|  |  |  |
| --- | --- | --- |
| gama_logo_color1 | **Cairo University Faculty of Engineering Aerospace Department** | þÿ |

**Propulsion**

Sheet 3 Problem 2

Optimum FPR Variation with BPR

Submitted to Eng. Abdelraziq

|  |  |  |
| --- | --- | --- |
| **Name:** | ***Mohamed Ahmed Hassan Ahmed*** | **BN: 10 Section: 2**  **Date: 2021 */* 1 */ 5*** |
|  |  |

Table of Contents

[Results Graph 3](#_Toc92243185)

[MatLab Code Implementation 4](#_Toc92243186)

[Clear Workspace 4](#_Toc92243187)

[Definition of Assumption variable 4](#_Toc92243188)

[Definition of Given variables 4](#_Toc92243189)

[Definition of Variables to be calculated from givens & assumption 4](#_Toc92243190)

[Solving by Iteration to obtain tau fan 5](#_Toc92243191)

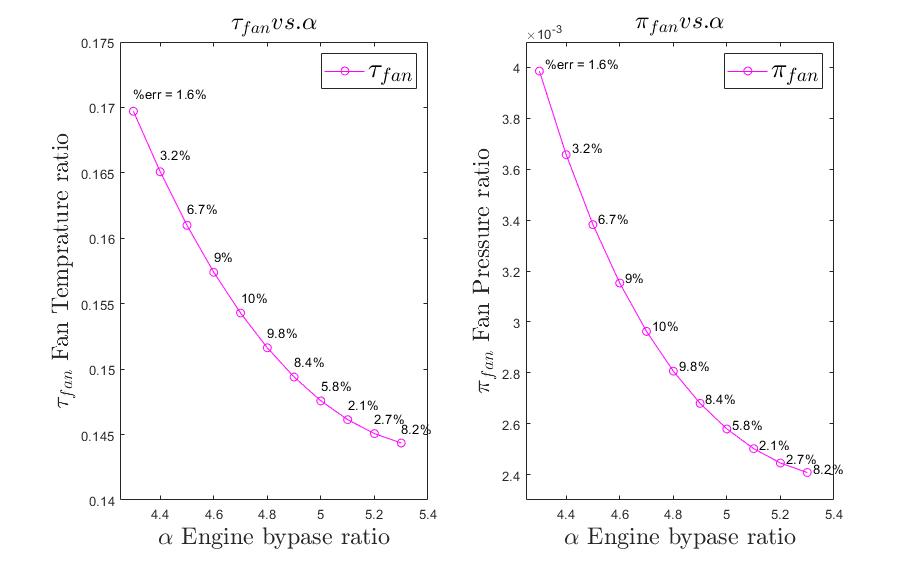
[Plot Results 5](#_Toc92243192)

[Function that Solve for Fan Pressure Ratio by Iteration 6](#_Toc92243193)

# Results Graph

The Percentage Values on the graph represent the percentage of accounted error during iteration.

All Assumptions made with respect to Level of technology 4



# MatLab Code Implementation

## Clear Workspace

clear all  
clc

## Definition of Assumption variable

pi\_diff = 0.995;  
pi\_fan\_diff = 0.995;  
e\_compressor = 0.9;  
e\_fan = 0.89;  
pi\_burner = 0.95;  
eta\_burner = 0.999;  
e\_turbine = 0.9;  
eta\_mechanical = 0.995;  
  
gama\_nozzle = 1.4;  
gama\_compressor = 1.4;  
gama\_turbine = 1.3;  
  
R = 287;  
h\_PR = 42800;

## Definition of Given variables

M\_0 = 0.85;  
T\_t4 = 1500; % kelvin  
T\_0 = 224.752; %kelvin @ 32000 ft  
P\_0 = 27448.9; % pa @ 32000 ft  
pi\_compressor = 33;

## Definition of Variables to be calculated from givens & assumption

Cp\_t = gama\_turbine/(gama\_turbine-1) \* R;  
Cp\_c = gama\_compressor/(gama\_compressor-1) \* R;  
tau\_ramp = (1 + ( (gama\_nozzle-1)/2\*M\_0^2) );  
pi\_ramp = tau\_ramp ^ (gama\_compressor / (gama\_compressor-1));  
tau\_lambda = (Cp\_t\*T\_t4)/(Cp\_c\*T\_0);  
tau\_c = pi\_compressor^( (gama\_compressor-1) / (gama\_compressor\*e\_compressor));  
% First Law of thermodynamics across Burner yields  
f = ((tau\_c\*tau\_ramp) - tau\_lambda) / (tau\_lambda - (eta\_burner\*h\_PR/(Cp\_c\*T\_0)));  
PI\_1 = pi\_burner \* pi\_compressor \* pi\_diff \* pi\_ramp;  
PI\_2 = pi\_fan\_diff \* pi\_ramp;  
A = (tau\_lambda/ (tau\_ramp-1)) \* ( (gama\_turbine-1) / (gama\_nozzle-1)) \* PI\_1 ^((1-gama\_nozzle)/gama\_nozzle);  
B = (tau\_lambda / (tau\_ramp-1)) \* PI\_2^((1-gama\_compressor)/gama\_compressor);

## Solving by Iteration to obtain tau fan

acceptable\_error = 11; % percentage  
alpha = 4.3:0.1:5.3; % initial bypass ratio  
tau\_fan = zeros(1, 10);  
pi\_fan = zeros(1, 10);  
error\_ = zeros(1, 10);  
for i = 1:length(alpha)  
 [tau\_fan\_, err] = solveTauFan(alpha(i), acceptable\_error, tau\_lambda, tau\_ramp, tau\_c, eta\_mechanical, f, ...  
 gama\_nozzle, gama\_turbine, gama\_compressor, e\_turbine, e\_fan, PI\_1, PI\_2, A, B);  
 error\_(i) = round(err, 1);  
 tau\_fan(i) = tau\_fan\_;  
 pi\_fan(i) = tau\_fan\_^(gama\_compressor \* e\_fan / (gama\_compressor-1));  
end

## Plot Results

fig = figure('Name','Engine pressure ratio vs. bypass ratio','NumberTitle','off');  
subplot(1, 2, 1);  
plot(alpha, tau\_fan, '-mo');  
title('$\tau\_{fan} vs. \alpha$', 'interpreter', 'latex','FontSize',18);  
legend('$\tau\_{fan}$', 'interpreter', 'latex', 'fontSize', 20);  
xlim([4.25 5.4]);  
ylim([0.14 0.175]);  
xlabel('$\alpha$ Engine bypase ratio', 'interpreter', 'latex', 'fontSize', 18);  
ylabel('$\tau\_{fan}$ Fan Temprature ratio', 'interpreter', 'latex', 'fontSize', 18);  
for i=1:length(alpha)  
 e = num2str(error\_(i)) + "%";  
 if i == 1  
 e = "%err = " + e;  
 end  
 text(alpha(i), tau\_fan(i) + 0.008\*tau\_fan(i), cellstr(e), 'fontSize', 10);  
end  
  
  
subplot(1, 2, 2);  
plot(alpha, pi\_fan, '-mo');  
title('$\pi\_{fan} vs. \alpha$', 'interpreter', 'latex','FontSize',18);  
legend('$\pi\_{fan}$', 'interpreter', 'latex', 'fontSize', 20);  
xlim([4.25 5.4]);  
ylim([2.3e-3 4.1e-3]);  
xlabel('$\alpha$ Engine bypase ratio', 'interpreter', 'latex', 'fontSize', 18);  
ylabel('$\pi\_{fan}$ Fan Pressure ratio', 'interpreter', 'latex', 'fontSize', 18);  
for i=1:length(alpha)  
 e = num2str(error\_(i)) + "%";  
 if i == 1  
 e = "%err = " + e;  
 end  
 text(alpha(i)+0.02, pi\_fan(i) + 0.008\*pi\_fan(i), cellstr(e), 'fontSize', 10);  
end

## Function that Solve for Fan Pressure Ratio by Iteration

function [best\_tau\_fan, best\_error] = solveTauFan(alpha, acceptable\_error, tau\_lambda, tau\_ramp, tau\_c, eta\_mechanical, f, ...  
 gama\_nozzle, gama\_turbine, gama\_compressor, e\_turbine, e\_fan, PI\_1, PI\_2, A, B)  
 % Holds the best Values achieve to be returned  
 best\_tau\_fan = 0;  
 best\_error = 100;  
 % initial values  
 error\_ = 100;  
 loopIndex = 0;  
 % initial Guess  
 tau\_fan = ((tau\_lambda/tau\_ramp) - tau\_c + 1 + alpha)/(1+alpha);  
 while (error\_ > acceptable\_error && loopIndex < 200)  
 if(loopIndex > 0)  
 tau\_fan = tau\_fan - 0.01;  
 if(tau\_fan <= 0)  
 break  
 end  
 end  
 % Compressor, Turbine & Fan Power Balance yields  
 tau\_turbine = 1 - ( (tau\_ramp/tau\_lambda)/ (eta\_mechanical\*(1+f)) ) ...  
 \*((tau\_c-1) + (alpha\*tau\_fan-1));  
  
 % subsitute PHI\_1 of tau Turbine  
 gama\_nozzel\_number = (gama\_nozzle -1) / gama\_nozzle;  
 tau\_turb\_power = (gama\_turbine\*(gama\_nozzle-1)) / ...  
 (gama\_nozzle\*(gama\_turbine-1)\*e\_turbine);  
 PHI\_1 = ((-alpha \*tau\_ramp\*A^0.5) / (2\*tau\_lambda\*eta\_mechanical\*(1+f))) ...  
 \*( (PI\_1^gama\_nozzel\_number \* tau\_turbine) - tau\_turbine^(1-tau\_turb\_power) )^-1 ...  
 \*(PI\_1^gama\_nozzel\_number - (1- tau\_turb\_power) \* (tau\_turbine^(-tau\_turb\_power)))^0.5;  
 % subsitute PHI\_2 of tau Fan  
 gama\_comp\_number = (gama\_compressor -1) / gama\_compressor;  
 PHI\_2 = 0.5\*B^0.5 \* (PI\_2^gama\_comp\_number - (1-e\_fan) \* tau\_fan^(-e\_fan)) ...  
 / ( (PI\_2^gama\_comp\_number\*tau\_fan) - tau\_fan^(1-e\_fan) )^0.5;  
  
 % Condition of optimum fuel consumption  
 a1 = PHI\_1 \* (1+f) / alpha;  
 a2 = -PHI\_2;  
 % Calculate Abs Error  
 error\_ = ( 1 - abs( min(a1, a2) / max(a1, a2) ) )\* 100;  
 if(error\_ < best\_error)  
 % Update best\_error to be returned  
 best\_error = error\_;  
 best\_tau\_fan = tau\_fan;  
 end  
  
 % Increament Loop Indexer  
 loopIndex = loopIndex + 1;  
 end  
end