Cairo University
Faculty of Engineering
Aerospace Department

AEROSPACE ENGINEERING

Propulsion

Sheet 3 Problem 2
Optimum FPR Variation with BPR

Submitted to Eng. Abdelraziq

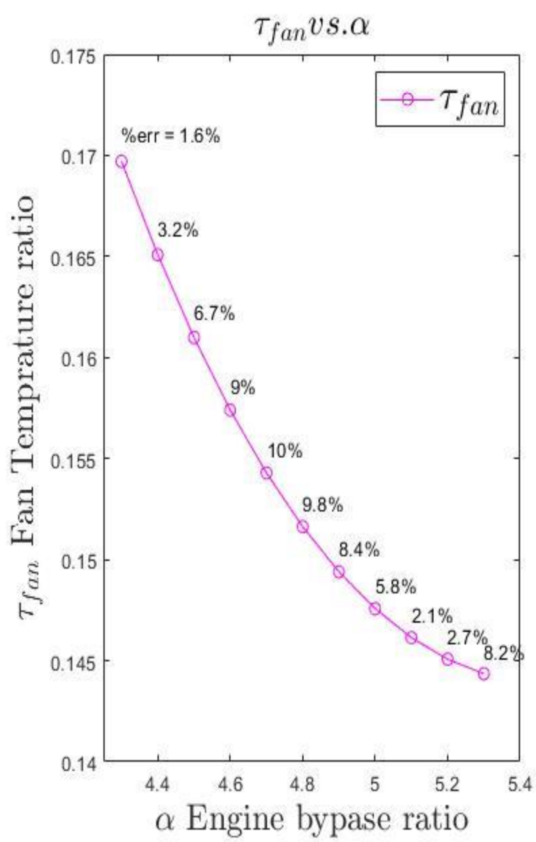
Name: ***Mohamed Ahmed Hassan Ahmed*** **BN: 10** **Section: 2**

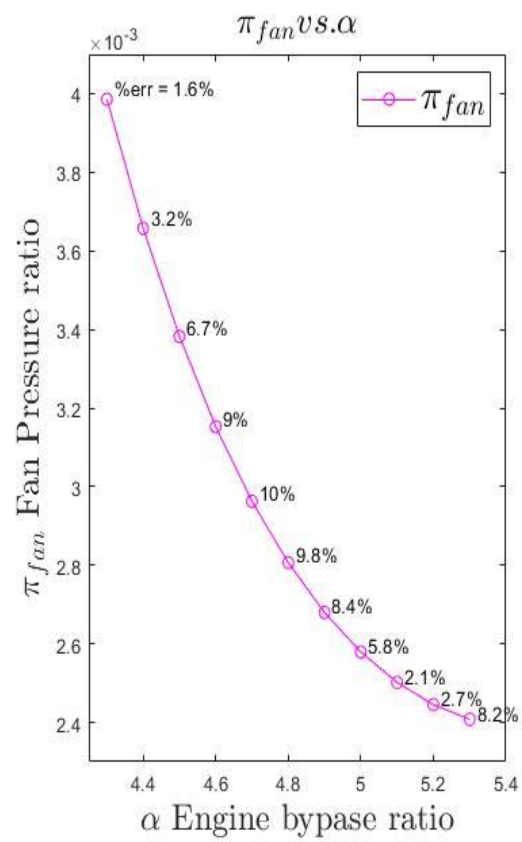
Date: 2021 / 1 / *5*

Table of Contents

Results Graph

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
```