Chapter 2
Engine Models and Simulation Tools

Abstract This chapter offers a brief overview of engine dynamics, aiming at the extraction of linearized models that can be used as a basis for design. The Commercial Modular Aeropropulsion System Simulation (CMAPSS) package developed by the NASA Glenn Research Center is also described.

This chapter outlines the methodologies commonly followed to arrive at dynamic models, which are suitable for control design. A public-domain simulation package developed by NASA is also described, which will be used for many simulation studies contained in this book.

2.1 Two-Spool Shaft Dynamics

Mechanical system dynamics due to rotating inertias constitute the most important contribution to engine transient behavior. In fact, shaft speeds are directly linked with mass flow through the engine and thrust, which is the main output to be manipulated by the propulsion control system. Additional dynamics due to gas mass storage and heat transfer between gas and metal are present, but their use is reserved to high-fidelity, detailed models. Likewise, the dynamics of actuator systems are significantly faster than those associated with rotating masses, and are usually left unmodeled. As it will be seen next, *outputs of interest such as turbine temperatures, pressure ratios and stall margins can be regarded as outputs of a dynamic model whose states are shaft speeds*.

In what follows, we assume the engine to be a two-spool turbofan, where N_f denotes the angular speed of the assembly formed by fan, LPC, and LPT, and N_c denotes the angular speed of the HPC-HPT shaft assembly (see Fig. 1.3). For consistency with the standard terminology and simulation software, N_f and N_c will be denoted *fan speed* and *core speed*, respectively. Newton's law for rotating masses is applied to each shaft as follows:

$$\dot{N}_\mathrm{f} = f_1(N_\mathrm{f}, N_\mathrm{c}, u, w), \tag{2.1}$$

$$\dot{N}_\mathrm{c} = f_2(N_\mathrm{f}, N_\mathrm{c}, u, w). \tag{2.2}$$

H. Richter, *Advanced Control of Turbofan Engines*, DOI 10.1007/978-1-4614-1171-0_2,

Here, f_1 and f_2 are the net torques delivered by the LPT and HPT, respectively, normalized by the mass moment of inertia of the shaft assemblies. Vector u contains the control input components. In a single-input control system, u is given by the fuel flow rate W_F. Additional control effectors may be enabled as actuators; for instance, VBV and VSV may be included as components of u in addition to W_F. These additional inputs may be changed in a closed-loop or in an open-loop fashion. In the latter case, VBV and VSV are scheduled, that is, calculated as functions of engine variables other than states or inputs. Special attention must be placed on whether these effectors are held constant – thus becoming parameters –, scheduled, or used as control inputs. These choices have important implications in subsequent linearization procedures and controller designs. Vector w is used to capture the effects of uncertain inputs such as disturbances. An important interpretation of w is given by the *health parameter inputs*, a set of quantities representing engine deterioration and faults. Health parameters are discussed in Sect. 2.1.3.

Engine dynamics arise from complex, interacting phenomena: gas-flow behavior in the compressor and turbine (affected by air inlet as well as engine conditions), shaft inertias and losses, fuel flow transport delay, combustion and the thermal behavior of the engine and its surroundings. Due to the intricate geometry of the engine components and the complexity of gas flow, algebraic expressions for f_1 and f_2 are unavailable. Even the more sophisticated model generation and simulation tools [2] resort to linearization of (2.1) and (2.2) as a basis to include information about f_1 and f_2 through their partial derivatives. Note that these functions are highly dependent on external variables such as aircraft speed and atmospheric conditions, which act as their parameters.

When constant values of u and w are applied, along with a fixed set of parameters for f_1 and f_2, the engine reaches a steady–state operating point, with corresponding constant values of core and shaft speeds. Small-signal linearization is performed at these conditions, yielding the model:

$$\begin{aligned}\Delta\dot{N}_f = &\left.\frac{\partial f_1}{\partial N_f}\right|_o \Delta N_f + \left.\frac{\partial f_1}{\partial N_c}\right|_o \Delta N_c + \left.\frac{\partial f_1}{\partial u_1}\right|_o \Delta u_1 + \left.\frac{\partial f_1}{\partial u_2}\right|_o \Delta u_2 + ... \\ &.... + \left.\frac{\partial f_1}{\partial w_1}\right|_o \Delta w_1 + \left.\frac{\partial f_1}{\partial w_2}\right|_o \Delta w_2 + ...,\end{aligned} \tag{2.3}$$

$$\begin{aligned}\Delta\dot{N}_c = &\left.\frac{\partial f_2}{\partial N_f}\right|_o \Delta N_f + \left.\frac{\partial f_2}{\partial N_c}\right|_o \Delta N_c + \left.\frac{\partial f_2}{\partial u_1}\right|_o \Delta u_1 + \left.\frac{\partial f_2}{\partial u_2}\right|_o \Delta u_2 + ... \\ &.... + \left.\frac{\partial f_2}{\partial w_1}\right|_o \Delta w_1 + \left.\frac{\partial f_2}{\partial w_2}\right|_o \Delta w_2 + ...,\end{aligned} \tag{2.4}$$

where the subscript $_o$ has been used to indicate that the partial derivatives are evaluated at steady-state conditions. The steady values of N_f and N_c corresponding to constant u, w and parameters are found by iterative procedures attempting to equate the right-hand sides of (2.1) and (2.2) to zero. The process of finding the steady

speeds and outputs corresponding to a set of steady inputs and parameters is carried out by an *engine model balancer*, or *steady-state solver*. The implicit relationship between constant inputs and steady outputs is called a *steady engine map*.

2.1.1 Model Construction from Cycle Deck Data

The partial derivatives of f_f and f_2 are obtained from a perturbation method based on experimental data. Engine manufacturers are able to operate the machine in a test stand, simulating multiple combinations of atmospheric conditions and airspeeds, in addition to any desired input values. Data is collected from a host of sensors, from which relevant quantities such as adiabatic efficiencies and stall margins are computed. To obtain an approximate value of a partial derivative, the dependent variable is slightly perturbed from its steady value, and the corresponding increment in net torque recorded. Such sensitivity of the net torque to perturbations in the dependent variable is the desired value for the partial derivative.

By repeating this procedure in a grid of values for u and function parameters, local gradient information for f_1 and f_2 is collected. While a controls-oriented model can be as simple as a single linear model corresponding to a point in the grid, a high-fidelity simulation model uses partial derivative information obtained from a fine grid covering the entire flight envelope. Such models "piece together" this massive information, enabling numerical integration of trajectories across a wide range of operating conditions.

Outputs of the form $y_i = y_i(N_\mathrm{f}, N_\mathrm{c}, u, w)$ are also linearized to yield

$$\Delta y_i = \left.\frac{\partial y_i}{\partial N_\mathrm{f}}\right|_o \Delta N_\mathrm{f} + \left.\frac{\partial y_i}{\partial N_\mathrm{c}}\right|_o \Delta N_\mathrm{c} + \left.\frac{\partial y_i}{\partial u_1}\right|_o \Delta u_1 + \left.\frac{\partial y_i}{\partial u_2}\right|_o \Delta u_2 + ... \\ .... + \left.\frac{\partial y_i}{\partial w_1}\right|_o \Delta w_1 + \left.\frac{\partial y_i}{\partial w_2}\right|_o \Delta w_2 + ... \tag{2.5}$$

It is worth emphasizing that engine dynamic models are available to the controls engineer only as a collection of linear models, each one valid in a neighborhood of an equilibrium point and corresponding to fixed set of parameters reflecting flight conditions.

These models are expressible in the standard state-space form:

$$\dot{x} = Ax + Bu + \Gamma w, \tag{2.6}$$

$$y = Cx + Du + \Lambda w, \tag{2.7}$$

where $x^\mathrm{T} = [\Delta N_\mathrm{f}\ \Delta N_\mathrm{c}]$, $y^\mathrm{T} = [\Delta y_1\ \Delta y_2....]$ and matrices A, B, C, D, Γ, and Λ contain the partial derivatives as follows:

$$A = \begin{bmatrix} \left.\frac{\partial f_1}{\partial N_f}\right|_o & \left.\frac{\partial f_1}{\partial N_c}\right|_o \\ \left.\frac{\partial f_2}{\partial N_f}\right|_o & \left.\frac{\partial f_2}{\partial N_c}\right|_o \end{bmatrix}, \qquad B = \begin{bmatrix} \left.\frac{\partial f_1}{\partial u_1}\right|_o & \left.\frac{\partial f_1}{\partial u_2}\right|_o & \cdots \\ \left.\frac{\partial f_2}{\partial u_1}\right|_o & \left.\frac{\partial f_2}{\partial u_2}\right|_o & \cdots \end{bmatrix},$$

$$C = \begin{bmatrix} \left.\frac{\partial y_1}{\partial N_f}\right|_o & \left.\frac{\partial y_1}{\partial N_c}\right|_o & \cdots \\ \left.\frac{\partial y_2}{\partial N_f}\right|_o & \left.\frac{\partial y_2}{\partial N_c}\right|_o & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}, \quad D = \begin{bmatrix} \left.\frac{\partial y_1}{\partial u_1}\right|_o & \left.\frac{\partial y_1}{\partial u_2}\right|_o & \cdots \\ \left.\frac{\partial y_2}{\partial u_1}\right|_o & \left.\frac{\partial y_2}{\partial u_2}\right|_o & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix},$$

$$\Gamma = \begin{bmatrix} \left.\frac{\partial f_1}{\partial w_1}\right|_o & \left.\frac{\partial f_1}{\partial w_2}\right|_o & \cdots \\ \left.\frac{\partial f_2}{\partial w_1}\right|_o & \left.\frac{\partial f_2}{\partial w_2}\right|_o & \cdots \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \left.\frac{\partial y_1}{\partial w_1}\right|_o & \left.\frac{\partial y_1}{\partial w_2}\right|_o & \cdots \\ \left.\frac{\partial y_2}{\partial w_1}\right|_o & \left.\frac{\partial y_2}{\partial w_2}\right|_o & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}.$$

2.1.2 Models from System Identification

System identification techniques – parametric and spectral – are also applicable to GTE model generation. These techniques are more formal and systematic than the empirical partial derivative evaluations described above. Also, measurement noise and its impact on model quality are addressed explicitly. In a parametric system identification approach, a model structure is first selected where system order and representation (state-space, transfer function, etc.) are defined. Then parameters are estimated from experimental data using a numerical optimization algorithm. In a nonparametric (spectral) model, the objective is to obtain numerical frequency response traces (Bode plots) using input/output experimental information. A parametric model may also be fitted to the frequency domain traces by least squares or any other suitable curve fitting procedure.

System identification techniques have several drawbacks, including the need for time-consuming trial-and-error procedures to arrive at the proper model structure, the influence of the chosen test input on the results and the need for long data records [15]. For comprehensive information on system identification, the reader is referred to classical works such as Ljung [16] or Sage and Melsa [17]. For a detailed treatment of these methods as applicable to GTEs, refer to Kulikov and Thompson [18].

2.1.3 Engine Aging and Deterioration Modeling

The response of engine states and outputs to actuator inputs changes with time, according to the engine's "age", typically measured in hours of operation. Mechanical

wear due to normal use is reflected as changes in internal flow characteristics and component efficiencies. Component damage and engine-to-engine manufacturing variations also result in significant changes to engine response. Although aging and deterioration effects are distributed through the engine, a finite set of *engine health parameters* is chosen that describe changes in specific engine components. For instance, a normalized health parameter defined for the turbine efficiency may vary between 0 and 1, with 0 representing a new turbine and 1 a "fully deteriorated" one. The maximum level of deterioration is chosen by the modeler to indicate that the component is no longer usable and that an engine overhaul is necessary. Health parameter changes are represented as a bounded disturbance input vector w in the linearized engine model of (2.6) and (2.7). Note that the same set of health parameter inputs may be used to represent both slow aging and sudden fault, the distinction being the particular function of time used in simulation studies.

Health parameters are used in several ways, ranging from condition monitoring and fault detection to active control. In the latter case, designers may resort to robustness properties or may rely on disturbance estimation techniques. Condition monitoring and fault detection techniques require reliable parameter estimates during engine operation. Health parameter estimation has been the subject of much research and continues to be an active area of interest. The interested reader is referred to the comparative study by Volponi et al. [19] covering Kalman filter and soft computing estimation techniques. Further details on related problems and specific applications of these techniques are found, for instance, in [20–22] and references therein.

This book does not cover health parameter estimation techniques or their use in fault detection and accommodation. However, control strategies will be studied in detail which strive to preserve engine performance in the face of such uncertain inputs. Some robust strategies incorporate a disturbance estimator, while others rely entirely on robustness and disturbance bounds. Accounting for engine model variations is crucial in propulsion control system design. Besides the need for consistent thrust response throughout engine life, health parameter changes cause the safety limits described in Sect. 1.3 to shift, reducing operability margins. For instance, compressor deterioration – as captured by a corresponding parameter – shifts the stall line in a direction that reduces the stall margin.

2.2 Commercial Modular Aero-Propulsion System Simulation

Most simulations presented in this book were generated using the state-of-the-art Commercial Modular Aeropropulsion System Simulation, or Commercial Modular Aero-Propulsion System Simulation (CMAPSS). This package was developed at the NASA Glenn Research Center and is intended for public distribution [2]. CMAPSS is a Simulink port and a database with a user-friendly graphical user interface (GUI) allowing the user to perform model extraction, elementary control design, and simulations without much effort. One version of CMAPSS contains a model

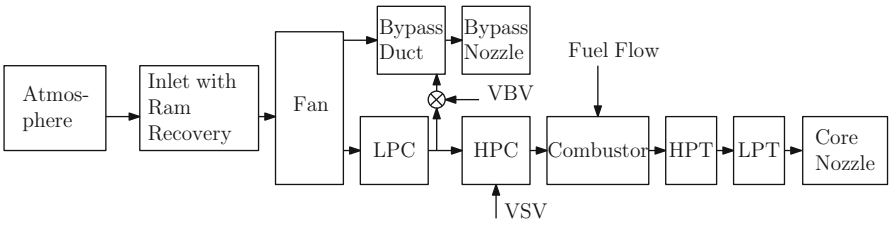

Fig. 2.1 CMAPSS software modules

corresponding to a large, high-bypass ratio turbofan engine similar to the GE90. Indeed, the CMAPSS-1 engine produces about 90,000 lbs of thrust at take-off conditions. A separate version, CMAPSS-40k [23], contains a model representing an engine in the 40,000-lb thrust class.

In this section, the main features of CMAPSS are described, along with a representative session including model linearization, basic controller specification, and simulation functions.

2.2.1 CMAPSS Main Features

The engine model is composed of several cascading modules, as shown in Fig. 2.1. The atmospheric model comprises air properties spanning altitudes from 0 to 40,000 ft above sea-level, Mach numbers from 0 to 0.9, and sea-level temperatures between −60 and 103°F.

2.2.1.1 Model Inputs

Model inputs are divided into two categories: control inputs and health parameter inputs. All versions of CMAPSS allow direct manipulation of fuel flow, with total freedom for the implementation of feedback laws. Health parameter inputs may only be specified as step functions of time. Simple modifications to the supplied Simulink model can be introduced to allow for more general health parameter variations. Depending on version, VSV and VBV may be accessible as control inputs or may be subjected to scheduling functions hidden from the user. CMAPSS-1 includes 14 inputs, fuel flow rate being the first. The remaining 13 inputs constitute the set of health parameters associated with pressure, flow, and efficiency characteristics of the fan, LPC, LPT, HPC, and HPT. VSV and VBV are not available as control inputs in this version. CMAPSS-40k includes 16 inputs: W_F, VSV, VBV, and 13 health parameters. The complete list of inputs in CMAPSS is shown in Table 2.1. Each time linearization is performed in CMAPSS-1, a B matrix with 2 rows and 14 columns is obtained. In CMAPSS-40k, B has 2 rows and 16 columns.

Table 2.1 CMAPSS engine model inputs (adapted from [2, 23])

Index	Input
1	Fuel flow (pps)
2	Variable stator vane (degrees)
3	Variable bleed valve (degrees)
4	Fan efficiency modifier
5	Fan flow modifier
6	Fan pressure-ratio modifier
7	LPC efficiency modifier
8	LPC flow modifier
9	LPC pressure-ratio modifier
10	HPC efficiency modifier
11	HPC flow modifier
12	HPC pressure-ratio modifier
13	HPT efficiency modifier
14	HPT flow modifier
15	LPT efficiency modifier
16	LPT flow modifier

2.2.1.2 Pilot Commands: TRA and PLA

The *throttle resolver angle*, or TRA, is the angular deflection of the pilot's power lever, having a range from 0 to 100%. In closed-loop operation, it is desirable to obtain an approximately linear steady-state relationship between TRA setting and net thrust across a wide range of inlet conditions. As elaborated in Sect. 3.1, thrust control is achieved indirectly by controlling fan speed. This is because no real-time thrust sensing is available in aircraft systems. For this reason, TRA is mapped into a corrected fan speed demand through a static function. Actual fan speed demand is then calculated using the inlet conditions reflected in θ. In control systems terminology, the TRA-to-$N_{f,cr}$ demand mapping constitutes a static reference prefilter placed outside the fan speed feedback loop. The subsequent conversion to N_{f} demand involves inlet variables which can be regarded as slowly-varying relative to the characteristic times of engine dynamics and its control system. Therefore, propulsion control design may proceed by assuming that a fan speed demand is supplied as an independent reference input. The *power lever angle*, or PLA, is an alternative designation for the same pilot control.

2.2.1.3 Model Outputs

Twenty-seven outputs are included in linearization models obtained via the GUI. These outputs, in addition to many other auxiliary quantities, are written to the workspace when a simulation is carried out. The 27 model outputs of CMAPSS-1 are listed in Table 2.2, and the reader is referred to the User's Guides [2, 23] for a complete listing of additional variables. The C and D matrices obtained through

Table 2.2 CMAPSS-1 engine model outputs (adapted from [2])

Index	Output	Units
1	Fan speed, N_f	rpm
2	Core speed, N_c	rpm
3	Engine pressure ratio, EPR=$\frac{P_{50}}{P_2}$	–
4	Total pressure at fan outlet, P_{21}	psia
5	Total temperature at fan outlet, T_{21}	°R
6	Total pressure at LPC outlet, P_{24}	psia
7	Total temperature at LPC outlet, T_{24}	°R
8	Total pressure at HPC outlet, P_{30}	psia
9	Total temperature at HPC outlet, T_{30}	°R
10	Total pressure at burner outlet, P_{40}	psia
11	Total temperature at burner outlet, T_{40}	°R
12	Total pressure at HPT outlet, P_{45}	psia
13	Total temperature at HPT outlet, T_{48}	°R
14	Total pressure at LPT outlet, P_{50}	psia
15	Total temperature at LPT outlet, T_{50}	°R
16	Fan massflow, W_{21}	pps
17	Net thrust, F_n	lbf
18	Gross thrust, F_g	lbf
19	Fan stall margin	–
20	LPC stall margin	–
21	HPC stall margin	–
22	Corrected fan speed	rpm
23	Corrected core speed	rpm
24	Total bypass duct pressure, P_{15}	psia
25	Percent corrected fan speed	–
26	Static pressure at HPC outlet, P_{s30}	psia
27	Ratio, $\frac{W_f}{P_{s30}}$	pps/psi

linearization in CMAPSS-1 have dimensions 27 by 2 and 27 by 14, respectively. Naturally, most of these outputs serve only a monitoring purpose, and only a few have a corresponding sensor in the real engine.

2.2.1.4 Pre-defined Flight Condition Data

Given a set of fixed inlet conditions and health parameters, each steady value of fuel flow rate W_F corresponds to a unique equilibrium point, defined by steady fan and core speeds. Also, each one of the 27 outputs adopts a corresponding steady value. Altitude, Mach number, and sea-level temperature completely define atmospheric conditions. These three quantities can be regarded as parameters of f_1, f_2 and y_i in (2.1), (2.2), and (2.5), and must be specified as part of the linearization process. The set of inlet conditions, steady fuel flow and corresponding steady states and outputs constitutes a *flight condition*. CMAPSS-1 is distributed with a set of predefined flight condition files.

Table 2.3 Equilibrium point data for representative flight conditions from CMAPSS-1 (adapted from [2]). All health parameter inputs are zero

	FC01	FC05	FC06	FC07	FC08	FC09
Alt, ft	0.00	10000.00	20000.00	25000.00	35000.00	42000.00
Mach	0.00	0.25	0.70	0.62	0.84	0.84
TRA, °	100.00	100.00	100.00	60.00	100.00	100.00
W_F, pps	6.84	4.66	3.86	1.67	2.12	1.52
N_f, rpm	2388.00	2319.00	2324.00	1915.00	2223.00	2212.00
N_c, rpm	9051.00	8774.00	8719.00	8006.00	8346.00	8317.00
EPR	1.30	1.26	1.08	0.94	1.02	1.02
T_{48}, °R	2072.00	1947.00	1909.00	1534.00	1750.00	1744.00
Ps30, psia	522.13	371.76	206.76	163.94	183.10	130.51
LPC Rline	1.64	1.63	2.31	1.70	1.52	1.54
HPC Rline	1.95	1.96	1.98	2.03	2.00	2.03
F_n, lbf	86636.00	45830.00	25774.00	11475.00	13552.00	9647.00

Arbitrary flight conditions may be calculated in CMAPSS by specifying atmospheric conditions and fuel flow, followed by simulation under any stabilizing controller. In CMAPSS-1, the results depend on the particular scheduling functions used for VSV and VBV. CMAPSS-40k treats these actuators as control inputs, thus two constant values must be selected along with a steady fuel flow value to define the point of linearization completely. CMAPSS includes a steady-state solver, which accepts steady input values and inlet conditions to find the steady values of the two states and all outputs.

A set of predefined flight conditions is included in the CMAPSS-1 distributions. These conditions may be invoked from the GUI as the first step in creating linearized models and designing elementary controllers. Six representative flight conditions extracted from CMAPSS-1 have been summarized in Table 2.3. Appendix B contains tables for the linearized model matrices at each one of the six representative flight conditions. This information is provided as an alternative for readers not using CMAPSS. No predefined flight condition data is distributed with CMAPSS-40k, however, several typical operating regimes are presented here for future reference. Six regimes representing several stages of flight are summarized in Table 2.4. The component maps introduced in Sect. 1.3 are customarily presented in the industry in terms of corrected mass flow and speeds rather than absolute quantities. CMAPSS also uses these quantities for internal calculations purposes, as well as for displaying performance evaluation plots. The controls engineer should regard corrected quantities as aids in understanding the effects of parametric changes to the model introduced by flight condition variations.

2.2.2 Example

A sample session with CMAPSS-1 is described as a quick start-up guide for interested readers. Matlab and Simulink must be available on a Windows platform,

Table 2.4 Equilibrium point data for representative flight conditions from CMAPSS-40k. All health parameter inputs are zero

	A	B	C	D	E	F
	Ground idle	Flight idle	Approach	Max cruise	Max climb	Max T.O.
Alt, ft	1000.00	6000.00	10000.00	38000.00	6000.00	1000.00
Mach	0.10	0.50	0.50	0.75	0.40	0.15
PLA, °	40.00	44.00	52.00	68.00	73.00	78.00
W_F, pps	0.33	0.59	1.03	1.00	2.69	3.02
VSV, °	−51.40	−39.90	−23.97	−1.43	−5.96	−4.88
VBV, %	1.00	0.90	0.50	0.01	0.04	0.00
N_f, rpm	1375.77	2240.21	2987.78	3764.46	3814.22	3803.43
N_c, rpm	8624.00	9515.11	10279.50	10932.01	11440.57	11525.14
EPR	1.03	0.94	1.06	1.53	1.44	1.49
T_{48}, °R	1091.13	1211.32	1417.84	1682.52	1839.91	1877.65
SmHPC, %	37.48	38.30	33.75	19.90	22.94	22.01
F_n, lbf	1879.92	2596.65	6620.98	6658.35	20521.32	28829.92

since dynamically linked libraries (DLL files) used in Matlab S-functions were compiled for this operating system. Note, however, that the source C code for these libraries is distributed with CMAPSS for the advanced user to recompile to match his/her own architecture.

CMAPSS is distributed as a zipped file, which must be expanded in a directory where the user has write permissions. Upon doing this, the file `setup_everything.m` must be run to load basic data and set up the necessary paths.

A main dialog is brought up from which the user can choose to create a linearized model, design controllers or simulate the closed-loop engine, as shown in the diagram of Fig. 2.2.

In this example, we first obtain a linearized model at FC01 by following the GUI menus. The Simulink diagram used to inject perturbations and obtain the partial derivative information described in Sect. 2.1.1 is brought up and ran automatically. At the end of the run, the variable `SSeng_14x27_unsc` appears in the workspace, containing a state-space system object with 14 inputs and 27 outputs, without scaling. The first input corresponds to the incremental fuel flow ΔW_F with respect to the steady value defined by the flight condition, while the remaining 13 are health parameter inputs (the reader may think of them as parametric disturbances). The first two outputs are simply the two incremental states: ΔN_f and ΔN_c. To extract the state-space matrices from `SSeng_14x27_unsc`, we use the command `[A,B,C,D]=ssdata(SSeng_14x27_unsc)`. The 2x2 system matrix A is found as

```
>> A
A =
   -3.8557     1.4467
   0.46897    -4.7081.
```

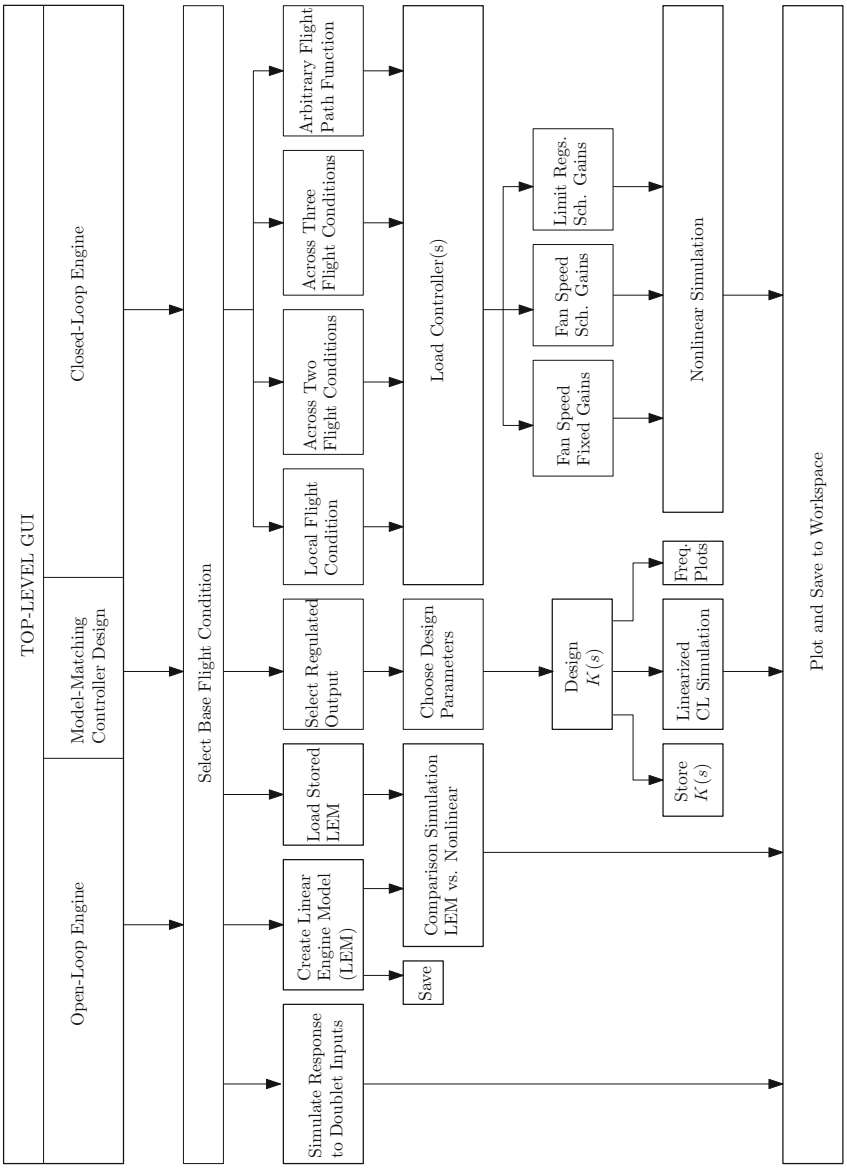

Fig. 2.2 CMAPSS functions diagram

Suppose an SISO model is desired having ΔW_f as input and ΔT_{48} as output. The following sequence of Matlab commands will store the desired model as `LEM001`:

```
>> B=B(:,1)
B =
    230.67
    653.55

>> C=C(13,:)
C =
    -0.057313  -0.32243

>> D=D(13,1)
D =
    146.37

>> LEM001=ss(A,B,C,D);
```

Thus, the transfer functions from ΔW_F to *any* of the 27 outputs at FC01 share the eigenvalues of A as poles: -3.3544 and -5.2093. In particular, the transfer function from ΔW_f to ΔT_{48} is

```
>> zpk(LEM001)
Zero/pole/gain: 146.37 (s+4.733) (s+2.301)
---------------------------
     (s+3.3544) (s+5.209)
```

Next, suppose a basic controller is to be tried for a fan speed set point change near FC01. The GUI design tool is limited to one form of compensation, consisting of lead-lag action with integration at plant input. The compensator pole is selected arbitrarily in the GUI, while the gain and zero are found from a least-squares algorithm referred to as "model-matching method". The optimization objective is to match a second-order closed-loop transfer function specified by the user by natural frequency and damping ratio. More details about the model-matching method and its limitations are given in Sect. 3.1.4. Letting the compensator pole to be -20 and choosing a closed-loop frequency of 4 rad/s and a damping ratio of 0.7, the following compensator is returned in variable `SSreg_unsc`:

```
>> zpk(SSreg_unsc)

Zero/pole/gain: 0.1122 (s+4.814)
---------------------------
     (s+20)
```

An incremental, linearized simulation may be launched from GUI for preliminary response evaluation. A Simulink diagram is brought up corresponding to this case. The default setpoint for ΔN_f may be readily changed by the user. Although the Simulink model uses scaled variables, the GUI plotting functions present the data in

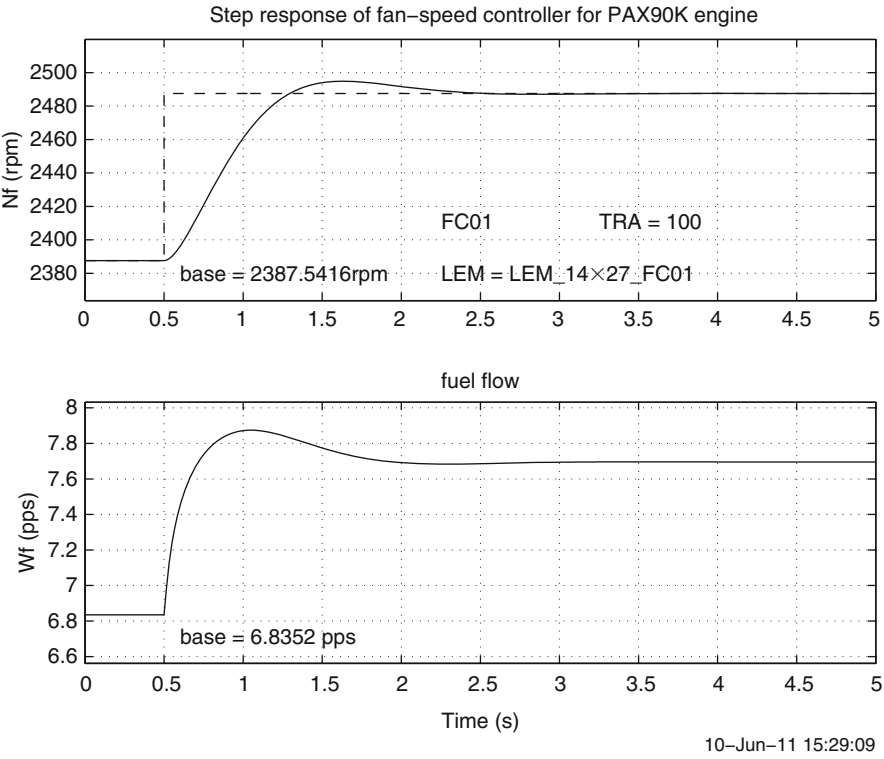

Fig. 2.3 Linearized simulation results produced by CMAPSS-1

absolute terms, as shown in Fig. 2.3. Now suppose the designed compensator is to be tested against the full nonlinear engine model, operating with a single fan speed control loop. Assuming `SSreg_unsc` still exists in the workspace, GUI dialogs may be followed to reselect FC01 as flight condition and build the closed-loop system. A Simulink diagram is brought up that contains the complete engine model and controller sections. Note that the setpoint reference is not directly given in terms of target fan speed or target incremental fan speed, but rather in terms of TRA, as described in Sect. 2.2.1.

The actual fan speed reference input can be accessed after running the simulation through variable `Nf_dmd`. In this example, a ramp input to reduce fan speed from 2387.5 to 2224.2 rpm with a slope of −300 rpm/s is used as a default. This rate and a positive rate limit of 500 rpm/s arise from a block in the Simulink diagram. The user may set these limits to `-inf` and `inf` to generate a step response. Upon doing this, various outputs of interest may be plotted using the GUI as shown in Fig. 2.4. Many other variables not accessible from the GUI are also written to the workspace.

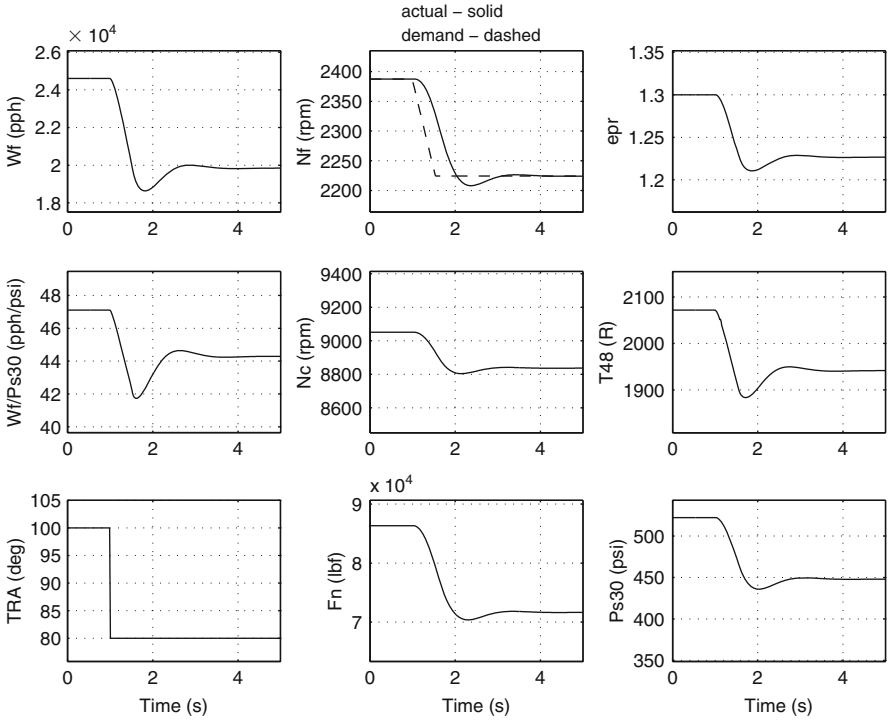

Fig. 2.4 Nonlinear engine simulation results panel produced by CMAPSS-1

2.2.2.1 Actuator Models (CMAPSS-40k)

CMAPSS-40k includes models for the three main actuators and their local control loops. Feedback is used for the fuel metering valve (FMV) and the VSV actuator, while VBV is operated in open-loop. The electrically-actuated FMV is placed around a proportional control loop to produce the W_F demanded by the engine controller. Since the valve opening dynamics from actuator current to position include a free integrator, P control is sufficient to achieve zero steady-state error (see Sect. 3.1.1) if the valve operates in a linear regime. Backlash, saturation, rate limits, and delay are present, however, as shown in Fig. 2.5. Note also that valve position – rather than actual flow – is fed back, implying that accurate calibration data must be available. If the nonlinearities and the delay are ignored, the closed-loop transfer function from desired to actual flow rate is given by

$$T_{\mathrm{FMV}}(s) = \frac{350}{s^2 + 40s + 350}$$

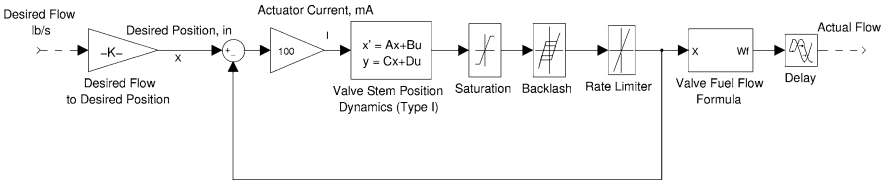

Fig. 2.5 Local control loop for the fuel metering valve (CMAPSS-40k)

with poles at -27.07 and -12.93. The VSV actuator is placed around a proportional-integral loop, which leads to the linear closed-loop transfer function:

$$T_{\mathrm{VSV}}(s) = \frac{3.75}{s + 3.75},$$

where an approximate pole-zero cancelation has been carried out. Indeed, the PI controller introduces a zero designed to nearly cancel a closed-loop pole near 0. The VBV actuator is operated in open-loop, with transfer function

$$T_{\mathrm{VBV}}(s) = \frac{23}{s + 23}.$$

An examination of Appendix C reveals that the A matrix associated with linearized engine dynamics has eigenvalues with real parts ranging from -0.75 to -4, according to flight condition. These values indicate that open-loop engine dynamics are significantly slower than those associated with the FMV and VBV actuators, but about as fast as the dynamics of the VSV actuator. The designer must decide whether to include actuator dynamics or to leave them as unmodeled dynamics and rely on the robustness properties of his/her designs. Actuator dynamics appear are cascaded at each plant input, making model extensions straightforward.

http://www.springer.com/978-1-4614-1170-3

Advanced Control of Turbofan Engines
Richter, H.
2012, XV, 266 p., Hardcover
ISBN: 978-1-4614-1170-3