Fitting tire data parameters - HW1

Luca Rigoni 247451

Table of Contents

Initialiastion

```
clc
clearvars
close all

% Set LaTeX as default interpreter for axis labels, ticks and legends
set(0,'defaulttextinterpreter','latex')
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');

% set(0,'DefaultFigureWindowStyle','docked');
set(0,'defaultAxesFontSize',  10)
set(0,'DefaultLegendFontSize',10)

addpath('tyre_lib/')

to_rad = pi/180;
to_deg = 180/pi;
```

Load data set

tyre geometric data:

Hoosier 18.0x6.0-10

18 diameter in inches

6.0 section width in inches

tread width in inches

```
diameter = 18*2.54; % [cm]
Fz0      = 670;   % [N] nominal load is given
R0       = diameter/2/100; % [m] get from nominal load R0 (m)

fprintf('Loading dataset ...')
```

```
Loading dataset ...
```

```


data_set_path = 'TTC_dataset/';
data = load ([data_set_path, 'B1464run23.mat']); % pure lateral
fields = fieldnames(data);
arrayData = struct();

for i = 1:numel(fields)
    value = data.(fields{i});
    % Check if it's a numeric or logical array AND has the same length as the others
    if isnumeric(value) || islogical(value)
        if isvector(value)
            arrayData.(fields{i}) = value(:); % ensure it's a column vector
        end
    end
end

writetable(struct2table(arrayData), 'B1464run23.csv')
tyre_data_tot = readtable('B1464run23.csv');
tyre_data_tot.FZ = -tyre_data_tot.FZ;
tyre_data_tot.SA = tyre_data_tot.SA*to_rad;
```

Note: since the Magic Formula of 1996 of Pacejka do not includes tire Pressure change, we directly cut the data with a certain level of pressure that can be approximated to be constant:

```
cut_start = 27813;
cut_end   = 55626;
smpl_range = (cut_start:cut_end)';

vec_samples = (1:1:length(smpl_range))';

tyre_data = tyre_data_tot(smpl_range, :);

fprintf('completed!\n')
```

```
completed!
```

1) Plot raw data

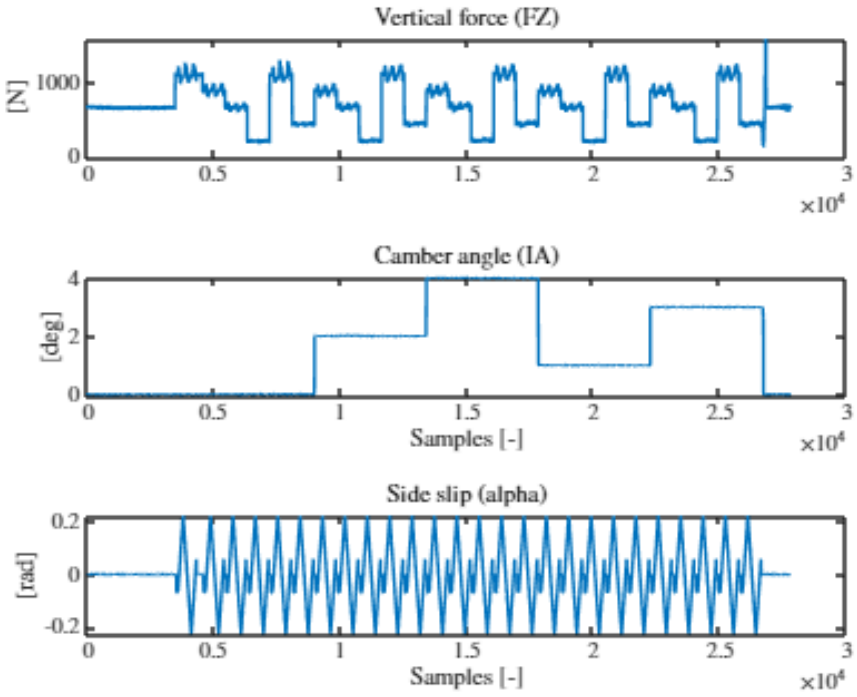
Plot of the row data in order to visualize and undertand how the grouping can be done:

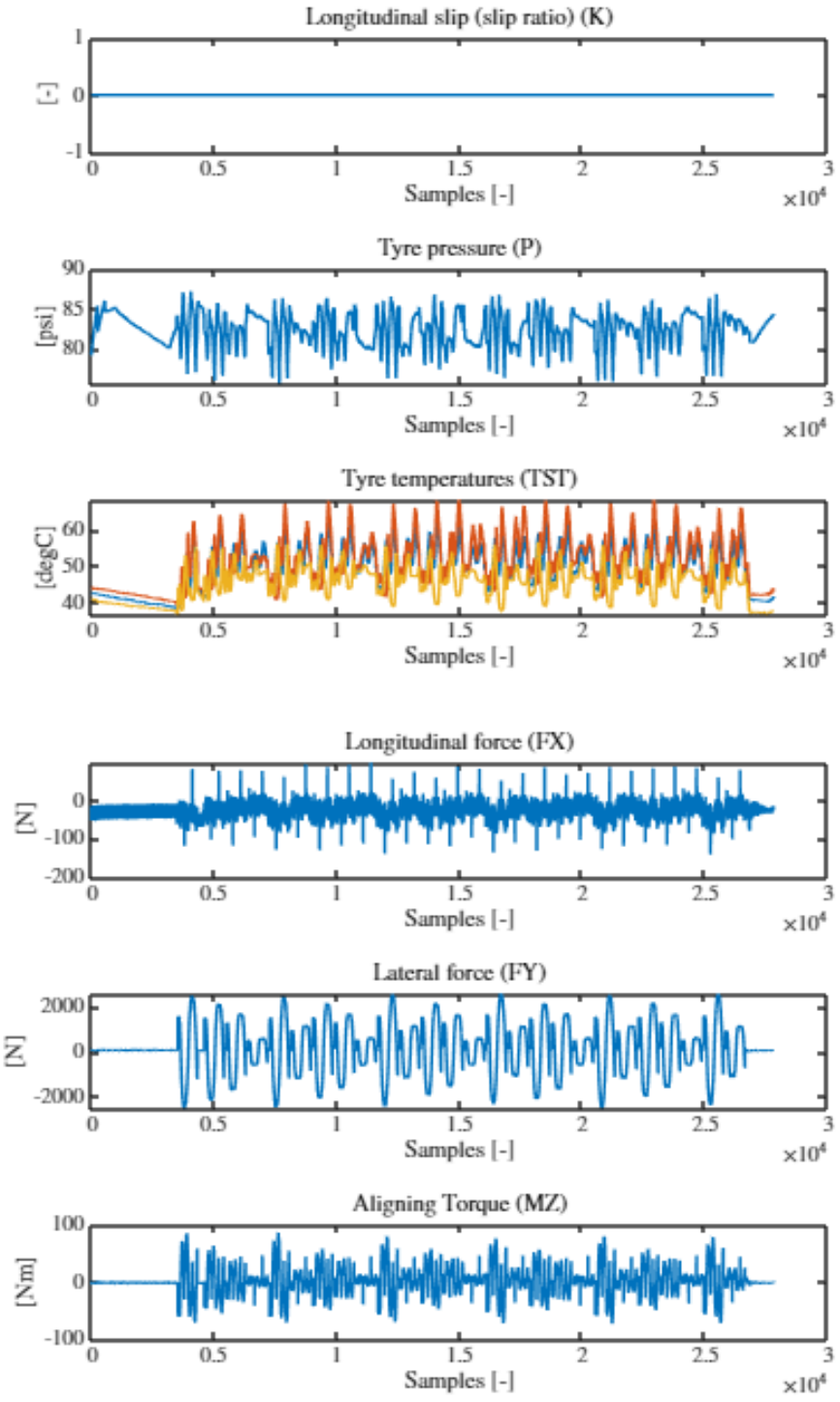
```
disp("Raw data")
```

```
Raw data
```

```
figure('Name','raw data')
tiledlayout(3,1)

ax_list(1) = nexttile;
plot(tyre_data.FZ)          % FZ = vertical load [N]
hold on
title('Vertical force (FZ)')
% xlabel('Samples [-]')
ylabel('[N]')

ax_list(2) = nexttile;
plot(tyre_data.IA)          % IA = inclination angle = camber [deg]
title('Camber angle (IA)')
xlabel('Samples [-]')
ylabel('[deg]')

ax_list(3) = nexttile;
plot(tyre_data.SA)          % SA = slip angle = alpha [deg]
hold on
title('Side slip (alpha)')
xlabel('Samples [-]')
ylabel('[rad]')

figure('Name','raw data2')
tiledlayout(3,1)

ax_list(4) = nexttile;
plot(tyre_data.SL)          % SL = longitudinal slip = slip ratio = K [-]
hold on
title('Longitudinal slip (slip ratio) (K)')
xlabel('Samples [-]')
ylabel('[-]')

ax_list(5) = nexttile;
plot(tyre_data.P)           % P = pressure [kPa]
hold on
title('Tyre pressure (P)')
xlabel('Samples [-]')
ylabel('[psi]')

ax_list(6) = nexttile;
plot(tyre_data.TSTC,'DisplayName','Center') % Tire Surface Temperature
hold on
plot(tyre_data.TSTO,'DisplayName','Out')
```

```
plot(tyre_data.TSTI,'DisplayName','Internal')
title('Tyre temperatures (TST)')
xlabel('Samples [-]')
ylabel('[degC]')

figure('Name','raw data3')
tiledlayout(3,1)

ax_list(7) = nexttile;
plot(tyre_data.FX)          % FZ = vertical load [N]
hold on
title('Longitudinal force (FX)')
xlabel('Samples [-]')
ylabel('[N]')

ax_list(8) = nexttile;
plot(tyre_data.FY)          % IA = inclination angle = camber [deg]
title('Lateral force (FY)')
xlabel('Samples [-]')
ylabel('[N]')

ax_list(9) = nexttile;
plot(tyre_data.MZ)          % SA = slip angle = alpha [deg]
hold on
title('Aligning Torque (MZ)')
xlabel('Samples [-]')
ylabel('[Nm]')

linkaxes(ax_list,'x')
```





Those plots are referred to raw data

Starting from the C**amber Angle (IA)**, we can see that the provided data set is divided into five different levels.

By observing the **Normal Load (FZ)** plot, it is clear we have 25 sectors, more precisely we have five levels per each camber angle level.

The side **Side Slip Angle (alpha)** is characterised by a linear variation of the quantity, instead of step profile like for the previous quantity. This behaviour is repeated for every interval defined by the vertical load data.

Observe the **longitudinal slip (K)** is purposely set constant to zero value for all samples

Select portion of data

Extract points at constant Inclination Angle (IA)

```
GAMMA_tol = 0.08;
idx.GAMMA_0 = 0.0-GAMMA_tol < tyre_data.IA & tyre_data.IA < 0.0+GAMMA_tol;
idx.GAMMA_1 = 1.0-GAMMA_tol < tyre_data.IA & tyre_data.IA < 1.0+GAMMA_tol;
idx.GAMMA_2 = 2.0-GAMMA_tol < tyre_data.IA & tyre_data.IA < 2.0+GAMMA_tol;
idx.GAMMA_3 = 3.0-GAMMA_tol < tyre_data.IA & tyre_data.IA < 3.0+GAMMA_tol;
idx.GAMMA_4 = 4.0-GAMMA_tol < tyre_data.IA & tyre_data.IA < 4.0+GAMMA_tol;

GAMMA_0  = tyre_data( idx.GAMMA_0, : );
GAMMA_1  = tyre_data( idx.GAMMA_1, : );
GAMMA_2  = tyre_data( idx.GAMMA_2, : );
GAMMA_3  = tyre_data( idx.GAMMA_3, : );
GAMMA_4  = tyre_data( idx.GAMMA_4, : );
```

Extract points at constant Vertical Load (FZ)

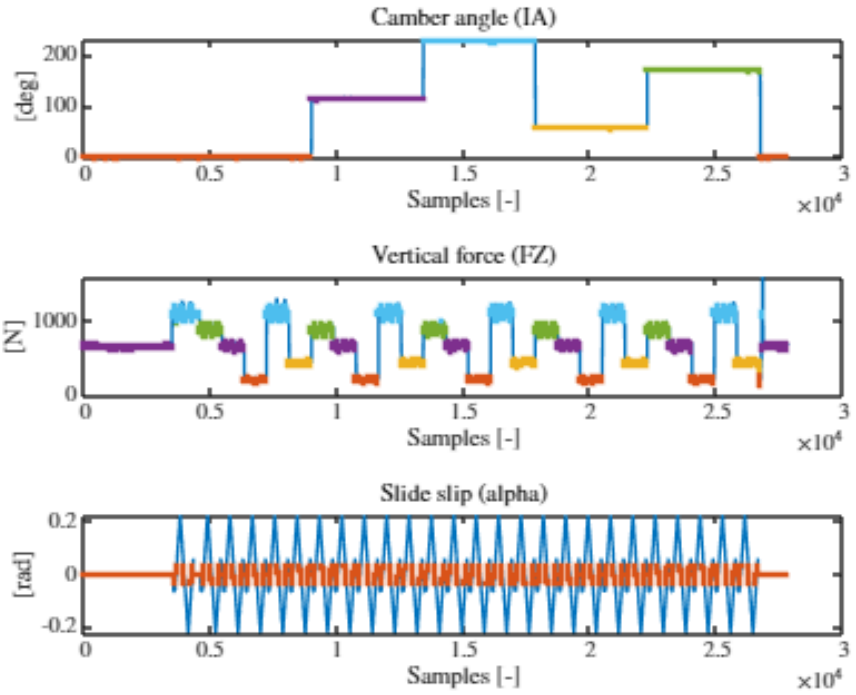
```
% Test data done at:
%   50lbf  ( 50*0.453592*9.81 =  222N )
%   100lbf (100*0.453592*9.81 =  445N )
%   150lbf (157*0.453592*9.81 =  670N )
%   200lbf (202*0.453592*9.81 =  890N )
%   250lbf (252*0.453592*9.81 = 1112N )

FZ_tol = 100;
idx.FZ_220  =  220-FZ_tol < tyre_data.FZ & tyre_data.FZ <  220+FZ_tol;
idx.FZ_445  =  445-FZ_tol < tyre_data.FZ & tyre_data.FZ <  445+FZ_tol;
idx.FZ_670  =  670-FZ_tol < tyre_data.FZ & tyre_data.FZ <  670+FZ_tol;
idx.FZ_890  =  890-FZ_tol < tyre_data.FZ & tyre_data.FZ <  890+FZ_tol;
idx.FZ_1110 = 1110-FZ_tol < tyre_data.FZ & tyre_data.FZ < 1110+FZ_tol;
FZ_220  = tyre_data( idx.FZ_220, : );
FZ_445  = tyre_data( idx.FZ_445, : );
FZ_670  = tyre_data( idx.FZ_670, : );
FZ_890  = tyre_data( idx.FZ_890, : );
FZ_1110 = tyre_data( idx.FZ_1110, : );
```

Extract point at constant Longitudinal Slip (kappa)

```
% Longitudinal Slip (K) is constant
% 0° , - 3° , -6°
SL_tol   = 0.001;
idx.SL_0 = 0-SL_tol < tyre_data.SL & tyre_data.SL < 0+SL_tol;
SL_0     = tyre_data( idx.SL_0, : );
```

Extract point at constant Longitudinal Slip (alpha)

```
% Lateral Slip (alpha) is constant [not necessary for pure lateral]
% 0° , - 3° , -6°
SA_tol   = 2*to_rad;
idx.SA_0 = 0-SA_tol < tyre_data.SA & tyre_data.SA < 0+SA_tol;
SA_0     = tyre_data( idx.SA_0, : );


figure('Name', 'selected data')
tiledlayout(3,1)

ax_list(1) = nexttile;
plot(tyre_data.IA*to_deg)
hold on
plot(vec_samples(idx.GAMMA_0),GAMMA_0.IA*to_deg,'.');
plot(vec_samples(idx.GAMMA_1),GAMMA_1.IA*to_deg,'.');
plot(vec_samples(idx.GAMMA_2),GAMMA_2.IA*to_deg,'.');
plot(vec_samples(idx.GAMMA_3),GAMMA_3.IA*to_deg,'.');
plot(vec_samples(idx.GAMMA_4),GAMMA_4.IA*to_deg,'.');
title('Camber angle (IA)')
xlabel('Samples [-]')
ylabel('[deg]')

ax_list(2) = nexttile;
plot(tyre_data.FZ)
hold on
plot(vec_samples(idx.FZ_220),FZ_220.FZ,'.');
plot(vec_samples(idx.FZ_445),FZ_445.FZ,'.');
plot(vec_samples(idx.FZ_670),FZ_670.FZ,'.');
plot(vec_samples(idx.FZ_890),FZ_890.FZ,'.');
plot(vec_samples(idx.FZ_1110),FZ_1110.FZ,'.');
title('Vertical force (FZ)')
xlabel('Samples [-]')
ylabel('[N]')


ax_list(3) = nexttile;
plot(tyre_data.SA)
hold on
plot(vec_samples(idx.SA_0),   SA_0.SA,'.');
% plot(vec_samples(idx.SA_3neg),SA_3neg.SA*to_deg,'.');
% plot(vec_samples(idx.SA_6neg),SA_6neg.SA*to_deg,'.');
title('Slide slip (alpha)')
xlabel('Samples [-]')
ylabel('[rad]')
```



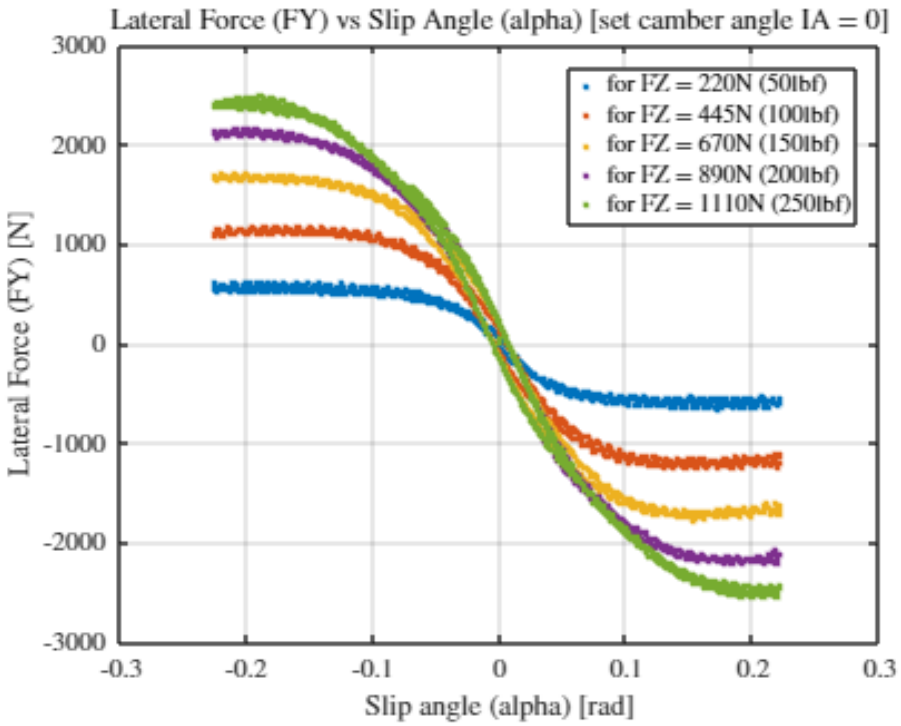

```
% ax_list(4) = nexttile;
% plot(tyre_data.SL*to_deg)
% hold on
% plot(vec_samples(idx.SL_0), SL_0.SL,'.');
% title('Longitudinal slip (K)')
% xlabel('Samples [-]')
% ylabel('[-]')
```

2) Plot "Fy vs alpha" for different Normal Load

Focus on the data with K = 0 and γ = 0, and plot the curves Fy vs for each of the 4 vertical loads Fz used in the experiments. Plot the 4 curves on the same graph, with different colors. Comment on what you see.

```
% Intersect tables to obtain specific sub-datasets
[TData_GAMMA_0_FZ_220, ~]  = intersect_table_data( SL_0, GAMMA_0, FZ_220 );
[TData_GAMMA_0_FZ_445, ~]  = intersect_table_data( SL_0, GAMMA_0, FZ_445 );
[TData_GAMMA_0_FZ_670, ~]  = intersect_table_data( SL_0, GAMMA_0, FZ_670 );
[TData_GAMMA_0_FZ_890, ~]  = intersect_table_data( SL_0, GAMMA_0, FZ_890 );
[TData_GAMMA_0_FZ_1110, ~] = intersect_table_data( SL_0, GAMMA_0, FZ_1110 );


figure('Name', 'Fy vs alpha')
plot(TData_GAMMA_0_FZ_220.SA, TData_GAMMA_0_FZ_220.FY,".");
hold on
plot(TData_GAMMA_0_FZ_445.SA, TData_GAMMA_0_FZ_445.FY,".");
plot(TData_GAMMA_0_FZ_670.SA, TData_GAMMA_0_FZ_670.FY,".");
```

```
plot(TData_GAMMA_0_FZ_890.SA, TData_GAMMA_0_FZ_890.FY,".");
plot(TData_GAMMA_0_FZ_1110.SA, TData_GAMMA_0_FZ_1110.FY,".");
title("Lateral Force (FY) vs Slip Angle (alpha) [set camber angle IA = 0]")
xlabel("Slip angle (alpha) [rad]")
ylabel("Lateral Force (FY) [N]")
legend("for FZ = 220N (50lbf)","for FZ = 445N (100lbf)","for FZ = 670N
(150lbf)","for FZ = 890N (200lbf)","for FZ = 1110N (250lbf)")
grid on
```



From the plot of the selected data is clear that lateral force increase with the lateral slip.

Here is reported a description of the plotted curves:

For small side slip angle the force can be approximated as linear due to the tire elastic behaviour.

But quickly the rubber starts to saturate, and from full adherence condition a part of the contact patch area begin to slip, so the lateral force begin to grow with a non-linear behaviour.

Then near +/- 10 deg, the tire reaches its maxinum lateral grip, representing the point in wich the contact patch is maximally utilized. This point is called peak slip angle, because for greater slip angle the tire begin to lose adherence and begin to generate less force.

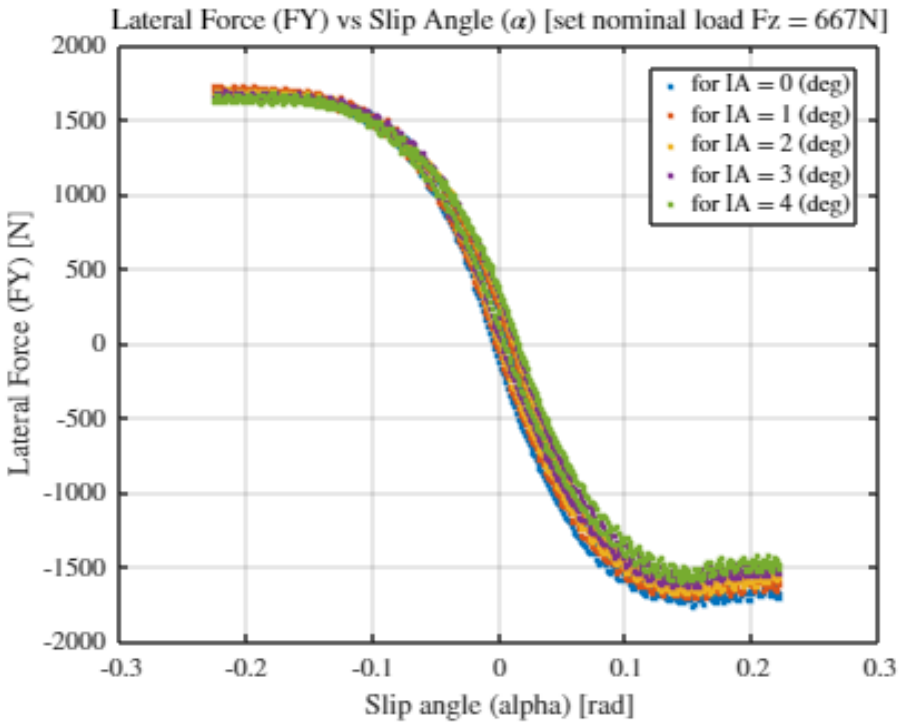
From the plot is clear that as the vertical load (Fz) increases, also the generated lateral force (Fy) increase.

But note this effect is not linear as expected: $F_y = \mu_y F_z$, in fact for growing vertical load, the lateral force grows slightly less than in a linear relationship. This can be due to different causes. Under high vertical load the tire is pressed harder on the road, so it cause the local slip saturation in parts of the contact patch and furthermore this leads to energy loss, less effective force, and a drop in the friction coefficient.

3) Plot "Fy vs alpha" for different Camber Angle

Note: the vertical load value choosen correspond to the nominal load of 667N

```
[TData_GAMMA_0_FZ_670, ~]  = intersect_table_data( FZ_670, GAMMA_0);
[TData_GAMMA_1_FZ_670, ~]  = intersect_table_data( FZ_670, GAMMA_1);
[TData_GAMMA_2_FZ_670, ~]  = intersect_table_data( FZ_670, GAMMA_2);
[TData_GAMMA_3_FZ_670, ~]  = intersect_table_data( FZ_670, GAMMA_3);
[TData_GAMMA_4_FZ_670, ~]  = intersect_table_data( FZ_670, GAMMA_4);

figure('Name', 'Fy vs alpha')
plot(TData_GAMMA_0_FZ_670.SA, TData_GAMMA_0_FZ_670.FY,".");
hold on
plot(TData_GAMMA_1_FZ_670.SA, TData_GAMMA_1_FZ_670.FY,".");
plot(TData_GAMMA_2_FZ_670.SA, TData_GAMMA_2_FZ_670.FY,".");
plot(TData_GAMMA_3_FZ_670.SA, TData_GAMMA_3_FZ_670.FY,".");
plot(TData_GAMMA_4_FZ_670.SA, TData_GAMMA_4_FZ_670.FY,".");
title("Lateral Force (FY) vs Slip Angle ($\alpha$) [set nominal load Fz = 667N]")
xlabel("Slip angle (alpha) [rad]")
ylabel("Lateral Force (FY) [N]")
legend("for IA = 0 (deg)","for IA = 1 (deg)","for IA = 2 (deg)","for IA = 3
(deg)","for IA = 4 (deg)")
grid on
```



Now the lateral force is again plotted in respect of ths side slip angle, but this time the nominal load of 667N is selected, and what is changing is the camber angle.

For changing camber we expect that the tire is able to generate lateral forces even with side slip angle set to 0. Infact what can be observed is a vertical shift od the curve for growing camber inclination.

Negative camber increases peak lateral force in cornering direction, it helps maintaining good grip in aggressive cornering. Furthermore, note also that the curve becomes asymmetric, in fact the tire behaves differently when turning left (positive turn) vs turning right (negative turn).

4) Coefficient fit: pure Lateral Force

Considerations:

since we are fitting the pure lateral magic formula coefficients, the data we are interested in are characterized by zero constant longitudinal slip, the side slip angle is the varying quantity we are interested.

After clarifying that, define the steps to be performed:

1. Fitting the tire parameters in the most simple conditions: the vertical load is set constant and equal to the nominal load, the camber angle is set to zero.
2. Fitting the tire paramaters for different vertical load conditions, with camber angle set to zero.
3. Fitting the tire parameters for the nominal load with varying camber.

Initialise tyre data

```
tyre_coeffs = initialise_tyre_data(R0, Fz0);
```

Fitting with Fz=Fz_nom= 670N, camber=0, alpha = 0, VX= 10

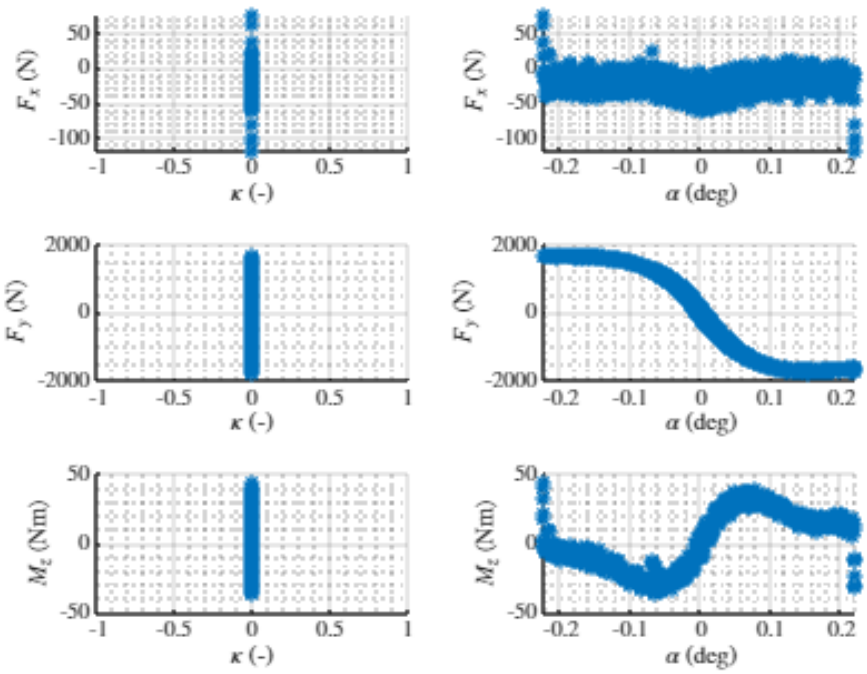
```
% lateral slip is varied
% Fit the coeffs {pCy1 pDy1 pEy1 pHy1 pKy1 pKy2 pVy1}
```

Guess parameters

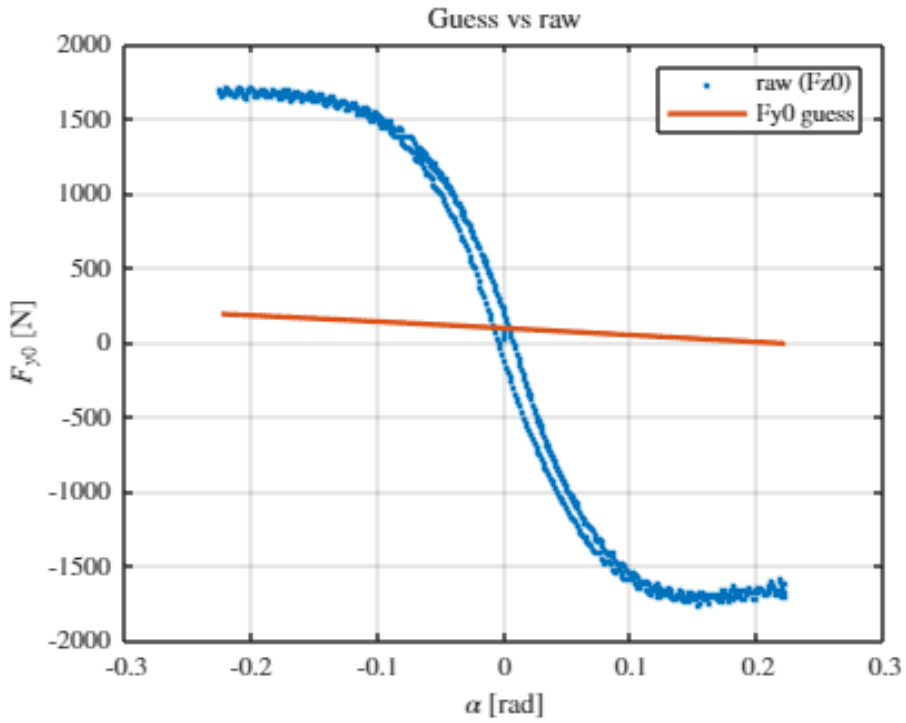
```
% Intersect tables to obtain specific sub-datasets
% data with
% - long slip (SL) = 0
% - camber angle = 0
% - Nominal load = 670 N
[TData0, ~] = intersect_table_data( SL_0, GAMMA_0, FZ_670); % FZ_670
```

Perform the plot of raw data in respect to both slip quantity, in order to check proper selection.

```
figure('Name','data-FZ0')
plot_selected_data(TData0);
```




```
FZ0 = mean(TData0.FZ); % nominal load
zeros_vec = zeros(size(TData0.SA));
ones_vec  = ones(size(TData0.SA));
FZ0_vec   = tyre_coeffs.FZ0*ones_vec; % vector of nominal load

% Vector of data: lateral slip and lateral force for nominal load
alpha_vec = TData0.SA;
FY_vec    = TData0.FY;


% Evalutae the MG for Fx0 with guess data on experimental slip points
FY0_guess = MF96_FY0_vec(zeros_vec,alpha_vec,zeros_vec,FZ0_vec,tyre_coeffs);

% Plot guess data check guess
figure('Name','Guess vs raw (nominal load)')
plot(alpha_vec,FY_vec,'.','Linewidth',2,'DisplayName','raw (Fz0)')
hold on
plot(alpha_vec,FY0_guess,'-','Linewidth',2,'DisplayName','Fy0 guess')
legend
xlabel('$\alpha$ [rad]')
ylabel('$F_{y0}$ [N]')
title("Guess vs raw")
grid on
```



4.1) Fitting for Nominal Load FZ = 670 N

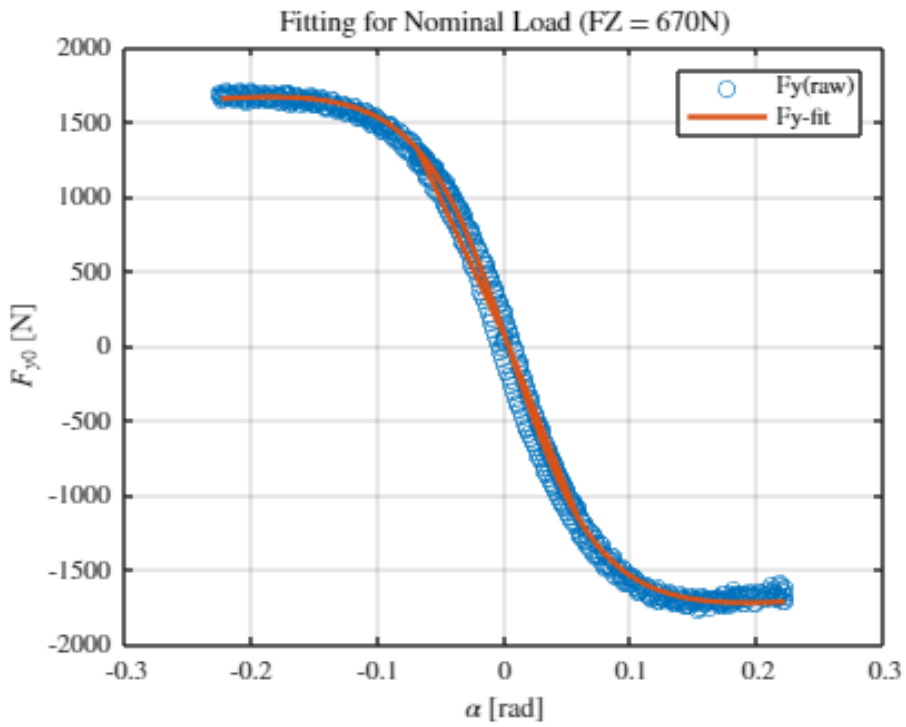
Fitting the tire parameters in the most simple conditions: the vertical load is set constant and equal to the nominal load, the camber angle is set to zero.

```
% Guess values for parameters to be optimised
%    [pCy1 pDy1 pEy1 pHy1 pKy1 pKy2 pVy1]
P0 = [  -1,   1,   0,   0,   0,    1,  0];

% NOTE: many local minima => limits on parameters are fundamentals
%    [pCy1 pDy1 pEy1 pHy1 pKy1 pKy2 pVy1]
ub = [];
lb = [];

[P_fz_nom,fval,exitflag] = fmincon(@(P)resid_pure_Fy(P,FY_vec, alpha_vec,0,FZ0,
tyre_coeffs),P0,[],[],[],[],lb,ub);
```

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
% Update tyre data with new optimal values
tyre_coeffs.pCy1 = P_fz_nom(1) ; % 1
tyre_coeffs.pDy1 = P_fz_nom(2) ;
```

```
tyre_coeffs.pEy1 = P_fz_nom(3) ;
tyre_coeffs.pHy1 = P_fz_nom(4) ;
tyre_coeffs.pKy1 = P_fz_nom(5) ;
tyre_coeffs.pKy2 = P_fz_nom(6) ;
tyre_coeffs.pVy1 = P_fz_nom(7) ;

% Update Fy0 coefficients for the experimental value of lateral slips
FY0_fz_nom_vec = MF96_FY0_vec(zeros_vec,alpha_vec,zeros_vec,FZ0_vec,tyre_coeffs);

figure('Name','Fy0(Fz0)')
plot(alpha_vec,TData0.FY,'o','DisplayName','Fy(raw)')
hold on
%plot(TDataSub.KAPPA,FX0_fz_nom_vec,'-')
plot(alpha_vec,FY0_fz_nom_vec,'-','LineWidth',2,'DisplayName','Fy-fit')
xlabel('$\alpha$ [rad]')
ylabel('$F_{y0}$ [N]')
legend
title("Fitting for Nominal Load (FZ = 670N)")
grid on
```



4.2) Fitting for Variable Load

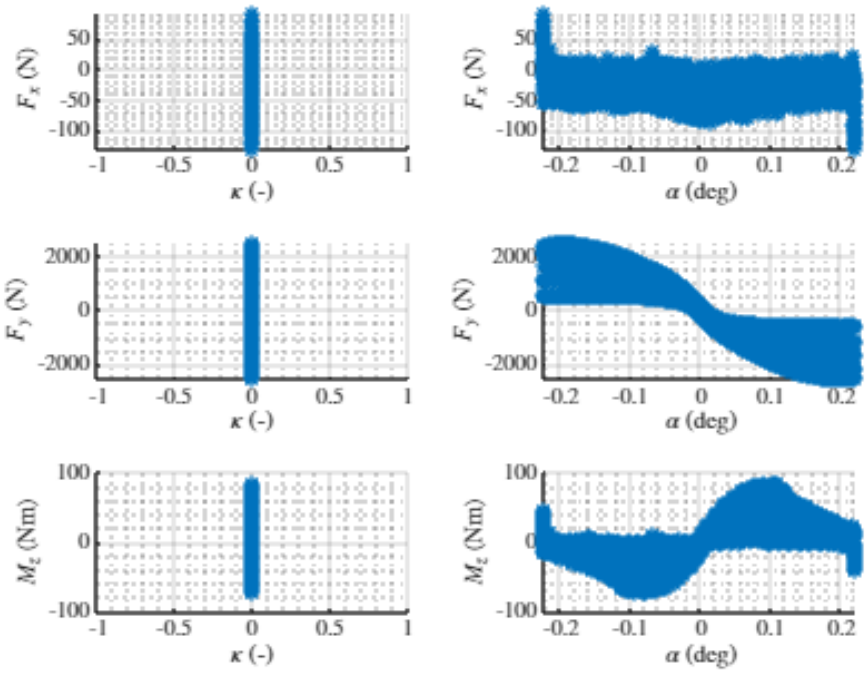
Fitting the tire paramaters for different vertical load conditions, with camber angle set to zero.

Extract data with variable load:

```
[TDataDFz, ~] = intersect_table_data( SL_0, GAMMA_0 );
```

Plot selected data, in order to check proper selection:

```
figure('Name','data-FZ var.')
plot_selected_data(TDataDFz);
```




```
FZ0 = mean(TData0.FZ);

% Vector of zeros and ones
zeros_vec = zeros(size(TDataDFz.SL));
ones_vec  = ones(size(TDataDFz.SL));
FZ0_vec   = tyre_coeffs.FZ0*ones_vec; % vector of nominal load

% Vector of data: lateral slip and lateral force for nominal load
alpha_vec = TDataDFz.SA;
FY_vec    = TDataDFz.FY;
FZ_vec    = TDataDFz.FZ;

% Guess values for parameters to be optimised
%    [pDy2 pEy2 pEy3 pHy2 pVy2]
P0 = [   0,   0,   0,   0,   0];

% NOTE: many local minima => limits on parameters are fundamentals
%    [pDy2 pEy2 pEy3 pHy2 pVy2]
lb = [];
ub = [];
```

```
% LSM_pure_Fx returns the residual, so minimize the residual varying X. It
% is an unconstrained minimization problem

[P_dfz,fval,exitflag] = fmincon(@(P)resid_pure_Fy_varFz(P,FY_vec,
alpha_vec,0,FZ_vec, tyre_coeffs),P0,[],[],[],[],lb,ub);
```

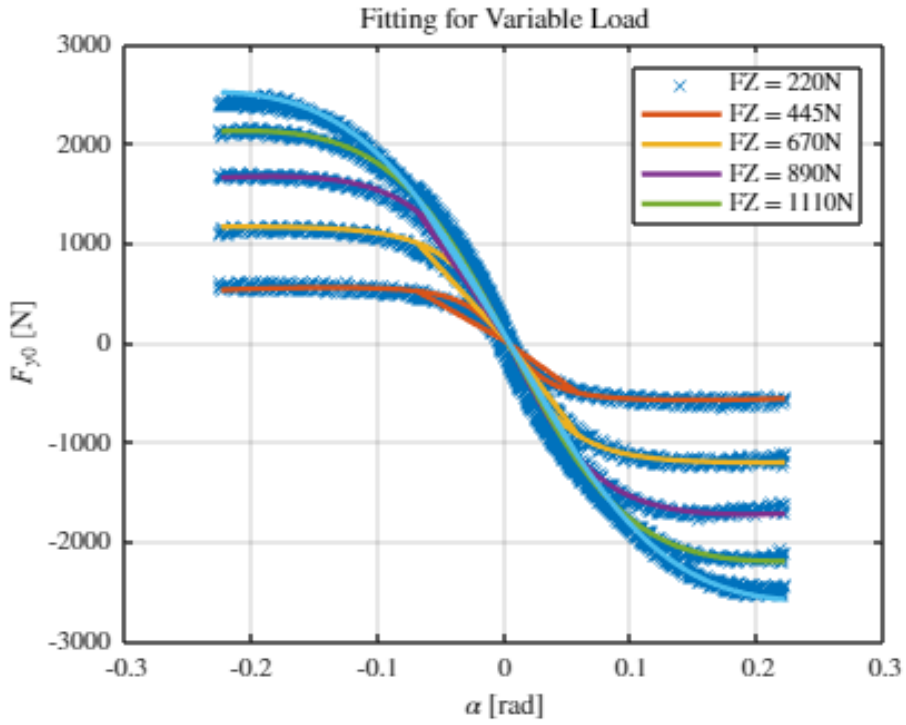
```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
disp(exitflag)
```

```
     1
```

```
% Change tyre data with new optimal values
tyre_coeffs.pDy2 = P_dfz(1); % 1
tyre_coeffs.pEy2 = P_dfz(2);
tyre_coeffs.pEy3 = P_dfz(3);
tyre_coeffs.pHy2 = P_dfz(4);
tyre_coeffs.pVy2 = P_dfz(5);

res_FY0_dfz_vec = resid_pure_Fy_varFz(P_dfz,FY_vec,alpha_vec,0 ,
FZ_vec,tyre_coeffs);

% Compute Fx for the four load conditions with fitted parameters
FY0_fz_var_vec1 = MF96_FY0_vec(zeros_vec, alpha_vec ,ones_vec,
mean(FZ_220.FZ)*ones_vec,tyre_coeffs);
FY0_fz_var_vec2 = MF96_FY0_vec(zeros_vec, alpha_vec ,ones_vec,
mean(FZ_445.FZ)*ones_vec,tyre_coeffs);
FY0_fz_var_vec3 = MF96_FY0_vec(zeros_vec, alpha_vec ,ones_vec,
mean(FZ_670.FZ)*ones_vec,tyre_coeffs);
FY0_fz_var_vec4 = MF96_FY0_vec(zeros_vec, alpha_vec ,ones_vec,
mean(FZ_890.FZ)*ones_vec,tyre_coeffs);
FY0_fz_var_vec5 = MF96_FY0_vec(zeros_vec, alpha_vec ,ones_vec,
mean(FZ_1110.FZ)*ones_vec,tyre_coeffs);

figure('Name','Fy0-var Fz')
plot(TDataDFz.SA,TDataDFz.FY,'x','DisplayName','Fy(raw)')
hold on
%plot(TDataSub.KAPPA,FX0_fz_nom_vec,'-')
%plot(SL_vec,FX0_dfz_vec,'-','LineWidth',2)
plot(alpha_vec,FY0_fz_var_vec1,'-','LineWidth',2)
plot(alpha_vec,FY0_fz_var_vec2,'-','LineWidth',2)
plot(alpha_vec,FY0_fz_var_vec3,'-','LineWidth',2)
plot(alpha_vec,FY0_fz_var_vec4,'-','LineWidth',2)
plot(alpha_vec,FY0_fz_var_vec5,'-','LineWidth',2)
xlabel('$\alpha$ [rad]')
```

```
ylabel('$F_{y0}$ [N]')
title("Fitting for Variable Load")
legend("FZ = 220N","FZ = 445N","FZ = 670N","FZ = 890N","FZ = 1110N")
grid on
```



4.3) Fitting for Variable Camber

Fitting the tire parameters for the nominal load with varying camber.

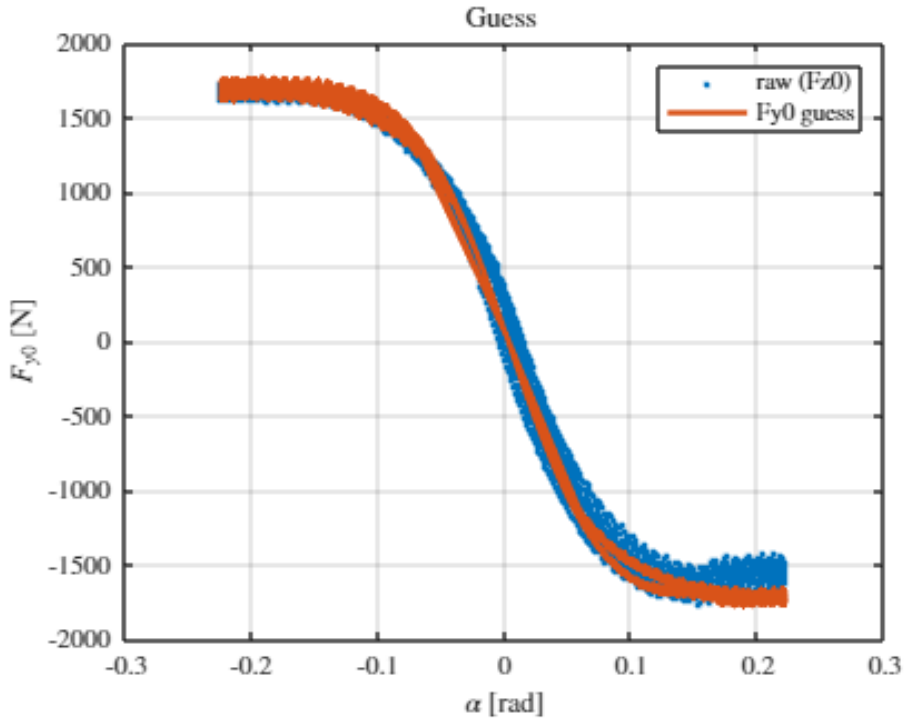
Extract data with variable load:

```
[TDataGamma, ~] = intersect_table_data( SL_0, FZ_670 );
```

Guess values for parameters to be optimised

```
%    [pDy3 pEy4 pHy3 pKy3 pVy3 pVy4]
P0 = [   0,   0,   0,   0,   0,   0];
lb = [];
ub = [];

zeros_vec = zeros(size(TDataGamma.SA));
ones_vec  = ones(size(TDataGamma.SA));

ALPHA_vec = TDataGamma.SA;
GAMMA_vec = TDataGamma.IA;
FY_vec    = TDataGamma.FY;
FZ_vec    = TDataGamma.FZ;
```

```
FY0_guess = MF96_FY0_vec(zeros_vec, ALPHA_vec, GAMMA_vec, FZ_vec, tyre_coeffs);

% Plot guess data check guess
figure('Name','Guess vs raw')
plot(ALPHA_vec,FY_vec,'.','Linewidth',2,'DisplayName','raw (Fz0)')
hold on
plot(ALPHA_vec,FY0_guess,'-','Linewidth',2,'DisplayName','Fy0 guess')
legend
xlabel('$\alpha$ [rad]')
ylabel('$F_{y0}$ [N]')
title("Guess ")
grid on
```

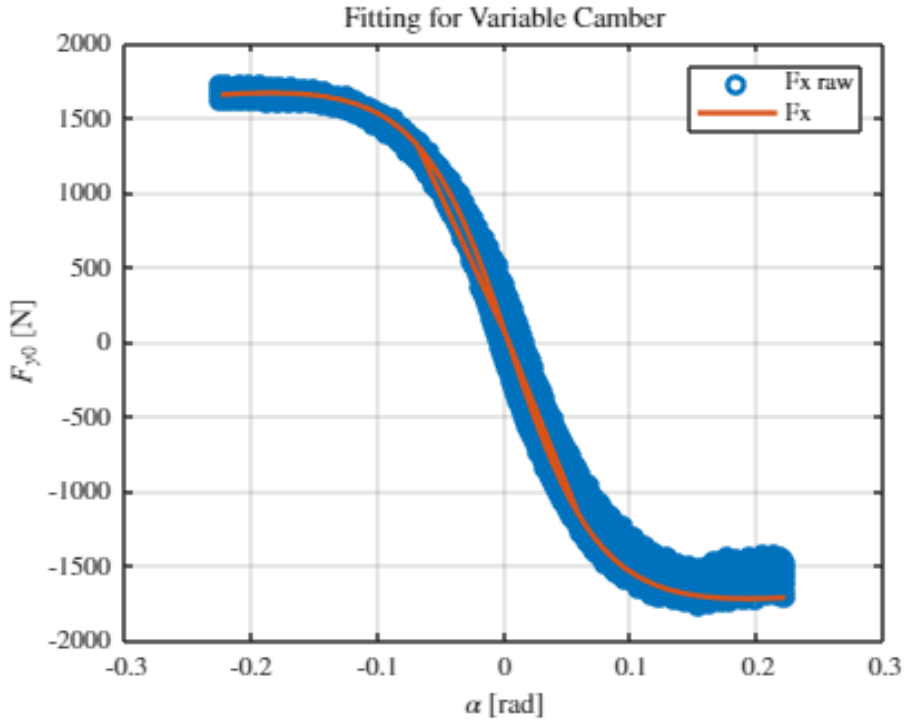



```
[P_varGamma,fval,exitflag] =
fmincon(@(P)resid_pure_Fy_varGamma(P,FY_vec,ALPHA_vec,GAMMA_vec,tyre_coeffs.FZ0,
tyre_coeffs),P0,[],[],[],[],lb,ub);
```

```
Initial point is a local minimum that satisfies the constraints.

Optimization completed because at the initial point, the objective function is non-decreasing
in feasible directions to within the value of the optimality tolerance, and
constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
% Change tyre data with new optimal values
tyre_coeffs.pDy3 = P_varGamma(1); % 1
tyre_coeffs.pEy4 = P_varGamma(2);
tyre_coeffs.pHy3 = P_varGamma(3);
```

```
tyre_coeffs.pKy3 = P_varGamma(4);
tyre_coeffs.pVy3 = P_varGamma(5);
tyre_coeffs.pVy4 = P_varGamma(6);


FY0_varGamma_vec = MF96_FY0_vec(zeros_vec, ALPHA_vec, GAMMA_vec,
tyre_coeffs.FZ0*ones_vec,tyre_coeffs);

figure('Name','Fy0 vs Gamma')
plot(ALPHA_vec,TDataGamma.FY,'o','Linewidth',2,'DisplayName','Fx raw')
hold on
plot(ALPHA_vec,FY0_varGamma_vec,'-','Linewidth',2,'DisplayName','Fx')
xlabel('$\alpha$ [rad]')
ylabel('$F_{y0}$ [N]')
legend
title("Fitting for Variable Camber")
grid on
```




```
% Calculate the residuals with the optimal solution found above
res_Fy0_varGamma  = resid_pure_Fy_varGamma(P_varGamma,FY_vec,
ALPHA_vec,GAMMA_vec,tyre_coeffs.FZ0, tyre_coeffs);


fprintf('R-squared = %6.3f\n',1-res_Fy0_varGamma);
```

```
R-squared = -0.000
```

```
[alpha__y, By, Cy, Dy, Ey, SVy] = MF96_FY0_coeffs(0, 0, GAMMA_vec(3),
tyre_coeffs.FZ0, tyre_coeffs);
%
fprintf('By      = %6.3f\n',By);
```

```
By      =  9.228
```

```
fprintf('Cy      = %6.3f\n',Cy);
```

```
Cy      = -1.598
```

```
fprintf('muy     = %6.3f\n',Dy/tyre_coeffs.FZ0);
```

```
muy     =  2.529
```

```
fprintf('Ey      = %6.3f\n',Ey);
```

```
Ey      =  0.343
```

```
fprintf('SVy     = %6.3f\n',SVy);
```

```
SVy     = -29.126
```

```
fprintf('alpha_y = %6.3f\n',alpha__y);
```

```
alpha_y = -0.004
```

```
fprintf('Ky      = %6.3f\n',By*Cy*Dy/tyre_coeffs.FZ0);
```

```
Ky      = -37.298
```

```

```

The fitting of the lateral force (Fy) versus side slip angle (α), using the Magic Formula 1996 under vertical load and camber angle changes, provide a curve that closely matches the experimental data in the linear region and captures the nonlinear saturation behavior around the peak slip angle.

The estimated parameters [especially pCy1 (shape factor), pDy1 (peak factor), and pKy1/pKy2 (stiffness)] describe how the tire builds up lateral force with increasing side slip.

A good fit confirms that:

- The tire exhibits a clear saturation point beyond ~10° of slip, where lateral force starts to drop (as expected in real tires).
- The Magic Formula is capable of modeling both the linear cornering stiffness and the nonlinear peak behaviour effectively.
- The residual error is acceptably low, indicating the model is valid for this operating condition.

The fitting has been properly concluded.

This confirms that the Magic Formula, is a robust tool for simulating tire behavior and is suitable for vehicle dynamics simulations and control design, even for Formula SAE tires.

[Fx fitting]

In this section it is simply loaded the structure that have memorized the parameters of a parallel fitting of the tire data regarding the pure longitudinal force. The struct appears in the workspace with the name "tyre_coeffs_x". We need those parameters in order to plot a complete Adherence ellipse.

```
load('tyre_1_x_B1464run30.mat');

% Inputs
Fz = Fz0;                    % nominal load
% muy = Dy/tyre_coeffs.FZ0;   % from fit

%
% % R-squared =  0.994
% Bx      = 17.014;
% Cx      =  1.539;
% mux      =  3.147;
% Ex      =  0.012;
% SVx     = -18.853;
% kappa_x = -0.000;
% Kx      = 82.413;
```

5) Adherence's Ellipse

We are going to use the brush model theoretical approach.

based on the theoretical slip normalised with the slip modulus.

Define 12 values of lateral slip α and other 12 for longitudinal slip κ

```
alpha_comb = linspace(-0.15, 0.15, 12); % [rad]
kappa_comb = linspace(-0.12, 0.12, 12); % [-]
```

Matrix preallocation to store sweep values of lateral slip and longitudinal slip

```
alpha_sweep = zeros(500, length(alpha_comb));
kappa_sweep = zeros(500, length(alpha_comb));
```

Matrix preallocation to store longitudinal (x) and lateral (y) and their norm, for both approaches

- *sweep kappa, fixed alpha*

```
theor_slip_x_1 = zeros(500, length(alpha_comb));
theor_slip_y_1 = zeros(500, length(alpha_comb));
theor_slip_norm_1 = zeros(500, length(alpha_comb));
```

- *sweep alpha, fixed kappa*

```
theor_slip_x_2 = zeros(500, length(alpha_comb));
theor_slip_y_2 = zeros(500, length(alpha_comb));
```

```
theor_slip_norm_2 = zeros(500, length(alpha_comb));
```

Matrix preallocation to store normalized longitudinal (FX/FZ) and lateral (FY/FZ) forces from Magic Formula

```
FX_comb_1 = zeros(500, length(alpha_comb)); % sweep kappa, fixed alpha
FY_comb_1 = zeros(500, length(alpha_comb));

FX_comb_2 = zeros(500, length(alpha_comb)); % sweep alpha, fixed kappa
FY_comb_2 = zeros(500, length(alpha_comb));
```

vectors used for vectorized computation in Magic Formula calls

```
zeros_vec = zeros(500, 1);
ones_vec = ones(500, 1);
```

Loop through each α (and corresponding κ) value

Here are reported the main formulas:

- Theoretical Slip Components: (they are the same for longitudinal and lateral slip)

$$\sigma_x = \frac{\kappa}{1+\kappa}$$

$$\sigma_y = \frac{\tan(\alpha)}{1+\kappa}$$

$$\|\sigma\| = \sqrt{\sigma_x^2 + \sigma_y^2}$$

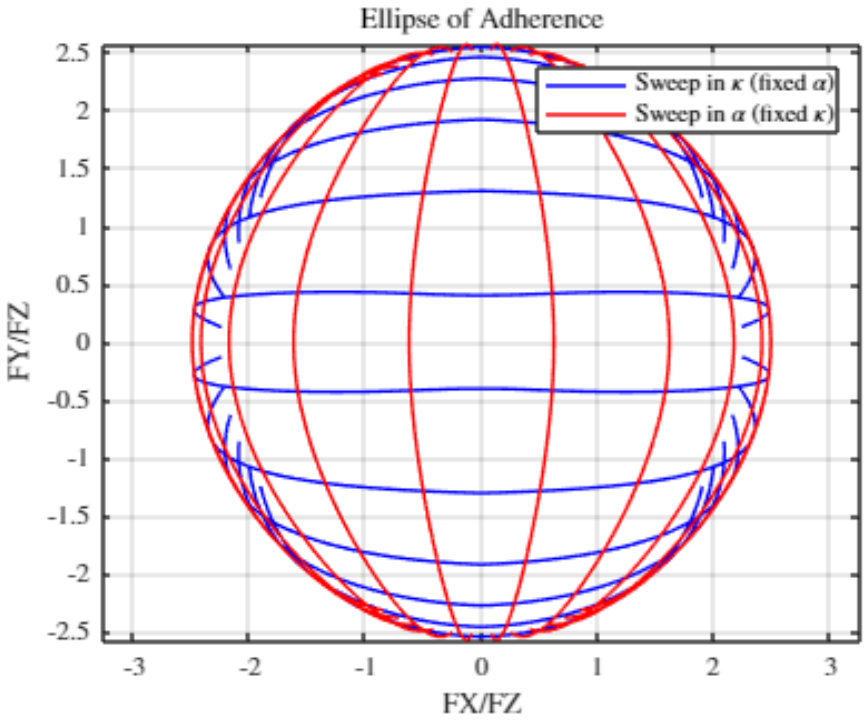
- Magic Formula-Based Force Projections:

$$F_x = \frac{\sigma_x}{\|\sigma\|} \text{MF\_96}(B_0, C_0, D_0, E_0, \|\sigma\|) \frac{1}{F_{z0}}$$

$$F_y = \frac{\sigma_y}{\|\sigma\|} \text{MF\_96}(B_0, C_0, D_0, E_0, \|\sigma\|) \frac{1}{F_{z0}}$$

```
for i = 1:length(alpha_comb)

    % Define a sweep range for both variables: from -0.25 to 0.25.
    alpha_sweep(:, i) = linspace(-0.27, 0.27 ,500);
    kappa_sweep(:, i) = linspace(-0.27, 0.27 ,500);

    % Compute normalized slip vector using brush model formula (like above formula)
    theor_slip_x_1(:, i) = kappa_sweep(:, i) ./ (1+abs(kappa_sweep(:,i)));
    theor_slip_y_1(:, i) = tan(alpha_comb(i)) ./ (1+abs(kappa_sweep(:,i)));
```

```
    theor_slip_norm_1(:, i) = sqrt(theor_slip_x_1(:, i).^2 + theor_slip_y_1(:,
i).^2);

    theor_slip_x_2(:, i) = (kappa_comb(i) ./ (1+abs(kappa_comb(i))) .* ones_vec );
    theor_slip_y_2(:, i) = tan(alpha_sweep(:, i)) ./ (1+abs(kappa_comb(i)));
    theor_slip_norm_2(:, i) = sqrt(theor_slip_x_2(:, i).^2 + theor_slip_y_2(:,
i).^2);

    % sweep kappa, fixed alpha
    % The Magic Formula returns total force magnitude depenfing on the slip norm
    tmp = MF96_FY0_vec(zeros_vec, theor_slip_norm_1(:, i), zeros_vec,
FZ0.*ones_vec, tyre_coeffs);
    % project the total force on the x/y component premultiplying for sigma_y/
sigma_norm
    % thus it is normalized by the nominal load
    FY_comb_1(:, i) = (theor_slip_y_1(: ,i) ./theor_slip_norm_1(:, i)) .* (1/FZ0).*
tmp;

    tmp = MF96_FX0_vec(theor_slip_norm_1(:, i),zeros_vec , zeros_vec,
FZ0.*ones_vec, tyre_coeffs_x);
    FX_comb_1(:, i) = (theor_slip_x_1(: ,i) ./theor_slip_norm_1(:, i)) .* (1/FZ0).*
tmp;

    % sweep alpha, fixed kappa
    tmp = MF96_FY0_vec(zeros_vec, theor_slip_norm_2(:, i), zeros_vec,
FZ0.*ones_vec, tyre_coeffs);
    FY_comb_2(:, i) = (theor_slip_y_2(: ,i) ./theor_slip_norm_2(:, i)) .* (1/FZ0).*
tmp;

    tmp = MF96_FX0_vec(theor_slip_norm_2(:, i),zeros_vec , zeros_vec,
FZ0.*ones_vec, tyre_coeffs_x);
    FX_comb_2(:, i) = (theor_slip_x_2(: ,i) ./theor_slip_norm_2(:, i)) .* (1/FZ0).*
tmp;
end

figure()
plot(FX_comb_1, FY_comb_1, "b-")
hold on
plot(FX_comb_2, FY_comb_2, "r-")
ylabel("FY/FZ")
xlabel("FX/FZ")
title("Ellipse of Adherence")
legend('Sweep in $\kappa$ (fixed $\alpha$)','','','','','','','','','','','',
'Sweep in $\alpha$ (fixed $\kappa$)')
grid on
axis equal
```



So the blue curves represent some values of sweep in longitudinal slip κ, for **fixed values of** lateral slip angle α

So the red curves represent some values of sweep in lateral slip angle α, for **fixed values of** longitudinal slip κ

Each curve represents combinations of normalized tire forces FX/FZ and FY/FZ, using the **Magic Formula** with the **brush model-based slip inputs.**

The plot **resembles an ellipse**, hence the name **"Ellipse of Adherence"** — this is typical of combined-slip behavior under the assumption of limited friction (tire saturation).

Note the curves flatten at edges, this means that tire force no longer increases with more slip, the more we get close to the edge, the less additional force is generated, this means the closer we are to the edge, the more non-linear behaviour characterises the tire.

The **outer envelope** (edge of the red and blue curves) represents the **maximum combined force** the tire can produce (i.e., when the slip magnitude σ reaches its peak).

The ellipse is symmetric about both axes, this type of ellipse is expected for a well-tuned tire model. Furthermore the shape of the edge tend to be circular, suggesting the model has isotropic friction i.e. same grip in both directions.