Appendix 2

TreadSim

A Matlab program for the calculation of steady-state force and moment response to side slip, longitudinal slip, camber and turn slip using a brush model with up to three rows of tread elements and a flexible carcass. Cf. Chapter 3, Section 3.3 and Figs.3.35,3.38.

```
%   **  TreadSim.m **

%   **  tread element following model   **
%   **  one, two or three rows   **

%   **  stationary response to lateral, longitudinal, camber and turn slip   **
%   **  carcass rigid with only lateral compliance effect in Mz   **
%   **  or
%   **  carcass lateral, yaw and bending compliance   **
%   **  with approximate slip speed dependent friction coefficient   **
%   **  conicity, ply-steer, correction factors for lateral roll and camber distortion   **
%   **  calculation of forces Fx, Fy and moment Mz (and t)
%       as function of kappa, alpha or a/R
%      (with kappa, a/R =aphit, gamma or alpha as changing motion parameter)
%   **  option for making deformation picture for one set of motion variables**

clf;
picture=1;  % if =1 then single set of values of motion variables is given and diagrams of
            % force distribution and of carcass and tread elements deformation are produced
rigid=0;        % =1 if carcass is rigid, else =0
if rigid ==1
iitend=1;       % =1 (no iterations at rigid carcass!)
else
iitend=5;            % =5....25 (number of iterations depends on carcass compliance)
end;
n=20;                       % number of elements -1
nrow=3;                     % number of rows (1 or 2 or 3)
a=0.1;                      % half contact length [m]

if picture ==1              % single set of values of motion variables
  alphapict= 4;             % slip angle in degrees
  gammapict= 0;             % camber angle in degrees
  Rpict= 1.00000000;        % turnslip = a/Rpict,  path curvature radius Rpict in m
  kappapict= -0.0;          % long. slip (from -1 to ....)
  n=30;
end;
                     % if running variable (abscissa) (then =1 else =0)
alpharuns=0;                % slip angle
```

```
kapparuns=1;                % longitudinal slip ratio
aphitruns=0;                % non-dimensional turnslip (aphit=a/R)

        % if not chosen as running variable but as changing parameter (then =1, else =0)
gammapar=0;
aphitpar=0;
alphapar=1;
kappapar=0;

    % side slip angle: alpha range and step [deg] (if chosen as running variable)
alphdegminr=-25.01;
alphdegmaxr=25;
g1a=0.05; g2a=0.2;      % deltalphdegr=g1a*abs(alphadeg)+g2a (varying alpha increment)
    % longidinal slip ratio: kappa range and step [-] (if chosen as running variable)
kappaminr=-0.99;        % negative: braking
kappamaxr= 0.8;         % positive: driving
g1k=0.1; g2k=0.01;% deltakappar=g1k*abs(kappa)+g2k  (varying kappa increment)
    % non-dim. turnslip: aphit (=a/R) range and step [-] (if chosen as running parameter)
aphitminr=0.01;
aphitmaxr=0.7;
g1t=0.01; g2t=0.01;     % deltaphitr=g1t*abs(aphit)+g2t   (varying aphit increment)

                    % constant motion parameters
    % (overruled if chosen as changed parameter or as running variable)
alphadeg=0;         % [deg]     (slip angle)
gammadeg=0;             % [deg]     (camber angle)
R=10000000;             % [m]       (aphit=a/R;   non-dimensional turnslip)
kappa=0;            % [-]       (long. slip ratio)

aphit=a/R;
jmax=1;
deltgamdeg=0; gamdegmin=gammadeg;
deltalphdeg=0; alphdegmin=alphadeg;
deltaphit=0; aphitmin=aphit;
deltakappa=0; kappamin=kappa;

    % changed motion parameters (number, minimum, step)

if gammapar==1          % **gamma** as changed motion parameter [deg]:
jmax=6; gamdegmin=-5; deltgamdeg=5;
end;
if aphitpar==1          % **aphit (=a/R)** as changed motion parameter [-]:
jmax=12; aphitmin=-0.3; deltaphit=0.1;
end;
if alphapar==1          % **alpha** as changed motion parameter [deg]:
jmax=5; alphdegmin=0.000001-2; deltalphdeg=2;
end;
if kappapar==1          % **kappa** as changed motion parameter [-]:
jmax=1; kappamin=-0.0; deltakappa=-0.05;
```

```
end;

% system parameters

re=0.3;                  % effective rolling radius [m]
Fz=3000;                 % normal load [N]
Vc=30;                   % speed of contact centre [m/s]
mu0=1;                   % friction coefficient [-]
amu=0.03;                % speed dependency coefficient for mu [s/m]
                         % ( mu=mu0/(1+amu*Vbi) ); 0.03
CFkappaFz=15;            % long. slip stiffness over Fz [-]
CFkappa=CFkappaFz*Fz;   %[N/-]   from this: cp=(1/(nrow*2*a*a))*CFkappa;
%  cornering stiffness coeff. CFalpha/Fz [-] is calculated if alpharuns=1 for last param. case
%  a (half contact length) defined above
b=0.08;                  % half contact width [m]
brow=0.05;               % half effective contact width [m]
 if nrow==1
 brow=0;
end;
alphadegply=0;           % ply-steer equivalent slip angle [deg]
gammadegcon=0;           % conicity equivalent camber angle [deg]

    %stiffnesses of tyre carcass at contact patch at base of tread elements
     % (as if measured on bare tyre)
y0=0.00;        % initial lateral off-set (vo) , if rigid: Mz=Mz-c*Fx*Fy-y0*Fx  (0.005)

if rigid==0
 cyaw= 6000.000;  % [Nm/rad]   8000
 cbend=4000.000;  % [mN]       8000
 clat= 100000;            % [N/m]      suggested: clat=Fz/(0.15*a)   200000
 else
 cyaw= 10000000;          % [Nm/rad]
 cbend=10000000;          % [mN]
 clat= 10000000;          % [N/m]
 c=1/100000;        % c=(1-epsyFy)/clat ,
                          % Mz=Mz-c*Fx*Fy-y0*Fx, (0.5/200000),   c is used if 'rigid'
end;
    %resulting stiffness of tread elements, per row, per unit of circumfernce
cp=(1/(nrow*2*a*a))*CFkappa;
theta=(1/3)*CFkappa/(mu0*Fz); % not used

    % correction parameters
epsyprime=0.0;  % reduction factor for moment arm of Fx causing yaw distortion (slope);
          % arm =~(1-epsyprime)*ymeff   (0.5)
          % where ymeff represents effective lateral displacement of belt in contact zone
epsyrgamma=4;  % moment (Mz) arm for Fx due to sideways rolling caused by camber =
         %  ygamroll=epsyrgamma*gamma*b; if abs(ygamroll)>b: ygamroll=b*sign(gamma)
epsyFy=0.0; % reduction factor moment(Mz)arm for Fx due to sideways rolling caused by Fy
          % and the counter effect of longitudinal deflection
```

```
        % total moment {Mz) arm= ymeff= ym0+ydefl(1-epsyFy)+ygamroll
epsdrgam=0.0;   % coefficient for reduced change eff. rolling radius left/right
                % caused by camber
        % delta_re_Right=~ -(1-epsdrgam)*gamma*brow; suggestion: epsdrgam          %
(=epsxgamma)=epsgamma
epsgamma=0.0;  % reduced camber curvature coefficient (will be >0) due to distorsion
    % (=epsygamma)
        % resulting camber contactline curvature (at mu=0)
        % 1/Rgamma=(1/re)*(1-epsgamma)*sin(gamma)

      % end of parameter settings

      % start of computation
a2=a*a; a3=a2*a;
nCFa=0;  CFalphaFz='?';

if picture==1
  jmax=1;
  alphadegmin=alphapict;
  gamdegmin=gammapict;
  aphitmin=a/Rpict;
  kappamin=kappapict;
end;

      % **  begin gamma, aphit(=a/R), alpha or kappa-loop (as motion parameter) **
for j=1:jmax
gammadeg=gamdegmin+(j-1)*deltgamdeg;
aphit=aphitmin+(j-1)*deltaphit;
alphadeg=alphdegmin+(j-1)*deltalphdeg;
kappa=kappamin+(j-1)*deltakappa;

alpha=alphadeg*pi/180;
gamma=(gammadeg+gammadegcon)*pi/180;
singam=sin(gamma);
cs0=alphadegply*pi/180;            % initial belt slope due to ply-steer
cc0=(1/re)*(1-epsgamma)*singam;% initial belt lateral curvature due to camber and conicity
ym0=-0.5*cc0*a2;        % initial belt lateral position at contact centre resulting from camber
                        % at mu=0. This is a guess; then, y0i=cs0*a at leading edge (x=a)
ygamroll=epsyrgamma*singam*b;  % moment arm of Fx due to sideways rolling at camber
 if abs(ygamroll)>b
 ygamroll=b*sign(gamma);
 end;

cs=cs0;
cc=cc0;
ym=ym0;
Fy=0; Mzprime=0; Fym1=0; Mzm1=0;
if alpharuns==1
alphadeg=alphdegminr;
```

```
imax=round((log(abs(1-g1a*alphdegminr/g2a))+log(abs(1+g1a*alphdegmaxr/g2a)))/g1a)+2;
end;
if kapparuns==1
kappa=kappaminr;
imax=round((log(abs(1-g1k*kappaminr/g2k))+log(abs(1+g1k*kappamaxr/g2k)))/g1k)+2;
end;
if aphitruns==1
aphit=aphitminr;
imax=round((log(abs(1-g1t*aphitminr/g2t))+log(abs(1+g1t*aphitmaxr/g2t)))/g1t)+5;
end;

if picture==1
  alphadeg=alphapict;
  alpha=alphadeg*pi/180;
  kappa=kappapict;
  aphit=a/Rpict;
  alphadegminr=alphapict;
  kappaminr=kappapict;
  aphitminr=a/Rpict;
  imax=1;
end;

for ia=1:imax       % ** begin kappa or alpha or a/R (if running variable) ia-loop **

if alpharuns==1
alpha=pi*alphadeg/180;
Fym1=Fy;                 % needed for calculation of cornering stiffness
end;
R=a/(aphit+0.0001);
Fy=0; Mzprime=0;         % initial value for iteration

                    % ** begin carcass deflection iteration, iit-loop **
for iit=1:iitend
slope=Mzprime/cyaw;
cs=cs0+slope;            % belt slope at contact centre w.r.t. wheel plane
curve=Fy/cbend;
cc=cc0-curve;            % belt lateral curvature in contact zone
ydefl= Fy/clat;
ym=ym0+ydefl;     % belt lateral position at contact centre w.r.t. wheel plane
yFyroll=-epsyFy*ydefl;
ymeff=ym+ygamroll+yFyroll;   %effective moment (Mz) arm for Fx

for iLR=1:nrow   % ** begin left (iLR=1) and right row (iLR=2) loop (if one row: nrow=1)
  if nrow==1
    ibLR=0; bLR=0;
  end;
  if nrow==2
    ibLR=(2*iLR-3); bLR=ibLR*brow;
  end;
```

```
  if nrow==3
    ibLR=(iLR-2); bLR=ibLR*brow;  % left: bLR=-brow,  mid: bLR=0,  right: bLR=+brow
  end;

 psidot= -Vc/R;

 Vcx=Vc*cos(alpha);
 Vsx= -Vcx*kappa;
 Vsy= -Vcx*tan(alpha);
 Vr=Vcx-Vsx;
 omega=Vr/re;
 VcxLR=Vcx-bLR*psidot;
 VsxLR=Vsx-bLR*psidot+bLR*omega*(1-epsdrgam)*singam;
 VsyLR=Vsy;
 VrLR=Vr;


deltat=2*a/(n*VrLR);
deltax=2*a/n;
xi=a; x(1)=a;
ei=0; exi=0; eyi=0; Fxim1=0; Fyim1=0; Mzim1=0;
px(1)=0; py(1)=0; pz(1)=0; p(1)=0;
sliding=0;
                     % ** begin i-loop: passage through contact length
for  i=2:n+1
eim1=ei; exim1=exi; eyim1=eyi;
xi=xi-deltax; xi2=xi*xi;
x(i)=xi;
dybdximean=cs+cc*(xi+0.5*deltax);
ybi=ym +cs*xi+0.5*cc*xi*xi+bLR;
ybieff=ymeff+cs*xi+0.5*cc*xi*xi+bLR;
Vbxi=VsxLR;
Vbyimean=VsyLR + (xi+0.5*deltax)*psidot - VrLR*dybdximean;
Vbi=sqrt(Vbxi*Vbxi+Vbyimean*Vbyimean); Vb(i)=Vbi;     % belt point velocity
mu=mu0/(1+amu*Vbi);
pzi=0.75*Fz*(a2-xi2)/(a3*nrow); pz(i)=pzi;     % vertical pressure distribution (qz)
deltasxi=deltat*Vbxi;
deltasyi=deltat*Vbyimean;

if sliding>0              % element is sliding
ei=mu*pzi/cp;
 if i==2
 eim1=0.00001;
 exim1=-eim1*Vbxi/Vbi;
 eyim1=-eim1*Vbyimean/Vbi;
 end;
gi=0.5*eim1*((exim1-deltasxi)*(exim1-deltasxi)+(eyim1-deltasyi)*...
     (eyim1-deltasyi)-ei*ei)/(exim1*(exim1-deltasxi)+eyim1*(eyim1-deltasyi));
exi=(1-gi/eim1)*exim1-deltasxi;
```

```
eyi=(1-gi/eim1)*eyim1-deltasyi;
ein=sqrt(exi*exi+eyi*eyi);
exi=(ei/ein)*exi;  %correction
eyi=(ei/ein)*eyi;  %correction
pxi=(exi/ei)*mu*pzi;
pyi=(eyi/ei)*mu*pzi;
 if gi<0                    % element starts to adhere to the road
 sliding=0;
 exi=-deltasxi+exim1;
 eyi=-deltasyi+eyim1;
 ei=sqrt(exi*exi+eyi*eyi); e(i)=ei;
 pii=cp*ei;
 pxi=cp*exi;
 pyi=cp*eyi;
  if pii>mu*pzi
  pxi=(exi/ei)*mu*pzi;
  pyi=(eyi/ei)*mu*pzi;
  pii=mu*pzi;
  sliding=1;
  end;
 end;
else                        % element is adhering to the road
exi=-deltasxi+exim1;
eyi=-deltasyi+eyim1;
ei=sqrt(exi*exi+eyi*eyi); e(i)=ei;
pii=cp*ei;
pxi=cp*exi;
pyi=cp*eyi;
 if pii>mu*pzi              % element starts to slide
 sliding=1;
  if i==2
  eim1=0.00001;
  exim1=-eim1*Vbxi/Vbi;
  eyim1=-eim1*Vbyimean/Vbi;
  end;
 ei=mu*pzi/cp;
 gi=0.5*eim1*((exim1-deltasxi)*(exim1-deltasxi)+(eyim1-deltasyi)*...
     (eyim1-deltasyi)-ei*ei)/(exim1*(exim1-deltasxi)+eyim1*(eyim1-deltasyi));
 exi=(1-gi/eim1)*exim1-deltasxi;
 eyi=(1-gi/eim1)*eyim1-deltasyi;
 ein=sqrt(exi*exi+eyi*eyi);
 exi=(ei/ein)*exi;  %correction
 eyi=(ei/ein)*eyi;  %correction
 pxi=(exi/ei)*mu*pzi;
 pyi=(eyi/ei)*mu*pzi;
 end;
end;

pii=sqrt(pxi*pxi+pyi*pyi);
```

```
Fxi=Fxim1+pxi*2*a/n;
Fyi=Fyim1+pyi*2*a/n;
Mzi=Mzim1+(xi*pyi -ybieff*pxi)*2*a/n;
py(i)=pyi;
Fxim1=Fxi;
Fyim1=Fyi;
Mzim1=Mzi;

if picture==1
 if ibLR==-1
 yL(1,i)=mu*pzi; yL(2,i)=pxi; yL(3,i)=pyi; yL(4,i)=pii; yL(5,i)=0;
 xelL(3*i-2)=xi; xelL(3*i-1)=xi+exi ; xelL(3*i)=xi;
 yelL(3*i-2)=ybi-bLR; yelL(3*i-1)=ybi-bLR+eyi ; yelL(3*i)=ybi-bLR;
 ey(i)=eyi;
 else
  if ibLR==0
    yM(1,i)=mu*pzi; yM(2,i)=pxi; yM(3,i)=pyi; yM(4,i)=pii; yM(5,i)=0; yM(6,i)=pzi;
yM(7,i)=1000*Vbi;
  xelM(3*i-2)=xi; xelM(3*i-1)=xi+exi ; xelM(3*i)=xi;
  yelM(3*i-2)=ybi-bLR; yelM(3*i-1)=ybi-bLR+eyi ; yelM(3*i)=ybi-bLR;
  ey(i)=eyi;
  else
  yR(1,i)=mu*pzi; yR(2,i)=pxi; yR(3,i)=pyi; yR(4,i)=pii; yR(5,i)=0;
  xelR(3*i-2)=xi; xelR(3*i-1)=xi+exi ; xelR(3*i)=xi;
  yelR(3*i-2)=ybi-bLR; yelR(3*i-1)=ybi-bLR+eyi ; yelR(3*i)=ybi-bLR;
  eyR(i)=eyi;
  end;
 end;
end;

end;                    % **  end i-loop (passage through contact length)

if ibLR==-1
 FxL=Fxi; FyL=Fyi; MzL=Mzi;
else
if ibLR==0
 FxM=Fxi; FyM=Fyi; MzM=Mzi;
else
 FxR=Fxi; FyR=Fyi; MzR=Mzi;
end;
end;
end;                    % **  end iLR-loop (one, two or three rows of elements)

if nrow==1
 Fx=FxM;
 Fy=FyM;
 Mz=MzM;
else
 if nrow==2
```

```
 Fx=FxL+FxR;
 Fy=FyL+FyR;
 Mz=MzL+MzR;
 else
 Fx=FxL+FxR+FxM;
 Fy=FyL+FyR+FyM;
 Mz=MzL+MzR+MzM;
 end;
end;

Mzprime=Mz+Fx*ymeff*epsyprime;  % for torsion (slope) calculation

end;                    % ** end iit-loop (carcass deflection iteration)

if rigid==1
  Mz=Mz-c*Fx*Fy-y0*Fx;
else
  Mz=Mz-y0*Fx;
end;

t=-Mz/Fy;                    % pneumatic trail

yfx(ia,j)=Fx;
yfy(ia,j)=Fy;
ymz(ia,j)=Mz;
yt(ia,j)=t;

if kapparuns==1
xa(ia,j)=kappa;
end;
if alpharuns==1
xa(ia,j)=alphadeg;
end;
if aphitruns==1
xa(ia,j)=aphit;
end;
if alpharuns==1              % calculation of CFalpha/Fz (take deltalphadegr small!)
     if nCFa==0
          if ia>1
               if alpha>0
               if gamma==0
                         if abs(aphit)<0.001
                              if kappa==0
                                   CFalphaFz=(Fy-Fym1)/(Fz*pi*deltalphdegr/180);
                                   nCFa=1;
                              end;
                         end;
                    end;
               end;
```

```
            end;
        end;
 else
  CFalphaFz='?';
 end;

 if alpharuns==1
  deltalphdegr=g1a*abs(alphadeg)+g2a;
  alphadeg=alphadeg+deltalphdegr;
 end;
 if kapparuns==1
  deltakappar=g1k*abs(kappa)+g2k;
  kappa=kappa+deltakappar;
 end;
 if aphitruns==1
  deltaphitr=g1t*abs(aphit)+g2t;
  aphit=aphit+deltaphitr;
 end;

end;          % ** end running variable ia-loop

end;          % ** end motion parameter j-loop


if picture==1
xelL(1)=a; xelL(2)=a ; xelL(3)=a;
yelL(1)=0; yelL(2)=0 ; yelL(3)=ym +cs*a+0.5*cc*a*a;
xelL(3*(n+1)+1)=-a; yelL(3*(n+1)+1)=0;
xelL(3*(n+1)+2)=a; yelL(3*(n+1)+2)=0;
xelM(1)=a; xelM(2)=a ; xelM(3)=a;
yelM(1)=0; yelM(2)=0 ; yelM(3)=ym +cs*a+0.5*cc*a*a;
xelM(3*(n+1)+1)=-a; yelM(3*(n+1)+1)=0;
xelM(3*(n+1)+2)=a; yelM(3*(n+1)+2)=0;
xelR(1)=a; xelR(2)=a ; xelR(3)=a;
yelR(1)=0; yelR(2)=0 ; yelR(3)=ym +cs*a+0.5*cc*a*a;
xelR(3*(n+1)+1)=-a; yelR(3*(n+1)+1)=0;
xelR(3*(n+1)+2)=a; yelR(3*(n+1)+2)=0;
  if rigid==1
    yc=Fy*c;
          for i=3:3*(n+1)
      yelL(i)=yelL(i)+yc;
      yelM(i)=yelM(i)+yc;
      yelR(i)=yelR(i)+yc;
    end;
     end;
              % plots picture of force distribution and deformations
if nrow~=2
figure(1);
plot(x,yM);
```

```
xlabel('x  [m]');
ylabel('d.blue: mu*pz, l.blue: p, green: px, red: py, sepia: pz[N/m], black: Vb[mm/s]');
text(-0.1,16500,...
     ['alpha= ',num2str(alphapict),' deg, gamma= ',num2str(gammapict),' deg, kappa= ',....
      num2str(kappapict),', R= ',num2str(Rpict),'m']);
text(-0.1,18500,...
     ['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),', clat= ',...
      num2str(clat),', cp= ',num2str(cp)]);
text(-0.1,14500,['theta= ',num2str(theta)]);
text(-0.1,12500,['epsyprime= ',num2str(epsyprime),...
     ', epsyrgamma= ',num2str(epsyrgamma)]);
text(-0.1,10500,['epsyFy= ',num2str(epsyFy),', epsgamma= ',num2str(epsgamma)]);
text(-0.1,8500,['epsdrgam= ',num2str(epsdrgam),', y0= ',num2str(y0*1000),'mm']);
 if nrow==1
  text(0.03,12500,['SINGLE ROW']);
 else
  text(0.03,12500,['MIDDLE ROW']);
 end;
axis([-0.11,0.11,-5000,20000]);
grid;

figure(2);
plot(xelM,-yelM);
xlabel('x and longitudinal deflections [m]');
ylabel('lateral deflections  [m]');
text(-0.1,0.007,['alpha= ',num2str(alphapict),' deg, gamma= ',num2str(gammapict),...
     ' deg, kappa= ', num2str(kappapict),', R= ',num2str(Rpict),'m']);
 if rigid==1
  text(-0.1,0.009,['** RIGID **,  c= ',num2str(c),'m/N']);
 else
  text(-0.1,0.009,['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),...
     ', clat= ',num2str(clat),', cp= ',num2str(cp)]);
 end;
text(-0.02,0.005,['Fy= ',num2str(round(Fy)),'N,  Mz= ',num2str(round(Mz)),'Nm']);
 if nrow==1
  text(-0.1,0.005,['SINGLE ROW']);
 else
  text(-0.1,0.005,['MIDDLE ROW']);
 end;
axis([-0.11,0.11,-0.04,0.01]);
end;
if nrow~=1
figure(3);
plot(x,yR);
xlabel('x  [m]');
ylabel('d.blue: mu*pz,  l.blue: p,  green: px,  red: py  [N/m]');
text(-0.1,16500,['alpha= ',num2str(alphapict),' [deg], gamma= ',num2str(gammapict),...
     ' [deg], kappa= ',num2str(kappapict),' [-], R= ',num2str(Rpict),' [m]']);
text(-0.1,18500,['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),...
```

```
      ', clat= ',num2str(clat),', cp= ',num2str(cp)]);
text(-0.1,12500,['RIGHT ROW']);
axis([-0.11,0.11,-5000,20000]);
grid;
figure(4);
plot(xelR,-yelR);
xlabel('x and longitudinal deflections [m]');
ylabel('lateral deflections  [m]');
text(-0.1,0.007,['alpha= ',num2str(alphapict),' [deg], gamma= ',num2str(gammapict),...
      ' [deg], kappa= ',num2str(kappapict),' [-], R= ',num2str(Rpict),' [m]']);
 if rigid==1
  text(-0.1,0.009,['** RIGID **,  c= ',num2str(c),'m/N']);
 else
text(-0.1,0.009,['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),', clat= ',num2str(clat),...
      ', cp= ',num2str(cp)]);
 end;
text(-0.1,0.005,['RIGHT ROW,   brow/a= ',num2str(brow/a)]);
axis([-0.11,0.11,-0.04,0.01]);

figure(5);
plot(x,yL);
xlabel('x  [m]');
ylabel('d.blue: mu*pz,  l.blue: p,  green: px,  red: py  [N/m]');
text(-0.1,16500,['alpha= ',num2str(alphapict),' deg, gamma= ',num2str(gammapict),...
      ' deg, kappa= ', num2str(kappapict),', R= ',num2str(Rpict),'m']);
text(-0.1,18500,['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),...
      ', clat= ',num2str(clat),', cp= ',num2str(cp)]);
text(-0.1,14500,['theta= ',num2str(theta)]);
text(-0.1,12500,['epsyprime= ',num2str(epsyprime),...
      ', epsyrgamma= ',num2str(epsyrgamma)]);
text(-0.1,10500,['epsyFy= ',num2str(epsyFy),', epsgamma= ',num2str(epsgamma)]);
text(-0.1,8500,['epsdrgam= ',num2str(epsdrgam),', y0= ',num2str(y0*1000),'mm']);
text(0.03,12500,['LEFT ROW']);
axis([-0.11,0.11,-5000,20000]);
grid;

figure(6);
plot(xelL,-yelL);
xlabel('x and longitudinal deflections [m]'); ylabel('lateral deflections  [m]');
text(-0.1,0.007,['alpha= ',num2str(alphapict),' deg, gamma= ',num2str(gammapict),...
      ' deg, kappa= ', num2str(kappapict),', R= ',num2str(Rpict),'m']);
 if rigid==1
  text(-0.1,0.009,['** RIGID **,  c= ',num2str(c),'m/N']);
 else
text(-0.1,0.009,['cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),', clat= ',num2str(clat),...
      ', cp= ',num2str(cp)]);
 end;
text(-0.02,0.005,['Fy= ',num2str(round(Fy)),'N,  Mz= ',num2str(round(Mz)),'Nm']);
text(-0.1,0.005,['LEFT ROW']);
```

```
axis([-0.11,0.11,-0.04,0.01]);
end;
                                    %    end if picture

else
        % plots with running variable
if alpharuns==1
xmin=alphdegminr; xmax=alphdegmaxr; xtext='slip angle  alpha  [deg]';
Fxmin=-mu0*Fz;   Fxmax=mu0*Fz;   Fymin=-mu0*Fz;   Fymax=mu0*Fz;
Mmin=1.1*0.1*a*Fymin; Mmax=0.9*0.1*a*Fymax;
tmin=-0.2*a; tmax=0.6*a; ax=0.25; ay=0;
partext1=['CFkap= ',num2str(CFkappaFz),'*Fz, CFalf= ',num2str(CFalphaFz),'*Fz'];
end;
if kapparuns==1
xmin=kappaminr; xmax=kappamaxr; xtext='longitudinal slip ratio  kappa  [-]';
Fxmin=-mu0*Fz;   Fxmax=mu0*Fz;   Fymin=-mu0*Fz;   Fymax=mu0*Fz;
Mmin=1.1*0.1*a*Fymin; Mmax=0.9*0.1*a*Fymax;
tmin=-0.2*a; tmax=0.6*a; ax=-0.25; ay=0;
partext1=['CFkap= ',num2str(CFkappaFz),'*Fz'];
end;
if aphitruns==1
xmin=0; xmax=aphitmaxr; xtext='non-dim. turn slip  aphit= a/R  [-]';
Fxmin=-mu0*Fz; Fxmax=mu0*Fz; Fymin=-mu0*Fz/3; Fymax=mu0*Fz; Mmin=1.1*a*Fymin/6;
Mmax=0.9*0.8*a*Fymax;
tmin=-a; tmax=0.2*a; ax=0.25; ay=0;
partext1=['CFkap= ',num2str(CFkappaFz),'*Fz'];
end;

xl=xmax-xmin; Fxl=Fxmax-Fxmin; Fyl=Fymax-Fymin; Ml=Mmax-Mmin; tl=tmax-tmin;

if kappapar==1
    if alpharuns==1
        partext0=['a/R= ',num2str(aphit),', gamma= ',num2str(gammadeg),' [deg]'];
    else
     partext0=['alpha= ',num2str(alphadeg),', gamma= ',num2str(gammadeg),' [deg]'];
  end;
    if jmax==1
    partext=['kappa= ',num2str(kappamin)];
  else
   partext=['kappamin= ',num2str(kappamin), ', deltakappa=',num2str(deltakappa),' [-]'];
  end;
elseif gammapar==1
    if alpharuns==1
        partext0=['a/R= ',num2str(aphit),', kappa= ',num2str(kappa)];
    elseif kapparuns==1
        partext0=['alpha= ',num2str(alphadeg),' [deg]',', a/R= ',num2str(aphit)];
    else
        partext0=['alpha= ',num2str(alphadeg),' [deg]',', kappa= ',num2str(kappa)];
    end;
```

```
    if jmax==1
    partext=['gamma= ',num2str(gamdegmin)],' [deg]';
  else
     partext=['gammamin= ',num2str(gamdegmin), ', deltagamma=',...
     num2str(deltgamdeg),' [deg]'];
  end;
elseif alphapar==1
    if kapparuns==1
         partext0=['a/R= ',num2str(aphit),', gamma= ',num2str(gammadeg),' [deg]'];
    else
   partext0=['kappa= ',num2str(kappa),', gamma= ',num2str(gammadeg),' [deg]'];
  end;
    if jmax==1
    partext=['alpha= ',num2str(alphdegmin),' [deg]'];
  else
    partext=['alphamin= ',num2str(alphdegmin), ', deltaalpha=',...
    num2str(deltalphdeg),' [deg]'];
  end;
elseif aphitpar==1
  if alpharuns==1
    partext0=['kappa= ',num2str(kappa),', gamma= ',num2str(gammadeg),' [deg]'];
    else
    partext0=['alpha= ',num2str(alphadeg),' [deg]',', gamma= ',num2str(gammadeg),' [deg]'];
  end;
    if jmax==1
   partext=['aphit= ',num2str(aphitmin)];
  else
   partext=['aphitmin= ',num2str(aphitmin), ', deltaaphit=',num2str(deltaphit),' [-]'];
  end;
else
    if alpharuns==1
         partext0=['a/R= ',num2str(aphit),', kappa= ',num2str(kappa)];
    elseif kapparuns==1
         partext0=['alpha= ',num2str(alphadeg),' [deg]',', a/R= ',num2str(aphit)];
    else
         partext0=['alpha= ',num2str(alphadeg),' [deg]',', kappa= ',num2str(kappa)];
    end;
    partext=['gamma= ',num2str(gamdegmin),' [deg]'];
end;
if rigid==0
  texttitle= ['Vc= ',num2str(Vc),'m/s, mu0= ',num2str(mu0),', amu= ',num2str(amu),....
     ', cyaw= ',num2str(cyaw),', cbend= ',num2str(cbend),', clat= ',num2str(clat)];
else
  texttitle= ['Vc= ',num2str(Vc),'m/s , mu0= ',num2str(mu0),' , amu= ',num2str(amu),....
     ' , c= ',num2str(c),' , y0= ',num2str(y0),'     (rigid carcass)'];
end;
partext2= ['a=',num2str(a),', b=',num2str(b),', brow=',num2str(brow), ', Fz=',num2str(Fz),...
 ', plyst=',num2str(alphadegply),', conic=',num2str(gammadegcon),',iitend=',num2str(iitend)];
```

```
figure(1);
plot(xa,yfy); ylabel('side force  Fy [N]'); xlabel(xtext);
title(texttitle);
text(xmin+0.02*xl, Fymin+(0.95-1.5*ay)*Fyl,['nr rows= ',num2str(nrow)]);
text(xmin+0.02*xl, Fymin+(0.9-1.5*ay)*Fyl,['theta= ',num2str(theta)]);
text(xmin+(0.27+ax)*xl, Fymin+0.205*Fyl,partext0);
text(xmin+(0.27+ax)*xl, Fymin+0.13*Fyl,partext);
text(xmin+(0.27+ax)*xl, Fymin+0.055*Fyl,partext1);
text(xmin+(0.27+ax)*xl, Fymin+0.01*Fyl,partext2);
axis([xmin,xmax,Fymin,Fymax]);
grid;

figure(2);
plot(xa,yfx); ylabel('longitudinal force  Fx [N]'); xlabel(xtext);
title(texttitle);
text(xmin+0.02*xl, Fymin+0.95*Fyl,['nr rows= ',num2str(nrow)]);
text(xmin+0.02*xl, Fymin+0.9*Fyl,['theta= ',num2str(theta)]);
text(xmin+(0.27-ax)*xl, Fymin+0.205*Fyl,partext0);
text(xmin+(0.27-ax)*xl, Fymin+0.13*Fyl,partext);
text(xmin+(0.27-ax)*xl, Fymin+0.055*Fyl,partext1);
axis([xmin,xmax,Fxmin,Fxmax]);
grid;

figure(3);
plot(xa,ymz); ylabel('align.torque  Mz [-]'); xlabel(xtext);
title(texttitle);
text(xmin+0.02*xl, Mmin+0.95*Ml,['nr rows= ',num2str(nrow),', nr elem= ',num2str(n)]);
text(xmin+0.02*xl, Mmin+0.9*Ml,['theta= ',num2str(theta)]);
text(xmin+0.52*xl, Mmin+0.95*Ml,['epsyprime= ',...
    num2str(epsyprime),', epsyrgamma= ',num2str(epsyrgamma)]);
text(xmin+0.52*xl, Mmin+0.88*Ml,['epsyFy= ',...
    num2str(epsyFy),', epsgamma= ',num2str(epsgamma)]);
text(xmin+0.52*xl, Mmin+0.81*Ml,['epsdrgam= ',...
    num2str(epsdrgam),', y0= ',num2str(y0*1000),'mm']);
text(xmin+0.02*xl, Mmin+(0.275+ay)*Ml,partext0);
text(xmin+0.02*xl, Mmin+(0.205+ay)*Ml,partext);
text(xmin+0.02*xl, Mmin+(0.13+ay)*Ml,partext1);
text(xmin+0.02*xl, Mmin+(0.055+ay)*Ml,partext2);
axis([xmin,xmax,Mmin,Mmax]);
grid;

figure(4);
plot(xa,yt); ylabel('pneum. trail [m]'); xlabel(xtext);  % (xamean,ytdiff)
title(texttitle);
text(xmin+0.02*xl, tmin+0.95*tl,['nr rows= ',num2str(nrow)]);
text(xmin+0.02*xl, tmin+0.205*tl,partext0);
text(xmin+0.02*xl, tmin+0.13*tl,partext);
text(xmin+0.02*xl, tmin+0.055*tl,partext1);
axis([xmin,xmax,tmin,tmax]);
```

```
grid;

figure(5);
plot(yfx,yfy); ylabel('side force [N]'); xlabel('longitudinal force [N]');
title(texttitle);
text(Fxmin+0.03*Fxl, Fymin+0.95*Fyl,['nr rows= ',num2str(nrow)]);
text(Fxmin+0.03*Fxl, Fymin+0.205*Fyl,partext0);
text(Fxmin+0.03*Fxl, Fymin+0.13*Fyl,partext);
text(Fxmin+0.03*Fxl, Fymin+0.055*Fyl,partext1);
text(Fxmin+0.03*Fxl, Fymin+0.01*Fyl,partext2);
axis([1.2*Fxmin,1.2*Fxmax,Fymin,Fymax]);
grid;

figure(6);
plot(yfx,ymz); ylabel('aligning torque [Nm]'); xlabel('longitudinal force [N]');
title(texttitle);
text(Fxmin+0.03*Fxl, Mmin+0.95*Ml,['nr rows= ',num2str(nrow)]);
text(Fxmin+0.52*Fxl, Mmin+0.95*Ml,['epsyprime= ',num2str(epsyprime),...
      ', epsyrgamma= ',num2str(epsyrgamma)]);
text(Fxmin+0.52*Fxl, Mmin+0.88*Ml,['epsyFy= ',num2str(epsyFy),...
      ', epsgamma= ',num2str(epsgamma)]);
text(Fxmin+0.52*Fxl, Mmin+0.81*Ml,['epsdrgam= ',num2str(epsdrgam),...
      ', y0= ',num2str(y0*1000),'mm']);
text(Fxmin+0.03*Fxl, Mmin+0.205*Ml,partext0);
text(Fxmin+0.03*Fxl, Mmin+0.13*Ml,partext);
text(Fxmin+0.03*Fxl, Mmin+0.055*Ml,partext1);
text(Fxmin+0.03*Fxl, Mmin+0.0*Ml,partext2);
%axis([1.2*Fxmin,1.2*Fxmax,Mmin,Mmax]);
grid;

end;
clear;
```