```matlab
% Thomas Satterly
% AAE 537
% HW 1, Problem 1

clear;
close all;

%import aeroBox.shockBox.*;

gamma = 1.4;
% Mach 2 conditions
rampAngle = 4; % degrees

% Part (i)
% Solve for the total turning angle at the end when M_0 = 2
[shockAngle, M1] = aeroBox.shockBox.calcObliqueShockAngle('rampAngle',
 rampAngle, ...
    'mach', 2, 'gamma', gamma, 'useDegrees', 1);
v1 = aeroBox.shockBox.prandtlMeyerFcn('mach', M1, 'gamma', 1.4);
v_end = aeroBox.shockBox.prandtlMeyerFcn('mach', 1, 'gamma', 1.4);
isoRampTurning = rad2deg(v1 - v_end);
totalTurning = isoRampTurning + rampAngle;
% End Part (i)


% Part (ii)
% Solve for conditions and geometry at mach 4

% Ramp section
[shockAngle, M1] = aeroBox.shockBox.calcObliqueShockAngle('rampAngle',
 rampAngle, ...
    'mach', 4, 'gamma', gamma, 'useDegrees', 1);
v1 = aeroBox.shockBox.prandtlMeyerFcn('mach', M1, 'gamma', 1.4);
x(1) = 0;
x_0 = 1 / tand(shockAngle);
h_0 = 1; % Total inlet height (to cowl lip)
% Solve for h_w
slope1 = tand(rampAngle + asind(1 / M1));
slope2 = tand(rampAngle);
yInt1 = h_0 - slope1 * x_0;
yInt2 = 0;
x_w = (yInt2 - yInt1) / (slope1 - slope2);

h_w = h_0 - slope2 * x_w ;

% Isentropic spike section
startMach = M1;
endMach = aeroBox.shockBox.calcPMExpansionFan('mach', M1, 'gamma',
 1.4, 'useDegrees', 1, 'turningAngle', -isoRampTurning);
numSteps = 20;
machs = linspace(startMach, endMach, numSteps);
```

```
slope2 = tand(rampAngle);
yInt2 = 0;
lastMach = M1;
isoAngles = rampAngle;
for i = 1:numel(machs);
    thisMach = machs(i);
    mu = asind(1 / thisMach);
    slope1 = tand(mu + isoAngles(i));
    yInt1 = h_0 - x_0 * slope1;
    isoSlope(i) = slope2;
    isoYInt(i) = yInt2;
    isoX(i) = (yInt2 - yInt1) / (slope1 - slope2);
    isoH(i) = slope1 * isoX(i) + yInt1;
    if (i ~= numel(machs))
        dAngle = rad2deg(aeroBox.shockBox.prandtlMeyerFcn('mach',
 machs(i), 'gamma', 1.4) - ...
            aeroBox.shockBox.prandtlMeyerFcn('mach', machs(i +
 1), 'gamma', 1.4));
        slope2 = tand(isoAngles(i));
        yInt2 = isoH(i) - isoX(i) * slope2;
        isoAngles(i + 1) = isoAngles(i) + dAngle;
    end
    %     v =  aeroBox.shockBox.prandtlMeyerFcn('mach', thisMach,
 'gamma', 1.4);
    %     isoH(i) = h_w * (M1 / thisMach) * ((2 + (gamma - 1) *
 M1^2) / (2 + (gamma - 1) * thisMach^2))^-((gamma + 1) / (2 * (gamma -
 1)));
    %     isoX(i) = x_0 - (isoH(i) / (tan(mu + deg2rad(rampAngle) + v1
 - v)));
end
isoSlope(end + 1) = tand(totalTurning);
% Inlet so far
x = [0, x_w, isoX];
y = [h_0 - [h_0, h_w], isoH];

% End of part (ii)



% Part (iii)
% Determine throat height & make throat with an entry


% Find the ratio of throat height to length
LtOverHt = 10 * ((endMach - 1) / 2.2)^0.5;

% Find the throat height, which is the distance from the end of the
 iso
% ramp perpendicular to the
y1 = isoH(end - 1);
y2 = isoH(end);
x1 = isoX(end - 1);
x2 = isoX(end);
```

```
ht = abs((y2 - y1) * x_0 - (x2 - x1) * h_0 + x2 * y1 - y2 * x1) /
 sqrt((y2 - y1)^2 + (x2 - x1)^2);
rad = 4 * ht; % If this is to be beleived

% Extend the iso ramp to the perpendicular point
y(end + 1) = h_0 - cosd(totalTurning) * ht;
x(end + 1) = x_0 + sind(totalTurning) * ht;

% Arc parameters
yc = y(end) - cosd(totalTurning) * rad;
xc = x(end) + sind(totalTurning) * rad;

% Draw the arc
numPoints = 200;
startAngle = 360 - totalTurning;
endAngle = 360;
angleStep = (endAngle - startAngle) / numPoints;
for i = 1:numPoints
    xr(i) = xc + sind(startAngle + angleStep * i) * rad;
    yr(i) = yc + cosd(startAngle + angleStep * i) * rad;
end

% Add the arc and throat
x = [x xr (xr(end) + ht * LtOverHt)];
y = [y yr yr(end)];

% Diffuser

% Area function, conical
areaFunc = @(x, angle, r) (pi * (r^2 - (r - tand(angle) * x)^2)) / (pi
 * r^2);

throatEndHeight = y(end);
diffAngle = 3;
dMin = 0;
dMax = throatEndHeight / tand(diffAngle);
numSteps = 200;
dStep = (dMax - dMin) / numSteps;
for i = 1:numSteps
    areaRat = areaFunc(i * dStep, diffAngle, throatEndHeight);
    xd(i) = x(end) + dStep * i;
    yd(i) = throatEndHeight - areaRat * throatEndHeight;
end

% Append to overall geometry
x = [x xd];
y = [y yd];

% Now let's make the cowl
xCowl(1) = x_0;
yCowl(1) = h_0;

% Translate the arc to the cowl
startAngle = 360 - atand((xc - xCowl) / (yCowl - yc));
```

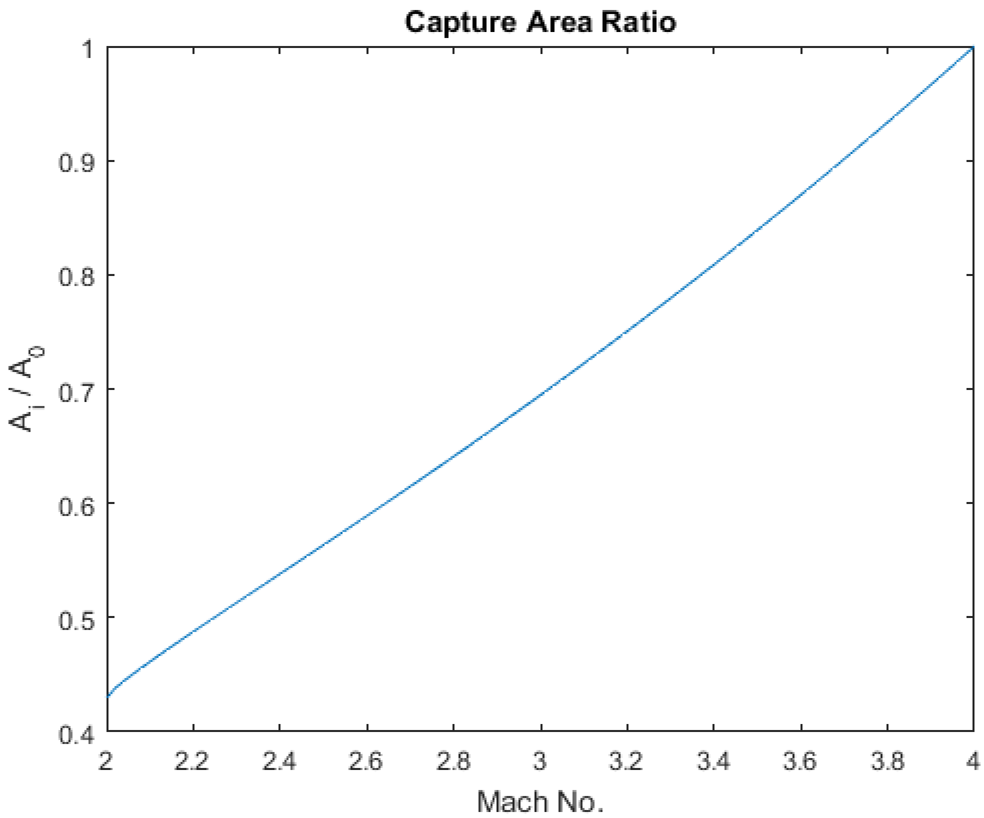
```
endAngle = 360;
numPoints = 200;
angleStep = (endAngle - startAngle) / numPoints;
rad = sqrt((xCowl - xc)^2 + (yCowl - yc)^2);
for i = 1:numPoints
    xCowl(i) = xc + sind(startAngle + angleStep * i) * rad;
    yCowl(i) = yc + cosd(startAngle + angleStep * i) * rad;
end

% Extend cowl to the end of the diffuser
xCowl(end + 1) = x(end);
yCowl(end + 1) = yCowl(end);


% Know the goemetry now, let's calculate the inlet properties at any
 given mach
machs = linspace(2, 4, 100);
doPlot = 0; % Flag for plotting inlet geometry with the shock and mach
 waves
for j = 1:numel(machs)
    thisMach = machs(j);

    if (doPlot)
        % Start the figure
        figure;
        hold on;
    end

    % Set up inital properties
    q = 1500;
    rho_1 = 1;

    T_t_0 = 273;
    T_0 = aeroBox.isoBox.calcStaticTemp('mach', thisMach, 'gamma',
 1.4, 'Tt', T_t_0);

    P_0 = (2 * q) / (1.4 * thisMach^2);
    P_t_0 = aeroBox.isoBox.calcStagPressure('mach', thisMach, 'gamma',
 1.4, 'P', P_0);

    % Pressure after shock
    P_1 = aeroBox.shockBox.calcPressureAfterOblique('gamma',
 1.4, 'mach', thisMach, 'rampAngle', rampAngle, 'useDegrees', 1, 'Ps',
 P_0);


    % Leading shock
    [shockAngle, startMach] =
 aeroBox.shockBox.calcObliqueShockAngle('rampAngle', rampAngle, ...
        'mach', thisMach, 'gamma', gamma, 'useDegrees', 1);

    % Total properties after the shock
    T_1 = aeroBox.isoBox.calcStaticTemp('mach', startMach, 'gamma',
 1.4, 'Tt', T_t_0);
```

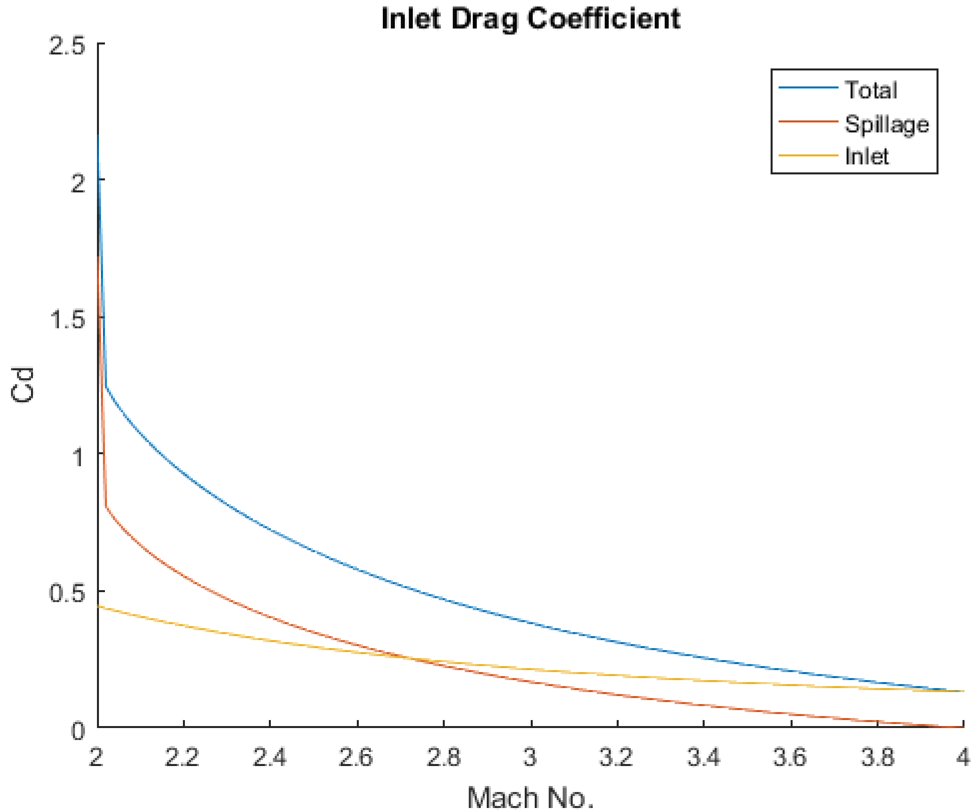
```
    T_t_1 = T_t_0;
    rho_t_1 = rho_1 * (1 + ((1.4 - 1) / 2) * startMach^2)^(1 / (1.4 -
1));
    Pt_1 = aeroBox.isoBox.calcStagPressure('mach', startMach, 'gamma',
1.4, 'Ps', P_1);
    v_1 = sqrt(1.4 * 286.9 * T_1) * startMach;

    if doPlot
        % Plot the shock
        dist = 5;
        plot([x(1), x(1) + cosd(shockAngle) * dist], ...
            [y(1), y(1) + sind(shockAngle) * dist], 'r');
    end


    % Calculate machs along the iso ramp
    for i = 1:numel(isoAngles)
        % Mach at point
        isoMachs(i) = aeroBox.shockBox.calcPMExpansionFan('mach',
startMach, ...
            'gamma', 1.4, 'useDegrees', 1, 'turningAngle', -
(isoAngles(i) - rampAngle));
        isoPressure(i) = aeroBox.isoBox.calcStaticPressure('mach',
isoMachs(i), 'gamma', 1.4, 'Pt', Pt_1);
        if (isoMachs(i) == 0)
            isoMachs(i) = 1;
            machAngle = 90;
        else
            machAngle = asind(1 / isoMachs(i));
        end
        % Calculate the line function
        isoMachSlope(i) = tand(machAngle + isoAngles(i));
        isoMachYInt(i) = isoH(i) - isoMachSlope(i) * isoX(i);
        if doPlot
            % Plot mach waves
            plot([isoX(i), isoX(i) + cosd(machAngle + isoAngles(i)) *
dist], ...
                [isoH(i), isoH(i) + sind(machAngle + isoAngles(i)) *
dist], 'g');
        end
    end

    % Calculate conditions after the terminal shock
    M3 = aeroBox.shockBox.calcMachAfterNormal('mach',
isoMachs(end), 'gamma', 1.4);
    P_3 = aeroBox.shockBox.calcPressureAfterNormal('mach',
isoMachs(end), 'gamma', 1.4, ...
        'Ps', aeroBox.isoBox.calcStaticPressure('mach',
isoMachs(end), 'gamma', 1.4, 'Pt', Pt_1));
    P_t_3 = aeroBox.isoBox.calcStagPressure('mach', M3, 'gamma',
1.4, 'Ps', P_3);
    rho_3 = (((1.4 + 1) * M3^2) / ((1.4 - 1) * M3^2 + 2))^-1 * ...
        aeroBox.isoBox.calcStaticDensity('mach', M3, 'gamma',
1.4 ,'rho_t', rho_t_1);
```

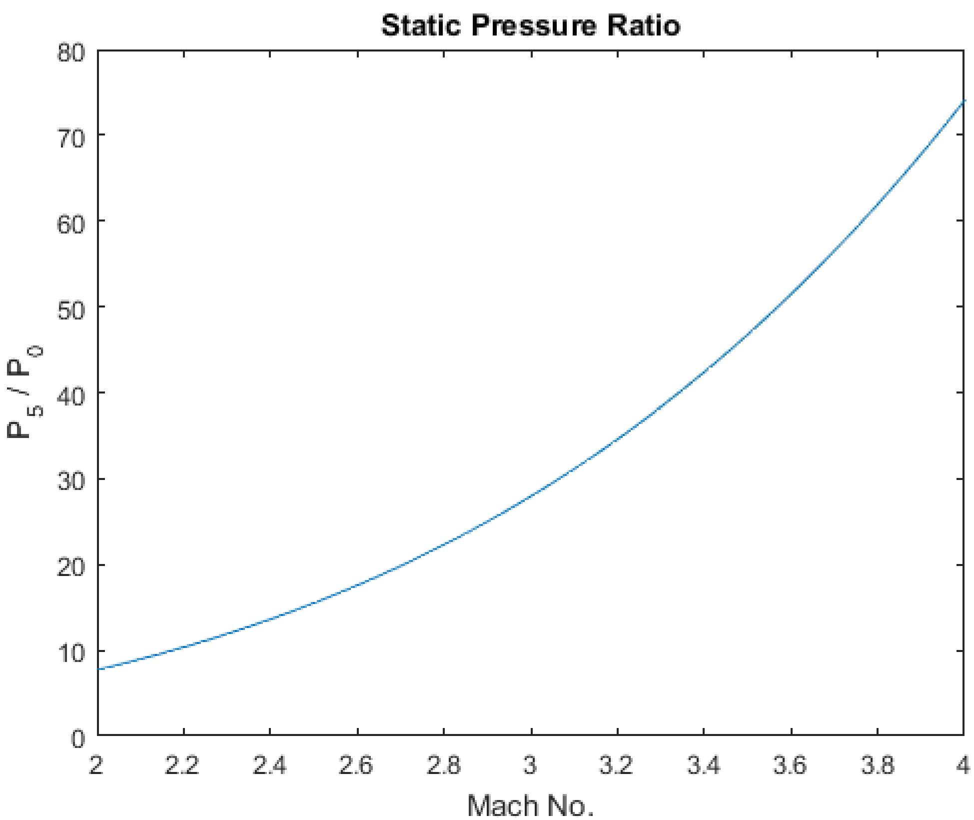
```
    T_t_3 = T_t_1;
    rho_t_3 = rho_3 * (1 + ((1.4 - 1) / 2) * M3^2)^(1 / (1.4 - 1));

    % Calculate exit properties
    A_e = yCowl(end);
    M5 = aeroBox.isoBox.machFromAreaRatio((A_e / ht) * ...
        aeroBox.isoBox.calcARatio(M3, 1.4), 1.4, 0);
    T_5 = aeroBox.isoBox.calcStaticTemp('mach', M5, 'gamma',
1.4, 'Tt', T_t_3);
    v_5 = sqrt(1.4 * 286.9 * T_5) * M5;
    P_5 = aeroBox.isoBox.calcStaticPressure('mach', M5, 'gamma',
1.4, 'Pt', P_t_3);
    rho_5 = aeroBox.isoBox.calcStaticDensity('mach', M5, 'gamma',
1.4, 'rho_t', rho_t_3);

    % Now go backwards and calculate the capture area profile
    capX(numel(isoAngles) + 2) = x_0;
    capY(numel(isoAngles) + 2) = h_0;
    for i = 1:numel(isoAngles)
        index = numel(isoAngles) - i + 1;
        % Bounding line
        capSlope(index) = isoSlope(index + 1);
        capYInt(index) = capY(index + 2) - capX(index + 2) *
capSlope(index);

        % Mach wave to intersect
        capX(index + 1) = (capYInt(index) - isoMachYInt(index)) / ...
            (isoMachSlope(index) - capSlope(index));
        capY(index + 1) = capYInt(index) + capSlope(index) *
capX(index + 1);
    end

    % Final intersect to shock
    slope = tand(rampAngle);
    yInt = capY(2) - capX(2) * slope;
    capX(1) = (-yInt) / (slope - tand(shockAngle));
    capY(1) = yInt + slope * capX(1);
    if doPlot
        % Plot the capture area contour
        plot(capX, capY, 'b');
    end
    % Calculate pressure force along the segment
    A_i = capY(1);
    pressures = [P_1, isoPressure] - P_0;
    % Calculate spillate Cd
    Cd_add(j) = sum(pressures .* diff(capY)) / (q * A_i);

    % Calculate inlet Cd
    Cd_inlet(j) = (A_i * P_1) / (q * A_i);

    % Total Cd
    Cd_tot(j) = Cd_inlet(j) + Cd_add(j);

    % Record properties
```

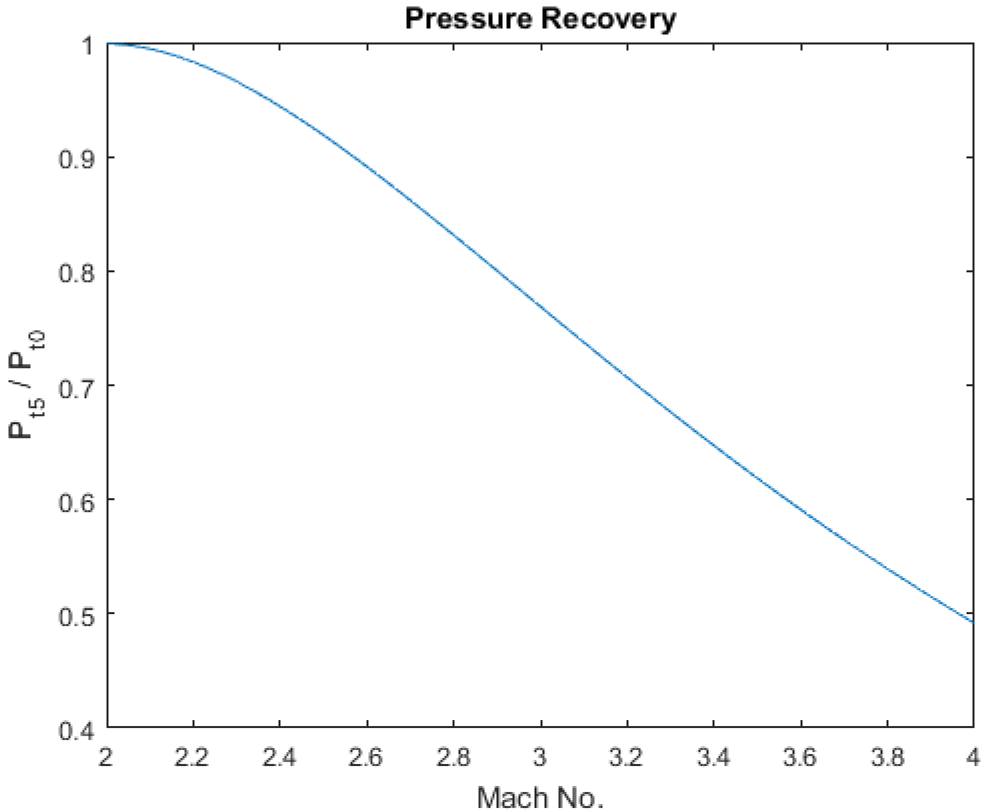
```
        aRat(j) = A_i;
        ptRat(j) = P_t_3 / P_t_0;
        pRat(j) = P_5 / P_0;
        tRat(j) = T_5 / T_0;
        throatM(j) = M3;
        exitMach(j) = M5;

        if doPlot
            % Plot inlet geometry over everything else
            title(sprintf('Inlet Geometry for M = %0.2f', thisMach));
            plot(x, y, 'k', 'LineWidth', 2);
            axis equal;
            plot(xCowl, yCowl, 'k', 'LineWidth', 2);
        end
    end

    figure;
    plot(machs, aRat);
    title('Capture Area Ratio');
    xlabel('Mach No.');
    ylabel('A_i / A_0');


    figure;
    hold on;
    plot(machs, Cd_tot);
    plot(machs, Cd_add);
    plot(machs, Cd_inlet);
    legend('Total', 'Spillage', 'Inlet');
    title('Inlet Drag Coefficient');
    xlabel('Mach No.');
    ylabel('Cd');

    figure;
    plot(machs, pRat);
    title('Static Pressure Ratio');
    xlabel('Mach No.');
    ylabel('P_5 / P_0');

    figure;
    plot(machs, ptRat);
    title('Pressure Recovery');
    xlabel('Mach No.');
    ylabel('P_t_5 / P_t_0');

    figure;
    plot(machs, tRat);
    title('Static Temperature Ratio');
    xlabel('Mach No.');
    ylabel('T_5 / T_0');

    figure;
    plot(machs, throatM);
    title('Throat Mach Number');
```

```
xlabel('Mach No.');
ylabel('M_5');

figure;
plot(machs, exitMach);
title('Exit Mach Number');
xlabel('Mach No.');
ylabel('M_5');
```

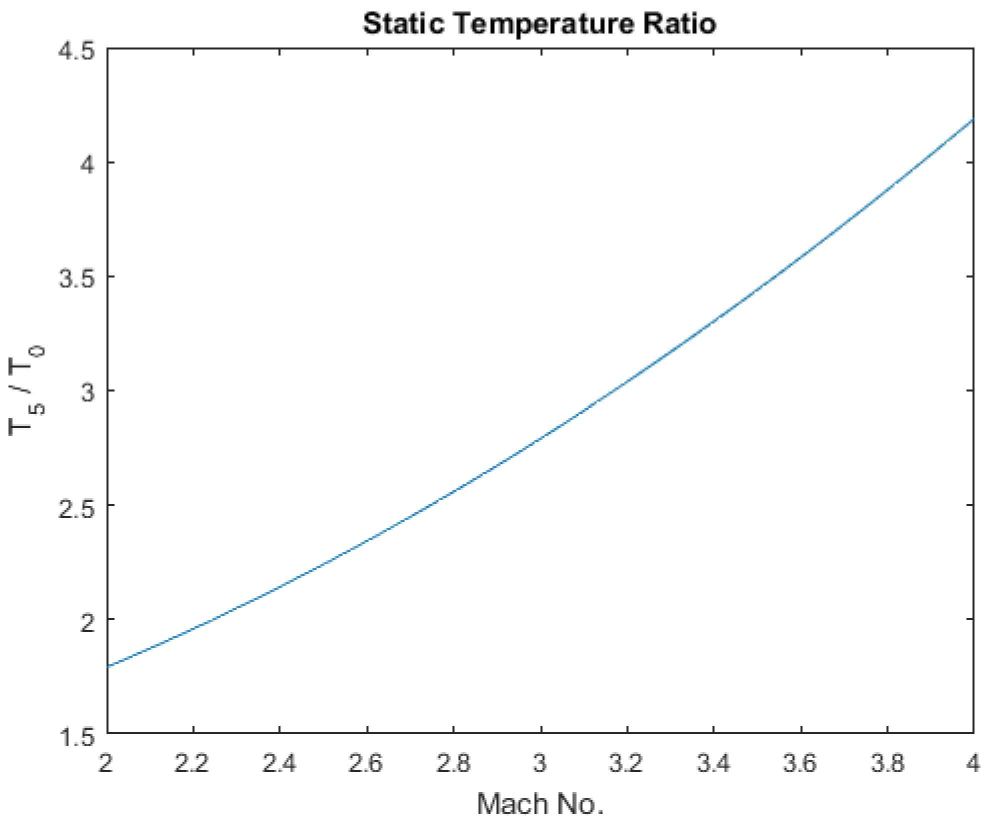



Inlet Drag Coefficient
Total
Spillage
Inlet
Cd
2.5
2
1.5
1
0.5
0
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.

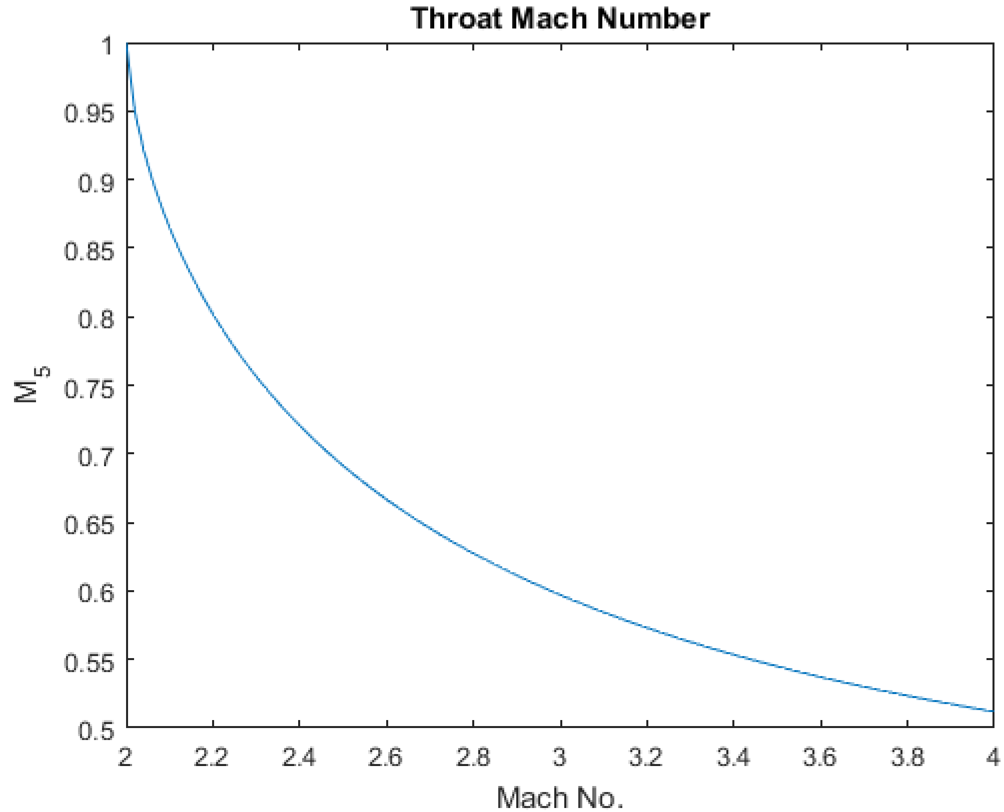

Static Pressure Ratio
P_5 / P_0
80
70
60
50
40
30
20
10
0
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.



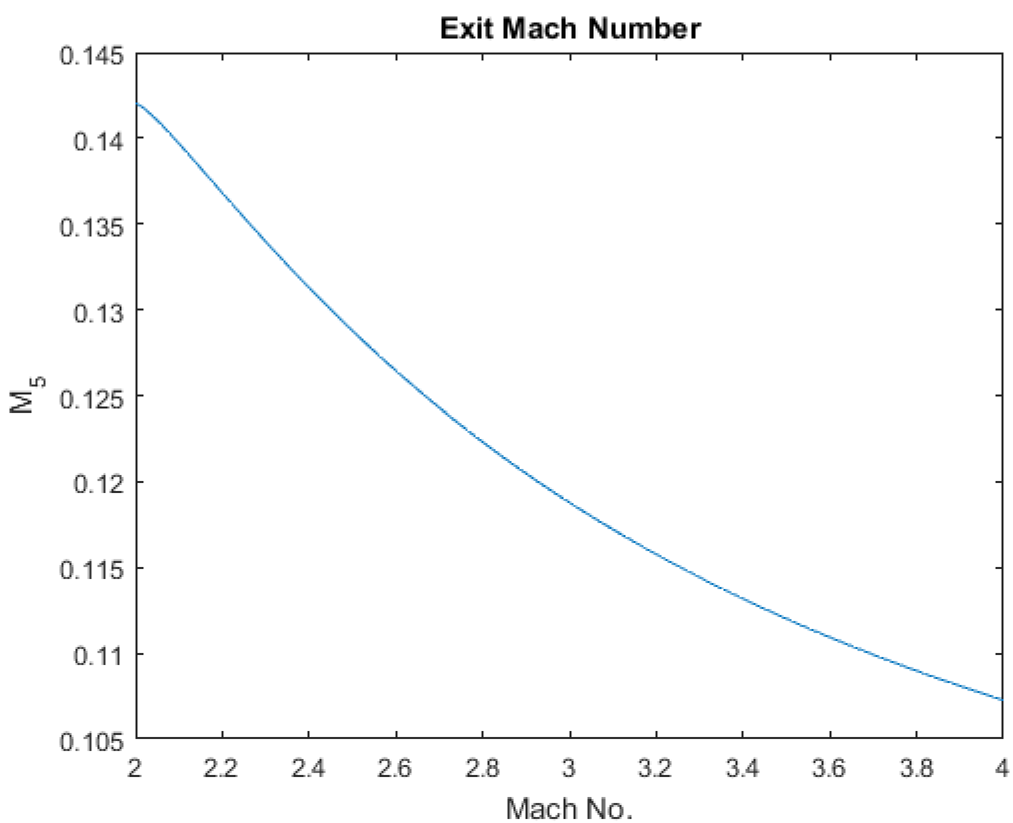
Pressure Recovery
P_{t5} / P_{t0}
1
0.9
0.8
0.7
0.6
0.5
0.4
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.



Static Temperature Ratio
T_5 / T_0
4.5
4
3.5
3
2.5
2
1.5
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.



Throat Mach Number
M_5
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.



Exit Mach Number
M_5
0.145
0.14
0.135
0.13
0.125
0.12
0.115
0.11
0.105
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
Mach No.

Published with MATLAB® R2016a