cosin
scientific software

cosin scientific software AG

Hans-Stützle-Str. 20
81249 München
GERMANY

info@cosin.eu
www.cosin.eu

FTire - Flexible Structure Tire Model

Modelization and Parameter Specification

Contents

1 Legal Notices

This documentation is intended for qualified users who will exercise sound engineering judgment and expertise in the use of the FTire software. The FTire software is inherently complex, and the explanations in this documentation are not intended to be exhaustive or to apply to any particular situation. Users are cautioned to satisfy themselves as to the accuracy and results of their analyses.

Cosin scientific software AG shall not be responsible for the accuracy or usefulness of any analysis performed using the FTire software or the explanations in this documentation. Cosin scientific software AG shall not be responsible for the consequences of any errors or omissions that may appear in this documentation.

The FTire software is available under license from cosin scientific software AG and may be used or reproduced only in accordance with the terms of such license. This documentation is subject to the terms and conditions of the then current software license agreement to which the documentation relates.

This documentation and the software described in this documentation are subject to change without prior notice.

The FTire software is a product of cosin scientific software AG, Muenchen, Germany.

2 Aims and Scope of FTire

FTire (**F**lexible Structure **Tire** Model) is a full 3D nonlinear in-plane and out-of-plane tire simulation model. It is used by engineers in the vehicle and tire industry worldwide. Sophisticated 2D and 3D rigid and flexible road surface description models and evaluation methods, and powerful toolboxes for tire and road data processing make **FTire** the most comprehensive software package for tire dynamics simulation on the market.

FTire is designed for vehicle comfort simulations and prediction of road loads on road irregularities even with extremely short wave-lengths. It can also be used as a structural dynamics based, highly nonlinear and dynamic tire model for handling studies without modifications of input parameters.

FTire explains most of the complex tire phenomena on a mechanical, thermodynamical, and tribological basis, with very good correlation to measurements:

- structural dynamics based, spatial nonlinear in-plane and out-of-plane tire model for simulation of belt dynamics, local contact patch pressure distribution, rolling resistance, side-wall contact, large camber angles and misuse scenarios;
- suitable for a frequency range up to 200 Hz (or even 300 Hz if air cavity model is activated), excited by short surface wavelengths, mass imbalance, non-uniformity of tire and/or rim, air cavity vibrations, or irregular tread patterns;
- very fast and flexible, up to real-time capability. Orders of magnitude faster than explicit FE models;
- simulation of imbalances by inhomogeneous mass and stiffness distribution, radius variation, and local tread wear;
- belt temperature distribution model;
- air volume vibration model;
- capability of tire slipping on rim for very large drive or brake torques;
- integrated flexible and/or viscoplastic rim model;
- support for user-defined wear, temperature distribution, and rim flexibility models;
- full integration of **cosin/road** digital road library with support for complex rigid time-invariant and time-variant road surfaces;
- full integration of **cosin/soil** digital road library with support for flexible and deformable road surfaces;
- advanced online animation with belt deformation animation, tire temperature distribution animation, pressure distribution plots, road surface visualization and movie export;
- robust, multi-core system enabled solver engine;
- parameter editing and validation tools;
- tailored parameter fitting tool (**FTire/fit**).

3 FTire Implementation and Interfaces

The **FTire** core library can be connected to all important simulation environments, by using **cosin**'s tire interface (CTI), a C/C++ API. CTI provides a **time-discrete generalized interface** and reduces the implementation effort of **FTire** to a minimum. CTI is used by the **FTire** implementations in Adams (all variants), MotionSolve, SIMPACK, Abaqus, VI-CarRealTime, Matlab/Simulink, dSPACE-ASM, CarSim/TruckSim/BikeSim, IPG Carmaker/Truckmaker, CASCaDE, **cosin/mbs**, DAFUL, Dymola/Modelica, FEDEM, Mesa Verde, PAM-Crash, RecurDyn, veDyna, Virtual.Lab Motion, and others.

For Matlab/Simulink an S-function layer is available (**FTire/link**). This S-function is completed by a respective Simulink block-set.

All interfaces are designed to run an arbitrarily large number of tire instances simultaneously.

In either case, the coupling to the vehicle or suspension model of the calling program is done by the **rigid body state variables** of the rim, that is:

- **position** of the rim center in the inertial frame;
- **translational velocity vector** of the rim center;
- **angular orientation** of the rim, defined by the transformation matrix from the rim-fixed frame to the inertial frame. Euler angles, Cardan angles, or Euler parameters of the rim can be passed by an alternative API call. So the user can pass coordinates in the native reference frame of the calling application;
- **rotational velocity vector** of the rim.

FTire returns forces and torques acting on the rim center, represented in the global coordinate system.

Alternatively, **FTire** can be used to simultaneously integrate the rim rotation with respect to the hub-carrier. In this use mode, not the rigid-body states of the rim, but rather those of the wheel-carrier are the inputs, together with the driving and the maximum absolute braking torque. The output torque vector then does not contain the share in the direction of the wheel rotation. **FTire** eventually modulates the braking torque, if the wheel is blocked, in order to maintain this blocking as long as it is necessary.

4 FTire Modelization

4.1 Mechanical Model

FTire is based on a structural dynamics based tire modeling approach.

In this core mechanical model, the tire belt is described as an extensible and flexible ring, carrying bending stiffnesses, elastically founded on the rim by distributed, partially dynamic stiffnesses in radial, tangential, and lateral direction. The degrees of freedom of this ring are such that belt in-plane, as well as out-of-plane, motions are possible. The ring is numerically approximated by a finite number of 'belt elements'. These belt elements are coupled with their direct neighbors by stiff springs and by bending stiffnesses both in-plane and out-of-plane.

All stiffnesses, bending stiffnesses, and damping factors are calculated during pre-processing, fitting the prescribed modal and static properties (cf. list of data below).

To every belt element, a number (typically 5 to 50) of mass-less 'tread blocks' are associated. These blocks carry nonlinear stiffness and damping properties in radial, tangential, and lateral direction. The radial deflections of the blocks depend on the road profile, locus, and orientation of the associated belt elements. Tangential and lateral deflections are determined by the sliding velocity on the ground and the local values of the sliding coefficient. The latter depends on ground pressure and sliding velocity. 'Radial', 'tangential', and 'lateral' is to be understood relatively to the orientation of the belt element, whereas 'sliding velocity' is the block end point velocity, projected onto the road profile tangent plane. By polynomial interpolation, certain precautions have been taken not to let the ground pressure distribution mirror the in-plane polygonal shape of the 'belt chain'.

To approximate reactions to out-of-plane excitations more accurately, every belt element has several additional degrees of freedom. These degrees of freedom describe the element's longitudinal rotation angle relative to the rim, and the element's bending in the lateral direction. The rotation angles are coupled by rotational stiffnesses, located between two adjacent belt elements, as well as rotational stiffness for each belt element, located between the belt elements and rim. At the same time, the coupling between the lateral displacement of a belt element and its torsion angle is taken into account by an appropriate coupling stiffness.

The tread blocks, as mentioned above, are located along several parallel lines, such that both longitudinal and lateral resolution of the road surface is optimized.

In unloaded condition, the belt elements are curvilinear in lateral direction. The actual curvature, however, is not only determined by these geometrical values (which are defined by cross-section spline data), but at the same time by the belt lateral bending degrees of freedom. The belt elements' bending values, in turn, are determined by the respective bending moments which are functions of the vertical forces of the belt elements' tread blocks.

All 6 components of tire forces and torques acting on the rim are calculated by integrating the forces in the elastic foundation of the belt.

The resulting overall tire model is accurate up to high frequencies, in both longitudinal and lateral direction. There are few restrictions in the applicability w.r.t. longitudinal, lateral, and vertical vehicle dynamics situations. **FTire** deals with large and/or short-waved obstacles. It works out of, and up to, complete stand still, with **no** additional computing effort and **without** any model switch. Finally, it is applicable and accurate in more delicate simulations such as ABS breaking on extremely uneven road surfaces, etc.

Optionally, **FTire** can take into account tire non-uniformity, which is a harmonic or more general variation of the radial or tangential stiffness, as well as static and dynamic imbalances.

Kernel of the **FTire** implementation is an implicit integration algorithm that calculates the belt shape. By using this specialized implicit BDF integrator, the belt extensibility may be chosen to be extremely small. **FTire** thus also allows the simulation of an in-extensible belt without any numerical drawback.

4.2 Thermal Model

FTire provides an optional **detailed thermal model**, the foundations of which are described in this chapter. This model is of special interest in case of strongly temperature-dependent friction properties. Activation and deactivation is described in chapter8.

4.2.1 Thermal Model Structure

The **FTire** thermal model consists of the following components:

- the **thermo-dynamical computation of the actual inflation pressure** as function of air (or filling gas) mass, 'cold tire inflation pressure', tire temperature, and actual interior volume. The tire filling gas is considered to be ideal.
- a **heat generation and transfer model**, introducing state variables for the temperature of the tire structure (including filling gas), and the individual temperature of each tread contact element. Heat generation and transfer is driven by the power loss distribution due to structural damping and dry friction on the road surface.
- the introduction of a third independent input variable **'rubber temperature'** of the tread friction characteristics, in addition to **ground pressure** and **sliding velocity**. This third independent variable is taken into account in terms of a temperature-dependent correction factor of the friction coefficients:

$$\mu = f\left(T\right) \cdot \mu_0\left(v,\, p_{ground}\right)$$

4.2.2 Heat Generation and Heat Transfer Model

For simplicity, in the heat generation and transfer model, the tire is assumed to being separated into three regions (figure 4.1), having different thermal properties each.

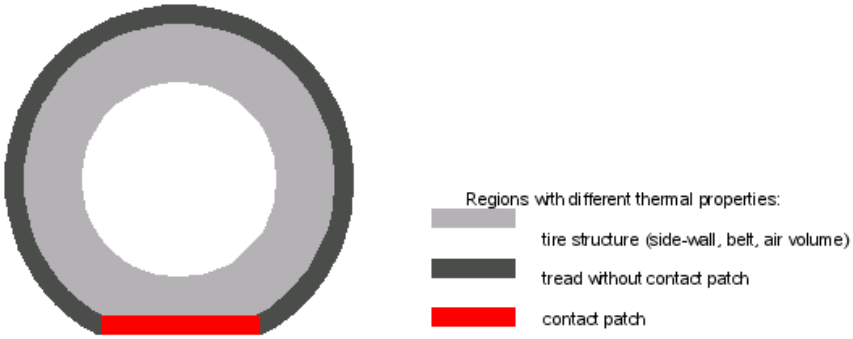

Figure 4.1: Regions of the thermal model

The following assumptions are made for the three different regions:

The **tire structure** is described by **one global temperature** only. The product of rate of change of this temperature and the tire structure's overall heat capacity is the balancing the sum of:

- the power lost in all damping elements in belt and side-wall, excluding friction and damping of the tread elements.
- the heat which flows from the tire structure into the two tread regions. This heat transfer is determined by the respective temperature differences, multiplied by an appropriate heat transfer coefficient. This heat transfer coefficient is assumed to be independent on rolling speed.
- the heat which flows from the side-walls into the air flowing around the tire. This heat transfer is determined by the temperature difference between tire structure and ambient air, multiplied by an appropriate heat transfer coefficient. This heat transfer coefficient is assumed to be strongly and nonlinearly dependent on rolling speed.

Document Revision: 2022-1-r26108

The **tread without contact patch** is described by a **distributed temperature**, assigning one temperature value individually to each tread element. The product of rate of change of this temperature and the tread element's heat capacity is balancing the sum of:

- the power lost in the tread element due to material damping.
- the share of the heat which flows from the tire structure into the particular tread element. This heat transfer is the same as the already described heat transfer above.
- the heat which is transferred from the tread element to the air flowing around the tire. This heat transfer is determined by the temperature difference between tread element and ambient air, multiplied by an appropriate heat transfer coefficient. This heat transfer coefficient is assumed to be strongly and nonlinearly dependent on rolling speed, but has the same value for all elements not in contact to the road.

The **contact patch** is described by a **distributed temperature**, assigning one temperature value to each individual tread element in the contact patch. The product of rate of change of this temperature and the tread element's heat capacity is balancing the sum of:

- a certain fraction $\triangle P_{frict,tread}$ of the power lost in the tread element due to dry friction (the remaining friction power is heating the road).
- the power lost in the tread element due to material damping.
- the share of the heat which flows from the tire structure into the particular tread element. This heat transfer is the same as already described above.
- the heat which flows from the tread element into the road surface. This heat transfer is determined by the temperature difference between tread elements and road surface, multiplied by an appropriate heat transfer coefficient. This heat transfer coefficient is assumed to be independent of rolling speed, and has the same value for all elements which are in contact with the road.

This model requires the following **data**:

- Heat capacity C_S of the tire structure.
- Heat capacity C_T of the tread; will be subdivided equally into the individual tread elements. The actual tread depth is used to correct the actual heat capacity.
- Heat transfer coefficient $\kappa(v) \cdot c_{SE}$ between the environment and the tire structure, assumed to be dependent on rolling speed. This dependency is described by the factor $\kappa(v)$.
- Heat transfer coefficient c_{ST} between the tread and the tire structure (will be subdivided equally into the individual tread elements). This coefficient is assumed to be independent of the rolling speed.
- Heat transfer coefficient $\kappa(v) \cdot c_{TE}$ between the environment and the tread, which is not in contact with the road. This coefficient is assumed to dependent on the rolling speed of the tyre and is described by the factor $\kappa(v)$.
- Heat transfer coefficient c_{TR} between the contact patch and the road surface (will be subdivided equally into the individual tread elements). This coefficient is assumed to be independent of the tire rolling speed.

To make data independent of the actual contact patch length, both values $\kappa(v) \cdot c_{TE}$ and c_{TR} are defined as heat transfer coefficients per unit length, multiplied by the tire circumference.

The above data is proposed to be approximated by the following procedures.

4.2.3 Determination of Heat Transfer Coefficients on Basis of Steady-State Temperatures

To determine **the heat transfer coefficients**, the following equations are used to describe the dependencies of a given set of steady-state temperatures:

$$\left.\begin{array}{c} \kappa_i\, c_{SE} \cdot (T_{S,i} - T_{env}) + c_{ST} \cdot (T_{S,i} - T_{T,i}) = 0 \\ c_{ST} \cdot (T_{T,i} - T_{S,i}) + (\lambda\, c_{TR} + (1-\lambda)\, f_i\, c_{TE}) \cdot (T_{T,i} - T_{env}) = P_{T.loss,i} \end{array}\right\} \; (i = 1, 2, \ldots)$$

Document Revision: 2022-1-r26108

Here, index i denotes different rolling speeds at which the steady-state temperatures of the tread and eventually the tire structure are assumed to being measured: v_1 is 25% of v_{max}, v_2 is 50% of v_{max}, and $v_3 = v_{max}$. λ is the relative share of the contact patch length to the rolling circumference at the assumed load (LI load, say) where the measurements have been taken. Factors $\kappa_i = \kappa(v_i)$ describe the dependency of the rolling speed on the heat transfer due to the air-flow around the tire. The intention is to determine heat transfer coefficients on the basis of measured steady-state temperatures as accurate, robust, and reliable as possible. Several different approaches indicate that the following choice is a good compromise between accuracy and robustness of identification:

- Choose $\alpha = \frac{c_{SE}}{c_{TE}}$ and $\beta = \frac{c_{ST}}{c_{TR}}$ to be given a fixed number (1, say) that is not subjected to the identification
- Normalize $\kappa_1 = 1$ for an arbitrary reference rolling speed
- Use $T_S = T_{S,1}$, $T_T = T_{T,1}$ and $P_{T.loss} = P_{T,loss,1}$ to determine and c_{TE} and c_{TR}
- For any arbitrary additional v_i, use $T_{T,i}$ to determine κ_i

This leads to the following equations:

$$\begin{gathered}\alpha\, c_{TE} \cdot (T_S - T_{env}) + \beta\, c_{TR} \cdot (T_S - T_T) = 0 \\ \beta\, c_{TR} \cdot (T_T - T_S) + (\lambda\, c_{TR} + (1-\lambda)\, c_{TE}) \cdot (T_T - T_{env}) = P_{T,loss}\end{gathered}$$

so

$$\begin{bmatrix} \alpha\,(T_S - T_{env}) & \beta\,(T_S - T_T) \\ (1-\lambda)\,(T_T - T_{env}) & \beta\,(T_T - T_S) + \lambda\,(T_T - T_{env}) \end{bmatrix} \begin{bmatrix} c_{TE} \\ c_{TR} \end{bmatrix} = \begin{bmatrix} 0 \\ P_{T,loss} \end{bmatrix}$$

which can easily be resolved for c_{TE} and c_{TR}.

For any other rolling speed, it holds:

$$T_{S,i} = \frac{\kappa_i\, \alpha\, c_{TE} \cdot T_{env} + \beta\, c_{TR} \cdot T_{T,i}}{\kappa_i\, \alpha\, c_{TE} + \beta\, c_{TR}}$$

or

$$T_{T,i} - T_{S,i} = \frac{\kappa_i\, \alpha\, c_{TE}}{\kappa_i\, \alpha\, c_{TE} + \beta\, c_{TR}} (T_{T,i} - T_{env})$$

Substitute into the equation above:

$$\left(\beta\, c_{TR} \cdot \frac{\kappa_i\, \alpha\, c_{TE}}{\kappa_i\, \alpha\, c_{TE} + \beta\, c_{TR}} + \lambda\, c_{TR} + (1-\lambda)\, \kappa_i\, c_{TE} \right) \cdot (T_{T,i} - T_{env}) = P_{T,loss,i}$$

After introducing the dimensionless variable $\gamma_i = \frac{P_{T,loss,i}}{c_{TR}\,(T_{T,i} - T_{env})}$, and some rearrangements, one gets the following quadratic equation for κ_i:

$$\left(\frac{c_{TE}}{c_{TR}}\, \kappa_i \right)^2 + \left(\frac{\beta}{\alpha} + \frac{\beta}{1-\lambda} - \frac{\gamma_i - \lambda}{1-\lambda} \right) \frac{c_{TE}}{c_{TR}}\, \kappa_i - \frac{\beta}{\alpha} \cdot \frac{\gamma_i - \lambda}{1-\lambda} = 0$$

which, due to the parameter ranges, will always have two real solutions. Moreover, one solution will be positive and the other one negative, as long as:

$$\frac{\gamma_i}{\lambda} = \frac{P_{T,loss,i}}{\lambda\, c_{TR}\, (T_{T,i} - T_{env})} > 1$$

meaning the amount of heat generated in the contact patch, due to friction, is greater than the heat being transferred to the road in steady state conditions. This equation is used as a plausibility check for $T_{T,i}$. In any case, the solution of f_i will be selected which is closer to $+1$.

As a matter of convenience, instead of $T_{S,i}$, $T_{T,i}$, the temperature increases $\triangle T_{S,i}$ and $\Delta T_{T,i}$ relative to ambient temperature are to be provided as model data.

4.2.4 Determination of Heat Capacities on Basis of Heating Time Constants

To determine the heat capacities from the heating time constants, the following 2 differential equations, of heat conduction at medium rolling speed, are used:

$$\begin{gathered} C_s\,\dot{T}_{S,2} + f_2\,c_{SE}\cdot(T_{S,2} - T_{env}) + c_{ST}\cdot(T_{S,2} - T_{T,2}) = 0 \\ C_T\,\dot{T}_{T,2} + c_{ST}\cdot(T_{T,2} - T_{S,2}) + (\lambda\,c_{TR} + (1-\lambda)\,f_2\,c_{TE})\cdot(T_{T,2} - T_{env}) = P_{T,loss,2} \end{gathered}$$

Here, for simplicity the assumption is made that tread temperature is constant along belt circumference. These equations are governed by the $2x2$ linear system matrix:

$$A = \begin{bmatrix} -\frac{f_2\,c_{SE}+c_{ST}}{C_S} & \frac{c_{ST}}{C_S} \\ \frac{c_{ST}}{C_T} & -\frac{\lambda\,c_{TR}+(1-\lambda)\,f_2\,c_{TE}+c_{ST}}{C_T} \end{bmatrix}$$

Using the following abbreviations: $p = f_2\,c_{SE} + c_{ST}$, $q = c_{ST}$, $r = \lambda\,c_{TR} + (1-\lambda)\,f_2\,c_{TE} + c_{ST}$, $s = p\,r - q^2$, the (real) eigenvalues of this matrix are determined by:

$$\begin{gathered} \lambda_1 + \lambda_2 = tr\,(A) = -\tfrac{p}{C_S} - \tfrac{r}{C_T} = -\tfrac{r\,C_s + p\,C_T}{C_S\,C_T} \\ \lambda_1\lambda_2 = \det(A) = \tfrac{s}{C_S\,C_T} \end{gathered}$$

Due to $T_i = -\frac{1}{\lambda_i}$, it follows $r\,C_S + p\,C_T = -s\frac{\lambda_1+\lambda_2}{\lambda_1\cdot\lambda_2} = s\cdot(T_1 + T_2)$, therefore:

$$\begin{gathered} r\,C_S + p\,C_T = s\cdot(T_1 + T_2) \\ C_S\cdot C_T = s\cdot T_1\cdot T_2 \end{gathered}$$

The last equation finally leads to a quadratic equation for either of the two heat capacities, having the two solutions:

$$\begin{gathered} C_S = \tfrac{s}{2\,r}\,(T_1 + T_2) \pm \sqrt{\tfrac{s^2}{4\,r^2}\,(T_1 + T_2)^2 - \tfrac{p\,s}{r}\,T_1\,T_2} \\ C_T = \tfrac{s\,T_1\,T_2}{C_S} \end{gathered}$$

These two equations tell us how to compute the heat capacities from heating time constants. Unfortunately, as a quadratic equation is underlying, they can have no, one or two solution(s). To get a unique approximation in either case, the following rules are introduced:

- if the solution is complex, the 'nearest' real solution is taken, the parameters are thus given by:

$$\begin{gathered} C_S = \tfrac{s}{2\,r}\,(T_1 + T_2) \\ C_T = \tfrac{s\,T_1\,T_2}{C_S} \end{gathered}$$

- if the solution is real, select the solution so that $C_S > C_T$;
- if the solution is real and $C_S > C_T$ holds for both solutions, or for no solution, then the pair of heat capacities is chosen for which $|C_S - C_T|$ has the smallest value.

4.3 Tread Wear Model

For any tread element, the individual instantaneous **power loss** due to road friction is available as output of the tire force model. Equally well, each tread element's estimated **instantaneous temperature** is available from the thermal sub-model.

The three variables: power loss $P_{frict} = |v_{slide}|\cdot|F_{friction}|$, temperature T, and tread-element's normal force F_N are fed into a 3D characteristic, predicting the instantaneous tread block's wear rate $\frac{dh}{dt}$:

$$\frac{dh}{dt} = -f\,(P_{frict}, T, F_N)$$

This wear rate is integrated, resulting in a tread-block-individual state variable 'tread element height h'. This state has location-dependent values, distributed along the tire circumferential and lateral coordinates.

Besides merely calculating the wear rate, this state variable is used as time-dependent tread depth in the force model. Thus, it affects the cross-sectional geometry, contact pressure distribution, radial tread stiffness, tread shear stiffness, tire mass, and the tread's heat capacity. Indirectly, tread wear affects the global tire stiffness and handling properties. Moreover, because actual tread depth is a function of the tread element's location in the tread, tire imperfections like 'spot wear' etc. are subject to possible investigation.

The real challenge of this approach is to determine the wear function. This function f can only be approximated on basis of appropriate measurements.

Starting point of the tread wear model is the following simple, two parameter, mathematical expression for f:

$$\frac{dh}{dt} = -c_{wear} \cdot \left(\frac{P_{frict}}{1 \frac{Nm}{s}} \right)^{e_{wear}}$$

These parameters, in turn, might be identified on basis of real life-time tire wear properties. For physical reasons the exponent e_{wear} is close to 1 while the coefficient c_{wear} determines the absolute wear rate. Because it is impractical to have to simulate 20000..60000 km to study the effects of tyre wear the user can do the following:

1. accelerate the wear, by choosing a coefficient value much larger than the actual wear rate coefficient. This leads to a correct wear distribution for a shorter simulation time. However, every single event with larger amount of local slip will then lead to flat spots, which will cause unrealistic tire vibrations. In order to prevent this from happening, the user should run the 'averaged' version of the wear model, which makes sure the wear is constant in every tread strip (i.e., in circumferential direction). For this case the exact value of the ware rate coefficient does not need to be determined.
2. don't let actual tread wear happen (that is, don't reduce the tread element hight) during a simulation, but only determine the operating conditions-dependent current wear *rate*. If 'only compute input' is selected the wear *rate* during a simulation is written to the mtl/mtb-file and can be used for further analysis. Again, for this approach the exact wear rate coefficient is not required.

Activation and deactivation of the tread abrasion model is described in chapter8.

4.4 Air Volume Vibration Model

FTire comes with an integrated optional **air volume vibration model.** This model describes fluctuations of the air density, air pressure, and air flow velocity inside the tire, mainly caused by the cross sectional area variations, due to the tire deflections of the rolling tire. Due to the unbalanced pressure reaction forces on the rim and tire structure, as well as small pressure-dependent structural stiffness variations, these vibrations might affect the overall tire forces in higher frequency ranges.

The following three equations establish the relationships between the air density $\rho(t,x)$, air pressure $p(t,x)$, and air flow velocity $u(t,x)$ in the circumferential direction. These relationships describe the one-dimensional compressible air flow through the tire volume, driven by the time- and location-dependent cross section area $A\,(t,x)$:

- Impulse conservation:

$$\frac{\partial}{\partial t}\left(A\rho u\right) + \frac{\partial}{\partial x}\left(A\rho u^2\right) = -A\frac{\partial p}{\partial t}$$

- Mass conservation (equation of continuity):

$$\frac{\partial}{\partial t}\left(A\rho\right) + \frac{\partial}{\partial x}\left(A\rho u\right) = 0$$

- Adiabatic pressure/density relationship (for absolute pressure):

$$\frac{p}{p_0} = \left(\frac{\rho}{\rho_0}\right)^{\kappa}$$

with initial and boundary conditions:

$$p(0,x) = p_0$$
$$u(0,x) = 0$$

$$p(t, 2\pi r_{rim}) = p(t,0)$$

$$u(t, 2\pi r_{rim}) = u(t,0)$$

These hyperbolic partial differential equations are timely and spatially discretized in accordance with the tire structure discretization. By this, each belt segment carries two additional state variables: one for the pressure fluctuation and one for the flow velocity. This system of ordinary differential equations, resulting from the spatial discretization, is solved implicitly, synchronized with the solution of the belt's structural equations of motion.

The air volume vibration model does not consume much extra computational power, nor does it require any extra parameters. Activation and deactivation is described in chapter 8.

4.5 Hydroplaning Model

The standard and real-time-capable hydroplaning model predicts the geometry of a pseudo-stationary **water wedge** near the leading edge of the contact patch, caused by water on the road that is not instantaneously displaced by the tire. This water wedge length might vary in lateral direction, and is a function of

- rolling speed
- ground pessure
- waterfilm depth in the respective lateral tread strip
- tread depth.

In the current version, it is assumed that the water wedge length is **proportional** to actual rolling speed and waterfilm depth, but **reciprocal** to ground pressure and tread depth. On basis of detailed CFD studies, this assumption will be refined in a future release.

The model then reduces the local road friction factor in the water wedge region to a very small residual value, which might be even zero. Waterfilm depth is a locally and timely varying value, which can either be specified as input signal or as RGR road attribute.

The hydroplaning model does not consume much extra computational power. Activation and deactivation is described in chapter 8.

4.6 Flexible and Viscoplastic Rim Model

Upon demand, **FTire** can replace the assumed rigid rim geometry by a more realistic flexible and/or viscoplastic rim model. Activation and deactivation of this rim model is described in chapter 8.

The rim model, if activated, takes the distributed spatial forces, exerted by the tire structure on left and right rim flange, and computes the resulting flexible and/or viscoplastic rim flange displacements and deformations. Both force and displacement vectors are expressed in rim-fixed cylindrical coordinates, to take advantage of any axisymmetric or pseudo-axisymmetric properties of the rim stiffness.

The computation of flexible displacement and/or plastic deformation can be done using either the **internal model** or a **user-written external model**.

The **internal model** uses an approximating **Green's function** of the rim stiffness. This function can be viewed to be the result of a **stiffness matrix condensation** and **inversion**, assuming loads are only applied

in the discretized rim flange nodes, and expressing the resulting displacements in terms of the convolution of the forces with Green's function. **FTire**'s internal rim model takes into account both **radial** and **lateral** displacements, but neglects circumferential displacements.

If the rim was perfectly axisymmetric (that is, all stiffness fluctuations caused by drillings, spokes etc. are neglected) and if rim flange forces were all in the linear range (that is, no plasticity applies yet), a Green's function will exist and if will be independent of the circumferential position of the nodes on the rim flanges. In this case, the stiffness condensation results in a simple relationship between the radial or lateral nodal forces (one node per belt segment on either rim flange) $f_i\,(i = 0, .., n_{seg} - 1)$ and the radial or lateral nodal displacements $d_i\,(i = 0, .., n_{seg} - 1)$. For the sake of simplicity, we neglect:

- longitudinal forces/displacements;
- dependency of radial displacements on lateral or longitudinal forces;
- dependency of lateral displacements on radial or longitudinal forces;
- and any coupling between left and right rim flange.

With these simplifications, to be relaxed later, the following relationship holds, independently for lateral and radial displacements/forces:

$$d_i = \sum_{k=-n_{seg}/2}^{n_{seg}/2} g_k \cdot f_{\mathrm{mod}(i+k,n_{seg})} \quad , \; i = 0, \ldots, n_{seg} - 1$$

The weight factors, g_k, are a discretization of the Green's function, and can be determined through FEA, or by processing respective load/deflection measurements. Typically, the weights will have a maximum at $k = 0$, will rapidly decay with $|k|$, and will be symmetric: $g_k = g_{-k}$. The current implementation of the internal rim model will approximately describes these weight factors with a single shape parameter, cf. chapter 8.

In most cases however, caused by **spoke design**, the rim can **not** be considered being perfectly axisymmetric. In this case, Green's function, and by this the weights g_k, will depend on i as well. The current implementation, of the internal rim model, assumes that this dependency approximately can be described by a **higher order harmonic** stiffness modulation:

$$g_{i,k} = \frac{g_{0,k}}{1 + a \cdot \sin\left(n_{spokes} \cdot \varphi_i\right)}$$

where $\varphi_i = 2\pi \frac{i}{n_{seg}}$, the number of rim spokes is denoted by n_{spokes}, and a is a measure of higher order harmonic stiffness fluctuation.

If the elastic deformations locally exceeds a certain threshold value, the rim will undergo **permanent plastic deformation**, and only a part of the elastic deformation will be reversible. These thresholds, one for radial and one for lateral displacement, are part of the **model data**. The permanent deformations, distributed along the rim circumference, are separately saved for the left and right rim flanges and for the lateral and radial directions, as part of the rim **state array**.

As an alternative to the internal model, **FTire** provides a straightforward program interface, to call a **user-written external model.** The user provides this model in terms of a dynamic library, containing a C/C++ function with the following calling syntax:

```
void urim (
  int ti,          /* tire handle (in) */
  int nseg,        /* number of equally distributed nodes on one
                      rim flange (in) */
  double rrim,     /* rim bead radius [m] (in) */
  double wrim,     /* axial rim flanges distance [m] (in) */
  double t,        /* simulation time [s]. Terminate, if t>=1e60 (in) */
  double fl[][3],  /* force array on left rim flange nodes,
                      in cylinder coordinates [N] (in) */
  double fr[][3],  /* force array on right rim flange nodes,
                      in cylinder coordinates [N] (in) */
  double del[][3], /* elastic displacements of left rim flange nodes,
```

```
                        in cylinder coordinates [m] (out) */
  double der[][3], /* elastic displacements of right rim flange nodes,
                        in cylinder coordinates [m] (out) */
  double dpl[][3], /* plastic deformation of left rim flange nodes,
                        in cylinder coordinates [m] (in/out) */
  double dpr[][3], /* plastic displacements of right rim flange nodes,
                        in cylinder coordinates [m] (in/out) */
  int *ret,        /* return code, 0=ok (out) */
  char *file       /* data file name (in) */
) {
  /* add program code to compute del, der, dpl, dpr, and ret here*/
}
```

Selection of the library, the module name within the library, and the data file is done with the CTI functions `ctiLoadRimModel()` and `ctiLoadRimData()`; please refer to the CTI documentation for more information. Default module name is `urim()`, default library name `urim.dll` or `urim.so`, respectively.

5 FTire Data

5.1 Data Files

All the **FTire** model data is contained in the **FTire** data-file. This data file is given either in TeimOrbit syntax, with file-extension `.tir`, or in cosin/io syntax, with extension `.ft`. Examples of such **FTire** data files are provided with the **FTire** distribution package.

The **FTire** model data, listed in such data files, is called **basic** parameters. These **basic** parameters are processed during the initialization(called pre-processing), resulting in the **preprocessed** parameters. These parameters are appended as binary data to the data file. Provided that the **basic** data has not changed, the preprocessed data will be used in the next simulation run and thus skipping the pre-processing.

The CTI interface automatically recognizes whether several wheels of the car share the same **basic** data file. In that case, pre-processing is done only once for all these files.

Data files might contain information which is **not** used by **FTire** itself, like data in the section `[VERTICAL]` in the *TeimOrbit* files. Such information either is required by the calling solver environment, or is included for compatibility with other tire models.

It is highly recommended **not** to edit data files directly, even if the file is saved in plain ASCII format. It is recommended to use **FTire/editor**, which will always perform automatic consistency checks, prevent typos, conveniently gives access to this documentation, and additionally lets you perform all kinds of visualizations, analysis, and processing tasks complementary to simple editing.

5.2 Parameterization Process

A list of experimental data to parameterize **FTire** is proposed in the FTire Parameterization documentation. Processing of these experimental datasets is best done by using **FTire/fit**, a tailored, user-friendly toolbox for measurement-based parameterization of **FTire**. The fitting process, guided by **FTire/fit**, benefits from certain known parameter sensitivities and physical properties of a tire model in general:

- usefulness of certain measurement types
- appropriate sequence in which measurements are evaluated
- appropriate sequence in which parameters are determined
- relevance of certain measurements for certain parameters
- plausible ranges of parameter values
- sensitivity between parameter values and model properties

One important, and recent, experience is about the role of modal data in the parameterization process. They seem to contain less relevant information than static measurements (in contrast to what has been assumed in the early days of **FTire**), and typically they are more laborious to obtain. There is one obvious cause for this lack of relevant information: during the modal measurements, on an unloaded tire, only small amplitudes will be reached. However, these measurements would then be used to parameterize a **FTire** model for load cases with large to extreme deflection values, that is, for a completely different operating condition.

Another experience is the amount of valuable information contained in the footprint bitmaps. The same holds for several kinds of static deflection curves, with and without camber angle, on a flat surface or on certain well-defined obstacles. Moreover, handling properties, like cornering stiffness and pneumatic trail, show a high correlation with certain out-of-plane stiffness data. In many cases, after a thorough analysis of static and steady-state behavior, there remain only few dynamically relevant parameters to be adjusted in order to get a good correlation for the dynamic cleat tests.

The following procedure has been successfully used to parameterize a **FTire** model, and is well supported by **FTire/fit** in all stages.

5.2.1 Preparation of the Identification Process

1. **Create a new data file** with **FTire/estim** (cf. figure 11), by specifying the tire and rim size, load index, speed range, mass, and inflation pressure(s). As a reference tire, use one which is as close to the new tire as possible. **FTire/estim** can be launched via the **FTire/fit** GUI, **specify all drum diameters** and cleat geometries used during the identification and validation process. **FTire/fit** provides a set of example road data files and functions to manage such obstacle-defining files**.**
2. **Specify ('check-in') all static, steady-state, and dynamic measurement files** used in the sequel. If these files are given in the TYDEX file format, a single mouse-click to check them in is sufficient in many cases. **FTire/fit** will automatically recognize the kind of measurement they contain, will determine the constant operating conditions like inflation pressure, wheel load, camber angle, etc., and will save the information on how the validation or identification is to be performed. Moreover, depending on the kind of measurement, it will occasionally extract relevant information like radial, longitudinal, lateral, and torsional stiffness, cornering stiffness, slip stiffness, pneumatic trail, camber thrust, sliding friction, etc., and insert this information in terms of 'nominal data' into the tire data file, or save it elsewhere. If the files are given in any other ASCII format, **FTire/fit** will assist in importing these files and will create TYDEX files out of them. If measurements are only given in terms of scanned images, **FTire/fit** also provides a digitizing tool based on MSPaint, which also assists in creating TYDEX files.
3. **Specify ('check-in') all footprint bitmap files**. **FTire/fit** will automatically calibrate these files and save the relevant information for later validation.
4. **Import or digitize the tread and carcass contour geometry data. FTire/fit** provides a respective digitizing tool.
5. **Identify the dynamic rolling radius** on the basis of the measurement of the angular velocity of a freely rolling tire at different drum speeds and wheel loads (or roughly estimate the rolling radius by subtracting the tread gauge from the maximum radius).

5.2.2 Identification/Validation of Footprint Images

1. **Footprint size and shape** at different wheel loads and camber angles. **FTire/fit** provides an automatic simulation preparation tool and superimposes the simulated footprint boundary over the measured contact patch bitmap. Again, all this is done by a single mouse-click. If there is a mismatch in size or shape, adjust the in-plane and lateral bending stiffness.

5.2.3 Identification/Validation of Static Properties

- **Vertical stiffness on a flat surface** (which is merely a validation of the two deflection values for half and full LI load that have been automatically inserted into the data file in step (5.2.1)). The respective simulation has been prepared by **FTire/fit**; a single mouse-click is sufficient to launch the validation and save all results for later report generation. If the actual stiffness deviates from the predicted one, adjust the respective deflection values. This might happen if there is a discrepancy between the static and the steady-state kind of simulation, caused by different treatment of hysteresis and friction properties, longitudinal and lateral stiffness on a flat surface.
- **Torsional stiffness** (turning the standing tire about the vertical axis). Adjust the belt torsional stiffness about radial axis accordingly, if the simulation deviates from the measurement.
- **Vertical stiffness on a longitudinal and transversal cleat.** Adjust the lateral belt bending stiffness and the belt in-plane bending stiffness accordingly, if the simulation deviates from the measurement.
- **Vertical stiffness at large camber angles on a flat surface and on a transversal cleat.** Adjust the belt torsion and twist stiffness about circumferential axis, if the simulation deviates from the measurement.
- **Longitudinal and lateral stiffness on flat surface.** Adjust the longitudinal and lateral tire stiffness, if the simulations deviate from the measurements.

5.2.4 Identification/Validation of Steady-State Rolling Properties

- **Longitudinal slip stiffness.** Either activate the measured nominal value directly, by replacing tread rubber stiffness, or identify the tread rubber stiffness manually. *Fire/fit* has prepared the validation, provided a respective measurement file is available.
- **Cornering stiffness and pneumatic trail.** Either activate the measured nominal values directly, by replacing the lateral stiffness and out-of-plane bending stiffness, or (re-)identify these two values manually. If there is a discrepancy to the value of the lateral stiffness as determined in section 5.2.3, find a compromise.

5.2.5 Identification/Validation of Friction Characteristics

- **Identify sliding friction coefficients.** During check-in of the measurement files, **FTire/fit** has automatically collected all available and relevant cases. Ideally, this identification is performed by one mouse-click only.

  After the identification, validate the relevant measurement cases. These cases, which have been automatically detected and collected by **FTire/fit**, include:

  1. pulling the tire in longitudinal direction
  2. pulling the tire in lateral direction
  3. turning the tire about vertical axis
  4. rolling at large longitudinal slip
  5. rolling at large slip angle

  If the identified values of stiction and sliding friction differ greatly, stick-slip-phenomena might occur in the lateral and longitudinal stiffness simulations. In this case, find a compromise by relaxing the differences of the friction coefficients.

5.2.6 Identification/Validation of Dynamic Cleat Tests

- **Run a in-plane cleat-test identification** (or just a validation), to determine the last few remaining parameters like the percentage of free mass, the structural damping (expressed in terms of the modal damping), longitudinal coupling of tread shear stiffness, tread rubber damping, etc.
- **Run a out-of-plane cleat-test identification** (or just a validation), to determine few more parameters like conicity, modal out-of-plane damping, the coupling between belt torsion and lateral displacement, etc.

In all the phases listed above, **FTire/fit** will automatically create diagrams showing the comparisons between the simulation results and the corresponding measurement. Finally, with another mouse-click, **FTire/fit** generates a comprehensive report file, containing all these comparisons and more.

Apparently, there are different combinations of parameters possible that all completely determine the structural stiffness and damping properties of **FTire**. The choice of these parameters (irrespective of static, steady-state, or modal nature) will depend on the kind of measurements that are available, or cost effective, or accurate enough. Note that all the modal data is only used to determine the spring stiffnesses and damping coefficients, such that the mathematical model, for small excitations, shows exactly the measured behavior in the frequency domain. **FTire** is not a modal model, nor is it linear.

To facilitate the parameterization, a second tool (**FTire/calc**) is available, which calculates all static and modal data used in **FTire** by means of a detailed FE model. In turn, this FE model only takes geometry data and material properties of all tire components, like the carcass and belt layers, bead rubber, bead wire, tread, and so on, as input. For those users that have access to these tire design data, the new approach might become promising. In the end, it is planned to establish a CAE process chain that takes the tire design data for the prediction, or at least rough estimation, of the tire handling and ride comfort characteristics.

The kernel of **FTire/calc** is a static version of the FE tire model FETire. Using the model, a list of relevant load cases are automatically processed by **FTire/calc**, describing the tire deflections in several directions and

magnitudes, as well as determining the natural frequencies and damping moduli of all relevant modes. The whole process of estimating the structural stiffness data of **FTire** will take only a few minutes on a standard PC.

6 FTire Parameter Specification

6.1 Scheme of tables

The meaning of all parameters in the main data section of a data file is described in the tables below. All tables shown here follow the same scheme:

name in data file	***unit*** *in data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
'software-friendly' name (does **not** contain blank spaces or special characters and is **not** case-sensitive) of the respective parameter, exactly as it appears in the data file	the physical unit as used in data files saved in **cosin/io** format	description of the units in case of *TeimOrbit* format data files, using physical base quantities such as **time**, **length**, **force**, **mass**, and **angle**. The actual units of these base quantities are defined in the `[UNITS]` section of the data file.	Sspecification of a mathematical symbol or formula of the parameter, for use in the subsequent description

6.2 Size, Geometry, Tire Specification, and Tread Pattern

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`rolling_circumference`	mm	$length$	$2\,\pi\,r_{belt}$

This is the rolling circumference of the tire under normal running conditions, at small load (approximately 1/4 rated load) and low speed ($\leqq$1/5 rated speed). This parameter, if specified, is used to enhance the tire-size-based belt radius estimation.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`tire_section_width`	mm	$length$	w_{tire}

This is the **nominal** maximum tire width, during inflated, but unloaded operating conditions. Typically, this value is the first number in the tire dimension string.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`actual_tire_section_width`	mm	$length$	$w_{tire,actual}$

This is the **actually** measured maximum tire width, during inflated, but unloaded operating conditions. By default, actual tire width is the same as nominal tire width.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`tire_aspect_ratio`	%	%	a_{tire}

This is the percentage of tire height to tire width. Typically, this value is the second number in the tire dimension string.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`rim_diameter`	$inch$	$length$	d_{rim}

This is the rim diameter. It is used to estimate the moment of inertia of the 'non-vibrating' parts of the tire (those parts that are assumed to be fixed to the rim), as well as the maximum, possible, tire deflection. Typically, this value is the third number in the tire dimension string.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`rim_contour`	$string$	$string$ in quotes	RCL
`rim_width`	$inch$	$length$	w_{rim}
`wheel_offset_ET`	mm	$length$	ET

These two parameters describe the basic rim geometry.

- w_{rim} is the rim width (distance between the two rim flanges). Among others, it is used as the default value for the belt width, in determining potential rim-to-road contact, and for the computation of **FTire**'s animation model.
- ET (German for 'Einpresstiefe') is the optional wheel offset (distance between rim mid-plane and rim mounting plane). ET is positive if rim mid-plane is closer to the vehicle's center-plane than the rim mounting plane, and negative if not.
- RCL is the rim contour letter as defined and used in ETRTO, currently one out of J, JJ, B, W, DW, TW. Default is J. The rim contour letter mainly specifies certain standardized geometrical values like rim flanges height and width, etc.

name in data file	***unit*** *in* ***cosin/io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***

`load_index`	$integer$	$integer$	LI
`speed_symbol`	$string$	$string$ in quotes	SS
`intended_use`	$string$	$string$ in quotes	u

These three parameters are the standardized codes for the tire's load capacity, maximum speed, and (optional) intended tire use.

- The load index LI ranges from 19 (which means 77kg max. load) to 204 (which means 16000kg max. load). Any value larger than 204 is assumed to be the maximum load in [kg].
- The speed symbol ranges from 'A1' (= 5 $\frac{km}{h}$) to 'Y' (= 300 $\frac{km}{h}$) and 'ZR' (maximum speed 240 $\frac{km}{h}$ and above). Any other numerical value (enclosed with quotes) is assumed to mean the encoded maximum speed in [$\frac{km}{h}$].
- The intended use symbol u is either blank (default), or P (p-metric), LT (light truck), ST (special trailor service), T (temporary spare), or C (commercial).

name in data file	***unit** in **cosin/io** data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`belt_width`	mm	$length$	w_{belt}
`tread_width`	mm	$length$	w_{tread}

w_{belt} is the width of the belt. This parameter is needed to determine the effective belt shape, under lateral bending, and other auxiliary stiffness values.

w_{tread} is the maximum width of the tread that might come into contact with the road surface under extreme running conditions. Among others, it is needed to determine the width of the belt element strips. Moreover, it influences the computation of the longitudinal and lateral stiffness of the discrete tread blocks, that represent the stiffness of the tread area. For this, the following relationships hold:

$$c_{radial} = \frac{P}{100}\frac{\triangle A}{h}E$$

and

$$c_{tangential} = \frac{1}{3}c_{radial}$$

where

$$\triangle A = \frac{2\,\pi\,r_{belt}\,w_{tread}}{n_{seg}\,n_{blocks}}$$

$$h = d_{tread} + d_{tread,0}$$

$$E = E(S) = 10^a \frac{N}{m^2}$$

$$a = 5.33905 + 0.020477 \cdot S$$

and

P	% of positive tread,
r_{belt}	belt radius if tire is not inflated and not loaded, at zero speed,
n_{seg}	number of belt segments
n_{blocks}	number of contact points per segment
d_{tread}	tread depth
$d_{tread,0}$	rubber height over steel belt, at zero tread depth
S	Shore-A hardness of tread rubber, under the operating conditions the model will be used in

name in data file	***unit** in **cosin/io** data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`tread_pattern_file`	*string*	*string* in quotes	
`tread_pattern_xmin`	`0.0 .. 1.0`	`0.0 .. 1.0`	x_{min}
`tread_pattern_xmax`	`0.0 .. 1.0`	`0.0 .. 1.0`	x_{max}
`tread_pattern_ymin`	`0.0 .. 1.0`	`0.0 .. 1.0`	y_{min}
`tread_pattern_ymax`	`0.0 .. 1.0`	`0.0 .. 1.0`	y_{max}
`tread_pattern_variation`	%	%	s_{TP}

The first parameter is the name of a b/w bitmap file which shows all or part of the tread pattern. This file is optional. If specified, it is used by **FTire** to compute the contact elements height accordingly. In doing so, the actual tire's tread pattern will be roughly approximated by **FTire**. Clearly, the accuracy of this approximation depends very much on the number of contact points, and will thus be limited.

Usage of this option is meant mainly for the modelization of tires with relatively large blocks and/or grooves.

The file format of the bitmap file must be `bmp`; file extension is expected to be '`.bmp`'. You might wish to use an image processing software (like GNU Gimp, Mac OS X Paintbrush, Microsoft Paint, Google Picasa, Adobe PhotoShop or any other), to import other image formats like png, tiff, or jpg.

The bitmap file is to be given in one of the standard resolutions (1, 4, 8, 16, 24, or 32 bits per pixel). **FTire** assumes that the part of the tread which comes into contact to the road is shown in black color, while groves appear in white color. Grayscale bitmaps may be used as well, to define a relative tread thickness. Tread element heights are set according to the local brightness. As with purely b/w bitmaps, 'black' means maximal thickness and 'white' means groove or void. Colored bitmaps can be used as well, even if not recommended; only the pixels' brightness will be used.

The bitmap file will automatically be scaled by **FTire** in such a way that the width of the file exactly covers the tread width of the **FTire** model. Moreover, scaling factor in circumferential direction is the same as scaling factor in lateral direction. This kind of scaling ensures there will be no distortion of the approximated tread pattern. If the file does not show the total circumference (normally, it will not), it will be periodically repeated to cover the full length of the tread.

By choosing a '`bounding box`', defined by x_{min}, x_{max}, y_{min}, y_{max}, a certain part of the bitmap can be selected to be repeated periodically in x- and/or y-direction. **These parameters are optional.** Default value for x_{min} and y_{min} is `0.0` (left or lower boundary, respectively); default value for x_{max} and y_{max} is `1.0` (right or upper boundary, respectively).

s_{TP} denotes an optional standard deviation of the tread pattern repetition length in case of tread pattern specification by a bitmap file. Such a stochastic unequal spacing is usually applied to reduce peaks in the rolling noise spectrum.

name in data file	***unit** in **cosin/io** data file*	***unit** in **TeimOrbit** data file*	***symbol***
`tread_pattern_geometry`	*string*	*string* in quotes	

This string variable will define a special tread pattern geometry, and is an alternative to defining the tread pattern by a bitmap file. It starts with a keyword, defining the kind of geometry, followed by a list of additional numerical values. At present, the following special kind of geometries are supported:

- '`grooves` $n\, w\, p_1\, p_2\, \ldots\, p_n$' will define a set of n longitudinal grooves, each with width w $[mm]$. The position of the i-th groove center relative to tread width is given by the percentage p_i. For example, '`grooves 2 8 50 75`' will define two 8 mm grooves, one at the tread center and another one in the middle between tread center and right-hand side tread end. For reasons of numerical accuracy, if defining grooves, the number of tread strips (n_{strips}) should be set large enough. In most cases, 50 tread strips will be sufficient.
- '`rib_edges` $q_1\, q_2\, \ldots\, q_n$' will define a set of $n-1$ longitudinal grooves. Position and width of groove # i is defined through absolute distances from the tread center line with the numbers q_{2i-1} and q_{2i} $[mm]$. For reasons of numerical accuracy, if defining grooves, the number of tread strips (n_{strips}) should be set large enough. If the optimized placement of tread blocks is activated, the number of tread strips should be at last 5*number of grooves + 2. In most cases, 50 tread strips will be sufficient.'
- '`expr` $f(s_x, s_y)$' will define a function expression of the dimensionless independent variables s_x and s_y. $f(s_x, s_y)$ returns values between 0 and 1, specifying the relative tread thickness at circumferential position s_x ($s_x = 0$: 0 deg ... $s_x = 1$: 360 deg) and lateral position s_y ($s_y = 0$: left tread boundary ... $s_y = 1$: right tread boundary). The function expression $f(s_x, s_y)$ can be any arbitrary function as described in cosin/iodocumentation.

name in data file	***unit** in **cosin/io** data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`belt_lat_curvature_radius`	mm	$length$	$r_{belt,lat}$
`rel_tread_shoulder_width`	%	%	$w_{shoulder,rel}$
`rel_min_tread_shoulder_height`	%	%	$h_{shoulder,rel,min}$
`belt_layers_thickness`	mm	$length$	h_{belt}
`inner_liner_thickness`	mm	$length$	h_{il}

These variables describe the geometrical shape of the tread surface and other cross-section data.

- $r_{belt,lat}$ denotes the curvature of the outermost belt layer in lateral direction (that is, perpendicular to the rim mid-plane). The tire is in inflated, but unloaded condition. This radius influences the0 ground pressure distribution in the contact patch, and thus, most of all, the shape of the contact patch, and the aligning torque.
- $w_{shoulder,rel}$ is the width of one tread shoulder relative to the tread width. The tread depth is assumed to vary in a smooth quadratic way between maximum height at the inner start of the tread shoulder, and a final value at the outer end of the shoulder, which coincides with the tread end.
- $h_{shoulder,rel,min}$ is the tread depth at the outer end of the shoulder, measured relative to the maximum tread depth
- h_{belt} is the mean height of all belt layers
- h_{il} is the mean height of the tire inner-liner

All five parameters are optional. Default curvature value is (nearly) infinity, which describes a flat tread surface. Default tread shoulder width is zero, whereas default tread shoulder height is 100%.

Obviously, the correct influence of the tread shoulder geometry on the overall tire properties depends very much on a sufficiently large number of contact elements, properly distributed in longitudinal and lateral direction.

h_{belt} is used to determine the neutral axis, of lateral bending, of the belt segments, if the cross-section is defined into carcass and tread spline data. If not provided, **FTire** assumes that the carcass line does **not** exactly follow the carcass in the belt region, but rather denotes the belt center.

The inner-liner thickness h_{il} is only used together with the automated inner volume computation, performed if the air vibration model is activated.

	name in data file	***unit*** *in* **cosin/*io*** *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
one of	`carcass_contour_spline`	mm	$length$	table of size $2 \times n$
	`carcass_contour_table`	mm	$length$	table of size $2 \times n$
	`carcass_contour_scalable_spline`	mm	$length$	table of size $2 \times n$
	`carcass_contour_scalable_table`	mm	$length$	table of size $2 \times n$
one of	`tread_contour_spline`	mm	$length$	table of size $2 \times n$
	`tread_contour_table`	mm	$length$	table of size $2 \times n$
`degree_cont_smoothing_poly`		-	-	d_{smooth}

These optional spline or table data allow the precise description of the tire's cross section geometry. If defined, they replace $r_{tread,lat}$, $w_{shoulder,rel}$ and $h_{shoulder,rel,min}$. The kind of interpolation to be used is indicated by the respective data item's suffix: `_spline` means spline interpolation, `_table` piecewise linear interpolation. The values of the independent variable in the first table column must be monotonously increasing, but may be chosen non-equidistant.

- **The carcass contour table** specifies arbitrarily many x/y data-points, following **the carcass line** (more precise, the effective **neutral axis of the carcass / belt structure** with respect to bending) both in side-walls and belt region. In most cases, the carcass line will be well approximated by the carcass location in the sidewalls, and the location of the midplane of all belt layers in the belt region.

  Data must begin with the left bead center, and reach either to the tire mid-plane, or to the right bead center. If only one half of the carcass contour is defined, the geometry will be automatically mirrored. x-coordinate is the axial direction, y-coordinate the radial direction. **Begin a new line for each x/y data pair.**

  If necessary, and if the `_scalable` qualifier is specified for the carcass contour, FTire will automatically mirror, shift, and stretch the carcass data points to match exactly the given tire size.

- **The tread contour table** specifies arbitrarily many x/y data-points, following the **tire outer surface line**, disregarding tread grooves.

  Data must begin with the left bead or tread boundary, and reach either to the tire mid-plane, or to the right bead or tread boundary. If only one half of the tread ontour line is defined (cf. above), the geometry will be automatically mirrored. x-coordinate is the axial direction, y-coordinate the radial direction. **Begin a new line for each x/y data pair.**

  If necessary, and if the `_scalable` qualifier is specified for the tread contour, FTire will automatically mirror, shift, and stretch the tread data points in the same way as the carcass data points to match exactly the given tire size.

- If d_{smooth} is greater than zero, the tread contour data will be smoothened by an approximation polynomial of the respective degree. This functionality is provided because the pressure distribution in contact patch is affected very sensitively if contour data are noisy. Accordingly, cross section smoothing might **only** be necessary if any spline or table data are specified for geometry. If, on the other hand, cross section geometry is specified merely by simple shape parameters like belt curvature, shoulder geometry, and tread depth, **further smoothing is not necessary and not recommended**.

Both tables are optional. If not specified, **FTire** will estimate the cross-section geometry on basis of the remaining geometry data.

6.3 Mass, Moments of Inertia, Inflation Pressure, and Volume

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`tire_mass`	kg	$mass$	m_{tire}
`tire_axial_moment_of_inertia`	kgm^2	$mass \cdot length^{22}$	$j_{yy,tire}$
`tire_polar_moment_of_inertia`	kgm^2	$mass \cdot length^2$	$j_{xx,tire} = j_{zz,tire}$

- m_{tire} is the tire's total mass. After subtracting the mass of the belt segments, this value is used to determine (or estimate) the mass of those tire parts that are to be added to the rim mass.
- $j_{yy,tire}$ and $j_{xx,tire} = j_{zz,tire}$ are the tire's total axial and polar moments of inertia. Similar to the tire's mass, both are used to determine the 'rim-fixed' parts of the tire's moments of inertia. These values are optional and will be estimated, on the basis of the tire's mass and geometry, if not explicitly defined.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`inflation_pressure`	bar	bar (or psi for USC units)	p_{meas}
`inflation_pressure_2`	bar	bar (or psi for USC units)	$p_{meas,2}$

p_{meas} denotes the inflation pressure, at which the tire data measurements have been taken. It is used to calculate radial forces acting on the belt nodes. These radial forces generate a membrane tension in the belt, which in turn acts similar to a bending stiffness on the belt nodes. Due to the fact that the pressure forces are compensated by a membrane tension, both in longitudinal and lateral direction, and that the lateral tension does **not** contribute to the in-plane bending stiffness, only a certain **fraction of the true pressure** forces are applied as effective forces. The direction of the pressure force, on a belt node, is calculated as follows: the connecting line of the two neighboring belt nodes is projected onto the rim mid-plane and rotated $90deg$ counter-clockwise about the wheel spin axis. The absolute value of the pressure force is given by:

$$|F_{pressure}| = 0.5 \cdot \frac{2\,\pi\, r_{belt}}{n_{seg}} \cdot w_{tread} \cdot p.$$

During a simulation, $p = p_{act}$ is used. This is one of the operating conditions parameters, and overrides p_{meas}, if specified. During pre-processing, $p = p_{meas}$ is used. p_{act} can vary during a simulation.

The inflation pressure also influences other tire-structural stiffness values. To take this into account, the following simple linear dependencies are used:

$$c_{belt,radial} = \left(0.20 + 0.80\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{belt,radial}$$

$$c_{belt,tang} = \left(0.20 + 0.80\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{belt,tang}$$

$$c_{belt,lat} = \left(0.10 + 0.90\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{belt,lat}$$

$$c_{bend,in} = \left(0.90 + 0.10\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{bend,in}$$

$$c_{bend,out} = \left(0.10 + 0.90\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{bend,out}$$

$$c_{bend,lat} = \left(0.10 + 0.90\,\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{bend,lat}$$

$$c_{tors} = \left(\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{tors}$$

$$c_{tors,rim} = \left(\frac{p_{actual}}{p_{meas}}\right) \overline{c}_{tors,rim}$$

The above values describe:

- the radial, tangential, and lateral stiffness, located between belt nodes and rim;
- the torsional stiffness about circumferential direction, between belt element and rim;
- the torsional stiffness about circumferential direction, between two adjacent belt elements;
- the in-plane additional bending stiffness of the belt;
- the out-of-plane bending stiffness of the belt;
- the bending stiffness of the belt in circumferential direction;
- the torsional stiffness of two adjacent belt elements; and
- the torsional stiffness of the belt element relative to the rim.

The residual values in the above formulae (stiffness values at $p_{actual} = 0$) can be modified by a respective parameter, see below

These dependencies can be described more accurately by performing and pre-processing static and modal tire measurements at a second inflation pressure $p_{meas,2}$. **This value is optional.** If specified, **FTire** will search for the respective measurement data of that pressure. These parameters carry the postfix _p2, see below. After pre-processing these parameters as well, **FTire** will linearly interpolate, for a given actual inflation pressure, between the stiffness and damping values that belong to p_{meas} and $p_{meas,2}$, respectively.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`interior_volume`	m^3	$length^3$	V_{meas}
`volume_gradient`	$\%/mm$	$\%/length$	G_V

V_{meas} is the interior tire volume when the tire is mounted on the rim and inflated with p_{meas}. G_V is the relative volume decrease of a small tire segment when that segment is deflected. By using the relative, instead of the absolute value, the parameter is independent on the number of belt segments. The volume decrease is expressed as a differential rate of change, and for simplicity assumed to be constant with respect to the tire deflection etc.:

$$G_V = -100\,\frac{1}{V_{seg}}\,\frac{\partial V_{seg}}{\partial d_{seg}}$$

(note: per definition, G_V is *positive* if volume *decreases* with deflection).

V_{meas} and G_V are only required if the small pressure increase is to be taken into account that occurs when the tire is deflected. Essentially, it is determined by the geometrical shape of the side-wall deformation. Pressure vs. volume dependency is assumed to be adiabatic (here, the difference between V_{meas} and V_{act} is neglected):

$$p = p_{act}\left(\frac{V_{act}}{V}\right)^{1.4} \approx p_{act}\left(\frac{V_{meas}}{V}\right)^{1.4},$$

$$V = V_{meas} + \sum_{segments}\frac{\partial V_{seg}}{\partial d_{seg}} \cdot d_{seg} = V_{meas} \cdot \left(1 - 0.01 \cdot \frac{G_V}{n_{seg}} \sum_{segments} d_{seg}\right)$$

Both V_{meas} and G_V are optional parameters. Default value is $G_V = 0$. In that case, V_{meas} is of no relevance. Moreover, V_{meas} and G_V will be computed or, more precise, replaced internally and thus do **not** have to be specified if the **air vibration model** is activated or if the **detailed volume computation** flag is set, see below. This computation is done on basis of detailed cross-section geometry data.

6.4 Structural Stiffness, Damping, and Hysteresis

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`first_deflection`	mm	$length$	d_1
`second_deflection`	mm	$length$	d_2
`stat_wheel_load_at_first_defl`	N	$force$	$F_{z,1}$
`stat_wheel_load_at_second_defl`	N	$force$	$F_{z,2}$
`max_radial_progressivity`	%	%	$P_{rad,max}$
`residual_radial_stiffn_perc`	%	%	$P_{rad,residual}$
`initial_radial_stiffness`	N/mm	$force/length$	$c_{rad,init}$

These parameters, together with natural frequency f_2 and actual inflation pressure, largely determine the radial element stiffness $c_{belt,radial}$ between belt nodes and rim, as well as the mass of the belt nodes. They define the total radial stiffness characteristic of the tire and allow that the non-linearity of that characteristic is prescribed in an easy way.

- $F_{z,1}$ is the static wheel load on a flat surface, at zero camber angle, and with an inflation pressure p_{meas}, if tire is deflected by d_1 mm;
- $F_{z,2}$ is the static wheel load under the same conditions, if tire is deflected by d_2 mm;
- $P_{rad,max}$ defines the maximum slope increase (or decrease) of a segment's radial stiffness characteristic, compared to the slope at zero deflection, in terms of a percentage;
- $P_{rad,residual}$ defines the (approximate) residual structural radial stiffness at zero inflation pressure as percentage of radial stiffness at inflation pressure p_{meas}. The default residual values of all other structural stiffness data will be modified accordingly if this data item is specified;
- $c_{rad,init}$ is **optional** and an alternative to $F_{z,1}$ and $F_{z,2}$. It defines the initial slope of the radial stiffness characteristic, and is used internally to find appropriate values for $F_{z,1}$ and $F_{z,2}$.

d_2 **and** $F_{z,2}$ **are optional and can be omitted.** In that case, the tire is assumed to have a slightly progressive radial characteristic; respective data are inserted in the data file if edited using **cosin/tools**. 'Deflection' is defined to be the vertical wheel displacement, starting with value zero when tire first touches ground.

$P_{rad,max}$ **is optional as well, its default value is 85 %.** If **FTire** does not succeed to match the specified wheel load characteristic during pre-processing, this value might be increased.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`rad_dynamic_stiffening`	%	%	$S_{dyn.rad}$
`tang_dynamic_stiffening`	%	%	$S_{dyn,tang}$
`lat_dynamic_stiffening`	%	%	$S_{dyn,lat}$
`time_const_dynamic_stiffening`	s	$time$	T_{dyn}

To describe the dynamic tire stiffening, a series of spring-damper connections are placed between each belt node and rim, in radial, tangential (longitudinal), and lateral direction. Stiffness and damping coefficient of these series connections are chosen to match the following conditions. The respective stiffness asymptotically reaches a final value for very large speeds. S_{dyn} defines the percentage of increase of the final stiffness, as compared to the stiffness at zero speed. T_{dyn} is the time constant of the equivalent low-pass filter established by the spring damper series connection; it is quotient of damper coefficient and spring stiffness.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`radial_hysteretic_stiffening`	%	%	$S_{hyst,r}$
`radial_hysteresis_force`	N	$force$	$F_{hyst,r}$
`tang_hysteretic_stiffening`	%	%	$S_{hyst,t}$
`tang_hysteresis_force`	N	$force$	$F_{hyst,t}$
`lat_hysteretic_stiffening`	%	%	$S_{hyst,l}$
`lat_hysteresis_force`	N	$force$	$F_{hyst,l}$

The data above describes the structural friction properties which cause the hysteresis loops in radial and tangential stiffness characteristics. In **FTire**, this kind of friction is described by two additional nonlinear force elements between each belt node, connecting the node in radial and tangential direction to the rim. Both force elements are given by the series connection of a spring and a dry sliding friction element.

$S_{hyst,r}$, $S_{hyst,t}$, and $S_{hyst,l}$ are the percentage of short-term stiffness increase when changing the deflection direction, in radial, tangential (longitudinal), and lateral direction, respectively.

$F_{hyst,r}$, $F_{hyst,t}$, and $F_{hyst,l}$ describe the maximum force width of the radial, tangential (longitudinal), and lateral hysteresis loops, respectively. This width depends on the tire deflection. Here, that deflection value is considered where $\frac{1}{12}^{th} = 30\ deg$ of the tread circumference is in contact with the road surface. For a typical passenger car tire, this corresponds to a foot-print length of about 160 mm.

To describe the radial hysteresis characteristic of the tire, five dry friction(hysteresis) elements are assembled in parallel. The tangential hysteresis characteristic is described by a single dry friction(hysteresis) element.

All hysteresis data are optional. If not specified, zero hysteretic friction is assumed.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`belt_extension_due_to_vmax`	%	%	$e_{belt,vmax}$
`belt_rad_displ_shorten_coupl`	-	-	$f_{belt,shorten}$
`belt_extension_damp`	%	%	$d_{belt,ext}$

$e_{belt,vmax}$ is the percentage of centrifugal-forces-induced rolling circumference growth at maximum (rated) speed, as compared to stand-still (both for an inflated tire). This parameter largely correlates with the stiffness of the longitudinal springs that are placed between two consecutive belt nodes. In older versions, a similar parameter was called `belt_extension_at_vmax` or `belt_extension_at_max_speed`, but its value was, by mistake, only one half of `belt_extension_due_to_vmax`.

$f_{belt,shorten}$ is a dimensionless factor, describing the reduction of effective belt length due to local flattening and other nonlinear effects near the contact patch. This reduction is the main reason for the decrease of the rolling circumference due to an increase in wheel load.

$d_{belt,ext}$ is the damping percentage of the belt extensibility as expressed by $e_{belt,vmax}$. This parameter is available only in versions 2022-1 and later. In previous versions, its value was hard-coded to 2 %, which still is its default value. As with all other damping parameters, $d_{belt,ext}$ is one of the main influence factors to rolling resistance.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`rel_long_belt_memb_tension`	%	%	$m_{belt,long}$
`rel_long_belt_memb_tension_red`	%	%	$m_{belt,long,red}$

$m_{belt,long}$ is the percentage by which inflation pressure forces, in the belt region, are compensated for by a membrane tension in the longitudinal direction, as compared to the total compensation in lateral and longitudinal direction. $m_{belt,long}$ is optional and can only be calculated by using an FE (Finite Element) model, or estimated by parameter identification. A value of 70 to 80 % is be appropriate for many tires.

$m_{belt,long,red}$ is the reduction percentage of $m_{belt,long}$ in case the respective belt portion is deflected such that is completely flat.

	name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
one of	`third_deflection`	mm	$length$	d_3
	`wheel_load_long_stiffn`	N	$force$	$F_{z,3}$
`tire_long_stiffn`		$\frac{N}{mm}$	$\frac{force}{length}$	c_{long}
`tire_long_stiffn_progr`		%	%	P_{long}
`tire_lat_stiffn`		$\frac{N}{mm}$	$\frac{force}{length}$	c_{lat}
`tire_lat_stiffn_progr`		%	%	P_{lat}
`tire_tors_stiffn`		$\frac{Nm}{deg}$	$\frac{force \cdot length}{angle}$	c_{tors}
one of	`fourth_deflection`	mm	$length$	d_4
	`wheel_load_slip_stiffn`	N	$force$	$F_{z,4}$
one of	`fifth_deflection`	mm	$length$	d_5
	`wheel_load_cornering_stiffn`	N	$force$	$F_{z,5}$
	`wheel_load_camber_stiffn`	N	$force$	$F_{z,5}$
`cornering_stiffn`		$\frac{N}{deg}$	$\frac{force}{angle}$	c_{corn}
`pneumatic_trail`		mm	$length$	s_{pneum}
`camber_stiffn`		$\frac{N}{deg}$	$\frac{force}{angle}$	c_{cam}

These numbers contribute to the determination of the structural stiffness of steel belt and side-wall.

- d_3 is the tire radial deflection value applied during the measurement of the subsequent longitudinal, lateral, and torsional stiffness values. Default value is d_1.
- c_{long} is the longitudinal stiffness of the tire, which is measured by moving the loaded tire a very small distance in forward direction, while the brake is actuated. Vertical deflection is assumed to be d_3.
- P_{long} is the percentual increase of the longitudinal stiffness at a longitudinal displacement d_1. If this value is negative, the longitudinal stiffness is degressive.
- c_{lat} is the lateral stiffness of the tire, which is measured by moving the loaded tire a small distance in the lateral direction. Vertical deflection is assumed to be d_3.
- P_{lat} is the percentual increase of the lateral stiffness at a lateral displacement d_1. If value is negative, the lateral stiffness is degressive.
- c_{tors} is the torsional stiffness of the tire, which is measured by turning the loaded tire a very small angle about vertical axis. Vertical deflection is assumed to be d_3.
- d_4 is the tire radial deflection value (or $F_{z,4}$ the static wheel load), applied during the measurement of the slip stiffness value (see below). Default value is d_1.
- d_5 is the tire radial deflection value (or $F_{z,5}$ the static wheel load), applied during the measurement of the cornering stiffness, camber stiffness, and pneumatic trail values. Default value is d_4.

- c_{corn} (cornering stiffness) is the rate of change of the side force with slip angle at very small slip values. Vertical deflection is assumed to be d_5 (or vertical load is assumed to be $F_{z,5}$), and the camber angle is 0.
- s_{pneum} (pneumatic trail) is the rate of change of the self-aligning torque with the side force, at very small slip values. Vertical deflection is assumed to be d_5 (or vertical load is assumed to be $F_{z,5}$), and the camber angle is 0.
- c_{cam} (camber stiffness) is the rate of change of the side force with camber angle at very small camber values. Vertical deflection is assumed to be d_5(or vertical load is assumed to be $F_{z,5}$, respectively), and both slip values are 0.

These parameters are optional. Use at most two out of c_{lat}, c_{tors}, c_{corn}, s_{pneum}, c_{cam}. Alternatively, the structural stiffness can be defined by specifying the appropriate natural frequencies.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`free_mass_percentage`	%	%	$P_{m,free}$

$P_{m,free}$ is the percentage of the total tire mass which is spread over the belt nodes. This parameter is optional. Alternatively, the mass can be given implicitly by specifying at least one natural frequency.

Please note: the difference between total tire mass and free tire mass, together with the respective axial and radial moments of inertia, are to be added to the rim part in the calling MBS model. When loading an **FTire** data file, the respective values will be displayed on the screen, and will be written to the message file.

name in data file		***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`f1`		Hz	Hz	f_1
one of	`f2`	Hz	Hz	f_2
	`f2_relative`	%	%	$f_{2,rel}$
one of	`f4`	Hz	Hz	f_4
	`f4_relative`	%	%	$f_{4,rel}$
`D1`		-	-	D_1
`D2`		-	-	D_2
`D4`		-	-	D_4

These numbers describe the rigid-body modes of the tire (see figure 6.1), when the rim is totally fixed, and the tire is inflated with p_{meas} but has no contact to the ground.

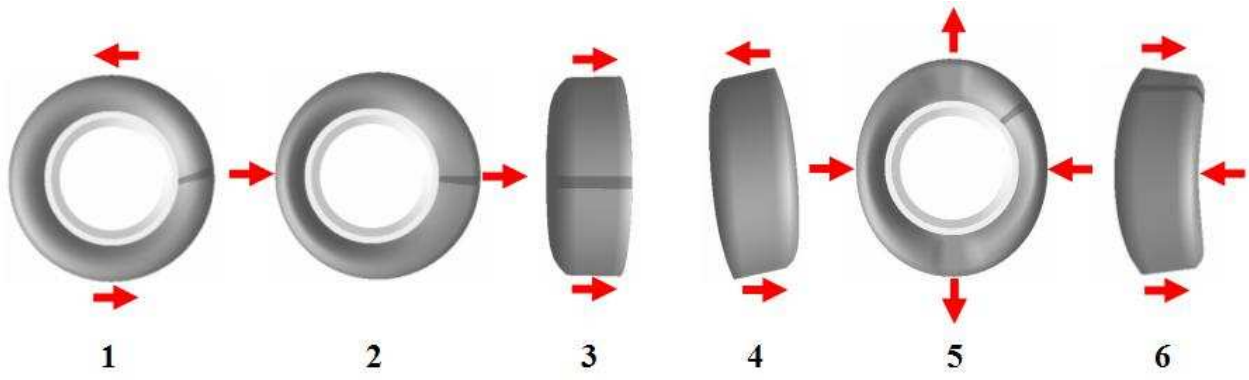

Figure 6.1: First six vibration modes of an unloaded tire with fixed rim

f_1 is the natural frequency of the in-plane 'rigid-body' rotation mode about wheel spin axis, f_2 is the natural frequency of the 'rigid-body' movement mode in longitudinal or vertical direction, and f_4 is the natural frequency of the out-of-plane 'rigid-body' rotation mode about any axis perpendicular to wheel spin axis (cf. figure 6.1). $f_{2,rel}$ and $f_{4,rel}$ are f_2 or f_4 , respectively, expressed as a percentage of f_1.

D_1, D_2, and D_4 are the respective modal damping values, expressed in absolute numbers between 0 (= completely undamped case) and 1 (= aperiodic limit case). Frequently, modal damping is expressed in percentage of the aperiodic damping. To use these values here, they have to be divided by 100.

If D_1 is not specified, the default value 0.05 (= 5 %) is set. If D_2 or D_4 is not specified, the value of D_1 is set instead.

The actual implementation of **FTire** uses viscous damping elements between belt nodes and rim to match these modal values. Due to the fact that rubber damping is more accurately described by frequency-independent hysteresis cycles, this damping model is of limited accuracy. Damping tends to be too small for low-frequent excitation. This is why for some conditions a better coincidence between measurement and model can be gained by increasing the model damping values above those obtained by a modal analysis. In future, it is planned to incorporate a more accurate time-domain description of material damping into the model.

These modal parameters are optional. Alternatively, the tire's structural stiffness can be defined by specifying static tire stiffness data together with the free vibrating mass, see above.

If available, the use of static data rather than eigenfrequencies is preferred.

	name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
one of	`f5`	Hz	Hz	f_5
	`f5_relative`	N	$force$	$f_{5,rel}$
	`belt_in_plane_bend_stiffn`	Nm^2		$C_{bend,in}$
	`Fz_decr_trans_cleat`	%	%	$\triangle F_{z,transcleat}$
`cleat_width`		mm	$length$	w_{cleat}
`cleat_bevel_edge_width`		mm	$length$	w_{bev}

As explained above, the in-plane bending mode is largely influenced by the belt tension, which in turn is a consequence of inflation pressure. To accurately adjust the respective natural frequency, an additional in-plane bending stiffness is provided in **FTire**, see figure 6.2. This torsional stiffness connects consecutive belt nodes with a torsional spring, that results in a torque about the direction of wheel spin axis.

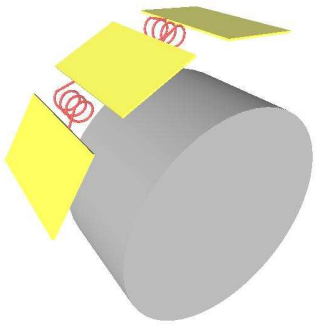

Figure 6.2: Illustration of the In-Plane Bending Stiffness - (this figure only shows a visualization of the stiffness placement; the actual implementation is slightly different)

The stiffness of the torsional spring is largely influenced by the following parameters. One of the parameters must be given:

- f_5 is the natural frequency of the first in-plane bending mode (cf. figure. 6.1)

- $f_{5,rel}$ is f_5, expressed as a percentage of f_1
- $\triangle F_{z,transcleat}$ denotes the percentage decrease of the static wheel load, when the tire is deflected by d_1 on a cleat, instead of a flat surface. The cleat is oriented in a transversal direction. The geometry of the cleat is defined through w_{cleat} (default value $20mm$) and w_{bev} (default value 0mm). If w_{cleat} is entered with a negative sign, a cleat with semicircular cross-section is used. $-w_{cleat}$ is the semicircle diameter in this case.
- $\overline{c}_{bend,in}$ is the direct measure for the bending stiffness of the belt/sidewall structure (about a lateral axis, parallel to the wheel spin axis), if the tire is not inflated.

$\overline{c}_{bend,in}$ is defined in such a way that it is **independent** on the number of belt nodes. Formally, it has the meaning and dimension of

$$\overline{c}_{bend,in} = EI$$

where E is the effective Young's modulus of elasticity (with dimension force/length²), and I the effective axial moment of inertia (with dimension length⁴). But this description might be misleading. Spatial geometry and material compound of the tire structure is so complex that these effective values are difficult to identify. They are not the respective values of the steel belt only. Rather, the value will have to be:

- roughly estimated (sometimes, it can be just set to zero), or
- identified, by using time histories of wheel load and longitudinal force when rolling over cleats, or
- identified, by using geometry data of the contact patch for different wheel loads and inflation pressures, or
- replaced by f_5 or $\triangle F_{z,transcleat}$, or
- calculated, by using FTire/calc or another detailed finite element model.

Unfortunately, experiments show that the bending stiffness is normally extremely dependent on f_5 . Using $\triangle F_{z,transcleat}$ instead might be troublesome occasionally as well. When the tire is deflected on a short cleat, the contact patch is much smaller, compared to the contact patch on a flat surface. This results in a much larger deflection of the tread rubber. So, the difference in radial stiffness does not only depend on differences in structural tire stiffness, being the underlying idea of this approach.

If more than one of the parameters, to define in-plane bending stiffness, is given, they constitute an over-determined system of equations for some of **FTire**'s internal stiffness values. **FTire** will try to find a compromise. Users can control this compromise by setting certain weight values, see below.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol***
`weight_f5`	-	$factor$	
`weight_in_plane_bend_stiffn`	-	$factor$	
`weight_Fz_trans_cleat`	-	$factor$	

If at least two of the above parameters, to define the in-plane bending stiffness, are given, they would constitute an over-determined system of equations that are used to determine **FTire**'s internal stiffness values. To avoid this, **FTire** makes use of certain internally defined precedence rules. If a weight is set to zero, the respective parameter is ignored in any case. By setting the weight to 1, the parameter selection is enabled in that case the respective precedence rule applies. If it is set to 20 or 02, the parameter usage is only enabled for the first or second inflation pressure, respectively.

These parameters are optional. Default values are all 1.

	name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***

Document Revision: 2022-1-r26108

one of	f6	Hz	Hz	f_6
	f6_relative	%	%	$f_{6,rel}$
	belt_out_of_plane_bend_stiffn	Nm	$force \cdot length^2$	$\overline{c}_{bend,out}$

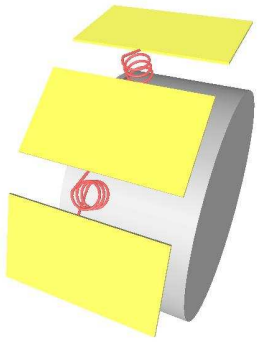

Figure 6.3: Illustration of the Out-of-Plane Bending Stiffness - (this figure only shows a visualization of the stiffness placement; the actual implementation is done differently)

All three data items in the table above determine the out-of-plane bending stiffness of the belt. Only one of them is required:

- f_6 is the natural frequency of the first out-of-plane bending mode (cf. figure. 6.1),
- $f_{6,rel}$ is f_6, expressed as a percentage of f_1, and
- $\overline{c}_{bend,out}$ the direct measure for the bending stiffness of the belt/sidewall structure (about the radial axis), if the tire is inflated (this is in contrast to the definition of $\overline{c}_{bend,in}$, where the tire is **not** inflated).

These parameters show similar properties as the above parameters that define the in-plane bending stiffness. Fortunately, out-of-plane bending stiffness has only very limited influence to many comfort-relevant simulations. For example, if the road excitation is completely symmetric with respect to the lateral direction, if camber and slip angle are zero, and if no tire imperfections are taken into account, the out-of-plane bending stiffness has no influence at all.

For an identification of the out-of-plane bending stiffness, it is advantageous to use steady state side-force and aligning torque characteristics. Bending stiffness has a strong impact on several aspects of the shape of these characteristics.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol***
weight_f6	-	$factor$	
weight_out_of_plane_bend_st	-	$factor$	

If both parameters above (f6 and belt_out_of_plane_bend_stiffn) are used to define the out-of-plane bending stiffness, they would constitute an over-determined system of equations for the respective **FTire**'s internal stiffness values. To avoid this, **FTire** makes use of certain internally defined precedence rules. If a weight is set to zero, the respective parameter is ignored in any case. By setting the weight to 1, the parameter selection is enabled and the respective precedence rule applies. If it is set to 20 or 02, the parameter usage is only enabled for the first or second inflation pressure, respectively.

These parameters are optional, default value is 1.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`belt_torsion_stiffn`	$\frac{N}{deg}$	$\frac{force}{angle}$	$\overline{c}_{tors}$
`belt_twist_stiffn`	$\frac{Nm^2}{angle}$	$\frac{force \cdot length^2}{angle}$	$\overline{c}_{twist}$
`belt_torsion_twist_damp`	%	%	D_{tors}
`Fz_decr_6deg_cam`	%	%	$\triangle F_{z,camber}$
`Fz_decr_trans_cleat_6deg_cam`	%	%	$\triangle F_{z,camber,transcleat}$
`belt_rad_torsion_stiffn`	$\frac{N}{deg}$	$\frac{force}{angle}$	$\overline{c}_{tors,rad}$

These six variables describe the additional properties of the belt stiffness and damping which are only relevant for out-of-plane excitations.

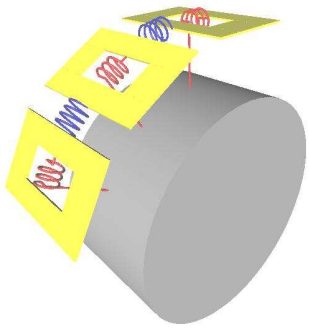

Figure 6.4: Illustration of Belt Circumferential Torsion and Twist Stiffness red: 'torsion' stiffness; blue: 'twist' stiffness

After pre-processing, $\overline{c}_{tors}$, results in a torsional stiffness (red spring in figure 6.3), which is placed between all belt segments and the rim. The value of $\overline{c}_{tors}$ is independent of the number of belt segments. It is defined to be the reaction torque, in circumferential direction, if a constant torsion angle of 1 deg is applied to a unit length belt section of the inflated tire.

In addition, after pre-processing, $\overline{c}_{twist}$ results in a second torsional stiffness, placed between any two adjacent belt segments about an axis in circumferential direction (blue spring in figure 6.4). Again, the value of $\overline{c}_{twist}$ does not depend on the number of belt segments. It is defined to be the reaction torque, in longitudinal direction, when applying a twist angle of 1 deg along a unit length belt segment of the inflated tire, and if the mean torsion angle relative to rim is 0. Unit is $\frac{torque*length}{angle} = \frac{force*length^2}{angle}$.

Both $\overline{c}_{tors}$ and $\overline{c}_{twist}$ might be difficult to determine. For this reason, the two values $\triangle F_{z,camber}$ and $\triangle F_{z,camber,transcleat}$ can be used instead. They are defined to be the percentage decrease in static wheel load, when the tire is deflected with camber angle, or with camber angle on a transversal cleat, respectively, as compared to deflection without camber angle on flat surface.

D_{tors} is the percentual belt segment damping, as compared to the aperiodic limit case, when a rotation about the circumferential axis is excited.

Parameter $\overline{c}_{tors,rad}$ introduces, after pre-processing, an additional torsional spring in radial direction, between each belt segment and the rim.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol***
`weight_belt_twist_stiffn`	-	$factor$	
`weight_belt_torsion_stiffn`	-	$factor$	
`weight_Fz_cam`	-	$factor$	
`weight_Fz_trans_cleat_cam`	-	$factor$	

If more than two of the parameters above are given, to define the belt twist and torsion stiffness, they would constitute an over-determined system of equations for the respective **FTire**'s internal stiffness values. To avoid this, **FTire** makes use of certain internally defined precedence rules. If a weight is set to zero, however, the respective parameter is ignored in any case. By setting the weight to 1, the parameter selection is enabled in case the respective precedence rule applies. If it is set to 20 or 02, the parameter usage is only enabled for the first or second inflation pressure, respectively.

These parameters are optional. Default value is 1.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`belt_lat_bend_stiffn`	Nm^2	$force \cdot length^2$	$\overline{c}_{bend,lat}$
`rel_belt_edge_width`	%	%	$w_{beltedge}$
`rel_belt_edge_lat_bend_stiffn`	%	%	$P_{bend,lat,edge}$
`belt_lat_bend_damp`	s	$time$	$\tau_{bend,lat}$
`Fz_decr_long_cleat`	%	%	$\triangle F_{z,longcleat}$
`belt_lat_bend_stiffn_long_coupl`	-	$factor$	$f_{bend,lat,long}$
`belt_lat_bend_stiffn_rad_coupl`	-	$factor$	$f_{bend,lat,rad}$

These variables describe the bending stiffness of belt elements about the circumferential direction.

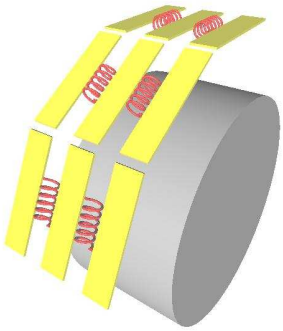

Figure 6.5: Illustration of the Belt Lateral Bending Stiffness (this figure only shows a visualization of the stiffness placement; the actual implementation is slightly different)

Internally, the lateral bending of a belt element is treated as additional degrees of freedom of the belt element, described in terms of the coefficients of certain shape functions. The lateral bending stiffness affects the pressure distribution in lateral direction. If the stiffness is small, the ground pressure on a flat surface will be nearly constant, when measured in the lateral direction. The larger the bending stiffness is, the greater the pressure distribution will be affected by $\overline{c}_{bend,lat}$.

$w_{beltedge}$ is the percentual width, relative to belt width, of both the left and right belt portion supported by edge covers or otherwise modified in effective lateral bending stiffness. For example, value 50% means both in left and right half of the belt (that is, in the complete belt) bending stiffness is modified.

$P_{bend,lat,edge}$ is the percentual lateral bending stiffness in the belt edges, relative to $\overline{c}_{bend,lat}$.

$\tau_{bend,lat}$ is the respective damping parameter, measured relative to $\overline{c}_{bend,lat}$.

$\triangle F_{z,longcleat}$ is the percentage decrease, in static wheel load of the inflated tire, when it is deflected by 10 mm, with zero camber angle, on a longitudinally orientated cleat as specified above, compared to the static wheel load on a flat surface. The cleat must be high enough so that the tire only touches the cleat. The cleat is oriented in longitudinal direction, along the foot-print centerline.

These parameters are optional. Default value for the bending stiffness is (nearly) infinity.

If both parameters are used to define the bending stiffness, they constitute an over-determined system of equations for the respective **FTire**'s internal stiffness values. **FTire** will try to find a compromise. Users can control this compromise by setting certain weight values, see below.

$f_{bend,lat,long}$ is an additional correction factor which can be used to increase or decrease the assumed coupling of the lateral belt bending in the longitudinal direction. Default value is 1.

The factor $f_{bend,lat,rad}$ controls the degree of independence of a segment's lateral belt bending to its radial deflection. Value 0 (default value, 'no coupling') means the effective segment deflection is independent of the lateral bending state, whereas value 1 means 'complete coupling'. In the latter case, the total radial stiffness is assumed to be located in the side-walls, whereas the first case is equivalent to a radial 'volume' stiffness.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol***
`weight_lat_bend_stiffn`	-	$factor$	
`weight_Fz_long_cleat`	-	$factor$	

If both parameters above (`belt_lat_bend_stiffn` and `Fz_decr_long_cleat`) are used to define the lateral bending stiffness, they would constitute an over-determined system of equations for the respective **FTire**'s internal stiffness values. To avoid this, **FTire** makes use of certain internally defined precedence rules. If a weight is set to zero, however, the respective parameter is ignored in any case. By setting the weight to 1, the parameter selection is enabled in case the respective precedence rule applies. If it is set to 20 or 02, the parameter usage is only enabled for the first or second inflation pressure, respectively.

These parameters are optional. Default value is 1.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`belt_lat_bend_stiffn_progr`	%	%	$f_{bend,lat,prog}$

This data item describes an increase in lateral bending stiffness of the belt sections in an unloaded condition, as compared to the loaded condition. Starting in an unloaded condition, the bending stiffness will decrease linearly with a decreasing belt curvature, until the total mean belt curvature reaches zero. From this point on, the stiffness will be held constant.

This behavior accounts for the geometrical de-stiffening due to the shell buckling effect.

This parameter is optional. Default value is 0.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`belt_torsion_lat_displ_coupl`	$\frac{deg}{mm}$	$\frac{angle}{length}$	$\lambda_{tors,lat}$
`belt_torsion_oop_bend_coupl`	-	$factor$	$f_{tors,bend}$

These parameters describe certain 'kinematic couplings' of the belt motion, which are influencing the distorted belt shapes.

- $\lambda_{tors,lat}$ describes the dependency between the torsion of a belt segment, and the respective lateral displacement. Reason for that effect can be the difference in the distance between the two rim flanges, and the width of the belt. Such a difference makes a belt segment move like a general four-bar mechanism.

  If a belt segment's torsion angle leads to a lateral displacement (exactly this effect is described by $\lambda_{tors,lat}$), then there will be a certain dependency on the belt torsion about the circumferential axis, and

the contact patch curvature. The contact patch curvature, in turn, strongly affects the side force and aligning torque. Thus, $\lambda_{tors,lat}$ influences the dependency between the camber angle and the side force. This fact can be used to determine $\lambda_{tors,lat}$.

The exact definition of $\lambda_{tors,lat}$ is as follows: it is a belt segments' torsion angle at 1 mm lateral displacement, if the belt is not twisted (that is, there are no torsion angles relative to adjacent segments), and if no lateral force is applied to the segment.

The parameter is optional. Default value is 0. The value can be computed by FTire/calc.

- $f_{tors,bend}$, on the other hand, defines up to what amount the in-plane and out-of- plane bending stiffnesses are rotated with belt element torsion. A value 0 means 'no rotation'. whereas 1 would mean 'full rotation'. The most plausible and recommended choice from the mechanical point of view is 1. For reasons of downward compatibility with older versions, **the default value of this optional parameter is 0**.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol***
`stat_wheel_load_at_first_d_p2`	N	$force$	
`stat_wheel_load_at_second_d_p2`	N	$force$	
`tire_long_stiffn_p2`	$\frac{N}{mm}$	$\frac{force}{length}$	
`tire_lat_stiffn_p2`	$\frac{N}{mm}$	$\frac{force}{length}$	
`tire_tors_stiffn_p2`	$\frac{Nm}{deg}$	$\frac{force \cdot length}{angle}$	
`f1_p2`	Hz	Hz	
`f2_p2`	Hz	Hz	
`f2_relative_p2`	%	%	
`f4_p2`	Hz	Hz	
`f4_relative_p2`	%	%	
`f5_p2`	Hz	Hz	
`f5_relative_p2`	%	%	
`f6_p2`	Hz	Hz	
`f6_relative_p2`	%	%	
`D1_p2`	-	-	
`D2_p2`	-	-	
`D4_p2`	-	-	
`belt_in_plane_bend_st_p2`	Nm^2	$force \cdot length^2$	
`belt_out_of_plane_bend_st_p2`	Nm^2	$force \cdot length^2$	
`belt_lat_bend_st_p2`	Nm	$force \cdot length$	
`belt_twist_st_p2`	$\frac{Nm^2}{deg}$	$\frac{force \cdot length^2}{angle}$	
`belt_torsion_st_p2`	$\frac{N}{deg}$	$\frac{force}{angle}$	
`Fz_decr_trans_cleat_p2`	%	%	
`Fz_decr_long_cleat_p2`	%	%	
`Fz_decr_6deg_cleat_p2`	%	%	
`Fz_decr_trans_cleat_6deg_cam_p2`	%	%	
`cornering_stiffn_p2`	$\frac{N}{deg}$	$\frac{force}{angle}$	
`pneumatic_trail_p2`	mm	$length$	
`camber_stiffn_p2`	$\frac{N}{deg}$	$\frac{force}{angle}$	

If measurements for a second inflation pressure (`inflation_pressure_2`) are available, the above parameters are defined in the data file. The meaning of the parameters are the same as described for the first pressure value above.

- `stat_wheel_load_at_first_defl`

- stat_wheel_load_at_second_defl
- tire_long_stiffn
- tire_lat_stiffn
- tire_tors_stiffn
- f1
- f2
- f2_relative
- f4
- f4_relative
- f5
- f5_relative
- f6
- f6_relative
- D1
- D2
- D4
- belt_in_plane_bend_stiffn
- belt_out_of_plane_bend_stiffn
- belt_lat_bend_stiffn
- belt_twist_stiffn
- belt_torsion_stiffn
- Fz_decr_trans_cleat
- Fz_decr_long_cleat
- Fz_decr_6deg_cam
- Fz_decr_trans_cleat_6deg_cam
- cornering_stiffn
- pneumatic_trail
- camber_stiffn

All these parameters are optional.

	name in data file	***unit*** *in cosin/io data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
one of	struct_stiffn_vs_temp_spline	$degC$, -	$degC$, -	table of size $2xi$
	struct_stiffn_vs_temp_table	$degC$, -	$degC$, -	table of size $2xi$

These optional spline or table data, respectively, define a temperature-dependent correction factor, applied to all the tire structure stiffness values, for both inflation pressures. By this correction factor, a potential loss in tire structural stiffness at higher temperatures can be taken in to account.

Please note that both the indirect effect, of structure stiffness changes due to temperature induced inflation pressure changes, as well as tread stiffness dependency on temperature is taken into account by **other** parameters. Both effects do not require the specification of the above-mentioned spline or table data.

6.5 Tread Thickness, Stiffness, Damping, and Friction

(see also 6.2 for specification of detailed tread pattern geometry)

name in data file		***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`tread_depth`		mm	$length$	$d_{tread,new}$
`tread_depth_at_vert_statics`		mm	$length$	$d_{tread,vstat}$
`tread_depth_at_horiz_statics`		mm	$length$	$d_{tread,hstat}$
`tread_base_height`		mm	$length$	$d_{tread,0}$
one of	`stiffness_tread_rubber`	$ShoreA$	$ShoreA$	S
	`Youngs_mod_tread_rubber`	$\frac{N}{mm^2}$	$\frac{force}{length^2}$	E_{tread}
	`slip_stiffn`	$N/\%$	$force/\%$	c_{slip}
`stiffn_progr_tread_rubber`		%	%	ΔE_{tread}
`tread_positive`		-	-	P
`tread_pattern_shape_factor_tang`		-	-	$f_{tread,shape,tang}$
`tread_pattern_shape_factor_long`		-	-	$f_{tread,shape,long}$
`tread_pattern_shape_factor_tors`		-	-	$f_{tread,shape,tors}$
`lat_to_long_tread_stiffn_ratio`		-	-	$R_{tread,aniso}$
`sidewall_to_tread_stiffn_ratio`		-	-	R_{sw}

$d_{tread,new}$ is the tread depth, which is understood as the maximum groove depth of the tread. $d_{tread,new}$ is assumed to be the value of the tread depth of a new tire. During a simulation, the value can easily be modified by using an actual 'operating condition' value $d_{tread,act}$ instead.

$d_{tread,vstat}$ is the actual tread depth during the vertical statics measurements (i.e. radial stiffness on flat surface and cleats). By default, $d_{tread,vstat} = d_{tread,new}$.

$d_{tread,hstat}$ is the actual tread depth during the horizontal statics measurements (i.e. longitudinal stiffness, lateral stiffness, and torsional stiffness). By default, $d_{tread,hstat} = d_{tread,vstat}$.

$d_{tread,0}$ is the height of rubber outside the steel belt (sometimes referred to as 'tread base'), if the tread depth is zero. The value coincides with the distance between steel belt and greatest depth of the tread grooves.

Both $d_{tread,meas}$ and $d_{tread,0}$ influence the vertical stiffness, shear stiffness, and damping of the contact blocks (cf. explanation of w_{tread} above), as well as the geometrical maximum tire radius.

S is a commonly used measure for the modulus of elasticity of the tread rubber. It is assumed that:

$$E_{tread} = 218300 \times 1.0482792^S \frac{N}{m^2} = 10^{5.33905+0.020477 \cdot S} \frac{N}{m^2}$$

As an alternative to S, E_{tread} tread or c_{slip} can be specified. c_{slip} is the rate of change of the fore-aft force with longitudinal slip at very small slip values. Vertical deflection is assumed to be d_4 (or wheel load is assumed to be $F_{z,4}$), and the camber angle is 0. d_4 and $F_{z,4}$ are specified above. If $R_{tread,aniso}$ is set (see below), both S and E_{tread} define the tread stiffness in **longitudinal** direction.

ΔE_{tread} denotes the (theoretical) percentile radial tread stiffness increase if the deflected tread block's height reaches zero.

The 'tread positive' P parameter is defined as the percentage of that share of the footprint that actually has road contact, relative to the overall footprint area. It will be computed automatically (and the value specified here will be ignored), if a tread pattern bitmap file or a special tread pattern geometry is specified.

The tangential tread pattern shape factor, $f_{tread,shape,tang}$, will modify the ratio of 1/3 between Young's modulus E and the shear stress modulus G:

$$G = \frac{1}{3} \cdot f_{tread,shape,tang} \cdot E.$$

Depending on the shape of the tread pattern blocks and the progression of the radial stiffness (which also results in virtually larger values of E), this correction factor typically will be greater than one, but might have values less than one as well. **This value is optional.** Default value is 1.

The longitudinal tread pattern shape factor $f_{tread,shape,long}$ will be used as an influence factor for the longitudinal stiffness coupling between adjacent tread contact blocks. **This value is optional.** The default value is 0, which is equivalent to totally neglecting the longitudinal stiffness coupling. Value 1 sets the theoretical maximum stiffness value, based upon Young's modulus of the tread rubber, and the actual tread depth. Realistic values will be less than 0.8.

The torsional tread pattern shape factor $f_{tread,shape,tors}$ will be used as an influence factor for the torsional stiffness of the tread rubber blocks about the lateral axis. **This value is optional.** The default value is 0, which is equivalent to totally neglecting the torsional stiffness. Value 1 sets the theoretical maximum stiffness value, based upon Young's modulus of the tread rubber, and the actual tread depth. Setting a non-zero value will slightly increase dynamic rolling radius.

The ratio $R_{tread,aniso}$ allows to distinguish between lateral and longitudinal tread stiffness. Such anisotropy might be caused by geometrical tread pattern effects, or due to certain rubber aging processes. It holds

$$E_{lat} = R_{tread,aniso} \cdot E_{lang}$$

This value is optional. Default value is 1, which refers to ideal stiffness isotropy.

The ratio R_{sw} defines the lateral sidewall stiffness, relative to the new tire's radial tread rubber stiffness. **This value is only used if the sidewall contact model is activated, and is optional.** Default value is 0.1.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`damping_tread_rubber`	s	$time$	τ_{tread}
`tread_gluing_force_percentage`	%	%	$r_{tread,glue}$

τ_{tread} is the quotient of tread rubber damping modulus and tread rubber elasticity modulus. The physical dimension of that quotient is time, sometimes referred to as 'relaxation time'. If a tread rubber block is elongated, then it will theoretically creep back into its initial position, following the exponential displacement law

$$s = s_0 e^{-\frac{t}{\tau_{tread}}}.$$

The above consideration only holds if the stiffness follows Hooke's law, damping is viscous and linear, and mass can be neglected. In that case, the phase lag between displacement and force, if a rubber block is displaced harmonically, is proportional to the displacement frequency.

In contrast, deflection/force phase-lag of elastomers often is nearly **independent** on excitation frequency. This behavior is not yet implemented in **FTire**; instead, ideal viscous damping is assumed.

Optionally, tread rubber blocks can be equipped with a small mass as effective for shearing displacement, which then is computed on basis of the volume and of the assumed tread rubber density $\varrho_{rubber} = 0.54 \cdot 10^{-3} \frac{kg}{m^3}$. This option is selected by the ouput control item `activate_cont_element_mass`. By default, this tread rubber block elongation inertia is neglected.

$r_{tread,glue}$ is the percentage of $F_{z,1}$ which is acting as 'gluey' force between tread and road surface, if 10% of the tread is in contact with the road. By default, this force is 0.

	name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
one of	`tread_stiffn_vs_temp_spline`	$degC$, -	$degC$, -	table of size $2xi$
	`tread_stiffn_vs_temp_table`	$degC$, -	$degC$, -	table of size $2xi$

These optional spline or table data, respectively, define a temperature-dependent correction factor of the tread rubber stiffness. Both compression and shear stiffness will be modified in the same way by this factor.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`max_friction_velocity`	$\frac{m}{s}$	$\frac{length}{time}$	v_{mfr}
`sliding_velocity`	$\frac{m}{s}$	$\frac{length}{time}$	v_{slide}
`blocking_velocity`	$\frac{m}{s}$	$\frac{length}{time}$	v_{block}
`low_ground_pressure`	bar	bar (or psi for USC units)	p_{low}
`med_ground_pressure`	bar	bar (or psi for USC units)	p_{medium}
`high_ground_pressure`	bar	bar (or psi for USC units)	p_{high}
`mu_adhesion_at_low_p`	-	-	$\lambda_{adh,0}$
`mu_max_at_low_p`	-	-	$\lambda_{max,0}$
`mu_sliding_at_low_p`	-	-	$\lambda_{slide,0}$
`mu_blocking_at_low_p`	-	-	$\lambda_{block,0}$
`mu_adhesion_at_med_p`	-	-	$\lambda_{adh,1}$
`mu_max_at_med_p`	-	-	$\lambda_{max,1}$
`mu_sliding_at_med_p`	-	-	$\lambda_{slide,1}$
`mu_blocking_at_med_p`	-	-	$\lambda_{block,1}$
`mu_adhesion_at_high_p`	-	-	$\lambda_{adh,2}$
`mu_max_at_high_p`	-	-	$\lambda_{max,2}$
`mu_sliding_at_high_p`	-	-	$\lambda_{sliding,2}$
`mu_blocking_at_high_p`	-	-	$\lambda_{block,2}$
`temp_friction_dep_kind`	-	-	$K_{T\rightarrow frict}$
`frict_temp_fact_minus_20_degC`	-	-	$f_{T,-20}$
`frict_temp_fact_plus_20_degC`	-	-	$f_{T,+20}$
`frict_temp_fact_Tmax`	-	-	$f_{T_{max}}$
`frict_Tmax`	$degC$	$degC$	T_{max}
`glass_temp_tread_rubber`	$degC$	$degC$	T_{glass}

This group of variables describes the dry friction characteristics of the friction couple tread rubber / road surface under normal road conditions.

FTire assumes that the friction coefficient is a function of the three independent variables **sliding velocity, ground pressure, and tread rubber temperature.**

This general function is approximated by a certain simplified function, using only very few parameters. A function value $\mu_0(v,\, p_{ground})$ is given for any of the combinations of four different sliding velocities: $v = 0$, v_{mfr}, v_{slide}, v_{block}, and three different ground pressure values: $p_{ground} = p_{low}, p_{medium}, p_{high}$. The prescription of these pressure values is optional; default values are 0.01 bar, 2 bar, and 10 bar. Interpolation is performed by using quadratic polynomials 'in p-direction', and piecewise linear interpolation of the respective coefficients 'in v-direction'.

This interpolated friction coefficient might be modified in dependency on local tread rubber **temperature**. The switch $K_{T\rightarrow frict}$, with value 0, 1, or 2, determines the kind of this modification.

- If $K_{T\rightarrow frict} = 0$, a factor is applied to the speed and pressure dependent friction coefficient, being itself a quadratic function of temperature:

$$\mu = f(T) \cdot \mu_0(v,\, p_{ground}) = (a_0 + a_1 T + a_2 T^2) \cdot \mu_0(v,\, p_{ground})$$

This quadratic polynomial will interpolate the data pairs ($-$ 20 $degC$, $f_{T,-20}$) , ($+$ 20 $degC$, $f_{T,+20}$) , and (T_{max}, $f_{T_{max}}$).

- If $K_{T \to frict} = 1$, a factor is applied to the speed and pressure dependent friction coefficient, which now is given by evaluating a spline or lookup-table defined by:

	name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
one of	`friction_vs_temp_spline`	$degC$, -	$degC$, -	table of size $2xi$
	`friction_vs_temp_table`	$degC$, -	$degC$, -	table of size $2xi$

- If $K_{T \to frict} = 2$, the speed axis is stretched or compressed by a temperature dependent factor a_T/a_{20}, which in turn is given by the so-called WLF transform for elastomers:

$$\log a_T = -8.86 \cdot \frac{T - T_{glass} - 50}{101.5 + T - T_{glass} - 50}$$

(all temperatures in $degC$). Note that this factor is 1 if $T = 20\,degC$. The so-called **master curve** required by the WLF transform is $\mu_0(v, p_{ground})$ which is assumed to be valid at temperature $T = 20\,degC$.

During a simulation, the friction value can be further adapted to actual road conditions by using an additional correction factor that is defined in the road data file. For more details, please consult the separate road description documentation.

6.6 Temperature and Wear

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`time_const_tire_heating`	s	$time$	T_1
`time_const_tread_heating`	s	$time$	T_2
`tire_heating_at_ref_slip_low_v`	$degC$	$degC$	$\Delta T_{S,1}$
`tread_heating_at_ref_slip_low_v`	$degC$	$degC$	$\Delta T_{T,1}$
`tread_heating_at_ref_slip_med_v`	$degC$	$degC$	$\Delta T_{T,2}$
`tread_heating_at_ref_slip_vmax`	$degC$	$degC$	$\Delta T_{T,3}$
`heating_ref_slip`	%	%	$\kappa_{T,ref}$
`perc_frict_power_heating_tread`	%	%	$\triangle P_{frict,tread}$

This group of variables describes the properties of the thermal model, as described in chapter 4.2. Please refer to this chapter for the exact definition of the time constants T_1 and T_2 and the stationary tire structure and tread temperature increases $\triangle T_{S,1}$, $\triangle T_{T,1}$. $\triangle T_{T,2}$, and $\triangle T_{T,3}$. The thermal model will influence tread rubber stiffness and friction (see chapter 6.5), as well as actual inflation pressure, and thus indirectly also the tire structure stiffness properties.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`wear_rate_coefficient`	$\frac{mm}{s}$	$\frac{length}{time}$	c_{wear}
`wear_rate_exponent`	-	-	e_{wear}

Document Revision: 2022-1-r26108

These two variables quantitatively define the default wear function in the abrasion model, as described in chapter 4.3. The following wear function is used:

$$\frac{dh}{dt} = -c_{wear} \cdot \left(\frac{P_{frict}}{1 \frac{Nm}{s}} \right)^{e_{wear}}$$

6.7 Hydroplaning

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`hydroplaning_scaling_factor`	mm	$length$	$f_{20,hydro}$
`hydroplaning_residual_frict_fact`	-	-	$f_{frict,hydro}$
`formula_water_wedge_length`	mm	mm	f_{length}
`formula_water_wedge_angle`	deg	deg	f_{angle}

These parameters define the properties of the standard real-time-capable hydroplaning model.

- $f_{20,hydro}$ is the length of the stationary water wedge at the leading edge of the contact patch in the following reference conditions:
  - rolling speed 20 m/s
  - ground pessure 0.5 MPa
  - waterfilm height 10 mm
  - tread depth of new tire.

  The model assumes that the actual, laterally varying water wedge length is proportional to actual rolling speed and waterfilm depth, but reciprocal to ground pressure and tread depth. This assumption will be refined in a future release.
- $f_{frict,hydro}$ is the residual effective friction modification factor between contact patch leading edge and waterfilm wedge. Typically and by default, this factor will be close to zero.
- f_{length} is a formula expression for the water wedge length, used mutually exclusively with $f_{100,hydro}$. If f_{length} is non-blank, it is interpreted as a string containing a formula for the local water wedge length as function of the independent variables
  - `hwf` (local water film height [mm])
  - `v` (forward velocity [m/s])
  - `pn` (local ground pressure [MPa])
  - `td` (local tread depth [mm])
  - `wfp` (footprint width [mm])

  Note that even in *TeimOrbit* data files with physical unit system other than SI or mmks, the formula is expected to return the water wedge length in [mm]. Similarly, the independent variables are expected to be given in units as indicated above.

  Example: the formula f_{length} = `0.02*hwf*v*wfp/pn/td` sets the water wedge length to be proportional to water film height at inlet, rolling speed, and footprint width, but reciprocal to ground pressure and tread depth.
- f_{angle} is a second formula expression for the opening angle of the water wedge [deg], using the same independent variables as f_{length}. It is only used if an f_{length} formula is set as well.

6.8 Imperfections

name in data file *e*	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`static_balance_weight`	g	$mass$	w_{stat}
`static_balance_ang_position`	deg	$angle$	α_{stat}
`dynamic_balance_weight`	g	$mass$	w_{dyn}
`dynamic_balance_ang_position`	deg	$angle$	α_{dyn}
`radial_non_uniformity`	%	%	$\triangle c_{rad}$
`radial_non_unif_ang_position`	deg	$angle$	α_{rad}
`tang_non_uniformity`	%	%	$\triangle c_{tang}$
`tang_non_unif_ang_position`	deg	$angle$	α_{tang}
`lateral_non_uniformity`	%	%	$\triangle c_{laz}$
`lateral_non_unif_ang_position`	deg	$angle$	α_{lat}
`conicity`	deg	$angle$	α_{cone}
`ply_steer_percentage`	%	%	σ_{ply}
`radius_variation`	mm	$length$	$s_{\triangle r}$
`radius_variation_ang_position`	deg	$angle$	$\alpha_{\triangle r}$
`lateral_runout`	mm	$length$	$s_{\triangle y}$
`lateral_runout_ang_position`	deg	$angle$	$\alpha_{\triangle y}$

By this **optional** group of parameters, certain tire imperfections can be defined:

- **static imbalance** is defined by a mass w_{stat} and a angular position α_{stat} of a balance weight that would compensate the imbalance;
- **dynamic imbalance** is defined by a mass w_{dyn} and a angular position α_{dyn} of a balance weight. If two of these weights are placed at inner and outer rim flange, 180 deg apart from each other, this would compensate the imbalance;
- **radial non-uniformity** in terms of a harmonic variation of the radial stiffness along the tire circumference. The non-uniformity is defined in terms of the maximum percentage of the deviation of the mean value ($\triangle c_{rad}$), and in terms of the angular position α_{rad} where this value is achieved;
- **tangential non-uniformity** in terms of a harmonic variation of the tangential stiffness between belt and rim, along the tire circumference. The non-uniformity is defined in terms of the maximum percentage of the deviation of the mean value ($\triangle c_{tang}$), and in terms of the angular position α_{tang} where this value is achieved;
- **lateral non-uniformity** in terms of a harmonic variation of the lateral stiffness between belt and rim. The non-uniformity is defined in terms of the maximum percentage of the deviation of the mean value ($\triangle c_{lat}$), and in terms of the angular position α_{lat} where this value is achieved;
- **conicity**, in terms of a rotation angle α_{cone} of all belt segments around the circumferential belt axis in unloaded tire condition. Conicity, besides ply-steer, is one of the reasons for non-zero side forces at zero slip angle. This residual side force does not change sign in a vehicle-fixed co-ordinate system when the tire rolling direction is reversed;
- **ply-steer**, in terms of a percentage σ_{ply} of the effective ply-steer torque about the radial axis, relative to the torque that would arise from wheel load times contact patch length. Ply-steer, besides conicity, is one of the reasons for non-zero side forces at zero slip angle. This residual side force changes sign in a vehicle-fixed co-ordinate system when the tire rolling direction is reversed;
- **radius_variation**, in terms of a maximum deviation $s_{\triangle r}$ of the local tire radius from the mean tire radius. This kind of radius variation, in contrast to the more detailed spline-based function (see below), is assumed to be a harmonic function of the angular position. $\alpha_{\triangle r}$ is the angular position where the maximum positive radius variation occurs.
- **lateral_runout**, in terms of a maximum lateral shift $s_{\triangle y}$ of the belt center relative to rim center. This kind of lateral runout, in contrast to the more detailed spline-based function (see below), is assumed to be a harmonic function of the angular position. $\alpha_{\triangle y}$ is the angular position where the maximum positive lateral shift occurs.

	name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
one of	`radial_non_unif_vs_angle_spline`	deg, %	$angle$, %	table of size $2xi$
	`radial_non_unif_vs_angle_table`	deg, %	$angle$, %	table of size $2xi$
one of	`tang_non_unif_vs_angle_spline`	deg, %	$angle$, %	table of size $2xk$
	`tang_non_unif_vs_angle_table`	deg, %	$angle$, %	table of size $2xk$
one of	`lateral_non_unif_vs_angle_spline`	deg, %	$angle$, %	table of size $2xk$
	`lateral_non_unif_vs_angle_table`	deg, %	$angle$, %	table of size $2xk$
one of	`radius_var_vs_angle_spline`	deg, %	$angle$, %	table of size $2xl$
	`radius_var_vs_angle_table`	deg, %	$angle$, %	table of size $2xl$
	`radius_var_2d_table`	mm	$length$	table of size $mxn + 1$
one of	`lat_runout_vs_angle_spline`	deg, %	$angle$, %	table of size $2xl$
	`lat_runout_vs_angle_table`	deg, %	$angle$, %	table of size $2xl$
one of	`mass_var_vs_angle_spline`	deg, %	$angle$, %	table of size $2xo$
	`mass_var_vs_angle_table`	deg, %	$angle$, %	table of size $2xo$
one of	`tread_gauge_var_vs_angle_spline`	deg, %	$angle$, %	table of size $2xp$
	`tread_gauge_var_vs_angle_table`	deg, %	$angle$, %	table of size $2xp$
	`tread_gauge_var_2d_table`	mm	$length$	table of size $qxr + 1$

These **optional** 1d- or 2d-look-up tables allow a more precise description of certain imperfections. All seven groups of look-up tables can be used either alternatively or in addition to the respective simplified definitions of the data in the preceding block.

Imperfections can be specified in one of two different formats (the second currently only is available for radius variation = radial runout):

- in terms of a one-dimensional, not necessarily equally spaced, function of the circumferential coordinate. In this case, the imperfection only depends on the angle about wheel spin axis in counter-clockwise sense; it is assumed to be constant along belt lateral direction; or
- in terms of an equally spaced two-dimensional function, taking into account a dependency on lateral coordinate as well. Currently, this format is only available for tire outer radius variation and for tread gauge variation.

In the first case, the imperfection values are given by at most $i = 10000$ data pairs each. These tables will be interpolated, using either a smooth cubic spline or the piecewise linear approach. The kind of interpolation is indicated by the data item's suffix: `_spline` means spline interpolation, `_table` means piecewise linear interpolation. The angle values must be monotonously increasing, but may be chosen non-equidistant. All imperfections in this case are specified in terms of a variation percentage of a nominal value.

The following types of 1d imperfections can be specified:

- **radial non-uniformity**, in terms of a local variation of radial stiffness which is defined as a percentage of the mean radial stiffness
- **tangential non-uniformity**, in terms of a local variation of longitudinal stiffness which is defined as percentage of the mean longitudinal stiffness
- **lateral non-uniformity**, in terms of a local variation of lateral stiffness which is defined as percentage of the mean lateral stiffness
- **radius variation (radial runout)**, in terms of a local belt radius variation, which is defined as percentage of the mean belt radius (this variation does not contain an additional optional tread gauge variation). The tire will automatically be statically balanced; any static or dynamic imbalance is to be entered as explained in the preceding data block
- **lateral runout**, in terms of a lateral belt center shift relative to rim center, which is defined as percentage of the mean belt radius.

- **mass variation**, in terms of a local mass density variation which is defined as a percentage of the mean mass density. The tire will automatically be statically balanced; any static or dynamic imbalance is to be entered as explained in the preceding data block
- **tread gauge variation**, in terms of a local variation of the tread gauge which is defined as a percentage of the tire's mean tread depth in a unworn condition

In the second case, it is assumed that a equally spaced grid dataset is specified, covering the whole belt region. Spacing in longitudinal and lateral direction is computed automatically, depending on the belt length, belt width, and on the number of entries that are specified in x- and y- direction. In contrast to the specification of the 1d dataset, absolute variation values in the current length unit are required in the case of the 2d data. The first numerical value in the data block specifies the number of grid-point values in lateral direction. The remaining $n_x \times n_y$ specify data on a grid with n_x points in longitudinal and ny points in lateral direction. This is an example of such a data block:

```
radius_var_2d_table =                    $ no. of columns
4
0.00   0.00   3.00   2.00
0.00   0.00   2.00   2.00
0.00   0.00   0.00   0.00
```

This data block species 3 grid points in longitudinal direction and 4 grid points in lateral direction, for which radius variation values in [mm] are given.

2d look-up tables are interpolated piecewise bi-linearly.

The following two types of 2d imperfections can be specified:

- **radius variation (run-out)**, in terms of an absolute length value. The tire will automatically be statically balanced; any static or dynamic imbalance is to be entered as explained in the preceding data block;
- **tread gauge variation**, in terms of an absolute length value. This type of tread gauge variation will not affect the tire's outer radius. Rather, the belt radius is automatically adjusted together with the tread gauge.

Note that **all** types of imperfections are de-activated during pre-processing.

6.9 Misuse

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`rim_to_flat_tire_distance`	mm	$length$	d_{rc}
`rim_flange_contact_stiffn`	$\frac{N}{mm}$	$\frac{force}{length}$	c_{rc}
`rim_flange_contact_stiffn_progr`	$\%$	$\%$	Δc_{rc}
`rim_to_road_contact_stiffn`	$\frac{N}{mm}$	$\frac{force}{length}$	c_{rrc}
`rim_to_road_contact_frict_coeff`	$-$	$-$	μ_{rrc}
`sidewall_stretch_stiffn_factor`	$-$	$-$	f_{csw}
`puncture_stress`	MPa	$\frac{force}{length^2}$	p_{punct}
`bead_slips_off_seat_Fy`	N	$force$	$F_{y,max}$

The items in this group, among others, define the progressive radial stiffness property of the tire when the tire inner liner near a belt edge touches the inner side of the bead, and thus comes into indirect contact with the rim flange. This situation will only occur under misuse conditions, or for flat running tires.

- d_{rc} is the distance between the rim flanges and tire inner liner, when the tire is running in loaded condition at zero inflation pressure;
- c_{rc} is the additional radial contact stiffness, measured under the condition that approximately 10 deg of the bead circumference has contact with the tire inner liner. Default stiffness value is 1000 $\frac{N}{mm}$;

Document Revision: 2022-1-r26108

- Δc_{rc} is the (theoretical) percentual increase in radial contact stiffness, if rim-flange-to-road distance vanishes;
- c_{rrc} defines the contact stiffness of the rim, if its flange directly touches the road surface. Typically, such an event will only happen at an extremely large camber angle, or when approaching high curbs with an acute heading angle. The stiffness acts along the local road surface normal near the point of contact, and it is measured under the condition that approximately 10 deg of the rim flange circumference has contact to the road;
- μ_{rrc} defines the sliding friction coefficient of direct rim-to-road contact. This coefficient will cause forces in the local road tangent plane during a rim-to-road contact event;
- f_{csw} is the local radial stiffness of a single side-wall when stretched beyond its geometrical length, relative to the local belt radial stiffness;
- p_{punct} is the minimum normal stress exerted on the belt by a contact element that will puncture the belt structure and in consequence causes a loss of filling gas;
- $F_{y,max}$ is the side force at which the tire bead starts sliding off the rim seat and thus causes a sudden permanent pressure drop.

6.10 Rim

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`sticking_torque_tire_on_rim`	Nm	$force \cdot length$	$T_{frict,rim,stick}$
`sliding_torque_tire_on_rim`	Nm	$force \cdot length$	$T_{frict,rim,slide}$

These two optional items define sticking and sliding friction torque between the tire and the rim. If the absolute value of drive or brake torque is greater or equal to the sticking torque, the tire will start sliding on the rim, until the drive/brake torque again becomes smaller or equal to the sliding torque. During sliding, the absolute drive or brake torque transmitted to the rim is limited by $T_{frict,rim,slide}$, whereas, in sticking state, it is limited by $T_{frict,rim,stick}$.

name in data file	***unit** in `cosin/io` data file*	***unit** in **TeimOrbit** data file*	***symbol** or **formula***
`rim_axial_moment_of_inertia`	kgm^2	$mass \cdot length^2$	$j_{rim,yy}$
`rim_radial_moment_of_inertia`	kgm^2	$mass \cdot length^2$	$j_{rim,xx}$

$j_{rim,yy}$ is the total moment of inertia of the rim and other rotating parts, but without the tire, with respect to the wheel spin axis ('axial'). **This parameter is only needed and used if rim rotation is integrated inside FTire.** It can be omitted if **FTire** is called by a 3^{rd}-party software in other than real-time speed mode. In real-time speed mode however, the axial moment of inertia should be the same as in the calling solver's suspension model. In this speed mode, it is used to automatically tune certain numerical parameters, requiring knowledge about the rim parameters.

$j_{rim,xx}$ is the total moment of inertia of the rim and other rotating parts, but without tire, with respect to a axes normal to the wheel spin axis ('radial'). **This parameter is only needed and used if rim rotation is integrated inside FTire**, for the computation of gyroscopic moments**.** It can be omitted if **FTire** is called by a 3^{rd}-party software.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`rim_data_file`	*string*	*string* in quotes	

The **optional** rim data file (which is either in ***cosin/io*** or in TeimOrbit format) either holds the internal rim flexibility model data, as specified in the next table, or is passed over to a user-specified rim model in case the respective model library is available and activated. If, at the same the, the rim model data file is specified with the CTI function `ctiLoadRimData`, the file name specified there will take precedence over the value of `rim_data_file`.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***
`flex_rim_lateral_stiffn`	$\frac{N}{mm^2}$	$\frac{force}{length^2}$	$c_{rim,lat}$
`flex_rim_radial_stiffn`	$\frac{N}{mm^2}$	$\frac{force}{length^2}$	$c_{rim,rad}$
`flex_rim_tors_stiffn`	$\frac{Nm}{deg}$	$\frac{force \cdot length}{angle}$	$c_{rim,tors}$
`flex_rim_compliance_shape_fact`	deg	$angle$	$\alpha_{rim,compl,shape}$
`number_rim_spokes`	$integer$	$integer$	n_{spokes}
`first_rim_spoke_pos`	deg	$angle$	φ_{rim}
`rim_stiffn_variation_by_spokes`	%	%	$\Delta P_{stiffn,spokes}$
`radial_stiffn_red_inner_side`	%	%	$f_{red,stiffn,rimside}$
`max_elastic_lat_rim_node_displ`	mm	mm	$d_{rim,max,lat}$
`max_elastic_rad_rim_node_displ`	mm	mm	$d_{rim,max,rad}$

This group of variables describes the properties of the internal rim model, if no rim model data file is specified, as described in chapter 4.6.

- $c_{rim,lat}$ and $c_{rim,rad}$ are specific stiffness values of the rim flanges in lateral and radial direction, measured in translational stiffness ($\frac{force}{length}$) per length unit
- $c_{rim,tors}$ is the torsional stiffness of the rim about wheel lateral axis
- $\alpha_{rim,compl,shape}$ is a shape factor for the rim flanges' Green's function. It specifies the circumferential angle at which the influence of the rim flange stress on rim flange displacement at zero angle has decayed by 50%, as compared to force influence at zero angle
- n_{spokes} denotes the number of rim spokes
- φ_{rim} denotes the angular position of the first rim spoke, relative to the rim rotation angle
- $\Delta P_{stiffn,spokes}$ is the percentual stiffness variation of the rim flanges, as measured near and in-between spokes
- $f_{red,stiffn,rimside}$ is the stiffness in radial direction of the inner rim side, expressed as percentage of the radial stiffness of the outer rim side
- $d_{rim,max,lat}$ and $d_{rim,max,rad}$ is the maximum elastic displacement of the rim nodes, beyond which plastic deformation will occur.

6.11 TPMS Sensor

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***

angle_pos_TPMS_sensor	deg	$angle$	a_{TPMS}
lateral_pos_TPMS_sensor	mm	$length$	y_{TPMS}
dist_TPMS_sensor_inner_liner	mm	$length$	d_{TPMS}
mass_TPMS_sensor	kg	$mass$	m_{TPMS}

These two optional items define the mounting position of a TPMS (tire pressure monitoring system) sensor. This sensor, if activated, provides pressure, temperature, and acceleration signals just like a real TPMS sensor. All TPMS signals are written to the output plot file. Thus, just like any other plot signal, they can be visualized with a scope during a running simulation, made available to the calling solver via CTI, or plotted and further analyzed during post-processing. Besides pressure and temperature, the sensor provides all 3 components of both translational and angular accelerations, measured in sensor-fixed coordinates, like a real TPMS sensor would. If the air cavity model is activated, the pressure provided by the signal is the local pressure near the sensor. Otherwise, the pressure is the current deflection and temperature dependent inflation pressure. Temperature is the temperature of the filling gas, no matter if the detailed thermal model is activated or no.

- a_{TPMS} defines the angular position of the sensor in circumferential direction. For a non-rotated tire a value of 0 deg defines bottommost position inside the tire; 90 deg the foremost position (if viewed along rolling direction), 180 deg the topmost position, and so on. The sensor is mounted on, and is moving with the tire inner liner, so the actual position of the sensor will cyclically move up and down
- y_{TPMS} defines the lateral position of the sensor, relative to the wheel mid plane. Positive values shift the sensor to the left if viewed along the rolling direction, and negative values to the right. Since the sensor 'sticks' to the tire inner liner, it will follow tire lateral motion and vibrations
- d_{TPMS} defines the effective distance of the sensor from the tire's inner liner surface
- m_{TPMS} defines the sensor's total mass.

6.12 Numerical Settings

name in data file		***unit*** *in cosin/io data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *o*
number_belt_segments		-	-	n_{seg}
one of	number_blocks_per_belt_segm	-	-	n_{blocks}
	tread_block_distance	mm	$length$	d_{blocks}
tread_discretization_type		-	-	$discr_{tread}$
number_tread_strips		-	-	n_{strips}
number_belt_bend_shape_funct		-	-	n_{bend}
maximum_time_step		s	$time$	h_{max}
maximum_angle_increment		deg	$angle$	$\triangle\alpha_{max}$
one of	BDF_parameter	-	-	β
	BDF_parm_wo_rigid_body_mode	-	-	$\overline{\beta}$
BDF_parameter_air_vibration		-	-	β_{air}
Jacobian_update_cycle_length		-	-	n_{BDF}
contact_processor_bound		%	%	B_{cont}
high_precision_tang_plane		-	-	P_{TP}
high_precision_tang_plane_RT		-	-	$P_{TP,RT}$
high_precision_contact_comp		-	-	P_{CC}
stat_time_step		s	$time$	h_{stat}
stat_BDF_parameter		-	-	β_{stat}
stat_mass_reduction		-	-	$\rho_{m,stat}$
stat_inertia_reduction		-	-	$\rho_{j,stat}$
step_size_Jacobian		-	-	h_{diff}
stat_conv_tolerance		-	-	ϵ_{stat}
stat_max_update_iter		-	-	$n_{max,stat}$

Document Revision: 2022-1-r26108

pp_conv_tolerance	-	-	ϵ_{pp}
pp_max_cycle	-	-	$n_{cycle,pp}$
surface_output_res_long	mm	$length$	$R_{surface,lo}$
cross_section_output_location	$-$	$-$	$s_{cross-sect}$
surface_output_res_lat	mm	$length$	$R_{surface,la}$

This is the set of data that controls the numerical properties of **FTire**:

- n_{seg} is the number of belt segments. Maximum value is virtually unlimited, but typically doesn't need to be larger than 200.
- n_{blocks} is the number of contact blocks per belt segment (which might be defined by the tread block distance d_{block} as well, like in RTire). The maximum number of contact blocks per belt segment is virtually unlimited, but typically doesn't need to be larger than 100.
- $discr_{tread}$ is the integer-coded tread discretization pattern:

  0 standard (harring-bone pattern)

  1 pseudo-random in longitudinal direction, equally spaced into strips in lateral direction
- n_{strips} is the number of equally spaced strips into which the contact blocks are arranged if $discr_{tread} = 1, 2$. n_{blocks} is automatically adjusted in such a way that the number of contact blocks is the same in each strip (that is, n_{blocks} is a multiple of n_{strips}).
- n_{bend} is the number of mutually orthogonal shape functions used to describe the shape of the belt bending in lateral direction. The coefficient, of each such shape function (being eigenfunctions of the beam differential equation), is one additional degree of freedom of the respective belt element. If n_{bend} is not specified, the original **FTire** approach of using one single degree of freedom (described by a quadratic polynomial) is used; this is done for compatibility reasons. The choice of n_{bend} is determined by another compromise between accuracy and computing effort. Typically, $n_{bend} = 3..5$ is a good compromise.
- h_{max} is the maximum internal time step that is allowed to be chosen by **FTire**. **FTire** may be called with very large external time steps, no matter whether this makes sense or not. Internally, **FTire** uses multi-step integration with an internal time step that is chosen on basis of h_{max}. This internal time step is kept constant if the external time step does not change. Changing external time-step may result in considerable longer computation time, because certain time-consuming pre-processing calculations have to be repeated. As a consequence, try to avoid changes in external time step.
- $\triangle\alpha_{max}$ is the maximum increment of the rim rotation angle per time step (if necessary the internal time step will be decreased during a simulation, to meet this specification). Default value is 1 deg.
- β is a numerical parameter to control the implicit (BDF) integration scheme. $\beta = 0$ would choose the explicit Euler scheme, $\beta = 0.5$ the trapezoidal rule, $\beta = 1$ and the implicit Euler scheme. Theoretically, every value between 0 and 1 is allowed; 0.5 is however recommended. $\overline{\beta}$ has the same meaning as β , except that the belt's translational rigid body modes are separated and integrated fully explicitly. This leads to an improved accuracy of steady-state conditions when running at higher rolling speeds. It is recommended to use $\overline{\beta}$ instead of β . However, this might lead to slightly different (more accurate) contact patch forces when rolling at higher speeds. For compatibility to older **FTire** versions, β continues to be recognized.
- β_{air} is a numerical parameter to control the implicit (BDF) integration scheme of the air vibration model solver. This parameter serves as 'numerical damping' of the air vibration; a value between 0.6 and 0.8 will be appropriate in most cases.
- n_{BDF} defines the number of steps between two BDF Jacobian updates (default value is 1, which causes update of the Jacobian in every integration step).
- B_{cont} (contact processor bound) defines the maximum height of any contact element which needs to be checked for road contact. This height is measured as a percentage of the tire's diameter. $B_{cont} = 50$, say, would mean that all contact elements which are located below rim center will be tested for contact to the road. It parameter is optional, default value is 35.

- P_{TP} (possible values 0 or 1, default value is 0) selects the precision and efficiency of the road contact plane computation for standard FTire model evaluation. If $P_{TP} = 0$, for all elements in contact with the road the belt normal direction is used as first order approximation of the road normal vector. For all other values of P_{TP}, a high-precision computation is performed, by evaluating the road in three different locations near the contact point.
- $P_{TP,RT}$ (possible values 0 or 1, default value is 0) selects the precision and efficiency of the road contact plane computation in case of accelerated FTire model evaluation. If $P_{TP,RT} = 0$, for all elements in contact with the road the belt normal direction is used as first order approximation of the road normal vector. For all other values of $P_{TP,RT}$, and if a road model in RGR format is used, a high-precision computation is performed, by interpolating not only height and friction factor from RGR data, but also the local road gradients and by this the local road normal vector. These extra computations might slightly increase computation time.
- P_{CC} (possible values 0 or 1, default value is 0) selects the precision of the tire contact computation at leading and trailing edges of the footprint. If $P_{CC} = 0$, the impact of local belt curvature (which might be non-neglectable near the footprint boundaries) on sliding velocity is disregarded. For downward compatibility reasons, this setting is chosen to be the default. Reason for this is it might have non-neglectable influence on longitudinal force when rolling over high sharp-edged obstacles. On the other hand, setting $P_{CC} = 1$ will improve the accuracy of the rolling circumference dependency on forward speed. For this reason, $P_{CC} = 1$ is recommended during parameter identifcation, if there is no intent to use the file with an older cosin version. Versions prior to 2020-1 behave as if $P_{CC} = 0$.
- h_{stat} (static step size) is the simulation step size which is to be used in statics calculation during pre-processing. This parameter will only affect computation speed and convergence during the statics computation. The parameter is optional, default value is 0.003.
- β_{stat} (static BDF parameter) is the BDF parameter which is to be used in statics calculation during pre-processing. This parameter will only affect computation speed and convergence during the statics computation. The parameter is optional, default value is 3.5.
- $\rho_{m,stat}$ (statics mass reduction factor) is a factor by which all nodal masses are to be reduced in statics computation during pre-processing. This parameter will only affect computation speed and convergence during the statics computation. The parameter is optional, default value is 0.01.
- $\rho_{j,stat}$ (statics moments of inertia reduction factor) is a factor by which all belt elements' moments of inertia are to be reduced in statics computation during pre-processing. This parameter will only affect computation speed and convergence during the statics computation. The parameter is optional, default value is 0.01.
- h_{diff} defines the step-size for numerical differentiation of the Jacobian. This Jacobian is needed for the Newton-Raphson solution of the non-linear sets of equations during pre-processing.
- ϵ_{stat} is the tolerated residual value during the statics iteration.
- $n_{max,stat}$ is the maximum number of iterations for a single statics case.
- ϵ_{pp} is the tolerated residual value during the pre-processing iterations.
- $n_{cycle,pp}$ is the maximum number of cycles of the pre-processing iterations. In each cycle, a pre-defined sub-set of the parameters are approximated. Only in the last cycle, all parameters are determined simultaneously. More than one cycle can be set, but is only needed if the convergence speed during pre-processing is poor.
- $R_{surface,long}$ and $R_{surface,lat}$ are the approximate resolutions of the additional output of surface nodes and nodal velocities in longitudinal and lateral direction, respectively. This output is meant for the use in prm and CFD co-simulation, and to be activated after opening the tire file with cosin/tools. The computations necessary might be time-consuming and are automatically skipped in higher speed modes(even if activated).
- $s_{cross-section}$ specifies the dimensionless circumferential location of a single cross-section to be written to the above-mentioned output file, as an alternative to the output of the complete tire surface geometry. If specified, $R_{surface,long}$ is disregarded but only $R_{surface,lat}$ is used as a resolution in lateral direction.

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol***
`only_use_FTire_updates_until`	*YYYYMMDD*	*YYYYMMDD*	-

This optional parameter (called **compatibility date** or previously **feature date**) allows specifying a date such that all **FTire** model changes, being implemented after this date, will be **deactivated** for the FTire instance using this data file. See to chapter 11 for a list of major changes that can be deactivated by setting the relevant compatibility date. The compatibilty date (or compatibilty version = emulated version) can also be set simultaneously for **all FTire** instances, using the cosin/tools GUI.

The basic parameters are pre-processed during initialization, resulting in the pre-processed parameters. The **pre-processed data** are appended to the basic parameters for use in further simulations. By this, the pre-processing calculation phase can be omitted. **FTire** automatically recognizes whether the data file contains valid pre-processed data. If so, and if these parameters are generated by the same **FTire** version, the pre-processing phase is skipped.

7 Operating Conditions

Certain tire data can be controlled during a simulation, without re-running the pre-processing step. These parameters are called **operating condition parameters**.

At present, these parameters are

- **inflation pressure** (not to be confused with the inflation pressure value in the `[FTIRE_DATA]` section, as described above). The operating condition value of 'inflation pressure' describes the **actual**, possibly time-dependent inflation pressure, whereas the tire data value describes the inflation pressure at which the measurements were taken.
- **tread depth** (not to be confused with the tread depth value in the `[FTIRE_DATA]` section, as described above). The operating condition value of 'tread depth' describes the **actual**, possibly time-dependent tread depth, whereas the tire data value describes the tread depth at which the measurements were taken.
- **ambient temperature** in $[degC]$. The operating condition value of 'ambient temperature' describes the **actual**, possibly time-dependent ambient temperature.
- **road surface temperature** in $[degC]$. The operating condition value of 'road surface temperature' describes the **actual**, possibly time-dependent road surface temperature.

Several methods are available to specify operating conditions:

- In **FTire/sim**, operating conditions are set in terms of 'sources & sinks' signals, specified in data block `$simulation` of the sim-file.
- In Matlab/Simulink, using the **FTire/link** blockset, some of the blocks have input ports for operating conditions.
- When using a cti-file to specify CTI data, operating conditions can be set in all CTI-driven environments just like in FTire/sim. **This is the most flexible and the preferred method.**
- In CTI-driven environments, special API functions are provided to set operating conditions, see CTI documentation (not all 3rd-party solvers make use of these functions).
- As an alternative, if not using one of the methods above, variable operating conditions can be specified in the tire data file as described below. Please note however that not all 3rd-party solvers call the necessary CTI API function `ctiReadOperatingConditions()`. This method is provided for compatibility reasons with older versions, but might be considered deprecated in a future release.

To determine the actual operating conditions with the last method, if triggered by the calling solver and when the data file is in format TeimOrbit, **FTire** looks for the section `[OPERATING_CONDITIONS]` in the tire data file. If this section is not found, or if it does not contain the respective definitions, **FTire** uses as operating conditions the measurement conditions, as defined in section `[FTIRE_DATA]`.

In case the section `[OPERATING_CONDITIONS]` is defined, **FTire** tries to read a constant value for each operating condition. This value either may be the same for all tires using the data file, or it can have individual values for each such tire instance. If no constant value is found, **FTire** looks for a table that is defining data points for operating condition vs. time. These data points will then be piecewise linearly interpolated with respect to the simulation time.

Constant values can be defined as follows:

name in data file	***unit*** *in* `cosin/io` *data file*	***unit*** *in* ***TeimOrbit*** *data file*	***symbol*** *or* ***formula***

one of	inflation_pressure	bar	bar (or psi for USC units)	p_{act}
	inflation_pressure_wheel_i	bar	bar (or psi for USC units)	p_{act}
one of	tread_depth	mm	$length$	$d_{tread,act}$
	tread_depth_wheel_i	mm	$length$	$d_{tread,act}$
one of	filling_gas_temperature	$degC$	$degC$	$T_{gas,act}$
	filling_gas_temperature_wheel_i	$degC$	$degC$	$T_{gas,act}$
one of	mean_tread_surface_temperature	$degC$	$degC$	$T_{tread,act}$
	mean_tread_surface_temperature_wheel_i	$degC$	$degC$	$T_{tread,act}$
one of	ambient_temperature	$degC$	$degC$	T_{amb}
	ambient_temperature_wheel_i	$degC$	$degC$	T_{amb}
one of	road_surface_temperature	$degC$	$degC$	T_{road}
	road_surface_temperature_wheel_i	$degC$	$degC$	T_{road}

The values of `inflation_pressure` and/or `tread_depth` and/or `filling_gas_temperature` and/or `mean_tread_surface_temperature` and/or `ambient_temperature` and/or `road_surface_temperature` will be taken for all wheels that use the data file, whereas the values with postfix `_wheel_i` only will be used for wheel # i (counting sequence is #1 $= fl$, #2 $= fr$, #3 $= rl$, #4 $= rr$, and so on).

The tire temperature values, `filling_gas_temperature` and `mean_tread_surface_temperature`, serve as initial conditions if the thermal model is active.

Look-up tables for the operating conditions vs. time are to be entered as sub-sections of section `[OPERATING_CONDITIONS]`.
These subsections may each contain up to 200 data pairs, one pair per line. Every data pair consists of a value for time and a corresponding value defining the operating condition. The units are the same as defined for the constant values (defined in the table above). Similarly as for the constant values, tables which are valid for all tires, or individual tables for each tire instance are allowed.

The names of these table sub-sections (with obvious meanings) are:

- `(TIME_TABLE_INFLATION_PRESSURE)`
- `(TIME_TABLE_INFLATION_PRESSURE_WHEEL_i)`
- `(TIME_TABLE_TREAD_DEPTH)`
- `(TIME_TABLE_TREAD_DEPTH_WHEEL_i)`
- `(TIME_TABLE_AMBIENT_TEMPERATURE)`
- `(TIME_TABLE_AMBIENT_TEMPERATURE_WHEEL_i)`
- `(TIME_TABLE_ROAD_SURFACE_TEMPERATURE)`
- `(TIME_TABLE_ROAD_SURFACE_TEMPERATURE_WHEEL_i)`

The following example defines a sudden pressure loss (between 5 and 5.2 s of the simulation time) in tire # 2 (front right). Moreover, it specifies constant inflation pressure (2.2 bar) for the other tires, and a certain, equal and constant extreme wear rate for all tires. The model level is chosen to be the standard variant for all tires at any time:

```
$------------------------------------------------------------OPERATING_CONDITIONS
[OPERATING_CONDITIONS]
 INFLATION_PRESSURE_WHEEL_1 = 2.2
 INFLATION_PRESSURE_WHEEL_3 = 2.2
 INFLATION_PRESSURE_WHEEL_4 = 2.2

(TIME_TABLE_TREAD_DEPTH)
   0     8.0
```

```
 100    7.5

(TIME_TABLE_INFLATION_PRESSURE_WHEEL_2)
   0    2.2
   5    2.2
   5.2  1.2
```

8 Additional Runtime and Output Control

Certain model settings and extra output can be requested, and controlled, by setting additional items in the section `[MODEL]`, provided that the data file is in TeimOrbit format. These control items are:

name in data file	***valid values***	***meaning***
`gravity`	*string*	set gravity value according to the one of • earth • moon • mars • zero Default is 'earth'
`tire_construction`	*string*	set tire construction type to one of • pneumatic_tire • tweel • solid_rubber Default is 'pneumatic_tire'
`tire_structure`	*string*	set tire structure letter one of • R • D • B • RF • ZR Default is 'R'
`tire_kind`	*string*	set tire kind to one of • unknown • pass_car_summer • pass_car_winter • two_wheeler_summer • two_wheeler_winter • truck_summer • truck_winter • aircraft • agricultural • earthmover_OTR Default is 'unknown'

`filling_gas_0_air_1_nitrogen`	0, 1, 2	set tire filling gas to: **0** air **1** nitrogen **2** helium
`statics_accuracy`	0,1,2	set statics accuracy (for statics and steady-state computations) to: **0** 'standard' (default) **1** 'improved' **2** 'highest In case of 'highest', additional steady-state computations are ran if required, which enable output of estimated dynamic rolling radius as function of load and rolling speed
`run_time_mode`	0,1,2,3,4,5	set run-time speed mode (for speed modes > 2, an extra license key is required): **0** standard mode **1** acceleration level 1: no model extensions (other than thermal and tread wear model) **2** acceleration level 2: no model extensions, no extra output **3** acceleration level 3: no model extensions, no extra output, coarse mesh, numerics optimized for speed **4** real-time mode level 1: no model extensions, no extra output, coarse mesh, numerics optimized for speed, limited TYDEX output; calling solver must run with constant step size **5** real-time mode level 2: no model extensions, no extra output, coarse mesh, numerics optimized for speed, no TYDEX output (cannot be used in certain 3rd-party solvers); calling solver must run with constant step size Remark: the run-time mode might be passed over to the calling CTI environment. The step-size condition for modes 4 and 5 does not affect the FTire model itself, but only the algorithm by which CTI calls FTire

activate_tread_pattern	0, 1	take into account tread pattern if defined by bitmap file or by simplified geometrical definition: **0** deactivate (default) **1** activate
activate_side_wall_model	0, 1	activate side-wall contact elements: **0** deactivate (default) **1** activate
activate_cont_element_mass	0, 1	activate mass for contact elements: **0** deactivate (default) **1** activate
activate_thermal_model	0, 1	activate thermal model, if data is supplied: **0** deactivate (default) **1** activate
activate_tread_wear_model	0, 1, 2, 3	activate tread wear prediction model, if data is supplied **0** deactivate (default) **1** activate, determine tread-block-individual element wear and resulting tread hight, **2** activate, averaged, determine circumferentially averaged tread wear and resulting tread hight, **3** only compute input, determine the circumferentially averaged tread wear without effecting the actual tread height
activate_geom_approx	0, 1, 2	activate detailed structure-distortion-dependent inner volume computation (no extra data required). This setting is overwritten by the one of the air cavity model if activated **0** deactivate (default); use V_{meas} and G_V for volume estimation if given **1** activate fast and simplified cross-section approximation **2** activate detailed cross-section computation (requires extra computing time and is only needed in special cases)
activate_air_vibration_model	0, 1, 2	activate air vibration model (no extra data is required apart from specification of filling gas) **0** deactivate (default) **1** activate, fast and simplified cross-section approximation **2** activate, detailed cross-section computation

`activate_flexible_rim_model`	0, 1, 2	activate flexible and/or viscoplastic rim model, if data or user-library is supplied: **0** rigid rim (default) **1** use internal model, data as described above **2** use user-provided model, if library and data file has been specified (see CTI documentation for more information)
`tire_side`	*string*	vehicle side on which tire is mounted, as assumed in the data processing. If tire is actually mounted on the opposite side of the vehicle, data will be mirrored. Possible string values: **asis** don't mirror data **left** mirror data if tire is mounted on the right-hand side of the vehicle **right** mirror data if tires is mounted on the left-hand side of the vehicle **symmetric** modify data to make the tire symmetric
`adjust_input_signals`	0, 1	if the rim velocity input signals, provided by the calling solver, are not consistent with the rim position signals, adjust them automatically (only used in 3rd-party environments). Default is 0 (don't adjust)
`balance_statically`	0, 1	automatically add the appropriate virtual counterweights to the rim-fixed part of the FTire structure in order to move the unloaded tire/rim center of gravity into the wheel center (effective only if some irregular mass variations are specified as imperfections). Note: automatic **dynamic** balancing is currently not provided. Default is 0 (don't balance automatically)
`user_library_path`	*string*	path to the directory holding optional user-defined libraries for rim and road models, etc.
`separate_animation`	0, 1, 2, ...	display an extra **FTire** animation window. If the value is 2, a movie file (**temp.mp4**), located in the working directory, will be generated. Please follow the instructions that are listed in the graphics window. If the data file is edited with **cosin/tools for tires**, more detailed animation scene, and sound, output controls are available.

number_cross_section_modifiers	0, 1, 2, ...	display additional control sliders, that can be used to vary the local belt radius, in the **FTire** animation window. If the value is 0, no additional sliders will be displayed. The cross-section modifiers will be equally spaced across the belt width;
road_grid_size	*length*	approximate width and length of the road visualization grid in the animation window
road_grid_line_dist	*length*	approximate grid line distance of the road visualization in the animation window
number_tire_structure_nodes	-	number of displayed nodes in a single cross-section, if the tire structure is visualized with a uniform mesh. Note that these nodes do not coincide with the tire structure point masses; they are only used to visualize the distortion of the belt structure.
rim_graphics_file	*string*	the name of an optional CAD data file in shl, obj, or stl format, containing the detailed rim geometry. Data will be automatically shifted, rotated, and scaled to exactly match the given rim size. This specification both is used when interfacing to **cosin/prm** and as rim animation model.
animation_time_step	*time*	approximate time step for the on-line animation, to be specified in the data file's time unit. If zero, or not specified, an approximate animation time step of 1 ms will be used.
animation_start	*time*	approximate simulation time to begin the animation. If not specified, the animation (if requested) will begin with the simulation.
animation_end	*time*	approximate simulation time to end the animation. If not specified, the animation will end with the simulation.
width_animation_window	*length*	animation window width
height_animation_window	*length*	animation window height
camera_control_time_const	*time*	time constant of low-pass filter used to smoothen camera motion when following a moving body
force_vector_scaling	-	scaling factor for contact force vectors in animation window. Default value is 1.0 Remark: contact forces will be shown for each contact element, either in red, orange or in yellow. A red color indicates that the element is sticking to ground, an orange color indicates that the element is starting to slide but has not reached the maximum friction level, whereas a yellow color indicates that the element is sliding.

force_diagrams_scaling	-	scaling factor for running force/moment diagrams other than those in scopes, which can be scaled differently. Default value is 1.0
scaling_footprint_contour_plot	-	logarithm of scaling factor for footprint state shown in contour plot; only affects the contour plot resolution by changing the upper scale value. Default value is 0.0
mm_cp_per_mm_img	-	factor that scales size of the contact patch if shown in the full window mode. It specifies the true size of contact patch relative to the shown size, in terms of an integer number. Default value is 1
plot_bg_opaqueness	%	opaqueness of background of footprint image and running diagrams. 0% means transparent, 100% means completely opaque. Default value is 0
verbose	0, 1	write extra messages to the message and log file: **0** standard **1** verbose

`additional_output_file`	0, 1, 2, 4, 5, 7	generate additional plot output files (one per tire instance) with **FTire** states and other output signals, with different levels of detail, depending on value: **0** no additional output **1** most important plot signals, saved in Matlab compatible ascii-file **ftireiiii.mtl, iiii = 0001, 0002, ...** **2** all available plot signals, saved in Matlab-compatible ascii file **ftireiiii.mtl, iiii = 0001, 0002, ...** **4** most important plot signals, saved in binary file **ftireiiii.mtb, iiii = 0001, 0002, ...** **5** all available plot signals, saved in binary file **ftireiiii.mtb, iiii = 0001, 0002, ...** **7** important set of plot signals, saved in TYDEX-format file **ftireiiii.tdx, iiii = 0001, 0002, ...** Post-processing of output files • with cosin/ip • with Matlab scripts provided by cosin/tools • with user-written tools • with every tool supporting the TYDEX file format (if value is 7)
`record_file`	0, 1, 2	generate a file which enables the analysis and exact 'replay' of the current **FTire** simulation: **0** no rec file **1** save all information that is required for replay in binary file **ftire.rec** **2** save the comprehensive information in binary file **ftire.**rec. With this value, the rec-file might become very large in size; it is only required for debugging in certain special case

`refresh_preprocessing`	0, 1	unconditionally re-compute pre-processed data: **0** only if necessary **1** unconditionally
`append_pp_data`	0, 1	allow pre-processed data to be appended to the original data file: **0** no **1** yes, if available and data file is not write-protected
`save_contact_forces`	0, 1	generate extra output files **ftireiiii.cfo, iiii = 0001, 0002, ...** (one per tire instance), containing the distributed ground contact forces and several state variables of the tread, for further user-specific post-processing purposes. Ascii format is Matlab-compatible

save_belt_states	0, 1, 2	generate extra output files **ftireiiii.bso, iiii = 0001, 0002, ...** (one per tire instance), containing the belt states and distributed tire/rim contact forces, for further user-specific post-processing purposes. Ascii format is Matlab-compatible. Available options are: **0** no additional output **1** rotating (rim-fixed) cartesian coordinates **2** rotating (rim-fixed) cylindrical coordinates Definition of columns: **1..3** tangential ('longitudinal'), lateral, and radial belt element displacements [mm] **4** distance between belt element and rim center in forward (rolling) direction (meant only for testing and diagnosis purposes) [mm] **5** flag if belt element is close to road surface (meant only for testing and diagnosis purposes) **6..8** forces acting between belt elements and left rim flanges [N] **9..11** forces acting between belt elements and right rim flanges [N] **12..14** points of attack of forces acting between belt elements and left rim flanges [mm] **15..17** points of attack of forces acting between belt elements and right rim flanges [mm] **18** belt extension near belt elements [%]
save_geometry	div.	generate extra output files **ftireiiii.geo, iiii = 0001, 0002,** (one per tire instance), containing actual, distorted carcass and belt geometry, for further user-specific post-processing purposes. The value stores coded information about file format, resolution, coordinate system, surface coloring, and more. It is recommended to use cosin/tools to set the respective values

`save_surface`	div.	compute (and occasionally save to file) actual distorted tread surface geometry and velocity, for coupling with cosin/prm or 3rd-party CFD code. It is recommended to use cosin/tools to set the respective file format
`save_rim_forces`	0, 1	generate extra output files **ftireiiii.rfo, iiii = 0001, 0002, ...** (one per tire instance), with the circumferentially distributed interfacial forces between the sidewalls and the rim flanges. Available coordinate systems for the latter: **0** no output file **1** rim-fixed cartesian coordinates Definition of columns: **1..3** forces acting between belt elements and left rim flanges, in rim-fixed cartesian coordinates [N] **4..6** forces acting between belt elements and right rim flanges, in rim-fixed cartesian coordinates [N] **7..9** points of attack of forces acting between belt elements and left rim flanges, in rim-fixed cartesian coordinates [mm] **10..12** points of attack of forces acting between belt elements and right rim flanges, in rim-fixed cartesian coordinates [mm]

`save_tread_states`	0, 1, 2, 3, 4, 5, 6	generate extra output files **ftireiiii.eee, iiii = 0001, 0002, (**one per tire instance), containing tread state values, interpolated onto a regular and uniform grid. Available tread states are: **0** no output file **1** max. ground pressure (**eee** = **gpo**) **2** max. longitudinal shear stress (**eee** = **sxo**) **3** max. lateral shear stress (**eee** = **syo**) **4** mean ground pressure (**eee** = **gpo**) **6** mean longitudinal shear stress (**eee** = **sxo**) **6** mean lateral shear stress (**eee** = **syo**) The output files consist of a sequence of $(n_x \times n_y)$-matrices, each containing tread state values at the simulation time written in the respective matrix head-line (for example: % time=0.1 dx=0.0032 dy=0.0032 nx=101 ny=101). The tread state values are given in [MPa] over a regular (equally spaced) grid with $n_x \times n_y$ grid points, centered to wheel coordinates, with grid line distances as specified in the headerline (3.2 mm both for x and y in the example above)
`save_tread_depths`	0, 1	generate extra output files **ftireiiii.tdo, iiii = 0001, 0002, ...** (one per tire instance), containing the tread depths of all contact elements, for further user-specific post-processing purposes. The tread depth output values include the tread base height.
`save_contact_patch`	0, 1	generate extra output files **ftireiiii.cpo, iiii = 0001, 0002, ...** (one per tire instance), containing the contact patch boundary data points, for further user-specific post-processing purposes.

`save_wheel_envelope`	0, 1, 2, 3	generate extra output files **ftireiiii.shl, iiii = 0001, 0002, ...** (one per tire instance), containing the current simulation wheel envelope, for further user-specific post-processing purposes: **0** no wheel envelope output file **1** combine tire surface geometry in wheel-carrier fixed coordinates **2** combine tire surface geometry in car-body fixed coordinates (if available; in coordinates moving iwith the wheel n xy-plane otherwise) **3** combine tire surface geometry in global coordinates
`save_rgr_file`	0,1	after a simulation within a 3rd-party environment (using CTI), rgr road data files **ftireiiii.rgr, iiii = 0001, 0002, ...** (one per tire instance) will be created. The bounds of the rgr domain rectangle in x/y-plane are determined by the area covered by the respective wheel during simulation. As an example, these rgr files can be used when replaying the simulation outside the 3rd-party environment, where the original road evaluation library might not be available
`output_time_step`	*time*	approximate time step for the additional output of the plot signals, contact forces, and belt states. Note that **FTire** does not interpolate any output signals, nor, for efficiency reasons, does it try to exactly match the desired time step. For these reasons, the actual output steps might slightly differ from this value. Output time step is to be specified in the data file's time unit. If zero or not specified, each successful integration step will be saved.
`output_start`	*time*	approximate simulation time to begin the additional output. If not specified, the output (if requested) will begin with the simulation.
`output_end`	*time*	approximate simulation time to end the additional output. If not specified, the output will end with the simulation.
`ground_pressure_grid_line_dist`	*length*	grid-line distance of the ground pressure output in the gpo file (see above)

Document Revision: 2022-1-r26108

9 Undocumented Data Items

The following data items can be set in cosin/tools, but are yet undocumented. Several ones have self-explaining meaning. Documentation will be added in a coming release:

name in data file	**unit** in *cosin/io* data file	**unit** in **TeimOrbit** data file	***symbol***
Jacobian_CP_fraction	-	-	n/a
Jacobian_CP_fraction_RT	-	-	n/a
camber_stiffness	-	-	n/a
camera_distance_to_rim	-	-	n/a
camera_shift	-	-	n/a
contact_processor_bound_RT	-	-	n/a
contact_processor_cycle_length	-	-	n/a
cornering_stiffness	-	-	n/a
damping_tread_rubber	-	-	n/a
fifth_deflection_p2	-	-	n/a
footprint_plot	-	-	n/a
force_diagrams_magn_factor	-	-	n/a
force_diagrams_time_interval	-	-	n/a
in_plane_bend_stiffn	-	-	n/a
lat_dynamic_stiffening	-	-	n/a
lat_hysteresis_force	-	-	n/a
maximum_angle_increment_RT	-	-	n/a
maximum_time_step_RT	-	-	n/a
number_belt_bend_shape_funct_RT	-	-	n/a
number_belt_segments_RT	-	-	n/a
number_blocks_per_belt_segm_RT	-	-	n/a
number_tread_strips_RT	-	-	n/a
output_end	-	-	n/a
point_of_ref	-	-	n/a
pp_max_cycles	-	-	n/a
radial_hysteresis_force_2	-	-	n/a
radial_hysteresis_force_3	-	-	n/a
radial_hysteresis_force_4	-	-	n/a
radial_hysteresis_force_5	-	-	n/a
radial_hysteretic_stiffening_2	-	-	n/a
radial_hysteretic_stiffening_3	-	-	n/a
radial_hysteretic_stiffening_4	-	-	n/a
radial_hysteretic_stiffening_5	-	-	n/a
rim_paint_gray_scale	-	-	n/a
running_diagrams	-	-	n/a
sidewall_texture_ar_corr	-	-	n/a
sidewall_texture_radial_shift	-	-	n/a
sidewall_texture_repetition	-	-	n/a
sidewall_texture_scaling	-	-	n/a
sidewall_texture_transp_thresh	-	-	n/a
slip_stiffness	-	-	n/a
stiffness_progr_tread_rubber	-	-	n/a

sun_position	-	-	n/a
tang_dynamic_stiffening	-	-	n/a
tang_hysteresis_force	-	-	n/a
third_deflection_p2	-	-	n/a
tire_gray_scale	-	-	n/a
tire_heating_at_10_p_slip_low_v	-	-	n/a
tire_lat_stiffn_progr	-	-	n/a
tire_long_stiffn	-	-	n/a
tire_long_stiffn_progr	-	-	n/a
tireside	-	-	n/a
tread_gluing_force_percentage	-	-	n/a
tread_heating_at_ref_slip_high_v	-	-	n/a
wheel_load_cornering_stiffn_p2	-	-	n/a
wheel_load_lat_stiffn	-	-	n/a
wheel_load_lat_stiffn_p2	-	-	n/a
wheel_load_long_stiffn_p2	-	-	n/a
wheel_load_tors_stiffn	-	-	n/a
wheel_load_tors_stiffn_p2	-	-	n/a

10 TYDEX-Conformal Output Signals

FTire provides a set of TYDEX-conform output signals, containing TYDEX standard signals 1..25, and some additional signals, as listed in the table below. A much more comprehensive collection of output signals is provided in the 'additional output file' with the file extension mtl or mtb, see chapter 7.

signal	***description***	***unit***
1-6	tire forces and torques, expressed in TYDEX W-axis	*N, Nm*
7	slip angle	*rad*
8	longitudinal slip	-
9	camber angle	*rad*
10-25	*- not used -*	
26-28	geometrical center of contact patch in global coordinates	*m*
29-37	3x3 transformation matrix from contact tangential plane to global coordinates (storage column-wise)	-
38-43	tire forces and torques, expressed in ISO axis system	*N, Nm*
44	tire deflection	*m*
45	vertical rim center velocity	*m/s*
46	longitudinal slip velocity at contact point	*m/s*
47	lateral slip velocity at contact point	*m/s*
48	longitudinal rim center velocity	*m/s*
49	maximum belt radius after inflation	*m*
50	rim angular velocity relative to wheel carrier (ABS signal)	*rad/s*
51-77	*- do not use, only for interfacing to certain 3rd-party software -*	
78	mean road friction factor in contact patch	-
79	tire deflection	*mm*
80	slip angle	*deg*
81	longitudinal slip	*%*
82-87	tire forces and torques, expressed in TYDEX C-axis system	*N, Nm*
88	rolling loss	*N*
89	maximum belt-to-rim contact intrusion	*m*
90	dynamic rolling radius	*m*
91-96	tire forces and torques, expressed in ISO axis system (same as signals 38-43)	*N, Nm*
97-119	*- do not use, only for interfacing to certain 3rd-party software -*	

11 Model Changes Log and Compatibility Mode

The table below lists the important model enhancements and bug-fixes, added since 2015/07/01, that can be deactivated

1. either by setting the 'compatibility mode' parameter '`only_use_FTire_updates_until`' in the data file
2. or by selecting **'emulate old cosin version..'** in cosin's launchpad
3. or by setting one of the environment variables COSIN_COMPVERS = YYYYQ (or YYYY-Q) or COSIN_COMPDATE = YYYYMMDD (or YYYY-MM-DD).

A selection in the launchpad will take precedence over values set by the environment variables.

Obviously, an old software version will not contain any enhancements with a compatibility mode date later than the final release date of the version. Setting a compatibility date will activate all FTire enhancements or modifications introduced until (and including) this date.

Full compatibility with older versions is comprehensively tested for all supported compatibility dates and versions.

In case a compatibility **version** (or **'emulated version'**) rather than a compatibility **date** is selected in the launchpad, the latest release date of any patch of the selected version will be used. If both in the data file **and** in the launchpad a compatibility date is selected, the older one of both is used.

It is generally recommended not to use the compatibility mode unless there is a good reason to do so. The compatibility date or version is no „version string" related to the actual library version used. Rather, in any case, the newest installed cosin library version is used, only 'simulating' the exact behavior of an older version at a given date.

It is not required to tag older input data if used with a more recent library version. Nor is the compatibility mode intended to fix model stability problems. On the contrary, in general, the most recent version using **all** enhancements and bug-fixes is the best. If ever you experience a certain solver issue that seems to be cured in compatibility mode, please contact support@cosin.eu for an investigation of the problem.

comp. date	since version	model changes
2021/12/31	2022−1	(1) contact elements strip distance made constant even for laterally extremely curved belt geometries (2) new option ds=<ds> in rgr data file header, to enforce exact synchronization between rgr and center-line grid line distances (3) small dependency of forces, torques, and total power loss output on internal time step reduced
2021/09/23	2021−4	(1) improved placement of contact element strips, observing groove geometry (2) computation of tread center temperature fixed (3) optional optimized placement of numerical tread blocks introduced, taking onto account tread pattern geometry if applicable
2021/06/18	2021−3	(1) made sure rim-to-road contact works properly for road types 'cleat' both in TeimOrbit and in cosin/io format (2) dynamic distributed air cavity reaction forces on rim flanges taken into account in rfo output file (3) maximum lateral and radial elastic rim deformation in simplified internal flexible rim model had been mixed up in older versions

		(4) avoid purely numerical air vibration excitation through segment periodicity at small non-zero rolling speeds by increasing numerical damping in respective wheel angular velocity range (5) plastic rim deformation, if any, taken into account in computation of rim-to-road contact geometry (6) made sure even extreme contact pressure values are treated in the same way in all speed modes
2021/03/31	2021−2	(1) pre-processsing of incompletely defined cross section spline data which did not work in certain rare cases fixed (2) rim torsional stiffness activated in simplified flex rim model (3) numerics in case of bottoming cases with extremely large contact forces made more robust (4) making sure geometry item 'belt layer thickness', if given, is taken into account even with higher speed modes
2021/01/01	2021−1	(1) TPMS acceleration signals computation accuracy improved, by evaluating each internal time step rather than each communication time step (2) improved and smoothened expression of limiting lateral side-wall stretching stiffness (3) fix of operating conditions usage in FTire/sim: if one or both had been set non-default, actual gas temperature and actual tread depth might have been interchanged
2020/10/01	2020−4	(1) improved accuracy of TYDEX W moments computation at high speed (2) steady-state computation on flat road simulating flat-trac rather than moving rim, for improved accuracy of surface geometry output
2020/06/22	2020−3	(1) computation of tread rubber normal stiffness at extreme deflections fixed (2) output sequence in rfo-files changed to 'rim-fixed', rather than beginning with lowest-most segment
2020/03/16	2020−2	(1) contour spline data smoothing (if ordered by smoothing polynomial degree > 0) now applies to both tread surface and carcass data (2) made sure side-wall geometry is correctly pre-processed even for extreme tire si (3) filling gas mass roughly estimated in either c (4) more accurate side-wall distortion geometry at extreme tire displacement
2019/12/26	2020−1	(1) fix of stochastic road profile generator for a given ISO road class. Previously, profile values were too small by a factor of approximately 2.5 (2) optionally taking into account more accurately belt curvature near leading and trailing edge of contact patch improves dynamic rolling radius accuracy (this model feature can be deactivated even in cosin versions higher or equal 2020-1 by using the parameter high_precision_contact_comp)
2019/09/30	2019−4	(1) C-axis wheel load made controllable in FTire/sim and FTire/fit, in addition to default W-axis wheel load (2) improved accuracy of footprint shape approximation in FTire/fit when using measured contact pressure distributions together with activated tread pattern model (3) accelerated contact computation in case of exactly flat roads and higher speed modes (4) properly take into account optional shift and scale of independent variable in computation of FTire/fit least squares score values
2019/06/24	2019−3	(1) optional file output of surface velocities and accelerations in first step fixed (2) bug in barrier computation of 2D road type tilt_table fixed (3) fix of computation of TYDEX signal 'tire deflection' during statics (4) ambient temperature rather than road surface temperature unconditionally used as contact body temperature if contact pressure is zero (5) encryption supported now for RGR files that contain a non-trivial centerline
2019/03/25	2019−2	(1) improved accuracy of ground pressure interpolation in case of tread pattern usage

		(2) more accurate viscosity in air cavity model (3) slightly improved convergence, robustness, and repeatability in statics and steady-state analysis (4) fix of occasional incomplete computation of signal RMS values for output
2018/12/21	2019−1	(1) slightly improved pre-processing convergence robustness (2) avoiding underflow floating point exceptions degrade performance in rare cases
2018/10/01	2018−4	(1) optional variable lateral belt bending stiffness introduced (2) belt geometry smoothing polynomial restricted to points sufficiently far away from sidewalls (3) improved numerical accuracy during bottoming events (4) inaccurate rim-to-road distance computation in rim-to-road misuse element fixed (5) computation of output signal 'dynamic rolling radius' in case of moving road surface fixed (6) inaccurate computation of bevel edges of oblique cleat on drum fixed
2018/06/12	2018−3	(1) tire sidewalls prevented from being stretched too much, by introducing strongly progressive radial stiffness in stretched condition (2) inflation pressure, tire temperature, and tread depth changes properly taken into account now in loaded modal analysis (3) orientation angle of oblique cleats consistently defined counter-clockwise, using new data item 'cleat_angle' instead of deprecated items 'direction' or 'cleat_direction' in all relevant road types (both in cosin/io and TeimOrbit format for drum, flatbelt, cleat, and plank) (4) slightly too small scaling of imperfection kind '2D tread gauge variation' fixed (5) improved rim-to-road vertical stiffness model
2018/03/06	2018−2	(1) approximate computation of sidewall geometry in undeformed and deformed state improved (2) in speed mode 6, rgr roads with local transformation are accepted (3) further acceleration of rgr road evaluation w/o and with curved centerline (4) interior volume output fixed in case of detailed air cavity model activation (5) more accurate computation of vibrating air cavity mass reaction forces on rim and belt
2017/11/15	2018−1	(1) improved contact patch triangulation algorithm in case of large tread shear displacements (2) more robust computation of steady-state contact patch geometry and contact forces if tread pattern is given by image (3) made sure tread pattern repetition follows same rules and is exactly the same in all speed modes (4) more accurate computation of sidewall height, taking into account exact ETRTO compliant rim flange geometry (5) numerically more robust steady-state tire initialization in case of strongly inclined roads (6) speed mode 6 now provides minimum TYDEX output and accelerated rgr evaluation
2017/09/02	2017−4	filling gas and tread surface temperature can be set in the tir-file even if the thermal model is not active and the respective CTI function is not called
2017/07/31	2017−4	(1) stiffness hysteresis (which might falsify results) automatically disregarded during linearization (2) vertical stiffness comparison of rolling road in FTire/fit made independent on speed-dependent maximum radius estimation

		(3) more accurate approximation of 'mean contact point' in case of very steep road surface steps like potholes etc, resulting in more accurate computation of peak output forces (only TYDEX and ISO systems) during related events. Only additional output signals, but no forces passed over to calling model are affected
2017/06/20	2017−3	(1) pressure dependency of belt circumferential torsion and twist stiffness in case of direct specification made effective (2) faster linearization if certain structural degrees of freedom are not used (3) automatic modification of all structural degrees of freedom, following abrupt rim angular position changes during restart etc. (4) more accurate eigenfrequency computation in case of a loaded rolling tire
2017/04/16	2017−3	(1) maximum lateral belt element displacement during extreme loading conditions is limited by the sidewall length (2) more accurate computation of the rim rotation angle if the camber angle is close to 90 deg (only alternative interface affected) (3) numerical damping (BDF parameter) in air vibration model solver made selectable (4) refined modeling of lateral belt bending stiffness degressivity
2017/03/08	2017−2	(1) more accurate formulation of the optional longitudinal tread stiffness coupling (2) more accurate eigenfrequency computation in case of a loaded rolling tire
2016/11/28	2017−2	(1) more accurate evaluation of the 'cleat on rotating drum' road profile (2) avoidance of a potential convergence issues when approximating 'contact point' output signal in case of very sharp and high obstacles
2016/10/19	2017−2	more accurate computation of the local velocity which effects the cooling of the sidewalls and tread, correct for all wheel speeds over ground and all rotational velocities
2016/07/22	2017−1	maximum values of the radial and horizontal tread block deflection and deflection velocity modified, thus improving stability in extreme misuse situations
2016/06/20	2016−3	(1) thermal model: speed-dependent cooling terms if tire is operated on drum improved (2) cooling by heat radiation introduced
2016/04/08	2016−2	thermal model now completely activated even if no friction coefficient dependency on temperature is specified, but thermal mode is requested anyway
2016/03/18	2016−2	computation of certain rarely used auxiliary output signals fixed (percentual contribution of friction characteristic intervals)
2016/02/02	2016−2	improved estimation of dynamic rolling radius for internal use, taking into account all temporarily changing operating conditions
2016/01/09	2016−2	load ranges used in the identification of static and steady-state properties in FTire/fit made dimensionless and independent on non-physical data
2015/11/03	2016−1	(1) several belt bending stiffness parameters are now allowed to be exactly 0. before, they had been set to a non-zero default value if specified with 0 (2) new refined measure for footprint validation, replacing the simple length-/width approximation in FTire/fit (3) improved computation of lateral stiffness during preprocessing. Previous algorithm might have depended on certain extra conditions
2015/10/17	2016−1	do not modify drum surface normal near cleats in FTire/fit cleat test output TYDEX W computation
2015/09/10	2015−4	do not fill TYDEX output array with undocumented standard output signals in case array is larger than necessary
2015/08/11	2015−4	TYDEX output signal #49 (dynamic rolling radius) might have been computed with reduced accuracy, depending on availability of additional preprocessed data. Behavior now is made independent on previous usage of the data file, controlled by flag STATICS_ACCURACY

2015/07/15	2015−4	(1) minor issue in longitudinal and lateral stiffness computation during pre-processing fixed. Most likely no impact on used data-files (2) accelerated computation of contact moment in y-direction (only additional output signal) in real-time mode fixed (3) tread pattern bitmap, if given any, made repeating exactly periodic

Compatibility mode will not switch formal changes or additional output channels back to older library versions. Refer to the following table for a list of output channels that are tested for result compatibility at the respective compatibility date. Channels not listed can be subject to extensions or changes not covered by the compatibility mode.

signal name	**unit**
fore-aft force (W-axis)	N
side force (W-axis)	N
wheel load (W-axis)	N
overturning torque (W-axis)	Nm
rolling torque (W-axis)	Nm
aligning torque (W-axis)	Nm
fore-aft force FP (W-axis)	N
side force FP (W-axis)	N
wheel load FP (W-axis)	N
overturning torque FP (W-axis)	Nm
rolling torque FP (W-axis)	Nm
aligning torque FP (W-axis)	Nm
fore-aft force (ISO)	N
side force (ISO)	N
wheel load (ISO)	N
overturning torque (ISO)	Nm
rolling torque (ISO)	Nm
aligning torque (ISO)	Nm
x-position rim center	m
y-position rim center	m
z-position rim center	m
x-velocity rim center	m/s
y-velocity rim center	m/s
z-velocity rim center	m/s
drag force	N
cornering force	N
tire deflection velocity	m/s
max. tread deflection	mm
actual pressure	bar
pneumatic trail	mm
pneumatic scrub	mm
total power loss	kW
power loss in plies	kW
power loss in tread	kW
power loss through road friction	kW
power loss through damping	kW
rolling loss	N
rolling resistance coefficient	-
brake slip	%
brake force	N
max. ground pressure	MPa
mean ground pressure	MPa
max. sliding velocity	m/s
mean sliding velocity	m/s

gyroscopic overt. torque	Nm
gyroscopic align. torque	Nm
footprint length	mm
footprint width	mm
footprint area	m^2
long. shift of Fz point of attack	mm
lat. shift of Fz point of attack	mm
footprint center line torsion	deg
footprint center line curvature	1/m
ply-steer moment	Nm
max. side-wall/road intrusion	mm
max. rim/road intrusion	mm
max. rim/belt intrusion	mm
rel. size of rim flange to belt contact area	%
rim/road contact force x	N
rim/road contact force y	N
rim/road contact force z	N
tire/rim slip	deg
rel. size of sliding area	%

Document Revision: 2022-1-r26108

FTire Model Data

click respective page to view parameter description