NASA TECHNICAL MEMORANDUM

NASA TM X-71552

NASA TM X-71552

GENERALIZED DYNAMIC ENGINE SIMULATION TECHNIQUES FOR THE DIGITAL COMPUTER

by James Sellers and Fred Teren
Lewis Research Center
Cleveland, Ohio

TECHNICAL PAPER proposed for presentation at
Forty-fourth Propulsion and Energetics Meeting
sponsored by AGARD
Ankara, Turkey, September 9-13, 1974

GENERALIZED DYNAMIC ENGINE SIMULATION TECHNIQUES FOR THE DIGITAL COMPUTER

James Sellers and Fred Teren
NASA Lewis Research Center
Cleveland, Ohio, USA 44135

SUMMARY

Recently advanced simulation techniques have been developed for the digital computer and used as the basis for development of a generalized dynamic engine simulation computer program, called DYNGEN. This computer program can analyze the steady state and dynamic performance of many kinds of aircraft gas turbine engines. Without changes to the basic program, DYNGEN can analyze one- or two-spool turbofan engines. The user must supply appropriate component performance maps and design-point information.

Examples are presented to illustrate the capabilities of DYNGEN in the steady state and dynamic modes of operation. The analytical techniques used in DYNGEN are briefly discussed, and its accuracy is compared with a comparable simulation using the hybrid computer. The impact of DYNGEN and similar all-digital programs on future engine simulation philosophy is also discussed.

E-7988

LIST OF SYMBOLS

Symbols

A_8	exhaust nozzle area, m^2
A	state matrix
a	coefficient
E_i	error variable
f( )	function
h	enthalpy, J/kg
J	polar moment of inertia, kg-m^2
K_i	control gain
M	matrix of $\partial E_i/\partial V_j$
N	rotor speed, rpm
N_1	high-pressure rotor speed, rpm
N_2	low-pressure rotor speed, rpm
P	pressure, N/m^2
R	gas constant, J/kg/K
S	Laplace transform variable
T	temperature, K
t	time, sec
u	specific internal energy, J/kg
V	component volume, m^3
V_j	independent variable
$\dot{w}$	mass flow rate, kg/sec
y	dependent variable
γ	specific heat ratio
Δ	incremental change
ϵ	parameter in difference equation
λ	eigenvalue of differential equation
μ	eigenvalue of difference equation
τ	time constant, sec

Φ	state matrix

Subscripts:

accel	acceleration schedule
c	compressor
decel	deceleration schedule
dem	demand
F	fuel flow
i	integer
in	into control volume
j	integer
max	maximum
min	minimum
n	integer
out	out of control volume
r	reference
T	turbine
0	base value

INTRODUCTION

The intent of this paper is to discuss new techniques in use at NASA Lewis Research Center for the simulation of turbojet and turbofan engine dynamics. An introductory discussion will be given on the relative merits of analog, hybrid, and digital computers for use in dynamic engine simulation, but the body of the paper discusses a new digital computer program, called DYNGEN, which possesses significant advantages over traditional digital simulation methods.

Computer programs which predict the performance of real and proposed aircraft engines have long been recognized as indispensable tools for preliminary and detail design work. As engines and aircraft have grown more complex, analytical techniques have grown along with them to assist in optimizing engine configurations, comparing predicted performance with applicable criteria, exploring off-design performance, developing control modes and schedules, and providing timely data inputs to airframe designers. Recently, the prediction of engine dynamics has begun to play a significant role even in the preliminary design phases. For V/STOL aircraft, in which engine thrust provides lift and attitude control, engine response considerations may have a decisive effect very early in design. Therefore, dynamic engine simulations may provide data for the most fundamental design decisions as well as fulfilling their traditional role of supporting control modes studies and other kinds of development experiments.

The analog computer is one of the traditional tools of the dynamics and control specialist. Its chief advantage lies in the use of amplifiers which directly integrate the differential equations used to model the dynamic system. The analog computer is also a parallel-processing device, which means its components are all generating solutions simultaneously instead of one step at a time as in a digital computer. Therefore, the size of an analog problem has no effect on its execution time, and real-time solutions can sometimes be obtained. However, experience has shown that large analog simulations can be unwieldy to operate because of hardware limitations and lengthy setup procedures. Some other problem areas in analog simulation are the difficulty of generating bivariant functions such as compressor and turbine maps, poor day-to-day repeatability of solutions, and the difficulty of transferring simulations to other users. Since analog computers cannot easily solve implicit algebraic equations, simplified models often must be used to obtain explicit solutions for all variables (ref. 1). As a result, analog solutions will not agree perfectly with the solutions generated by highly-detailed steady-state digital simulations.

Modern hybrid computers have alleviated some of the problems noted above for analog computers. Specifically, the problem of bivariant function generation can be handled on the digital part of the hybrid computer. Also, the digital computer can be used to automate many of the setup and checkout procedures which confront the user of all-analog equipment. Although the digital part of a hybrid computer is slow compared with the analog part, real time solutions are still an attractive possibility for hybrid simulations.

As digital computers have grown larger and faster their attractiveness for dynamic engine simulation has improved. Their chief advantages lie in their ability to solve large numbers of complex algebraic equations and the ease with which logic can be implemented. Bivariant functions can be simulated with data tables, and systems of implicit algebraic equations can be solved by iterative methods. Digital programming languages have become standardized sufficiently to allow transfer of "engine decks" among users. Also, digital computers produce highly repeatable results. The main disadvantage of the digital computer, as far as dynamic simulations are concerned, is the need to choose an approximate method for solving the differential equations which model the system. Traditionally, this has led to numerical stability problems

and very long execution times for engine simulations. Furthermore, since most methods for solving differential equations require an explicit solution for all the derivatives (just as on the analog computer), the analyst frequently must use equations which differ from those used in a purely steady-state program. As a consequence the digital dynamic engine simulation will often share a problem with its analog counterpart: it won't agree with the steady-state deck.

Clearly, traditional methods of dynamic engine simulation involve problems which are worth eliminating. The purpose of this paper is to describe an all-digital computer program called DYNGEN which successfully solves some of the problems noted above. Historically, DYNGEN is a derivative of the GENENG program (refs. 2 and 3) developed at NASA Lewis Research Center. GENENG, in turn, is a derivative of the SMOTE program (refs. 4 and 5) developed by the U.S. Air Force Aero Propulsion Laboratory. SMOTE and GENENG are purely steady-state programs, but DYNGEN uses a recently developed method of solving differential equations which extends the capability of GENENG to include engine dynamics. As a result, DYNGEN includes in a single program all the steady-state capabilities of GENENG plus the added capability for dynamic calculations. This eliminates the traditional discrepancy between the results of dynamic and steady-state programs. DYNGEN is a generalized program: it enables the user to analyze one-, two-, or three-spool engines without reprogramming. Only component performance maps and certain design point information need be provided. DYNGEN also allows a variable time step in computing the time-dependent solution of the differential equations used in the engine model. This feature shortens execution times for long transients where the user is concerned only with low-frequency dynamics. DYNGEN is written in the FORTRAN IV language for the IBM 7094 computer. The basic version of DYNGEN requires about 32,000 words of storage.

The description of DYNGEN will begin with an overview of the amount of detail included in the thermodynamic and component calculations. Next, a description of the procedure used to obtain steady-state operating points will be given. The discussion will then proceed to explain how the solution of differential equations can be made a natural extension of the steady-state techniques by using a modified Euler method for solving differential equations. Appendix A is included to give mathematical details of the Newton-Raphson iterative technique, which DYNGEN uses to obtain both steady-state and dynamic operating points, and the modified Euler method for solving differential equations.

The latter portion of the paper is devoted to user-oriented subjects. A few examples are given to show the variety of engines that can be simulated without reprogramming, and the possible options for specifying off-design points are described. Finally, some examples of transient operation are presented to show how DYNGEN operates in connection with user-supplied control system subroutines.

ANALYTICAL TECHNIQUES

Thermodynamic and Component Calculations

Since DYNGEN is a modified version of a steady-state program, it contains details which are not always found in purely dynamic simulations. A discussion of these details will be given to assist the reader in visualizing the level of sophistication in the analytical model without actually presenting specific equations.

Engine components are represented in the usual way by performance maps. A typical fan or compressor map is shown in figure 1: pressure ratio and efficiency are plotted as functions of corrected flow and corrected speed. A typical combustor map is shown in figure 2: efficiency is plotted as a function of temperature rise across the combustor and entry pressure. The form of the turbine maps is one which has become common in steady-state engine simulations. A work parameter, $\Delta h/T$, and efficiency are plotted as functions of corrected speed and flow parameter $\dot{w}\sqrt{T}/P$, as shown in figure 3. Afterburner efficiency is calculated using three separate functions, as shown in figure 4. The user specifies a design-point afterburner efficiency, which is then adjusted to account for changes in fuel-air ratio, afterburner inlet Mach number, and afterburner inlet total pressure.

DYNGEN has been programmed to provide automatic map scaling. This feature is useful for preliminary design work since it means the user need only supply maps which he thinks would resemble the true maps of the engine he is simulating. In addition to supplying maps, the user must also specify the design operating point for each of the maps. The program will then linearly scale the map inputs and outputs to make their values compatible with the design pressure ratios, flows, etc., which have also been specified at the design point. The scale factors computed at the design point are saved and applied to the map inputs and outputs for all off-design cases.

In all thermodynamic calculations, gas properties are calculated as a function of temperature and fuel-air ratio. The temperature rise across a compressor is calculated without resorting to assumptions about the average gas properties across the component. Instead, a three-step procedure is used: first, the isentropic enthalpy rise is computed from knowledge of the pressure rise; second, the isentropic enthalpy rise is corrected by the efficiency; and third, the actual temperature is calculated from the actual enthalpy and pressure. A similar procedure is used to compute the temperature drop across a turbine.

Turbine cooling bleed is accounted for by mixing the bleed air downstream of the turbine; the bleed itself is assumed to do no work. In its basic form, DYNGEN contains no provision for compressor variable geometry or interstage bleeds. This problem was left to the user to account for by making appropriate adjustments to the component maps. Where the core and duct streams come together in a mixed-flow turbofan, a static-pressure balance calculation is made. The user also has the option of specifying separate nozzles for the core and duct if he wishes to simulate an engine with no flow mixing. Finally, in line with current NASA policy, DYNGEN was written to accept either English or SI units. This is accomplished by changing physical constants within the program rather than converting only the input and output.

Steady-State Balancing Technique

An example case will now be presented to assist the reader in understanding how DYNGEN calculates engine steady-state operating points. For simplicity a turbojet engine will be used in the example, but similar methods are used for more complicated configurations. Figure 5 shows a turbojet engine with its major components labelled. Pressures (P), temperatures (T), and flows ($\dot{w}$) are also labelled with appropriate station numbers. The example will illustrate how the calculation of variables proceeds through the engine. DYNGEN is written so that the user can select off-design points by specifying speed (N), turbine inlet temperature (T_4), or fuel flow ($\dot{w}_F$). In this example, fuel flow is assumed to be the specified variable. First, an inlet calculation is made to determine P_2 and T_2 from the free-stream values of pressure, temperature, and Mach number. To calculate $\dot{w}_c$, P_3, and T_3 from the compressor map (fig. 1) and thermodynamic relations, guesses must be made for the values of speed (N) and pressure ratio (P_3/P_2). Once $\dot{w}_c$, P_3, and T_3 are obtained, the combustor calculations for $\dot{w}_4$, P_4, and T_4 can be made using thermodynamic relations, the combustor map (fig. 2), and specified values for fuel flow ($\dot{w}_F$) and compressor bleed flow. To calculate turbine variables another guess is made, this time for the value of turbine flow function ($\dot{w}_4\sqrt{T_4}/P_4$). Then, from the known value of ($N/\sqrt{T_4}$), the turbine map (fig. 3) is used to calculate turbine work (Δh) and efficiency. P_7 and T_7 are then calculated using the thermodynamic relations. Finally, the compressible-flow relations are used to calculate nozzle pressure (P_7) from $\dot{w}_8$, T_7, and specified values for P_0 and nozzle area.

The reader may have noticed that the above calculation procedure is redundant; that is, certain variables can be calculated in more than one way. This fact is used to generate error variables which must equal zero to yield a consistent solution of the simulation equations. In writing a program such as DYNGEN, the analyst has great freedom in choosing what error variables to use. This discussion simply points out the choices which were made by the authors of SMOTE; experience has shown that these were good choices for most engine configurations, and the same error variables were retained in SMOTE's descendants, GENENG and DYNGEN.

In the previous discussion it was stated that guesses were made for rotor speed (N), compressor pressure ratio (P_3/P_2), and turbine flow function ($\dot{w}_4\sqrt{T_4}/P_4$). From the first two guesses (and other variables) one may calculate the power absorbed by the compressor ($\dot{w}_c\Delta h_c$). From the turbine flow function (and other variables) one may calculate the power supplied by the turbine ($\dot{w}_T\Delta h_T$). For steady-state operation the power supplied must equal the power absorbed. Therefore, the difference ($\dot{w}_c\Delta h_c - \dot{w}_T\Delta h_T$) may be used for an error variable. Similarly, one can calculate a value for turbine flow function ($\dot{w}_4\sqrt{T_4}/P_4$)' based only on the first two guesses, but for a consistent solution the calculated value must equal the guessed value. Hence, the difference ($\dot{w}_4\sqrt{T_4}/P_4$) - ($\dot{w}_4\sqrt{T_4}/P_4$)' can be used as a second error variable. Finally, from the compressible-flow equations we know that the variable P_7 is specified by the variables $\dot{w}_8$, T_7, P_0, and nozzle area. This value for P_7 must equal the value P_7' which is calculated from thermodynamic relations at the turbine exit and from pressure losses in the duct between turbine and nozzle. Therefore, the third error variable is ($P_7 - P_7'$).

Once three variables have been guessed and three errors have been specified, the analyst can use an iterative method to obtain a consistent solution to the simulation equations. SMOTE, GENENG, and DYNGEN all use the Newton-Raphson technique of iteration. The details of this method are given in Appendix A. In figure 6, a simplified flow chart shows how the Newton-Raphson method is used in connection with the engine calculations discussed in the preceding example. Although more complicated engines will require more guesses and more error variables in the iterative procedure, the analysis will be quite similar to the one described in the example.

Simulation Differential Equations

So far the discussion has been devoted to the methods which DYNGEN uses to obtain steady-state operating points. Now the method of implementing and solving time-dependent differential equations will be discussed. DYNGEN uses a modified Euler method of solving differential equations. This method is derived from a numerical analysis viewpoint in Appendix A. Appendix A also discusses the numerical stability of the modified Euler method and shows that it does not require extremely small time steps to obtain a stable solution. This advantage is important in engine simulations because in the past it has often been necessary to select integration time steps small enough to guarantee stability for high-frequency dynamics typical of mass and energy storage in unsteady flow. This can result in very long execution times even though the simulation user may only be interested in low-frequency dynamics. With the modified Euler method, the user can select longer time steps without worrying about numerical stability. The main disadvantage of the modified Euler method is that an iterative solution is required for the difference equations which approximate the solution to the differential equations. However, this fact turns out to be useful in DYNGEN since it means that the analyst no longer has to solve explicitly for derivatives. They may be embedded anywhere in an overall set of simultaneous algebraic equations which are to be solved by an iterative method such as Newton-Raphson. The following discussion shows how this advantage was employed in modifying a steady-state simulation, GENENG, to form a dynamic simulation, DYNGEN. Figure 7 shows the three kinds of equations which were modified to include dynamic terms: the power balance, continuity, and energy equations. The steady-state power balance equation simply implies that the power output of a turbine must equal the power absorbed by a fan or compressor. By adding a rotor acceleration term, the equation can be used to model engine dynamics: any excess power provided by the turbine will go into rotor acceleration. If the time derivative is arbitrarily set equal to zero, the dynamic equation becomes the steady-state equation. Similar considerations also hold for the continuity equation. DYNGEN treats unsteady flow dynamics in a way which has become traditional for engine simulation: a control volume is associated with each component, and pressure, temperature, and density are assumed constant throughout the control volume. At steady state the flow into the volume must equal the flow out, but for unsteady flow mass can be stored in the volume at a rate proportional to the time derivative of pressure, dP/dt. If dP/dt is zero, the continuity equation reverts to its steady-state form. The control volume approximation is also used for the energy equation. At steady-state the rate of energy into the volume must equal the rate out, but, in unsteady flow, energy storage is accounted for by two terms: one reflecting the rate of change of specific internal energy, du/dt, and another reflecting

energy storage caused by mass storage.

DYNGEN was formed from GENENG by modifying the power balance, continuity, and energy equations wherever they occurred in GENENG. In GENENG, the steady-state power balance equation was used to form an error variable:

$$E_1 = \dot{\omega}_c \Delta h_c - \dot{\omega}_T \Delta h_T$$

In DYNGEN, the same error is formed with the dynamic term added:

$$E_1 = \dot{\omega}_c \Delta h_c - \dot{\omega}_T \Delta h_T + JN \frac{dN}{dt}$$

To implement the dynamic forms of the continuity and energy equations, a volume was associated with each component, and the flow and enthalpy out of the component were modified by the dynamic terms. For example, if $\dot{\omega}_c$ is the flow rate through the compressor specified by the compressor map, and h_3 is the enthalpy at the compressor exit, then the flow and enthalpy entering the combustor will be given by $\dot{\omega}_c'$ and h_3', where:

$$\dot{\omega}_c' = \dot{\omega}_c - \frac{V_3}{\gamma R T_3} \frac{dP_3}{dt}$$

$$h_3' = \frac{\dot{\omega}_c h_3 - (\dot{\omega}_c - \dot{\omega}_c') u_3 - \frac{P_3 V_3}{R T_3} \frac{du_3}{dt}}{\dot{\omega}_c'}$$

The derivatives are calculated by the simplest possible approximation:

$$\frac{dy}{dt} = \frac{y_i - y_{i-1}}{\Delta t}$$

where y_i is the current value of a variable and y_{i-1} is the value for the previous time step. This approximation is adequate provided that the time step, Δt, is about one-tenth the magnitude of the smallest time constant the user wants to observe. For example, if the user wants to observe rotor dynamics with a 1.0-second time constant, he should use a Δt no greater than 0.10 second.

Adding the derivative terms to the steady-state equations did not require any change to the basic iteration scheme used in GENENG. Therefore, none of the flexibility or generality of the program was lost; its capability was simply extended to include dynamics.

DYNGEN CAPABILITIES

Engine Configurations

The discussion so far has concentrated on the analytical techniques used in DYNGEN, or how it works. The remainder of the discussion will cover the user-oriented question of what can be done with it. First, a few examples will be given to show the variety of engines that can be analyzed without reprogramming. These examples are meant only to suggest the range of options the user has in specifying engine configurations: a complete list of options would be too lengthy for this paper.

Figure 8 shows the most complicated engine that can be handled by the basic version of DYNGEN: a three-spool, three-stream engine. The original motivation for simulating this kind of engine was the need to study blown-flap or ejector-wing propulsion systems for STOL aircraft. Hence, the third duct is called the "wing duct," and its air does not mix with the core or fan duct streams. If desired, the wing duct can be eliminated to simulate a three-spool, two-stream engine such as the Rolls-Royce RB.211. Figure 9 shows another step down in complexity: a two-spool, boosted-fan engine. The fan and booster may be represented by separate component maps. All of the turbofan engines simulated by DYNGEN can use a mixed-flow option rather than separate core and duct nozzles as shown in the figures. Duct burning or core afterburning are also available options. Finally, figure 10 shows the lowest level of complexity: the one- or two-spool turbojet.

Steady-State Capabilities

A brief discussion will now be given of the steady-state capabilities which DYNGEN inherited from its predecessor, GENENG. For each engine configuration that the user wishes to examine, a design point case must be run. At the design point the user must specify certain cycle parameters such as turbine inlet temperature, component efficiencies, Mach numbers at various stations, corrected flows, etc. Component maps must be provided, and logical controls must be set to establish how many spools the engine has, whether its core and duct streams mix, whether it is a turbojet or turbofan, and so on. As mentioned earlier, the component maps will be automatically scaled to conform with the cycle parameters specified by the user at the design point. After the design point case is run, DYNGEN will output calculated values for main and afterburner fuel flow, thrust areas at various stations, map scale factors, etc.

The user has a wide variety of options for running off-design steady-state points (refs. 3 and 4). First, there are four basic operating modes for specifying off-design points: constant turbine inlet temperature, constant main fuel flow, constant fan speed, or constant core speed. Once the user specifies which of these is to remain constant, and what its value is to be, he may then vary a wide variety of oper-

ating parameters such as altitude, Mach number, inlet recovery, bleed or power extraction, and core, duct, or wing nozzle areas. If afterburning is to be investigated, the user may specify either afterburner fuel flow or temperature. Nozzle area will be recalculated to maintain the same core engine operating point specified for some previous, nonafterburning case. These options also hold for duct burning. If the user wishes to investigate the effect of a converging-diverging nozzle on thrust performance, he may hold exit area constant or have it automatically recalculated for fully expanded flow. These options make DYNGEN a very useful tool for investigating steady-state off-design performance.

Dynamic Calculations

A few examples will now be given to show some of DYNGEN's capabilities for simulating engine dynamics. The first example will show an open-loop engine response to a step in fuel flow, and the remaining two will demonstrate how DYNGEN can be used in connection with user-supplied engine control subroutines. The input requirements for running engine dynamics are identical to the requirements for steady-state operation except that a few additional constants must be supplied at the design point: rotor moments of inertia, rotor design speeds in revolutions per minute, and component volumes. A design point case must be run, just as in steady-state operation, and if the user wants to specify some off-design point as the initial condition for a transient, he may do so using the steady-state options discussed previously. Finally, the user must specify a time step for the transient solution, and he must set a program index indicating that a transient is to be run. The transient disturbance input is supplied by a user-written subroutine. Any of the off-design input parameters which are available for steady-state operation, such as fuel flow, compressor bleed, inlet recovery, etc., may also be used as transient inputs. Furthermore, the user may generate transient inputs appropriate to the control system he is simulating, for example, a change in power lever angle.

The first example shows the response of a three-spool, three-stream engine (like the one shown in fig. 8) to an open-loop step in fuel flow. Figure 11 shows time histories of fan, middle spool, and core speeds. Also shown is the response of turbine inlet temperature. All variables are presented as percentages of their design values. Apart from showing DYNGEN's capability to simulate a three-spool engine, figure 11 also demonstrates the effect of using different time steps in the modified Euler solution of the simulation equations. The results are shown for two time steps: 0.01 second and 0.10 second. Close examination shows some small differences between the two solutions, but they are substantially identical. There is a big difference, however, in computer execution time to run the three-second transient shown in figure 11. Using the 0.10-second time step, execution time was 1.4 minutes; using the 0.01-second time step, execution time was 12.3 minutes. This example demonstrates one of the main advantages of a modified Euler solution method: the user may select the time step to show the frequency range of interest. If low-frequency effects, such as rotor dynamics, are the subject of interest, a time step of 0.10 second may be adequate. If higher frequency effects, such as temperature and pressure dynamics, are to be observed, then a smaller time step will be needed. In any case, execution times can be held to a minimum compatible with the user's interests.

The next example shows a large throttle transient for a two-spool turbofan similar to the one shown in figure 9. This engine was simulated along with the speed control system shown in figure 12. The primary input to the control system is demand speed, $N_{2,dem}$ which is set by the pilot's throttle lever. The only output of the system is fuel flow, $\dot{w}_F$, which goes to the combustor. During small throttle transients the control is proportional-plus-integral on speed error, but for large transients the control is closed-loop on the acceleration fuel flow schedule. Acceleration fuel flow is computed from compressor speed, N_1, compressor exit pressure, P_3, and compressor inlet temperature $T_{2.1}$. This moderately complex control system was simulated using subroutines compatible with DYNGEN's modified Euler solution method. A throttle step from 50 percent thrust to 100 percent thrust was applied to the simulation, and the results are shown in figure 13. Time histories of thrust and turbine inlet temperature are shown with the variables expressed as a percentage of their design value. This figure also presents a comparison of DYNGEN's results with those from a hybrid computer simulation of the same engine. In figure 13, the continuous lines are the hybrid computer solution and the discrete points are DYNGEN's solution. The hybrid computer model is quite detailed (ref. 6), but owing to differences in the simulation equations, the steady-state results of the two simulations differ by about 3 percent. The differences in the dynamic solutions are of the same order. The comparison shown in figure 13, though not perfect, tends to confirm the validity of DYNGEN's method of solving the differential equations used in modeling the engine and control system. Even though a fairly long time step of 0.10 second was used, DYNGEN's solution is quite similar to the continuous solution produced by the hybrid computer.

The final example of DYNGEN's flexibility involves a single-spool, afterburning turbojet similar to the one shown in figure 10. This type of engine requires exhaust nozzle and main fuel control subsystems as shown in figure 14. The main fuel control is a simple proportional control on speed error with acceleration and deceleration fuel flow limiting. The main input is demand speed, N_{dem}, which is set by the pilot's throttle. The acceleration schedule is the usual $(\dot{w}_F/P_3)_{accel}$ as a function of N and T_2, and the deceleration schedule is obtained simply by taking one-third of the acceleration schedule. The nozzle control is used only in the afterburning mode of operation. Its purpose is to null out any change in compressor pressure ratio, (P_3/P_2), which might occur when going from non-afterburning to afterburning operation. This is accomplished by proportional-plus-integral control of nozzle area, A_8, in response to pressure ratio error.

This control system was simulated in connection with a turbojet engine, and a throttle slam from idle to full afterburning was applied. The results are shown in figure 15. Time histories of rotational speed, main fuel flow, afterburner fuel flow, nozzle area, and thrust are shown. All variables are presented as percentages of their design value. To simulate a throttle slam, afterburner fuel flow was ramped from zero to its maximum value in two seconds, beginning as soon as rotor speed reached 100 percent. This example shows that DYNGEN can be used successfully to simulate the dynamics of an afterburning engine. Furthermore, it demonstrates that DYNGEN is not limited to small-perturbation problems. It has all the capabilities of its predecessors for simulating gross transients, but it is faster than most traditional simulations. The five-second transient shown in this example required about two minutes of computer execution time.

CONCLUSIONS

A generalized digital computer program for simulating the steady-state and dynamic performance of turbojet and turbofan engines has been described and discussed. This computer program, called DYNGEN, possesses significant advantages over traditional methods of digital engine simulation. Specifically, it eliminates the need to operate two separate computer programs to obtain steady-state and dynamic results. It uses a modified Euler method for solving differential equations which enables the user to specify a solution time step compatible with the frequency range of interest. This saves computer execution time when long transients are to be run. Finally, DYNGEN can simulate a wide variety of engine types without reprogramming. This saves money and man-hours when new engines are to be simulated. When real-time engine simulations are required the analyst must turn to analog or hybrid methods to achieve his goals. However, owing to the new digital simulation techniques used in DYNGEN, all-digital engine simulations are now capable of accomplishing nearly all the user's tasks with convenience and flexibility.

APPENDIX A

ANALYTICAL TECHNIQUES

Steady-State Balancing Technique

The following discussion explains the iterative method which DYNGEN and its predecessor GENENG use to calculate steady-state operating points. As noted earlier, the calculation of a steady-state operating point requires solution of a system of nonlinear equations, corresponding to various engine matching constraints such as rotational speeds, air flows, compressor and turbine work functions and nozzle flow functions. In order to satisfy these constraints, there are available an equal number of engine parameters which may be varied, such as compressor and turbine pressure ratios and flow functions. The specific number of engine parameters (independent variables) to be varied and engine matching constraints (dependent variables) to be satisfied depends on the type of engine configuration being studied, and varies from three for a single-spool turbojet engine to nine for a three-spool engine. The computer program (DYNGEN) searches for the values of the engine parameters which result in the engine matching constraints being satisfied.

If the independent variables are denoted by V_j and the dependent variables by E_i, the matching equations can be written as

$$E_i(V_j) = 0 \qquad \begin{matrix} i = 1,2,\ldots,n \\ j = 1,2,\ldots,n \end{matrix}$$

This is a set of nonlinear equations, which must be satisfied for a steady-state solution. The procedure used to satisfy these equations is the multi-variable Newton-Raphson method (ref. 7). With this method, changes in E are assumed to be related to changes in V by first-order, finite-difference equations:

$$\Delta E = M\Delta V$$

where ΔV and ΔE are n-vectors denoting changes in V and E from some reference condition, and M is an $n \times n$ matrix of partial derivations of E with respect to V:

$$M_{ij} = \frac{\partial E_i}{\partial V_j}$$

The matrix M is obtained by calculating a reference case and n independent perturbed cases, such that only the J-th variable V_j is perturbed from its reference value on the J-th case. Then for the J-th case,

$$M_{ij} \approx \frac{\Delta E_i}{\Delta V_j} \qquad i = 1,2,\ldots,n$$

Once the matrix M is obtained, the reference case is improved by using

$$V = V_r - M^{-1}E_r$$

If the system of equations were linear, this process would lead to convergence in one iteration. In practice, nonlinearities in the system prevent immediate convergence. In this case, the new V and E are taken to be the reference values, and a new matrix is generated. If the system is not too nonlinear, and initial guess for V are reasonably accurate, convergence is achieved in several iterations.

Dynamic Equations

Once an initial steady-state solution has been obtained, a time-varying solution may be generated. This requires the solution of a set of differential equations which model the system. The specific equations which are used to model the engine were discussed in the text. In this section, the procedure used to solve the differential equations in DYNGEN will be discussed.

Consider first the differential equation

$$\frac{dy}{dt} = f(y,t) \tag{1}$$

In order to obtain a numerical solution using a digital computer, this differential equation must be replaced by a difference equation, in such a way that the solution of the difference equation is in some sense close to that of the differential equation. There are many ways in which this can be done, as discussed, for example, in reference 7. A common method is to use a difference equation of the form

$$y_{j+1} = y_j + \Delta t[\varepsilon f(y_j,t_j) + (1 - \varepsilon)f(y_{j+1},t_{j+1})] \tag{2}$$

where

$$y_j \triangleq y(t_o + j\Delta t)$$

and

$$0 \leq \varepsilon \leq 1$$

The bracketed quantity in (2) represents a weighted average of the derivative $f(y,t)$ over the integration interval $[t_j,t_{j+1}]$. For $\varepsilon = 1$, equation (2) becomes

$$y_{j+1} = y_j + \Delta t f(y_j,t_j) \tag{3}$$

Equation (3) is known as Euler's method, and allows explicit calculation of y_{j+1} as a function of the previous values y_j and t_j. On the other hand, for $\varepsilon \neq 1$, equation (2) is the modified Euler method, and in general cannot be solved explicitly for y_{j+1}, because of the dependence of f on y_{j+1} which appears on the right hand side of the equation. In this case, some form of iteration must be used at each integration step to solve for y_{j+1}.

From the standpoint of simplicity of the integration formula, use of (3) is clearly preferable to the use of (2). However, there are two other important considerations: accuracy and stability. As discussed in the literature (e.g., ref. 7), use of (2) can lead to greater integration accuracy. Even more important for the dynamic engine simulation problem is the stability consideration.

To illustrate the stability consideration, consider the linear differential equation

$$\frac{dy}{dt} = ay \tag{4}$$

For this equation, (2) becomes

$$y_{j+1} = y_j + a\Delta t[\varepsilon y_j + (1 - \varepsilon)y_{j+1}] \tag{5}$$

which can be solved for y_{j+1} to give

$$y_{j+1} = \left(\frac{1 + a\varepsilon\Delta t}{1 + a\varepsilon\Delta t - a\Delta t}\right) y_j \tag{6}$$

the general solution for y_j can be written

$$y_j = r^j y_o \tag{7}$$

where

$$r = \frac{1 + a\varepsilon\Delta t}{1 + a\varepsilon\Delta t - a\Delta t} \tag{8}$$

the original differential equation (4) is stable for $a < 0$; the difference equation solution (7) is stable for $|r| < 1$. From (8), the requirements for stability of (7) can be established in terms of the requirements on integration step size, Δt. Solving (8) for Δt yields

$$\Delta t = \frac{1 - r}{a(\varepsilon r - r - \varepsilon)} \tag{9}$$

The upper and lower bounds for Δt are obtained by setting $r = \pm 1$ in (9). This results in

$$\Delta t < \frac{2}{a(1 - 2\varepsilon)}, \quad \varepsilon > \frac{1}{2} \tag{10a}$$

$$\Delta t \sim \text{unconstrained}, \quad \varepsilon < \frac{1}{2} \tag{10b}$$

In particular, for the Euler method ($\varepsilon = 1$), the step size must be less than $(-2/a)$ in order to avoid numerically-induced instability, while for $\varepsilon < 1/2$, the numerical method leads to a stable solution for any value of integration step size.

The above results are readily generalized to a system of linear differential equations. Consider the system of equations

$$\frac{dy}{dt} = Ay \tag{11}$$

where y is an n-vector and A is the $n \times n$ system matrix. Use of the numerical algorithm in (2) results in

$$y_{j+1} = y_j + A\Delta t[\varepsilon y_j + (1 - \varepsilon)y_{j+1}] \tag{12}$$

which has the general solution

$$y_j = \Phi^j y_o \tag{13}$$

where

$$\Phi = (I + A\varepsilon\Delta t - A\Delta t)^{-1}(I + A\varepsilon\Delta t)$$

As shown in reference 8, (11) is stable if and only if the eigenvalues of A all have negative real parts; the difference equation solution (13) is stable if and only if all the eigenvalues of Φ have magnitude less than unity.

It will now be proved that if λ is an eigenvalue of A, then

$$\mu = \frac{1 + \lambda\varepsilon\Delta t}{1 + \lambda\varepsilon\Delta t - \lambda\Delta t} \tag{14}$$

is an eigenvalue of Φ.

Proof

Let λ is an eigenvalue of A. Then

$$|A - \lambda I| = 0$$

If μ is an eigenvalue of Φ, then

$$|\Phi - \mu I| = 0$$

But

$$|\Phi - \mu I| = |(I + A\varepsilon\Delta t - A\Delta t)^{-1}(I + A\varepsilon\Delta t) - \mu I|$$

$$= \frac{|(I + A\varepsilon\Delta t) - \mu(I + A\varepsilon\Delta t - A\Delta t)|}{|I + A\varepsilon\Delta t - A\Delta t|}$$

$$= \frac{|(1 - \mu)(I + A\varepsilon\Delta t) + \mu A\Delta t|}{|I + A\varepsilon\Delta t - A\Delta t|}$$

But from (14),

$$1 - \mu = \frac{-\lambda\Delta t}{1 + \lambda\varepsilon\Delta t - \lambda\Delta t}$$

so that

$$|\Phi - \mu I| = \frac{|-\lambda\Delta t(I + A\varepsilon\Delta t) + (1 + \lambda\varepsilon\Delta t)\Delta t A|}{(1 + \lambda\varepsilon\Delta t - \lambda\Delta t)|I + A\varepsilon\Delta t - A\Delta t|}$$

$$= \frac{\Delta t|A - \lambda I|}{(1 + \lambda\varepsilon\Delta t - \lambda\Delta t)|I + A\varepsilon\Delta t - A\Delta t|} = 0$$

which completes the proof.

The similarity of (14) and (8), together with the requirement that all eigenvalues μ have magnitude less than unity, allow the conclusion, similar to (10), that

$$\Delta t < \frac{2}{\lambda_{max}(1 - 2\varepsilon)}, \quad \varepsilon > \frac{1}{2} \tag{15a}$$

$$\Delta t \text{ - unconstrained}, \quad \varepsilon < \frac{1}{2} \tag{15b}$$

where λ_{max} is the eigenvalue of A having the greatest magnitude. In particular, for the Euler method, the step size is restricted by

$$\Delta t < \frac{-2}{\lambda_{max}} \tag{16}$$

in order to avoid numerical instability.

The above results are valid only for a linear system, and no such general proofs are available for nonlinear systems. However, in an intuitive sense, it seems reasonable that equation (16) is applicable to nonlinear systems if the matrix A and eigenvalues λ are interpreted as "average" values over an integration step, and the system of equations (11) is not too nonlinear.

The significance of equation (16), particularly for the dynamic engine simulation problem, is the following. The dynamic engine simulation generally contains a mix of high and low frequencies. The high frequencies result from the lumped-volume representation of component dynamics, which includes the storage of mass and energy. The low frequencies result, for example, from rotor dynamics, and the slow motion of exhaust nozzle and associated control logic. Frequently, the simulation user is interested in low-frequency effects, such as overall engine spool-up time, and is not concerned with high frequency effects. Typical transients are of five- to ten-seconds in duration.

If the simulation uses Euler's method, the integration step size is restricted by the highest frequency in the system, even though the user is not interested in high frequency information. In this case, a step size of 10^{-4} seconds, or smaller, is frequently required. On the other hand, if an implicit (modified

Euler) technique is used ($\varepsilon < 1/2$), the step size is not restricted. It can be chosen to suit the desired frequency content of the output, which typically allows a step size of 0.1 seconds or larger.

ITERATIVE SOLUTION PROCEDURE

A problem which exists with the use of implicit methods, as noted earlier, is that for nonlinear differential equations, some iterative scheme is required to solve for the values of y_{j+1} at each integration step. The differential equations corresponding to the dynamic model of the engine may be written as

$$\frac{dy}{dt} = f(y) \tag{17}$$

where y and f are vectors. The state vector y represents pressures, temperatures and rotor speeds. The dimension of y (and f) depends on the type of engine configuration being studied. Nine state variables are required for a single-spool turbojet engine, and a greater number for more complex engines. The number of state variables required for a dynamic solution always exceeds the number of steady-state iteration variables required for the Newton-Raphson iteration discussed earlier.

The difference-equation representation used in DYNGEN utilizes $\varepsilon = 0$, so that (17) becomes

$$y_{j+1} = y_j + \Delta t f(y_{j+1}) \tag{18}$$

The discussion of the sample configuration in the main body of the report showed how the dynamic equations are incorporated into the structure of the steady-state solution. The steady-state continuity, energy and power equations are modified to be dynamic equations. The resulting dynamic equations are then included either as error equations, or used to calculate flows and enthalpies at various stations throughout the engine.

REFERENCES

1. Seldner, K., Mihaloew, J. R., and Blaha, R. J., 1972: Generalized Simulation Technique for Turbojet Engine System Analysis. NASA TN D-6610, National Aeronautics and Space Administration, Washington, D.C., 63 pp.

2. Koenig, R. W., and Fishback, L. H., 1972: GENENG - A Program for Calculating Design and Off-Design Performance for Turbojet and Turbofan Engines. NASA TN D-6552, National Aeronautics and Space Administration, Washington, D.C., 158 pp.

3. Fishbach, L. H., and Koenig, R. W., 1972: GENENG II - A Program for Calculating Design and Off-Design Performance of Two- and Three-Spool Turbofans with as Many as Three Nozzles. NASA TN D-6553, National Aeronautics and Space Administration, Washington, D.C., 184 pp.

4. McKinney, J. S., 1967: Simulation of Turbofan Engine. Part I: Description of Method and Balancing Technique. AFAPL-TR-67-125, Pt. 1, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio, 38 pp.

5. McKinney, J. S., 1967: Simulation of Turbofan Engine. Part II: User's Manual and Computer Program Listing. AFAPL-TR-67-125, Pt. 2, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio, 94 pp.

6. Szuch, J. R.: HYDES - A Generalized Hybrid Computer Program for Studying Turbojet or Turbofan Engine Dynamics. NASA TM X-3014, 1974.

7. Carnahan, B., Luther, H. A., and Wilkes, J. O., "Applied Numerical Methods", John Wiley & Sons Inc., New York, 1969.

8. Ogata, K., "State Space Analysis of Control Systems", Prentice-Hall, Englewood Cliffs, 1967.

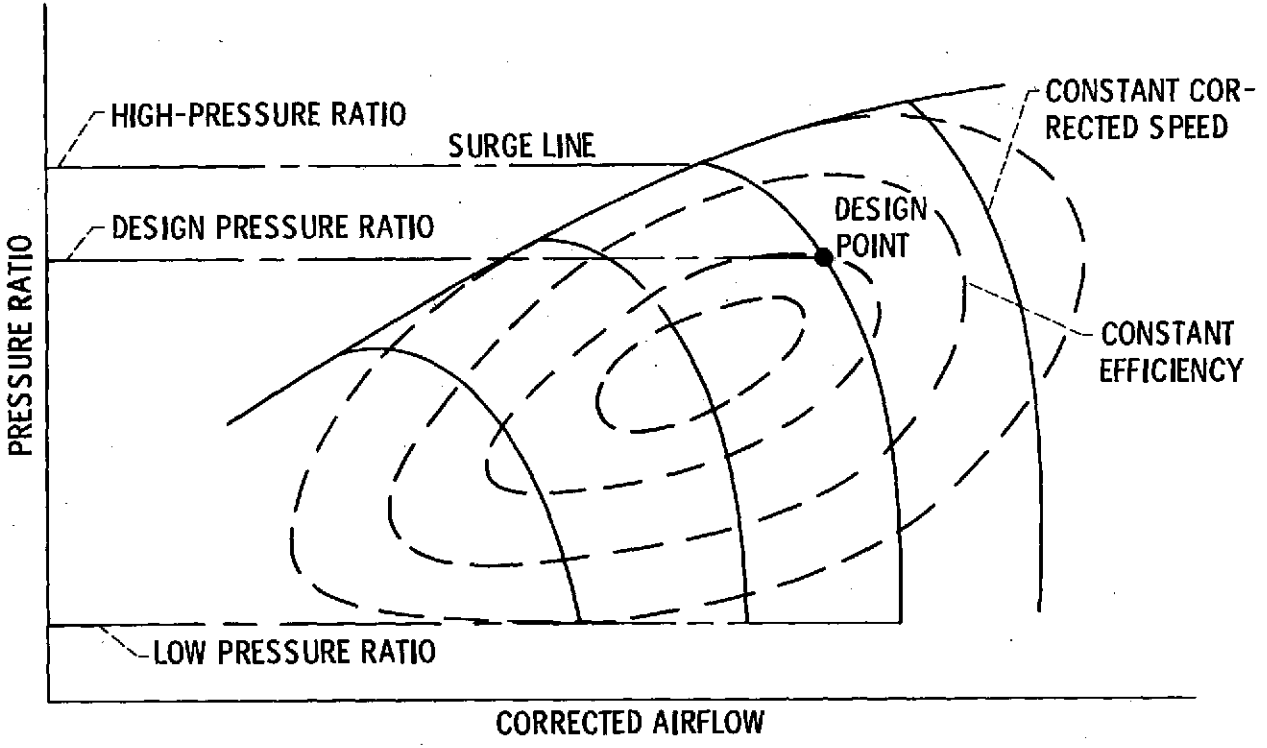


Figure 1. - Fan or compressor map.

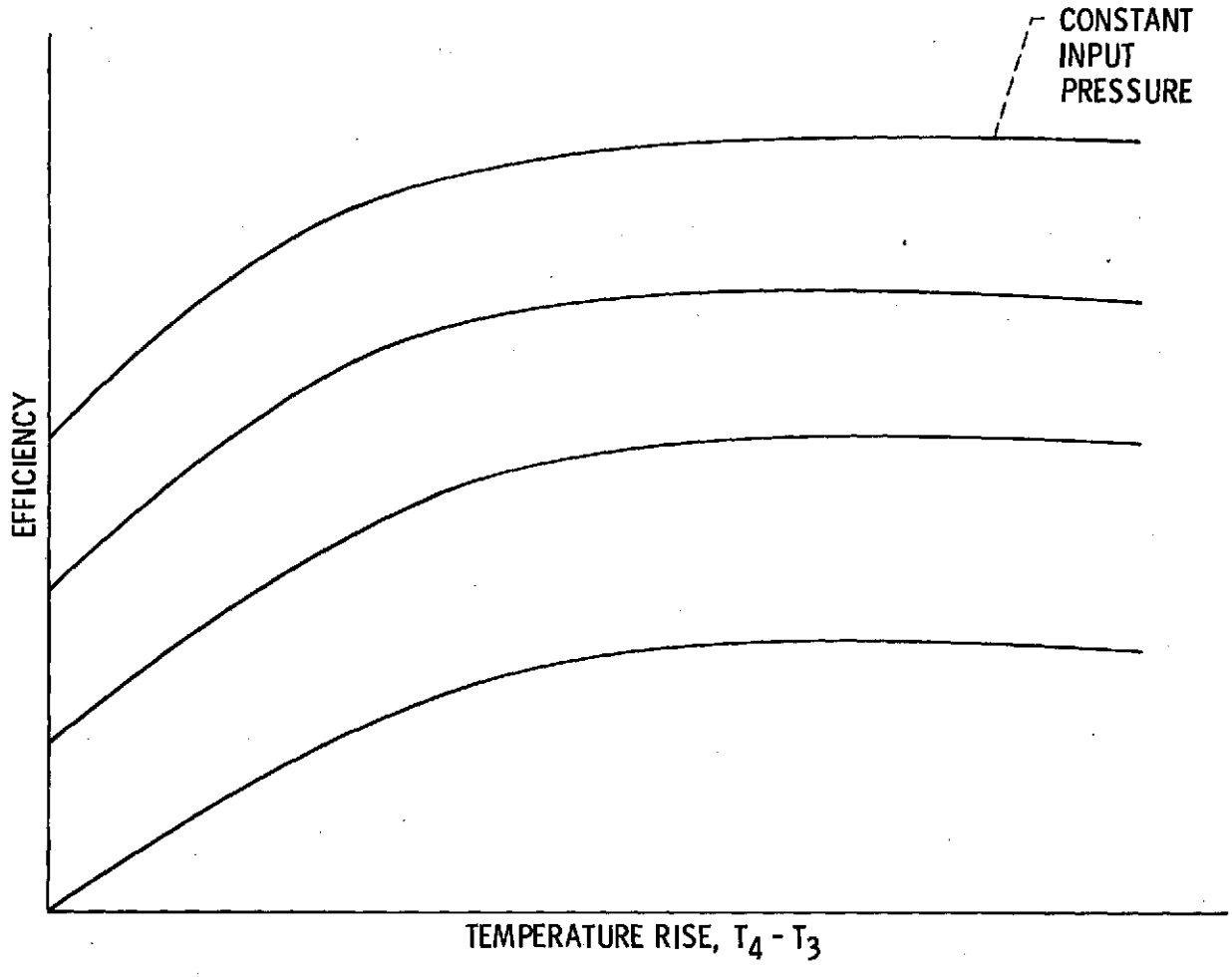


Figure 2. - Combustor map.

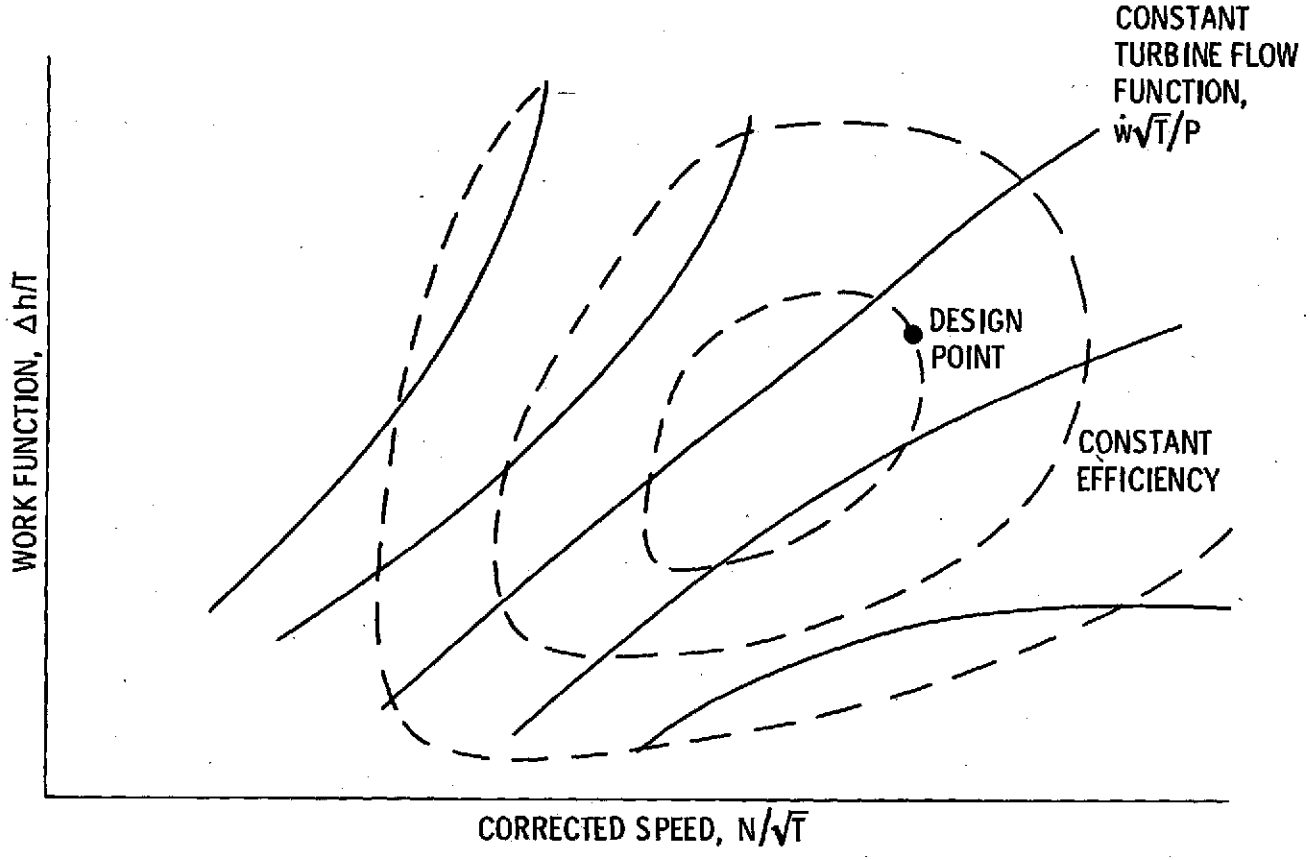
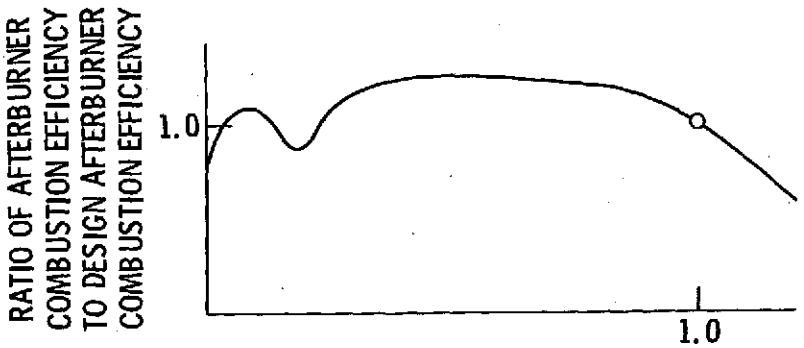
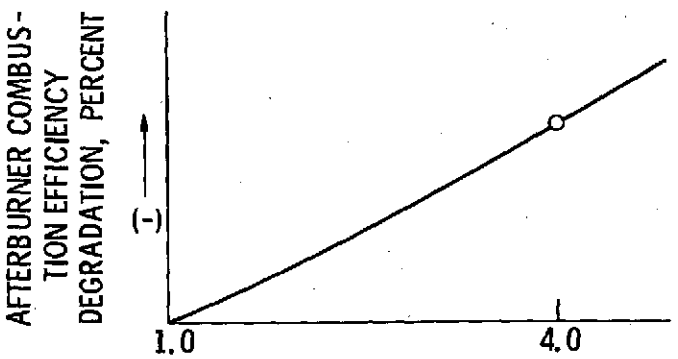


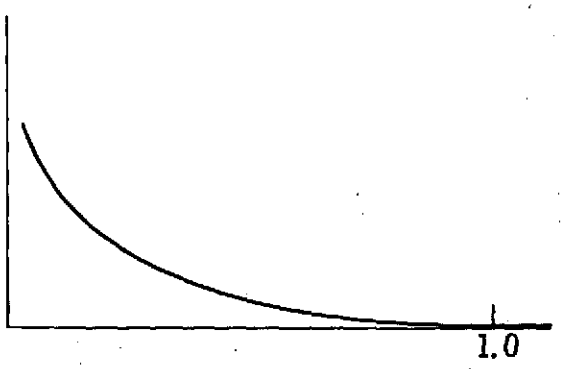
Figure 3. - Turbine map.



(a) GENERALIZED AFTERBURNER COMBUSTION EFFICIENCY AS FUNCTION OF FUEL-AIR RATIO.

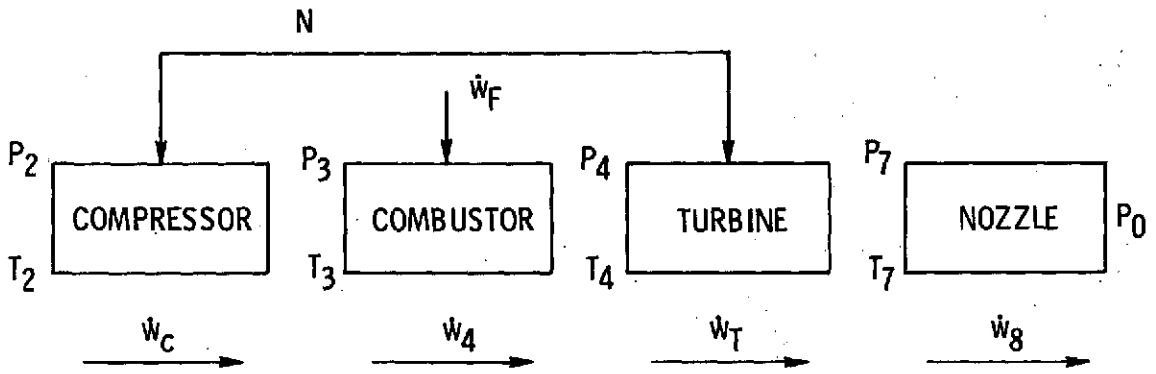


(b) EFFICIENCY CORRECTION FACTOR AGAINST AFTERBURNER INLET MACH NUMBER.



(c) EFFICIENCY CORRECTION FACTOR AGAINST AFTERBURNER INLET TOTAL PRESSURE.

Figure 4. - Afterburner maps.



GUESSED VARIABLES, V_i	ERROR VARIABLES, E_i
SPEED, N	POWER, $\dot{w}_c \Delta h_c - \dot{w}_T \Delta h_T$
COMPRESSOR PRESSURE RATIO, P_3/P_2	TURBINE FLOW FUNCTION, $\frac{\dot{w}_4\sqrt{T_4}}{P_4} - \left(\frac{\dot{w}_4\sqrt{T_4}}{P_4}\right)^1$
TURBINE FLOW FUNCTION, $\frac{\dot{w}_4\ \overline{T_4}}{P_4}$	NOZZLE PRESSURE, $P_7 - P_7'$

E-7968

Figure 5. - Steady-state engine calculations for a turbojet.

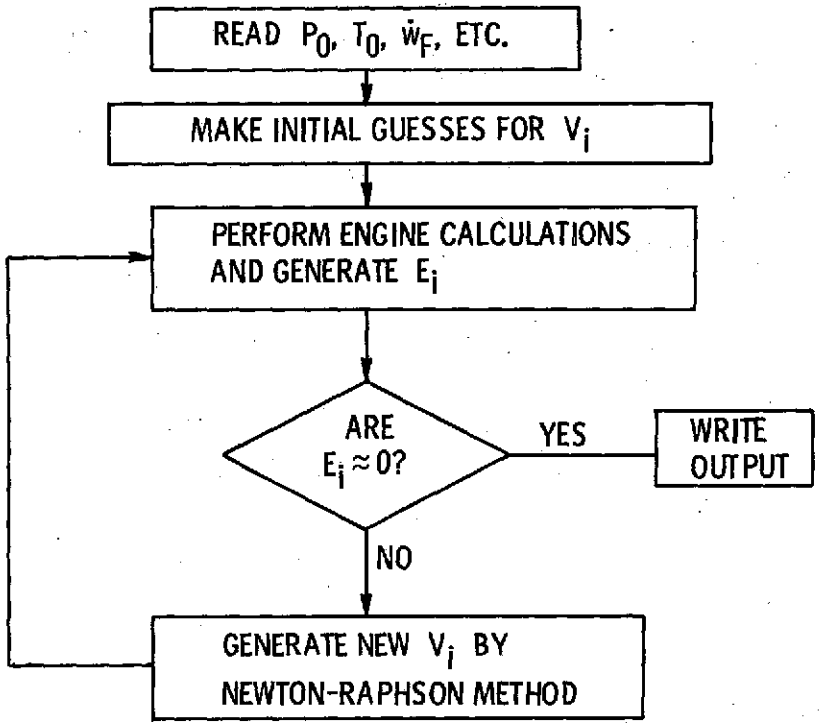


Figure 6. - Flow chart of balancing technique.

POWER BALANCE:

STEADY STATE

$$\dot{w}_T \Delta h_T = \dot{w}_c \Delta h_c$$

DYNAMIC

$$\dot{w}_T \Delta h_T = \dot{w}_c \Delta h_c + JN \frac{dN}{dt}$$

CONTINUITY:

STEADY STATE

$$\dot{w}_{OUT} = \dot{w}_{IN}$$

DYNAMIC

$$\dot{w}_{OUT} = \dot{w}_{IN} - \frac{V}{\gamma RT} \frac{dP}{dt}$$

ENERGY:

STEADY STATE

$$\dot{w}_{OUT} h_{OUT} = \dot{w}_{IN} h_{IN}$$

DYNAMIC

$$\dot{w}_{OUT} h_{OUT} = \dot{w}_{IN} h_{IN} - (\dot{w}_{IN} - \dot{w}_{OUT}) u - \frac{PV}{RT} \frac{du}{dt}$$

Figure 7. - Simulation dynamics.

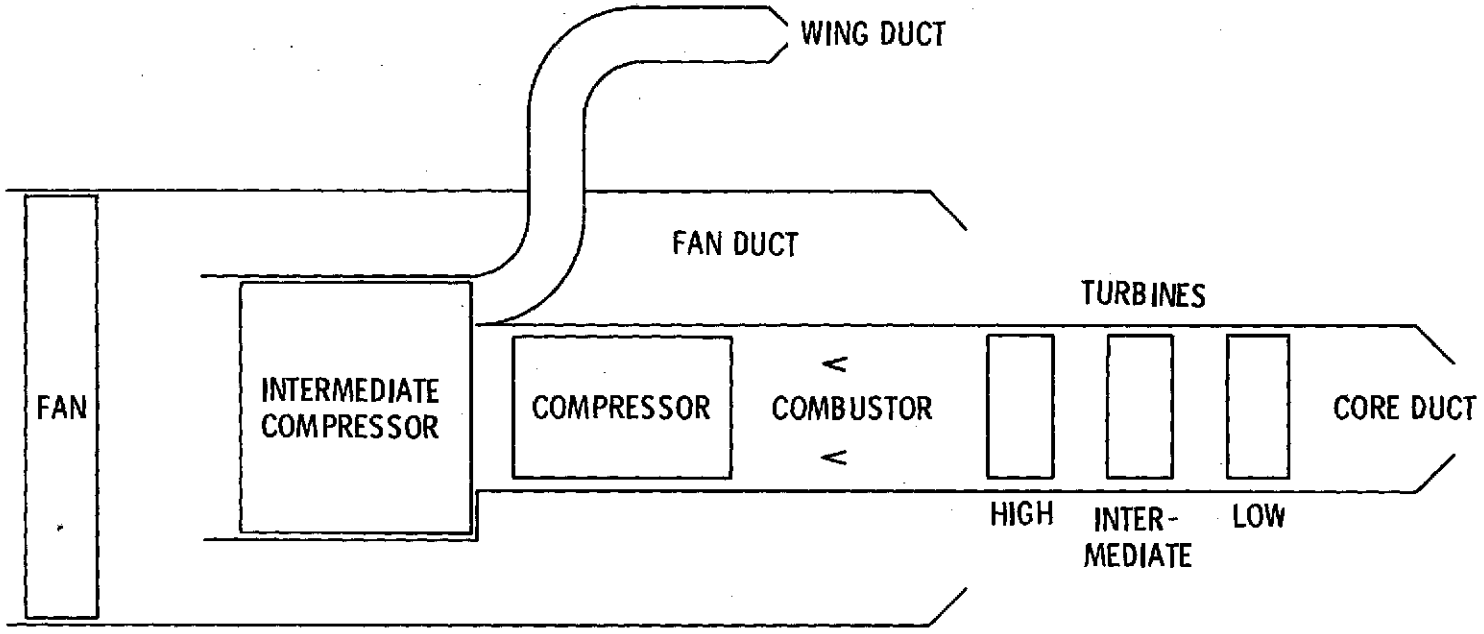


Figure 8. - Three-spool, three-stream engine.

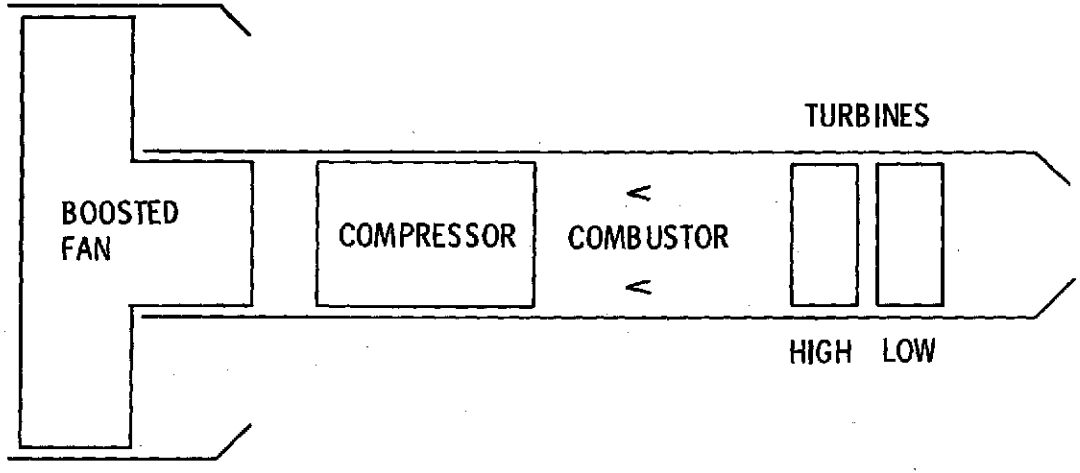


Figure 9. - Two-spool boosted fan.

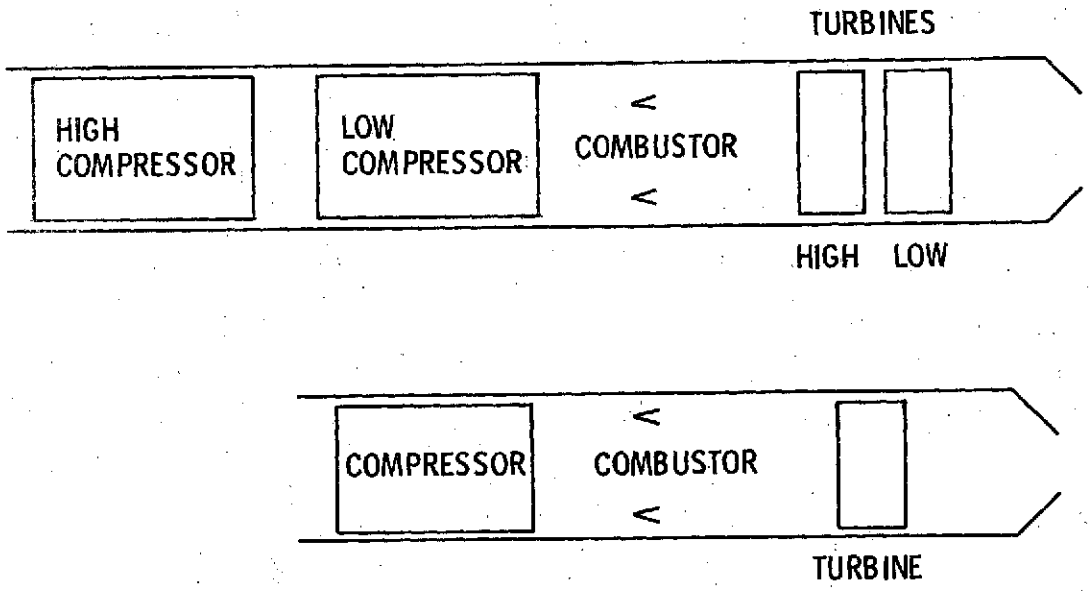


Figure 10. - One- and two-spool turbojets.

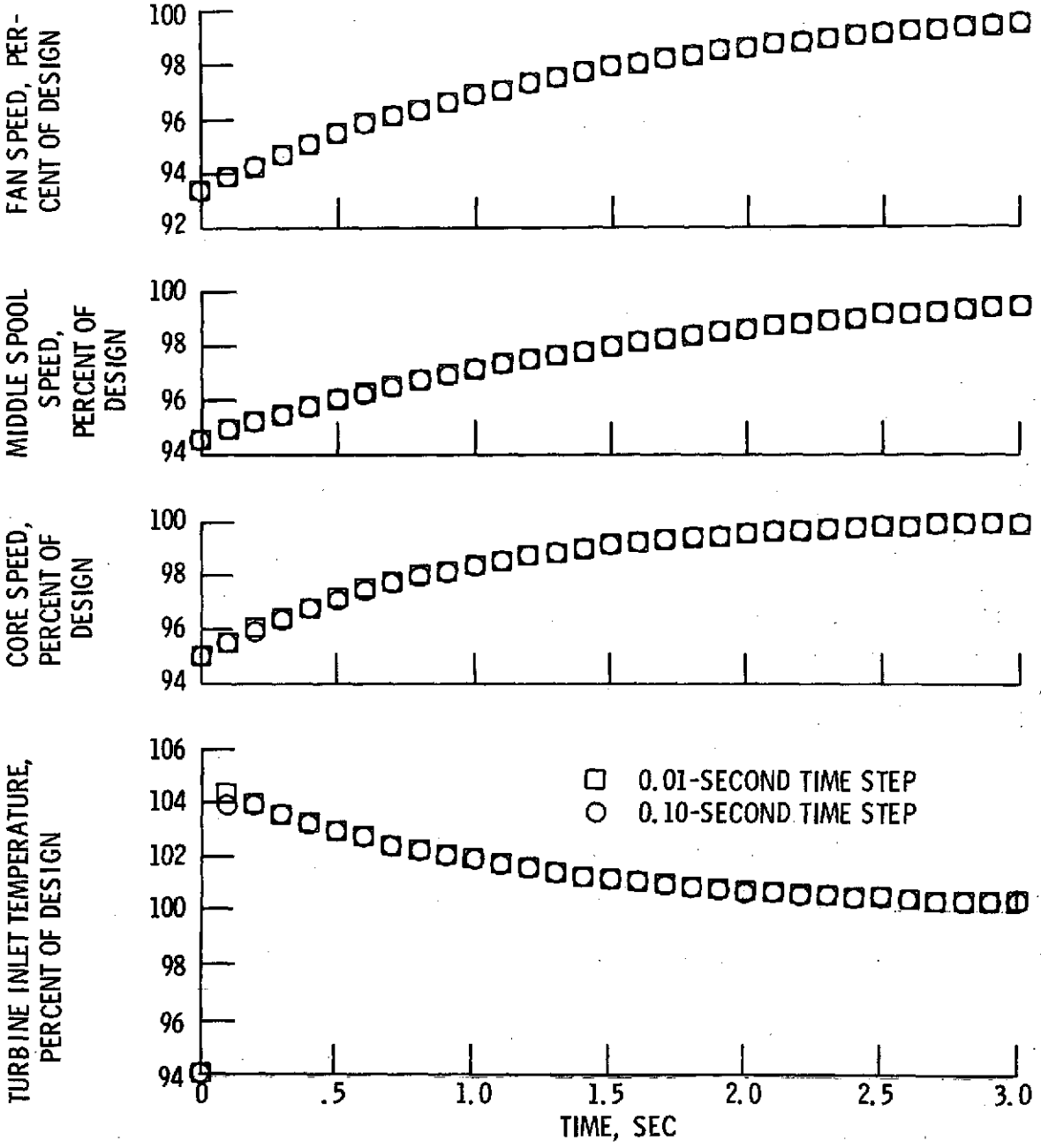
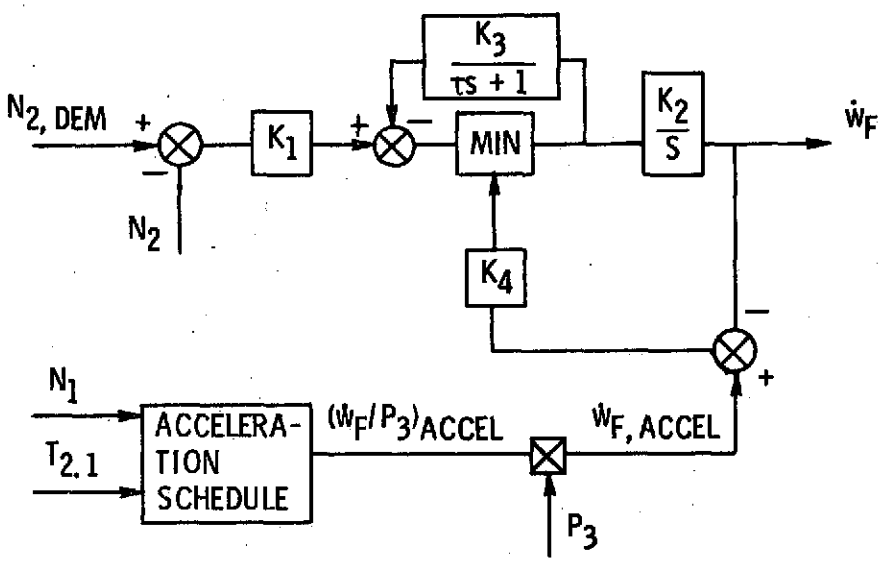


Figure 11. - Three-spool turbofan; response to fuel flow step.

E-7968



N_2 FAN SPEED

N_1 CORE SPEED

$T_{2.1}$ COMPRESSOR INLET TEMPERATURE

P_3 COMPRESSOR EXIT PRESSURE

$\dot{w}_F$ FUEL FLOW

Figure 12. - Two-spool turbofan speed control.

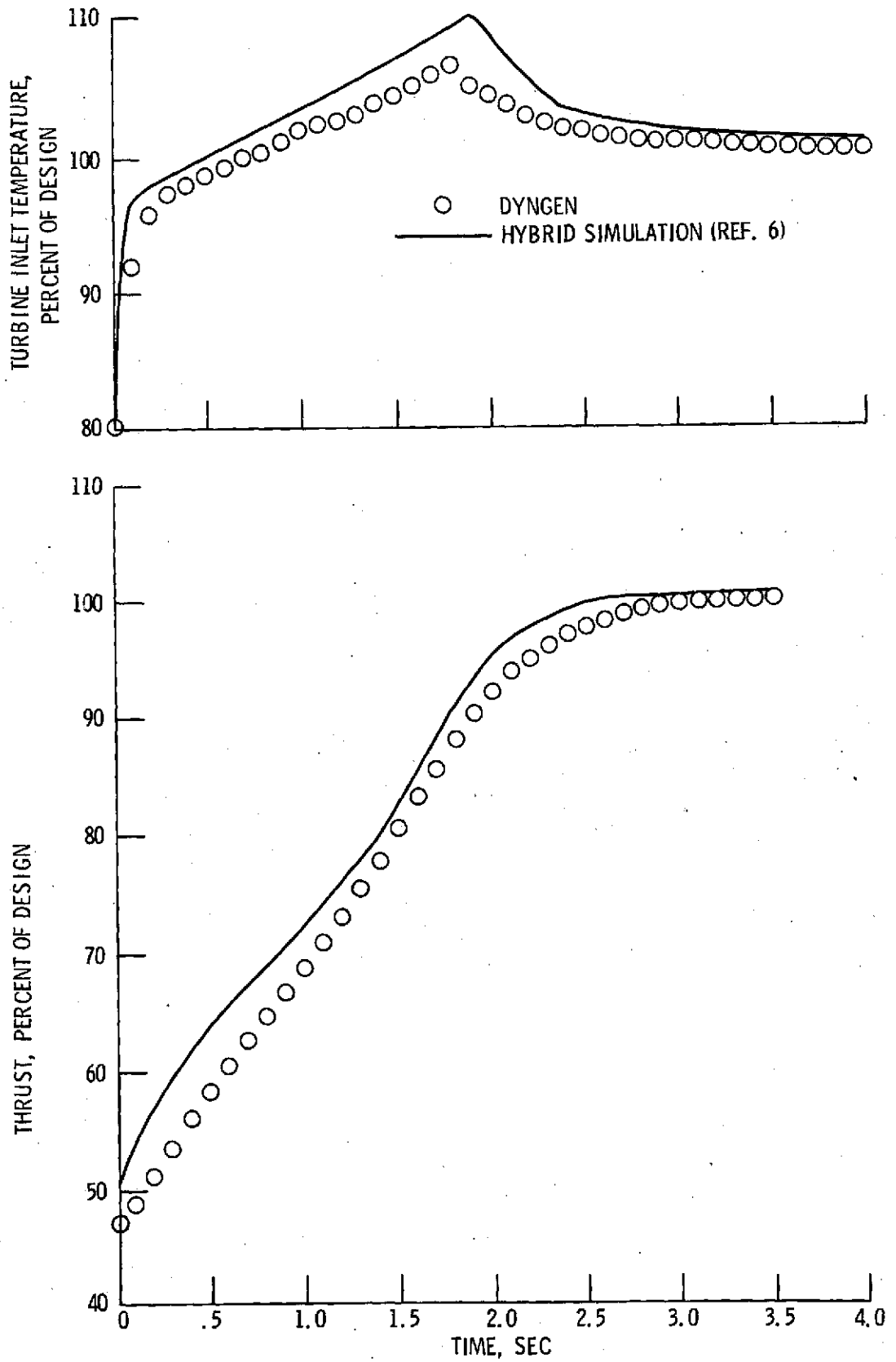


Figure 13. - Two-spool turbofan response to throttle step.

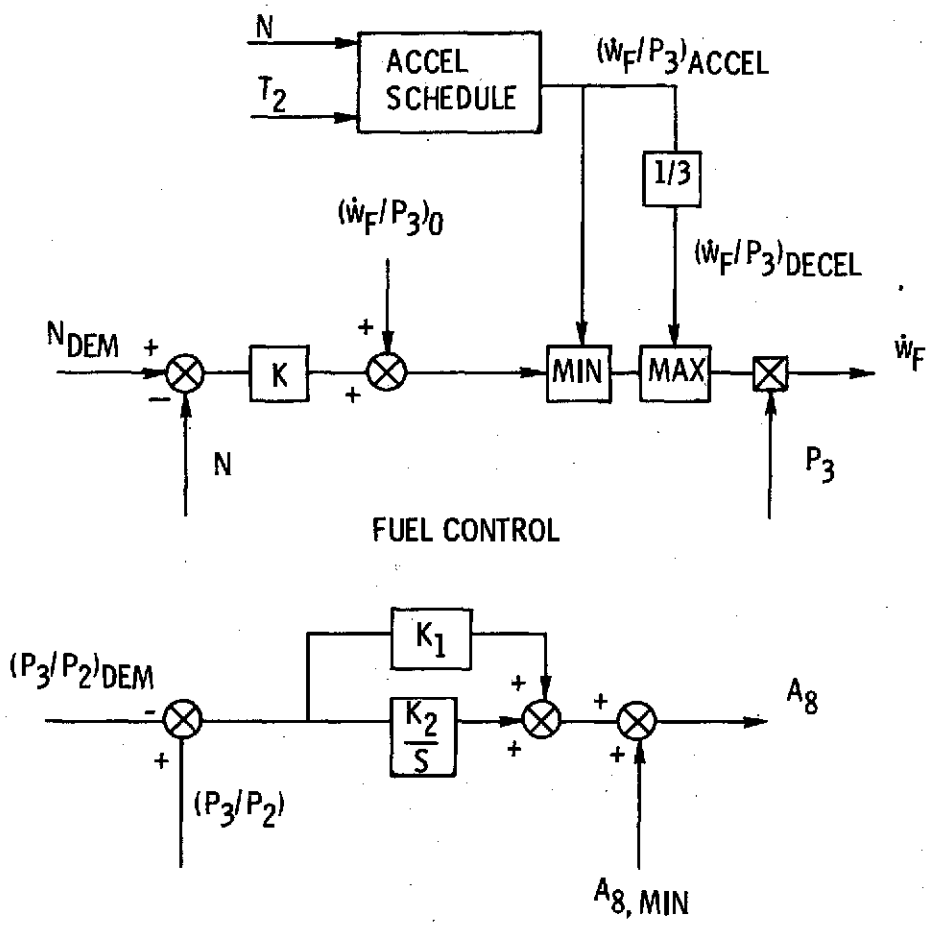


Figure 14. - Afterburning turbojet control system.

E-7968

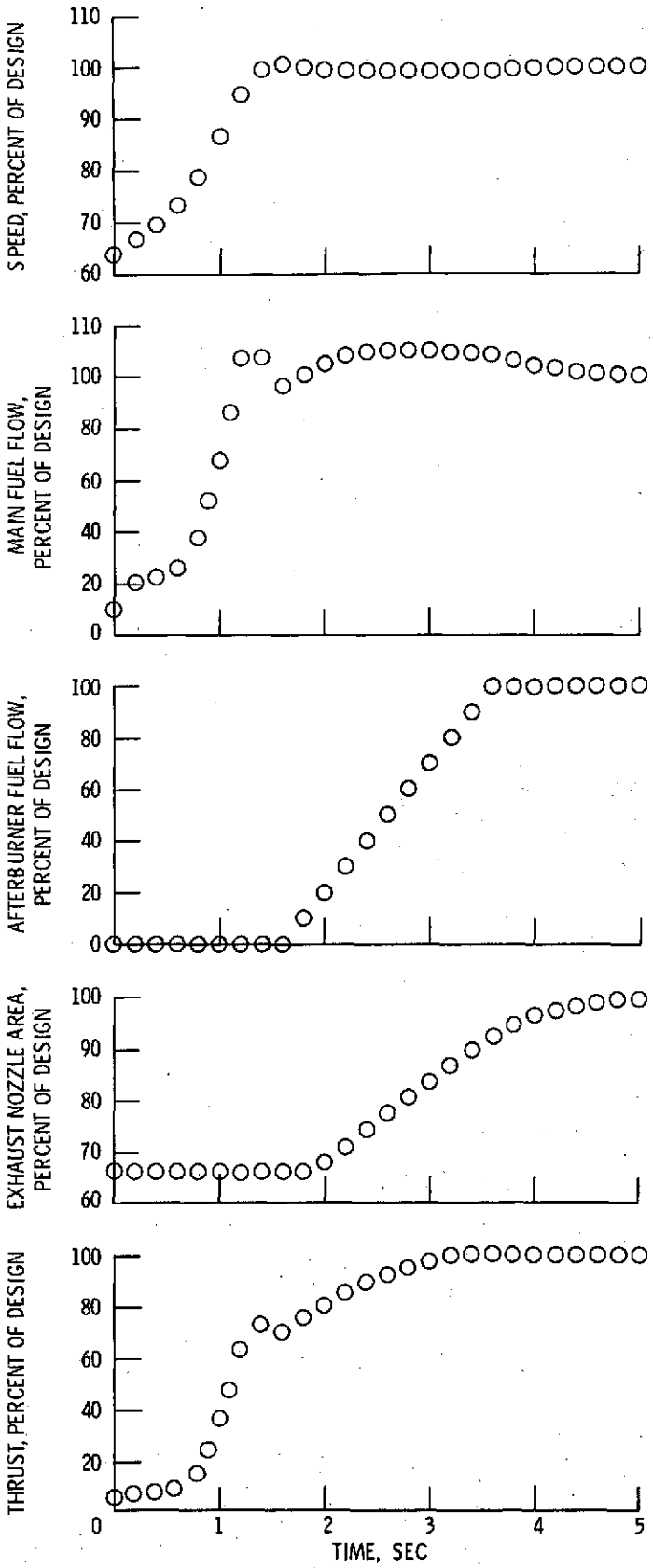


Figure 15. - Afterburning turbojet response to throttle slam.

NASA-Lewis