LPV MODELING OF A TURBOFAN ENGINE

Luc Reberga [*] **Didier Henrion** [*,***]
Jacques Bernussou [*] **Florian Vary** [**]

[*] *LAAS-CNRS, 7 av. Colonel Roche, 31077 Toulouse, France*
[**] *Snecma Moteurs, Centre de Villaroche, 77550 Moissy-Cramayel, France*
[***] *Faculty of electrical Engineering, Czech Technical University in Prague, Technická 2, 16627 Praha, Czech Republic*

Abstract: This article describes the application of quasi-LPV modeling techniques to an industrial military turbofan engine simulator developed by Snecma Moteurs, the French aerospace propulsion company. Two approaches are used, classical Jacobian linearization and velocity based linearization. First, we present briefly the theoretical aspects of both linearizations. Second, we describe practical implications and limitations. Finally, we present the application of those techniques to the Snecma Moteurs Matlab Simulink turbofan engine simulator. *Copyright ©2005 IFAC.*

Keywords: Turbofan engine, modeling, linear parameter varying (LPV), linearization.

1. INTRODUCTION

Improving control policies for turbofan engines still represents a challenge nowadays. Major objectives are reduced development costs and improved in-flight engine performance (Skogestad and al, 1996), (Wolodkin and al, 1999), (Grimble, 2001). One of the first difficulties faced by control engineers is the lack of satisfactory analytic models for turbofan engines. Currently, the most-promising control technique may be gain scheduling achieved by interpolating controllers synthesized at different linearized operating points throughout the flight enveloppe, see (Wolodkin and al, 1999), (Skogestad and al, 1996), (Nobakhti and Munro, 2002). Sometimes, the controller is not scheduled at all, and just designed at one operating point. It is then verified afterwards if the controlled system is robust enough to parameters variations (Wang and al, 1999). If not, a precompensating gain is scheduled to adapt the controller to parameters variations.

The key ingredient for these elaborated control strategies is the choice of the synthesis model. During the last ten years, a new modeling framework has been developed for gain scheduling, resulting in the so called LPV (Linear Parameter Varying) models, (Packard, 1994), (Apkarian and Gahinet, 1995). This method offers a theoretical framework to ensure stability, performance and/or robustness of the controlled system via convex optimisation over LMI (Linear Matrix Inequalities), (Boyd and al, 1994).

When LPV design is chosen for control synthesis, an LPV description of the plant is needed. Since no satisfying analytical model of turbofan engine is available, the synthesis model must be found numerically. In (Henrion and al, 2004), we described some techniques to obtain local linearised models around operating points. This paper is aimed at showing how to obtain an LPV model based on local linearized models. This paper reports research carried out during the second phase of a three-year contract between the French aerospace propulsion company Snecma Moteurs and LAAS-CNRS. The outcome of the first phase, the derivation of linearized models, was published in (Henrion and al, 2004). The third and last phase, the design of control laws based on this LPV model, is the subject of current research and will be published elsewhere.

The outline of this paper is as follows. In section 2, we present two ways to obtain a global model based on local models. In section 3, we show how the general principles developed in section 2 can be applied. This method is then applied in section 4 on a military turbofan engine simulator developed by Snecma Moteurs.

2. DERIVATION OF LPV MODEL

The goal of linearisation is to produce linear models of the local behaviour of a nonlinear plant formally described by:

$$\begin{aligned} \dot{x} &= f(x,u,\theta) \\ y &= g(x,u,\theta) \end{aligned} \tag{1}$$

with x the state of the system, u the control input, θ the exogenous input. Functions f and g are assumed differentiable along any possible trajectory of the nonlinear system.

We can distinguish between two major cases:

- analytical expressions of f and g in (1) are available;
- System (1) is available only through a simulator or data from a test bench.

Two approaches can be used to build the local linear models:

- Linearization by state and input perturbations, if the state is accessible (Leith and Leithead, 2000)
- Identification from input-output data (Henrion and al, 2004).

Both techniques lead to a familly of linear models:

$$\begin{aligned} \dot{x} &= A_{loc}\,(x-x_0) + B_{loc}\,(u-u_0) \\ &\quad + P_{loc}(\theta-\theta_0) \\ y-y_0 &= C_{loc}\,(x-x_0) + D_{loc}\,(u-u_0) \\ &\quad + Q_{loc}(\theta-\theta_0) \end{aligned} \tag{2}$$

with A_{loc}, B_{loc}, C_{loc}, D_{loc}, P_{loc} and Q_{loc} constant matrices of appropriate sizes approximating the behaviour of system (1) around the linearization point (x_0, u_0, y_0). When not stated otherwise, chosen linearization points are equilibrium points.

Assumption 1. θ varies slowly so that both $\theta - \theta_0$ and $\dot{\theta}$ can be neglected.

Assumption 2. θ is measurable.

Thanks to assumption 1, P_{loc} and Q_{loc} can be neglected. Assumption 2 ensures we can schedule the controller with respect to θ.

The next step is to create a global linear parameter varying (LPV) model of system (1) based on a family of local models (2). We see two ways to tackle the problem:

- A classical one based on Taylor series expansion, also called Jacobian linearization;
- A second method proposed by (Leith and Leithead, 1998), perhaps less known, called "velocity based linearization", based on time derivatives of system (1).

The final aim is to get a model of the LPV form:

$$\begin{cases} \dot{x} = A(\theta)\,x + B(\theta)\,u \\ y = C(\theta)\,x + D(\theta)\,u \end{cases} \tag{3}$$

where parameter θ belongs to some compact, bounded set Θ such that all trajectories of system (1) are covered by possible trajectories of LPV system (3).If θ contains elements of x, the system is called quasi-LPV.

Such models are conservative because many of the possible trajectories of the LPV model cannot be trajectories of the nonlinear model. But they provide a constructive framework for nonlinear system controller synthesis.

2.1 Jacobian Linearization

When applied to the system (1) around a point (u_0, x_0) and under assumption 1, the Taylor series expansion theory yields the model:

$$\begin{aligned} &\dot{x} = \dot{x_0} + \dot{\Delta x} = f(x_0,u_0,\theta_0) + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial u}\Delta u \\ &\Delta x = x - x_0 \\ &\Delta u = u - u_0 \end{aligned} \tag{4}$$

This is the classical way to linearize a nonlinear system, also called Jacobian linearization. Note that in general model (4) is affine due to the term $f(x_0, u_0, \theta_0)$ in the expression of $\dot{x}$. For practical purpose, (u_0, x_0) is chosen as an equilibrium point $(f(x_0, u_0, \theta_0) = 0)$ so that an LTI (Linear Time Invariant) model of the plant is obtained in the neighborhood of the equilibrium point.

2.2 Velocity-based Linearization

The idea here is to differentiate equation (1) to get:

$$\begin{cases} \ddot{x} = \dfrac{\partial f(x,u,\theta)}{\partial x}\dot{x} + \dfrac{\partial f(x,u,\theta)}{\partial x}\dot{u} \\ \dot{y} = \dfrac{\partial g(x,u,\theta)}{\partial x}\dot{x} + \dfrac{\partial g(x,u,\theta)}{\partial x}\dot{u} \end{cases} \tag{5}$$

which is equivalent to

$$\begin{aligned} \dot{z} &= A(x,u,\theta)z + B(x,u,\theta)\dot{u} \\ y &= [0 \quad 1]z \\ z &= \begin{bmatrix} \dot{x} \\ y \end{bmatrix} \\ A(x,u,\theta) &= \begin{bmatrix} \dfrac{\partial f(x,u,\theta)}{\partial x} & 0 \\ \dfrac{\partial g(x,u,\theta)}{\partial x} & 0 \end{bmatrix} \\ B(x,u,\theta) &= \begin{bmatrix} \dfrac{\partial f(x,u,\theta)}{\partial u} \\ \dfrac{\partial g(x,u,\theta)}{\partial u} \end{bmatrix} \end{aligned} \tag{6}$$

Hence, we get systematically a quasi-LPV system whereas the technique of section 2.1 yields an affine system. The price to pay is an extension of the state, which now incorporates both $\dot{x}$ and y. Hence, the order of the model is the order of the nonlinear system plus its number of outputs.

3. IMPLEMENTATION

The linearization techniques of section 2 suggest different implementations depending on the chosen interpolation technique. Let us enumerate the advantages and the drawbacks of each method.

3.1 Jacobian Implementation

As seen on figure 1, u_0 and y_0 are inputs to the system and a strategy must be adopted to set them. A basic way is to switch from one equilibrium point (u_0, y_0) to another depending on the output y of the model. That implies interpolation between *equilibrium points*, e.g. by a piecewise constant function. A limitation of this technique is that it results in an algebraic loop $y_0 = g(y)$ in simulation for most of the interpolation choices.

Of course, another important choice is the way to interpolate the local models. It is our experience that the piecewise linear interpolation implemented in the `look-up table` block of Simulink (The MathWorks Simulink, 2002) gives excellent results in simulation but is not convenient for synthesis. In practice, the synthesis model uses polytopic representation or rational interpolation within the framework of Linear Fractional Transforms (LFT) (Packard, 1994). Since interpolation means loss of knowledge about the system anyway, it is difficult to know a priori which interpolation policy is more appropriate. Indeed, the set of trajectories of nonlinear system (1) can be included in the set of trajectories of a quasi-LPV model obtained by mathematical manipulations. But when the LPV model is obtained by interpolation over local models, nothing ensures that this inclusion is preserved.

3.2 Velocity based linearization Implementation

One of the advantages of velocity based linearisation is the use of off-equilibrium points. If we were able to linearise around these points, we could include them in the modeling process to enhance the model quality, without changing its properties (its linearity).However without the knowledge of an analytical model, we have no way to derive off-equilibrium models to improve the quality of the model. It means that the quasi-LPV model we obtain is only valid around equilibrium points as in the Jacobian approach.

The velocity based linearization leads to an LPV model:

$$\begin{cases} \dot{x} = x_2 \\ \dot{x_2} = A_{loc}(x,u,\theta)x_2 + B_{loc}(x,u,\theta)\dot{u} \\ \dot{y} = C_{loc}(x,u,\theta)x_2 + D_{loc}(x,u,\theta)\dot{u} \end{cases} \tag{7}$$

with x is the former state, x_2 and y the new model state, θ the exogenous input and A_{loc}, B_{loc}, C_{loc}, D_{loc} are as in equation (2).

Here, the problem due to interpolation of equilibrium linearization points disappears avoiding algebraic loops. The same remark as in section 3.1 can be made about interpolation of the local models and the lack of representativity of the quasi-LPV model (no link between the two sets of state trajectories).

Equations (2) imply that the output of the controller is $\dot{u}$ so this signal must be integrated to retrieve the input u sent to the real plant. Equations (2) mean also that the global behaviour of the system is the integral of the local one. Hence modeling errors are integrated too, generating a static error in contrast with the Jacobian linearization.

4. APPLICATION

4.1 Description of the plant

Our system is a numerical simulator of a military turboreactor designed by Snecma Moteurs. To be more precise, this turbofan engine is a twin spool turbofan with separate jets and a low bypass ratio.The two main inputs are WF32 (fuel flow) and A8 (nozzle area). The five main outputs are XN25 (compressor speed), XN2 (fan speed), PS32 (pressure in combustion chamber), DPQ23 (pressure ratio between high pressure (HP) spool

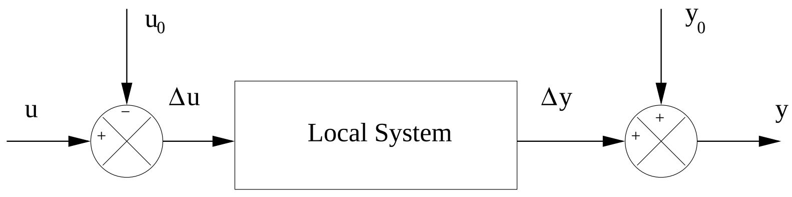

Fig. 1. Classical linearization scheme

and low pressure (LP) spool) and TM49 (metal temperature of HP turbine). The exogenous signal (eventually to be scheduled) are the Mach number, the altitude and the PLA (Pilot Lever Angle). As we have no analytic models of the system, we identified a family of linearised models along the working line as described in (Henrion and al, 2004). The point is now to get a global model of the plant. We formulate two key assumptions: first, the plant nonlinearity can be modeled by an LPV system; second, the model is linear in the input signal.

4.2 Implementation and comparison

We choose to present our results on the transfers from A8 and WF32 to PS32. The reason of this choice is that PS32 is the best known scheduling parameter and it is the only state-dependent parameter we use. It would be useless to test the models on a more complicated system. Hence, we have identified a two-input-one-output transfer of order 2 for 18 differents PLA from idle to maxthrust (without afterburning) on the ground (altitude 0) and for a Mach number of 0.3. Recall the two assumptions 1 and 2 to be made about this system to allow LPV modeling:

- PS32 shall change slowly.
- Since $\theta = y$, θ is measurable.

Both Jacobian linearization and velocity based linearization require interpolated local models to provide an LPV system. Since θ is the output, the system is quasi-LPV. We tested several methods to interpolate the matrix coefficients. We obtained the best results with a polynomial interpolation of degree 3 and a 3-segment piecewise linear interpolation. Each of these approaches leads to a different controller synthesis:

- polynomial interpolation requires the LFT framework;
- piecewise linear interpolation yields polytopic models.

We can indifferently use these two interpolation methods with our two linearization methods.

4.2.1. Jacobian Linearization We choose the simplest way to interpolate our equilibrium points: a piecewise constant interpolation, which corresponds to switching between operating points depending on the scheduling variable (here, PS32). Three variables are scheduled that way: u_0, x_0 and y_0. The values for each operating points are obtained by identification but the values of x_0 yield impulsive modes during transitions between two operating points. So we had to schedule x_0 to ensure the continuity of y which increased greatly the quality of the response. We chose to schedule u_0, x_0 and y_0 that way for the simple reason that it avoids algebraic loops.

First, we tested small step responses to compare the numerical simulator and the Jacobian linearised model. We got very good results because assumption 1 was respected. Then, we simulated an acceleration from idle to maxthrust followed by a deceleration from maxthrust to idle. This test is one of the most demanding the model must perform, because of the speed of variation. In particular, assumption 1 does not hold anymore. This causes problems on simulations for acceleration and even more serious problems for deceleration because of faster dynamics.

With Jacobian linearisation, the first problem encountered in simulation was the existence of algebraic loops that Simulink could not handle. This is particularly true when using local models with a non-zero feed through D matrix. As a remedy, it is possible to enforce a transfer function with large bandwidth between θ and PS32. The larger the bandwidth, the more accurate but also the more demanding is the simulation. Another way to cope with those simulation limitations is to soften the way the matrices are interpolated. The first interpolation we tried was a piecewise linear interpolation (joining all the linearizing points). As one can see on figure 2, such an interpolation is very irregular. To soften it, we used the two interpolations defined in section 4.2 and represented on figure 2, our operating points interpolated in the least squares sense. This has two advantages: first, it eases the calculation of algebraic loops for Simulink, second, it gives simpler expressions of the model, leading to simpler controllers.

Remark 1. It is important to take care about what happens when θ goes outside the range of known values. The easiest way to solve the problem is to put a saturation on θ. In other

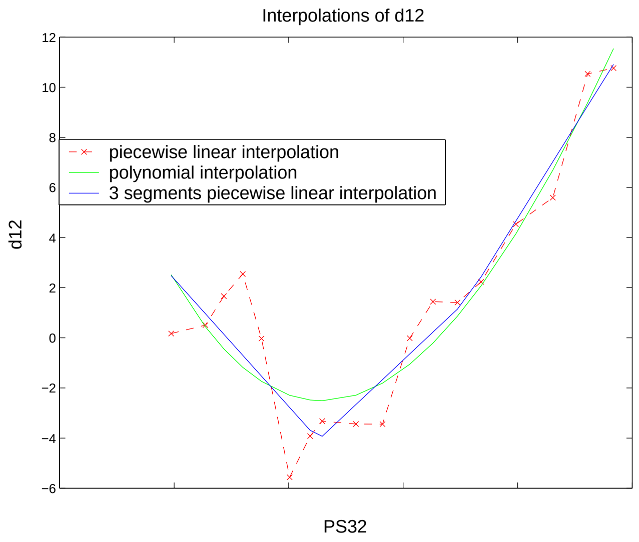

Fig. 2. Different interpolations of the (1,2) coefficient of feed through D matrix

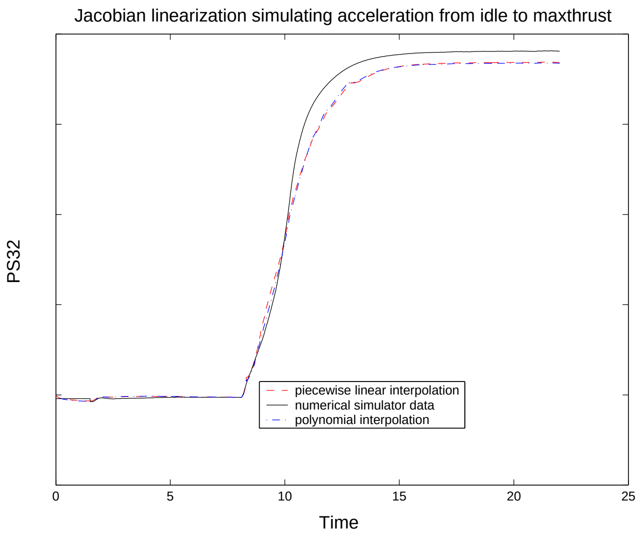

Fig. 3. Comparison between piecewise and polynomial interpolation for Jacobian linearization

words, under and over the known range, the scheduling function is constant.

Figure 3 shows the result of Jacobian linearization. It shows in particular that both interpolating functions give quite the same result. Also the behaviour of the plant is correct but not perfectly reproduced, probably because of the poor quality of some identifications and the necessary non respect of assumption 1. These results show that the controller must be robust to modeling errors. It is interesting to note that the static error only depends on the quality of the identification of the plant since at steady state, the model behaves just like the local model around the equilibrium point.

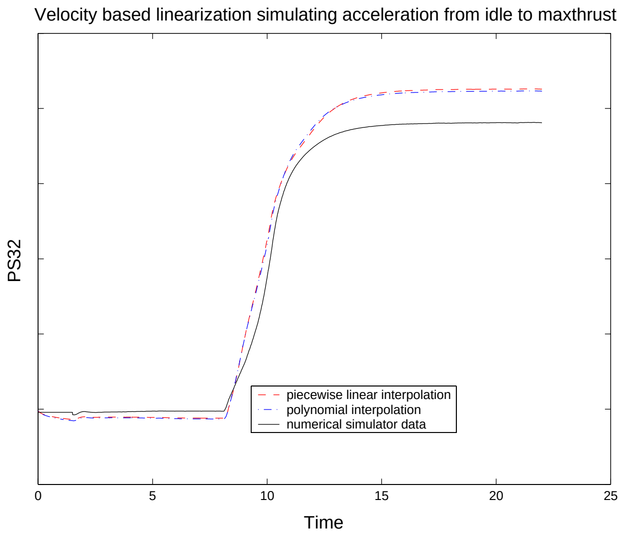

Fig. 4. Comparison between piecewise and polynomial interpolation for velocity based linearization

4.2.2. Velocity Based Linearization The same simulations are performed as for the Jacobian linearization, using the same interpolation points.

We see on figure 4 that both interpolations give the same result: on the one hand, the static error is important (as expected, see section 3.2), and on the other hand, the model seems representative of the dynamics. As for the Jacobian implementation, these results show that we will have to synthesize a very robust controller. But for velocity based linearization, there is already an integrator in the loop (as explained in section 3.2) ensuring good tracking properties.

Remark 2. For simulation needs, it is possible to change the poles of the integrator into Hurwitz-stable poles thanks to another loop described on figure 5 so that the static error vanishes. The price is to interpolate y_0 as a function of the input u. The chosen poles must be slower than the system dynamics. Otherwise the system will follow equilibria curve irrespective of local dynamics.

4.3 Conclusion

We have seen that the quasi-LPV models obtained with our approach are quite representative of the plant even if the static gain is not always well reproduced. It does not seem to be a major problem because our controller will ensure appropriate tracking. The results presented here also show that both linearization methods yield plant models of comparable quality. This allows us to choose any of those two models to the next step, the synthesis, and the good properties of velocity-based models lead us to choose them for the controller design step.

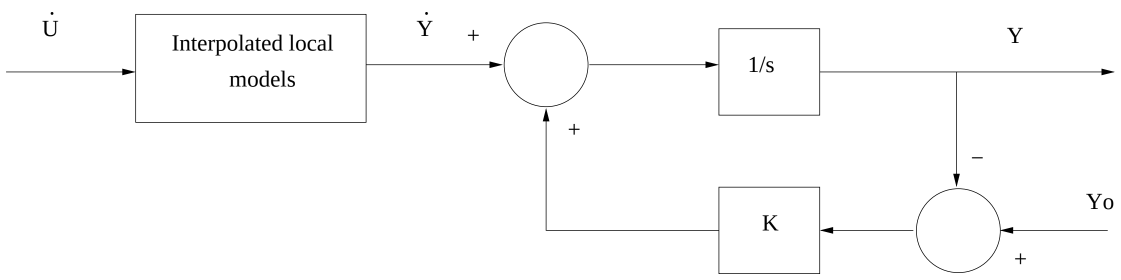

Fig. 5. Add to the velocity based implementation to suppress the static error.

5. CONCLUSION

We have compared two different approaches for quasi-LPV systems modeling. We have emphasized the advantages and shortcomings of each technique, discussing their potential applications to controller design. Indeed, we hope that the better the model fits the system response (especially the transients), the more efficient the controller design will be. Nevertheless, the choice of the modeling technique depends on the most important closed-loop specifications. Indeed, some modeling errors will not impact on the quality of the control of the plant. As an example, the integrator in the open loop due to velocity based linearisation will remove steady state errors, so modeling errors on the static gain of the global system do not seem to be critical. Since both modeling techniques offer us the same modeling qualities, we have chosen the velocity-based model to apply our synthesis procedures. These works are the focus of the next part of the contract between Snecma Moteurs and LAAS-CNRS and will be the subject of further publications.

REFERENCES

P. Apkarian, P. Gahinet (1995). A Convex Characterization of Gain Scheduled H_∞ Controllers. *IEEE Trans. Automatic Control*, vol. 40, no.5, pp.853-864.

S. Boyd, L. El Ghaoui, E. Feron and V. Balakrishnan. Linear Matrix Inequalities in systems and Control Theory. vol. 15 of *SIAM Studies in Applied Mathematics*, SIAM, Philadelphia.

F. Bruzelius, C. Breitholz and S. Pettersson (2002). LPV-based gain scheduling technique applied to a turbofan engine model. *Proc. IEEE Int. Conf. Control Applications*, Glasgow, Scotland, UK.

M.J. Grimble. Industrial Control Systems Design, chap. 13, pp.569-617, John Wiley and Sons Ltd, Chichester, UK.

D. Henrion, L. Reberga, J. Bernussou, F. Vary (2004). Linearization and Identification of Aircraft Turbofan Engine Models. *IFAC Symposium on Automatic Control in Aerospace*, St Petersburg, Russia.

D.J. Leith, W.E. Leithead (1998). Gain-scheduled and nonlinear systems: dynamics analysis by velocity-based linearization families. *International Journal of Control*, vol. 70, 289-317.

D.J. Leith, W.E. Leithead (2000). Survey of gain-scheduling analysis and design. *International Journal of Control*, vol. 73, no 11, 1001-1025.

A. Nobakhti and N. Munro (2002). A Simply Structured Multivariable Control System For The Rolls Royce Spey Engine. *IEEE International Conference on Control Applications*, Glasgow, Scotland, UK.

A. Packard (1994). Gain scheduling via linear fractional transformations. *Systems and Control Letters* no 22, pp.79-92.

S. Skogestad, I. Postlethwaite (1996). *Multivariable feedback control*, section 12.3, pp. 480-489, Wiley, London.

S. Wang, J.H. Chow, R. Rajamani, K.D. Minto (1999). Fixed-order controller design for aircraft engine control applications. *IEEE Conference on Decision and Control*, pp. 1709-1714, Phoenix, Arizona.

G. Wolodkin, G.J. Balas, W.L. Garrad (1999). Application of parameter-dependent robust control synthesis to turbofan engines. *AIAA Journal of Guidance, Control and Dynamics*, vol 22, no. 6, pp. 833-838.

The MathWorks Inc. (2002), *Simulink v5.0*, Release 13, Natick, MA.