Trabajo de final de doble Máster

Master Universitario en Ingeniería Industrial

Master Universitario en Ingeniería de la Automoción

INSTALACIÓN DE UN MOTOR ROTAX EN UN VEHÍCULO TIPO FÓRMULA: ADAPTACIÓN DE LA ADMISIÓN Y EL ESCAPE

ANEXO

Autor: Cristian Díez Quílez

Director: Emilio Hernández

Convocatoria: Septiembre 2017

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resumen

El presente proyecto detalla el rediseño e instalación de un motor Rotax 125 MAX de dos tiempos a un vehículo tipo fórmula del fabricante Ashenkoff, concretamente el modelo K100.

Este vehículo está orientado a niños de hasta 10 años que participarían en una competición organizada por la misma firma con el objetivo de introducir a los niños que actualmente compiten en karts en el mundo de los monoplazas tipo fórmula.

El proyecto tiene como objetivo adaptar dicho motor de Rotax al vehículo de Ashenkoff salvando todas las diferencias que hay entre un kart y el modelo K100. En el proyecto se detalla el funcionamiento de estos motores, las exigencias que éste impone sobre el diseño y las limitaciones que implica un vehículo como el K100. Durante el desarrollo del proyecto se clarificó la necesidad de adaptar ciertos componentes que venían de serie con el motor debido al espacio y el formato del mismo disponible en el vehículo. A tal fin se han rediseñado algunos componentes para garantizar que el motor cumple sus funciones con las máximas prestaciones. La validación de dichos diseños se ha realizado mediante simulaciones con el programa ANSYS Fluent así como simulaciones con un código de simulación de motores de dos tiempos implementado en Matlab.

En el presente anexo se muestran varias figuras complementarias de las simulaciones realizadas y el código usado para validar el diseño del tubo de escape.

Índice de figuras del anexo

Sumario

1. Simulaciones de la admisión

1.1. Simulaciones de la 'Aerodinámica 1'

Contorno de presión estática, la mayor presión se presenta en los puntos de estancamiento.

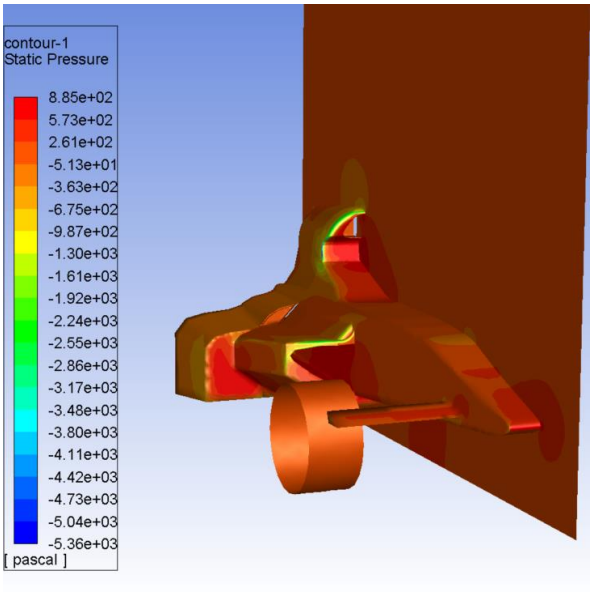

Figura 1: Contorno de presión estática

Líneas de corriente del modelo, se puede observar como atraviesan las secciones de entrada.

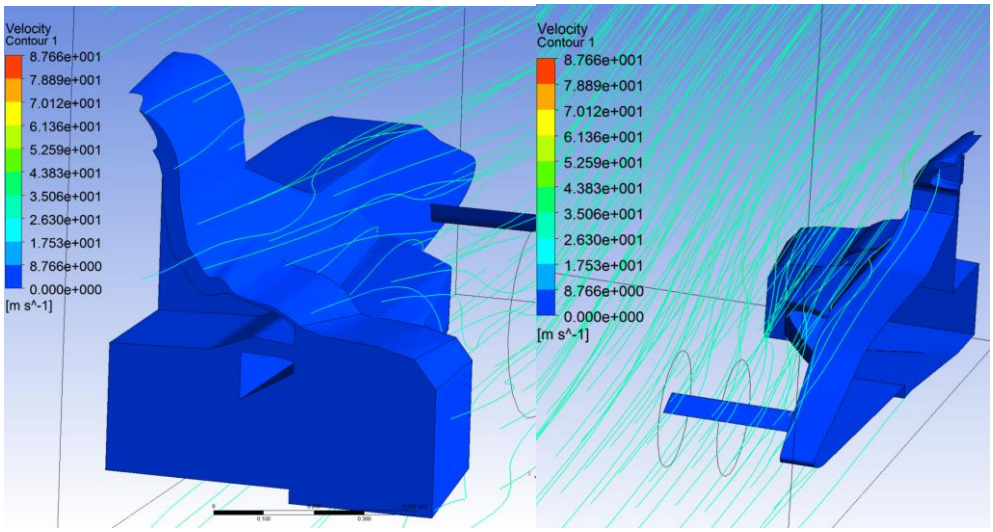

Figura 2: Líneas de corriente

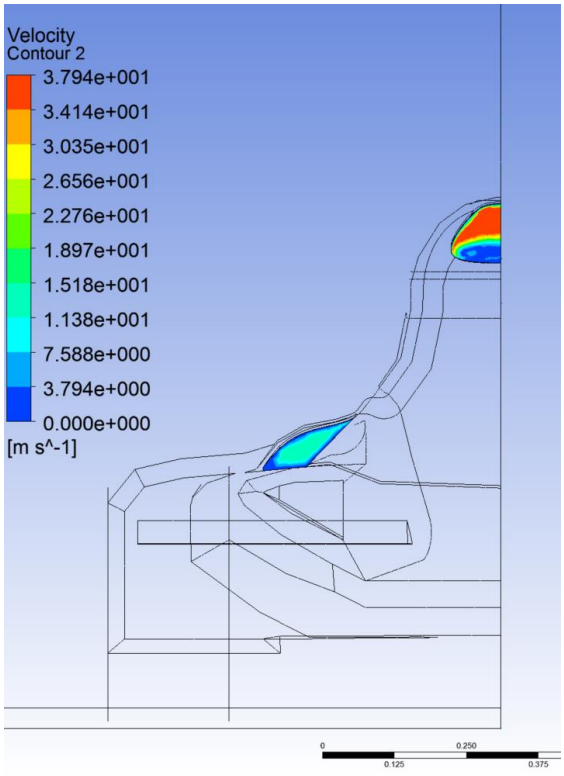

Figura 3: Contornos de velocidad en las tomas de aire

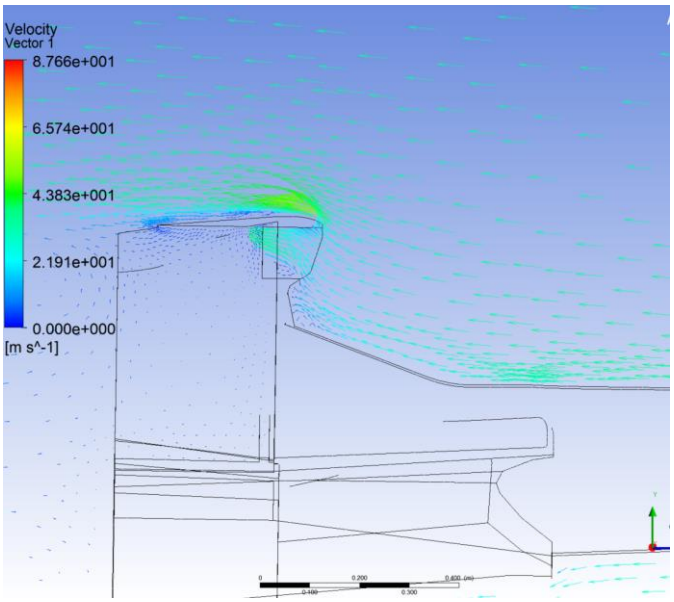

Figura 4: Vectores velocidad en el plano de simetría

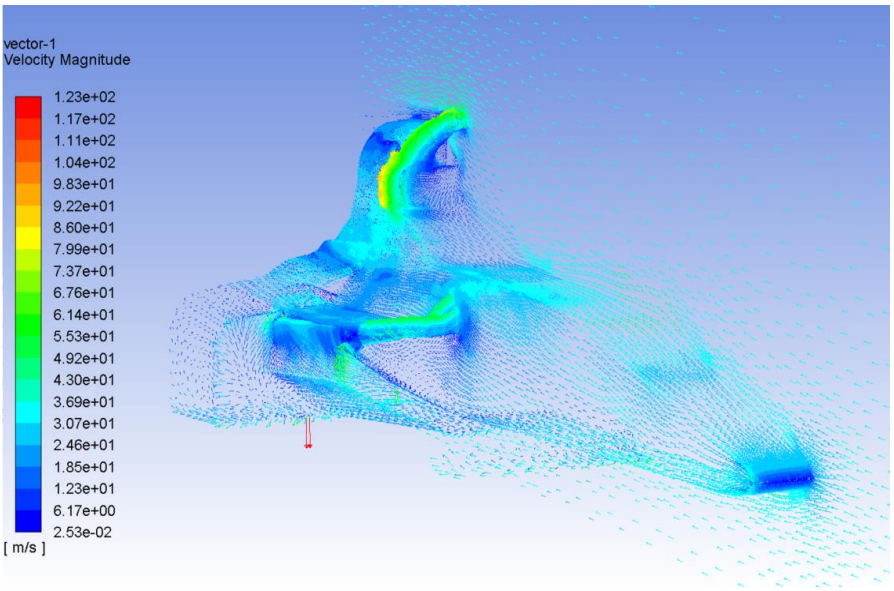

Figura 5: Vectores velocidad alrededor del vehículo

En los vectores velocidad se aprecia el comportamiento inverso al contorno de presión salvo en las zonas donde hay desprendimiento.

1.2. Simulaciones de la ‘Aerodinámica 2’

Contornos de presión estática en el vehículo. De nuevo correspondientes a los puntos de mayor estancamiento.

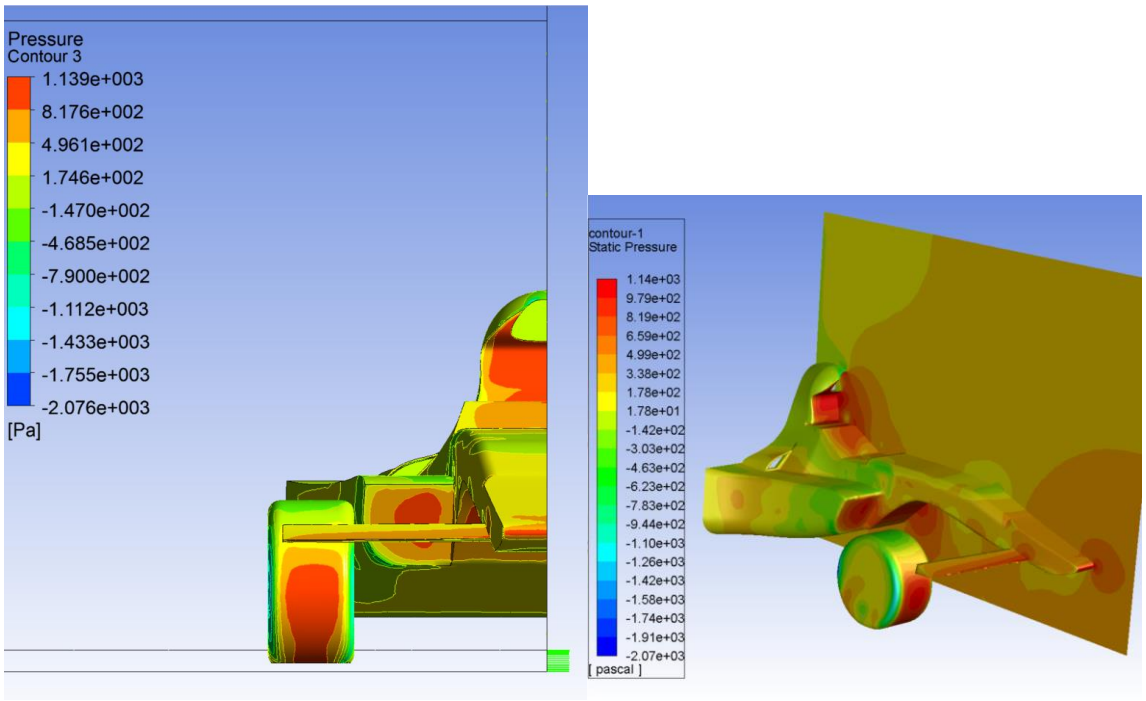

Figura 6: Presión estática en el vehículo

Líneas de corriente que atraviesan las tomas de aire. Se crean turbulencias importantes antes de éstas.

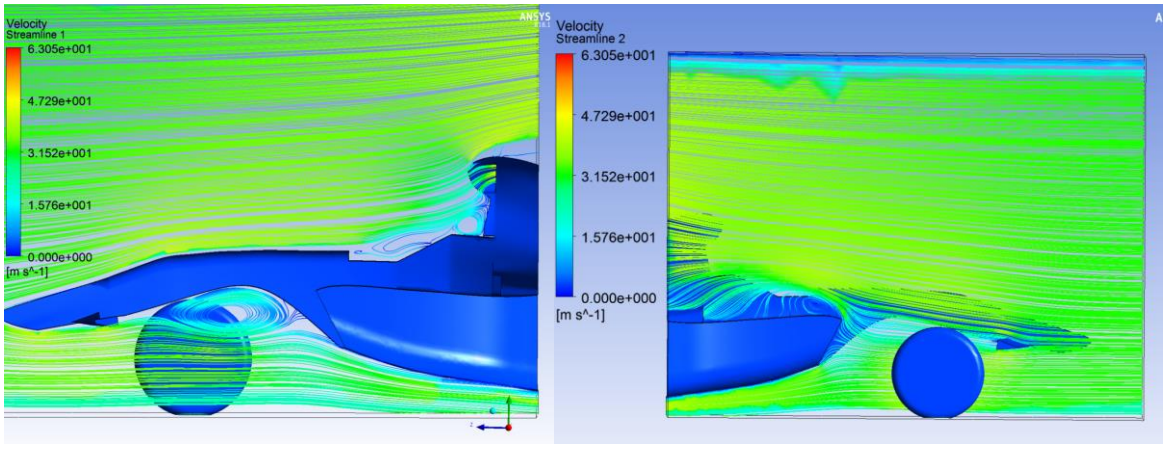

Figura 7: Líneas de corriente a través de las tomas

Presión total sobre el vehículo. Se diferencian las zonas de presión baja por desprendimiento de capa.

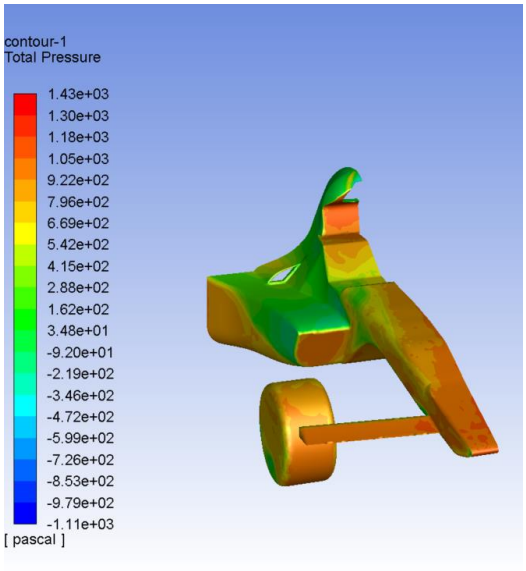

Figura 8: Presión total del fluido en el vehículo

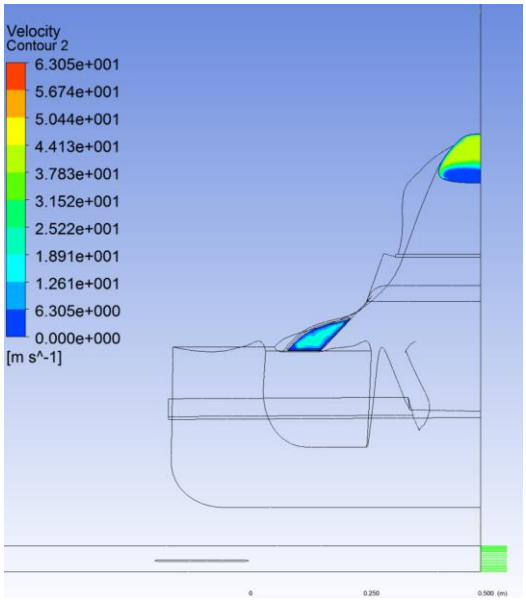

Figura 9: Contornos de velocidad en las tomas de aire

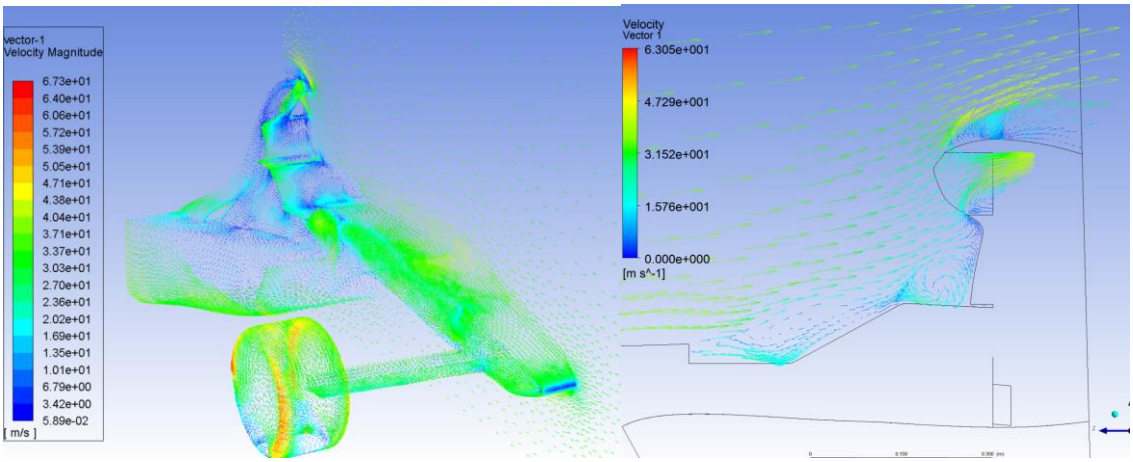

Figura 10: Vectores velocidad en el vehículo y en el plano de simetría

1.3. Simulaciones de la caja de admisión 1

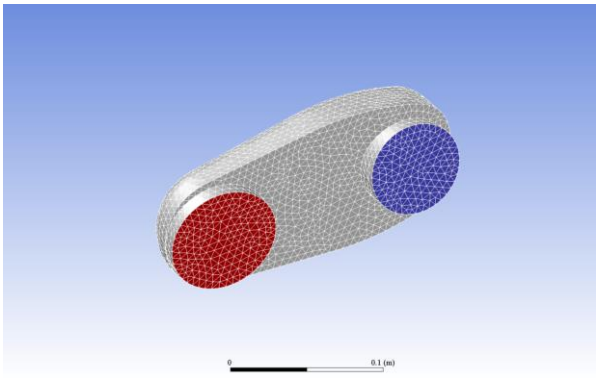

Figura 11: Malla del componente

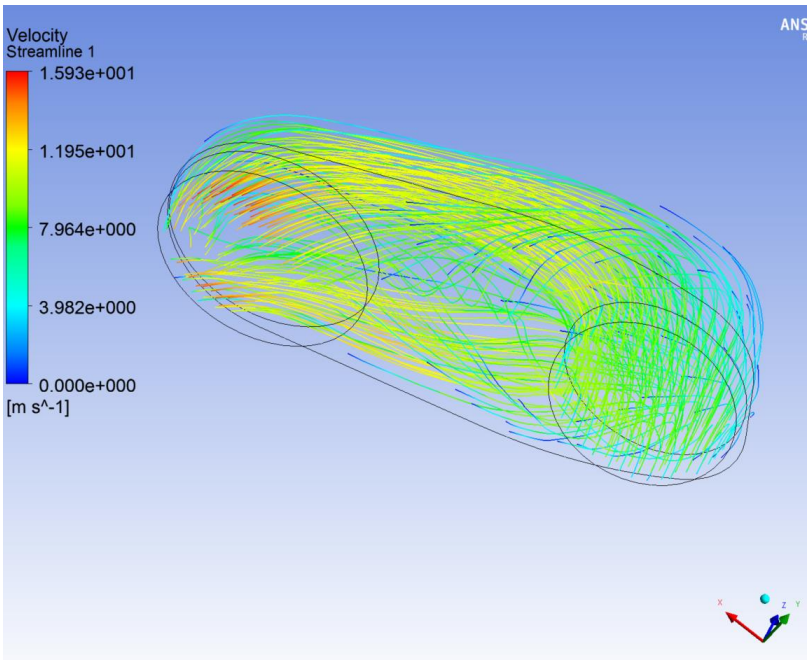

Figura 12: Líneas de corriente a través de la caja

En el contorno de presión total se observa claramente una zona de presión muy baja debido a un gran desprendimiento del fluido. En esa zona se generan turbulencias y reflujo.

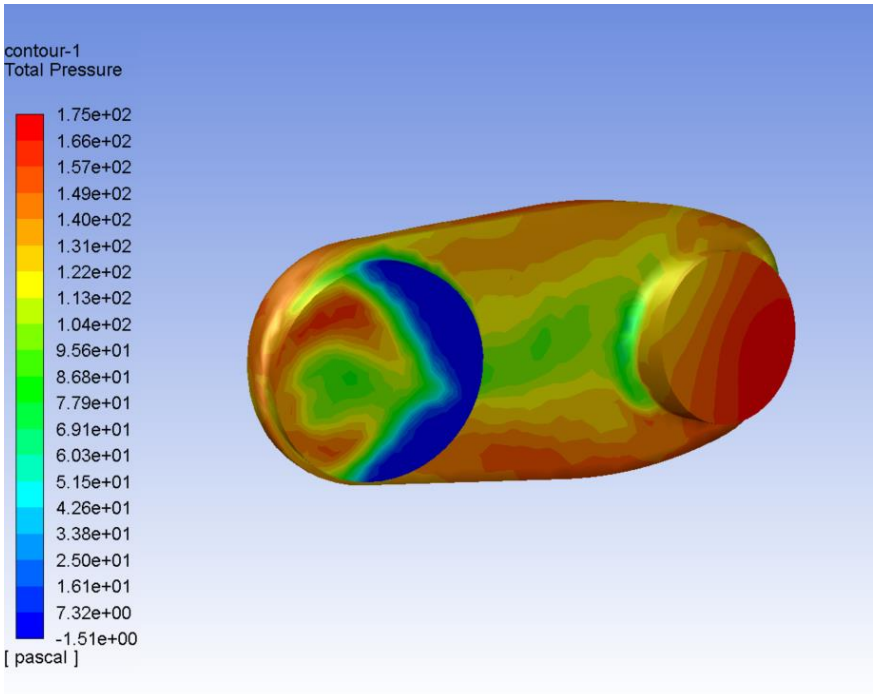

Figura 13: Contorno de presión total

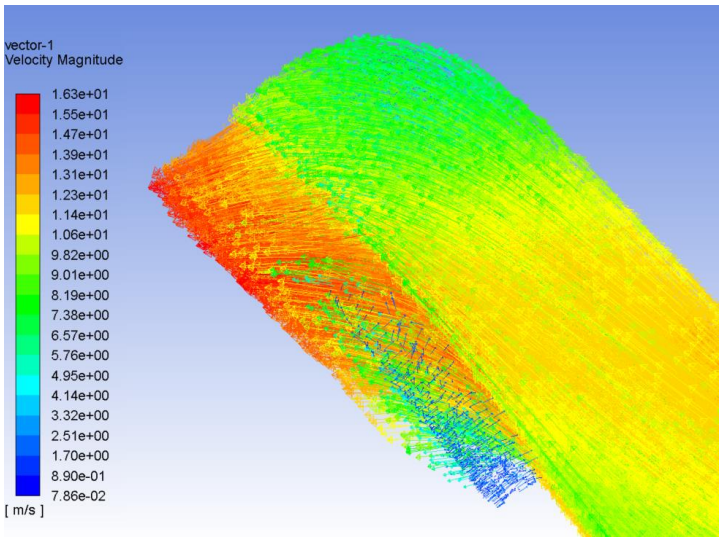

Figura 14: Vectores velocidad a la salida del componente

En ambos vectores velocidad se aprecia el efecto del reflujo y la aceleración del fluido por el estrechamiento de la zona de paso de las líneas de corriente.

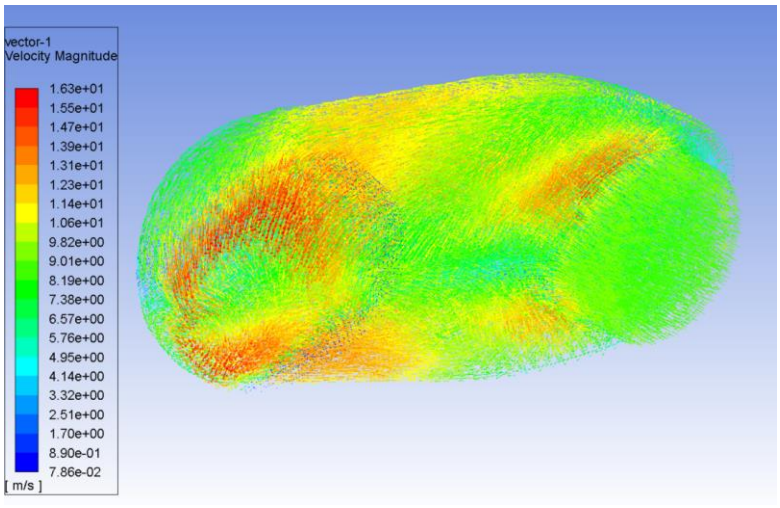

Figura 15: Vectores velocidad en el componente

1.4. Simulaciones de la caja de admisión 2

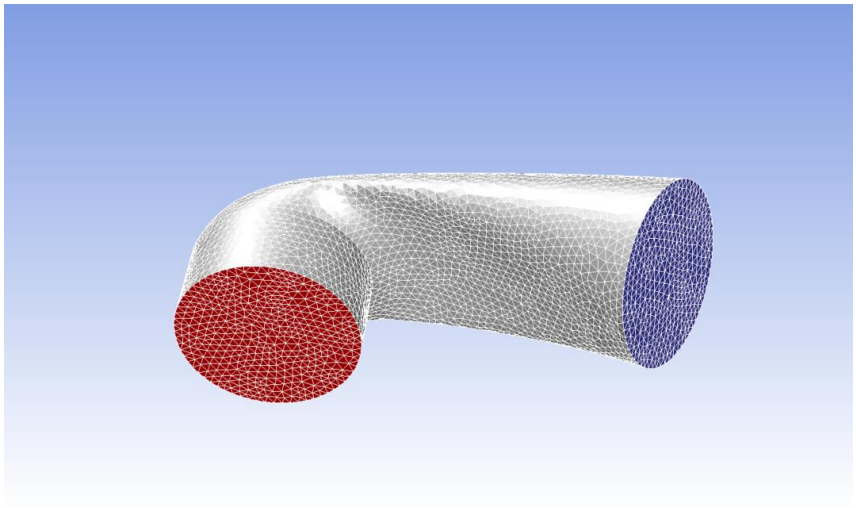

Figura 16: Malla del componente

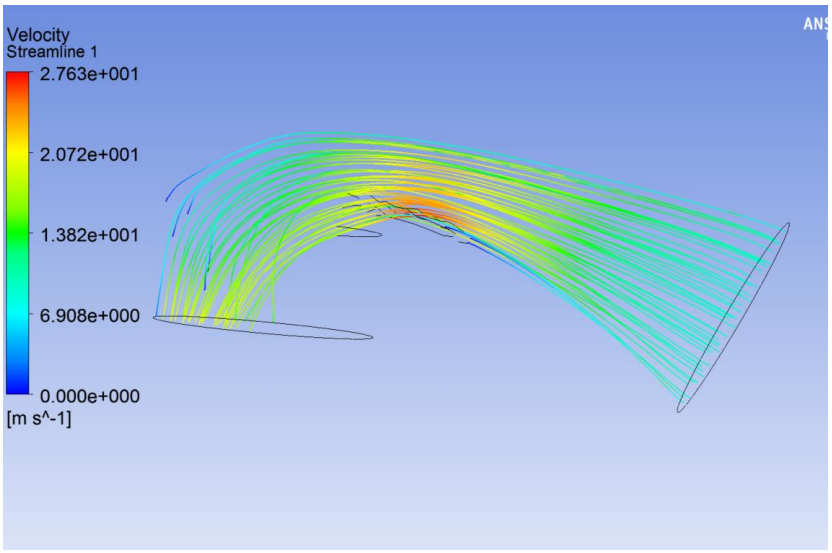

Figura 17: Líneas de corriente por el componente

En este diseño se puede apreciar claramente como en el contorno de presión total hay una enorme depresión en la sección de salida debido a la separación del flujo. Este efecto se aprecia claramente en las líneas de corriente donde éstas no penetran en esa zona debido a que hay reflujo y turbulencias por desprendimiento.

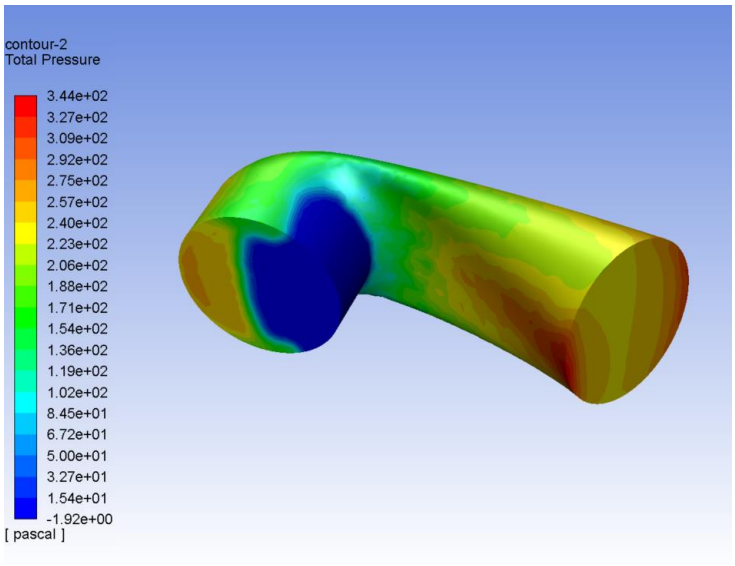

Figura 18: Contorno de presión total

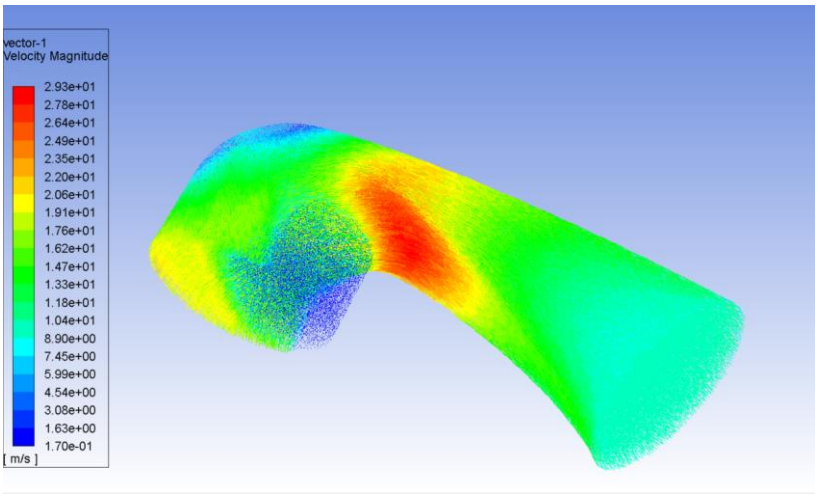

Figura 19: Vectores velocidad en el codo

1.5. Colector lateral

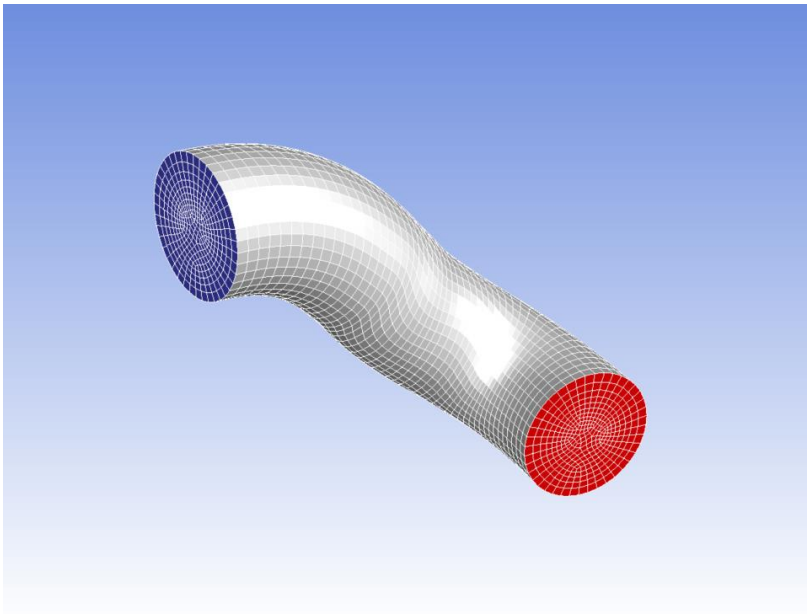

Figura 20: Malla del colector

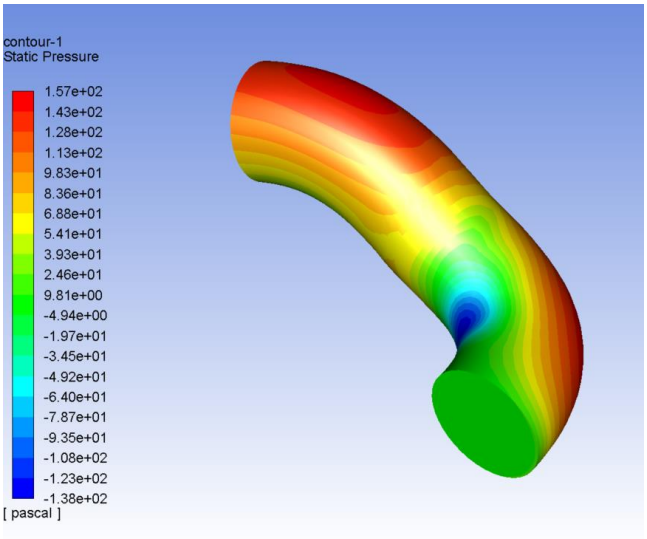

Figura 21: Contorno de presión estática

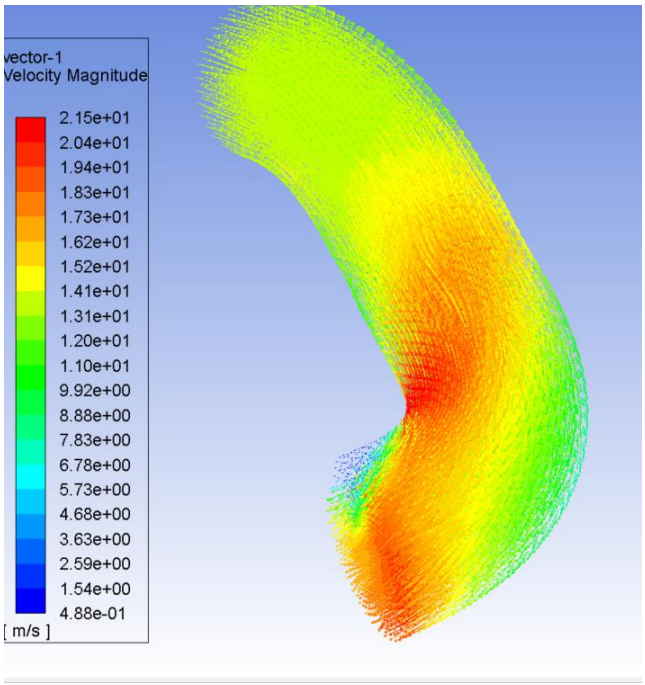

Figura 22: Vectores velocidad en el colector

1.6. Colector superior

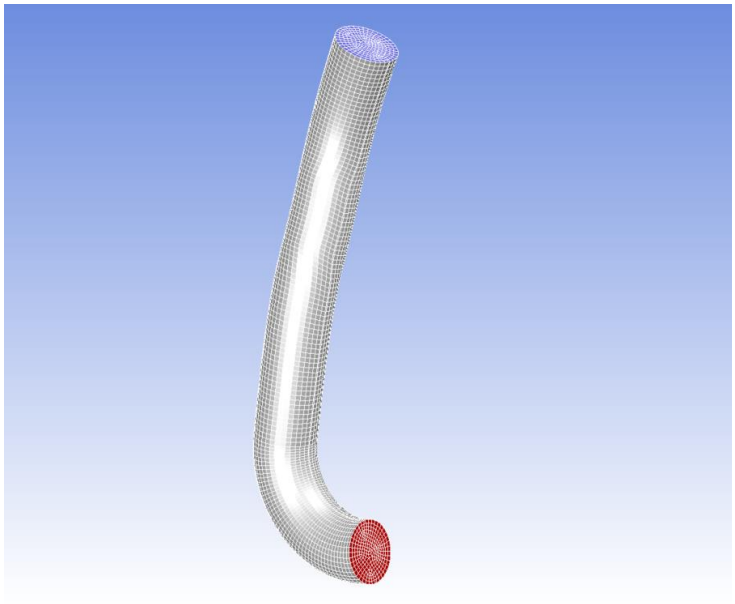

Figura 23: Malla refinada del colector

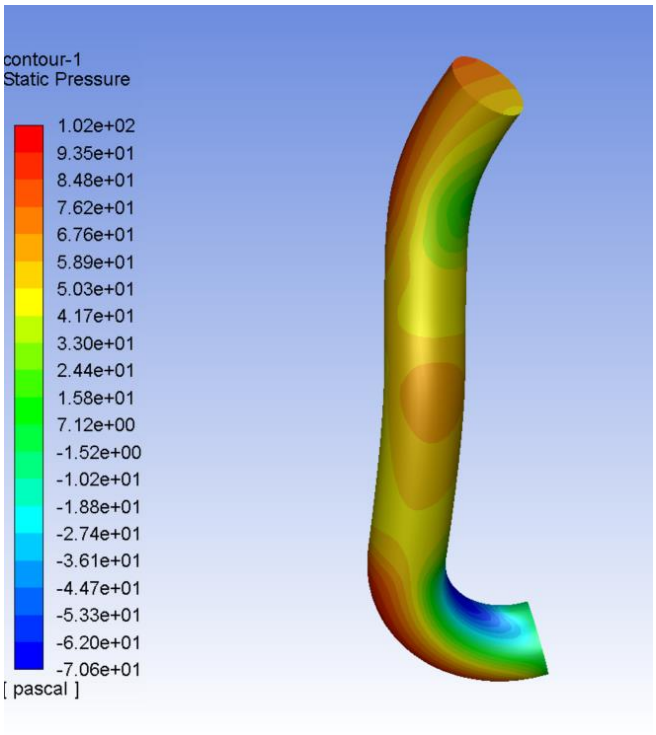

Figura 24: Contorno de presión estática en el colector

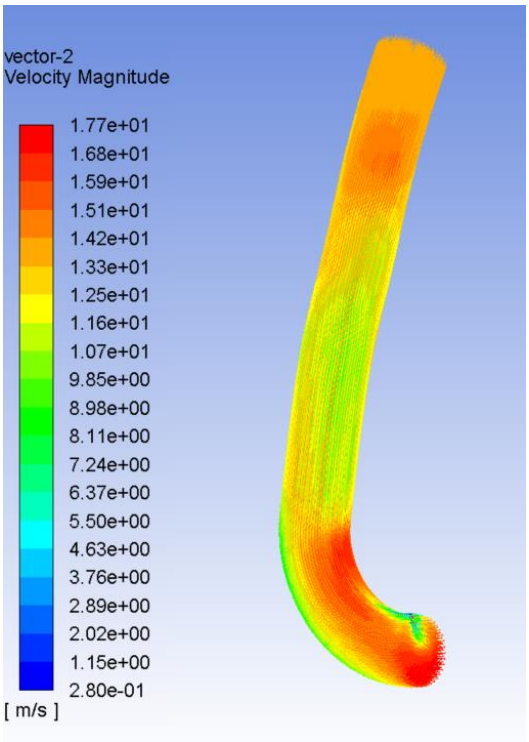

Figura 25: Vectores velocidad en el colector

2. Teoría tubo de escape

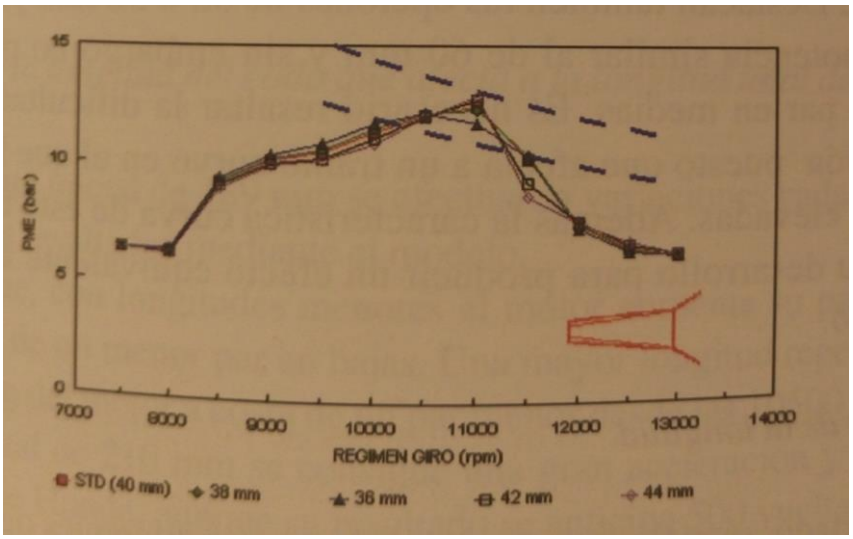

Figura 26: Variación del diámetro inicial

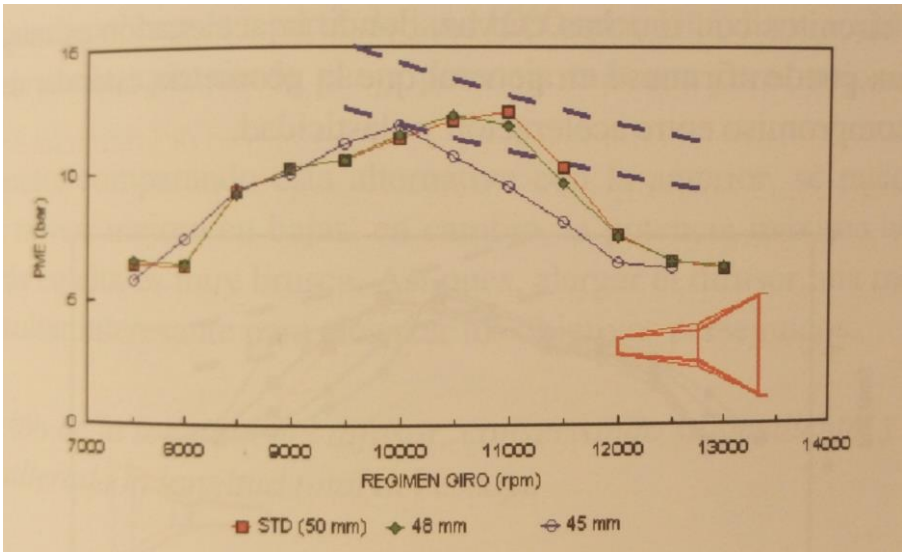

Figura 27: Diámetros menores

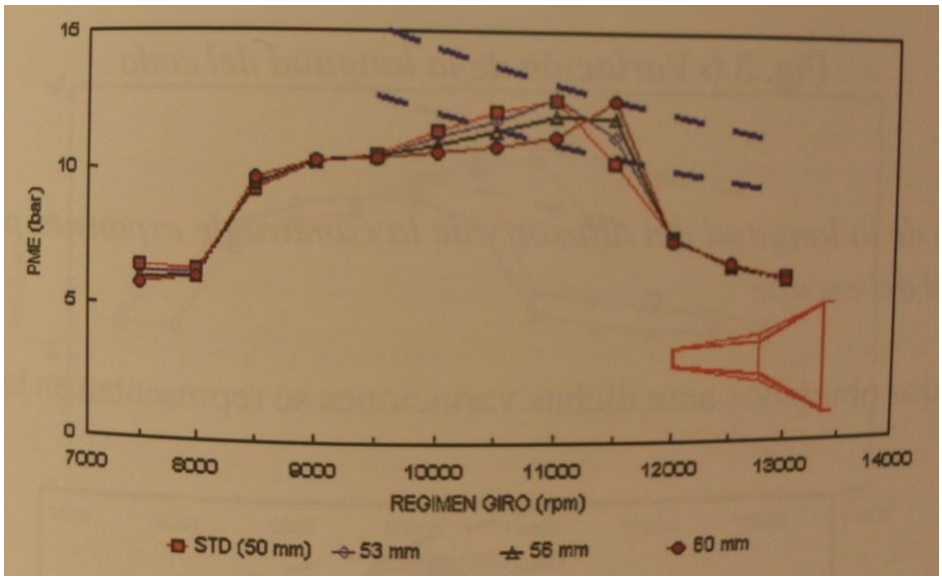

Figura 28: Diámetros mayores

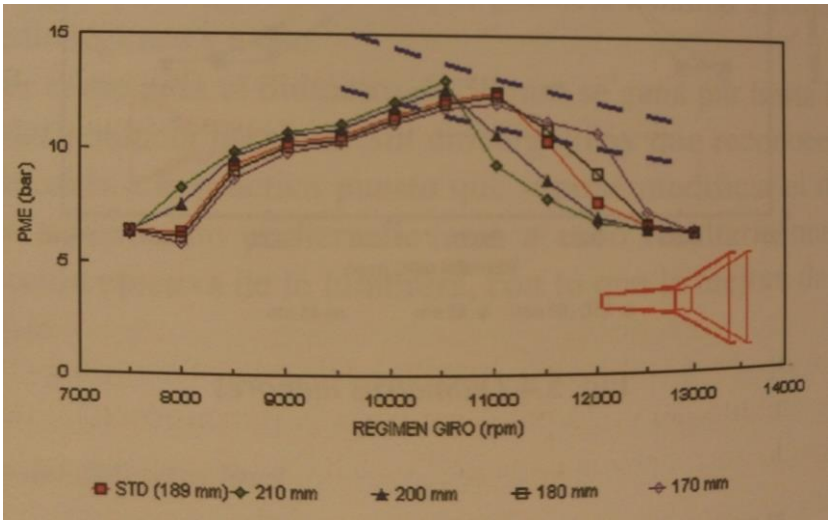

Figura 29: Variación de la longitud del codo

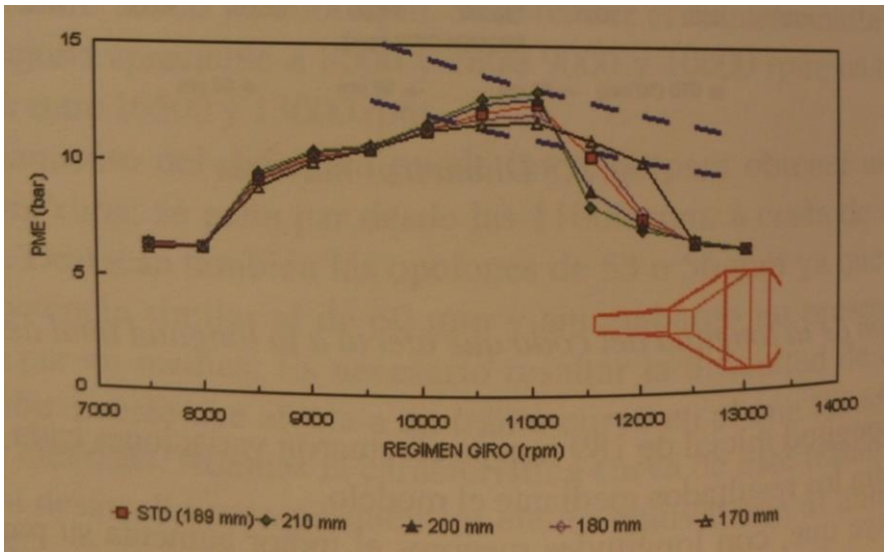

Figura 30: Variación de las longitudes del difusor y de la cámara de expansión

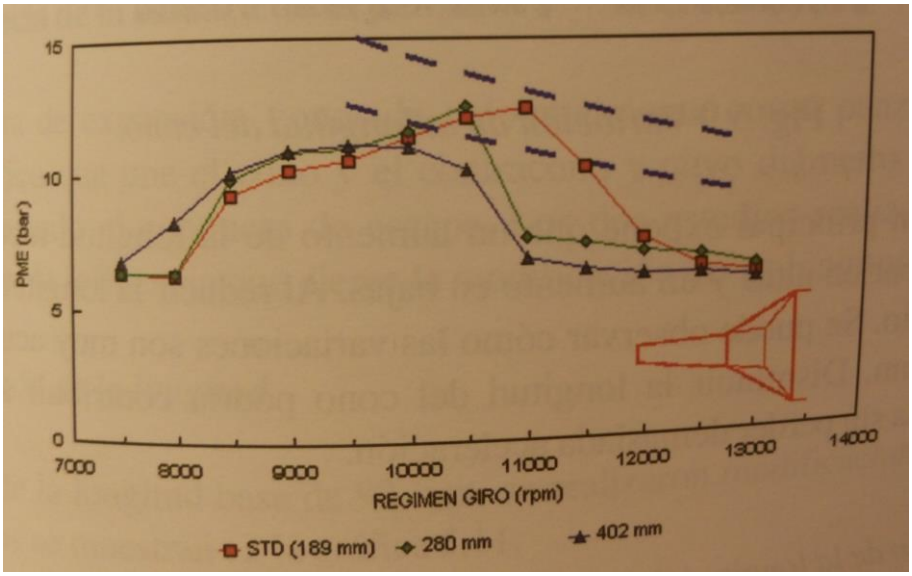

Figura 31: Variación de la longitud del codo conservando el pendiente

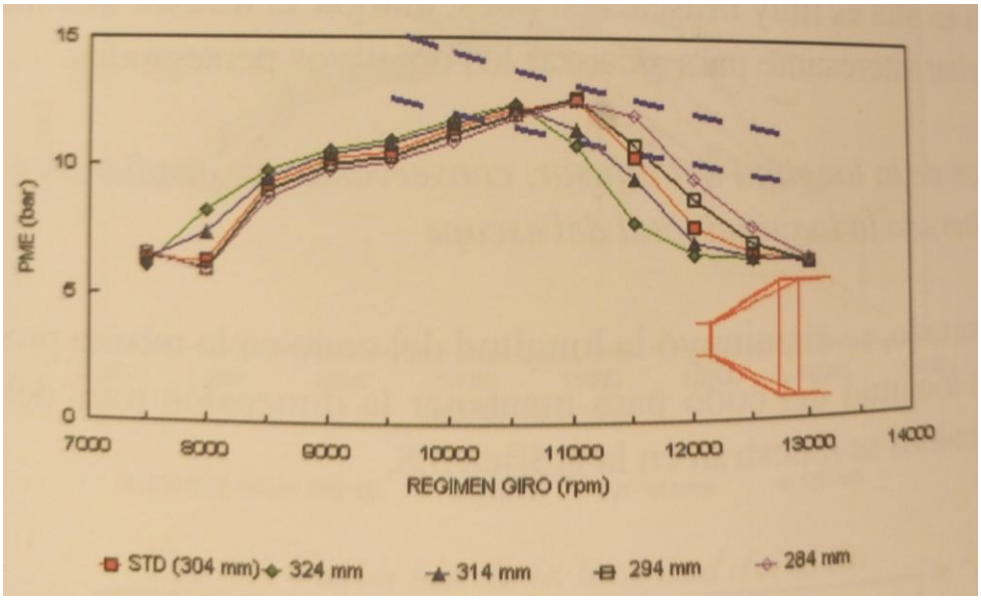

Figura 32: Variación de la longitud del cono

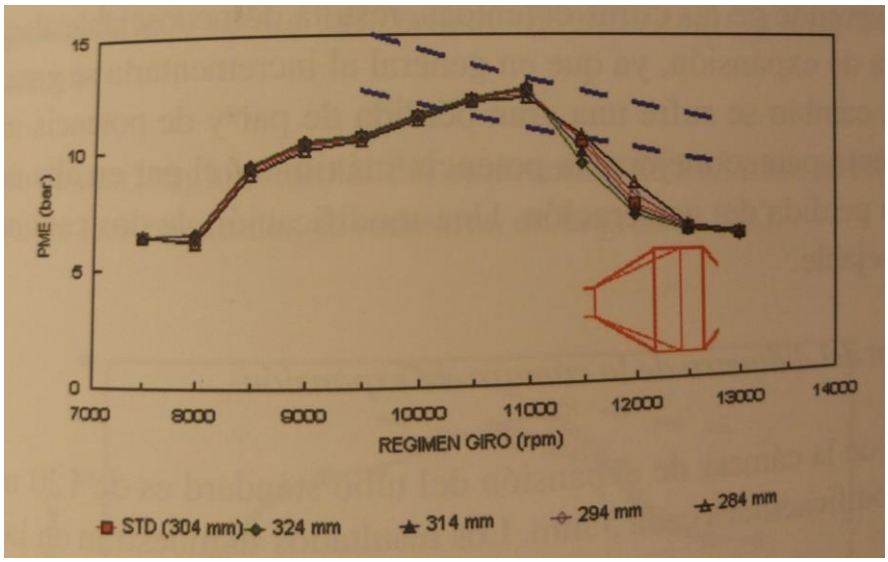

Figura 33: Variación de la longitud del cono a costa de la cámara de expansión

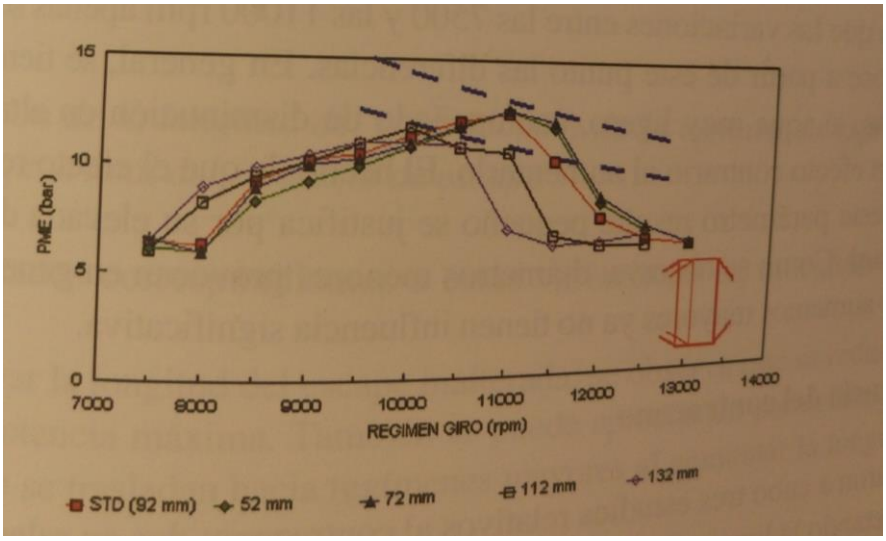

Figura 34: Variación de la longitud de la cámara de expansión

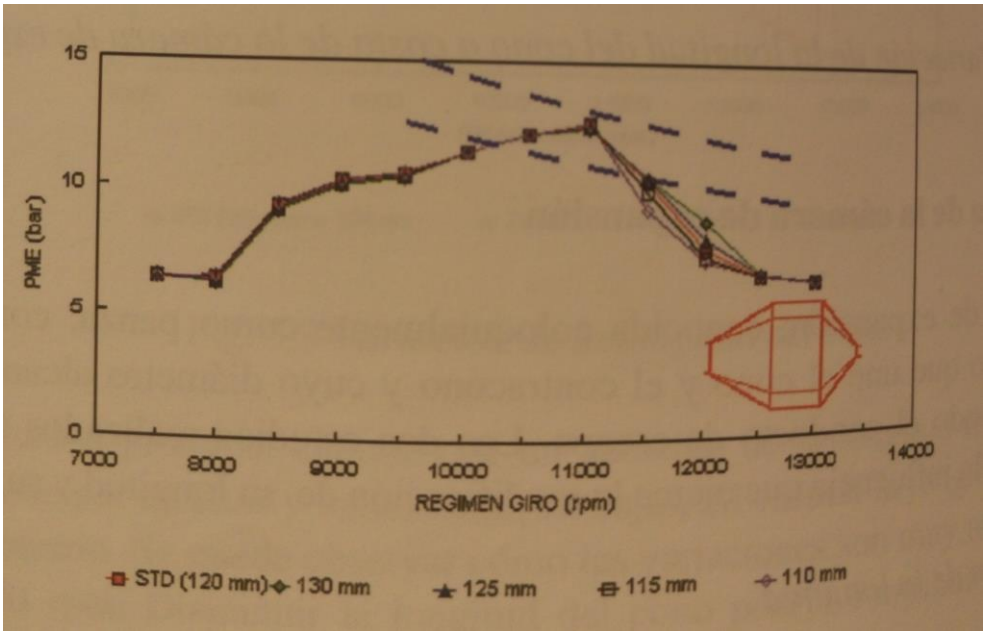

Figura 35: Variación de diámetro de la cámara de expansión

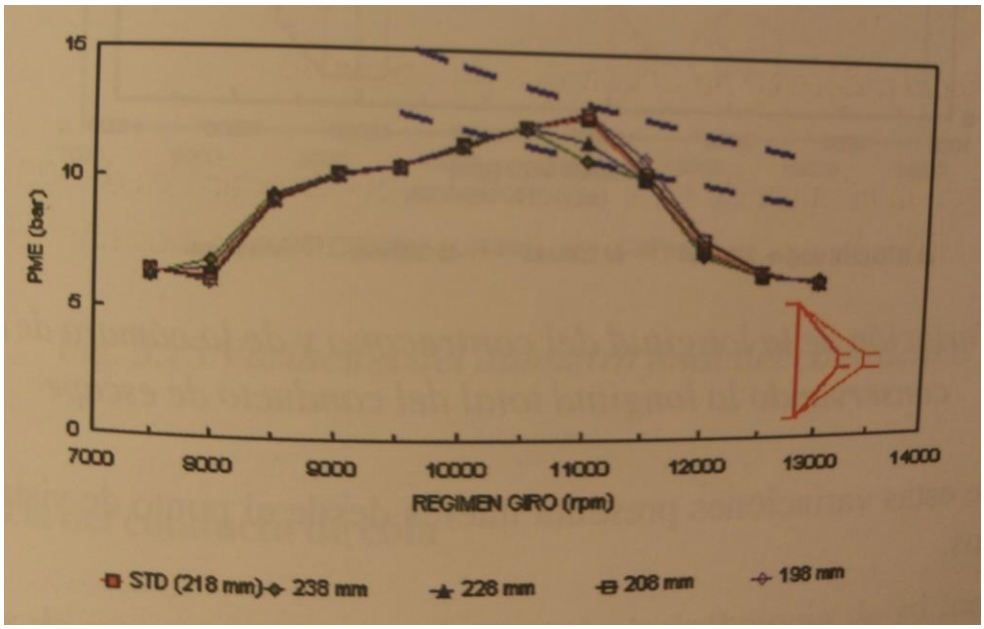

Figura 36: Variación de la longitud del contracono

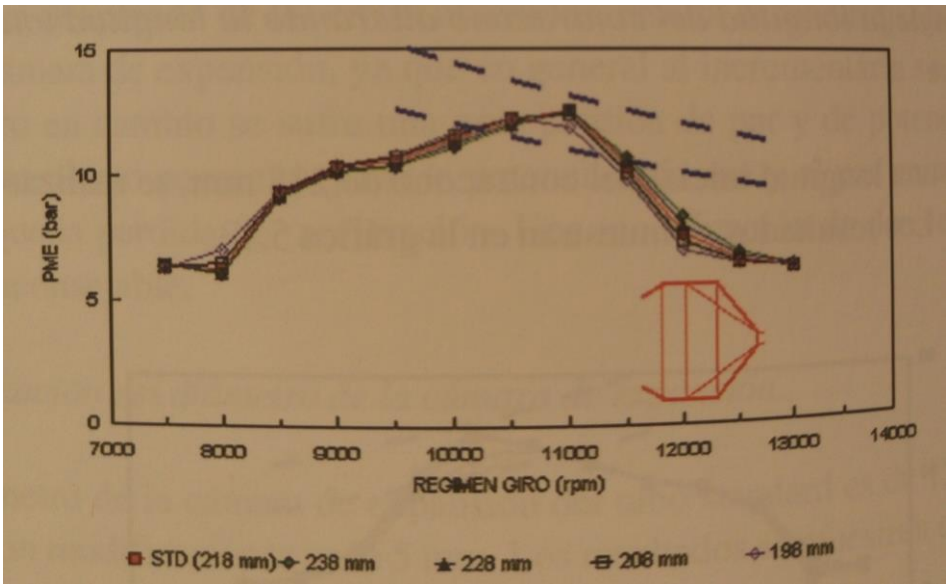

Figura 37: Variación de la longitud del contracono y la cámara a longitud constante

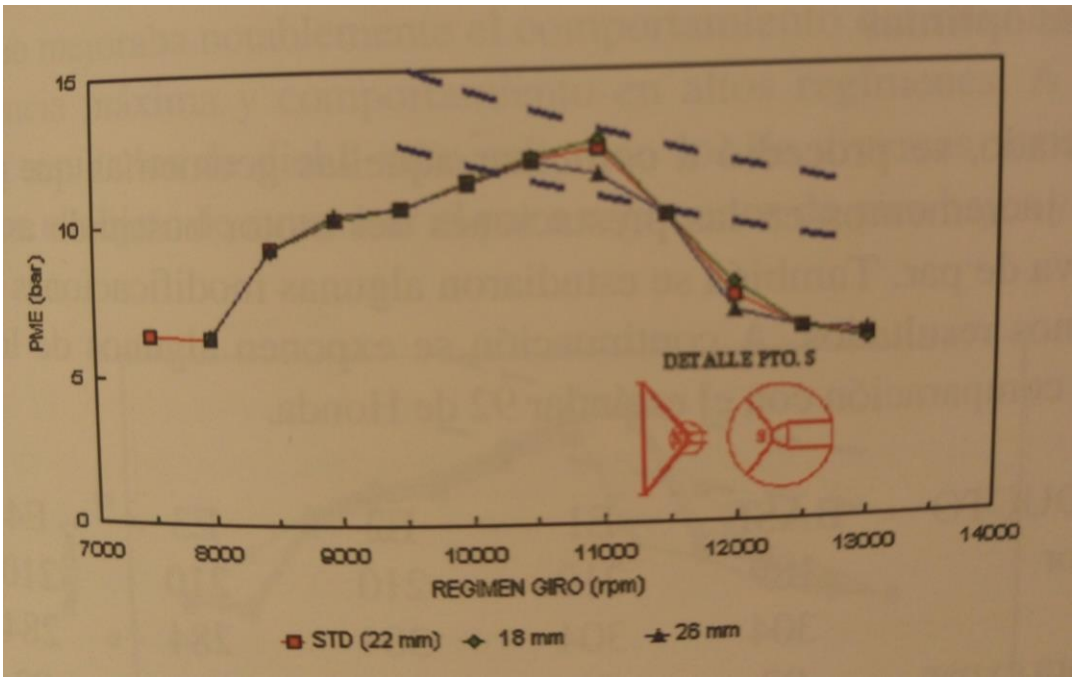

Figura 38: Variación del diámetro final del contracono

3. Código de Matlab

3.1. Funciones

3.1.1. Funciones implementadas del libro

```
function [p1d,p2d,X1d,X2d]=discont(X1,X2,a01,a02)
X2d = (2*X2-X1*(1-(a01)/(a02)))/(1+a01/a02);
p2d = p0*X2d^G7;
X1d = X1 + X2d - X2;
p1d = p0*X1d^G7;
end

function [Xp,Xq,alphap,alphaq,xp,dt]=element(Xr,Xl,Xr1,Xl1,L,dt) %pr--
>,pl<--|ELEMENT|pr1-->,pl1<--
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup();
% Xr = (Pr)^G17;
% Xl = (Pl)^G17;
% Xr1 = (Pr1)^G17;
% Xl1 = (Pl1)^G17;
XX = [Xr,Xl,Xr1,Xl1];
dt2 = [];
for i=1:4,
    P = XX(i)^G7;
    [p,X,c,alpha,rho,M,m]=propertieswave(P);
    dt1 = (L/alpha);
    dt2 = [dt2;dt1];
end
if dt == 0,
    dt = min(dt2);
end
V = (pi/4)*d^2*L;
Xj = ((Xr+Xl-1)+(Xr1+Xl1-1))/2;
pj = p0*Xj^G7;
rhoj = rho0*Xj^G5;
Tj = T0*Xj^2;
aj = a0*Xj;
mj = rhoj*V;
E = (a0*dt)/L;
A1 = E*(Xr1-Xr);
B = E*(Xl-Xl1);
C = Xr1/A1;
D = Xl/B;
FR = (G6+(1/A1))/G4;
FL = (G6+(1/B))/G4;
if Xr == Xr1 && Xl == Xl1,
    Xq = Xl1;
    Xp = Xr1;
elseif Xl == Xl1;
    Xq = Xl1;
    Xp = (1+C+G4*Xq)/(G6+(1/A1));
elseif Xr == Xr1,
    Xp = Xr1;
    Xq = (1+D+G4*Xp)/(G6+(1/B));
```

```
else
    Xp = (1+D+FL+FL*C)/(G4*(FR*FL-1));
    Xq = (1+C+FR+FR*D)/(G4*(FR*FL-1));
end
alphap = a0*(G6*Xp-G4*Xq-1);
alphaq = -a0*(G6*Xq-G4*Xp-1);
xp=alphap*dt;
Xs = Xp+Xq-1;
rhos = rho0*Xs^G5;
cs = G5*a0*(Xp-Xq);
end

function [Pr,Xr]=openend(Xin)
%superposition with atm pressure
Xr = 2- Xin;
Pr = Xr^7;
% csr = 2*cin;
End

function
[P1f,P2f,X1f,X2f,dpf,psf,Xsf,csf,alpharf,alphalf,dQ]=ploss(ps,cs,rhos,as,
Ts)
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(Ts);
N = 1000; %[rpm]
angle = 2/360;
dt = angle*60/N;
Cf = 0.004; %(0.003+0.008)/2;
Tau = Cf*rhos*(cs^2)/2;
dx = cs*dt;
%F = pi*dtau*dx;
dpf = 2*Cf*rhos*(cs^3)*dt/d;
psf = ps - dpf; %+-
Xsf = (psf/p0)^G17;
csf = cs + (psf-ps)/(rhos*cs);
X1f = 0.5*(1+Xsf+(csf/(G5*a0)));
X2f = 1 + Xsf -X1f;
p1f = p0*X1f^G7;
p2f = p0*X2f^G7;
P1f = p1f/p0;
P2f = p2f/p0;
alpharf = as +csf;
alphalf = -as +csf;
dQ = (pi*d*Cf*rhos*cs^4*dt^2)*0.5;
end

function [p,X,c,alpha,rho,M,m]=propertieswave(P,T0)
%Properties from waves
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T0);
p = P*p0;
X = P^((y-1)/(2*y));
c = 5*a0*(X-1);
alpha = a0*(G6*X-G5);
rho = rho0*X^5;
M = G5*(X-1)/X;
m = rho*A*c;
end

function [A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T0)
d = 25;
d = d/1000;
A = pi*(d^2)/4;
```

```
p0 = 101325; %[Pa]
y = 1.4;
% T0 = 293.15; %[K]
R = 287; %[J/kg*K]
a0 = (y*R*T0)^0.5; %[m/s]
rho0 = (p0/(R*T0)); %[kg/m3]
G3 = (4-2*y)/(y-1);
G4 = (3-y)/(y-1);
G5 = (2)/(y-1);
G6 = (y+1)/(y-1);
G7 = (2*y)/(y-1);
G17 = G7^-1;
G35 = (y)/(y-1);
G67 = (y+1)/(2*y);
End

function [Psh,alphash,csh]=shock(p,X)
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup();
Psh = p0/p;
alphash = a0*X*(G67*Psh+G17)^0.5 + G5*a0*(X-1);
csh =  (a0*(p/p0-1))/(y*alphash/a0);
end

function [Pr1,Pr2,Xr1,Xr2,T02]=sudden(A1,A2,Xin1,Xin2,sn,T0) %Xin1 --->
Xe,Xi, Xin2 ---> X=1.
Ar = A2/A1;
if Ar < 1/6 || Ar > 6,
   print('Warning, not relyable result!')
end
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T0);
Xr1 = ((1-Ar)*Xin1+2*Xin2*Ar)/(1+Ar);
Xr2 = (2*Xin1-Xin2*(1-Ar))/(1+Ar);
T02 = p0/(R*rho0*Xr2^G5);
if sn,
    if Ar > 1 || Ar < 1,
        %Variables: pr1(Xr1),pr2(Xr2),T02(a02).
        [Xr1,Xr2,T02]=NWRPH(Xin1,Xin2,T0,A1,A2,Xr1,Xr2,T02);
        Ms1 = (G5*abs(Xin1-Xr1))/(Xin1+Xr1-1);
        if Ms1 > 1,
            Xr1 = (1+G4*Xin1)/G6;
            [Xr1,Xr2,T02]=NWRPH(Xin1,Xin2,T0,A1,A2,Xr1,Xr2,T02);
            print('supersonic flow')
        end
    elseif Ar < 1,
        [Xr1,Xr2]=NWRPH2(Xin1,Xin2,T0,A1,A2,Xr1,Xr2);
        Ms2 = (G5*abs(Xin2-Xr2))/(Xin2+Xr2-1);
        if Ms2 >= 1,
            Xr2 = (1+G4*Xin2)/G6;
            [Xr1,Xr2]=NWRPH2(Xin1,Xin2,T0,A1,A2,Xr1,Xr2);
            print('supersonic flow')
        end
    end
end

pr1 = p0*Xr1^G7;
pr2 = p0*Xr2^G7;
Pr1 = pr1/p0;
Pr2 = pr2/p0;
End
```

```
function
[ps,Ts,cs,as,Xs,rhos,ms,alphar,alphal]=superposition(Pl,Pr,T1)%,c1,c2,alp
ha1,alpha2)
%   -----> right +
%   <----- left  -
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T1);
X1 = Pr^G17;
X2 = Pl^G17;
c1 = 5*a0*(X1-1);
alpha1 = a0*(G6*X1-G5);
c2 = -5*a0*(X2-1);
alpha2 = -a0*(G6*X2-G5);
a1 = a0*X1;
a2 = a0*X2;
cs = G5*a0*(X1-X2);
Xs = X1 + X2 - 1;
as = a0*Xs;
Ts = p0/(R*rho0*Xs^G5);
Ps = Xs^G7;
ps = Ps*p0;
rhos = rho0*Xs^G5;
ms = G5*a0*rho0*A*((X1+X2-1)^G5)*(X1-X2);
%ms1 = rhos*A*cs;
alphar = as + cs;
alphal = -as + cs;
Ms = abs(G5*a0*(X1-X2)/(a0*Xs));
if Ms >= 1,
    psn = ps*((2*y*Ms^2)/(y+1) - (y-1)/(y+1));
    Msn = ((Ms^2+2/(y-1))/(2*y*Ms^2/(y-1) - 1))^0.5;
    Tau1 = ((Ms^2 + 2/(y-1))/(2*y*Ms^2/(y-1) - 1));
    Tau2 = (2*y*Ms^2)/(y+1) - (y-1)/(y+1);
    Tau3 = ((y-1)/2)*(Tau1)^0.5;
    Tau4 = Xs*Tau2^((y-1)/(2*y));
    X1n = (1+Tau4+Tau3*Tau4)/(2);
    X2n = (1+Tau4-Tau3*Tau4)/2;
    Xsn = X1n + X2n -1;
    p1n = p0*X1n^G7;
    p2n = p0*X2n^G7;
end
```

3.1.2. Funciones auxiliares creadas para la implementación

```
function Data=createwave(dt,t,x0,N,P,T0,s)
[p,X,c,alpha,rho,M,m] = propertieswave(P,T0);
x = x0 + alpha*dt;
L=0.010;
if abs(x) > L,
    W = 1;
    x = x - L;
    N = N + s; %element index
else
    W=0;
end
x0 = x;
Data = [W x N P X T0 s];

End
```

```
function i=findwave(Data,Node,s)
i=1;
[RW,COL]=size(Data);
for e=1:RW,
    if Data(e,3)==Node && Data(e,7)==s,
        i=e;
    end
end

function [T2]=heat(A1,A2,L,P1,T1,Tw)
[p,X,c,alpha,rho,M,m]=propertieswave(P1,T1);
c=abs(c);
if A1==A2,
    S=pi*A1*L;
elseif A1<A2,
    S=0.5*pi*A2*(L+L*A1/(A2-A1));
else
    S=0.5*pi*A1*(L+L*A2/(A1-A2));
end
Ck = 6.1944e-3 + 7.3814e-5*T1 - 1.2491e-8*T1^2;
d = ((A1/pi)*4)^0.5;
mu = 7.457e-6 + 4.1547e-8*T1 - 7.4793e-12*T1^2;
Re = c*rho*d/mu;
Cf = 0.0791/(Re^0.25);
Ch = Ck*Cf*Re/(2*d);
dQ = S*Ch*L*(Tw-T1);
m = c*A1;
h1 = 1.1402*T1 - 66.396;
h2 = (dQ/m) + h1;
T2 = (h2 + 66.396)/1.1402;
end

function [Xr1,Xr2,T02]=NWRPH(Xi1,Xi2,T01,A1,A2,Xr1,Xr2,T02)
%Parameters definition
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T01);
rho01 = p0/(R*T01);
a01 = (y*R*T01)^0.5;

tol = 1e-4; %converge tolerance
%initial guess
% Xr1 = ;
% Xr2 = ;
% T02 = ;
iter = 0; %iterations
xnew =[Xr1;Xr2;T02];
xold =zeros(size(xnew));
while norm(xnew-xold) > tol,
    iter = iter +1;
    xold = xnew;
    %update variables
    Xr1 = xold(1);
    Xr2 = xold(2);
    T02 = xold(3);
    %Functions
    f = rho01*((Xi1+Xr1-1)^G5)*A1*G5*a01*(Xi1-
Xr1)+(p0/(R*T02))*((Xi2+Xr2-1)^G5)*A2*G5*((y*R*T02)^0.5)*(Xi2-Xr2); %OK
    g = ((G5*a01*(Xi1-Xr1))^2+G5*(a01^2)*(Xi1+Xr1-1)^2)-
((G5*((y*R*T02)^0.5)*(Xi2-Xr2))^2+G5*y*R*T02*(Xi2+Xr2-1)^2); %OK
    h = p0*A2*((Xi1+Xr1-1)^G7-(Xi2+Xr2-1)^G7)+(rho01*((Xi1+Xr1-
1)^G5)*A1*G5*a01*(Xi1-Xr1))*(G5*a01*(Xi1-Xr1)+G5*((y*R*T02)^0.5)*(Xi2-
Xr2)); %OK
```

```
    %Partial derivatives
    dfdXr1 = rho01*G5*((Xi1+Xr1-1)^(G5-1))*A1*G5*a01*(Xi1-Xr1)-
rho01*((Xi1+Xr1-1)^G5)*A1*G5*a01; %OK
    dfdXr2 = (p0/(R*T02))*G5*((Xi2+Xr2-1)^(G5-
1))*A2*G5*((y*R*T02)^0.5)*(Xi2-Xr2)-(p0/(R*T02))*((Xi2+Xr2-
1)^G5)*A2*G5*((y*R*T02)^0.5); %OK
    dfdT02 = (-p0*R/((R*T02)^2))*((Xi2+Xr2-
1)^G5)*A2*G5*((y*R*T02)^0.5)*(Xi2-Xr2)+(p0/(R*T02))*((Xi2+Xr2-
1)^G5)*A2*G5*(1/(2*(y*R*T02)^0.5))*y*R*(Xi2-Xr2); %OK

    dgdXr1 = 2*(G5*a01*(Xi1-Xr1))*(-G5*a01) + G5*(a01^2)*2*(Xi1+Xr1-1);
    dgdXr2 = -2*(G5*((y*R*T02)^0.5)*(Xi2-Xr2))*(-G5*((y*R*T02)^0.5)) -
G5*(y*R*T02)*2*(Xi2+Xr2-1);
    dgdT02 = -(G5*(Xi2-Xr2))*y*R - G5*((Xi2+Xr2-1)^2)*2*y*R;

    dhdXr1 = p0*A2*G7*((Xi1+Xr1-1)^(G7-1)) + rho01*G5*((Xi1+Xr1-1)^(G5-
1))*A1*G5*a01*(Xi1-Xr1)*( G5*a01*(Xi1-Xr1) + G5*((y*R*T02)^0.5)*(Xi2-Xr2)
) + rho01*((Xi1+Xr1-1)^G5)*(-A1*G5*a01)*( G5*a01*(Xi1-Xr1) +
G5*((y*R*T02)^0.5)*(Xi2-Xr2) ) + rho01*((Xi1+Xr1-1)^G5)*A1*G5*a01*(Xi1-
Xr1)*(-G5*a01);
    dhdXr2 = -p0*A2*G7*((Xi2+Xr2-1)^(G7-1)) + rho01*((Xi1+Xr1-
1)^G5)*A1*G5*a01*(Xi1-Xr1)*(-G5*(y*R*T02)^0.5);
    dhdT02 = rho01*((Xi1+Xr1-1)^G5)*A1*G5*a01*(Xi1-Xr1)*G5*(Xi2-
Xr2)*(1/(2*(y*R*T02)^0.5))*y*R;

    %Jacobian matrix
    J = [dfdXr1 dfdXr2 dfdT02;dgdXr1 dgdXr2 dgdT02;dhdXr1 dhdXr2 dhdT02];
    %Newton-Raphson Method
    xnew = xold -J\[f;g;h];
    disp(sprintf('iter=%6.15f, Xr1=%6.15f, Xr2=%6.15f, T02=%6.15f',
iter,xnew));
    if iter > 1000,
        disp 'Failed to converge';
        break
    end
end
% Xr1 = real(Xr1);
% Xr2 = real(Xr2);
% T02 = real(T02);

function [Xr1,Xr2]=NWRPH2(Xi1,Xi2,T01,A1,A2,Xr1,Xr2)
%Parameters definition
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T01);
rho01 = p0/(R*T01);
a01 = (y*R*T01)^0.5;

tol = 1e-4; %converge tolerance
%initial guess
% Xr1 = ;
% Xr2 = ;
% T02 = ;
iter = 0; %iterations
xnew =[Xr1;Xr2];
xold =zeros(size(xnew));
while norm(xnew-xold) > tol,
    iter = iter +1;
    xold = xnew;
    %update variables
    Xr1 = xold(1);
    Xr2 = xold(2);
    %Functions
```

```
    f = ((Xi1+Xr1-1^G5)*A1*(Xi1-Xr1)+((Xi2+Xr2-1)^G5)*A2*(Xi2-Xr2));
    g = (G5*(Xi1-Xr1)^2+G5*(Xi1+Xr1-1)^2)-(G5*(Xi2-Xr2)^2+G5*(Xi2+Xr2-
1)^2);
    %Partial derivatives
    dfdXr1 = G5*((Xi1+Xr1-1)^(G5-1))*A1*(Xi1-Xr1)+((Xi1+Xr1-1)^G5)*(-A1);
%OK
    dfdXr2 = G5*((Xi2+Xr2-1)^(G5-1))*A2*(Xi2-Xr2)+((Xi2+Xr2-1)^G5)*(-A2);
%OK

    dgdXr1 = 2*G5*(Xi1-Xr1)*(-1)+2*G5*(Xi1+Xr1-1);
    dgdXr2 = -(2*G5*(Xi2-Xr2)*(-1)+2*G5*(Xi2+Xr2-1));

    %Jacobian matrix
    J = [dfdXr1 dfdXr2;dgdXr1 dgdXr2];
    %Newton-Raphson Method
    xnew = xold -J\[f;g];
    disp(sprintf('iter=%6.15f, Xr1=%6.15f, Xr2=%6.15f, T02=%6.15f',
iter,xnew));
    if iter > 1000,
        disp 'Failed to converge';
        break
    end
end
% Xr1 = real(Xr1);
% Xr2 = real(Xr2);

end

function [P1f]=ploss2(A1,P1,T1,dt)
[p,X,c,alpha,rho,M,m]=propertieswave(P1,T1);
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(T1);
d = ((A1/pi)*4)^0.5;
mu = 7.457e-6 + 4.1547e-8*T1 - 7.4793e-12*T1^2;
Re = c*rho*d/mu;
Cf = 0.0791/(Re^0.25);
dpf = 2*Cf*rho*(c^3)*dt/d;
% dpf = (8*Cf*0.01*(c*A1)^2)/(9.8*pi^2*d^5);
psf = p - dpf; %+-
Xsf = (psf/p0)^G17;
csf = c + (psf-p)/(rho*c);
X1f = 0.5*(1+Xsf+csf/(G5*a0));
P1f = X1f^G7;
end
```

3.2. Script principal

```
[A,d,p0,y,T0,R,a0,rho0,G3,G4,G5,G6,G7,G17,G35,G67]=setup(700);
 %Geometry 1

 Lt = [28 8 13 23 0];
 LL = Lt(1);
 LLL = LL;
 A0 = [46 104 125 125 18];
 AA = A0(1);
 e=1;
 A=[];
 for i=1:72,
```

```
    if i==LLL,
        pen=(A0(e+1)-A0(e))/(LL);
        for z=1:LL,
            A=[A;AA+pen*(z-1)];
        end
        e=e+1;
        LLL = LLL+Lt(e);
        LL = Lt(e);
        AA = A0(e);
    end
end

 %Geometry 2

%  Lt = [36 8 13 23 0];
%  LL = Lt(1);
%  LLL = LL;
%  A0 = [46 104 125 125 18];
%  AA = A0(1);
%  e=1;
%  A=[];
%  for i=1:80,
%      if i==LLL,
%          pen=(A0(e+1)-A0(e))/(LL);
%          for z=1:LL,
%              A=[A;AA+pen*(z-1)];
%          end
%          e=e+1;
%          LLL = LLL+Lt(e);
%          LL = Lt(e);
%          AA = A0(e);
%      end
%  end

%  A = [20;20;20;20;30;40;40;40;30;20];
 col = length(A);
 L = ones(col,1)*10/1000;

%Initial wave
P(1) = 1.2;
T(1) = 700;
s(1)=1;

Z=[];
Tps=[];
MS=[];

ID = 1;
dt=5e-6;
ttotal=3e-3;
t=0;
Nmax = col;
Data=createwave(dt,t,0,1,P(1),T(1),1);
DataR = ones(col,3);
DataL = ones(col,3);
tol = 0.001;
while t<ttotal,
    [RW,COL]=size(Data);
    for i=1:RW,
```

```
        Data(i,:) =
createwave(dt,t,Data(i,2),Data(i,3),Data(i,4),Data(i,6),Data(i,7));
%update wave
    end

%     a =1;
%     while a<RW,
%         F = find(Data(:,3)==Data(a,3));
%         S = find(Data(:,7)==Data(a,7));
%         C = intersect(F,S);
%         if isempty(C),
%             a = a + 1;
%         else
%             PP1 = Data(C(1),4);
%             TT = Data(C(1),6);
%             SS1 = Data(C(1),7);
%             for e=2:length(C),
%                 PP2 = Data(C(e),4);
%                 SS2 = Data(C(e),7);
%
[PP1,TT,cs1,as1,Xs1,rhos1,ms1,alphar1,alphal1]=superposition(PP1,PP2,TT);
%                 PP1=PP1/p0;
%             end
%             Data(a,:)=
createwave(0,t,Data(a,2),Data(a,3),PP1,TT,Data(a,7));
%             a=a+1;
%             for e=2:length(C),
%                 Data(C(e),:)=[];
%             end
%         end
%         [RW,COL]=size(Data);
%     end
    DataR = ones(col,3);
    DataL = ones(col,3);
    NR = [];
    NL = [];
    for i=1:RW,
        if Data(i,1)==1 && Data(i,3)<=Nmax && Data(i,3)>0,
            if Data(i,7)==1,
                FR = find(NR==Data(i,3));
                if isempty(FR),
                    DataR(Data(i,3),1)=Data(i,4);
                    DataR(Data(i,3),2)=Data(i,5);
                    DataR(Data(i,3),3)=Data(i,6);
                else
                    PP1 = DataR(Data(i,3),1);
                    TT = DataR(Data(i,3),3);
                    PP2 = Data(i,4);

[PP1,TT,cs1,as1,Xs1,rhos1,ms1,alphar1,alphal1]=superposition(PP1,PP2,TT);
                    PP1=PP1/p0;
                    DataR(Data(i,3),1)=PP1;
                    DataR(Data(i,3),2)=PP1^G17;
                    DataR(Data(i,3),3)=TT;
                    Data(i,1)=-1;
                end
                NR = [NR;Data(i,3)];
            elseif Data(i,7)==-1,
                FL = find(NL==Data(i,3));
                if isempty(FR),
                    DataL(Data(i,3),1)=Data(i,4);
                    DataL(Data(i,3),2)=Data(i,5);
```

```
                    DataL(Data(i,3),3)=Data(i,6);
                else
                    PP1 = DataL(Data(i,3),1);
                    TT = DataL(Data(i,3),3);
                    PP2 = Data(i,4);
                    if TT<300,
                        TT = 690;
                    end

[PP1,TT,cs1,as1,Xs1,rhos1,ms1,alphar1,alphal1]=superposition(PP1,PP2,TT);
                    PP1=PP1/p0;
                    DataL(Data(i,3),1)=PP1;
                    DataL(Data(i,3),2)=PP1^G17;
                    Data(i,1)=-1;
                end
                NL = [NL;Data(i,3)];
            end
%           elseif Data(i,3)>Nmax,
%               if Data(i,7)==1,
%                   DataR(Nmax,:)=[1,1,1];
%               elseif Data(i,7)==-1,
%                   DataL(Nmax,:)=[1,1,1];
%               end
%           elseif Data(i,3)==1,
%               if Data(i,7)==1,
%                   DataR(1,:)=[1,1,1];
%               elseif Data(i,7)==-1,
%                   DataL(1,:)=[1,1,1];
%               end

        end
%           if Data(i,6)==1,
%               Data(i,6)=690;
%           end
    end
    for i=RW;
        if Data(i,1)==-1,
            Data(i,:)=[];
        end
    end
    for i=2:col,
        if DataR(i,1)~=1,
            if A(i)==A(i-1),
                PR = DataR(i,1);
                PL = DataL(i,1);
                TR = DataR(i,3);

[ps,Ts,cs,as,Xs,rhos,ms,alphar,alphal]=superposition(PL,PR,TR);

[P1f,P2f,X1f,X2f,dpf,psf,Xsf,csf,alpharf,alphalf,dQ]=ploss(ps,cs,rhos,as,
TR);
                PR = P1f;
                TR2 = heat(A(i),A(i),L(i),PR,TR,400);
                TR = TR2; %T change
                e=findwave(Data,i,1);
                Data(e,:) =
createwave(0,t,Data(e,2),Data(e,3),PR,TR,Data(e,7));
            else
                XR = DataR(i,2);
                XL = DataL(i,2);
                TR = DataR(i,3);
                A1=A(i-1);
```

```
                A2=A(i);
                [Pr1,Pr2,Xr1,Xr2,T02]=sudden(A1,A2,XR,XL,1,DataR(i,3));
                PR = Pr2;
                PL = Pr1;
                e=findwave(Data,i,1);

[ps,Ts,cs,as,Xs,rhos,ms,alphar,alphal]=superposition(PL,PR,TR);

[P1f,P2f,X1f,X2f,dpf,psf,Xsf,csf,alpharf,alphalf,dQ]=ploss(ps,cs,rhos,as,
TR);
                PR = P1f;
                TR2 = heat(A(i),A(i),L(i),PR,T02,400);
                TR = TR2; %T change
                if abs(PR-1)>tol,
                    Data(e,:) =
createwave(0,t,Data(e,2),Data(e,3),PR,TR,Data(e,7));
                end
                if  abs(PL-1)>tol,
                    ID = ID +1;
                    Data(ID,:) = createwave(0,t,0,(Data(e,3)-
1),PL,DataR(i,3),-1);
                end
            end
            if i==Nmax,
                XR = DataR(i,2);
                [Pr,Xr]=openend(XR);
                if PR-1>tol,
                    ID = ID + 1;
                    Data(ID,:) = createwave(0,t,0,i,Pr,DataR(i,3),-1);
                end
            end
        end
        if DataL(i,1)~=1,% && DataR(i,1)==1,
            if A(i+1)==A(i),
                PR = DataR(i,1);
                PL = DataL(i,1);
                TL = DataL(i,3);

[ps,Ts,cs,as,Xs,rhos,ms,alphar,alphal]=superposition(PL,PR,TL);

[P1f,P2f,X1f,X2f,dpf,psf,Xsf,csf,alpharf,alphalf,dQ]=ploss(ps,cs,rhos,as,
TL);
                PL = P2f; %-dpf
%                 TL2 = heat(A(i),A(i),L(i),PL,TL,400);
%                 TL = TL2; %T change
                e=findwave(Data,i,-1);
                Data(e,:) =
createwave(0,t,Data(e,2),Data(e,3),PL,TL,Data(e,7));
            else
                XR = DataR(i,2);
                XL = DataL(i,2);
                A1=A(i);
                A2=A(i-1);
                [Pr1,Pr2,Xr1,Xr2,T02]=sudden(A1,A2,XL,XR,1,DataL(i,3));
                PL = Pr2;
                PR = Pr1;
                e=findwave(Data,i,-1);
                if abs(PL-1)>tol,
                    Data(e,:) =
createwave(0,t,Data(e,2),Data(e,3),PL,T02,Data(e,7));
                end
                if  abs(PR-1)>tol,
```

```
                    ID = ID +1;
                    Data(ID,:) =
createwave(0,t,0,(Data(e,3)+1),PR,DataR(i,3),1);
                end
            end
        end
    end
    Yplot = [];
    Xplot = [];
    for i=1:ID,
        if Data(i,3)>0 || Data(i,3)<=col,
            Yplot = [Yplot;Data(i,4)];
            Xplot = [Xplot;(Data(i,7)*Data(i,2)*100+Data(i,3))];
        end
    end
    figure(1)
    plot(Xplot,Yplot,'o')
    grid on
    xlim([1 col]);
    ylim([0 1.5]);
    set(gca,'ytick',[0:0.1:2])
    hold on
    plot((1.4/125)*A,'r')
    hold off
    figure(2)
    [pR,XR,cR,alphaR,rhoR,MR,mR]=propertieswave(DataR(2,1),DataR(2,3));
    [pL,XL,cL,alphaL,rhoL,ML,mL]=propertieswave(DataL(2,1),DataL(2,3));
    if DataR(2,1)~=1,
        MS = [MS;mR];
        Z = [Z;DataR(2,1)];
        Tps = [Tps;t];
    elseif DataL(2,1)~=1,
        MS = [MS;mL];
        Z = [Z;DataL(2,1)];
        Tps = [Tps;t];
    else
        MS = [MS;0];
        Z=[Z;1];
        Tps = [Tps;t];
    end
    plot(Tps,Z);
    DataR = ones(col,3);
    DataL = ones(col,3);
    Data;
    t=t+dt;
end
```