidence in their values. Indeed, as stated in Section 1, the purpose of this work is mainly to model the smoothing effect and the related loss of information in the estimated wind field. This loss of information is mainly driven by the number of measurements and noise standard deviation σ_i. The impact of a non-centered noise $E(\delta_{noise}) \neq 0$ would be limited and could be compensated through means not discussed in the present paper. Therefore, the noise considered here is assumed to follow centered Gaussian distributions and with standard deviation values that depend on the range, range resolution, and *PAP*. It should be noted that the respective values of the standard deviation are sufficient for the proposed method: no stochastic evaluation e.g. based on random number generators is required to determine the average information loss induced by the wind estimation.

Secondly, the scan angle approximation in Eq. 11 may result in significant differences when compared to the actual estimate, especially far ahead of the aircraft, where fewer measurements are available. For example, if the most recent measurements (i.e. those furthest ahead of the aircraft) are made with ϕ_{scan} near 0 or 180 deg, $\sin\phi_{scan}$ is smaller, so the linear filter will be somewhat more accurate than the estimate. Similarly, if the scan rate is too slow, such that large parts of the measurements in the buffer have $|\sin\phi_{scan}| < 1/\sqrt{2}$, the estimate will be less accurate than the filter, and vice versa.

The linear interpolation introduced in Eq. 18 effectively adds an additional low-pass filtering effect. The true wind field is not restricted to varying linearly between nodes, so unless it is a predominantly low-frequency signal, some information will be lost. In fact, one can expect to see the greatest difference for turbulence fields with strong high-frequency components. A higher native sampling rate, such as that used by the *controller-native* implementation, could help reduce this effect.

The assumption that the rotational axis of the lidar sensor is aligned with the aircraft's flight path, i.e. that $\cos\alpha \approx 1$, may cause the linear filter to produce different results than the actual estimator in some situations. The real wind estimation system saves the position and orientation of measurements in absolute spatial coordinates, so it can partially compensate for the effects of aircraft motion and orientation. The effects this may have on the quality of the estimated wind field have, however, not yet been quantified.

Finally, deterioration in the controller wind field due to resampling is not modeled in the linear filter. As mentioned in Section 2.6, this is caused by the controller wind field node coordinates moving relative to the estimated wind field nodes between estimation steps. Depending on how the controller wind field coordinates are chosen, the effect on the linear filter can change: if the 'nominal' coordinates are used, i.e. those corresponding to the first instant of the current estimation step, the filter will be overly optimistic. To be conservative, the controller wind field nodes could be chosen further forward so as to always use the worst case, i.e. immediately before the following estimation step. For the rest of this paper, only the nominal controller node positions are considered.

3 Results and evaluation of the linear filter

The default lidar settings used in the examples below are defined in Table 1. Note that σ_i/R_i is a function of several other parameters (see Eq. 10), however it also depends on other parts of the surrogate model presented in [12]. Its calculation falls beyond the scope of this paper, so this may be used as a reference value in combination with Eq. 10.

Table 1 Default lidar system parameters

Parameter	η_{apert}	f_{scan}	PRF	PAP	R_{min}	ΔR	$n_{meas,LOS}$
Value	15°	13 Hz	500 Hz	0.05 Wm^2	60 m	15 m	9
Parameter	n_{nodes}	τ_{lead}	τ_{lag}	γ_1	γ_2	V_{TAS}	σ_i/R_i
Value	33	0.55 s	0.3 s	3.2	5.5	260 m/s	0.0278 s^{-1}

3.1 Evaluation of Estimation-Native filter

The lidar system described in Table 1 results in an *estimation-native* (see 2.5) sample time $\Delta t_\theta = 0.0266\,\mathrm{s}$, implying a sampling frequency $f_s \approx 37.6\,\mathrm{Hz}$. The resulting discrete filter is of order 32. To make it easier to compare it with the original wind field estimation, K_{interp} is chosen such that $\vec{w}_{filt,est} = \vec{w}_{filt,ctrl}$, i.e. the outputs correspond to the estimated wind field nodes $\boldsymbol{\theta}$. Figure 11 shows the Bode magnitude plots of the transfer paths from $w_{turb,lead}$ to nodes 1, 5, 11, 22, and 33, and Fig. 12 shows the values of the corresponding row in K_{WFE}.

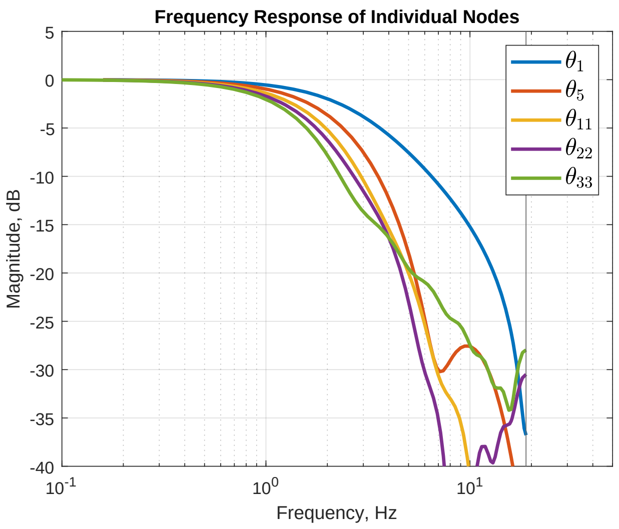


Fig. 11 Bode magnitude plot of a selection of nodes for the *estimation-native* filter

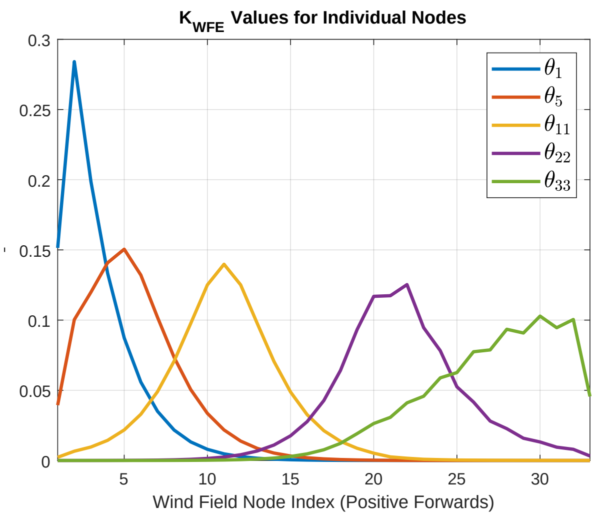
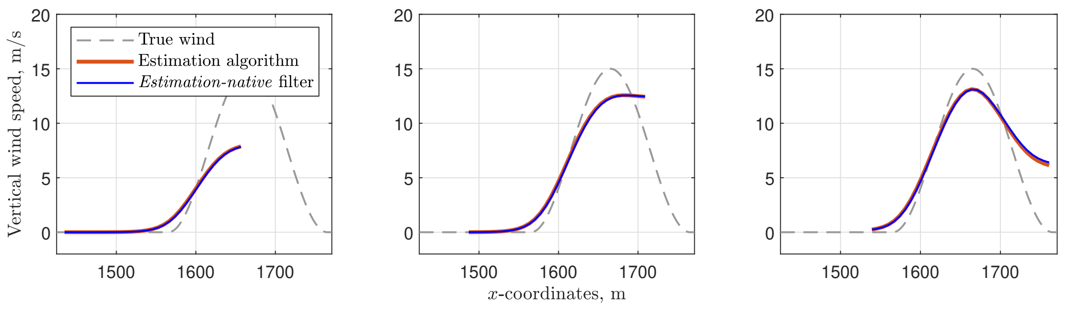


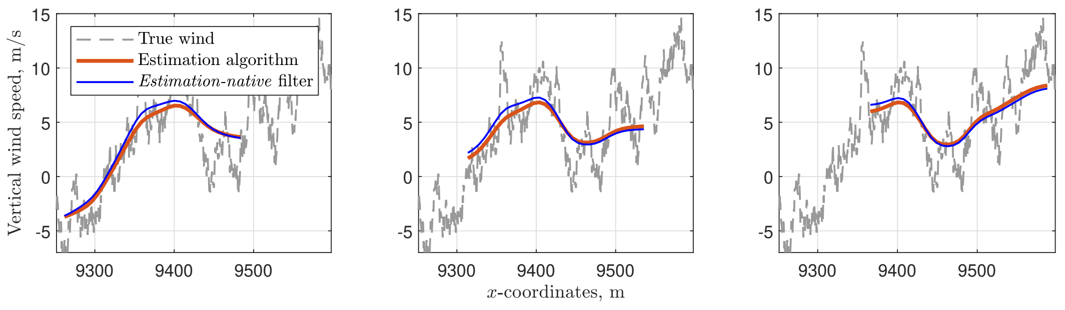
Fig. 12 Filter gains from K_{WFE} for a selection of nodes

It is clear from both figures that the further forward a node is positioned, the stronger the smoothing effect is, as evidenced by the lower cut-off frequencies in Fig. 11 and the wider, more gently sloped curves in Fig. 12. Due to their positions at the edges of the estimation window, θ_1 and θ_{33} are somewhat more sensitive at higher frequencies than their neighboring nodes. Comparing both figures to Figs. 7, 8, and 10, the influence of the density of available measurements and their standard deviations on the filtering at each node is clearly visible.

Figure 13 compares the actual estimated wind field (with no measurement error) with the output of the linear filter for both a discrete gust and continuous turbulence. In each row, each window shows the estimated wind field at a given point in time; as the aircraft moves forward, the estimate proceeds along the true wind field. In Fig. 13a, the match is nearly perfect, with the greatest error appearing near the leading edge. Fig. 13b contains more significant errors, particularly around x = 9350 - 9400 m. Given the presence of strong high-frequency content in that area, it is likely due to interpolation approximation discussed in Sec. 2.7.



(a) Discrete gust with H = 100 m



(b) Continuous turbulence (von Kármán spectrum)

Fig. 13 Comparison of estimated wind field $\vec{w}_{est}$ with the estimation-native filtered wind field $\vec{w}_{filt,est}$ over 3 sequential estimation windows.

3.2 Power spectral analysis

Another way to compare the filter with the estimation algorithm is to perform a power-spectral analysis similar to that presented in [13]. In essence, a fairly long (>1000 s) continuous turbulence wind field is simulated. The power spectral density (PSD) of the error between the estimated wind speed and the true wind at a particular point in the estimated wind field $\Phi_{\Delta w}$ is calculated and then normalized using the PSD of the true wind Φ_{turb}. The resulting curve $\Phi_{\Delta w}/\Phi_{turb}$ gives an indication of the relative precision of the wind field estimation across the frequency spectrum. A value near 0 indicates that the wind is estimated nearly perfectly at those frequencies, whereas a value near 1 indicates that it is not tracked at all. Accordingly, the closer the curve comes to 1, the worse the performance of the estimation.

Figure 14 shows an analysis of this type evaluated at the aircraft nose. The curve representing the output of the estimation algorithm is calculated through simulation as described. The curve for the linear filter is instead derived directly from the transfer function of the filter, i.e. by augmenting the filter with an output Δw_{filt} which subtracts the filtered wind speed at the nose $w_{filt,0}$ from the true wind speed at the nose $w_{turb,0}$, taking the magnitude of this transfer function $w_{turb,lead} \rightarrow \Delta w_{filt}$, and squaring it to find its power spectral density. It is clear here that the filter matches the true estimate quite well up to approximately 3 Hz, but between 3 and 8 Hz the estimation algorithm performs visibly worse. The reasons for this discrepancy are currently unclear, however the frequencies relevant for gust load alleviation are generally below 4 Hz, so it is still a suitable approximation for this application.

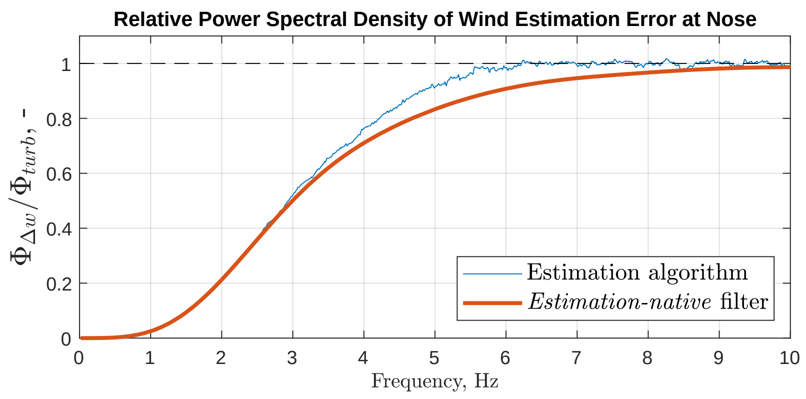


Fig. 14 Comparison of relative PSD of wind estimation error at the aircraft nose between the estimation algorithm and the *estimation-native* filter.

3.3 Comparison of discretization methods

Seeing as the *estimation-native* filter described above is not necessarily compatible with the controller's sample time, Section 2.5 defines two ways to convert it to the controller's sample time: the *resampled estimation-native* and the *controller-native* filters. Here, the controller sample time is taken as 0.01 s, i.e. 100 Hz, which results in 85 delay states for the *controller-native* filter. To allow for comparison with the *estimation-native* filter, all filters have as output the original estimated wind field nodes (instead of the controller's wind field nodes).

To provide a quantitative measure of their performance against the estimation algorithm, the mean ε_{WF} of the root mean square error can be defined, in which the root mean square error between the filtered values and the estimated values of a given estimation window is averaged over an $n_{windows}$ set of estimation windows. These samples are chosen 0.1 s apart to take into account the normal estimation rate of 10 Hz. To avoid mistakenly taking the frames before and after the actual gust encounter into account, they include only frames in which at least one node of either of the two wind fields is nonzero.

$$\varepsilon_{WF} = \frac{\sum_{s=1}^{n_{windows}} \sqrt{\frac{\sum_{j=1}^{n_{nodes}} \left(w_{filt,est,j}^{[s]} - w_{est,j}^{[s]}\right)^2}{n_{nodes}}}}{n_{windows}} \tag{21}$$

Table 2 Comparison of ε_{WF} among available discretization methods

Test case	Estimation-Native	Resampled E-N	Controller-Native
10 m Discrete Gust	0.041 m/s	0.107 m/s	0.021 m/s
30 m Discrete Gust	0.061 m/s	0.145 m/s	0.043 m/s
60 m Discrete Gust	0.075 m/s	0.069 m/s	0.058 m/s
100 m Discrete Gust	0.066 m/s	0.035 m/s	0.047 m/s
Continuous Turbulence	0.272 m/s	0.208 m/s	0.107 m/s

Table 2 compares the mean-RMS error of the 3 filter implementations for a set of turbulence cases. In all but one of the listed cases (100 m discrete gust), the *controller-native* filter has the best performance and the *resampled* filter has the worst of the three.

4 Effect on control synthesis

To demonstrate the effect of the filter developed above on control design, this section relies on the means and methods of the GLA benchmark presented in [12].[5] Firstly, the *default* controller from the benchmark is used to show that the filter is representative of the real system for the purposes of control design. Secondly, a simple preview controller is synthesized using the new synthesis problem along with precisely the same method and requirements as the default controller. It is then compared with the default controller and evaluated against the benchmark problem.

Figure 15 shows time-domain responses to a set of discrete gusts with H = 30, 70, and 80 m, including the incremental wing root bending moment (row 1), the elevator deflection (row 2), and the incremental HTP-root bending moment (row 3) of the aircraft using the *default* controller. Three curves are shown: the ‘hybrid’ simulation including the full nonlinear model of the lidar and wind estimation system (Fig. 3), the linear model without the filter (Fig. 4), and the linear model including the *controller-native* filter (Fig. 5). It is obvious that the model including the filter matches the hybrid simulation much more closely than the one without. For all three gust lengths, without the filter, the elevator reaches greater deflections, leading to an improved reduction in wing root bending moment and increased HTP loads. The very small differences between the linear model including the filter and the hybrid simulation are possibly caused by the simplifications discussed in Sec. 2.7. Altogether, this indicates that the linear system including the filter is much more representative of the real system for the purposes of control synthesis and evaluation.

Next, the control synthesis example is described. Recall that the *default* control synthesis problem was shown in Fig. 4, whereas the *new* control synthesis presented here uses the setup from Fig. 5. As a rule, the linear filter uses the *controller-native* discrete implementation. To summarize briefly, the controller wind field $\overrightarrow{w}_{ctrl}$ has 66 nodes, with 40 nodes ahead of the aircraft nose and 25 behind. This covers a section of the flight path starting from the aircraft’s tail to approximately 100 m ahead. The controller itself is a 3×66 gain matrix in which each row calculates commanded deflections for one of three actuator inputs: the elevator, the outer (symmetric) ailerons, or the inner (symmetric) ailerons.

The tuning goals include an H_2 requirement on the wing root bending moment which is weighted using a linear filter derived from the von Kármán continuous-turbulence PSD, roll-off H_∞ requirements for all actuators, and roll-on H_∞ requirements for the inner and outer ailerons. Once synthesized, the controller is scaled down to ensure the maximum commanded actuator deflections and rates do not exceed the actuator limits for any given discrete gust encounter.

The benchmark problem involves a series of simulations using an aeroelastic model of a large commercial aircraft in combination with the full lidar simulation and nonlinear actuators. Responses to the full spectrum of discrete gusts and continuous turbulence as defined in EASA CS-25 [14] are simulated, and the bending moment along the wing as well as several constrained quantities are evaluated. Two scenarios are envisioned: a *nominal* scenario in which all parameters have their nominal values, and a *robust* scenario in which the *PAP*, actuator dynamics, and system time delay are varied. For more information about the benchmark problem, see [12].

Figure 16 compares the results of the benchmark problem evaluation of the peak bending loads along the wing for the default and new controllers. The new controller, shown in Fig. 16b, shows a *nominal*

[5]The publicly available benchmark, including the controllers and filter discussed here, can be accessed at: `https://github.com/dlr-ft-gla/GLA-Benchmark`

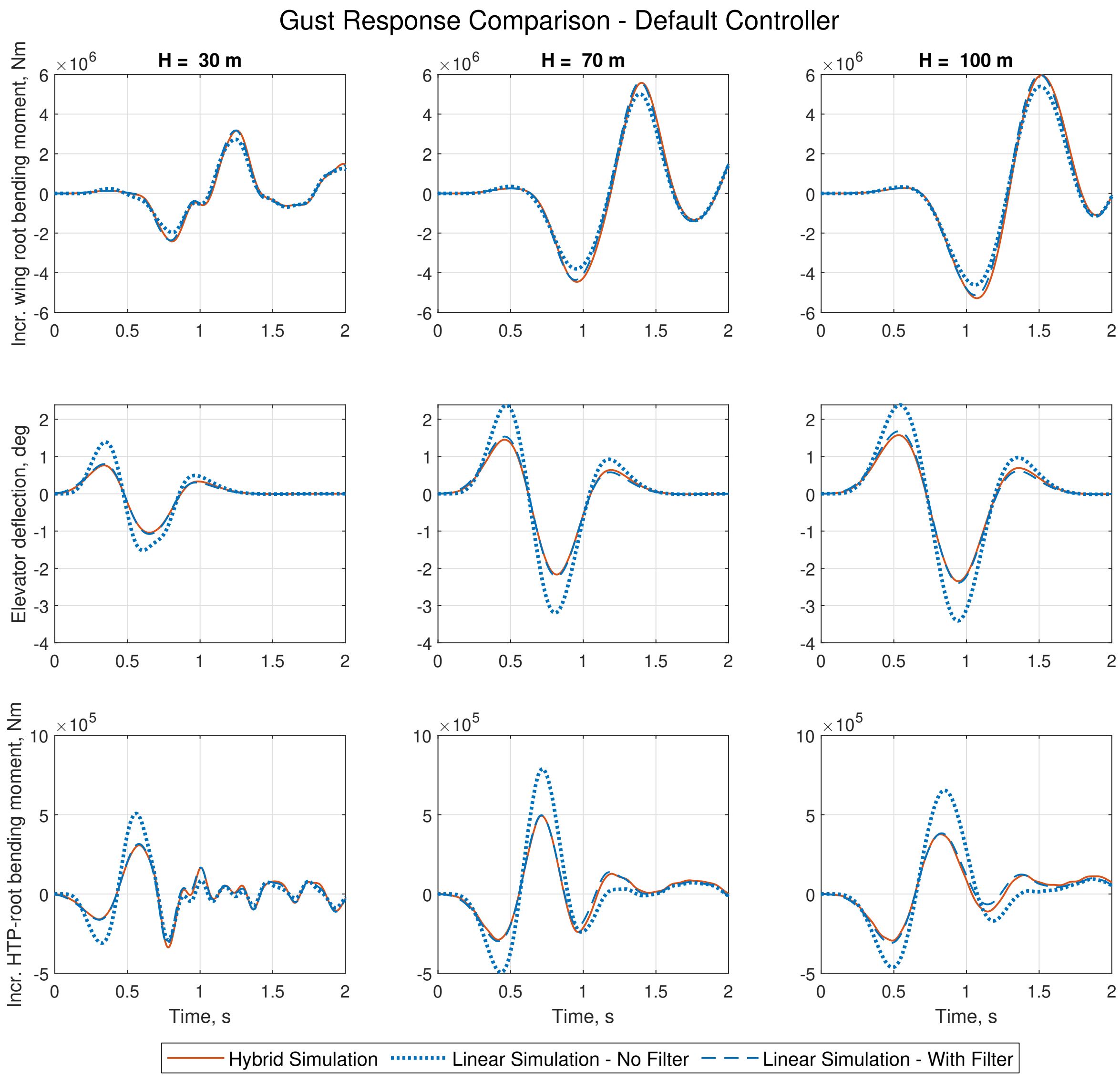
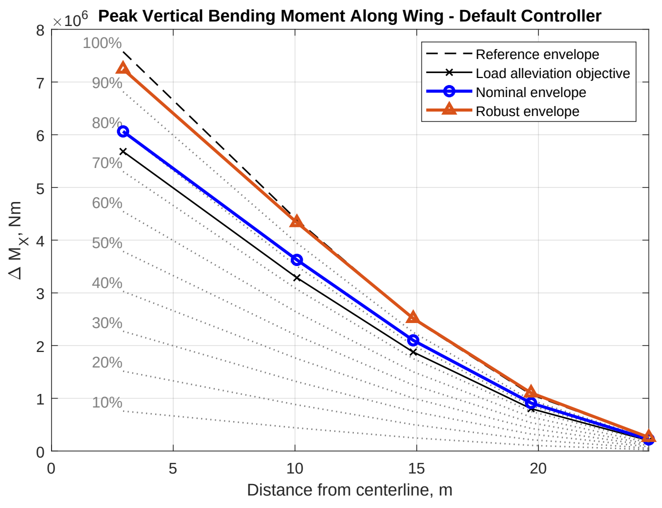


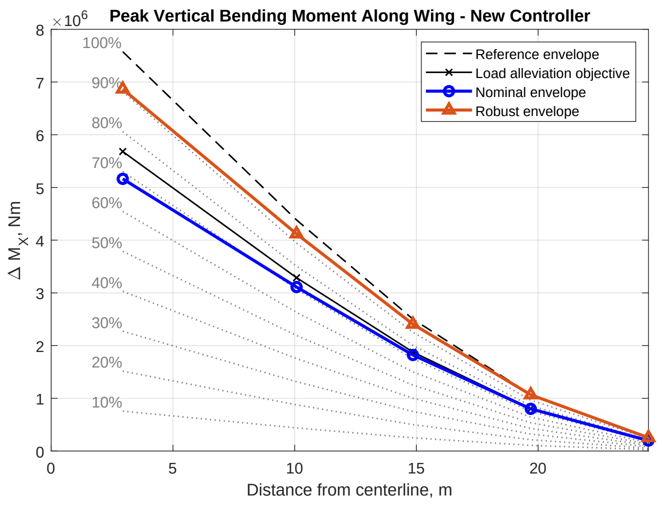
Fig. 15 Simulation of discrete gust responses of the default controller from [12] comparing the full hybrid simulations against linear models with and without the linear filter.

improvement of slightly more than 10 % of the reference bending load envelope compared to the default controller. The *robust* case, however, improves by, at best, less than 6 %. For conciseness, most of the constrained variables have been omitted here, however one in particular is worth mentioning: the HTP-root bending moment. The limit load of $10.45 \cdot 10^5$ Nm was never exceeded by the default controller; the new controller exceeds the old one in both the *nominal* and *robust* scenarios, and violates the limit load in the *robust* scenario ($13.6 \cdot 10^5$ Nm). This suggests that the new controller makes more aggressive use of the elevator.

Figure 17 contains a series of Bode magnitude plots for the transfer paths between the gust input and five key outputs: the wing root bending moment, the HTP-root bending moment, and the deflections of the 3 sets of control surfaces. Each plot, in addition to the linear models including the linear filter (i.e. Fig. 5), shows the model with the default controller but without the linear filter (i.e. Fig. 4, the same model used to tune the default controller).



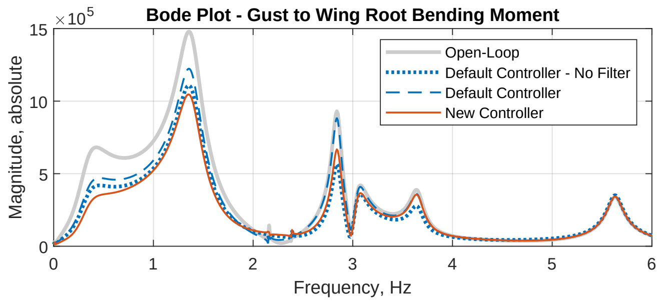
(a) Default controller from [12].



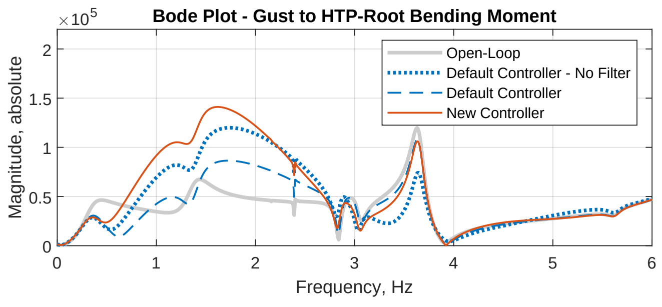
(b) New controller synthesized using linear filter.

Fig. 16 Comparison of wing bending moment envelopes from the benchmark problem in [12]

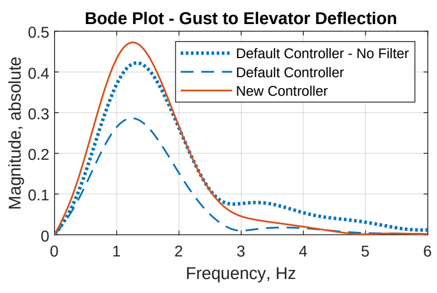
A direct comparison of the default controller with and without the linear filter illustrates the effect of the estimation-induced losses on the controller's effectiveness. As might be expected, having seen Fig. 11, the default controller without the filter is more active with increasing frequency (Figs. 17c, 17d, 17e) and enjoys noticeably improved load alleviation performance up to 3 Hz (Fig. 17a). Due to this increased activity, HTP loads are increased below 3 Hz, however they are significantly reduced between 3 and 4 Hz (Fig. 17b).



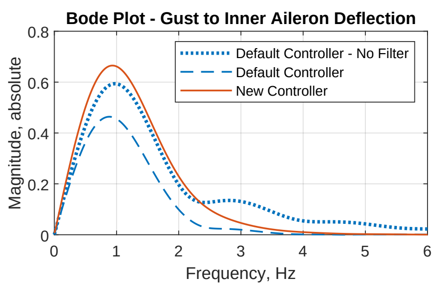
(a) Wing root bending moment



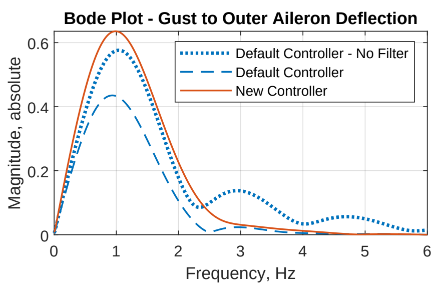
(b) HTP-root bending moment



(c) Elevator deflection



(d) Inner aileron deflection

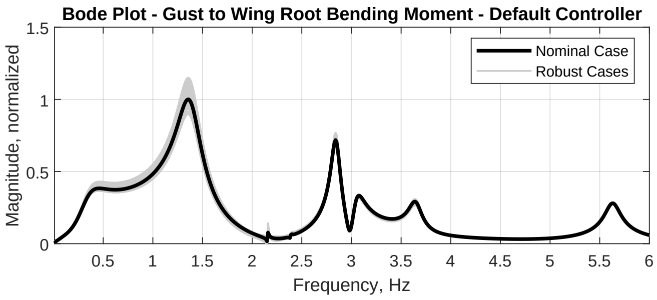


(e) Outer aileron deflection

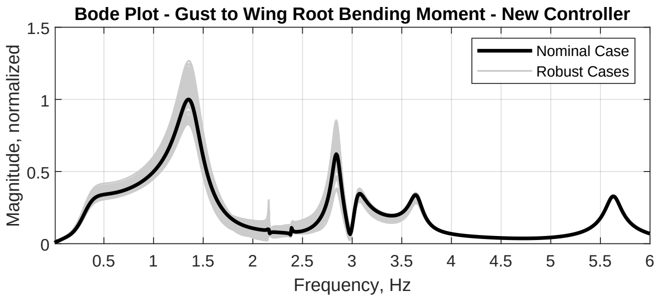
Fig. 17 Comparison of magnitudes of transfer functions from the gust input to key outputs.

Next, the default controller can be compared with the new one. Looking at the loads in Figs. 17a and 17b, it is clear that the new controller is more successful at reducing the wing root bending moment, and that it generates more loads on the HTP. The control surface deflection transfers in Figs. 17c, 17d, and 17e show that below 2 Hz, the new controller mostly resembles a more aggressive version of the default controller, whereas above 2.5 Hz or so it is clearly less aggressive.

The filter can also be used to efficiently analyze the sensitivity of the new controller to the parameter variations in the *robust* scenario using linear techniques. Figures 18 and 19 illustrate the variation of the magnitude of the normalized transfer function between the gust input and the wing root and HTP-root bending moments, respectively, when subject to all the possible parameter variations of the *robust* scenario. The wing root plots in Fig. 18 show that the new controller is overall more sensitive to the parametric uncertainties: the variation in its magnitude for nearly all frequencies between 0.5 Hz and 4 Hz is greater than for the default controller. Its H_∞ norm, i.e. the peak around 1.3 Hz, increases by nearly 30%, compared to less than 15% for the default controller. The HTP-root plots in Fig. 19 reveal that the default controller is more sensitive below 1.5 Hz, the new controller is somewhat more sensitive between 2.5 and 3.5 Hz, and they are roughly similar between 1.5 and 2.5 Hz.

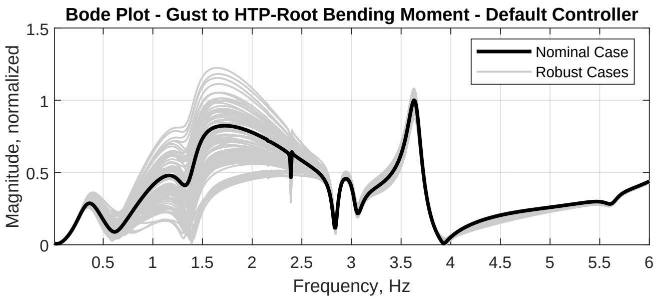


(a) Default controller from [12].

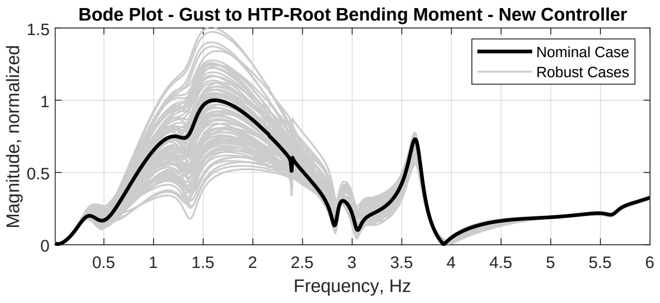


(b) New controller synthesized using linear filter.

Fig. 18 Comparison of variation in magnitude of the transfer function between the gust input and the wing root bending moment for all cases in the *robust* scenario. Magnitudes are normalized with respect to the H_∞ norm of their respective nominal case.



(a) Default controller from [12].



(b) New controller synthesized using linear filter.

Fig. 19 Comparison of variation in magnitude of the transfer function between the gust input and the HTP-root bending moment for all cases in the *robust* scenario. Magnitudes are normalized with respect to the H_∞ norm of their respective nominal case.

5 Conclusion

A method for designing a linear filter which closely approximates the smoothing effects of the wind field estimation algorithm has been presented. This method allows for the direct calculation of the filter based on the lidar system and estimator parameters. The performance of the filter compared to the real estimator has been evaluated, demonstrating an excellent match in several different kinds of turbulence. Furthermore, some of its frequency-domain characteristics have been examined to show the information loss caused by the filter compared to the estimator. Finally, its effect on gust load alleviation control has been demonstrated through an example of a controller synthesized using the linear filter.

Altogether, the filter has been shown to be sufficiently representative of the true lidar-based wind field estimation. The control design problems built using this filter closely match the characteristics of the nonlinear system including the real wind estimation system. Consequently, the behavior and performance of controllers tuned on these design problems should be obtained on the final system as well. The legacy control design problems led to larger differences in behavior which had to be artificially compensated by tweaking the control specifications. This compensation would typically be made manually and iteratively by the designer. The effort this process requires is saved when using the proposed filter. Future work will focus on modeling the uncertainties in the wind field estimate induced by the measurement noise. This could, for instance, be used to tune the controller for improved robustness against noise.

Acknowledgments

The authors would like to acknowledge the funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2163/1 - Sustainable and Energy Efficient Aviation – Project-ID 390881007 as well as the DLR-internal project COLOCAT (Compact Optical sensors for LOad alleviation of Clear Air Turbulence).

References

[1] C. D. Regan and C. V. Jutte. Survey of Applications of Active Control Technology for Gust Alleviation and New Challenges for Lighter-weight Aircraft. Technical Report NASA/TM—2012–216008, NASA, Dryden Flight Research Center, Edwards, California, April 2012.

[2] A. Wildschek, R. Maier, K.-U. Hahn, D. Leissling, M. Press, and A. Zach. Flight Test with an Adaptive Feed-Forward Controller for Alleviation of Turbulence Excited Wing Bending Vibrations. In *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, August 2009. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2009-6118.

[3] A. Galffy, R. Gaggl, R. Mühlbacher, D. Frank, J. Schlarp, and G. Schitter. Turbulence load prediction for manned and unmanned aircraft by means of anticipating differential pressure measurements. *CEAS Aeronautical Journal*, 12(3):535–548, August 2021. DOI: 10.1007/s13272-021-00512-y.

[4] N. P. Schmitt, W. Rehm, T. Pistner, H. Diehl, P. Navé, G. Jenaro-Rabadan, P. Mirand, and M. Reymond. Forward Looking Clear Air Turbulence Measurement with the AWIATOR LIDAR Sensor. pages 179–184, Berlin, 2007.

[5] N. Fezans, J. Schwithal, and D. Fischenberg. In-flight remote sensing and identification of gusts, turbulence, and wake vortices using a Doppler LIDAR. *CEAS Aeronautical Journal*, 8(2):313–333, June 2017. DOI: 10.1007/s13272-017-0240-9.

[6] H.-G. Giesseler, M. Kopf, P. Varutti, T. Faulwasser, and R. Findeisen. Model Predictive Control for Gust Load Alleviation. In *IFAC Proceedings Volumes*, volume 45, pages 27–32, 2012. DOI: 10.3182/20120823-5-NL-3013.00049.

[7] A. Wildschek and R. Maier. Integrated adaptive feed-forward control of atmospheric turbulence excited rigid body motions and structural vibrations on a large transport aircraft. In *2008 American Control Conference*, pages 3281–3286, Seattle, WA, June 2008. IEEE. DOI: 10.1109/ACC.2008.4586998.

[8] H. Fournier, P. Massioni, M. Tu Pham, L. Bako, R. Vernay, and M. Colombo. Robust Gust Load Alleviation of Flexible Aircraft Equipped with Lidar. *Journal of Guidance, Control, and Dynamics*, pages 1–15, September 2021. DOI: 10.2514/1.G006084.

[9] N. Fezans, H.-D. Joos, and C. Deiler. Gust load alleviation for a long-range aircraft with and without anticipation. *CEAS Aeronautical Journal*, 10(4):1033–1057, December 2019. DOI: 10.1007/s13272-019-00362-9.

[10] N. Fezans, P. Vrancken, P. Linsmayer, C. Wallace, and C. Deiler. Designing and Maturating Doppler Lidar Sensors for Gust Load Alleviation: Progress Made Since AWIATOR. Bordeaux, France, 2020.

[11] A. Khalil and N. Fezans. Performance Enhancement of Gust Load Alleviation Systems for Flexible Aircraft using H_∞ Optimal Control with Preview. In *AIAA Scitech 2019 Forum*, San Diego, California, January 2019. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2019-0822.

[12] D. Cavaliere, N. Fezans, D. Kiehn, D. Quero, and P. Vrancken. Gust Load Control Design Challenge Including Lidar Wind Measurements and Based on the Common Research Model. In *AIAA SCITECH 2022 Forum*, San Diego, CA & Virtual, January 2022. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2022-1934.

[13] D. Kiehn, N. Fezans, and P. Vrancken. Frequency-domain performance characterization of lidar-based gust detection systems for load alleviation. In *Deutscher Luft- und Raumfahrtkongress (DLRK) 2021*, Bremen, Germany and virtual, September 2021.

[14] *EASA CS-25: Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes*. European Union Aviation Safety Agency, December 2021.