PEAK-SEEKING CONTROL OF PROPULSION SYSTEMS

A Thesis
Presented to
The Academic Faculty

by

Timothée J.W Cazenave

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2012

PEAK-SEEKING CONTROL OF PROPULSION SYSTEMS

Approved by:

Doctor Eric Feron, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Doctor Manuj Dhingra
School of Aerospace Engineering
Georgia Institute of Technology

Doctor Eric Johnson
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: March 19th 2012

To Angela and my family,

For their love and support

ACKNOWLEDGEMENTS

I would like to thank my advisor Eric Feron for believing in me and introducing me to new wonderful opportunities in life. I would like to thank him for being so generous and always caring for his students.

I would like to thank Manuj Dhingra for taking the time to teach me so much.

I would like to thank Mehrdad Pakmehr for his time and help and making this partnership work very well.

I would like to thank Erwan Salaun for his great advices.

TABLE OF CONTENTS

DEDICATION . . . . . **iii**

ACKNOWLEDGEMENTS . . . . . **iv**

LIST OF FIGURES . . . . . **vii**

I INTRODUCTION **1**

1.1 Literature review 3

II BACKGROUND **4**

2.1 Kalman Filter 4

2.1.1 Linear Kalman Filter 4

2.1.2 Extended Kalman Filter 8

2.2 Linear Quadratic Regulator 11

2.2.1 Derivation 11

2.2.2 LQR robustness 13

III DC MOTOR **17**

3.1 Problem Definition 17

3.2 Peak-Seeking Control 20

3.3 Kalman Filter Design 23

3.4 Simulation Results 28

3.5 Hardware setup 30

IV TURBOPROP ENGINE **34**

4.1 Turbofan model 35

4.2 Control Architecture 39

4.2.1 Control design 41

4.2.2 Gain-scheduling 42

4.2.3 Switching mechanism 45

4.2.4 Control approach considered and simulation results 50

4.3 Peak-seeking 52

4.3.1 Turboprop analysis 54
4.3.2 Study case : Fuel flow optimization 56
4.3.3 Study case : Core Temperature optimization 59
V CONCLUSION AND FUTURE WORK 67
REFERENCES 68

LIST OF FIGURES

1 Common systems were the presented Peak-seeking method could have an impact [29] . . . 3

2 Linear Kalman Filter [25] . . . 8

3 Extended Kalman Filter [25]] . . . 11

4 Representation of GM and PM on a Nyquist diagram [24] . . . 14

5 Robustness properties of LQR controller [24] . . . 16

6 DC Motor with variable pitch propeller schematic . . . 18

7 Resolution of forces acting on a propeller section [4] . . . 18

8 Model Power consumption over pitch angle for an iso-Thrust command 22

9 Change in optimal pitch angle for variations in damping and back emf of the DC motor . . . 22

10 Simulation of the power behavior for changes in Thrust, pitch angle and and rotor speed . . . 24

11 Peak-seeking control structure for DC motor and the variable pitch propeller . . . 24

12 Upper graph: Measurement of noisy δ_p. Lower graph: Hessian estimate 29

13 Peak-seeking over Time . . . 30

14 Power minization over Time . . . 31

15 DC motor with variable pitch system schematic . . . 31

16 DC motor with variable pitch system hardware . . . 32

17 Schematics of turboprop . . . 35

18 JetCAT engine . . . 36

19 CAD model of the Turboprop . . . 37

20 Turboprop engine test bed . . . 37

21 Turboprop engine distributed hardware layout . . . 38

22 Turboprop engine distributed control layout . . . 38

23 Actuators and sensors nodes . . . 39

24 Control strategies of the Turboprop engine . . . 40

25 Gain scheduling procedure 47
26 Gain scheduled control Thrust response 47
27 Input and states history 49
28 Gain scheduled control Thrust response with input rate saturation . . 52
29 Input and states history under input rate saturation 53
30 Fuel flow optimization problem 55
31 Core temperature optimization problem 55
32 Other possible performance functions to minimize 56
33 'Fuel consumption 'Optimal pitch angle for different thrusts 57
34 Fuel flow Peak-seeking 57
35 Blade pitch angle for fuel flow optimization over Time 58
36 Fuel flow minimization over Time 59
37 Actual and measured core temperature change for a constant thrust of 185N 60
38 Peak-seeking structure for noisy performance function 60
39 Gradient estimation for different pitch angle increments 62
40 Gradient estimate through the Peak-seeking process 64
41 Pitch angle for Turboprop Core Temperature optimization over time . 64
42 Turboprop Core Temperature optimization over time 65

CHAPTER I

INTRODUCTION

Propulsion systems, such as turboprop engines, are commonly designed to operate at optimal conditions, which are investigated and set during their design phase. However, effects such as component aging or the operating environment might change these optimal operating conditions. As a result, over a long period of time, valuable resources may be wasted if the propulsion system does not adapt to perform in an optimal manner.

This thesis focuses on the design and implementation of an adaption mechanism that allows the engine to perform optimally with regard to a desired cost function when running at steady state. This adaption mechanism is referred to as Peak-seeking control and is a real time, model-free, steady state optimization process.

Thanks to its model-free approach, the peak-seeking strategy allows modularity in the engine's design. It allows the user to arbitrarily choose any of the engine's components, without having to perform a tedious analysis in order to determine its optimal operating conditions. Also, since the peak-seeking strategy is operating based on steady state measurements, the control law used for the engine is left to the user's choice. Overall, the Peak-seeking strategy designed in this thesis embraces the notion of "Plug and Play".

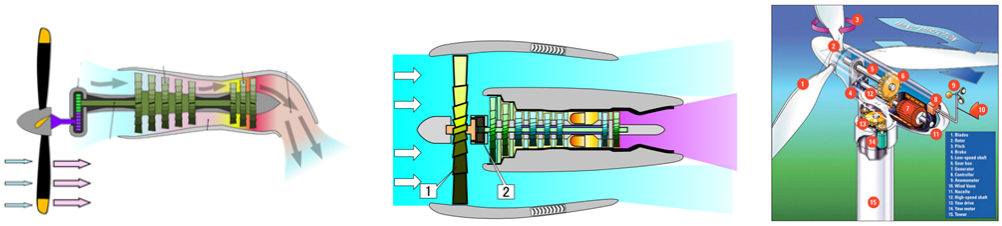
Only one control input (the blade pitch angle for a turboprop) is driven by the Peak-seeking process to progressively minimize a predefined cost function, while the other system's inputs insure stability and performance of the close-loop system. Such technique could also find some applications in other systems such as geared turbofan (with continuously variable transmissions), where the gear ratio would be the optimizing

parameter, or wind turbines, where the blade pitch angle could be optimized for better efficiency. Figure 1 shows the different systems where the Peak-seeking approach developed here could have an impact.

Either a Newton or Gradient descent method are used to drive the optimizing parameter. However, these methods both require the knowledge of the performance function partial derivatives with respect to the optimizing parameter. These derivatives are not known and have to be estimated based only from noisy measurement of the performance function. Using a Taylor's series approximation, a linear model of the derivatives behavior was built and an Extended Kalman Filter (EKF) was used to estimate them.

In order to first evaluate the feasibility of the method, an initial study was performed on a DC motor driving a variable pitch propeller. The DC motor system was simulated, but a hardware platform was built for future implementation of the Peak-seeking. The method was then applied to a simulated turboprop engine, and similarly to the DC motor case, showed some very satisfying results.

This thesis is segmented in five parts. Within this introduction, a literature review is performed. A background part is then presented to explain and derive some of the control techniques used in this research. The next section deals with the DC motor and contains the derivation of the analytical framework of the Peak-seeking control as well as the simulation results and hardware setup of the DC motor system. Next, is a section on the turboprop engine which in addition of presenting the Peak-seeking method and results, also contains a thorough examination of the control design of this nonlinear system. Finally the last part is the conclusion, where we talk about the future work.



(a) Turboprop (b) Geared turbofan (c) Wind turbine

Figure 1: Common systems were the presented Peak-seeking method could have an impact [29]

1.1 Literature review

Peek-seeking control (which is also referred to as extremum seeking or optimalizing control) was first investigaed by C.S. Draper and Y. T. Li [5, 6].

In the past decades, peak-seeking control has been investigated for various applications such as formation flight drag reduction [9, 11, 10], bioreactors [12], antilock braking systems (ABS) [13, 14], aeroengine compressor instabilities [15, 16], combustion instabilities [17, 18, 19], variator in a pushbelt continuous variable transmission (CVT)[26], and other[27, 28]. Most of the methods previously developed are cumbersome to apply on real systems however, as they require complicated excitation signals and highly reliable measurements.

In this thesis, we make use of a technique presented in J. J. Ryan and J. L. Speyer [7] to design the peak-seeking strategy. This method uses a Kalman filter, and is easily implementable.

CHAPTER II

BACKGROUND

We present and derive in this section some of control and estimation tools that will later be used in this report.

2.1 Kalman Filter

2.1.1 Linear Kalman Filter

An Extended Kalman Filter (EKF) is necessary for the peak-seeking process. Before going into more details on the EKF, it is important to understand first how the original Kalman filter works. The derivation presented here is a detailed version of the found in [25].

The goal of the filter is to estimate the state $\mathbf{x} \in \mathbb{R}^n$ of a discrete-time controlled process governed by the linear stochastic equation

$$\mathbf{x}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_{k+1} + \mathbf{w}_k \tag{1}$$

With measurement equation,

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{2}$$

where $\mathbf{w}_k$ and $\mathbf{v}_k$ in Eqs. (1) and (2) represent the process and measurement noises respectively. Both are Gaussian noises and are assumed to be normally distributed with zero mean. The variables $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{z} \in \mathbb{R}^p$ represent the inputs and outputs of the digital system. The matrix $\mathbf{A}$ is the state transition matrix, $\mathbf{B}$ relates the input to the state dynamics and $\mathbf{H}$ is the measurement matrix.

Both process and measurement noises are uncorrelated with respective probability distributions

$$p(\mathbf{w}_k) \sim N(0, \mathbf{Q}_k) \tag{3}$$

$$p(\mathbf{v}_k) \sim N(0, \mathbf{R}_k) \tag{4}$$

$\mathbf{R}$ is the noise covariance associated with the measurement device and is generally given by the device's manufacturer while $\mathbf{Q}$ is associated with the uncertainty in the modeling of the system and most of the time has to be guessed using engineering intuition.

The estimation process consists of two main steps, an *a priori* and an *a posteriori*. The *a priori* step consists of trying to propagate the current estimate based only on the system process noise $\mathbf{w}_k$. The *a posteriori* process then take into account the measurement made and refines the *a priori* estimate by taking into account the measurement noise $\mathbf{v}_k$. The *a priori* and *a posteriori* errors are defined as

$$\mathbf{e}_k^- \equiv \mathbf{x}_k - \hat{\mathbf{x}}_k^- \tag{5}$$

$$\mathbf{e}_k \equiv \mathbf{x}_k - \hat{\mathbf{x}}_k \tag{6}$$

The errors between the true state and its estimates can be thought as an uncertainty with associated covariance matrices

$$\mathbf{P}_k^- = E[\mathbf{e}_k^- \mathbf{e}_k^{-T}] \tag{7}$$

$$\mathbf{P}_k = E[\mathbf{e}_k \mathbf{e}_k^T] \tag{8}$$

As mentioned, the measurement noise is not taken into account for the *a priori* estimate and the predicted measurement is

$$\mathbf{z}_k = \mathbf{H}_k \hat{\mathbf{x}}_k^- \tag{9}$$

The difference between the actual and predicted measurement is known as the measurement innovation $\delta \mathbf{z}_k^-$

$$\delta \mathbf{z}_k^- = \mathbf{z}_k - \hat{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^- \tag{10}$$

The *a priori* estimate of the next state is found using the previous state estimate

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{A}_k \hat{\mathbf{x}}_k + \mathbf{B}_k \mathbf{u}_{k+1} \tag{11}$$

Furthermore the *a priori* covariance is propagated by the the following Lyapunov equation

$$\mathbf{P}_{k+1}^{-} = \mathbf{A}_k \mathbf{P}_k \mathbf{A}_k^T + \mathbf{Q}_k \tag{12}$$

The goal of the Kalman filter is to improve the *a priori* state estimate by incorporating, in an optimal manner, the information retrieved from the noisy measurements. The updated estimate is a weighted sum of the *a priori* state and the product of a gain matrix with the measurement innovation.

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k + \delta \mathbf{z}_k^{-} = \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^{-}) \tag{13}$$

We will now show how to find the optimal Kalman gain $\mathbf{K}$. The *a posteriori* error can be expanded as

$$\begin{aligned}
\mathbf{e}_k &= \mathbf{x}_k - \hat{\mathbf{x}}_k \\
&= \mathbf{x}_k - \hat{\mathbf{x}}_k^{-} - \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^{-}) \\
&= -(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\hat{\mathbf{x}}_k^{-} - \mathbf{K}_k \mathbf{z}_k + \mathbf{x}_k \\
&= -(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\hat{\mathbf{x}}_k^{-} - \mathbf{K}_k(\mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k) + \mathbf{x}_k \\
&= -\hat{\mathbf{x}}_k^{-} - \mathbf{K}_k \mathbf{H}_k \hat{\mathbf{x}}_k^{-} - \mathbf{K}_k \mathbf{H}_k \mathbf{x}_k - \mathbf{K}_k \mathbf{v}_k + \mathbf{x}_k \\
&= (\mathbf{x}_k - \hat{\mathbf{x}}_k^{-}) + \mathbf{K}_k \mathbf{H}_k(\hat{\mathbf{x}}_k^{-} - \mathbf{x}_k) - \mathbf{K}_k \mathbf{v}_k \\
&= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k^{-}) - \mathbf{K}_k \mathbf{v}_k
\end{aligned} \tag{14}$$

Using the definition of the *a priori* error from Eq. (5) the *a posteriori* error is

$$\mathbf{e}_k = [\mathbf{I} - \mathbf{K}_k \mathbf{H}_k]\mathbf{e}_k^{-} - \mathbf{K}_k \mathbf{v}_k \tag{15}$$

The *a posteriori* covariance matrix is therefore

$$\mathbf{P}_k = E[\mathbf{e}_k \mathbf{e}_k^T] = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}_k^{-}(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T \tag{16}$$

In order to have the best estimate possible we are looking into minimizing the variances of the errors between the actual and estimated states. This correspond to

the diagonal terms of the *a posteriori* covariance matrix. Therefore the optimal gain Kalman Gain $\mathbf{K}_k$ should minimize the trace of $\mathbf{P}_k$. To get the trace of $\mathbf{P}_k$, Eq. (16) is expanded as follow

$$\begin{aligned}
\mathbf{P}_k &= E[\mathbf{e}_k\mathbf{e}_k^T] = (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
&= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I}-\mathbf{H}_k^T\mathbf{K}_k^T) + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
&= (\mathbf{P}_k^- - \mathbf{K}_k\mathbf{H}_k\mathbf{P}^-)(\mathbf{I}-\mathbf{H}_k^T\mathbf{K}_k^T) + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
&= \mathbf{P}_k^- - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^- - \mathbf{P}_k^-\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \qquad (17)
\end{aligned}$$

The trace is

$$Tr(\mathbf{P}_k) = Tr(\mathbf{P}_k^-) - 2Tr(\mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^-) + Tr(\mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T)\mathbf{K}_k^T) + Tr(\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T) \qquad (18)$$

the optimal gain will have to satisfy

$$\frac{\partial Tr(\mathbf{P}_k)}{\partial \mathbf{K}_k} = 0 \qquad (19)$$

To compute this derivative we need to use the two following properties

$$\frac{\partial Tr(\mathbf{M}_1\mathbf{M}_2\mathbf{M}_1^T)}{\partial \mathbf{M}_1} = 2\mathbf{M}_1\mathbf{M}_2 \qquad (20)$$

$$\frac{\partial Tr(\mathbf{M}_1\mathbf{M}_2)}{\partial \mathbf{M}_1} = \mathbf{M}_2^T \qquad (21)$$

This results in the following optimality condition

$$\frac{\partial Tr(\mathbf{P}_k)}{\partial \mathbf{K}_k} = -2(\mathbf{H}_k\mathbf{P}_k^-)^T + 2\mathbf{K}_k(\mathbf{H}_k)\mathbf{P}_k^-\mathbf{H}_k^T) + 2\mathbf{K}_k\mathbf{R}_k \qquad (22)$$

Rearranging this equation leads to the optimal Kalman gain

$$\begin{aligned}
-2(\mathbf{H}_k\mathbf{P}_k^-)^T + 2\mathbf{K}_k(\mathbf{H}_k)\mathbf{P}_k^-\mathbf{H}_k^T) + 2\mathbf{K}_k\mathbf{R}_k &= 0 \\
\mathbf{K}_k^{-1}(\mathbf{H}_k\mathbf{P}_k^-)^T &= (\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T) + \mathbf{R}_k \\
\mathbf{K}_k^{-1} &= (\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)(\mathbf{P}_k^-\mathbf{H}_k^T)^{-1}
\end{aligned} \qquad (23)$$

Finally taking the inverse

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k) \tag{24}$$

Based on this optimal Kalman gain the *a posteriori* covariance matrix in Eq(.16) can be simplified to

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- \tag{25}$$

Figure (2.1.1) summarizes the way the recursive Kalman filter works.

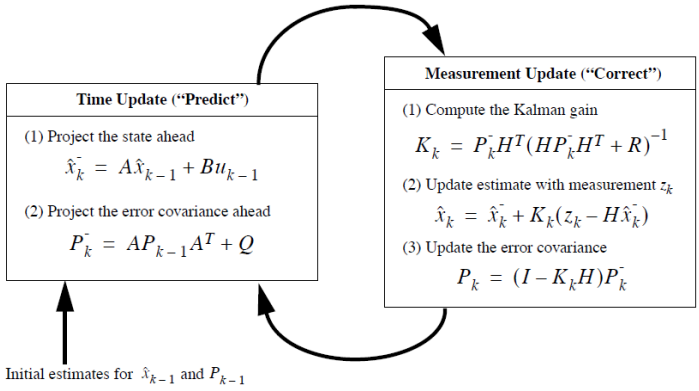


Figure 2: Linear Kalman Filter [25]

2.1.2 Extended Kalman Filter

The linear Kalman filter, is limited as its name indicates to estimate linear processes. However most systems are non-linear, and the filter needs to be adapted to correctly estimate non-linear processes. The Extended Kalman Filter (EKF) solves this issue by linearizing the nonlinear system around the current estimate and by propagating the covariance matrix using the previously developed linear Kalman filter. The nonlinear system is expressed as

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \tag{26}$$

with the measurement vector

$$\mathbf{z}_k = \mathbf{f}(\mathbf{x}_k, \mathbf{v}_k) \tag{27}$$

The noise parameters are the same as in Eqs. (3) and (4)

$$p(\mathbf{w}_k) \sim N(0, \mathbf{Q}_k) \tag{28}$$

$$p(\mathbf{v}_k) \sim N(0, \mathbf{R}_k) \tag{29}$$

The *a priori* estimate can be derived from the previous *a posteriori* estimate using

$$\hat{\mathbf{x}}_{k+1}^{-} = \mathbf{f}(\hat{\mathbf{x}}, \mathbf{u}_{k+1}, \mathbf{0}) \tag{30}$$

The measurement is estimated using

$$\hat{\mathbf{z}}_k = \mathbf{h}(\hat{\mathbf{x}}_k^{-}, \mathbf{0}) \tag{31}$$

The state and the measurement are approximated by the linear equations

$$\mathbf{x}_{k+1} \approx \hat{\mathbf{x}}_{k+1}^{-} + \mathbf{A}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{W}_k\mathbf{w}_k \tag{32}$$

$$\mathbf{z}_k \approx \hat{\mathbf{z}}_k^{-} + \mathbf{H}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k^{-}) + \mathbf{V}_k\mathbf{v}_k \tag{33}$$

The terms $\mathbf{A}_k, \mathbf{H}_k, \mathbf{V}_k$ and $\mathbf{W}_k$ are the Jacobians of Eqs.(26) and (27) at iteration k.

$$\mathbf{A}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, 0} \tag{34}$$

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k^{-}, 0} \tag{35}$$

$$\mathbf{V}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right|_{\hat{\mathbf{x}}_k^{-}, 0} \tag{36}$$

$$\mathbf{W}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right|_{\hat{\mathbf{x}}_k, \mathbf{u}_{k+1}, 0} \tag{37}$$

The Jacobians are generally derived analytically before hand. The prediction error, equivalent to the *a priori* estimation error of the Linear Kalman filter is defined as

$$\hat{\mathbf{e}}_{xk}^{-} \equiv \mathbf{x}_k - \hat{\mathbf{x}}_k^{-} \approx \mathbf{A}_{k-1}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}) + \varepsilon_k \tag{38}$$

The measurement residual, which is equivalent to the measurement innovation of the linear Kalman filter, is defined as

$$\hat{\mathbf{e}}_{zk}^{-} \equiv \mathbf{z}_k - \hat{\mathbf{z}}_k^{-} \approx \mathbf{H}_k(\hat{\mathbf{e}}_{xk}^{-}) + \eta_k \tag{39}$$

The variables ε and η are the linear approximations of $\mathbf{w}$ and $\mathbf{v}$. Using the linear variance propagation laws, the distributions of these two noise parameters are

$$p(\varepsilon_k) \sim N(\mathbf{0}, \mathbf{W}_{k-1}\mathbf{Q}_{k-1}\mathbf{W}_{k-1}^T) \tag{40}$$

$$p(\eta_k) \sim N(\mathbf{0}, \mathbf{V}_k\mathbf{R}_k\mathbf{V}_k^T) \tag{41}$$

The prediction error estimate, equivalent to the linear Kalman filter state update, is defined as the difference between the *a posteriori* and the *a priori* states.

$$\hat{\mathbf{e}}_k \equiv \hat{\mathbf{x}}_k - \hat{\mathbf{x}}_k^{-} \tag{42}$$

with

$$p(\hat{e}_k) \sim N(\mathbf{0}, E[\hat{\eta}_k\hat{\eta}_k^T]) \tag{43}$$

Assuming the predicted value of $\hat{\mathbf{e}}_k$, its estimation can be written

$$p(\hat{e}_k) = \mathbf{K}_k(\hat{\mathbf{e}}_{zk}^{-}) = \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{z}}_k^{-}) \tag{44}$$

The *a posteriori* state estimate can now be written as

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{x}}_k) \tag{45}$$

This is the same equation as in the linear Kalman filter case with the difference that the noise parameters are replaced by their linearized counterparts. The covariance propagation is

$$\mathbf{P}_{k+1}^{-} = \mathbf{A}_k\mathbf{P}_k\mathbf{A}_k^T + \mathbf{W}_k\mathbf{Q}_k\mathbf{W}_k^T \tag{46}$$

And the gain can be found by

$$\mathbf{K}_k = \mathbf{P}_k^{-}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^{-}\mathbf{H}_k^T + \mathbf{V}_k\mathbf{R}_k\mathbf{V}_k^T)^{-1} \tag{47}$$

A summary of the functioning of the Extended Kalman Filter is represented on Figure (2.1.2).

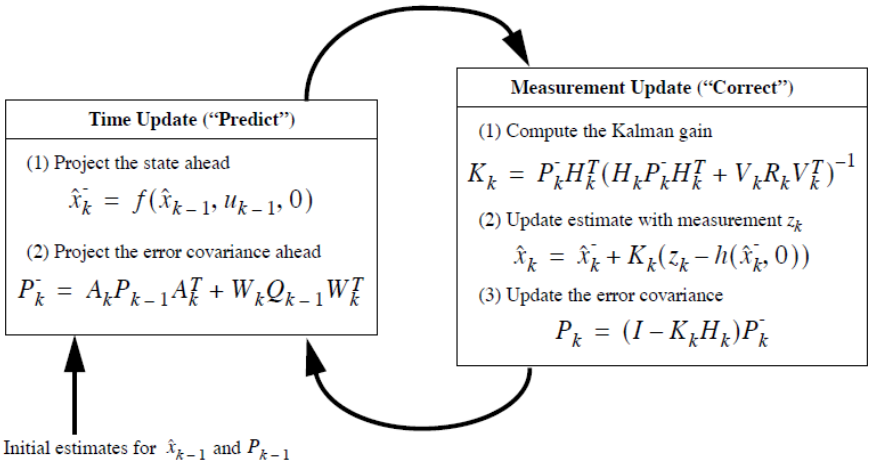


Figure 3: Extended Kalman Filter [25]]

2.2 Linear Quadratic Regulator

2.2.1 Derivation

Linear Quadratic Regulator (LQR) design is an optimal control problem for linear systems. LQR is widely used in industry as it is easily tunable and have good robustness properties. Considering a linear system with states $\mathbf{x} \in \mathbb{R}^n$ and inputs $\mathbf{u} \in \mathbb{R}^m$

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{48}$$

The cost to minimize is quadratic and is generally formulated as

$$J = \frac{1}{2}\int_0^T (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})dt + \frac{1}{2}\mathbf{x}^T(T)\mathbf{F}_T\mathbf{x}(T) \tag{49}$$

Where $\mathbf{Q}, \mathbf{F}_T \in \mathbb{R}^{n\times n}$ and $\mathbf{R} \in \mathbb{R}^{m\times m}$. These matrices are the weights put respectively on the states, final states and control inputs. For simplicity, these weight matrices are generally chosen to be diagonal. This presents the advantage of making the control tuning more intuitive. For instance increasing the diagonal terms of $\mathbf{R}$ penalizes the control effort, which generally leads to a longer settling time and reduces the chances of saturation. On the other hand a large $\mathbf{Q}$ compare to $\mathbf{R}$ can lead to fast

system performance but great changes in control inputs. The control designer needs to have a good understanding of the system to adequately tune an LQR controller; hence the need for intuition.

Using Pontryagin maximum principle the Hamiltonian can be formulated as

$$H = \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u} + \lambda^T(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) \tag{50}$$

Where $\lambda \in \mathbb{R}^n$ is the costate vector satisfying the differential equation

$$-\dot{\lambda} = \left(\frac{\partial H}{\partial \lambda}\right)^T = \mathbf{Q}\mathbf{x} + \mathbf{A}^T\lambda \qquad \lambda(T) = \mathbf{F}_T\mathbf{x}(T) \tag{51}$$

The system dynamics can be re-expressed as

$$\dot{\mathbf{x}} = \left(\frac{\partial H}{\partial \mathbf{x}}\right)^T = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \qquad \mathbf{x}(0) = \mathbf{x}_o \tag{52}$$

and the optimal control can be found by taking the partial derivative of the Hamiltonian with regards to the control inputs

$$0 = \left(\frac{\partial H}{\partial \mathbf{x}}\right) = \mathbf{R}\mathbf{u} + \lambda^T\mathbf{B} \tag{53}$$

The optimal control strategy is therefore

$$\mathbf{u} = -\mathbf{R}^{-1}\mathbf{B}^T\lambda \tag{54}$$

The following Hamiltonian matrix can now be formed

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & -\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T \\ -\mathbf{Q} & -\mathbf{A}^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{55}$$

With initial conditions $\mathbf{x}(0) = \mathbf{x}_o$ and $\lambda(T) = \mathbf{F}_T\mathbf{x}(T)$ it is a *two-point boundary value problem* which is hard to solve. However we can make an educated guess that the costate solution is in the form

$$\lambda(t) = \mathbf{P}(t)\mathbf{x}(t) \qquad \lambda(T) = \mathbf{F}_T\mathbf{x}(T) \tag{56}$$

The costate differential equation Eq.(52) can be rewritten

$$\begin{aligned}\dot{\lambda}(t) &= \dot{\mathbf{P}}(t)\mathbf{x}(t) + \mathbf{P}(t)\dot{\mathbf{x}}(t) \\ &= \dot{\mathbf{P}}(t)\mathbf{x}(t) + \mathbf{P}(t)(\mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P}(t))\mathbf{x}(t) \end{aligned} \tag{57}$$

Expanding the costate term leads to the differential equation on $\mathbf{P}(t)$

$$\dot{\mathbf{P}}(t) = \mathbf{P}(t)\mathbf{A} + \mathbf{A}^T\mathbf{P}(t) - \mathbf{P}(t)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P}(t) + \mathbf{Q} \qquad \mathbf{P}(T) = \mathbf{F}_T \tag{58}$$

Equation (58) is called the *Riccati ODE*. It can be solved backward and the optimal input becomes $\mathbf{u}(t) = -\mathbf{K}_{LQR}(t)\mathbf{x}(t)$ with

$$\mathbf{K}_{LQR}(t) = \mathbf{R}^{-1}\mathbf{B}\mathbf{P}(t) \tag{59}$$

This finite optimal control problem leads to a Time-varying control gain $\mathbf{K}_{LQR}(t)$. Implementing such Time-varying gain can be cumbersome, the *Riccati ODE* will however at some point rich steady-state and the gain will eventually converge and remain constant. This steady-state gain can be predetermined by extending the Time horizon and relaxing the terminal constraint of the optimal control problem. The cost function can then be written as

$$J = \int_0^\infty (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})dt \tag{60}$$

as before the LQR gain can be written as Eq(59) with the difference that P is constant and satisfies what is called the *Algebraic Riccati Equation*

$$0 = \mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q} \tag{61}$$

2.2.2 LQR robustness

An optimal LQR controller is given by $\mathbf{u} = -\mathbf{K}\mathbf{x}$ where $\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}^T\mathbf{P}$. Implementation of the controller therefore requires a complete and accurate knowledge of the system states $\mathbf{x}$. However $\mathbf{x}$ is generally corrupted by measurement noise, and in

addition the system matrices **A** and **B** may not be an exact representation of the system. It is important that the controller performs well despite model uncertainty and noisy measurement.

It can be shown however that the LQR controller has some strong robustness properties in term of Gain (GM) and Phase margin (PM). These robustness indicators are used for SISO systems and can be misleading when analysing MIMO systems, but are still a reassuring fact.

The gain and phase margins can be represented on a Nyquist plot as shown on Figure (2.2.2). The Gain Margin represents how much the controller signal can be amplified before driving the system unstable while the Phase margin is an indicator of how much time delay the system can endure before going unstable.

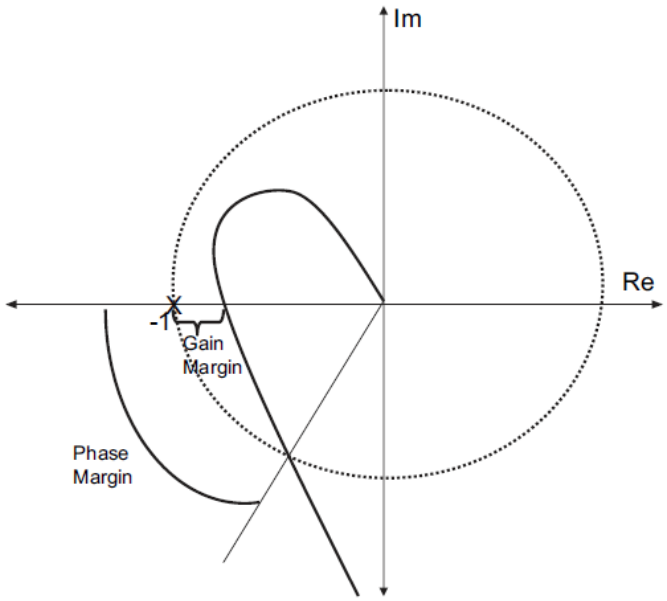


Figure 4: Representation of GM and PM on a Nyquist diagram [24]

We will now demonstrate some of the robustness properties of the LQR controller when dealing with SISO systems. These properties can however be extended to MIMO system with few changes. Let us consider the system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u \tag{62}$$

Where $u \in \mathbb{R}, \mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ and $\mathbf{A} \in \mathbb{R}^{n \times n}$, with the LQR cost function

$$J = \int_0^\infty (\mathbf{x}^T \mathbf{Q}\mathbf{x} + u^2 r)dt \tag{63}$$

where $r > 0 \in \mathbb{R}, \mathbf{Q} \geq 0 \in \mathbb{R}^{n \times n}$, based on the previous derivation, the optimal control law is

$$u = -r^{-1} b^T \mathbf{P}\mathbf{x} \tag{64}$$

Where $\mathbf{P} \in \mathbb{R}^{n \times n}$ satisfies the ARE

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}br^{-1}b^T\mathbf{P} + \mathbf{Q} = 0 \tag{65}$$

Adding and subtracting jw as well as substituting the control gain $\mathbf{K} = -r^{-1}b^T\mathbf{P}$ leads to

$$\mathbf{P}(jw\mathbf{I} - \mathbf{A}^T) + (-jw\mathbf{I} - \mathbf{A}^T)\mathbf{P} + \mathbf{K}^T r\mathbf{K} = \mathbf{Q} \tag{66}$$

Multiplying Eq.(66) on the left by $b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}$ and by $(jw\mathbf{I} - \mathbf{A})^{-1}b$ on the right yields

$$b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}\mathbf{P}b + b^T\mathbf{P}(jw\mathbf{I} - \mathbf{A})^{-1}b + b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}\mathbf{K}^T r\mathbf{K}(jw\mathbf{I} - \mathbf{A})^{-1}b = b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}\mathbf{Q}(jw\mathbf{I} - \mathbf{A})^{-1}b \tag{67}$$

Simplifying gives

$$\begin{aligned} r + b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}\mathbf{Q}(jw\mathbf{I} - \mathbf{A})^{-1}b &= (1 - b^T(-jw\mathbf{I} - \mathbf{A}^T)^{-1}\mathbf{K}^T)r(1 - \mathbf{K}(jw\mathbf{I} - \mathbf{A})^{-1}b) \\ &= r|1 - \mathbf{K}(jw\mathbf{I} - \mathbf{A})^{-1}b|^2 \end{aligned} \tag{68}$$

It follows that

$$|1 - \mathbf{K}(jw\mathbf{I} - \mathbf{A})^{-1}b|^2 \geq 1 \tag{69}$$

We now define the following equation as the Loop-transfer function of the system

$$\mathbf{L}(jw) = -\mathbf{K}(jw\mathbf{I} - \mathbf{A})^{-1}b \tag{70}$$

Equation (69) can then be rewritten as

$$|1 + \mathbf{G}(jw)| \geq 1 \tag{71}$$

Equation (71) implies that the Loop-Transfer function $\mathbf{L}(jw)$ never enters the unitary disk centered at (-1,0) in the complex plane, as represented on Figure (2.2.2)

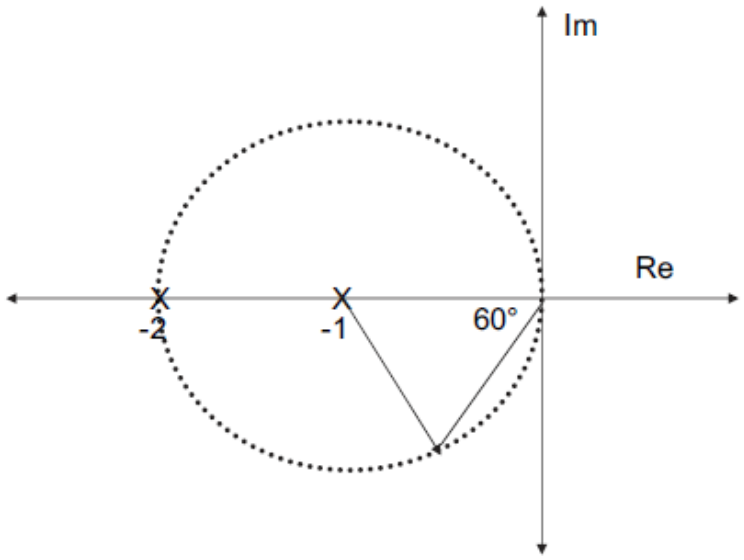


Figure 5: Robustness properties of LQR controller [24]

It is clear from Figure (2.2.2) that the Loop-Transfer function crosses the negative real axis only for values less than -2, which correspond to a negative gain margin of at least -6db. Furthermore $\mathbf{L}(jw)$ must leave the unit circle with at least an angle of -60 deg or 60 deg. A SISO LQR controller therefore offers an infinite positive GM and a -6 db negative GM. It also ensures a PM of at least 60 degrees. These margins are guaranteed and are an indication of a robust controller. Both SISO and MIMO LQR controllers will be used for this thesis.

CHAPTER III

DC MOTOR

In this section we present the design and implementation of the Peak-seeking strategy on a DC motor driving a variable pitch propeller. This setup was chosen to first validate the Peak-seeking procedure before applying it to more complex system such as a turboprop engine. The DC motor system was simulated, but an experimental setup was also built for future testing.

3.1 Problem Definition

The following model describes a system comprising an electric motor and a variable pitch propeller. This is a simplistic representation of the turbofan system (core gas turbine and fan). The equations of motion for a DC motor and a variable pitch propeller system are:

$$\begin{aligned} \tfrac{d}{dt}\omega_m(t) &= \tfrac{1}{I_m}\left[k_b i(t) - B\omega_m(t) - Q(\omega_m(t), \beta(t))\right], \\ \tfrac{d}{dt}i(t) &= \tfrac{1}{L}\left[-Ri(t) - k_b\omega_m(t) + v(t)\right], \\ y(t) &= T(\omega_m(t), \beta(t)), \end{aligned} \tag{72}$$

where $v(t)$ is the voltage across the armatures and is used to control the motor speed, $i(t)$ is the current through the motor , $\omega_m(t)$ is the motor shaft angular speed, I_m is the motor inertia, k_b is the induced emf constant, B is the motor damping coefficient, R is the armature resistance, L is the self-inductance, Q is the propeller torque, T is the propeller thrust, and $\beta(t)$ is the propeller pitch angle. Figure 6 is an illustration of the DC motor and variable pitch propeller system.

By considering the propeller blade as a twisted wing, we used blade element theory *blade element theory* to compute the forces acting on it. Figure 7 shows a

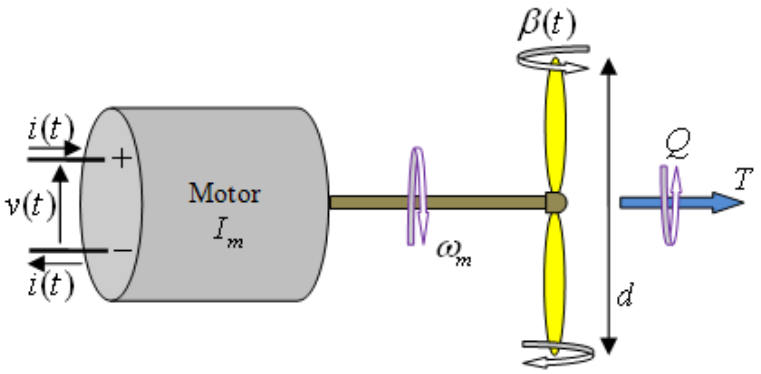


Figure 6: DC Motor with variable pitch propeller schematic

cross section of the blade, with the corresponding forces and important angles. The forces exerted by the surrounding air on this blade cross section, are the lift and drag forces, respectively perpendicular and parallel to the velocity of the blade. The

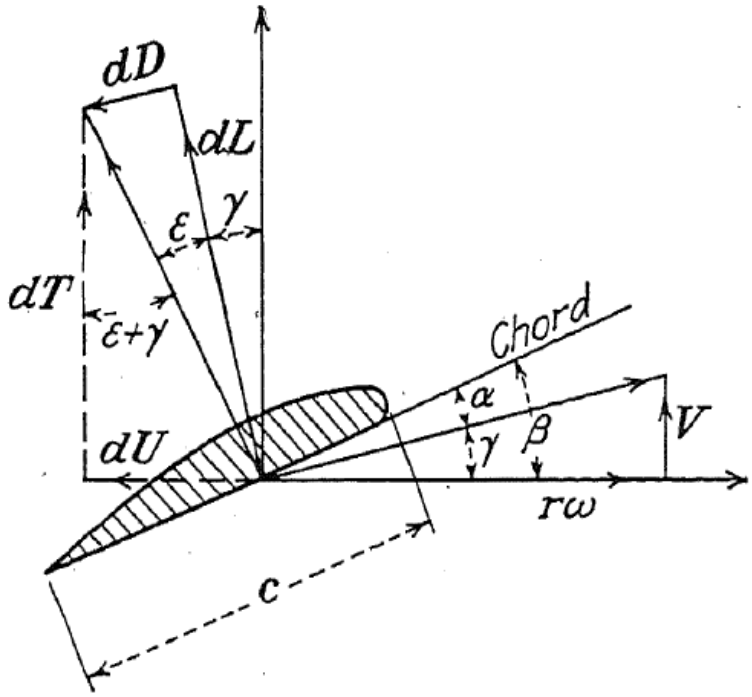


Figure 7: Resolution of forces acting on a propeller section [4]

velocity appears on Figure 7 as a vector with horizontal component $r\omega_m = 2\pi rn$ and with vertical component V. The square of the resulting velocity is

$$V^2 + (r\omega)^2 = V^2 + (2\pi rn)^2 = n^2 d^2 \left[J^2 + (2\pi r/d)^2\right], \tag{73}$$

where $J = (V/nd)$ denotes the advance ratio, n denotes the round per second and d denotes the propeller diameter. Let $G := J^2 + (2\pi r/d)^2$. The angle between the velocity vector and the horizontal axis is $\gamma = \arctan(V/r\omega)$. If β denotes the angle between the plane of rotation and blade section chord, the angle of attack is $\alpha = \beta - \gamma$. In the following it is assumed that, for all blade sections, $C_L = C_{L_\alpha}(\alpha - \alpha_0)$ and $C_D = C_{D_0} + kC_{L_\alpha}^2(\alpha - \alpha_0)^2$ are functions of the angle α. The lift and drag forces for the blade element between the radii r and $r + dr$ can be obtained as:

$$\begin{aligned} dL &= 0.5 C_L \rho V^2 dS \\ &= \tfrac{1}{2} C_{L_\alpha}(\alpha - \alpha_0)\rho n^2 d^2 G c(r) dr, \\ dD &= 0.5 C_D \rho V^2 dS \\ &= \tfrac{1}{2}(C_{D_0} + kC_{L_\alpha}^2(\alpha - \alpha_0)^2)\rho n^2 d^2 G c(r) dr, \end{aligned} \tag{74}$$

where $c(r)$ denotes the chord length of the blade section at the radii r. It will be useful to decompose the resultant of the lift and drag forces into the components dT and dU parallel and perpendicular to the propeller axis. These components are given by

$$\begin{aligned} dT &= dL\cos\gamma - dD\sin\gamma, \\ dU &= dL\sin\gamma + dD\cos\gamma. \end{aligned} \tag{75}$$

The overall forces on the propeller are

$$\begin{aligned} T &= 0.5 m\rho n^2 d^2 \textstyle\int_0^{d/2} G(C_L\cos\gamma - C_D\sin\gamma)c(r)dr, \\ U &= 0.5 m\rho n^2 d^2 \textstyle\int_0^{d/2} G(C_L\sin\gamma + C_D\cos\gamma)c(r)dr, \end{aligned} \tag{76}$$

where m is the number of blades of the propeller that contributes to the propeller torque Q and to power P, which can be written as

$$\begin{aligned} Q &= m\textstyle\int_0^{d/2} r dU \\ &= 0.5 m\rho n^2 d^2 \textstyle\int_0^{d/2} G(C_L\sin\gamma + C_D\cos\gamma) r c(r) dr, \\ P &= \omega Q = 2\pi n m \textstyle\int_0^{d/2} r dU \\ &= \pi n m \rho n^2 d^2 \textstyle\int_0^{d/2} G(C_L\sin\gamma + C_D\cos\gamma) r c(r) dr. \end{aligned} \tag{77}$$

For the purpose of this experiment, the overall system is assumed to be attached to fix point which makes the vertical speed of the propeller null. The thrust is then a unique function of the lift, and the generated torque now only depends on the drag force. Generating the thrust, torque and power of the propeller using the aforementioned equations is not trivial, and it would take too much time if these terms had to be calculated at each iterations, when simulating the overall system. Therefore a static table was constructed to relate $T(t)$, $Q(t)$ and $P(t)$ to the changes in $\omega_m(t)$ and $\beta(t)$.

$$[T(t), Q(t), P(t)] = PropTable\left(\omega_m(t), \beta(t)\right). \tag{78}$$

This table will be constantly used when simulating the system.

3.2 Peak-Seeking Control

Looking at the system in a simplistic manner, its control inputs are the propeller pitch angle β and the DC motor voltage V which ultimately controls the propeller velocity ω_m. The DC motor parameters were chosen to fit a standard DC motor used in industry and the characteristics of the blade were chosen in accordance to the DC motor performances. In order for the DC motor to track a commanded speed in a minimum amount of time and power, an LQR controller was designed to minimize the speed settling time as well as the current drawn. Furthermore, thanks to its robust characteristics, the LQR controller allows the closed loop DC motor system to adequately perform speed tracking, despite the effects of the disruptive load force generated by the propeller blade.

An off-line study was first performed to analyze the power consumption of the system, and to determine the impact of each of the system parameters on the location of the optimal blade pitch angle. Using equation 72 and equation 78 to describe the system when operating at steady state, the power consumption was calculated when staying at a constant thrust and varying the blade pitch angle from 0° to 15°. Figure 8 shows the results of this simulation. It is clear from this Figure that a global

minimum exists.

Intuition might suggests that the pitch angle allowing a maximum Lift to Drag ratio, would lead to an optimal power consumtion, as at this specific angle, the load is at its minimum (for a given Thrust). However, following equation 78, we can see that as the back emf and damping coefficients values increase, the motor speed ω_m has a greater impact on the power drawn. This quick analysis reveals that these two coefficients are of a major importance, and cannot be disregarded when trying to determine the optimal pitch angle. Figure 9 shows how these parameters exactly interfere with the minimum location. The change in color represent the change in the back emf coefficient. If both of these parameters are null, the minimum will be located at the maximum lift over drag pitch angle. While if they tend to infinity the minimum will move towards the propeller stall angle, as at this angle the blade velocity is at a minimum (for a given thrust). One can also note that the damping coefficient has the bigger impact on the minimum location than the back emf coefficient. This information reinforces the idea of using an online Peak-seeking method, as the damping coefficient is likely to change over the life time of the DC motor.

A simulation of the power consumed for different Thrust commands was also investigated and is represented on Figure 10. As shown on the figure the set is convex and allows the existence of a minimum power consumption for every iso-Thrust line. A thrust controller is required in order for the system to remain at a certain thrust level while being optimized through the peak-seeking process. The thrust could be modeled as $(C + \delta)\omega_n^2$ where C is a constant determined from equation 76 and δ represents a parametric uncertainty. The thrust was controlled using gain-scheduling, with the motor speed as the scheduling parameter. Three PI controllers were chosen to cover the all thrust operating region. An extended Kalman filter was also implemented to have a better estimate of the Thrust measurement. Figure 10 shows the limits between these regions and also help understanding the Peak-seeking procedure as the

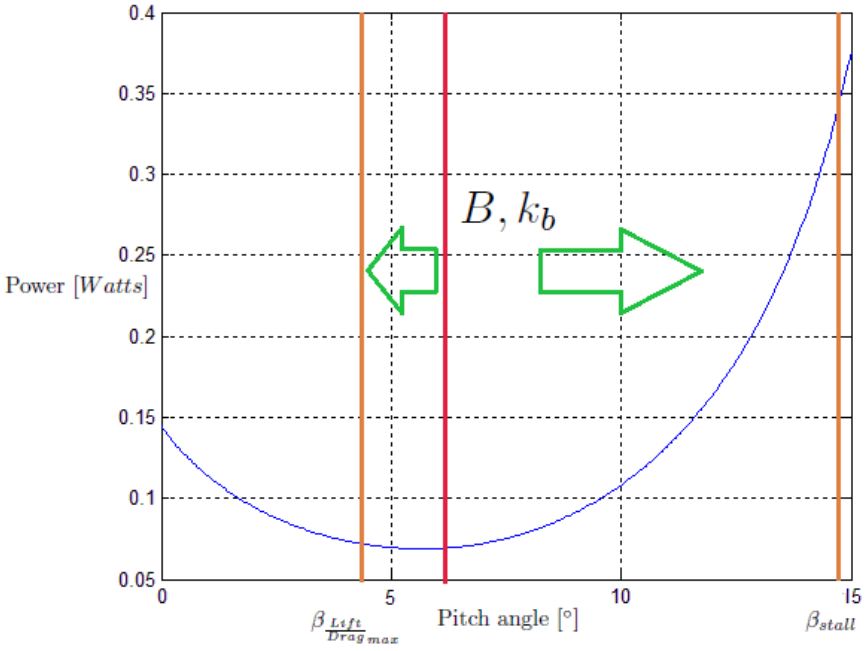


Figure 8: Model Power consumption over pitch angle for an iso-Thrust command

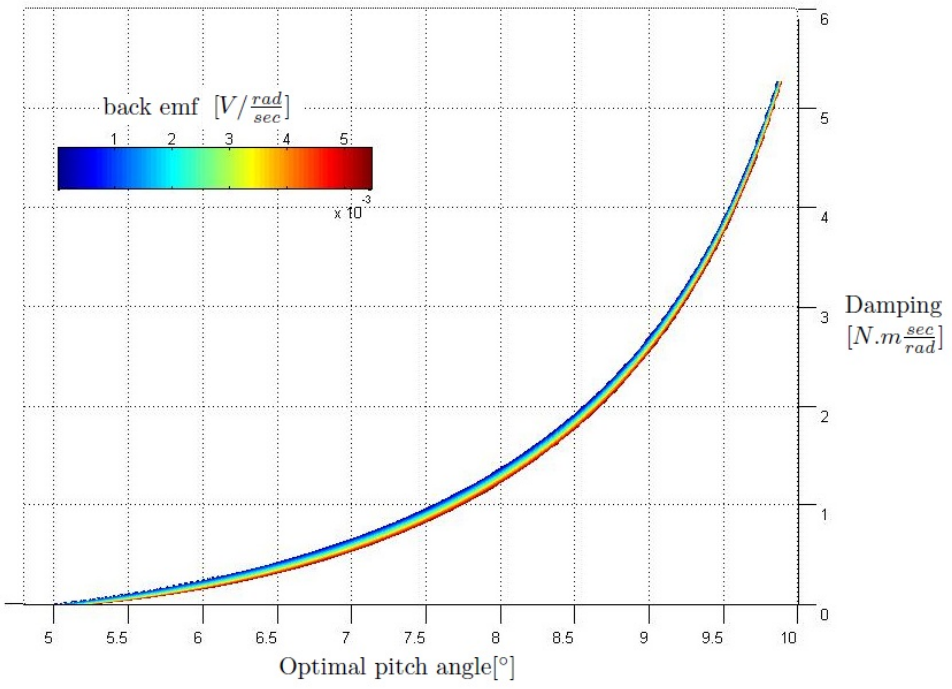


Figure 9: Change in optimal pitch angle for variations in damping and back emf of the DC motor

goal is to keep the system on an iso-thrust line represented by the dark dashed-line while the Peak-seeking algorithm varies the pitch angle β. A representation of the

Peak-seeking strategy applied to the DC motor system is also shown on Figure 11. The power needs to be constantly changing in order for the EKF to estimate its derivatives. A linear sweep around the pitch angle of interest β_l was therefore perform, and some excitations were also added to allow for better observability. Using the estimates of the power derivatives, the Newton method was carried using

$$\beta_l = \beta_{l-1} - \zeta \frac{\hat{\nabla}_p}{\hat{H}_p} \tag{79}$$

Where $\hat{\nabla}_p$ and $\hat{H}_p$ are respectively the Gradient and Hessian estimates of the power consumption at the pitch angle β_{l-1} . ζ is a fading coefficient based on the standard deviation of the set of the previously commanded pitch angles.

$$\zeta = \eta \frac{1}{l-n} \sum_{i=l-n}^{l} (\beta_i - \mu)^2 \tag{80}$$

Where η is a tuning parameter, μ is the mean and n is used to determine the size of the set. As the pitch angle is converging to the its optimal value, the fading coefficient ζ decreases, since the standard deviation of the previously calculated pitch angles is decreasing. Therefore once getting closer to the optimal value, only small changes could occur. This is to compensate for the fact that the Hessian estimate might be noisy and could randomly drive the system away from its minimum.

3.3 Kalman Filter Design

We will present in this section the Kalman filter used to estimate the derivatives of the power function. The filter is a modified version of the one used by J. J. Ryan and J. L. Speyer [7]. The power function could be approximated by a Taylor series around a nominal pitch angle β_k.

$$p(\beta) \approx p(\beta_k) + p'(\beta_k)(\beta - \beta_k) + \frac{1}{2}p''(\beta_k)(\beta - \beta_k)^2 + \frac{1}{6}p'''(\beta_k)(\beta - \beta_k)^3 + o((\beta - \beta_k)) \tag{81}$$

Where $p(\beta_k)$ is the power function, $p'(\beta_k)$ and $p''(\beta_k)$ are respectively the first, second and third derivatives of the power function evaluated at the pitch angle β_k,

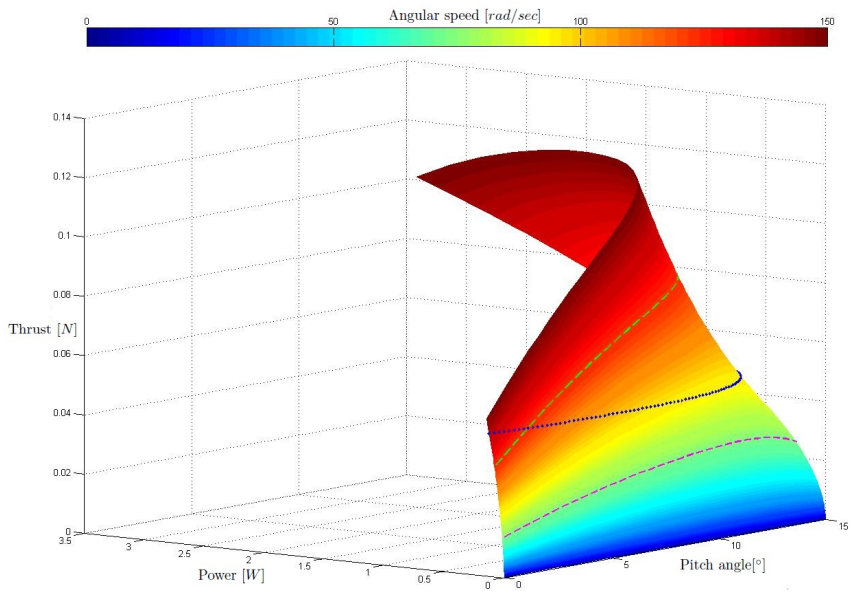


Figure 10: Simulation of the power behavior for changes in Thrust, pitch angle and and rotor speed

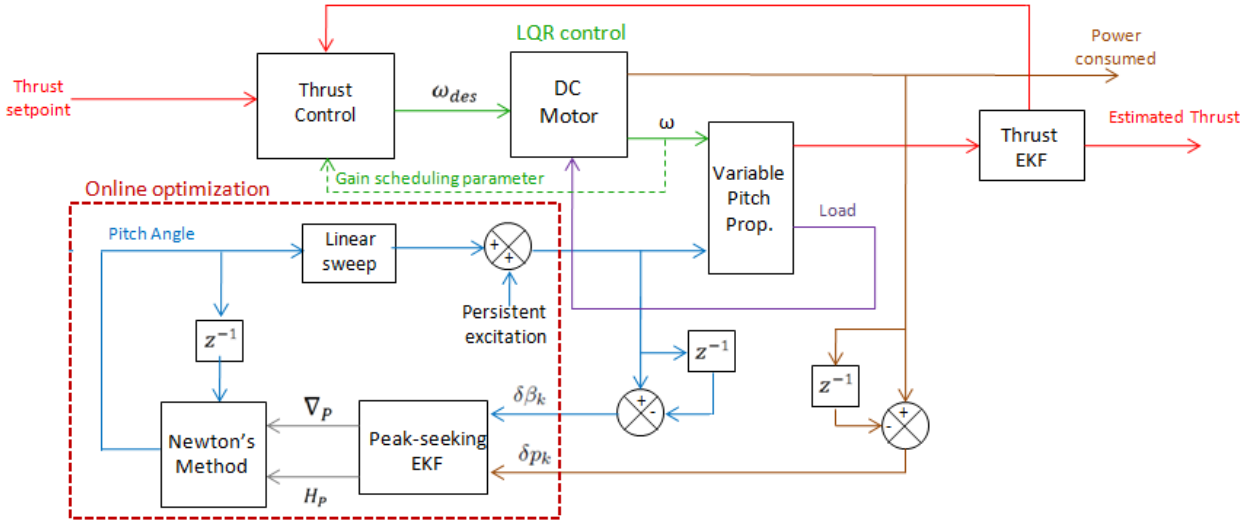


Figure 11: Peak-seeking control structure for DC motor and the variable pitch propeller

and o represents the higher order terms. Even though only the first two derivatives are required for the peak-seeking process, an estimation of the third derivative enhance the filter performances.

Evaluating(81) at β_{k-1} leads to

$$\delta p_k = \delta\beta_k p'_k + \frac{1}{2}\delta\beta_k^2 p''_k + \frac{1}{6}\delta\beta_k^3 p'''_k \tag{82}$$

where $\delta p_k = p_k - p_{k-1}$ and $\delta\beta_k = \beta_k - \beta_{k-1}$. This a discrete time representation of the power function, the derivatives enter linearly the power equation and could be considered as the states of the system. The vector to estimate is therefore

$$\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{p}'_k & \hat{p}''_k & \hat{p}'''_k \end{bmatrix}^T, \tag{83}$$

Several measurements could be taken at each iteration k , taking more than 3 measurements will lead to a set of over determined equation, which will ultimately improve the estimation's accuracy. However more measurements also imply a slower filter convergence. By making the distinction between the pitch angles generated by the Newton methods β_k and the one added by the persistent equation β_s, the measured vector can be written as

$$\Delta\mathbf{p}_k = \begin{bmatrix} p(\beta_k) - p(\beta_{k-1}) \\ p(\beta_k) - p(\beta_{k-2}) \\ \vdots \\ p(\beta_k) - p(\beta_{k-n}) \\ p(\beta_k) - p(\beta_{s-1}) \\ \vdots \\ p(\beta_k) - p(\beta_{s-m}) \end{bmatrix} \tag{84}$$

The filter will keep track of the previously visited steady-state points and use them for the next derivative estimation. A certain number of excitation points β_s can also be added to the estimation process to increase observability. In order to increase the Peak-seeking process speed, these excitation points do not have to be generated

at each iteration, but could be created once every few new pitch angle command is issued. Following a Taylor serie approximation and based on the previously written measurement vector, the measurement matrix is,

$$\mathbf{H}_k = \begin{bmatrix} \beta_k - \beta_{k-1} & \frac{1}{2}(\beta_k - \beta_{k-1})^2 & \frac{1}{6}(\beta_k - \beta_{k-1})^3 \\ \beta_k - \beta_{k-2} & \frac{1}{2}(\beta_k - \beta_{k-2})^2 & \frac{1}{6}(\beta_k - \beta_{k-2})^3 \\ \vdots & \vdots & \vdots \\ \beta_k - \beta_{k-n} & \frac{1}{2}(\beta_k - \beta_{k-n})^2 & \frac{1}{6}(\beta_k - \beta_{k-n})^3 \\ \beta_k - \beta_{s-1} & \frac{1}{2}(\beta_k - \beta_{s-1})^2 & \frac{1}{6}(\beta_k - \beta_{s-1})^3 \\ \vdots & \vdots & \vdots \\ \beta_k - \beta_{s-m} & \frac{1}{2}(\beta_k - \beta_{s-m})^2 & \frac{1}{6}(\beta_k - \beta_{s-m})^3 \end{bmatrix} \tag{85}$$

The excitation pitch angles β_s are generally generated to be far away from the current pitch angle of interest and in the same direction as pointed by the Newton method. As taking measurement ahead of the current point, leads to better derivatives estimate. More excitations points leads to better estimate, but also increase the Peak-seeking process time. Most of the time only one excitation point is used, for it still gives a significant improvement in the derivatives estimate and does not impede much on the Peak-seeking completion time.

The measurement equation for the filter is written as

$$\Delta \mathbf{p}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{86}$$

where $\mathbf{x}_k$ is a 3×1 vector representing the derivative parameters to estimate, and $\mathbf{v}_k$ is the Gaussian measurement noise of variance $\mathbf{R}_k$

Considering taking the Taylor series expansion of the derivatives term and evaluating at the next time step with a change of $\delta\beta_k$ in pitch angle yields to

$$
\begin{aligned}
p'_{k+1} &= p'_k + (\beta_k - \beta_{k-1})p''_k + \tfrac{1}{2}(\beta_k - \beta_{k-1})^2 p'''_k \\
p''_{k+1} &= p''_k + (\beta_k - \beta_{k-1})p'''_k \\
p'''_{k+1} &= p'''_k
\end{aligned} \tag{87}
$$

The discrete state matrix representing the derivatives dynamics can therefore be written as

$$
\mathbf{A}_k = \begin{bmatrix} 1 & \beta_k - \beta_{k-1} & \frac{1}{2}(\beta_k - \beta_{k-1})^2 \\ 0 & 1 & \beta_k - \beta_{k-1} \\ 0 & 0 & 1 \end{bmatrix} \tag{88}
$$

The third derivative is modeled as a random walk process. The filter is generally tuned by only looking at this derivative, as if correctly estimated, the first and second derivatives will also be. The number of derivatives taken into account in the Taylor expansion could be increased, but it would in turns require more computation, and would stop making sense after a certain order.

The discrete time process representing the derivatives behavior is,

$$
\mathbf{x}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{w}_k \tag{89}
$$

where $\mathbf{w}_k$ is the gaussian process noise with variance $\mathbf{Q}_k$ which can be written as

,

$$
\mathbf{Q}_k = \begin{bmatrix} \omega_{Gr} & 0 & 0 \\ 0 & \omega_{He} & 0 \\ 0 & 0 & \omega_{Third} \end{bmatrix} \tag{90}
$$

where ω_{Gr}, ω_{He}, ω_{Third} represent the variances of the expected process noises (on each derivatives). These parameters were determined by trial and error, with basic a rule of thumb that $\omega_{Third} >> \omega_{He} >> \omega_{Gr}$

The a priori error covariance matrix P_k^- is computed following:

$$
P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \tag{91}
$$

Where W_k represents the jacobian matrix of the state transition model with respect to the process noise w_k.

The derivatives estimation was then computed following a regular Extended Kalman Filter approach.

$$\begin{aligned} K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \\ \hat{x_k} &= \hat{x_k}^- + K_k(\Delta p_k - H_k \hat{x}_{k+1}^-) \\ P_k &= (I - K_k H_k) P_k^- \end{aligned} \tag{92}$$

3.4 Simulation Results

To first demonstrate the filter performance, the pitch angle was sweep from 0° to 12° while the power measurement was corrupted by a Gaussian noise with a 0.001 standard deviation. A sweep was performed with an increment of $\beta_k - \beta_{k-1}$ of $\frac{1}{30}^\circ$ between two iterations. An excitation point of 0.5 degrees in front of the a pitch angle of interest was added every 10 iterations. The LQR of the DC motor was set to iterate at 100 Hz . Figure (12) shows on the upper graph the corrupted measurement in difference in power, while the lower Graph represent the Hessian estimate. The EKF is keeping track of 25 measurement per iteration, and the excitations present in each iteration are not represented on this graph. On the upper graph the red dots represent the power measurements, the blue line represent the estimate of the difference, reconstructed from the derivatives estimates, and the pink link is the true difference value with its process noise. On the lower graph The red line is the true Hessian, the blue line is the Hessian estimate and the green line is the Hessian estimate with a low pass filter to minimize abrupt changes. We can see on this graph the very good estimation performance and the quick filter convergence. The Gradient estimate was not represented as, the Hessian one is much more subjected to noise, a good Hessian estimate implies a good Gradient too.

Once the filter correctly tuned, the Peak-seeking process was implemented. Power

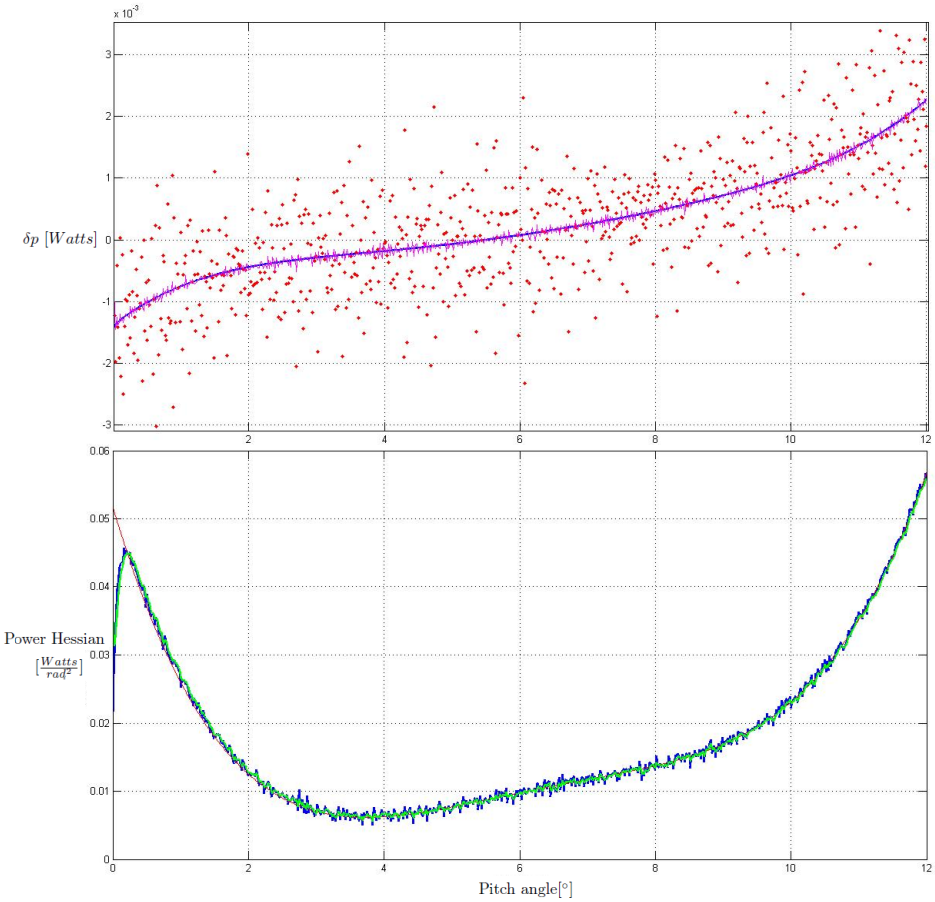


Figure 12: Upper graph: Measurement of noisy δ_p. Lower graph: Hessian estimate

Measurements were taken at 2 hz while the Kalman filter also iterated at 2 hz. The measurements were corrupted by white noise with a standard deviation of 0.01, which is very noisy considering that the power consumed is in the order of 0.15 $Watts$. The results of the Peak-seeking algorithm is shown on Figure (13), with the red dash line representing the actual minimum location and the blue line is the estimation made by the system. To avoid that the pitch angle creates large disturbances that could lead to unstability, the pitch angle could only be changed with a maximum of 0.5 °. Figure (14) shows how the power consumption is minimized over time. In this case, with an initial guess for the optimal angle to be at zero degrees, the optimization allows the system to reduce its power consumption by 0.07 W, which is considerable for such small system. If used on a UAV for instance, this could add some operating

time to the drone.

The filter took approximately 6 seconds to converge to a good derivatives estimate and once it converged it took less than 15 seconds for the Peak-seeking to reach the optimal pitch angle, which is a very satisfying time. The Peak-seeking completion time could even be faster if larger changes in pitch angle are allowed, but with the risk of introducing too much disturbances. Therefore, depending on the robustness of the controller used, a trade-off exists between the Peak-seeking completion time and the stability of the closed-loop system.

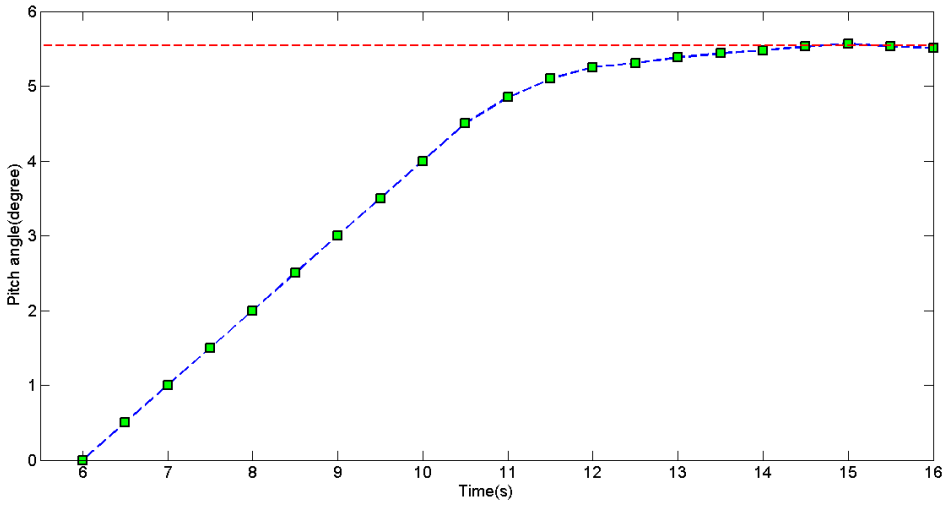


Figure 13: Peak-seeking over Time

3.5 Hardware setup

An experimental setup was also designed to demonstrate the feasibility of the Peak-seeking method on a real system. The hardware was put together and the code was designed to perform the Peak-seeking on a brushless DC motor driving a variable pitch propeller. The platform is not fully operational yet, but is presented here as part of the work that will be performed in the future. A schematic of the hardware setup is represented on Figure 15, while a picture of the real hardware is represented on figure 16.

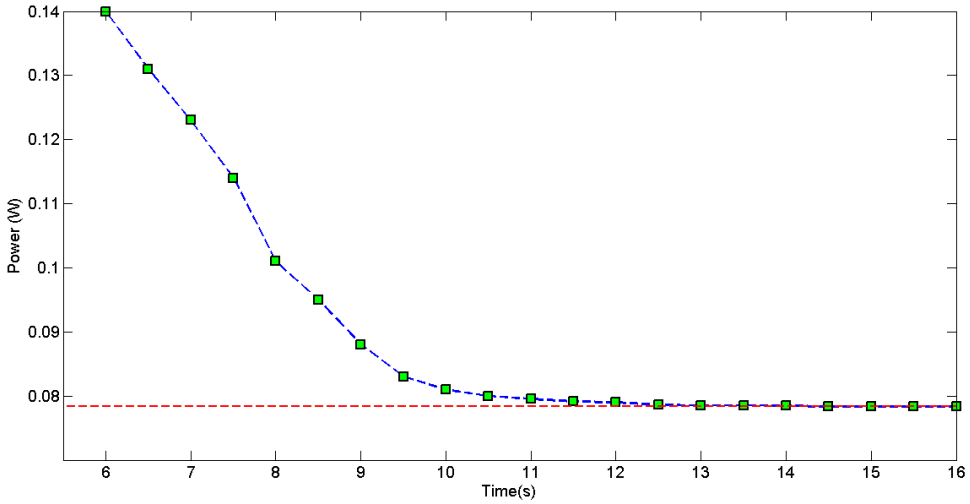


Figure 14: Power minization over Time

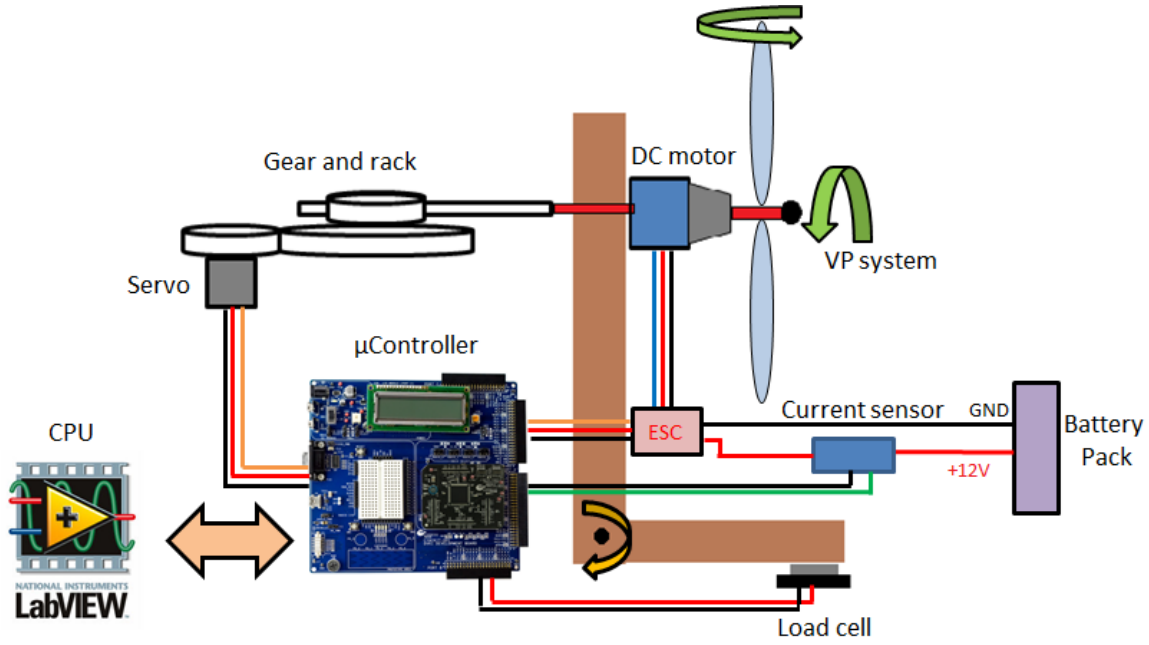


Figure 15: DC motor with variable pitch system schematic

A gear and rack system was used to increase the torque out of the servo motor, as well as have a finer precision when changing the pitch angle command. A PSOC 5 microcontroller is used to send the measurements from the current sensors to a computer through RS-232 serial communication. Using LabView the computer is then running the Kalman Filter and can display both the measurement and estimates in

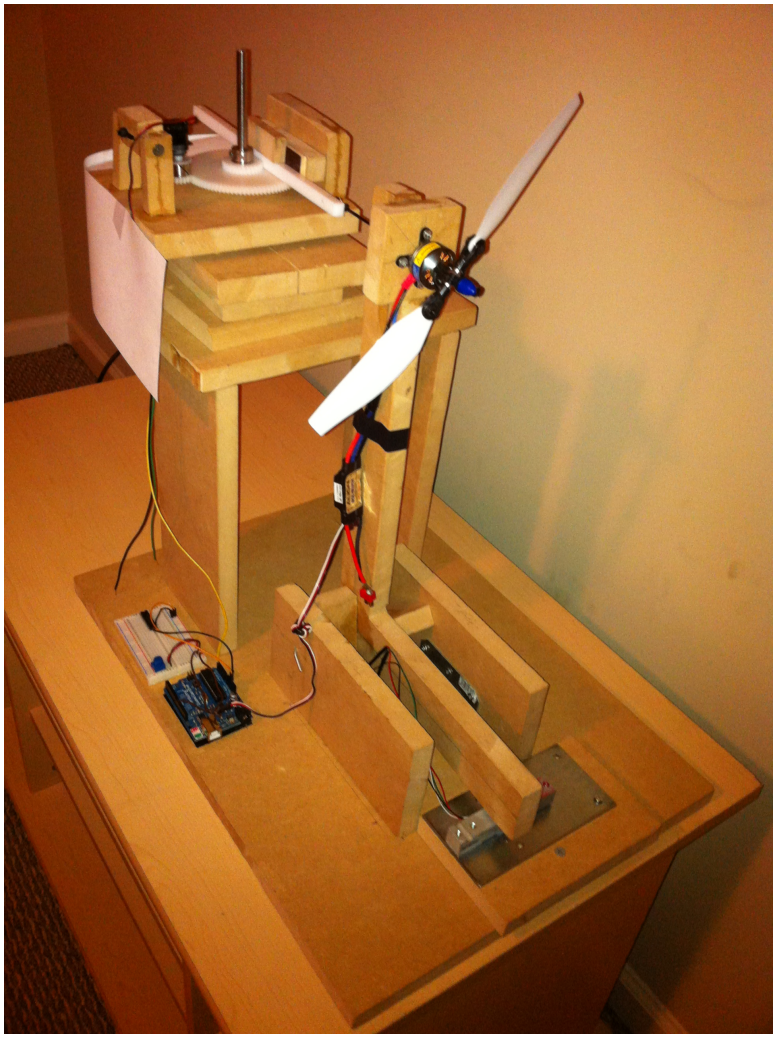


Figure 16: DC motor with variable pitch system hardware

real time. LabView allows also to tune the Kalman gains in real time, which facilitates and expedite the filter tuning process

An ESC with governor mode (already closed-loop on motor speed) was used to control the hollow shaft brushless DC motor. The PWM command sent to the ESC sets the reference speed for the motor, and the ESC makes sure it stays at this speed regardless of the load applied to it. The ESC inner speed controller is taking the back emf voltage of the motor as a measurement and was tuned by the manufacturer. A simple PI controller running on the PSOC 5 was added to control the thrust using the ESC PWM command as a control input. The thrust measurement was performed

using a load cell as represented on Figure 15 As shown on Figure 16 the hardware was built and ran successfully in open loop. Also the LabView VI was tested with fictitious data and performed adequately. All the code has been written and only the thrust PI controller remains to be tuned. However, this project encountered some setbacks and is currently not fully operational for the Peak-seeking process. It will be nonetheless working in a close future, and could be used in the future as well as platform with educational purpose.

CHAPTER IV

TURBOPROP ENGINE

The main goal of this thesis is to apply the peak-seeking control technique to systems widely used in industry such as Turboprop engines. The online optimization scheme could reduce the operating cost of the engine by minimizing the fuel consumption for instance. Furthermore, the "Plug and Play " aspect of the peak-seeking control is even more appealing for Turboprop engines where components may come from different companies and have not initially been designed to work with each other. The optimization algorithm would allow to find the optimal performing conditions without having to go through pain-stacking system analysis or extensive testing. It could be therefore very useful for small companies working with such engines.
For this research, the Peak-seeking algorithm has been developed for a simulation of a Turboprop engine located in MIT's Gas Turbine Laboratory. This engine belongs to the Air Force Research Laboratory and both Georgia Tech, under the supervision of Dr. Eric Feron, and Aurora Flight Science, under the supervision of Dr. James Paduano, have been working on it. The engine has unfortunately not been fully operational during the preparation of this thesis and the peak-seeking algorithm could only be simulated with the hope it will be tested in a near future.

In this section, we will first go over the Turbofan model and simulation results. We will then present the control architecture developed for the Turboprop engine and we will finally talk about the peak-seeking approach designed for this system and analyze its performance.

4.1 Turbofan model

In this section we will briefly present and explain the engine model used for this research. The engine simulation was principally investigated by Nathan Fitzgerald in Aurora Flight Science and most of the engine details can be found in [30].

The turboprop engine was constructed by adding a variable pitch propeller to a small JetCat SPT5 twin spool turboshaft engine. Some mechanical contraption was designed to connect the propeller to the engine's shaft and a schematic of the system is represented on Figure 17. Despite a more complicated dynamics, the overall functioning of the turboprop engine is very similar to the DC motor driving a variable pitch propeller system. Both systems are controlled through two inputs, the blade angle and DC voltage for the brushed DC motor case and the blade angle and fuel flow for the turboprop engine. Because of these similarities in the control, the Peak-seeking procedure for the turboprop engine is almost identical to the DC motor case, as we will discuss soon.

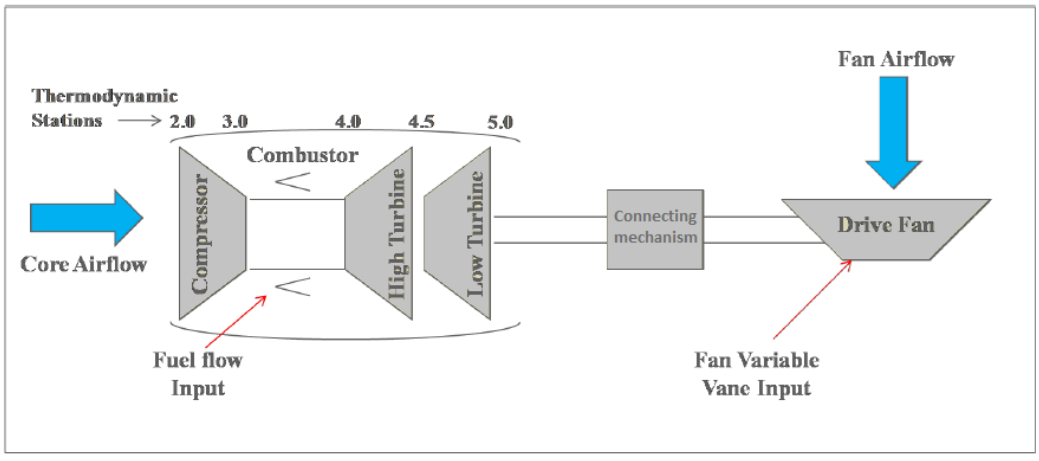


Figure 17: Schematics of turboprop

Fuel is provided to the core of the engine which consists of a compressors, combustor, and a high speed turbine. The exhaust gases from the core engine power a

low pressure turbine that transmits power to the variable pitch blade. The core spool speed and fan spool speed are generally respectively referred as N_2 and N_1 speeds. The core spool speed and acceleration $\dot{N}_2$ are both of a primordial importance when it comes to safely run the engine, as if these values are above certain limits, the engine could be irreversibly damaged. These speed and acceleration constraints need to be addressed in the control strategy, as later presented in this report.

Because of the lack of information furnished by both the JetCat SPT5 and the propeller blade furnishers, the turboprop simulation was built upon experimental data. The dynamics of the engine was determined using a lumped-parameter approach and several key assumptions had to be made in order to model the engine behavior as presented in [30].

Some general information regarding the JetCat SPT5 engine is represented on Fig.18(b) as well as a picture of the Jetcat engine on Fig.18(a). A CAD model of the turboprop test bed as well as a picture of the actual set-up are also both represented on Fig.19 and 20.

(a) JetCAT SPT5 engine

	SPT-5
Shaft Power	11hp
Thrust	55lbs
Service Life	25hrs
Output Speed	1500-7000
Spool Speed	50-170,000
Core Pressure Ratio	2:1
Outer Diameter and Length	83mmx365mm

(b) Performance table

Figure 18: JetCAT engine

Another aspect of this specific engine is its complete distribution in its control architecture. This is worth mentioning as it allows some modularity and therefore, combined with the Peak-seek procedure, can lead to a very modular and performant system that embrace the notion of "Plug and Play". The distribution takes place on both the hardware and software side of the control. Each actuators and sensors have

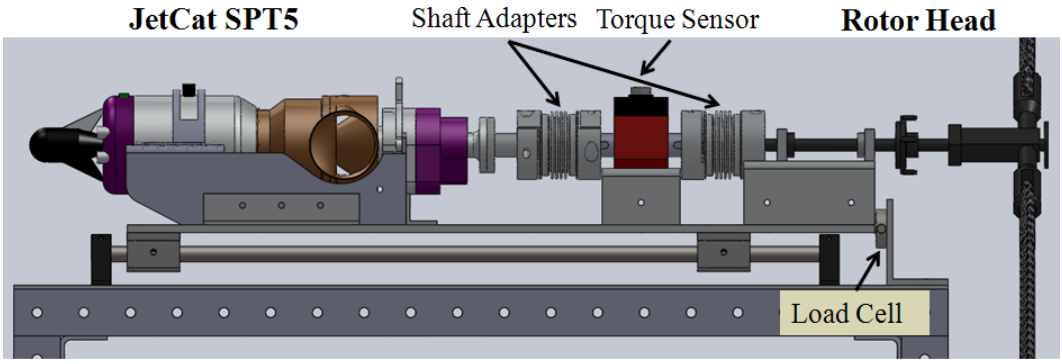


Figure 19: CAD model of the Turboprop

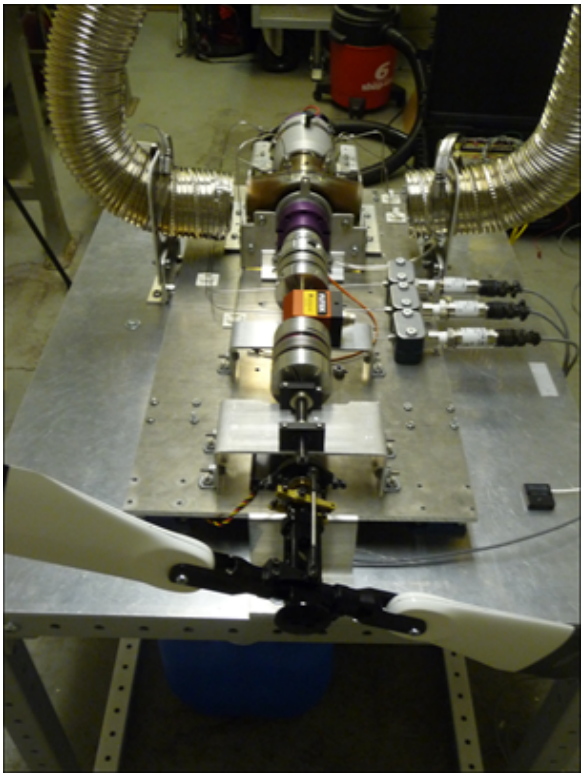


Figure 20: Turboprop engine test bed

their own micro-controllers which are all networked on a CAN bus line. By doing so, adding or taking of a sensor out of the engine is made easy as it only requires to connect or disconnect the smart sensor from the networked line, while it might be cumbersome with a centralized approach. A diagram representing the networked sensor and actuator layout as well as the distributed control law is shown on Fig. 21. A similar diagram this time more representative of the the control law distribution is

represented on Fig. 22, this diagram also has the advantage of showing the sensors location on the different thermodynamics stations.

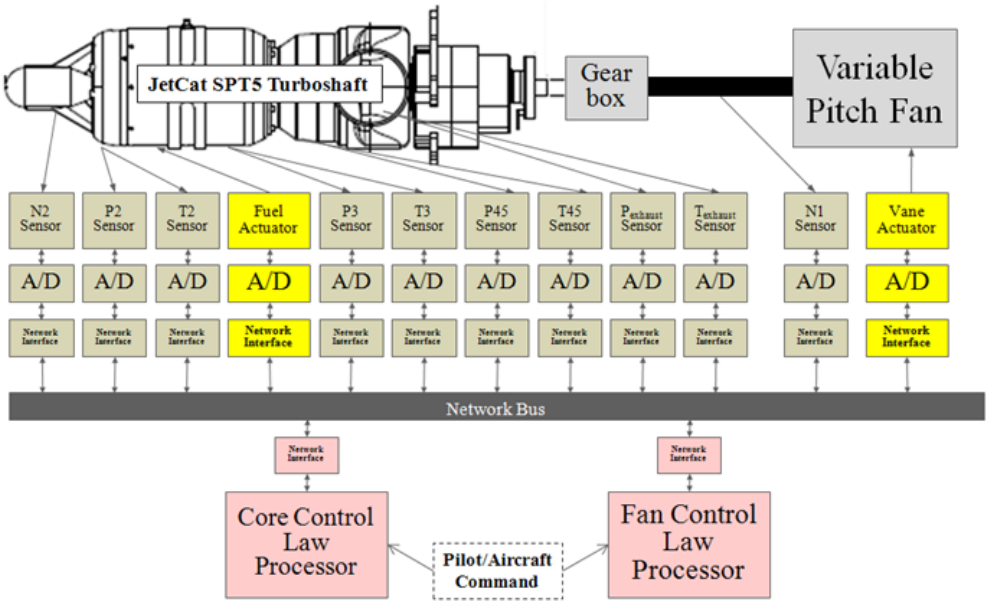


Figure 21: Turboprop engine distributed hardware layout

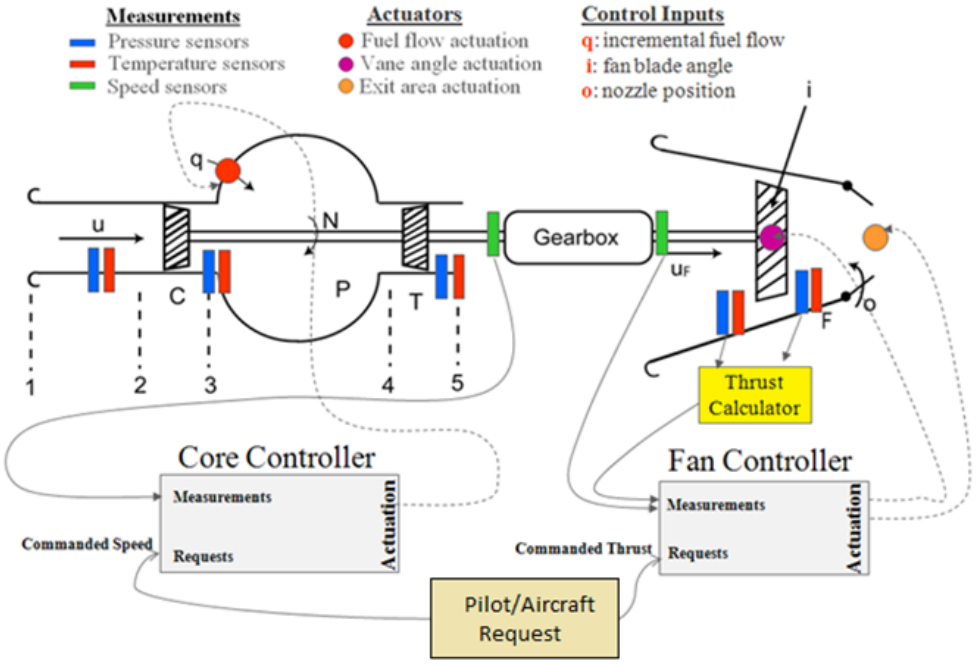
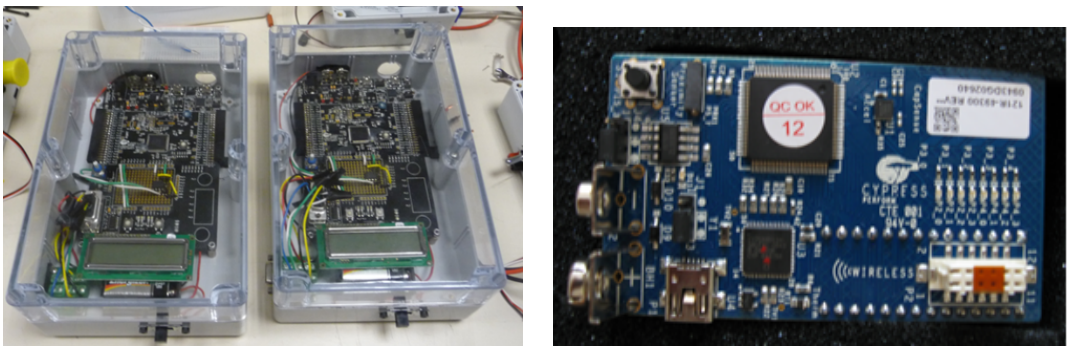


Figure 22: Turboprop engine distributed control layout

The hardware set-up was put together by the author of this thesis, Nathan Fitzgerald and Dr. Manuj Dhigra. The micro-controllers used for the smart sensors and actuators were PSOC 5 because of their versatility and their ability to communicate on a CAN line. The actuators controllers were chosen to be more performant than the sensor's one as the control laws are be computed on it. A picture representing both actuators and sensors controllers is shown on Fig. 32(a) and 32(b).



(a) smart-actuator controller node (b) smart-sensor controller node

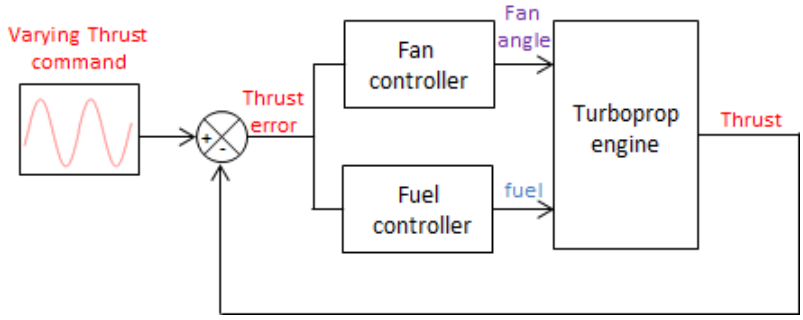
Figure 23: Actuators and sensors nodes

4.2 Control Architecture

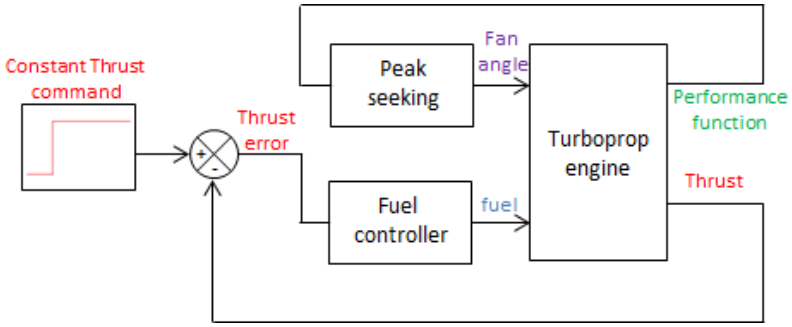
The diagrams shown on Figure 21 and 22 represent the control architecture of the engine without the Peak-seeking process. Originally the turboprop engine is controlled using two inputs, the vane angle and the fuel flow. However, the Peak-seeking procedure developed in this thesis does not allow feedback control on the input used for the online-optimization, in this case the fan pitch angle. The turboprop control law has to be changed from two controllable inputs system to a one controllable input only, with the fan pitch angle controlled by the Peak-seeking process.

Suppressing the vane angle as part of the feedback control scheme certainly has an impact of the performance of the engine, notably when trying to track fast changes in thrust command. However, the Peak-seeking procedure is designed to optimized the engine when running at a constant thrust level, and should not be employed anyway

when having fast changes in the thrust command. Taking the case of a commercial jet-liner, the Peak-seeking algorithm will be most useful when the aircraft is at cruise speed for instance. The Peak-seeking cannot work when the thrust is undergoing fast and large changes, which is the case when the aircraft is performing maneuvers at take-off and landing. A switch mechanism could then be used to switch between a two control input strategy when having to deal with varying thrust command (landing, take-off), and a one control input strategy when operating at constant thrust (cruise) and running the Peak-seeking algorithm. This switching concept between a peak-seeking and a varying thrust controller is represented on Fig.24.



(a) Two control inputs strategy for varying thrust command



(b) One control input strategy and Peak-seeking for constant thrust command

Figure 24: Control strategies of the Turboprop engine

The turboprop engine simulation is non-linear and is not easy to control. The problem is now to design a controller that will ensure thrust tacking for the engine.

4.2.1 Control design

We will now present the control design of the Turboprop engine. We will focus on controlling the system with only the fuel flow as represented on Fig.24(b). We do not worry in this thesis about designing a new control law, as we rather use a simple method to control the engine. A simple control method used for non-linear system is gain scheduling. It has the advantages of using linear control techniques and is therefore relatively easy to design while partially ensuring some robustness and performance properties. Considering the non-linear system representing the Turboprop dynamics

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t)) \\ y(t) &= g(\mathbf{x}(t), \mathbf{u}(t)) \end{aligned} \tag{93}$$

Where for the Turboprop engine $\mathbf{x} \in \mathbb{R}^{2\times 1}$ is the state vector consisting of the core spool speed x^c and the fan speed x^f. $\mathbf{u} \in \mathbb{R}^{2\times 1}$ is the control input vector consisting of the fuel flow u^c and the blade pitch angle u^f. Finally y is the thrust output of the Turboprop.

Considering $\Omega \subset \mathbb{R}^4$ to be the region representing all possible state and control vectors during the turboprop operation. A set of properly separated equilibrium operating points $\mathbf{x}_{e_k}$ and inputs $\mathbf{u}_{e_k}$, $k \in I = \{1, 2, ...L\}$ can be chosen to represent the center of L operating subregions Ω_k. It is important to chose the subregions Ω_k such that the interior of each region does not intersect, i.e $int(\ \Omega_k) \bigcap int(\Omega_j) = \varnothing$, $\forall k \neq j$ and also that $\bigcup_{k=1}^{L} \Omega_k = \Omega$. By rewriting $\mathbf{x} = \mathbf{x}_{e_k} + \delta\mathbf{x}$, $\mathbf{u} = \mathbf{u}_{e_k} + \delta\mathbf{u}$ and $y = y_{e_k} + \delta y$, the nonlinear model can be linearized using Taylor's series approximation

and expressed in a piece-wise manner as,

$$\left.\begin{aligned}
\dot{\delta\mathbf{x}}(t) &= \mathbf{A}_k\delta\mathbf{x}(t) + \mathbf{B}_k\delta\mathbf{u}(t)\\
\delta y(t) &= \mathbf{C}_k\delta\mathbf{x}(t) + \mathbf{D}_k\delta\mathbf{u}(t)\\
\mathbf{A}_k &= \frac{\partial f}{\partial \mathbf{x}}\Big|_{(\mathbf{x}_{e_k},\mathbf{u}_{e_k})}\\
\mathbf{B}_k &= \frac{\partial f}{\partial \mathbf{u}}\Big|_{(\mathbf{x}_{e_k},\mathbf{u}_{e_k})}\\
\mathbf{C}_k &= \frac{\partial g}{\partial \mathbf{x}}\Big|_{(\mathbf{x}_{e_k},\mathbf{u}_{e_k})}\\
\mathbf{D}_k &= \frac{\partial g}{\partial \mathbf{u}}\Big|_{(\mathbf{x}_{e_k},\mathbf{u}_{e_k})}
\end{aligned}\right\} \quad \forall(\mathbf{u},\mathbf{x}) \in \Omega_k \tag{94}$$

The nonlinear model of the Turboprop was developed based on experimental data, and is completely numerical. The Jacobians of the nonlinear plant could not therefore be determined analytically and were numerically approximated by perturbing the model around the equilibrium points $(\mathbf{x}_{e_k}, \mathbf{u}_{e_k})$, $k \in I$.

It is important to notice that at each time t, $\mathbf{x}(t)$ and $\mathbf{u}(t)$ belong to only one subregion Ω_k. The new state vectors $\delta\mathbf{x}(t) \in \mathbb{R}^2$ now follows a linear dynamic around its designated subregion Ω_k and is piecewise differentiable. The fuel flow control input δu^c can now follow a control law that has been designed using linear control techniques as we will see now.

4.2.2 Gain-scheduling

Assuming an appropriate model linearization that correctly represents the engine behavior across its whole operating space Ω, a set of linear controllers $K_k, k \in I$, each in effect when the system is operating in an associated region Ω_k can be used to control the system. With an adequate switching process, these controllers could stabilize the nonlinear plant around its whole operating environment. This switching between a predetermined set of linear-controllers is called Gain-scheduling and is widely used in industry to control nonlinear systems.

The linearization of the plant remains an approximation and the system still contains some nonlinearities when operating around any equilibrium points $(\mathbf{x}_{e_k}, \mathbf{u}_{e_k})$.

These nonlinearities can be seen as model uncertainty. Furthermore the states and thrust measurements will most likely be corrupted by noise while the system may encounters external disturbances. Therefore it is primordial to design robust compensators that will ensure both stability and performances of the Turboprop.

For the Peak-seeking case, since we are only interested in controlling the thrust of the turboprop and only the fuel flow can be used as a control input, the system is actually a SISO. In addition of simplifying the control law design, the fact that the system is SISO also allows us to use SISO robustness characteristics such as gain and phase margins (GM and PM) to ensure the robustness of the controller. It is always possible to do so on MIMO system, using μ analysis or the small gain theorem, but using GM and PM makes makes more physical sense of the robustness of the controller regarding distrubances and time delays.

A control technique that is generally used for MIMO systems but that ensure very performing and robust controllers also for SISO systems, is LQR (Linear Quadratic Regulator) control. Section 2 of this thesis summarize the derivation of an LQR controller and demonstrates its robustness, notably ensuring an infinite GM and a PM of at least 60 degrees. A set of LQR controllers were therefore chosen to perform thrust tracking for the tuboprop engine.

In order to perform tracking of the thrust output we assumed in our case that the thrust is directly measurable as the Peak-seeking process will first be tested on a test bench and not in flight where the thrust has to be estimated. If performed in flight the controller could be changed to an LQG that will both estimate and control the thrust at the expense of loosing robustness. For the test-bed case, assuming a measurable thrust and assuming for now that the engine only operates within a region Ω_k described by a linear model as in eq. 94, the system has to be augmented first with a state x_i representing the integral of the tracking error,

$$\dot{x}_i = e = \delta y - \delta r \tag{95}$$

Where e is the tracking error and $\delta r = r - y_{e_k}$ is the thrust reference when the system is operating in the linear region Ω_k. Since only the fuel flow is used as a control input we can rewritten Eq. 94 and replacing the input matrices by $\mathbf{B}_k^c = \mathbf{B}(:,1)_k$ and $\mathbf{D}_k^c = \mathbf{D}(1,1)_k$. Also we can notice that Eq 95 can be rewritten more simply as $e = y - r$ but it is wiser to use it as originally written in order to augment the dynamics of the system with the tracking error as follow,

$$\begin{bmatrix} \delta\dot{\mathbf{x}} \\ \dot{x}_i \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_k & 0 \\ -\mathbf{C}_k & 0 \end{bmatrix}}_{\mathbf{A}_{aug}} \begin{bmatrix} \delta\mathbf{x} \\ x_i \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{B}_k^c \\ -\mathbf{D}_k^c \end{bmatrix}}_{\mathbf{B}_{aug}} \delta u^c + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \delta r \tag{96}$$

A LQR controller is designed on the augmented system to minimize the following cost function

$$J = \int_0^T \mathbf{x}_{aug}^T \mathbf{Q} x_{aug} + \delta u^T R \delta u dt. \tag{97}$$

where $x_{aug} = [\delta\mathbf{x}^T x_i^T], \in \mathbb{R}^{3\times 1}$ is the augmented state, $\mathbf{Q} \in \mathbb{R}^{3\times 3}$ is the weighting matrix on the augmented state and have important weights on the error state to allow for better tracking, and $R \in \mathbb{R}^{1\times 1}$ is the weighting matrix on the input. The LQR gains are calculated as,

$$\mathbf{K} = R^{-1}\mathbf{B}_{aug}^T\mathbf{P} \tag{98}$$

Where $\mathbf{P} \in \mathbb{R}^{3\times 3}$ solves the algebraic Riccati equation

$$\mathbf{Q} + \mathbf{A}_{aug}^T\mathbf{P} + \mathbf{P}\mathbf{A}_{aug} - \mathbf{P}\mathbf{B}_{aug}R^{-1}\mathbf{B}_{aug}^T\mathbf{P} = 0. \tag{99}$$

It is important to separate $\mathbf{K}$ in two subcontrollers, $\mathbf{K}_o$ used as state feedback and K_i acting on the integral of the tracking error.

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_o \\ K_i \end{bmatrix} \tag{100}$$

The closed system under static feedback $\delta u = -\mathbf{K}\mathbf{x}_{aug}$ can be written as,

$$\begin{bmatrix} \delta\dot{\mathbf{x}} \\ \dot{x}_i \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_k - \mathbf{B}_k^c\mathbf{K}_o & -\mathbf{B}_k^c K_i \\ -\mathbf{C}_k^c + \mathbf{D}_k^c\mathbf{K}_0 & \mathbf{D}_k^c K_i \end{bmatrix}}_{\mathbf{A}_{closed}} \begin{bmatrix} \delta\mathbf{x} \\ x_i \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \delta r \tag{101}$$

4.2.3 Switching mechanism

Ideally, in order to ameliorate the gain-scheduled control procedure, one could design a very large number of linear controllers covering the whole operating space. However, designing a large number of compensators takes time and it was found for the purpose of this experiment that the operating space could sufficiently be divided in four regions.

Operating region 1:

$$\begin{gathered} x_e^c = 1.0,\ x_e^f = 0.9524,\ u_e^c = 1.0,\ u_e^f = 16,\ y_e = 255.868 \\ \mathbf{A}_1 = \begin{bmatrix} -5.0 & 0 \\ 3.5 & -2.3 \end{bmatrix}, \mathbf{B}_1 = \begin{bmatrix} 1.4 & 0 \\ 0.63 & -0.085 \end{bmatrix} \\ \mathbf{C}_1 = \begin{bmatrix} 354.25 & 380.25 \end{bmatrix}, \mathbf{D}_1 = \begin{bmatrix} -6 & 15 \end{bmatrix} \end{gathered} \tag{102}$$

Operating region 2:

$$\begin{gathered} x_e^c = 0.726,\ x_e^f = 0.5,\ u_e^c = 0.4685,\ u_e^f = 16,\ y_e = 70.5125 \\ \mathbf{A}_2 = \begin{bmatrix} -1.9 & 0.061 \\ 0.45 & -1.10 \end{bmatrix}, \mathbf{B}_2 = \begin{bmatrix} 1.57 & 0.0002 \\ 0.3 & -0.023 \end{bmatrix} \\ \mathbf{C}_2 = \begin{bmatrix} 42.25 & 226.20 \end{bmatrix}, \mathbf{D}_2 = \begin{bmatrix} -1.75 & 4.80 \end{bmatrix} \end{gathered} \tag{103}$$

Operating region 3:

$$x_e^c = 0.5327,\ x_e^f = 0.3678,\ u_e^c = 0.3,\ u_e^f = 16,\ y_e = 38.155$$
$$\mathbf{A}_3 = \begin{bmatrix} -0.85 & 0.032 \\ 0.32 & -0.64 \end{bmatrix}, \mathbf{B}_3 = \begin{bmatrix} 1.10 & 0.0002 \\ 0.17 & -0.011 \end{bmatrix} \tag{104}$$
$$\mathbf{C}_3 = \begin{bmatrix} 27.95 & 167.7 \end{bmatrix}, \mathbf{D}_3 = \begin{bmatrix} -1.1 & 2.6 \end{bmatrix}$$

Operating region 4:

$$x_e^c = 0.295,\ x_e^f = 0.161,\ u_e^c = 0.145,\ u_e^f = 16,\ y_e = 7.317$$
$$\mathbf{A}_4 = \begin{bmatrix} -0.38 & -0.0008 \\ 0.26 & -0.34 \end{bmatrix}, \mathbf{B}_4 = \begin{bmatrix} 0.7 & 0.0001 \\ 0.1 & -0.0024 \end{bmatrix} \tag{105}$$
$$\mathbf{C}_4 = \begin{bmatrix} 9.425 & 79.3 \end{bmatrix}, \mathbf{D}_4 = \begin{bmatrix} -0.42 & 0.56 \end{bmatrix}$$

Note that each region was linearized around the same pitch angle of 16° which correspond to the position of the blade with no inclination and is the center of the pitch angle range that goes from 0° to 32° . A thrust tracking LQR $K_i, i \in 1, 2, 3, 4$ controller was developed for each region Ω_i following the previously explained procedure. The switching between two controllers was chosen to be dictated by the change in the N2 core spool speed. Let's denote $\Theta_i = \{x_{e_i}^c, x_{e_i}^f, u_{e_i}^c, u_{e_i}^f, y_{e_i}\}$ the set of equilibrium points used for the linearization, the switching goes as follow,

$$(K_c, \Theta_c) = \begin{cases} (K_1, \Theta_1) & \text{if} \quad 0.86 < N_2 \\ (K_2, \Theta_2) & \text{if} \quad 0.62 < N_2 \leq 0.86 \\ (K_3, \Theta_3) & \text{if} \quad 0.42 < N_2 \leq 0.62 \\ (K_4, \Theta_4) & \text{if} \quad 0 < N_2 \leq 0.42 \end{cases} \tag{106}$$

Where (K_c, Θ_c) denote the controller gains and operating points to use to control the turboprop engine at a certain instant. The Gain scheduling procedure is summarized by Figure 25.

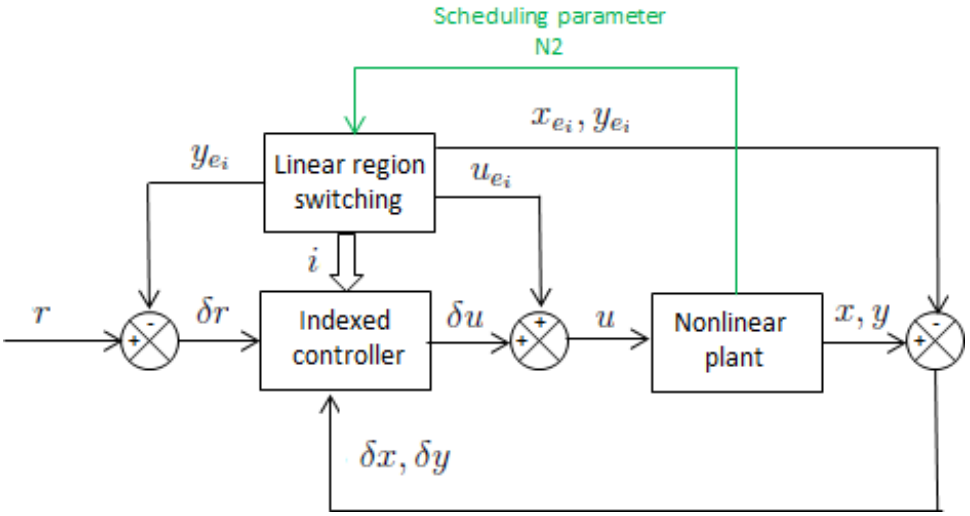


Figure 25: Gain scheduling procedure

A first set of simulations were performed to validate the feasibility of the gain-scheduling approach. The turboprop was controlled to track a thrust reference of 185 N starting from idle condition. As an initial test, no requirements were formulated on either state or input limitations, and the controller gains were just designed for stability and performance. The Thrust response of the system is shown on Figure 26.

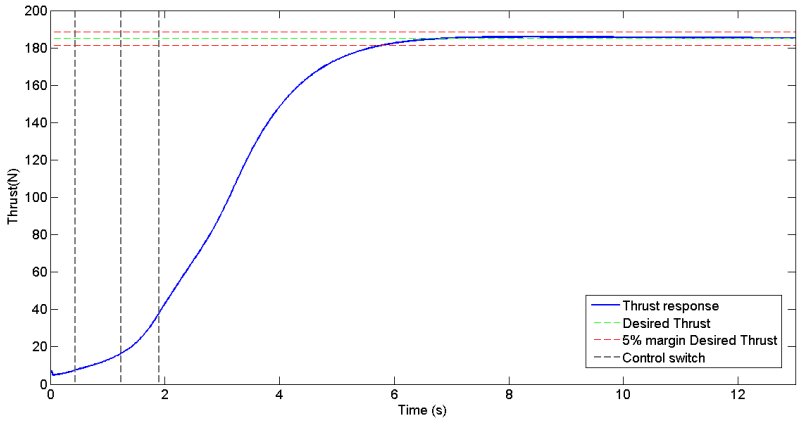


Figure 26: Gain scheduled control Thrust response

As seen on Figure 26, the controllers performances are satisfying as the system remain stable while switching between regions. Furthermore the thrust reaches steady

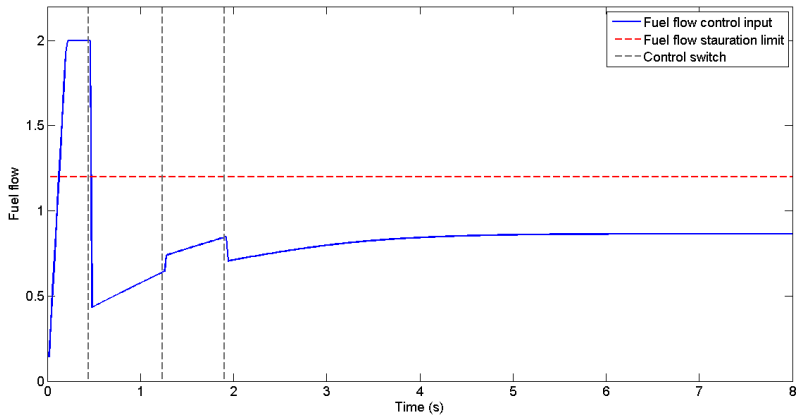
state with no overshoot in less than 8 seconds which is a very satisfying time. However, a satisfying thrust performance is not the only requirement of the closed loop system. A very important aspect of the controller is to make sure the system operates safely, meaning it can not exceed certain state limits such as a core spool speed and acceleration limits. If the turboprop operates above these state limits, the whole system could have a mechanical failure which would have catastrophic consequences when on flight. In addition to these states limits, the fuel flow is subjected to saturation which needs to be taken into account in the control law design.

Several important issues can be pointed out from Figure 27. The most important one is that under this specific control law the system exceeds the core spool speed limit which is a hard limit. It is also important to notice that because of the switching between controllers, the fuel flow control input is not differentiable. Furthermore, on the real system the fuel flow might be subjected to saturations, which were not taken into account on this first test.

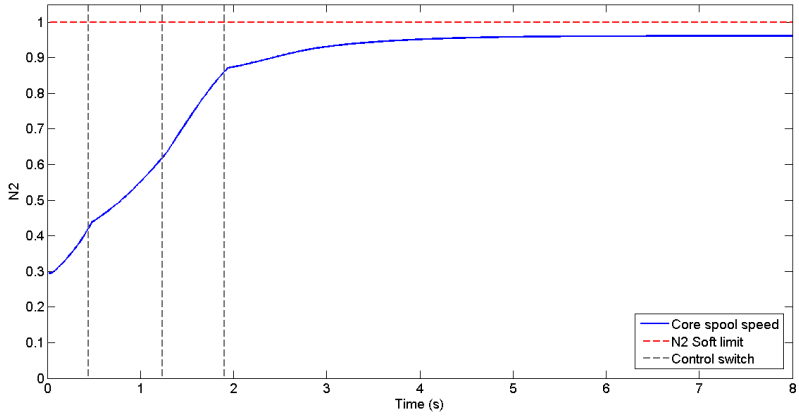
Taking care of the control saturation is not a hard problem, especially since the system is SISO. An anti-windup controller can be designed in order to avoid input saturations.

Also, to avoid having this non smooth control input, a much greater number of linear controllers could be used. By doing so, the transition between two linear controllers would be much smaller, which will get rid of the discontinuities in the control derivative. These controllers could be interpolated between the one already created and therefore would not require a long and tedious design.

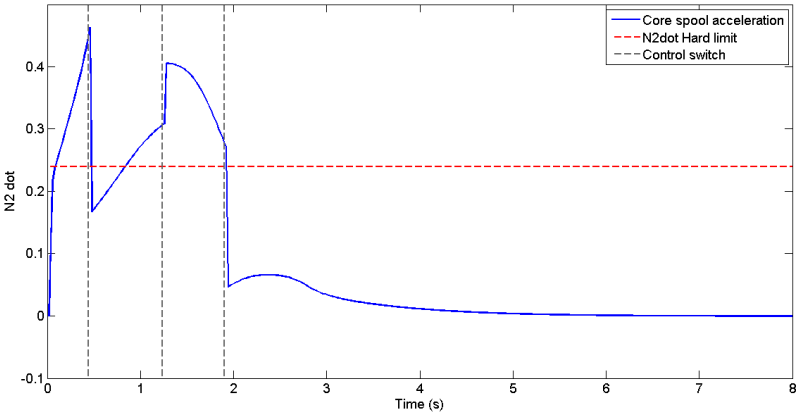
However, even neglecting these problems of non-smoothness and input saturation, designing a controller that would operate within the states limits is a complicated task and is a very common problem for turbofan engines. Several approaches could be considered to solve this problem, each varying in level of difficulty.



(a) Fuel flow control input



(b) core spool speed, N_2



(c) core spool acceleration, $\dot{N}_2$

Figure 27: Input and states history

A commonly used technique in industry, sometimes referred as min-max control strategy, is to design a set of controllers that would each take care of one state limit and have a switching mechanism that would alternate between each limit controllers and baseline controller.

For instance a controller could be designed to keep the $\dot{N}_2$ acceleration below its limit while disregarding the system performance or N_2 limit. The error between the current acceleration and $\dot{N}_2$ limit would be fed to this controller. A similar controller could be designed for the N_2 speed limit as well while the baseline controller will only care about the system performance. A switching mechanism will then alternate between the limit controllers and baseline performance controller to ensure stability of the turboprop while avoiding the states limits.

This technique was implemented first on the engine simulation by Mehrdad Pakhmer. The design of these controllers as well as the switching mechanism was a long and tedious task and was performed for the two control inputs operation of the turboprop. Also, even though the system operates safely under this specific control strategy, the performance remained poor and no stability proofs could be derived under this switching.

4.2.4 Control approach considered and simulation results

Another and simpler approach was chosen however by the author of this thesis to deal with this state limits problem. No stability proof however could be derived either, but the technique showed good performance and was tested to be relatively robust. Running the turboprop simulation, two important facts were noticed. First, if the fuel flow does not saturate the N_2 limit will most likely be respected and secondly

the $\dot{N}_2$ core spool acceleration was noticed to be closely related to the change in fuel flow as a jump in the fuel flow command generally induced a peak in the acceleration. Smoothing out the control command and making sure the control input does exceed a certain rate of change is therefore a way to avoid large accelerations.

Even though, using a large number of interpolated controllers would smooth out the control input; the control input rate of change could still be high and induce large acceleration. It was therefore decided to deliberately saturate the rate of change of the fuel flow in order to both smooth the control command and avoid large accelerations. The saturation rate was chosen according to which region the turbofan was operating in. For instance when close to idle, the saturation rate was allowed to be higher than when operating between idle and cruise. This was done so, as when the engine is close to idle the spool acceleration is not affected as much by the fuel flow rate as when operating at cruise as summarized by,

$$\dot{u}^f = \begin{cases} sat_{rate_1} & \text{if } \dot{u}^f > sat_{rate_1} \text{ and turboprop operates around idle} \\ sat_{rate_2} & \text{if } \dot{u}^f > sat_{rate_2} \text{ and turboprop operates above idle} \end{cases} \tag{107}$$

With Typically $sat_{rate_1} > sat_{rate_2}$. Figures 28 and 29 represent the thrust response and input and state history of the system under the gain-scheduled control law with deliberated saturations on the fuel flow.

The thrust response of the system is still vert satisfying as it reaches settling time in less than 8 seconds. Also as shown on Figure 29, none of the limits are violated. This was predictable for the $\dot{N}_2$ core spool acceleration but not necessarily for the N_2 and fuel flow limits. This is explained by the fact that by saturating the fuel flow rate of change, the input command is made slower and should not overshoot. Therefore if the desired steady-state operating condition is within N_2 state and fuel flow input limits, the system should not exceeds both of these limits under this saturated control law.

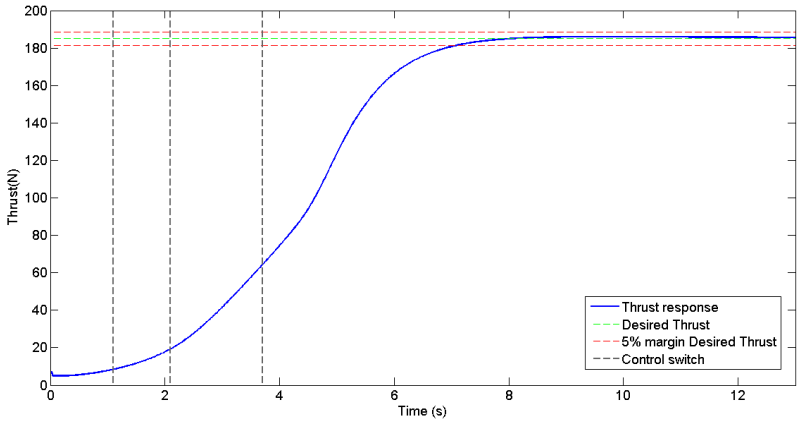
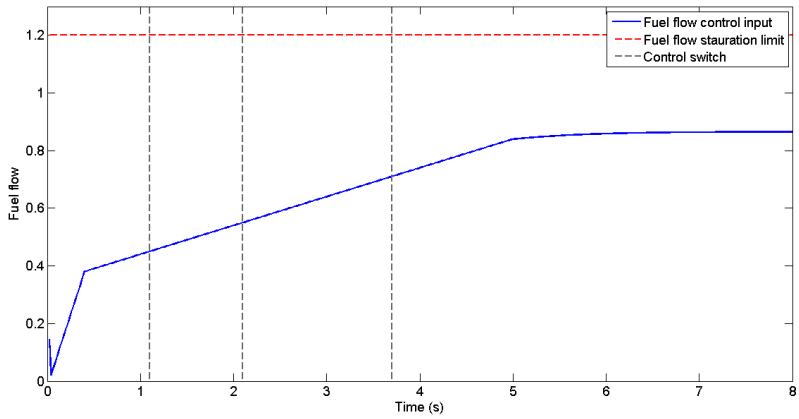


Figure 28: Gain scheduled control Thrust response with input rate saturation

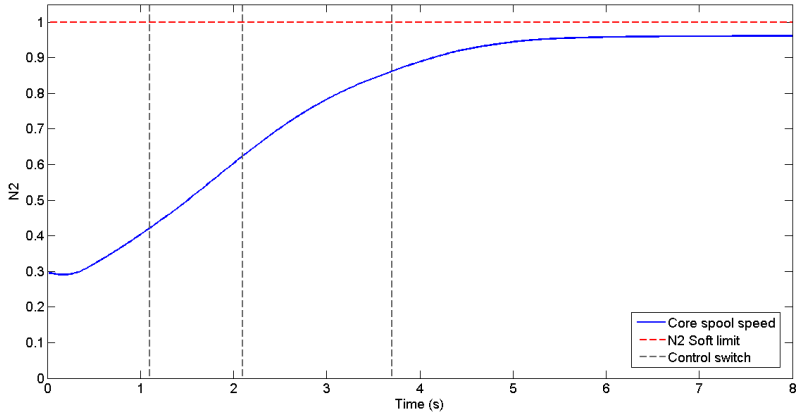
Despite showing good performance, this control technique has nos stability proofs. However through simulation it was tested to be robust, and work under different thrust conditions, model inaccuracies and external disturbances. One could argue that limiting the rate of the control input could destabilize the system if the engine needs to react quickly to a certain situation, to remain stable. However, the engine has a large inertia and its dynamics is very slow and this kind of situation where the engine has to react quickly do not exist. Therefore because of the engine's slow dynamics, this limited rate control input technique can be considered safe to be used on the engine.

4.3 Peak-seeking

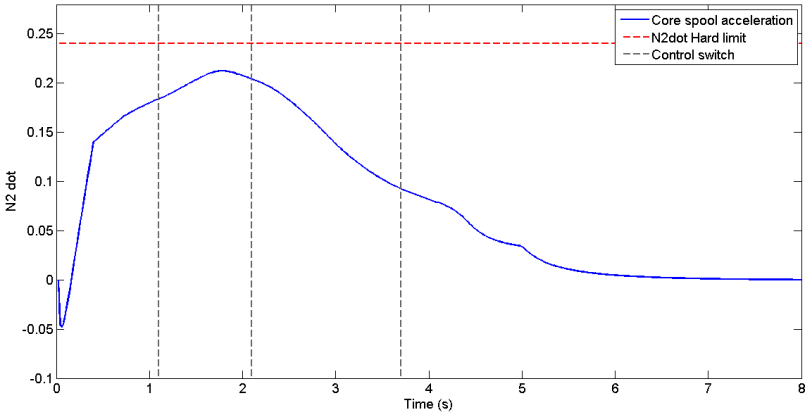
We will now present the Peak-seeking method applied to the Turboprop engine. We will first go over some analysis of the system and consider the different performance functions to minimize through peak-seeking. We will then see the updated Peak-seeking method compared to the one used for DC motor and finally go over the simulation results.



(a) Fuel flow control input



(b) core spool speed, N_2



(c) core spool acceleration, $\dot{N}_2$

Figure 29: Input and states history under input rate saturation

4.3.1 Turboprop analysis

Prior to developing the Peak-seeking method on the Turboprop, it is important to have a thorough understanding of the system behavior. It is first therefore important to choose an adequate performance function to minimize as well as to verify the system's convexity regarding this function in order to ensure the existence of a unique minimum.

The first performance function that comes in mind would be the steady state fuel consumption of the Turboprop at a certain thrust level. The fuel consumption can be directly affiliated to the cost of running the engine, and if optimized, could considerably reduce flight expenses. Another performance function that is in a more "long-term" objective, would be the engine's core temperature. Minimizing the temperature at which the engine is running could, over time, increase the operational lifetime of the engine, as high core temperatures generally leads to structural wearing. This is a non-exhaustive list of performance functions, as many more could be considered such as the N_2 spool speed, the spool torque, etc... each with their own advantages. For the purpose of assessing the feasibility of the Peak-seeking method, we will only present the work performed for the core engine temperature and fuel flow minimization, as these two problems are representative enough.

An off-line analysis of these performance functions was performed using the turboprop numerical simulation. The fuel flow and core temperature minimization problems are represented on Figures 30 and 31. Several other performances functions that could be considered to be minimized are also represented on Figure 32. These performance functions maps are represented in function of the thrust and propeller pitch angle, and are always convex when plotted in function of the propeller pitch angle only. This ensure the existence of a unique pitch angle minimum at any thrust level.

The minimum pitch angle location, regardless of the chosen performance function,

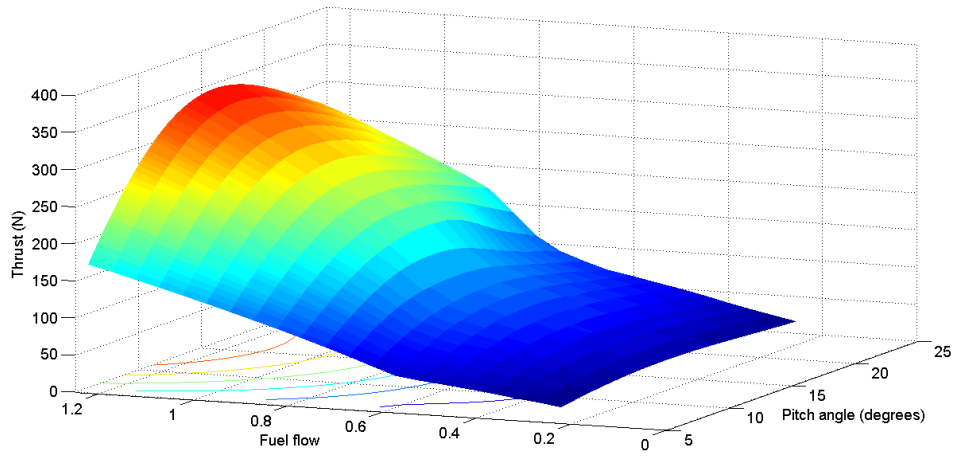


Figure 30: Fuel flow optimization problem

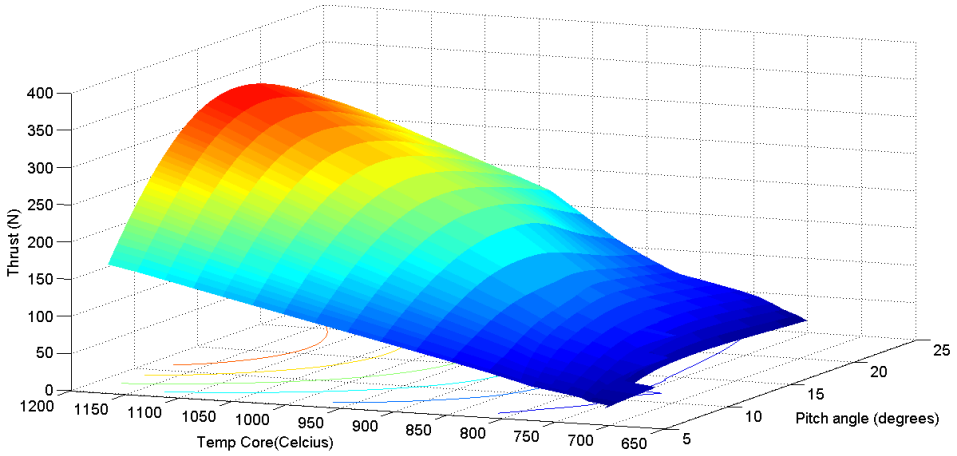
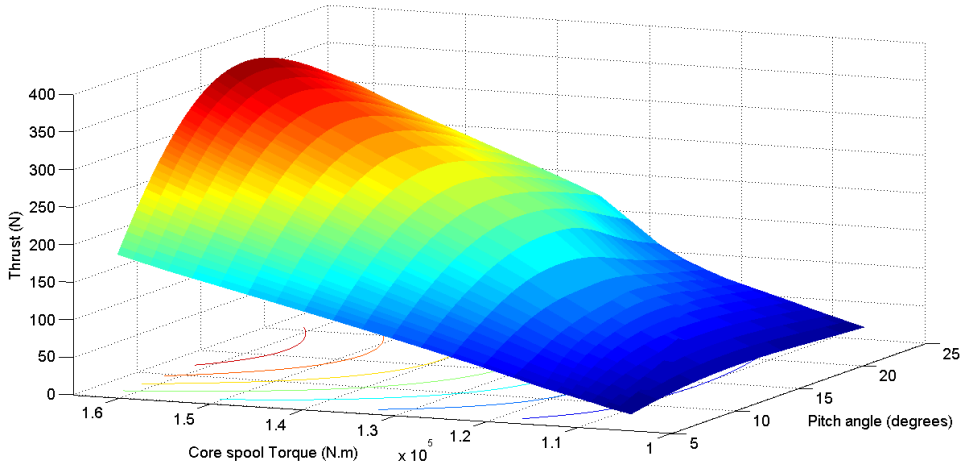
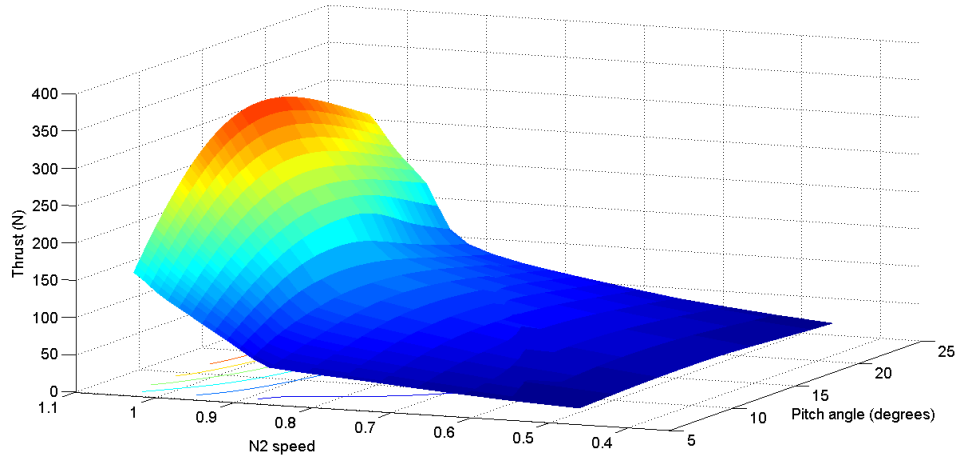


Figure 31: Core temperature optimization problem

depends also on the the thrust level. A simulation was run to determine the optimal pitch angle that will give the minimum fuel consumption for each thrust level, and is represented on Figure 33. Because the numerical simulation of the turboprop engine takes a long time to run, only few minimum pitch angle points were collected and had to be interpolated to illustrate the change in the minimum pitch angle location. As shown on 33 , the optimal pitch angle location can have a difference of almost 1.5° between two given thrusts. Therefore it would be wise to re-use the peak-seeking algorithm when the change in thrust command is important. However, it is also important to notice that when the performance function is plotted with respect to



(a) Torque



(b) N2 spool speed

Figure 32: Other possible performance functions to minimize

the propeller pitch angle, the function is generally flat around its minimum and leads to almost the same performance within 0.5° of the minimum. Therefore the optimal angle found through the Peak-seeking can be considered valid in a small region around the current Thrust, but has to be recomputed when encountering large changes in Thrust level.

4.3.2 Study case : Fuel flow optimization

As mentioned before, the fuel flow is probably the most interesting performance function to minimize as it could considerably reduce the cost of operating the engine. It

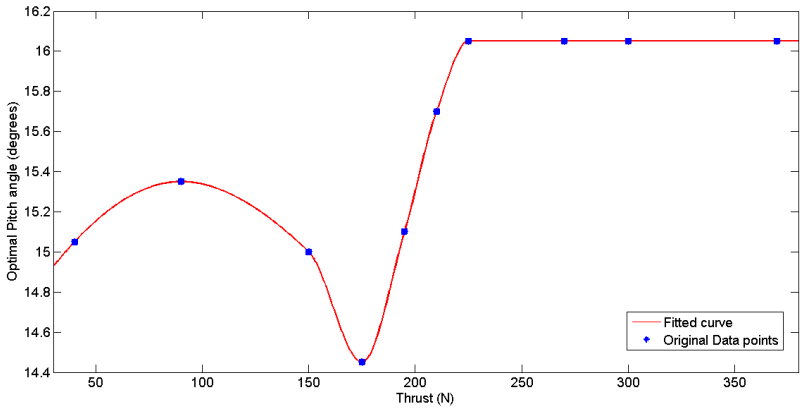


Figure 33: 'Fuel consumption 'Optimal pitch angle for different thrusts

also happens that it is the easiest and fastest performance function to minimize as the fuel flow is an input to the system and is not corrupted by noise since it is directly issued by the controller. The Kalman filter part of the Peak-seeking process is not therefore required and the online-optimization process can be represented by Figure 34.

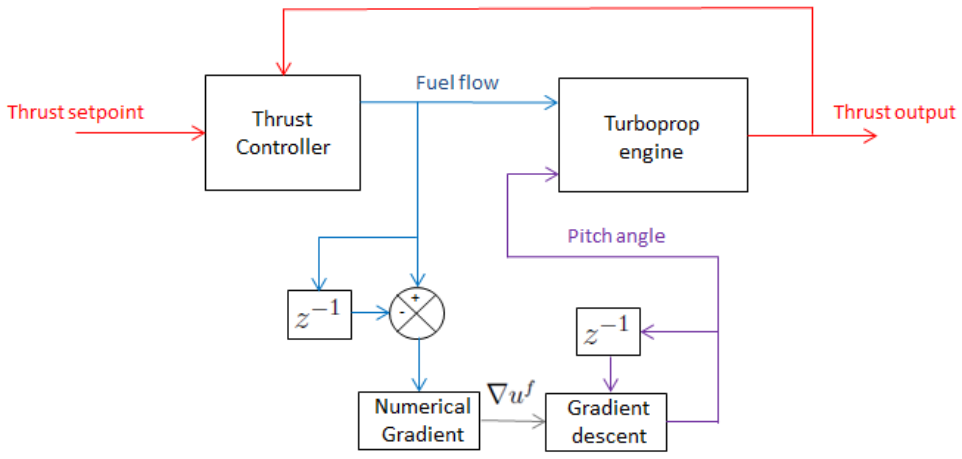


Figure 34: Fuel flow Peak-seeking

A Gradient descent method was preferred for the turboprop peak-seeking process as the numerical Hessian is much more sensitive to innacuracies. The Gradient descent method followed the same technique used for the DC motor using a fading term in the pitch angle update law as previously introduced in Equation 80. Also, because

the Gradient does not have to be estimated but instead can directly be calculated numerically , the Peak-seeking process is fast and only requires few steady-state points to converge to a minimum. This Peak-seeking algorithm was tested on the simulation while keeping the thrust around 185 Newtons. The fuel flow measurement was performed once the system reached steady state, which took approximately 8 seconds in average. The result of this specific fuel flow Peak-seeking process in represented on Figure 35.

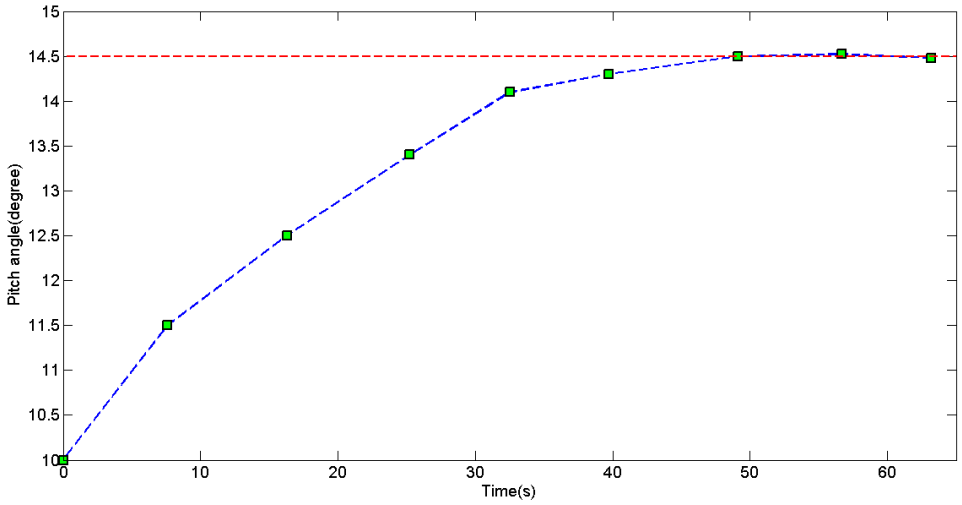


Figure 35: Blade pitch angle for fuel flow optimization over Time

Based on Figure 35 we can notice that the process converges in less than 50 seconds which is very fast considering the slow engine dynamics. This is partially due also by the fact that the gradient can be calculated instantaneously the beginning of the process instead of waiting for the Kalman filter to converge. Also as shown on the figure, the pitch angle was limited to a maximum change of 1.5° to limit the disruptive behavior of the pitch angle change on the thrust controller. Minimizing the fuel flow is therefore a relatively easy and fast process, however some performance functions may have to be measured and will be corrupted by noise. We will now see the Peak-seeking approach when dealing with such performance functions.

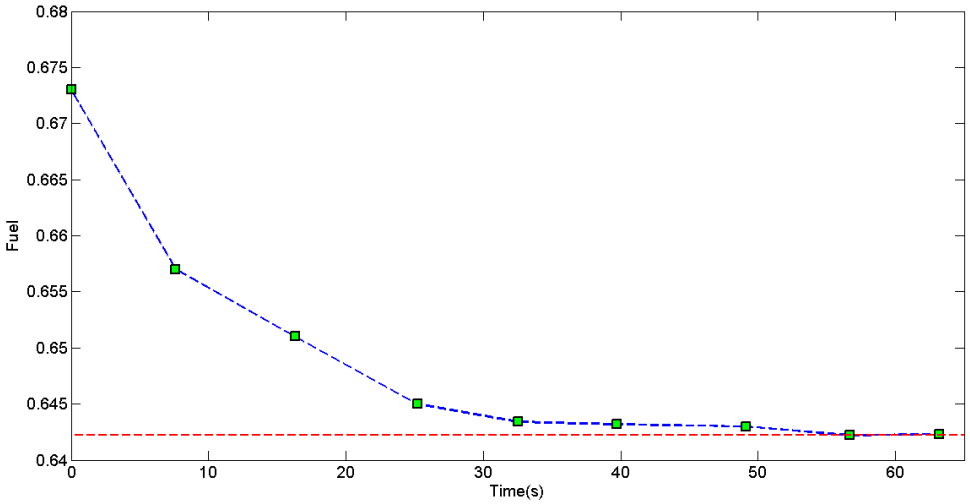


Figure 36: Fuel flow minimization over Time

4.3.3 Study case : Core Temperature optimization

As previously discussed, optimizing the core temperature would have the advantage of reducing the risk of structural damages on the long term. As for the fuel flow case, when operating at a constant thrust level, the core temperature function is always convex with regards of changes in the pitch angle. Using the turboprop simulation, the core temperature was determined for different pitch angle at a thrust level of 185 N, and is represented on Figure 37. A simulated noisy temperature , corrupted by white noise with a standard deviation of 0.5°C, was added to this plot to represent the supposed sensor measurement.

The Peak-seeking scheme is now represented on Figure 38. Because of the slow engine dynamics each steady state measurement takes around 8 seconds, it is therefore primordial to minimize the number of measurements in order to reach an optimal pitch angle in a decent time. No persistent excitation was therefore added for the turboprop peak-seeking as it involves taking more measurements and takes more time. Instead of adding extra measurements, the update law of the pitch angle was rethought to drive the pitch angle to a optimum, but also at the same time, increase the gradient

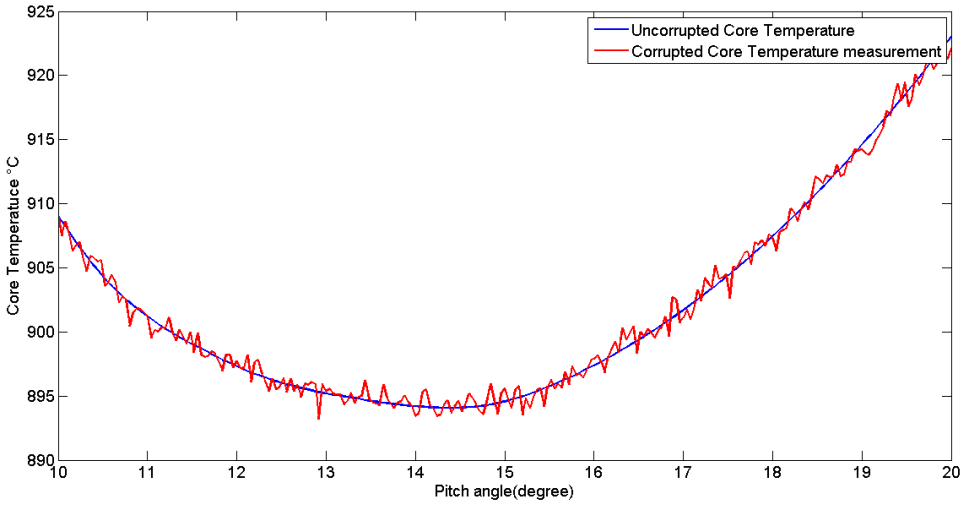


Figure 37: Actual and measured core temperature change for a constant thrust of 185N

observability and reduce the number of required steady state measurements. This is done through the "pitch step mechanism" block represented on Figure 38 and will be explained now.

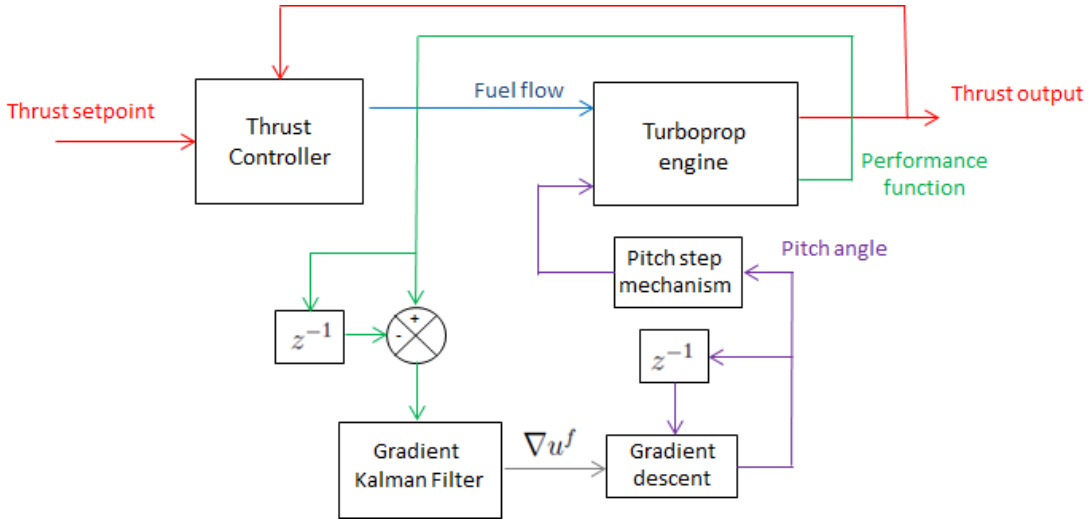
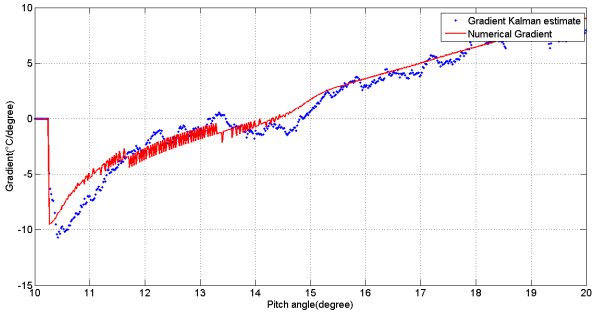


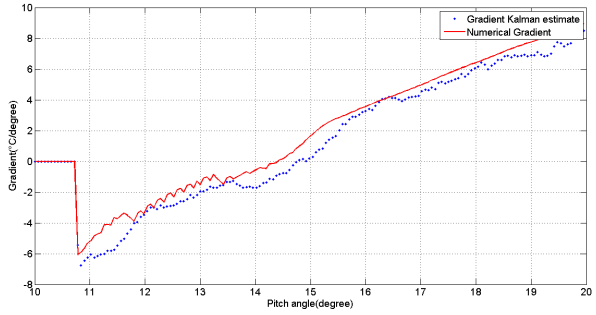
Figure 38: Peak-seeking structure for noisy performance function

The Kalman filter was designed as presented in DC motor part of this thesis, with the exception that equations 84 and 85 do not contain excitation terms. Also even though only the Gradient is used for the pitch angle update law, the Hessian and third

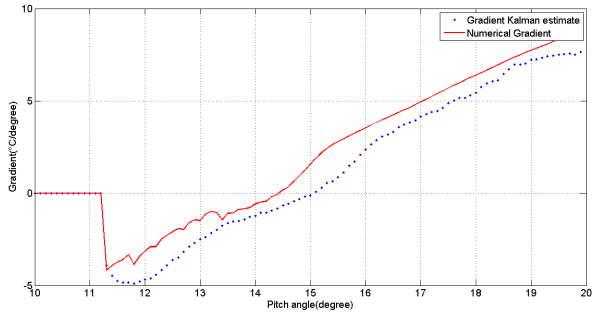
derivative are still used in the estimation process to improve the Gradient estimation quality. An important fact that was noticed for the DC motor case was that when taking large pitch angle steps between two measurement, the Gradient estimate would experience some "lag" compare to the true Gradient. Meaning that the estimated gradient behavior would be the same overall, but would follow the actual Gradient few degrees behind. This was explained by the fact that the derivative is estimated based on a Taylor's series model which is valid only in small region around the nominal pitch angle, and when large pitch angle steps are taken the Taylor approximation becomes less accurate. A test was performed to demonstrate the fact about the lag on the Gradient estimate. While staying at constant thrust of 185 N, a sweep of pitch angle was performed to cover the operating space of the engine. This sweep was performed with different pitch steps and the Kalman filter estimated the gradient of the performance function in each case. Only the last 10 measurements were taken into account for the derivative estimate, and the filtering process started only after a certain of measurements were collected. These results are represented on Figure 39. Since there is no analytical model of the turboprop, the estimated Gradient could only be compared to the Gradient with no noise. The small discontinuities on the numerical gradient, especially when using small pitch angle steps, are due to discrepancies in the turboprop code and could be disregarded.



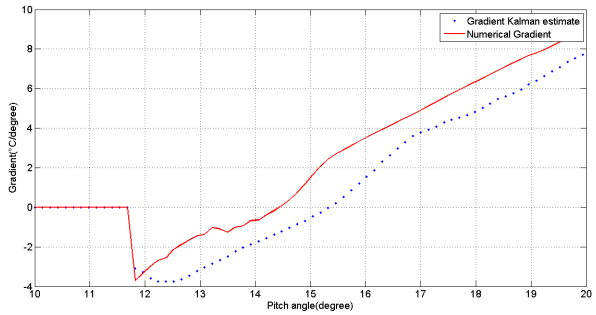
(a) pitch angle sweep of $\beta_k - \beta_{k-1} = 0.02$



(b) pitch angle sweep of $\beta_k - \beta_{k-1} = 0.06$



(c) pitch angle sweep of $\beta_k - \beta_{k-1} = 0.1$



(d) pitch angle sweep of $\beta_k - \beta_{k-1} = 0.15$

Figure 39: Gradient estimation for different pitch angle increments

We can clearly see on Figure 39 that the gradient estimate "lag" is proportional to the size of the pitch angle step taken. Taking a small pitch angle step will give an estimate close to the real gradient but which is more sensitive to noise, as only a small region is used to estimate the derivative. On the other hand, when large pitch angle steps are taken, the region covered is much larger and allows for a better measure of the function behavior and hence its derivatives. Therefore when large steps are taken the estimation is more accurate but biased. The "pitch step mechanism" was designed to combine the advantages of having both a non biased-estimate when using small steps and having a more accurate estimation when using large ones, while taking as few measurements as possible.

The mechanism works in a way that after taking some initial measurements, the filter first takes large steps to converge to the minimum as fast as possible. As the Gradient estimate gets closer to zero, the size of the these steps are reduced while the direction is still dictated by the Gradient descent. The number of measurements taken into account in the measurement matrix written in Equation 85 is also increasing at each iteration, to keep track on the history and allow for much better estimate. However, if the computational power available is limited, the number of measurements could be limited to a certain value; but the measurements will have to be chosen in a way to cover the current region of interest in detail and still have few data points further away, to successfully catch the function's behavior.

The Peak-seeking process was implemented on the simulation and following Figure 38. The Kalman filter was set to iterate as soon as a steady state measurement was available, which is approximately at 0.125Hz. For the purpose of this simulation, no computational limitation were set, and the filter was taking into account all of the previously made measurements at each iteration. The optimum pitch angle was initially guessed to be around 10 degrees, and the filter started working after taking 5 initial measurements around the first guess. Figure 40 shows the evolution of the

gradient estimate through the peak-seeking process, and helps visualizing the pitch step mechanism previously described. Figure 41 is as before, the evolution of the pitch angle over time under the peak-seeking update law.

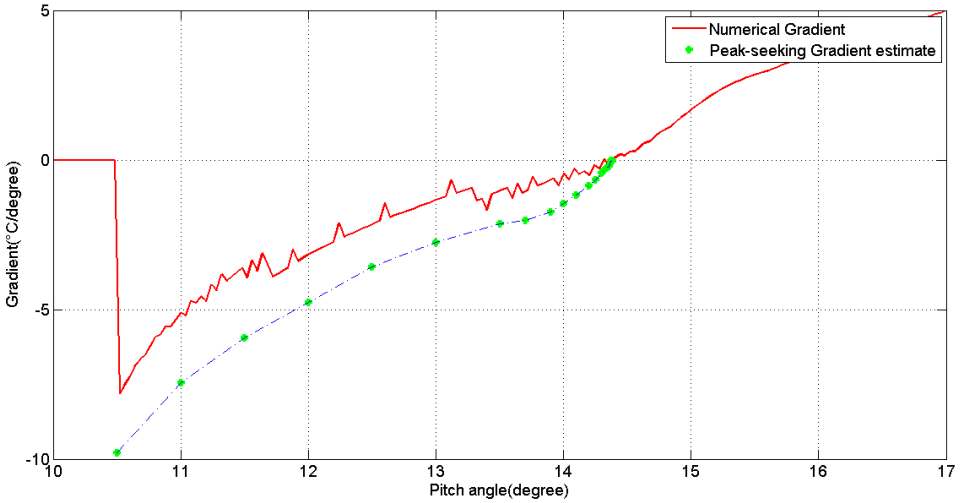


Figure 40: Gradient estimate through the Peak-seeking process

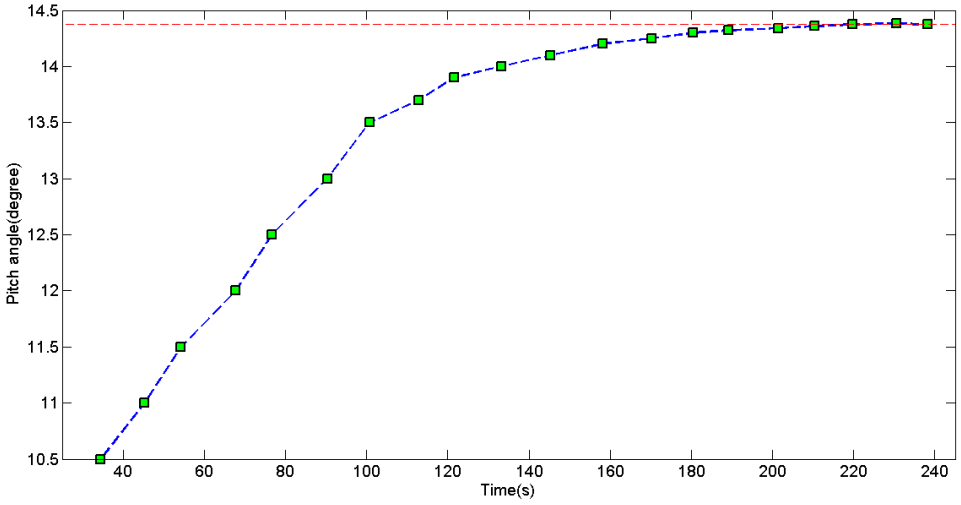


Figure 41: Pitch angle for Turboprop Core Temperature optimization over time

As shown on Figure 41 the process takes around 220 seconds to reach the optimum. Even though that this core temperature optimization takes more than three

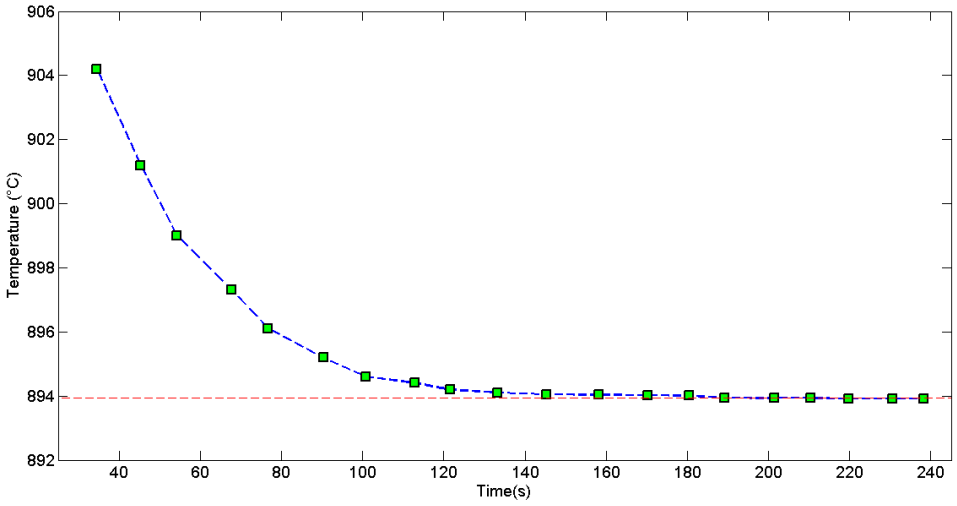


Figure 42: Turboprop Core Temperature optimization over time

times longer than the fuel flow optimization, it is a very satisfying performance considering the gradient needs to be estimated from noisy data. It took approximately 40 seconds for the system to initially collect the necessary measurements for the first estimate and approximately 20 other measurements after that to finally converge to the optimum.

In this specific case, the core temperature optimal pitch angle is really close to the fuel flow optimal one and is likely to be close for different thrusts level as well, as the fuel flow and core temperature are closely related. Meaning that a decrease in fuel flow will most likely induce a decrease in the core temperature. Furthermore, both the Core temperature and fuel flow functions are really flat around there minimum, meaning that even if not operating a the minimum, but close to it, the system can be considered to perform optimally. It is therefore most advantageous for the core temperature optimization, to actually optimize the fuel flow and follow the Peak-seeking process presented on Figure 34, as it is faster and requires less computation. However, the core temperature optimization process was presented in this thesis just as an example to demonstrate the feasibility of the Peak-seeking process when a noisy

performance function is used. Not all performance functions will lead to an optimum pitch angle close to the fuel flow one, and for these functions, the peak-seeking process as presented on Figure 38 should be used.

CHAPTER V

CONCLUSION AND FUTURE WORK

A peak-seeking method was developed to optimize propulsion systems in real time. The method is model-free and has been tested to be fast and reliable. The Peak-seeking process also allows modularity in the system's design and embraces the notion of "Plug and Play".

Only the propeller's blade pitch angle is controlled by the Peak-seeking process. The blade pitch angle is updated using either a Newton or a Gradient Descent method to minimize a predefined performance function. Both of these methods require the knowledge of the function's derivatives, which in turn requires the use of an extended Kalman filter.

The Peak-seeking was first analytically derived and simulated for a DC motor driving a variable pitch propeller. The method was then adapted to work on a turboprop engine simulation, and showed in both cases satisfactory results. The method was capable of optimizing the system in a short amount time despite noisy measurements and without any knowledge of the system.

Future work consists of testing the Peak-seeking on the turboprop engine as well as combining it with the work of Mehrdad Pakmehr which deals with designing a decentralized turboprop adaptive controller. Combining these two works would lead in the future to a fully modular engine.

REFERENCES

[1] K. B. Ariyur and M. Krstic, *Real-Time Optimization by Extremum-Seeking Control*, John Wiley and Sons, Hoboken, NJ, 2003.

[2] M. Krstic, Performance improvement and limitations in extremum seeking control, *Systems and Control Letters*, Vol. 39, No. 5, 2000, pp. 313-326.

[3] M. Krstic and H. H. Wang, Stability of extremum seeking feedback for general nonlinear dynamic systems, *Automatica*, Vol. 36, No. 4, 2000, pp. 595-601.

[4] R. V. Mises, *Theory of Flight*, McGraw Hill, New York, NY, 1945.

[5] C. S. Draper and Y. T. Li, *Principles of Optimalizing Control and an Application to the Internal Combustion Engines*, American Society of Mechanical Engineers Publication, New York, NY; Sep. 1951.

[6] Y. T. Li, Optimalizing System for Process Control, *Instruments*, Vol. 25; Jan.-Feb.-Mar. 1952.

[7] J. J. Ryan and J. L. Speyer, "Peak-Seeking Control Using Gradient and Hessian Estimates", *Proceedings of the 2010 American Control Conference*, Baltimore, MD, 2010, pp. 611-616.

[8] H. S. Tsien and S. Serdengecti, Analysis of Peak-Holding Optimalizing Control, *Journal of the Aeronautical Sciences*, Aug. 1955, pp. 561-570.

[9] D. F. Chichka, J. Speyer and C. G. Park, "Peak-seeking control with application to formation flight", *Proceedings of the 38th IEEE Conference on Decision and Control*, Phoenix, AZ, December 1999, pp. 2463-2470.

[10] P. Binetti, K. B. Ariyur, M. Krstic and F. Bernelh, Formation flight optimization using extremum seeking feedback, *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 26, January-February 2003, pp. 132-142.

[11] D. Chichka, J. Speyer, C. Fanti, and C. Park, Peak-Seeking Control for Drag Reduction in Formation Flight, *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 5, September-October 2006, pp. 1221-1230.

[12] H.-H. Wang, M. Krstic, and G. Bastin, Optimizing Bioreactors by Extremum Seeking, *International Journal of Adaptive Control and Signal Processing*, Vol. 13, 1999, pp. 651-659.

[13] Z.-H. Li, *Optimal Lyapunov Design of Robust and Adaptive Nonlinear Controllers*, PhD thesis, University of Maryland, College Park, 1997.

[14] H. Yu, U. Ozguner, "Extremum-seeking control strategy for ABS system with time delay", *Proceedings of the 2002 American Control Conference*, Anchorage, AK, May 2002, pp. 3753-3758.

[15] K. B. Ariyur, M. Krstic, "Slope Seeking and Application to Compressor Instability Control", *Proceedings of the IEEE Conference on Decision and Control*, Las Vegas, Nevada, 2002, pp. 3690-3697.

[16] H. H. Wang, S. Yeung, M. Krstic, Experimental Application of Extremum Seeking on an Axial-Flow Compressor, *IEEE Transactions on Control Systems Technology*, Vol. 8, No. 2, 2000, pp. 300-309.

[17] K. B. Ariyur, M. Krstic, "Analysis and Design of Multivariable Extremum Seeking", *Proceedings of the 2002 American Control Conference*, Anchorage, AK, May 2002. pp. 2903-2908.

[18] A. Banaszuk, K. B. Ariyur, M. Krstic and C. A. Jacobson, An Adaptive Algorithm for Control of Combustion Instability, *Automatica*, Vol. 40, No. 11, Nov. 2004, pp. 1965-1972.

[19] A. Banaszuk, Y. Zhang, C. A. Jacobson, "Adaptive Control of Combustion Instability using Extremum-Seeking", *Proceedings of the 2000 American Control Conference*, 2000, pp. 416-422.

[20] Stilwell, D.J.; Rugh, W.J.; , "Interpolation methods for gain scheduling," *Proceedings of the 37th IEEE Conference on Decision and Control*, vol.3, no., pp.3003-3008 vol.3, 1998

[21] Rugh, Wilson J.; , "Analytical Framework for Gain Scheduling,"*American Control Conference*, 1990 , vol., no., pp.1688-1694, 23-25 May 1990

[22] Shamma, J.S.; Athans, M.; , "Analysis of gain scheduled control for nonlinear plants," *Automatic Control, IEEE Transactions* on , vol.35, no.8, pp.898-907, Aug 1990

[23] Zhou,K.; Doyle,John.; Glover,K.; "Robust and Optimal Control", 1996

[24] Tsiotras, P.; "AE6511 Robust Control class notes", 2010

[25] Welch, G ; Bishop,G "An introduction to the Kalman Filter", 2006

[26] S. van der Meulen, B. de Jager, F. Veldpaus, M., Steinbuch, "Combining Extremum Seeking Control and Tracking Control for High-Performance CVT Operation", *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, GA, December 15-17, 2010.

[27] D. Popovic, M. Jankovic, S. Magner, and A. Teel, Extremum seeking methods for optimization of variable cam timing engine operation, *IEEE Transactions on Control Systems Technology*, Vol. 14, No. 3, 2006, pp. 398407.

[28] R. E. Klein, "A derivative extremal adaptive controller with applications to rotating power systems", *IEEE Symposium on Adaptive Processes (9th) Decision and Control*, 1970.

[29] Wikipedia

[30] M. Pakhmer, N. Fitzgerald, J.Paduano, E. Feron, A. Behbahani "Dynamic Modeling of a Turboshaft Engine Driving a Variable Pitch Propeller: a Decentralized Approach" *AIAA Joint propulsion conference and Exhibit* San Diego, CA, 2011

[31] M. Pakhmer, N. Fitzgerald, J.Paduano, E. Feron, A. Behbahani "Distributed Modeling and Control of Turbofan Systems" *AIAA Guidance, Naviguation and Control conference* Chicago, Illinois 2009

[32] T. Cazenave , M. Pakhmer, E. Feron "Peak-seeking control of a DC motor driving a variable pitch propeller" *AIAA Guidance, Naviguation and Control conference* Portland, Oregon 2011