HYPERSONIC FLOWS

Table of Contents

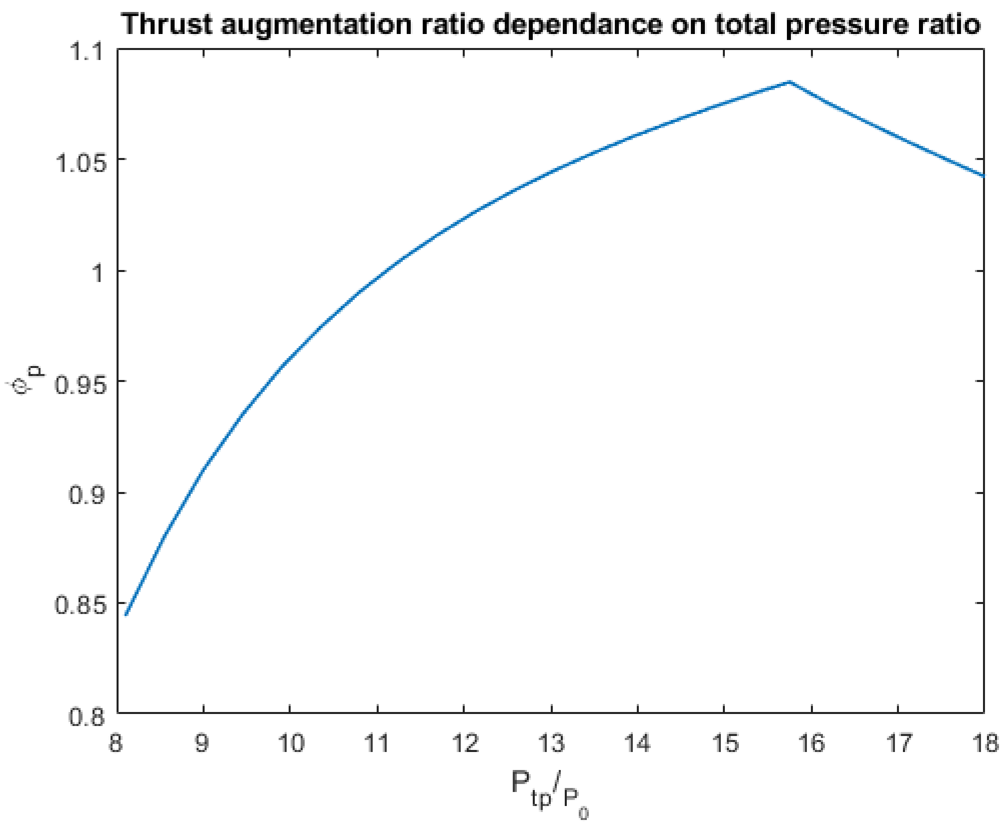
EX. 8.10 PAGE 475

```
clear all
close all
```

Varying P_tp/P_0

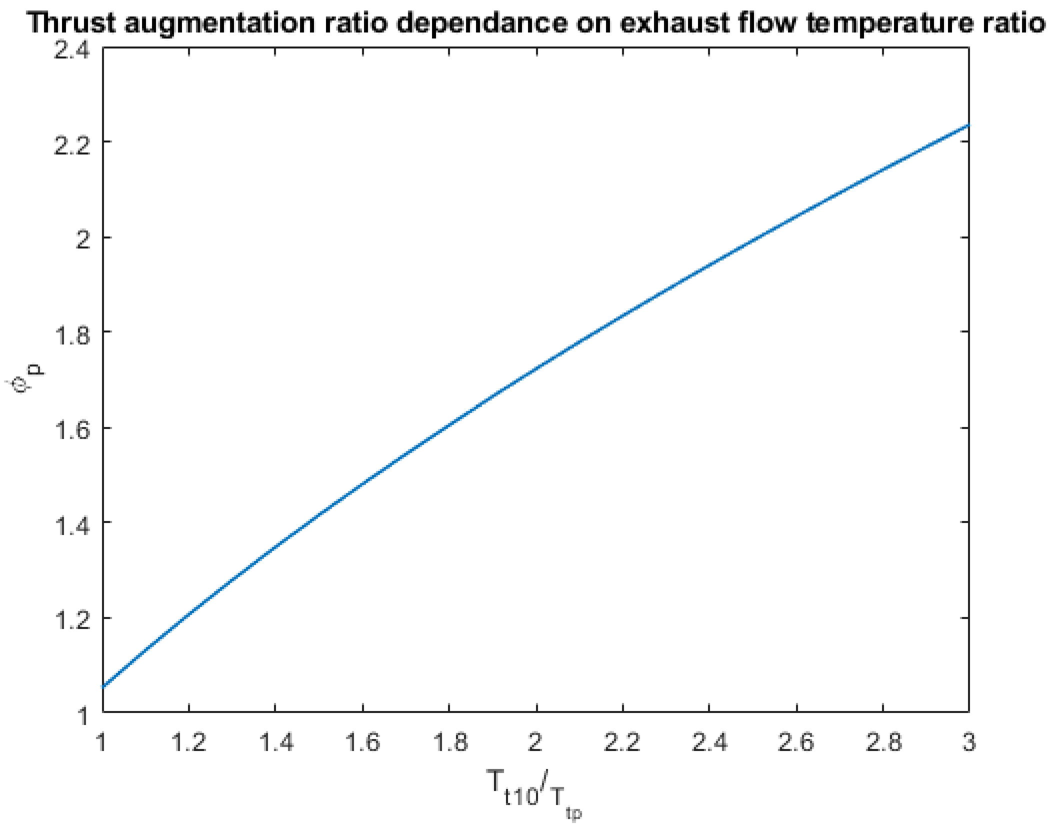
```
M02=2;
g=1.35; %heat air coefficient

Cf=0.0027; %friction coefficient (reasonable value)
Aw=pi*0.5^2; %duct area (1 meter diameter)
v3=295; % @20000 meters of altitude (for Mach 1) [m/s]
R=8.134; %[J/mol*K]
Rs=287.058; % [J/kg*K]
T3=213.31; %temperature at 20000m
V=Aw*1; %volume of a 1 meter cylinder [m^3]
n=V/0.0224; %number of moles of gas
P=n*R*T3/V; %Pa
rho=P/(Rs*T3); %[kg/m^3]
Fbx=-Cf*rho*v3*v3*Aw/2; %additional momentum force due to wall
 friction
P0=5474.89; % atmospheric pressure at 20000m

%Primary flow
pp_p02=0.9*[9:0.5:20];  %pressure ratio
tp_t02=10;          %temperature ratio
a_ap2=12;           %area ratio

% Secondary flow
ps_p02=0.9*1.18;
ts_t02=1.044;
% Exhaust flow
p10_p02=1;
t10_tp2=1;
dp2=.0001;
```

```
pi_p02=zeros(10000);
pi_p02(1)=(2/(g+1))^(g/(g-1))+dp2;   %inlet plane static pressure
                                    %initial guess

for i=1:length(pp_p02)
    for j=1:10000
            %eq. 8.26
            mp2(i,j)=sqrt((2/(g-1))*((pp_p02(i)/pi_p02(j))^((g-1)/
g)-1));
            %eq. 8.27
            api_aps2(i,j)=(1/mp2(i,j))*((2/(g+1))*(1+((g-1)/2)*...
                mp2(i,j)^2))^((g+1)/(2*(g-1)));
            %eq. 8.28
            api_a2(i,j)=api_aps2(i,j)/a_ap2;
            %eq. 8.29
            asi_a2(i,j)=1-api_a2(i,j);
            %eq. 8.30
            msi2(i,j)=sqrt((2/(g-1))*((ps_p02/pi_p02(j))^((g-1)/
g)-1));
            %eq. 8.31 (bypass ratio eqn)
            alpha2(i,j)=ps_p02/pp_p02(i)*asi_a2(i,j)/api_a2(i,j)*...
                (msi2(i,j)/mp2(i,j))*sqrt(tp_t02/
ts_t02)*((1+((g-1)/2)...
                *mp2(i,j)^2)/(1+((g-1)/2)*msi2(i,j)^2))^((g+1)/
(2*(g-1)));
            %eq. 8.32
            te_tp2(i,j)=(2/(g+1))*((1+alpha2(i,j)*ts_t02/tp_t02)/...
                (1+alpha2(i,j)));
            %eq. 8.33
            pe_p02(i,j)=(1+alpha2(i,j))*pp_p02(i)/a_ap2*...
                sqrt(te_tp2(i,j))*(2/(g+1))^((g+1)/(2*(g-1)));
            %calculating the ratio at eqn 8.34
            nr2(i,j)=pe_p02(i,j)*(g+1);
            %the denominator represents the left-hand side of the
 equation,
            %which is basically the condition before the burner. For
 ex.
            %8.10 the additional momentum component was added (F_bx)
            dr2(i,j)=pi_p02(j)*mp2(i,j)^2*g*api_a2(i,j)+pi_p02(j)*...
                msi2(i,j)^2*g*asi_a2(i,j)+pi_p02(j)+Fbx/P0;
            ratio2(i,j)=nr2(i,j)/dr2(i,j);
            %in order to obtain a correct result, the ratio must be
 equal
            %to 1 (or very close to it!). Hence, we are going to know
 that
            %our inlet pressure guess was correct.
            if (abs(ratio2(i,j))-1)<10e-5
                pratio2(i)=pi_p02(j);
                %eqn 8.35
                pte_p02(i)=0.9*pe_p02(i,j)*((g+1)/2)^(g/(g-1));
                k2(i)=alpha2(i,j);
                %eq. 8.37
                mp02(i)=sqrt((2/(g-1))*(pp_p02(i)^((g-1)/g)-1));
                %eq. 8.38
                v0_vp02(i)=(M02/mp02(i))*(ts_t02/
tp_t02*((1+((g-1)/2)...
                    *mp02(i)^2)/(1+((g-1)/2)*M02^2)))^(1/2);
                %eq. 8.39
                m102(i)=sqrt((2/(g-1))*(pte_p02(i)^((g-1)/g)-1));
                %eq. 8.40
                v10_vp02(i)=(m102(i)/
mp02(i))*(t10_tp2*((1+((g-1)/2)*...
                    mp02(i)^2)/(1+((g-1)/2)*m102(i)^2)))^(1/2);
                %eq. 8.36
                phi2(i)=(1+k2(i))*v10_vp02(i)-k2(i)*v0_vp02(i);
                break
            else
                %incrementing of a small value our initial guess in
 case
                %our ratio did not satisfy the unity
                pi_p02(j+1)=pi_p02(j)+dp2;
            end
    end
end
%plotting
figure(2)
plot(pp_p02,phi2,'LineWidth',1.2)
title('Thrust augmentation ratio dependance on total pressure ratio')
xlabel('{P_t_p}/_{P_0}')
ylabel('\phi_p')
```



Varying T_t10/T_p

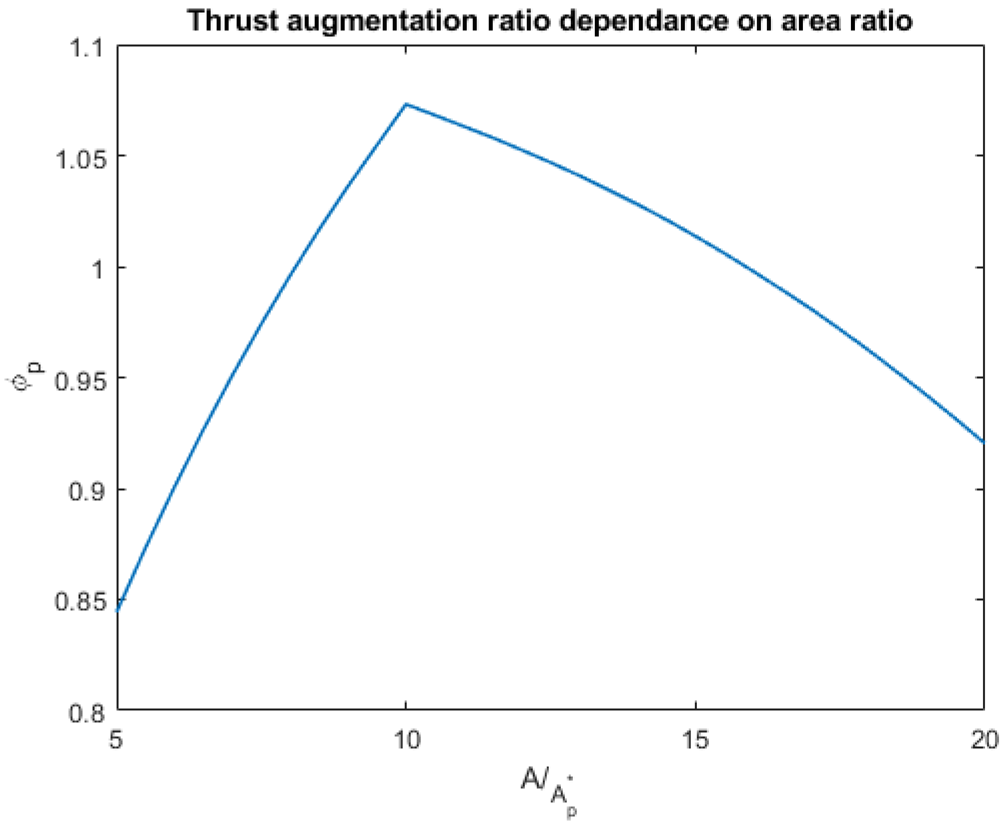
```
M0=2;           %mach numbers

% Primary flow
pp_p0=0.9*15;
tp_t0=10;
a_ap=12;
g=1.35;

% Secondary flow
ps_p0=0.9*1.18;
ts_t0=1.044;

% Exhaust flow
p10_p0=1;
t10_tp=[1:0.1:3];


dp=.0001;
pi_p0(1)=(2/(g+1))^(g/(g-1))+dp;
for i=1:10000
        mp(i)=sqrt((2/(g-1))*((pp_p0/pi_p0(i))^((g-1)/g)-1));
        api_aps(i)=(1/mp(i))*((2/(g+1))*(1+((g-1)/2)*...
            mp(i)^2))^((g+1)/(2*(g-1)));
        api_a(i)=api_aps(i)/a_ap;
        asi_a(i)=1-api_a(i);
        msi(i)=sqrt((2/(g-1))*((ps_p0/pi_p0(i))^((g-1)/g)-1));
        alpha(i)=ps_p0/pp_p0*asi_a(i)/api_a(i)*(msi(i)/mp(i))*...
            sqrt(tp_t0/ts_t0)*((1+((g-1)/2)*mp(i)^2)/(1+((g-1)/2)*...
            msi(i)^2))^((g+1)/(2*(g-1)));
        te_tp(i)=(2/(g+1))*((1+alpha(i)*ts_t0/tp_t0)/(1+alpha(i)));
        pe_p0(i)=(1+alpha(i))*pp_p0/a_ap*sqrt(te_tp(i))*(2/(g+1))...
            ^((g+1)/(2*(g-1)));
        nr(i)=pe_p0(i)*(g+1);
        dr(i)=pi_p0(i)*mp(i)^2*g*api_a(i)+pi_p0(i)*...
                msi(i)^2*g*asi_a(i)+pi_p0(i)+Fbx/P0;
        ratio(i)=nr(i)/dr(i);
        if (abs(ratio(i))-1)<10e-5
            pratio=pi_p0(i);
            pte_p0=0.9*pe_p0(i)*((g+1)/2)^(g/(g-1));
            k=alpha(i);
            break
        else
            pi_p0(i+1)=pi_p0(i)+dp;
        end
end

mp0=sqrt((2/(g-1))*(pp_p0^((g-1)/g)-1));
for i=1:length(t10_tp)
    v0_vp0(i)=(M0/mp0)*(ts_t0/tp_t0*((1+((g-1)/2)*mp0^2)/...
        (1+((g-1)/2)*M0^2)))^(1/2);
    m10=sqrt((2/(g-1))*(pte_p0^((g-1)/g)-1));
    v10_vp0=(m10/mp0)*(t10_tp(i)*((1+((g-1)/2)*mp0^2)/...
        (1+((g-1)/2)*m10^2)))^(1/2);
    phi(i)=(1+k)*v10_vp0-k*v0_vp0(i);
end
figure(1)
plot(t10_tp,phi,'LineWidth',1.2)
title('Thrust augmentation ratio dependance on exhaust flow
 temperature ratio')
xlabel('{T_t_1_0}/_{T_t_p}')
ylabel('\phi_p')
```



Varying A/A_p

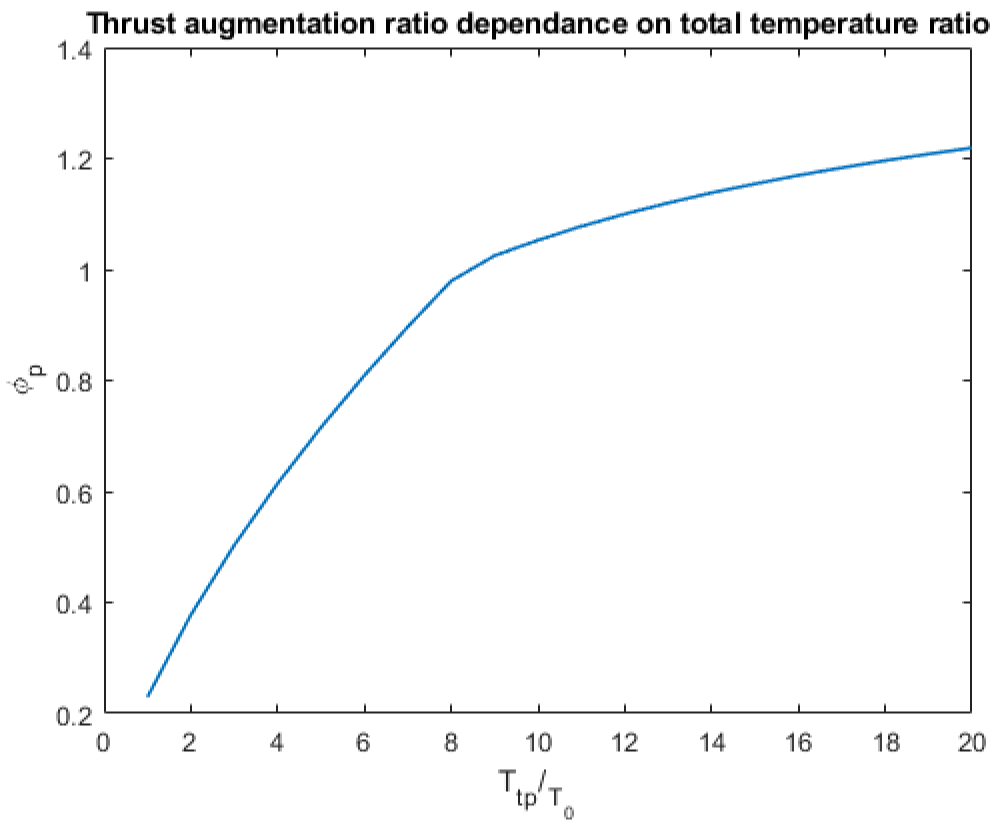
```
M03=2;
%Primary flow
pp_p03=0.9*15;
tp_t03=10;
a_ap3=[5:0.5:20];
g=1.35;
% Secondary flow
ps_p03=0.9*1.18;
ts_t03=1.044;
% Exhaust flow
p10_p03=1;
t10_tp3=1;
dp3=.0001;
```

```
pi_p03=zeros(10000);
pi_p03(:)=(2/(g+1))^(g/(g-1))+dp3;

for i=1:length(a_ap3)
    for j=1:10000
            mp3(i,j)=sqrt((2/(g-1))*((pp_p03/pi_p03(j))^((g-1)/g)-1));
            api_aps3(i,j)=(1/mp3(i,j))*((2/(g+1))*(1+((g-1)/2)...
                *mp3(i,j)^2))^((g+1)/(2*(g-1)));
            api_a3(i,j)=api_aps3(i,j)/a_ap3(i);
            asi_a3(i,j)=1-api_a3(i,j);
            msi3(i,j)=sqrt((2/(g-1))*((ps_p03/pi_p03(j))^((g-1)/
g)-1));
            alpha3(i,j)=ps_p03/pp_p03*asi_a3(i,j)/api_a3(i,j)*...
                (msi3(i,j)/mp3(i,j))*sqrt(tp_t03/ts_t03)*...
                ((1+((g-1)/2)*mp3(i,j)^2)/...
                (1+((g-1)/2)*msi3(i,j)^2))^((g+1)/(2*(g-1)));
            te_tp3(i,j)=(2/(g+1))*((1+alpha3(i,j)*ts_t03/...
                tp_t03)/(1+alpha3(i,j)));
            pe_p03(i,j)=(1+alpha3(i,j))*pp_p03/a_ap3(i)*...
                sqrt(te_tp3(i,j))*(2/(g+1))^((g+1)/(2*(g-1)));
            nr3(i,j)=pe_p03(i,j)*(g+1);
            dr3(i,j)=pi_p03(j)*mp3(i,j)^2*g*api_a3(i,j)+pi_p03(j)*...
                msi3(i,j)^2*g*asi_a3(i,j)+pi_p03(j)+Fbx/P0;
            ratio3(i,j)=nr3(i,j)/dr3(i,j);
            if (abs(ratio3(i,j))-1)<10e-50
                pratio3(i)=pi_p03(j);
                pte_p03(i)=0.9*pe_p03(i,j)*((g+1)/2)^(g/(g-1));
                k3(i)=alpha3(i,j);
                mp03(i)=sqrt((2/(g-1))*(pp_p03^((g-1)/g)-1));
                v0_vp03(i)=(M03/mp03(i))*(ts_t03/
tp_t03*((1+((g-1)/2)...
                    *mp03(i)^2)/(1+((g-1)/2)*M03^2)))^(1/2);
                m103(i)=sqrt((2/(g-1))*(pte_p03(i)^((g-1)/g)-1));
                v10_vp03(i)=(m103(i)/
mp03(i))*(t10_tp3*((1+((g-1)/2)...
                    *mp03(i)^2)/(1+((g-1)/2)*m103(i)^2)))^(1/2);
                phi3(i)=(1+k3(i))*v10_vp03(i)-k3(i)*v0_vp03(i);
                break
            else
                pi_p03(j+1)=pi_p03(j)+dp3;
            end
    end
end
figure(3)
plot(a_ap3,phi3,'LineWidth',1.2)
title('Thrust augmentation ratio dependance on area ratio')
xlabel('{A}/_{A^*_p}')
ylabel('\phi_p')
```



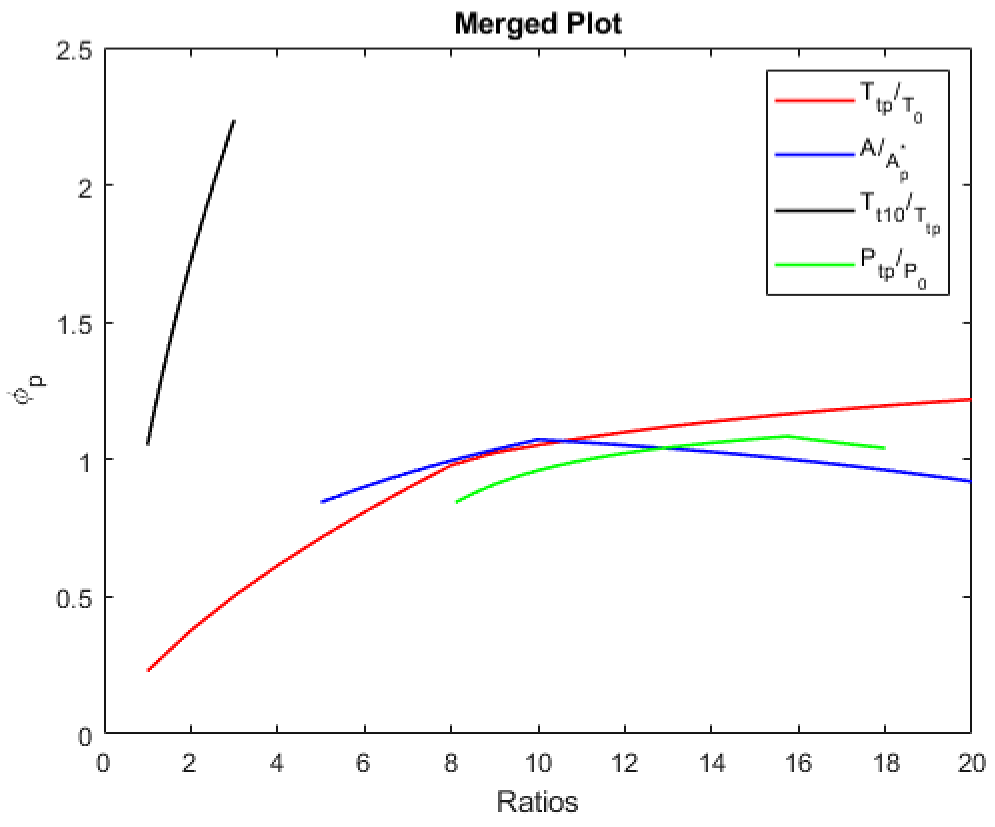
Varying T_tp/T_0

```
M04=2;
%Primary flow
pp_p04=0.9*15;
tp_t04=[1:1:20];
a_ap4=12;
g=1.35;
% Secondary flow
ps_p04=0.9*1.18;
ts_t04=1.044;
% Exhaust flow
p10_p04=1;
t10_tp4=1;
dp4=.0001;
pi_p04=zeros(10000);
pi_p04(:)=(2/(g+1))^(g/(g-1))+dp4;

for i=1:length(tp_t04)
    for j=1:10000
            mp4(i,j)=sqrt((2/(g-1))*((pp_p04/pi_p04(j))^((g-1)/g)-1));
            api_aps4(i,j)=(1/mp4(i,j))*((2/(g+1))*(1+((g-1)/2)...
                *mp4(i,j)^2))^((g+1)/(2*(g-1)));
            api_a4(i,j)=api_aps4(i,j)/a_ap4;
            asi_a4(i,j)=1-api_a4(i,j);
            msi4(i,j)=sqrt((2/(g-1))*((ps_p04/pi_p04(j))^((g-1)/
g)-1));
            alpha4(i,j)=ps_p04/pp_p04*asi_a4(i,j)/api_a4(i,j)*...
                (msi4(i,j)/mp4(i,j))*sqrt(tp_t04(i)/ts_t04)*...
                ((1+((g-1)/2)*mp4(i,j)^2)/...
                (1+((g-1)/2)*msi4(i,j)^2))^((g+1)/(2*(g-1)));
            te_tp4(i,j)=(2/(g+1))*((1+alpha4(i,j)*ts_t04/tp_t04(i))...
                /(1+alpha4(i,j)));
            pe_p04(i,j)=(1+alpha4(i,j))*pp_p04/
a_ap4*sqrt(te_tp4(i,j))*...
                (2/(g+1))^((g+1)/(2*(g-1)));
            nr4(i,j)=pe_p04(i,j)*(g+1);
            dr4(i,j)=pi_p04(j)*mp4(i,j)^2*g*api_a4(i,j)+pi_p04(j)*...
                msi4(i,j)^2*g*asi_a4(i,j)+pi_p04(j)+Fbx/P0;
            ratio4(i,j)=nr4(i,j)/dr4(i,j);
            if (abs(ratio4(i,j))-1)<10e-5
                pratio4(i)=pi_p04(j);
                pte_p04(i)=0.9*pe_p04(i,j)*((g+1)/2)^(g/(g-1));
                k4(i)=alpha4(i,j);
                mp04(i)=sqrt((2/(g-1))*(pp_p04^((g-1)/g)-1));
                v0_vp04(i)=(M04/mp04(i))*(ts_t04/tp_t04(i)*...
                    ((1+((g-1)/2)*mp04(i)^2)/...
                    (1+((g-1)/2)*M04^2)))^(1/2);
                m104(i)=sqrt((2/(g-1))*(pte_p04(i)^((g-1)/g)-1));
                v10_vp04(i)=(m104(i)/
mp04(i))*(t10_tp4*((1+((g-1)/2)...
                    *mp04(i)^2)/(1+((g-1)/2)*m104(i)^2)))^(1/2);
                phi4(i)=(1+k4(i))*v10_vp04(i)-k4(i)*v0_vp04(i);
                break
            else
                pi_p04(j+1)=pi_p04(j)+dp4;
            end
    end
end
figure(4)
plot(tp_t04,phi4,'LineWidth',1.2)
title('Thrust augmentation ratio dependance on total temperature
 ratio')
xlabel('{T_t_p}/_{T_0}')
ylabel('\phi_p')
```



Final Plot

```
figure(5)
plot(tp_t04,phi4,'-r','LineWidth',1.2)
hold on
plot(a_ap3,phi3,'-b','LineWidth',1.2)
hold on
plot(t10_tp,phi,'-k','LineWidth',1.2)
hold on
plot(pp_p02,phi2,'-g','LineWidth',1.2)
hold on
legend('{T_t_p}/_{T_0}','{A}/_{A^*_p}','{T_t_1_0}/_{T_t_p}','{P_t_p}/
_{P_0}')
xlabel('Ratios')
ylabel('\phi_p')
title('Merged Plot')
```



Comments

```
%{
In order to take into account of the correction factors, the pressure
ratios were multiplied by a coefficient of 0.9 which resulted to be
 very
common in the literature as a pretty accurate factor. As a second loss
 the
wall friction of the constant area mixer was considered. For
 simplicity of
calculations I assumed a 1 meter diameter and 1 meter length duct.
 Given
the Mach conditions at that point (M=1) a coefficient of friction was
picked based on the graphs provided in the notes. Considering an
 altitude
of 20.000 meters as the operation altitude it was then possible to
calculate through trivial ideal gas laws steps the final F_bx momentum
contribution. Therefore that factor was added to the left-hand side of
 the
momentum equation derived for an ideal ejector ramjet. From then on I
proceeded with the same script of problem 8.9. It ended up having
 slightly
different curves which were still very similar to the ideal
non-loss-considering ones. It appears that only the pressure curve was
greatly influenced by the presence of losses. In the end we can assume
 that
depending on which kind of analysis needs to be conducted we can
 either
neglect losses in the calculations or not in order to still get
 reliable
numbers.

Basic script:
The fundamental script procedure follows the equations broken down in
 the
book for the ejector analysis. The value of Pi/P0 needs to be guessed
 and
verified through the momentum equation. A condition was given for
 which a
certain value depending on the heat air coefficient represents the
 minimum
value for that ratio. So starting from there and incrementing it of a
 small
quantity each loop, eventually it gets to a point where the momentum
 fluxes
are very close to be equal. It was found that they never reach the
 total
equality, probably due to approximations and so, but they can get very
 very
close to that, at a point which we can consider the result reliable
 enough.
%}
```

Published with MATLAB® R2019b