Phase Plane Generation with Linear and Non-Linear Tyre Models

Mc Glinchey, Odhrán
26-08-2025

Contents

Nomenclature

Vehicle Parameters

m	Vehicle mass	kg
I_z	Yawing Moment of Inertia (About Z Axis)	kgm^2
V	Forward velocity	$\frac{m}{s}$
β	Vehicle Side Slip	rad
r	Yawing Velocity	$\frac{\text{rad}}{\text{s}}$
δ	Steering input angle	rad or °
l_f , l_r	Distance from CoG to front and rear axle	m
F_{zf} , F_{zr}	Vertical load on front and rear axle	N
F_{z0}	Nominal (reference) tyre load	N
γ	Camber angle	rad or °

Linear Tyre Model Parameters

$C_{\alpha f}$, $C_{\alpha r}$	Front and rear cornering stiffness	$\frac{N}{rad}$
α_f , α_r	Front and rear slip angles	rad
F_{yf} , F_{yr}	Front and rear lateral tyre forces	N

Nonlinear Pacejka Tyre Model Parameters

α	Slip angle	rad
α_y	Adjusted slip angle	rad
$S_{h,y}$	Horizontal Shift	$-$
$S_{v,y}$	Vertical Shift	N
B_y	Stiffness factor	$-$
C_y	Shape factor	$-$
D_y	Peak factor	N
E_y	Curvature factor	$-$
K_y	Cornering stiffness	$\frac{N}{rad}$
μ_y	Friction Coefficient	$-$
df_z	Normalised load variation	$-$

Pacejka Fitting Coefficients

p_{Cy1}	Shape factor scaling constant	$-$
$p_{Dy1}, p_{Dy2}, p_{Dy3}$	Friction scaling coefficients (nominal, load sensitivity, camber sensitivity)	$-$
$p_{Ey1}, p_{Ey2}, p_{Ey3}, p_{Ey4}$	Curvature coefficients (load, camber, asymmetry terms)	$-$
$p_{Ky1}, p_{Ky2}, p_{Ky3}$	Cornering stiffness scaling, load stiffness control, camber sensitivity	$-$

Introduction

Understanding vehicle stability requires models that accurately capture the essence of lateral dynamics. The 2-DOF bicycle model is an industry-standard workhorse for this purpose, using sideslip angle (β) and yaw rate (r) to describe the vehicle's state.

The phase plane plot of these variables is an intuitive tool for engineers, visually mapping out the vehicle's tendency to return to equilibrium or spin out.

However, the accuracy of this map depends entirely on how well the model describes tyre forces. While a linear model is useful for small slips, it breaks down during aggressive manoeuvres where tyres operate in their nonlinear, saturated region. The Pacejka "Magic Formula" model is therefore essential for a more realistic simulation.

This report provides a practical comparison of these two modelling approaches. We generate and analyse phase portraits for both a linear and a nonlinear Pacejka-based bicycle model. The results clearly show the limitations of the linear model and underscore the critical importance of incorporating nonlinear tyre dynamics for predicting true vehicle behaviour, especially at the limits of handling.

Aim / Objective

1. **Formulate a linear tyre vehicle model.**
    - Develop a baseline bicycle model using a linear tyre approximation for comparison.
2. **Extend the model to include nonlinear tyre behaviour.**
    - Implement a nonlinear (Pacejka-based) tyre model to capture more realistic dynamics.
3. **Simulate and compare system responses.**
    - Generate phase portraits and trajectory simulations for both linear and nonlinear models to visualise stability behaviour.
4. **Identify and classify equilibrium states.**
    - Determine equilibrium points of the system and assess their stability under different operating conditions.
5. **Investigate parameter sensitivity.**
    - Explore the effect of vehicle and tyre parameters (e.g. cornering stiffness, steering input, speed) on transitions between stable and unstable behaviour.
    - Translate the analysis into an understanding of how vehicle dynamics can be manipulated to improve handling and control.

Theory

The Vehicle Model

A standard ISO-coordinate bicycle model was selected for this study. This well-established two-degree-of-freedom (2-DOF) model provides sufficient fidelity to capture the target vehicle states while maintaining simplicity.

Assumptions

- Tyres left/right at each axle are identical → symmetry.
- Forces act at a single "equivalent tyre" per axle.
- No roll or pitch motion (flat plane model)
- Reference point ***O*** is the Centre of Gravity
- Small Steering angle $\sin\delta \approx \delta, \tan\delta \approx \delta, \cos\delta \approx 1$
- No Aero Loading

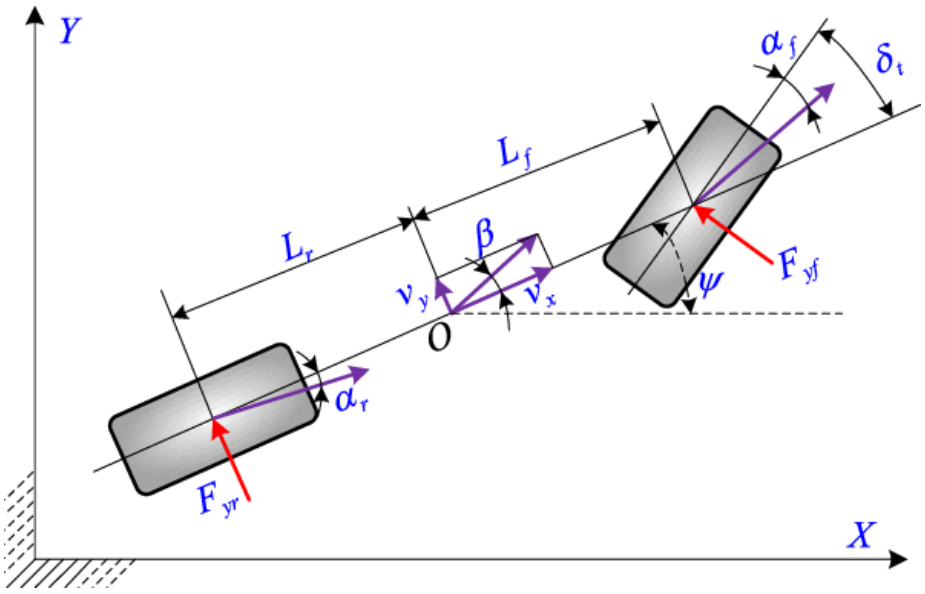

Figure 1 - Single Track Vehicle Model (ISO)

Expressions Relating v_y ,v_x and β

Using the velocity vectors, it can simply be stated that:

$$\tan\beta = \frac{v_y}{v_x} \tag{1}$$

Applying small angle approximation ($\tan\beta = \beta$) the equation becomes:

$$\beta = \frac{v_y}{v_x} \tag{2}$$

Because we are using the small angle approximation, it can be also assumed that:

$$v_x \approx V \tag{3}$$

Therefore, our equations become:

$$v_y = V\beta \tag{4}$$

Meaning the lateral acceleration is defined as:

$$\dot{v}_y = V\dot{\beta} \tag{5}$$

Lateral Dynamics

Using Newtons second law and balancing forces in the y direction

$$ma_y = \sum Y = F_{yr} + F_{yf}\cos\delta_t = F_{yr} + F_{yf} \tag{6}$$

Applying small angle approximation $\cos\delta_t \approx 1$

$$a_y = \dot{v}_y + rv_x \tag{7}$$

rv_x represents the cetripetal acceleration, and r being the yawing velocity

$$m(\dot{v}_y + rv_x) = F_{yr} + F_{yf} \tag{8}$$

Using the previously defined relationship for v_y and v_x from equations (3) and (5)

$$m(V\dot{\beta} + rV) = F_{yr} + F_{yf} \tag{9}$$

$$mV(\dot{\beta} + r) = F_{yr} + F_{yf} \tag{10}$$

Yaw Dynamics

$$I_z\dot{r} = \sum N = -L_rF_{yr} + L_fF_{yf}\cos\delta_t = -L_rF_{yr} + L_fF_{yf} \tag{11}$$

Applying small angle approximation $\cos\delta_t \approx 1$

$$I_z\dot{r} = -L_rF_{yr} + L_fF_{yf} \tag{12}$$

Slip Angle Geometry

Side slip angles of both front and rear tyres, can be described using equations (13) and (14):

$$\alpha_r = -\beta + \frac{L_r\, r}{V} \tag{13}$$

$$\alpha_f = \delta - \beta - \frac{L_f\, r}{V} \tag{14}$$

Linear Tyre Model Assumption

While nonlinear models such as the Pacejka "Magic Formula" can capture tyre characteristics across the full slip range, they are complex and extremely nonlinear. For the purpose of stability analysis and phase plane studies, a simplified **linear tyre model** is often employed. This section outlines the assumptions and reasoning behind this choice.

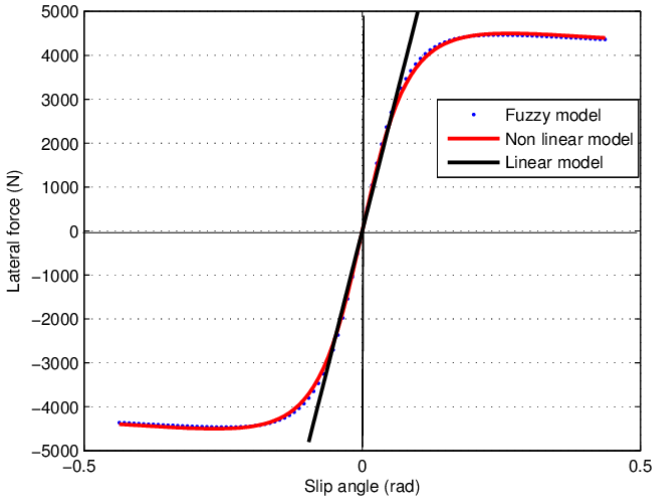

Figure 2 - Linear Tyre Model Assumption

Reasoning for Using the Linear Model

- The linear tyre model allows the equations of motion to be expressed in **linear state-space form.**
- Valid in the Normal Driving Range – manoeuvres typically occur within small slip angles, where tyre behaviour is approximately linear.
- In this range, the linear model accurately predicts lateral forces and captures stability trends.

Limitations

- The model is not valid at high slip angles (beyond ~5–10°).
- It cannot capture tyre saturation, limit understeer/oversteer, or combined slip effects.
- Nonlinearities such as aligning torque reversal and force asymmetry are excluded.

The lateral forces generated when assuming a linear tyre model can be represented as follows, where the cornering stiffness is N/rad and the slip angle still being represented in radians.

$$F_{yf,tyre} = C_{\alpha f,tyre} \cdot \alpha_f \tag{15}$$

$$F_{yr,tyre} = C_{\alpha r,tyre} \cdot \alpha_r \tag{16}$$

Since the tyre model describes a single tyre, the calculated forces must be doubled for the bicycle model. Representative of axial lateral force

$$F_{yf} = 2\, C_{\alpha f,tyre} \cdot \alpha_f \qquad or \qquad F_{yr} = C_{\alpha f} \cdot \alpha_f \tag{17}$$

$$F_{yr} = 2\, C_{\alpha r,tyre} \cdot \alpha_r \qquad or \qquad F_{yr} = C_{\alpha r} \cdot \alpha_r \tag{18}$$

Where $C_{\alpha f,r}$ represents the stiffness for the entire axle

Linear State Space Equation of Bicycle Model with Linear Tyres

$$\dot{\beta} = \frac{F_{yr} + F_{yf}}{mV} - r \tag{19}$$

$$\dot{r} = \frac{-L_r F_{yr} + L_f F_{yf}}{I_z} \tag{20}$$

Rearranging equations (10) and (11), implementing our tyre force notation from (15) and (16) we can simply write:

$$\dot{\beta} = \frac{C_{\alpha f} \cdot \alpha_f + C_{\alpha r} \cdot \alpha_r}{mV} - r \tag{21}$$

$$\dot{r} = \frac{-L_r C_{\alpha r} \cdot \alpha_r + L_f C_{\alpha f} \cdot \alpha_f}{I_z} \tag{22}$$

If the vehicle was to sustain a constant slip angle (e.g. steady state cornering), then $\dot{\beta}$ & $\dot{r} = 0$

Transforming equations into the standard state-space representation is advantageous for investigating the properties of linear systems.

$$\dot{x} = \begin{bmatrix} \dot{\beta} \\ \dot{r} \end{bmatrix} \tag{23}$$

$$\dot{x} = Ax + Bu \tag{24}$$

Were the control input, u, being **the front steering angle $\boldsymbol{\delta}$.**

Inserting our slip angle equations and rearranging we get formula (26)

$$\dot{\boldsymbol{\beta}} = \frac{C_{\alpha f}\left(\delta - \beta - \frac{L_f\, r}{V}\right) + C_{\alpha r}\left(-\beta + \frac{L_r\, r}{V}\right)}{mV} - r \tag{25}$$

$$\dot{\boldsymbol{\beta}} = -\frac{(C_{\alpha f} + C_{\alpha r})}{mV} \cdot \boldsymbol{\beta} + \left(\frac{\left(C_{\alpha r} L_r - C_{\alpha f} L_f\right)}{mV^2} - 1\right)\boldsymbol{r} + \frac{C_{\alpha f}}{mV} \cdot \boldsymbol{\delta} \tag{26}$$

Repeating the same for the yaw acceleration

$$\dot{\boldsymbol{r}} = \frac{-L_r C_{\alpha r}\left(-\beta + \frac{L_r\, r}{V}\right) + L_f C_{\alpha f}\left(\delta - \beta - \frac{L_f\, r}{V}\right)}{I_z} \tag{27}$$

$$\dot{\boldsymbol{r}} = \frac{\left(L_r C_{\alpha r} - L_f C_{\alpha f}\right)}{I_z}\,\boldsymbol{\beta} - \frac{\left(L_f^2 C_{\alpha f} + L_r^2 C_{\alpha r}\right)}{I_z V}\boldsymbol{r} + \frac{L_f C_{\alpha f}}{I_z}\boldsymbol{\delta} \tag{28}$$

Using equations () and () are **linear functions** of $\boldsymbol{\beta}, \boldsymbol{r}$ and $\boldsymbol{\delta}$ the **linear state space model** can be rewritten as:

$$\boldsymbol{x} = \boldsymbol{A}[\beta \quad r]^T + \boldsymbol{B}(\delta) \tag{29}$$

$$\boldsymbol{A} = \begin{bmatrix} -\frac{(C_{\alpha f} + C_{\alpha r})}{mV} & \left(\frac{\left(C_{\alpha r} L_r - C_{\alpha f} L_f\right)}{mV^2} - 1\right) \\ \frac{\left(L_r C_{\alpha r} - L_f C_{\alpha f}\right)}{I_z} & -\frac{\left(L_f^2 C_{\alpha f} + L_r^2 C_{\alpha r}\right)}{I_z V} \end{bmatrix}, \qquad \boldsymbol{B} = \begin{bmatrix} \frac{C_{\alpha f}}{mV} \\ \frac{L_f C_{\alpha f}}{I_z} \end{bmatrix}$$

Steady State Equilibrium

We are looking at the **stability of the open-loop system**. That means: if the car gets a small disturbance (small sideslip β or yaw rate r), does it come back to straight-line motion, or does the disturbance grow?

The equilibrium occurs when $\dot{\beta}$ & $\dot{r} = 0$

Eigenvalues can be used to represent the stability of the system represented by

$$\det(A - \lambda I) = 0 \tag{30}$$

Characteristic polynomial

$$\boldsymbol{\lambda}^2 - tr(A)\boldsymbol{\lambda} + \det(A) = 0 \tag{31}$$

Where the following terms are defined as

Trace

$$tr(A) = a_{11} + a_{22} = -\left(\frac{(C_{\alpha f} + C_{\alpha r})}{m} \pm \frac{\left(L_f^2 C_{\alpha f} + L_r^2 C_{\alpha r}\right)}{I_z}\right)\frac{1}{V} \tag{32}$$

Determinant

$$\det(A; V) = \frac{A_0}{V^2} + B_0 \tag{33}$$

$$\mathrm{A}_0 = \frac{(C_{\alpha f} + C_{\alpha r})\left(L_f^2 C_{\alpha f} + L_r^2 C_{\alpha r}\right) - (L_r C_{\alpha r} - L_f C_{\alpha f})^2}{m I_z}, \qquad B_0 = \frac{(L_r C_{\alpha r} - L_f C_{\alpha f})}{I_z}$$

Eigenvalues

$$\lambda_{1,2}(V) = \frac{tr(A)}{2} \pm \sqrt{\left(\frac{tr(A)}{2}\right)^2 - \det(A; V)} \tag{34}$$

- If $\left(\frac{tr(A)}{2}\right)^2 - \det(A; V) > 0$: two real eigenvalues
- If $\left(\frac{tr(A)}{2}\right)^2 - \det(A; V) = 0$: one repeated real eigenvalue
- If $\left(\frac{tr(A)}{2}\right)^2 - \det(A; V) < 0$: two complex conjugate eigenvalues

Phase Plane Plot using Linear Tyre Model

The **phase plane** plots the system states (β, r) against each other, with trajectories showing how the system evolves over time from different initial conditions.

- The **vector field** $(\dot{\beta}, \dot{r})$ illustrates the instantaneous direction of motion at each point.
- The **trajectories** show how the states converge to an equilibrium (steady-state cornering) or diverge if the system is unstable.
- The equilibrium point is found by solving $\dot{\beta} = \dot{r} = 0$ for given δ.

For a stable, understeering vehicle, trajectories spiral or curve toward a unique equilibrium point. For an oversteering vehicle at high speed, the equilibrium may lose stability, visible in the phase portrait as trajectories diverging away.

An example phase portrait using the following parameters:

Vehicle Parameters

m = 220 # mass in kg

Iz = 120.0 # yaw inertia in kgm^2

lr = 0.8 # CoG to front axle in m

lf = 0.8 # CoG to rear axle in m

Caf = 30000 # front cornering stiffness

Car= 30000 # rear cornering stiffness

Speed and Steering

V = 12 # speed in m/s

delta_deg = 1 # steering input angle (degrees)

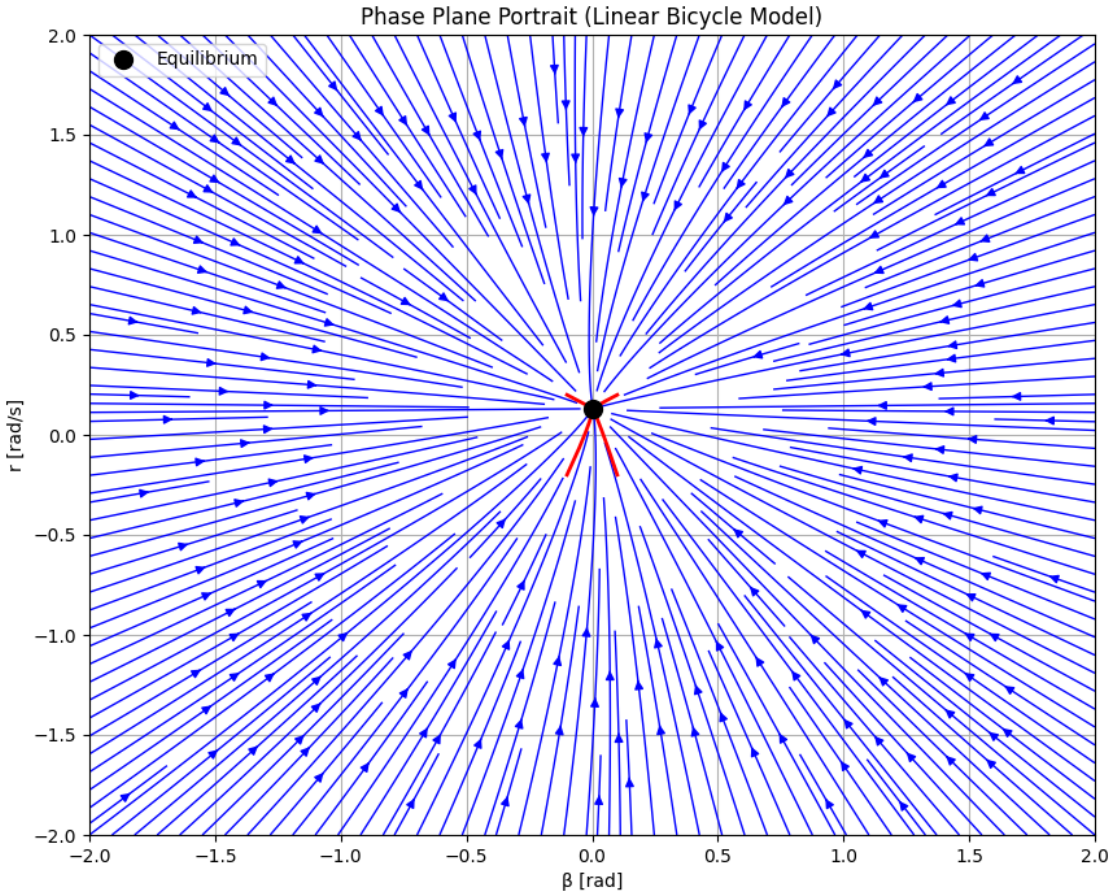

Figure 3 - Phase Portrait using Linear Tyre Assumption

Non-Linear Tyre Model Assumption

To accurately capture the vehicle's behaviour across its full operational range, particularly at the limits of handling, a non-linear tyre model is essential. This study employs a simplified version of the Pacejka "Magic Formula" tyre model under the following fundamental assumptions:

Pure Slip Conditions: The model operates under the assumption of *pure lateral slip*. This means the effects of longitudinal forces, combined slip, and transient tyre dynamics (relaxation length) are neglected.

- **Combined Slip Effects:** In practice, the simultaneous presence of lateral and longitudinal slip (e.g., during braking-in-a-turn or acceleration out of a corner) reduces the maximum potential force that a tyre can generate in either direction. This force saturation effect under combined slip is **not modelled** in this analysis.
- **Aligning Torque Effects:** The self-aligning moment (M_z), which is the torque that causes the steering wheel to centre itself, is a critical component of driver feel and vehicle stability. Its calculation and influence on the yaw moment equation are **excluded** from this model.
- **Transient Dynamics:** The tyre forces are calculated instantaneously for a given set of inputs, implying the tyre has reached a steady-state condition. Transient effects related to the relaxation length, which cause a phase lag between slip angle and force generation, are **neglected**.

Steady-State Behaviour: The tyre forces are calculated instantaneously for a given set of inputs, implying that the tyre has reached a steady-state condition.

The core of the non-linear tyre model is the Pacejka Magic Formula, which uses a combination of trigonometric functions to empirically fit measured tyre data.

Pure Slip Conditions – Lateral Force (Steady State)

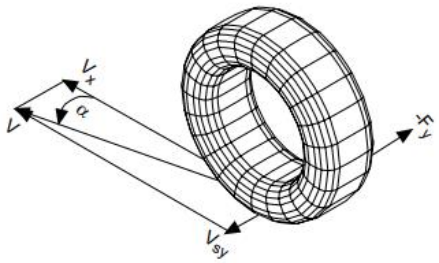

Figure 4 – Slip angle Lateral Tyre Force Visual

$$F_y = F_{y0}(\alpha, \gamma, F_z) \tag{35}$$

$$F_{y0} = D_y \cdot \sin[C_y \cdot \arctan(B_y\alpha_y - E_y\left(B_y\alpha_y - \arctan(B_y\alpha_y)\right))\,] + S_{v-y} \tag{36}$$

Where:

$$\alpha_y = \alpha + S_{h-y} \tag{37}$$

$$C_y = p_{C_{y1}} \tag{38}$$

$$D_y = \mu_y \cdot F_z \tag{39}$$

$$E_y = \left(p_{E_{y1}} + p_{E_{y2}} \cdot df_z\right) \cdot \left\{1 - \left(p_{E_{y3}} + p_{E_{y4}}\gamma_y\right) sgn(\alpha_y)\right\} (\leq 1) \tag{40}$$

$$B_y = \frac{K_y}{C_y D_y} \tag{41}$$

$$K_y = p_{K_{y1}} \cdot F_{z0} \cdot \sin\left[2 \arctan\left(\frac{F_z}{\left(p_{K_{y2}} F_{z0}\right)}\right)\right] \cdot \left(1 - p_{K_{y3}}|\gamma_y|\right) \tag{42}$$

$$\gamma_y = \gamma \tag{43}$$

$$\mu_y = \left(p_{D_{y1}} + p_{D_{y2}} \cdot df_z\right)\left(1 - p_{D_{y3}} \cdot \gamma_y^2\right) \tag{44}$$

Normalised Load Variation

$$df_z = \frac{F_z - F_{z0}}{F_{z0}} \tag{45}$$

Implementation of Pure Lateral Slip Pacejka Model using Bicycle Model

Recalling the equations used prior to implementation of the linear tyre model, the equations for the side slip rate, and yaw acceleration are given in equations (19) and (20)

$$\dot{\beta} = \frac{F_{yr} + F_{yf}}{mV} - r \tag{19}$$

$$\dot{r} = \frac{-L_r F_{yr} + L_f F_{yf}}{I_z} \tag{20}$$

Also recalling the geometric description of the slip angles for the front and rear respectively, given in equations (13) and (14)

$$\alpha_r = -\beta + \frac{L_r\, r}{V} \tag{13}$$

$$\alpha_f = \delta - \beta - \frac{L_f\, r}{V} \tag{14}$$

Equations (13) and (14) can be implemented into equations (19) and (20) to represent Pacejka tyre model. (The F_y is the output of the Pacejka model, which takes the respective inputs, **the tyre force is multiplied by two** because the model represents a singular tyre)

$$\dot{\beta} = \frac{2F_y(\alpha_r, F_{zr}) + 2F_y(\alpha_f, F_{zf})}{mV} - r \tag{46}$$

$$\dot{r} = \frac{-2L_r F_y(\alpha_r, F_{zr}) + 2L_f F_y(\alpha_f, F_{zf})}{I_z} \tag{47}$$

The formal formation of the nonlinear

$$\dot{x} = f(x), \quad x = \begin{bmatrix} \beta \\ r \end{bmatrix} \tag{48}$$

Implications of Nonlinearity

With the linear tyre model, the system is linear and can be written in state-space form $\dot{x} = Ax + Bu$.

- Eigenvalues of A reveal stability and allow analytic solutions.

With the Pacejka model, the system is nonlinear because F_y is a nonlinear function of slip angle.

- The superposition principle no longer applies.
- For large deviations (as in Pacejka [1] "*Large deviations with respect to steady state motion*"), we must solve the full nonlinear ODEs.

Solution Approach: Numerical Integration

To analyse this system, a numerical solution is required. The chosen method is the Fourth-Order Runge-Kutta (RK4) scheme, which provides a robust balance of accuracy and computational efficiency for integrating the ordinary differential equations (ODEs) forward in time.

Python includes packages which are able to perform solutions to ODEs, making things easier.

The methodology involves:

- Defining initial conditions
- For each initial condition, using the RK4 algorithm to compute the resulting trajectory.
- Plotting these trajectories on a phase portrait (β vs. r).

This approach allows for the investigation of key dynamic characteristics, including:

- The attraction of trajectories to a stable equilibrium point.
- The divergence of trajectories, indicating unstable motion.
- The potential existence of multiple equilibria or complex attractors, which are hallmarks of non-linear systems.

The ODEs define the derivatives.

This is the fundamental law of the system. For any point (β, r) in the state space, the ODEs describe the instantaneous direction and rate of change ($\dot{\beta}, \dot{r}$).

The vector field shows those derivatives everywhere in the plane.

By evaluating $f(t, x)$ at every point on a grid and drawing an arrow, you create the vector field. It's a static snapshot of all possible instantaneous motions. At a glance, you can see:

- **Direction:** Where the system "wants to go" from any point.
- **Magnitude:** How strongly it's being pushed in that direction.

The trajectory curves show how the system evolves over time when you integrate from a specific starting point.

A trajectory is the path you get by starting at a specific initial condition (β_0, r_0).

An example phase portrait using the following parameters:

Vehicle Parameters

m = 300 # mass in kg
Iz = 150.0 # yaw inertia in kgm^2
lr = 0.8 # CoG to front axle in m
lf = 0.8 # CoG to rear axle in m

Speed and Steering

V = 12 # speed in m/s
delta_deg = 1 # steering input angle (degrees)

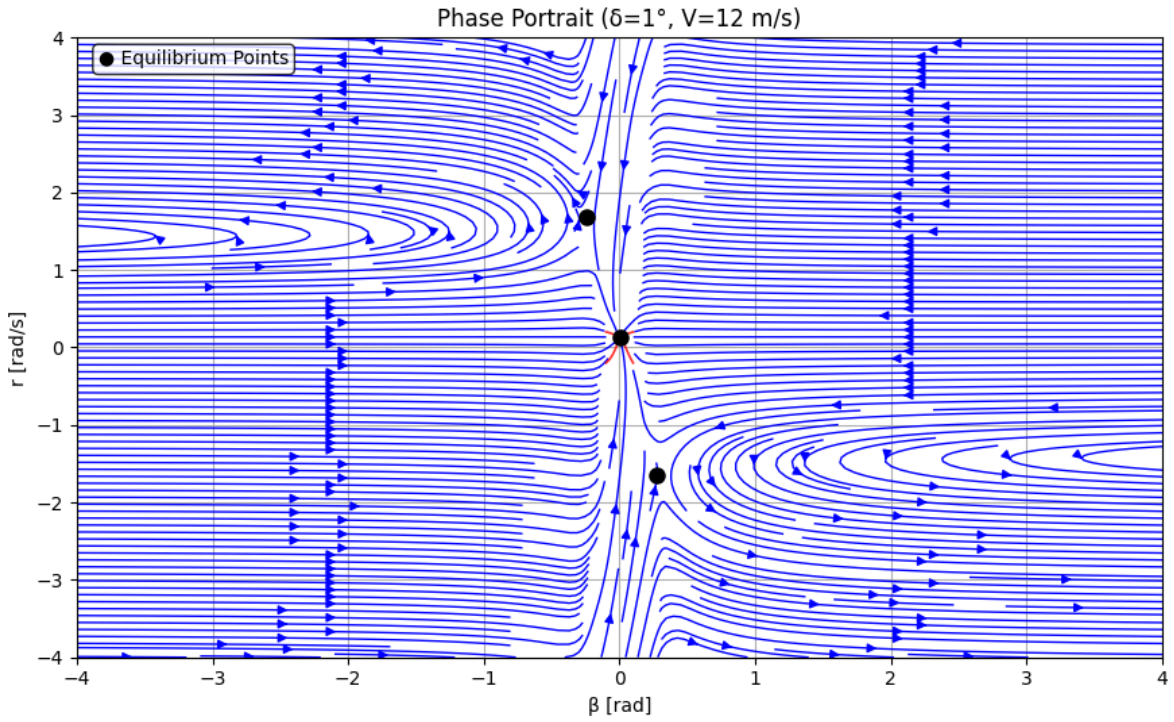

Figure 5 - Phase Portrait using MF Tyre Model

Curl and Divergence

Curl

The curl of a vector field is usually interpreted as **local rotation or swirling tendency** in the vector field

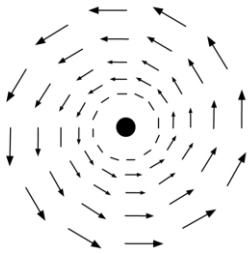

Figure 6 - Visual of Curl

Represented mathematically by the cross product between the del operator ∇ and the vector field, $\boldsymbol{F}$

$$\Delta \times F = \frac{\partial \dot{r}}{\partial \beta} - \frac{\partial \dot{\beta}}{\partial r} \tag{49}$$

In vehicle dynamics, a significant nonzero curl indicates **oscillatory behaviour** around the equilibrium point, often linked to understeer or oversteer tendencies.

Curl measures **microscopic circulation** around a point, not macroscopic bending of trajectories.

Divergence

The divergence measures local expansion or contraction in phase space, meaning how much the vector field is spreading out / converging to a point.

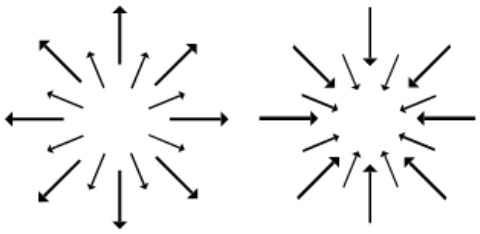

Figure 7- Visual of Divergence

Represented mathematically by the dot product between the del operator ∇ and the vector field, $\boldsymbol{F}$

$$\Delta \cdot \boldsymbol{F} = \frac{\partial \dot{\beta}}{\partial \beta} + \frac{\partial \dot{r}}{\partial r} \tag{50}$$

For the linear model, divergence is constant (equal to the trace of the system matrix) and therefore represents a global stability condition. For the nonlinear Pacejka model, divergence varies with slip angle and yaw rate, showing that stability is state-dependent.

Numerical Evaluation (Central Difference Method)

Since closed-form derivatives are not always available for nonlinear models, divergence and curl are approximated numerically on a grid.

$$\left.\frac{\partial \dot{\beta}}{\partial \beta}\right|_{i,j} \approx \frac{\dot{\beta}(\beta_i, r_{j+1}) - \dot{\beta}(\beta_i, r_{j-1})}{2\Delta\beta} \tag{51}$$

$$\left.\frac{\partial \dot{\beta}}{\partial r}\right|_{i,j} \approx \frac{\dot{\beta}(\beta_{i+1}, r_j) - \dot{\beta}(\beta_{i+1}, r_j)}{2\Delta r} \tag{52}$$

$$\left.\frac{\partial \dot{r}}{\partial \beta}\right|_{i,j} \approx \frac{\dot{r}(\beta_i, r_{j+1}) - \dot{r}(\beta_i, r_{j-1})}{2\Delta\beta} \tag{53}$$

$$\left.\frac{\partial \dot{r}}{\partial r}\right|_{i,j} \approx \frac{\dot{r}(\beta_{i+1}, r_j) - \dot{r}(\beta_{i+1}, r_j)}{2\Delta r} \tag{54}$$

For a mesh of points (β_i, r_j) with spacing $\Delta\beta, \Delta r$:

Taken together, the divergence and curl fields highlight where the vehicle behaves predictably (stable convergence to equilibrium) and where nonlinear effects may cause complex, oscillatory, or unstable dynamics.

Python includes methods which are able to calculate the gradient without having to worry about the numerical method.

Phase Portrait Analysis

Examine Vector Field Patterns

Converging arrows/streamlines → stability.

- Trajectories move toward an equilibrium point; small perturbations decay over time.

Diverging arrows → instability.

- Small deviations grow, indicating the vehicle may leave a steady-state trajectory.

Spirals → underdamped oscillatory motion.

- Trajectories rotate around an equilibrium while gradually converging (stable focus) or diverging (unstable focus).

Straight-line convergence → overdamped motion.

- Trajectories move directly toward equilibrium without oscillation (stable node).

Closed loops (in nonlinear models) → limit cycles / sustained oscillations.

- Vehicle may oscillate indefinitely around a state due to nonlinear effects (e.g., tire saturation or nonlinear damping).

Curl → local rotational tendency.

- Positive curl indicates counterclockwise rotation around a point.
- Negative curl indicates clockwise rotation.
- Zero curl indicates purely convergent/divergent behaviour.

In vehicle dynamics, curl highlights yaw oscillations or tendency for the car to rotate around its CoG when disturbed.

Interpret Trajectories

Pick initial conditions and observe how trajectories evolve:

- **Do they spiral into equilibrium** → stable focus.
    - Indicates understeer or damped yaw oscillations; car naturally returns to path with slight rotation.
- **Do they curve directly in** → stable node.
    - Suggests strong stability margin; vehicle returns to trajectory without oscillation.
- **Do they shoot away from equilibrium** → unstable.
    - Indicates potential oversteer or yaw instability; small perturbations amplify.
- **Do multiple equilibria exist** → nonlinear phenomena.
    - Multiple stable/unstable points may correspond to nonlinear tire behaviour, load transfer effects, or CoG shifts.
- **Analyse curl along trajectories**.
    - High curl near an equilibrium → significant rotational dynamics, possible oscillatory yaw.
    - Low curl → linearised response valid; vehicle translates without rotation.

Compare Linear vs Nonlinear Models

- Linear model → usually one equilibrium, smooth convergence patterns, curl mostly uniform.
    - Useful for small slip angles, low lateral loads.
- Nonlinear model → may have multiple equilibria, saturation effects, divergent trajectories, or localized regions of high curl.
    - Highlights where linear approximations fail (e.g., high speed, large slip angles, sudden steering inputs).

Phase Portrait Analysis for Vehicle Dynamics

1. **Identify equilibrium points**: These correspond to steady-state conditions (straight-line or constant-radius turn).
2. **Observe trajectories**: How the vehicle responds to small perturbations in yaw rate and lateral velocity.
3. **Use curl to assess rotational tendency**: Helps identify understeer/oversteer characteristics.
4. **Evaluate stability margins**: The size of converging regions indicates robustness to disturbances.
5. **Compare linear vs nonlinear predictions**: Determines safe operating limits and regions where nonlinearity may cause loss of control.

Parameter Influence on Phase Portraits

Forward Velocity V

At low speeds

- Front slip dominates; tire forces are more linear.
- Vehicles usually exhibit **understeer stability** (trajectories converge to equilibrium).

At high speeds:

- The rear slip angle dominates, and the vehicle can transition into oversteer.
- The system stability depends not only on the sign of the **understeer gradient** but also on whether there is **enough yaw damping**.

In the phase plane:

- Equilibrium points may shift position, and divergence around the equilibrium can change sign, indicating the onset of instability. Above the critical speed, if yaw stiffness is weak and yaw damping is low, the system diverges quickly.

Cornering Stiffness $C_{\alpha f}$, $C_{\alpha r}$

Increasing $C_{\alpha f}$ (front stiffness):

- This promotes understeer, enhancing stability. Higher front stiffness → more yaw damping.

Increasing $C_{\alpha r}$ (rear stiffness):

- This promotes oversteer, reducing stability. Higher rear stiffness → less yaw damping.

In the phase plane:

- Higher rear stiffness makes trajectories diverge more quickly, while higher front stiffness causes trajectories to settle faster.

Steering Angle δ

Small inputs:

- The system remains close to linear, with a single stable equilibrium.

Larger inputs:

- Nonlinear tyre effects dominate, potentially introducing multiple equilibria.

In the phase plane:

- Equilibrium shifts further away from the origin, and nonlinear distortion becomes more evident.

Camber Angle γ

Negative camber:

- increases lateral force capability in cornering manoeuvres, improving cornering at moderate slips.

Excessive camber:

- Can reduce stability (uneven tire contact patch, asymmetric force).

In the phase plane:

- The trajectories may converge more quickly (enhanced stability) or diverge asymmetrically depending on camber direction.

Mass Distribution (l_f, l_r)

Rearward CoG (higher $\frac{l_f}{l_r}$)

- Yaw stiffness decreases → rear axle dominates, restoring yaw moment weakens → tendency toward oversteer.
- Yaw damping decreases → shorter front lever arm, front tires less effective at damping yaw rate.

A more forward CoG (lower $\frac{l_f}{l_r}$)

- Yaw stiffness increases → front axle dominates, restoring yaw moment stronger → understeer tendency.
- Yaw damping increases → longer front lever arm, more yaw-rate resistance → greater stability margin.

In the phase plane: stability boundaries shift; for rear-heavy setups, unstable regions expand.

Parameter Adjustment Phase Portraits

Taking the following example as a base design and adjusting the parameters to visualise how the state of the model changes.

The base parameters will be denoted as follows (representative of a FSAE car) using identical front and rear tyres.

Vehicle Parameters
m = 300 # mass in kg
Iz = 150.0 # yaw inertia in kgm^2
lr = 0.8 # CoG to front axle in m
lf = 0.8 # CoG to rear axle in m

Speed and Steering
V = 12 # speed in m/s
delta_deg = 1 # steering input angle (degrees)

These will be the baseline when comparing the variation in vehicle states. For this I will also use the nonlinear tyre model

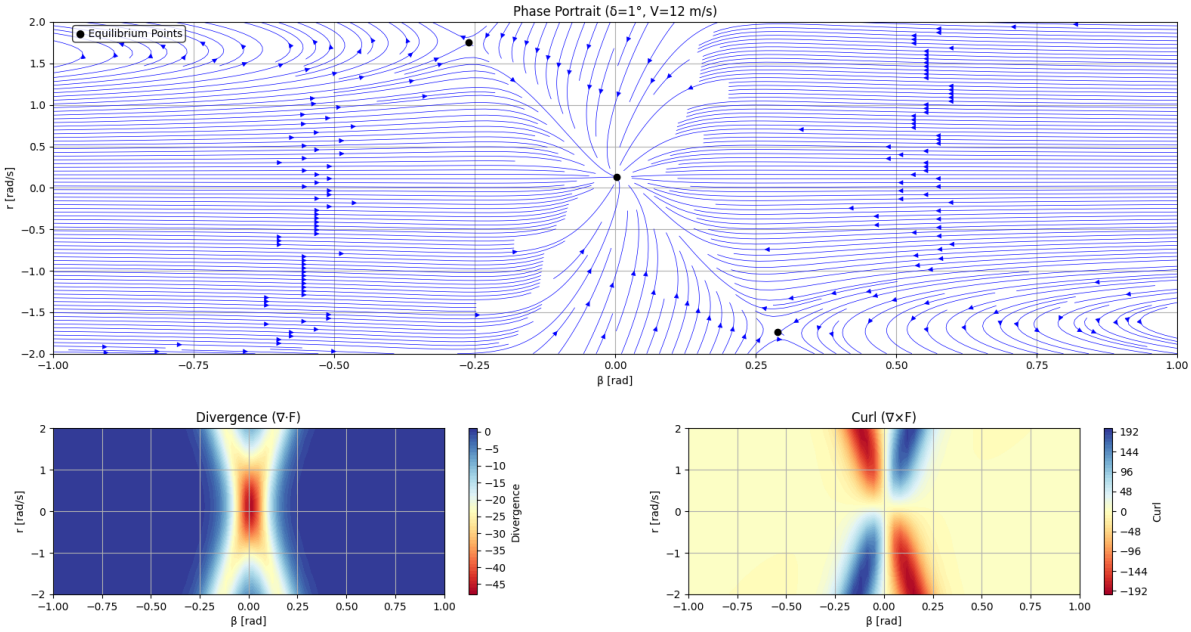

Figure 8 - Baseline Stability Phase Portrait

Forward Velocity V

Keeping all the parameters the same, the forward velocity will be increase from 12m/s to 40m/s

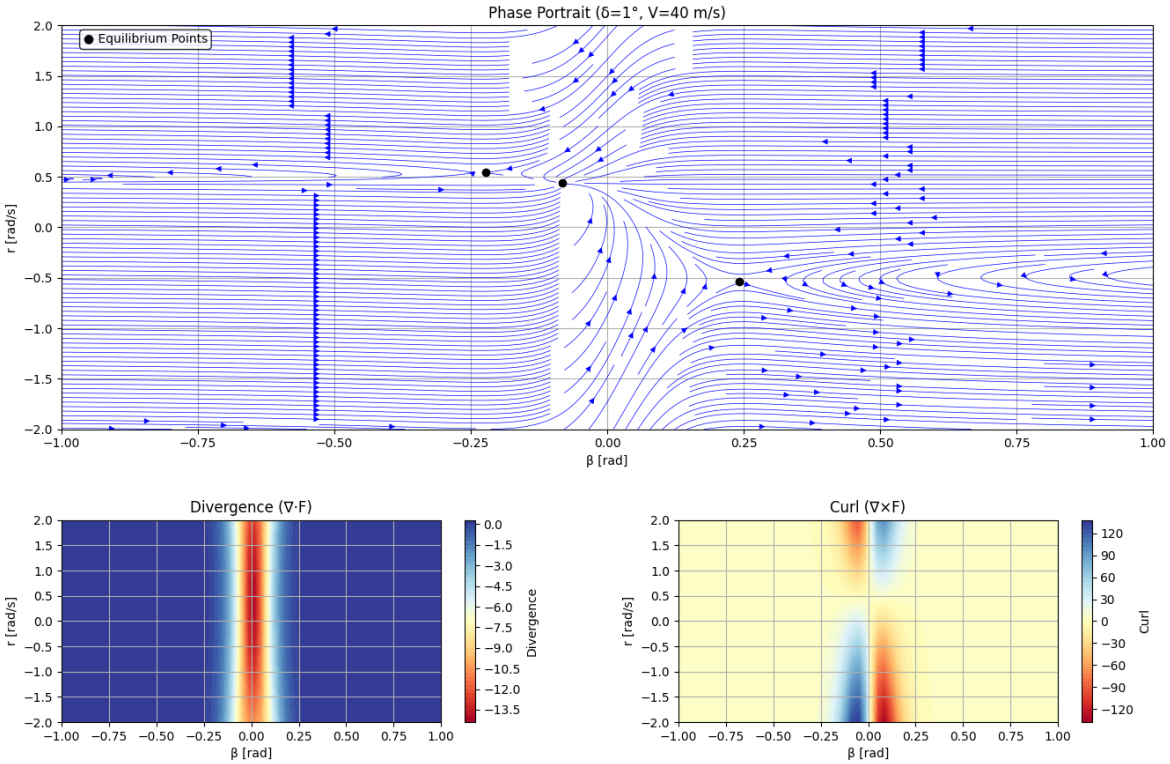

Figure 9 – Phase portrait at V = 40 m/s, $\delta = 1°$

Steering Angle δ

Reverting back to the stable parameters, adjusting the steering angle to $\delta = 10\,°$ at a velocity of $V = 12\ m/s$.

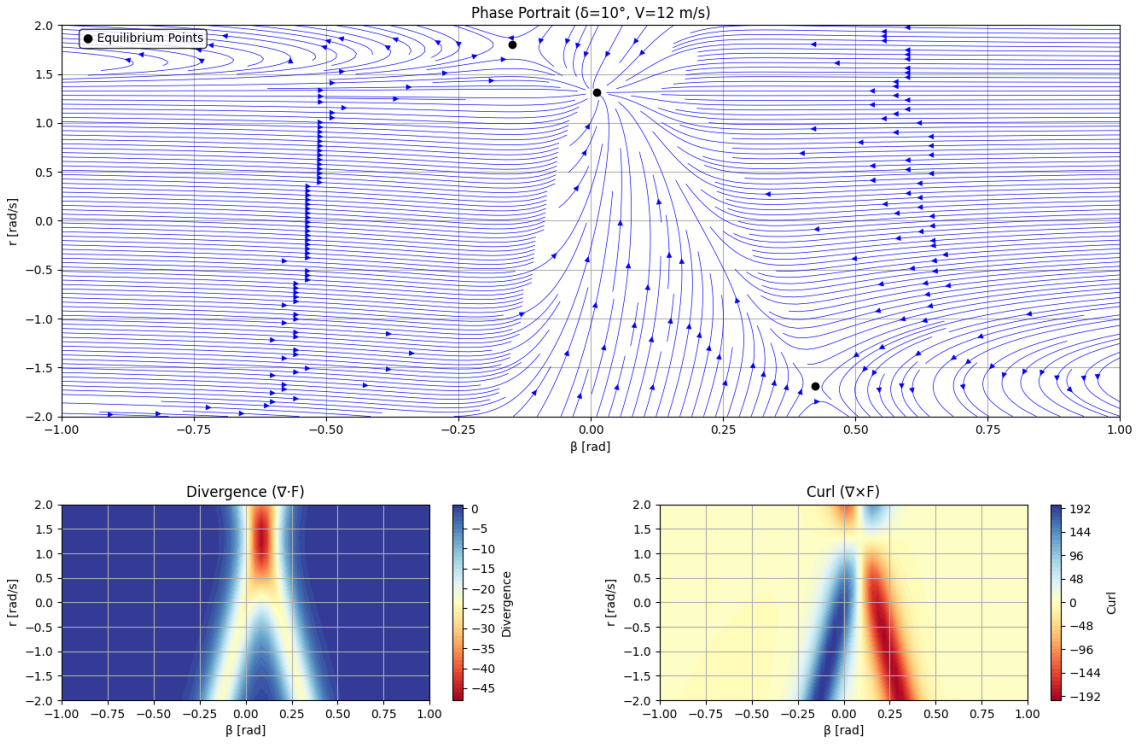

Figure 10 – Increased Steer angle $\delta = 10\,°$, low speed $V = 12\ m/s$

Increasing the velocity up to $V = 25\ m/s$ whilst keeping the steering angle at $\delta = 10\,°$.

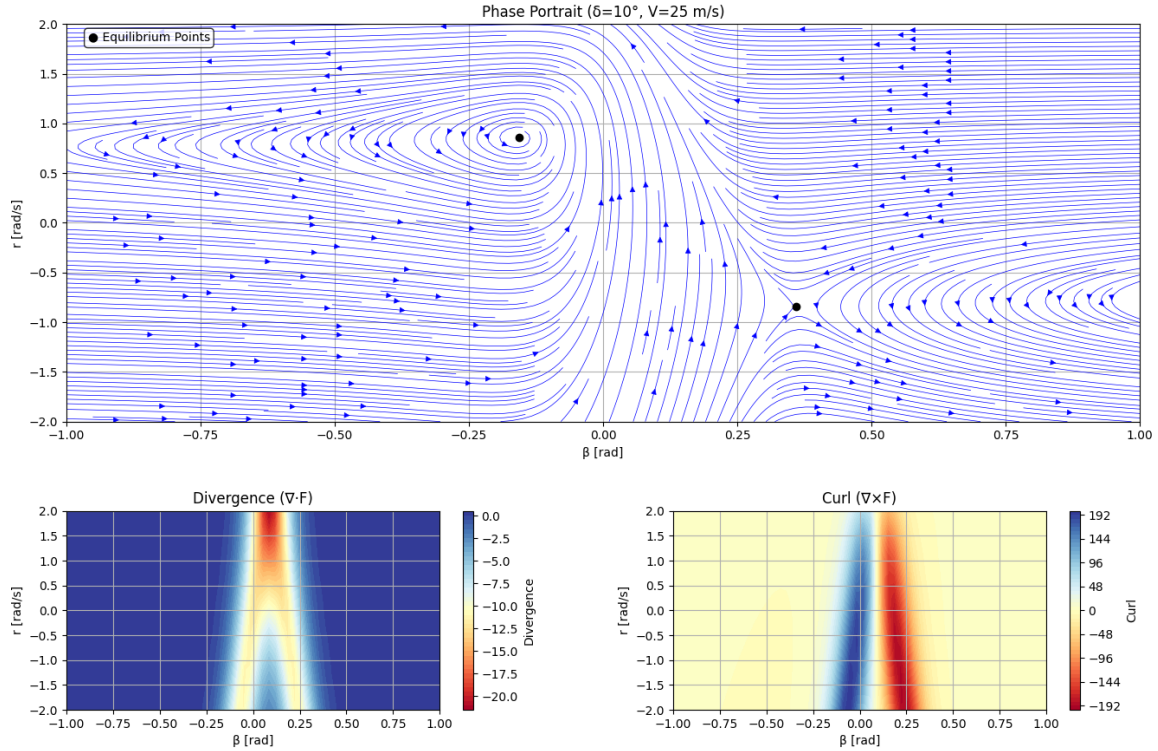

Figure 11– Increased Steer angle $\delta = 10\,°$, higher speed $V = 25\ m/s$

Mass Distribution (l_f, l_r)

Changing the weight distribution rearward with $l_r = 0.3$ and the $l_f = 1.3$

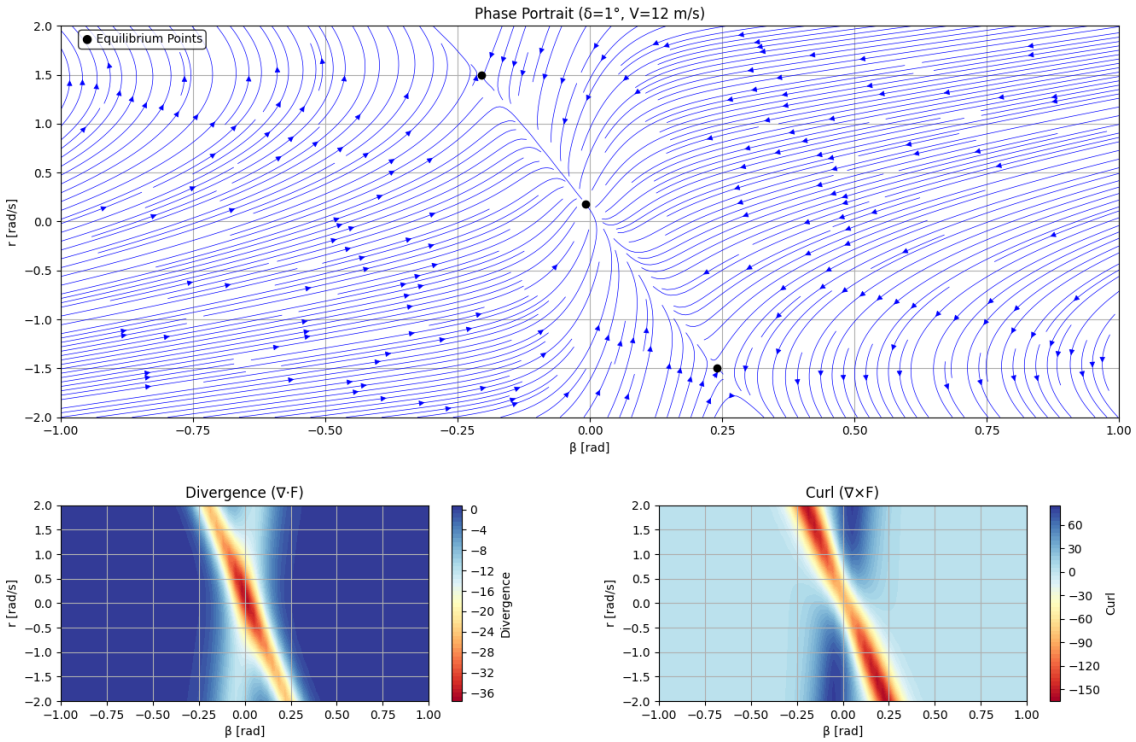

Figure 12- Rearward Centre of Gravity, low speed $V = 12\ m/s$ and low steering angle $\delta = 1°$

Increasing the steering angle and speed with the wheelbase rearwards

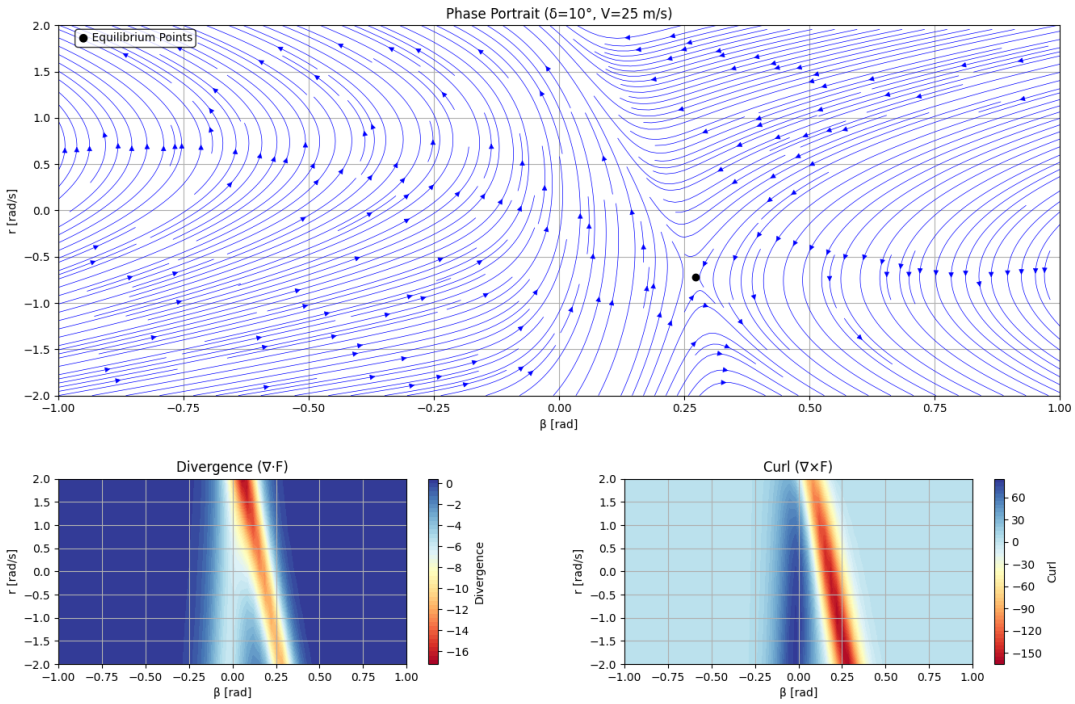

Figure 13 - Rearward Centre of Gravity, higher speed $V = 25\ m/s$ and higher steering angle $\delta = 10°$

Changing the weight distribution forward with $l_f = 0.3$ and the $l_r = 1.3$

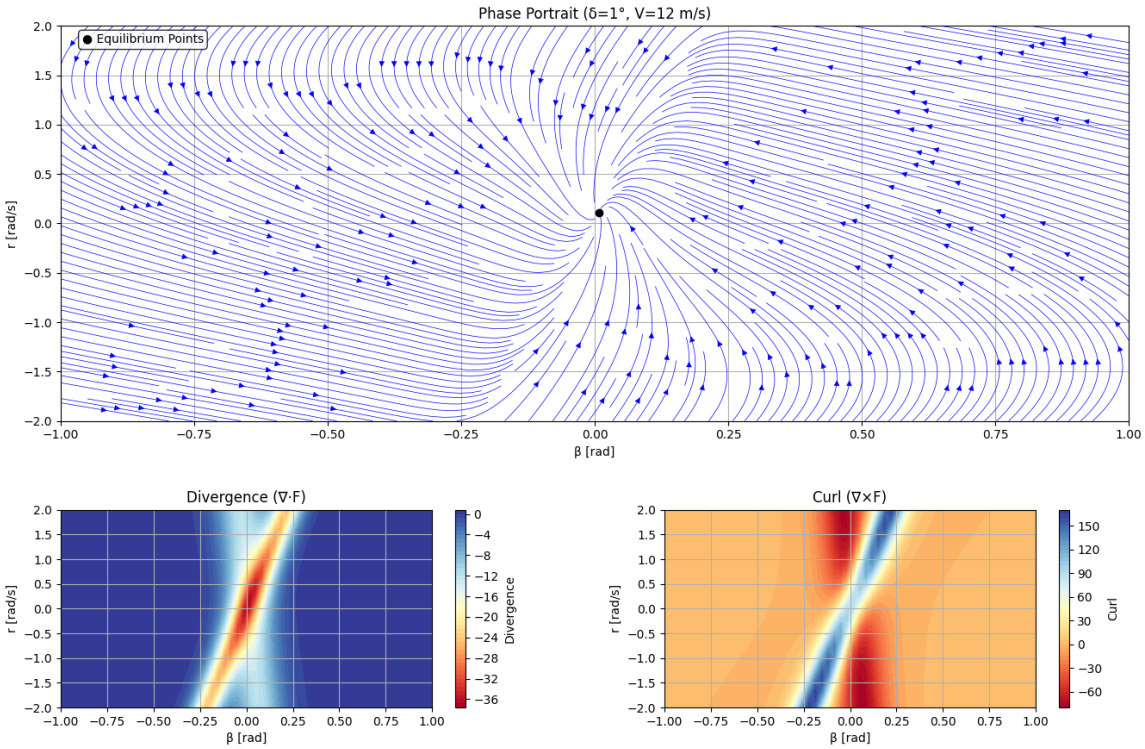

Figure 14 - Frontward Centre of Gravity, low speed $V = 12\ m/s$ and low steering angle $\delta = 1°$

Increasing the steering angle and speed with the wheelbase forward

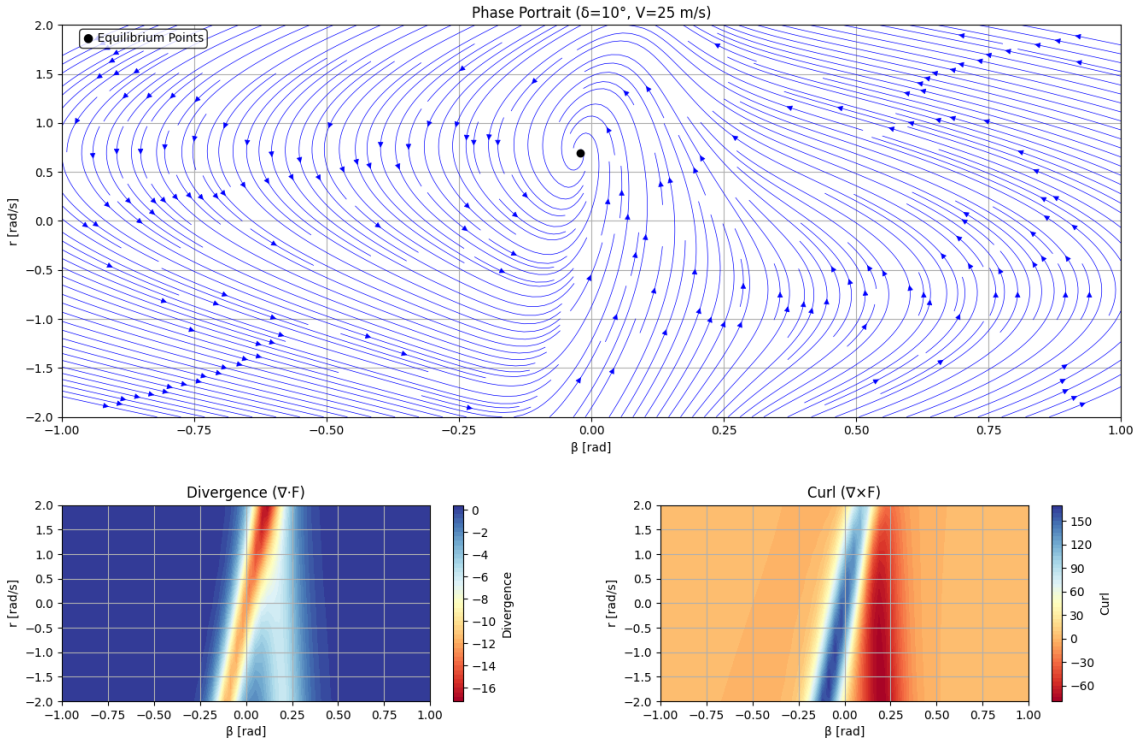

Figure 15- Frontward Centre of Gravity, higher speed $V = 25\ m/s$ and higher steering angle $\delta = 10°$

Analysis of Parameter Adjustments

Figure 8 – Baseline Stability Portrait

- At **δ = 1°, V = 12 m/s**, the central equilibrium (β ≈ 0, r ≈ 0) is **stable** – trajectories converge smoothly.
- **Divergence is strongly negative** near the origin, confirming strong damping and stability.
- **Curl alternates in lobes** around β ≈ ±0.1–0.25 rad, showing mild oscillatory yaw behaviour as the system settles.
- Off-centre equilibria appear due to **nonlinear tyre effects**, but the dominant attractor is the central equilibrium.
- Overall, the vehicle is in a **safe, understeering regime** where small disturbances decay naturally

Figure 9 – High Velocity (V = 40 m/s, δ = 1°)

- Multiple **equilibria** appear (central and off-centre), indicating the nonlinear tyre behaviour dominates at this higher speed.
- **Trajectories no longer converge smoothly** to the central equilibrium – some diverge or get "pushed away," suggesting the onset of yaw instability.
- **Divergence** is close to zero or weakly negative in many regions, unlike the strongly negative values seen at 12 m/s. This shows the system has lost strong damping → stability margin is reduced.
- **Curl values** are larger and more spread, indicating stronger oscillatory yaw dynamics before instability develops.
- Overall: at **40 m/s**, the vehicle exhibits **reduced stability and an oversteer tendency** — small disturbances may grow rather than decay, matching the critical-speed behaviour seen in real cars.

Figures 10–11 – Larger Steering Angle (δ = 10°)

- At **V = 12 m/s**, still reasonably stable but convergence is slower: some oscillations visible.
- At **V = 25 m/s with δ = 10°**, system becomes **highly unstable**:
    - Trajectories diverge rapidly.
    - Tyre forces are in the saturated region → nonlinear instability dominates.
- This shows how *large steering + higher speed* accelerates the loss of stability.

Figures 12–13 – Rearward CoG (lf = 1.3, lr = 0.3)

- Rear-heavy balance weakens yaw damping.
- Even at modest speed/steer input, trajectories show strong divergence.
- Increasing δ or V exaggerates this effect: the unstable region in the phase plane expands dramatically.
- Matches the intuition: "rear-heavy cars are twitchy" → classic oversteer.

Figures 14–15 – Forward CoG (lf = 0.3, lr = 1.3)

- Forward-heavy setup increases yaw damping and understeer tendency.
- Phase portrait shows **much stronger convergence** compared to rear-heavy case.
- Even at higher δ and V, instability takes longer to appear.
- Trade-off: stability improves, but steering response becomes more sluggish (understeer)

Comparison Between Linear and Nonlinear Tyre Model

To match our formula student tyre within our linear tyre model, an instantaneous cornering stiffness of $C_\alpha \approx 29430/rad$ is selected (this is the axial stiffness and not tyre stiffness). The assumption for a linear model also breaks down in this situation at approximately 0.05 radians.

This cornering stiffness is computed from the Nonlinear tyre code by using numerical differentiation to compute the following:

- Cornering stiffness is the slope of the **lateral force vs slip angle curve** at zero slip angle:

$$C_\alpha = \frac{dF_y}{d\alpha}\Big|_{\alpha=0} \tag{55}$$

Table 1 - Pacejka Tyre Fitment Parameters, FSAE tyre

Fitted Tyre Parameter	Value
$p_{C_{y1}}$	1.4
$p_{E_{y1}}$	-0.1
$p_{E_{y2}}$	-0.05
$p_{E_{y3}}$	0
$p_{E_{y4}}$	0
$p_{K_{y1}}$	20
$p_{K_{y2}}$	2
$p_{K_{y3}}$	-0.5
$p_{D_{y1}}$	1.2
$p_{D_{y2}}$	-0.2
$p_{D_{y3}}$	10
p_{hy1}	0.0001
p_{hy2}	0.0002
p_{hy3}	0.003
p_{vy1}	0.001
p_{vy2}	0.002
p_{vy3}	0.003
p_{vy4}	0.004

(**Note**: The data used from Table 1 is not real data just assumed parameters for purposes of the project)

Error Calculation between the models

Model parameters are given in Table 2.

Table 2 - Model Parameters for Comparison Test

Linear Parameters	Nonlinear parameters
$m = 300\ kg$	$m = 300\ kg$
$I_z = 150\ kgm^2$	$I_z = 150\ kgm^2$
$l_f = 0.8\ m$	$l_f = 0.8\ m$
$l_r = 0.8\ m$	$l_r = 0.8\ m$
$C_{\alpha f} = 29430N/rad$	For tyre parameters reference Table 1

$C_{\alpha r} = 29430N/rad$	

Both models are tested at $V = 12m/s$ and at $\delta = 1°$

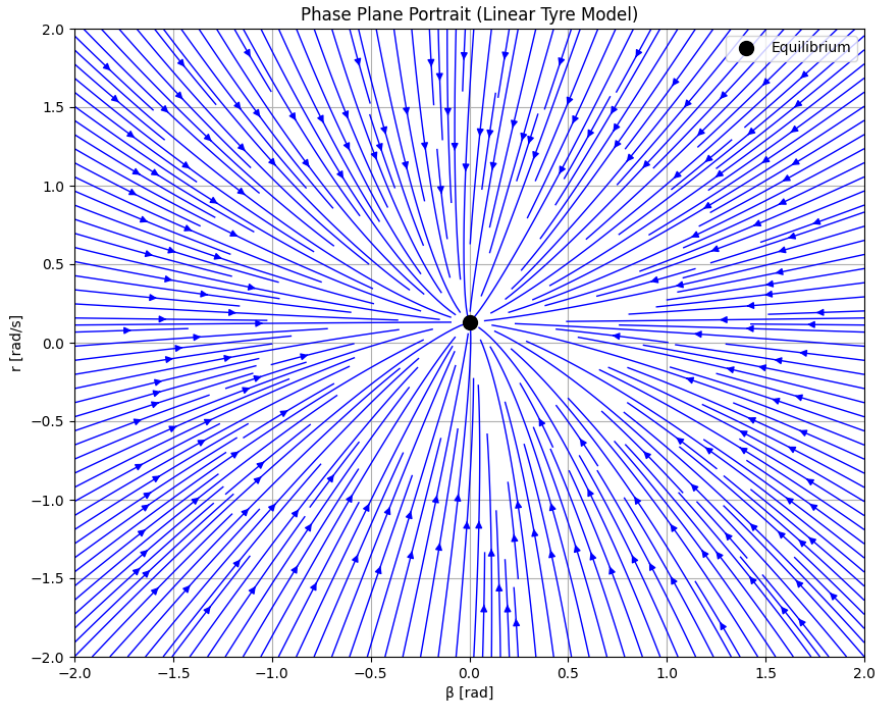

Figure 166-Linear Tyre Model Phase Plain for Comparison

Table 3 -Linear Tyre Model Phase Plane Results

Equilibrium	Beta and r values
Equilibrium 1	β = 0. 0007 rad, r = 0.1309 rad/s

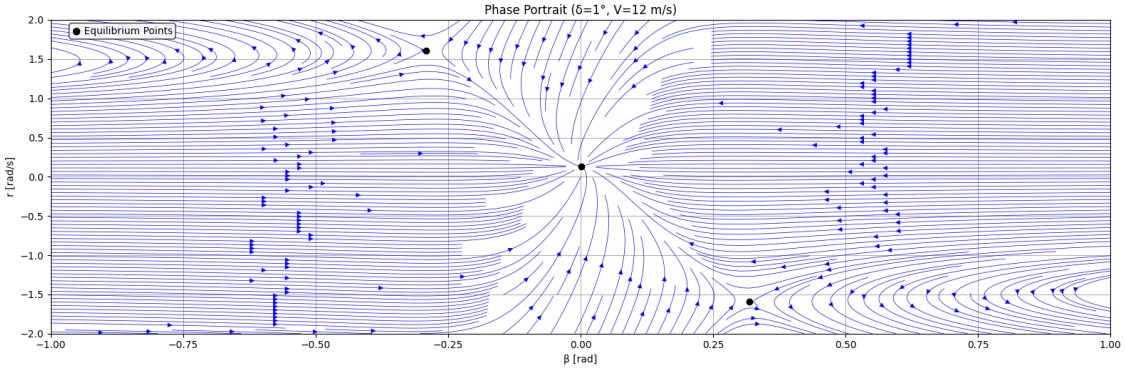

Figure 17 - Nonlinear Tyre Model Phase Plain for Comparison

Table 4 – Nonlinear Tyre Model Phase Plane Results

Equilibrium	Beta and r values
Equilibrium 1	β = -0. 2911rad, r = 1. 6104 rad/s
Equilibrium 2	**β = 0. 0013 rad, r = 0.1309 rad/s**
Equilibrium 3	β = 0.3180 rad, r = -1. 5922rad/s

The linear and nonlinear bicycle models were compared under identical vehicle parameters and a steering input of δ = 1°. After ensuring both models used equivalent initial cornering stiffness values derived from the Pacejka model, a significant discrepancy of **46% in the sideslip angle (β)** persisted.

- **Linear Model Equilibrium:** β = 0.0007 rad, r = 0.1309 rad/s
- **Nonlinear Model Equilibrium (Central Point):** β = 0.0013 rad, r = 0.1309 rad/s

Interpretation:

This discrepancy is a direct result of the nonlinear Pacejka model's ability to capture critical real-world phenomena that the linear model inherently ignores:

1. **Force Offsets** (S_{Hy}, S_{Vy}): These parameters model **ply steer** and **conicity**—inherent traits where a tyre generates lateral force even at zero slip angle. The linear model must return zero force at zero slip, forcing an incorrect equilibrium.
2. **Camber Effects (γ):** The linear model has no mechanism to incorporate camber angle, which significantly influences tyre force generation.
3. **Load Sensitivity (df_z):** The linear model assumes cornering stiffness is constant. However, tyre force does not scale linearly with vertical load (Fz).
4. **Saturation and Multiple Equilibria:** Most profoundly, the linear model can only ever predict a **single, unique equilibrium point**. In contrast, the nonlinear model reveals **three equilibrium states** under the same conditions.

Conclusion:

The 46% error in sideslip angle and the existence of multiple equilibria demonstrate that a linear tyre model is fundamentally insufficient for high-fidelity vehicle dynamics analysis. It fails to capture:

- Inherent force offsets,
- Camber influence,
- Nonlinear load-force relationships,
- And the complex, multi-state behaviour of a vehicle at the limits of handling.

This analysis justifies the essential use of a nonlinear tyre model, like the Pacejka formulation, for accurate simulation, stability analysis, and vehicle setup in any serious engineering context, especially in motorsports where these effects dominate performance.

References

[1] - Pacejka, Hans B. *Tyre and Vehicle Dynamics*. Amsterdam, Elsevier/Butterworth-Heinemann, 2007.

[2] - Milliken, William F, and Douglas L Milliken. *Race Car Vehicle Dynamics*. Warrendale, Pa, U.S.A., Sae International, 1995.

[3] - Felipe. “MF52_EquationManual.” *Scribd*, 2025, www.scribd.com/document/665983112/MF52-EquationManual. Accessed 20 Aug. 2025.

[4] - “Magic Formula Tyre Tool - File Exchange - MATLAB Central File Exchange - MATLAB Central.” *Mathworks.com*, 20 Dec. 2023, https://nl.mathworks.com/matlabcentral/fileexchange/111375-magic-formula-tyre-tool?s_tid=FX_rc3_behav . Accessed 20 Aug. 2025.

[5] - Ma, Biao, et al. “A Shared Steering Controller Design Based on Steer-By-Wire System Considering Human-Machine Goal Consistency.” *Journal of the Franklin Institute*, vol. 356, no. 8, May 2019, pp. 4397–4419, https://doi.org/10.1016/j.jfranklin.2018.12.028. Accessed 20 Aug. 2025.

[6] - Hao, Zhang, et al. “Phase Plane Analysis for Vehicle Handling and Stability.” *International Journal of Computational Intelligence Systems*, vol. 4, no. 6, Dec. 2011, pp. 1179–1186, https://doi.org/10.1080/18756891.2011.9727866. Accessed 20 Aug. 2025.

Python Code

Note: the appropriate imported packages must be successfully installed in order for these programs to function effectively.

Phase Portrait Generation Linear Tyre Model

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

# Vehicle parameters
m = 300.0 #kg
Iz = 150.0 #yaw inertia in kgm^2
lf = 0.8 # CoG to front axle in m
lr = 0.8 # CoG to rear axle in m
Caf = 30000  #front tyre stiffness N/rad
Car = 30000 # rear tyre stiffness N/rad

# Speed and steer input
V = 12 # speed in m/s
delta = 1 #enter the steering angle in deg
delta = np.deg2rad(delta)

#increase this number if you want to increase size of vector field
grid_size = 2   #increase this number of the vector field is cut off
# ------------------------------------------------
# State-space dynamics
# ------------------------------------------------
def f(t, x):
   beta, r = x
   beta_dot = -(Caf + Car)/(m*V) * beta \
           + ((Car*lr - Caf*lf)/(m*V**2) - 1.0) * r \
           + (Caf/(m*V)) * delta

   r_dot    = (lr*Car - lf*Caf)/Iz * beta \
           - (lf**2*Caf + lr**2*Car)/(Iz*V) * r \
           + (lf*Caf/Iz) * delta
   return [beta_dot, r_dot]

# System matrices
A = np.array([
   [-(Caf + Car)/(m*V), (Car*lr - Caf*lf)/(m*V**2) - 1],
   [(lr*Car - lf*Caf)/Iz, -(lf**2*Caf + lr**2*Car)/(Iz*V)]
])
B = np.array([[Caf/(m*V)], [lf*Caf/Iz]])

# Equilibrium point
```

```
x_eq = -np.linalg.inv(A) @ B * delta

# ------------------------------------------------
# Phase plane construction
# ------------------------------------------------

beta_vals = np.linspace(-grid_size, grid_size, 80)
r_vals    = np.linspace(-grid_size, grid_size, 80)
BETA, R = np.meshgrid(beta_vals, r_vals)

beta_dot, r_dot = np.zeros_like(BETA), np.zeros_like(R)
for i in range(BETA.shape[0]):
   for j in range(BETA.shape[1]):
      dx = f(0, [BETA[i,j], R[i,j]])
      beta_dot[i,j], r_dot[i,j] = dx

# ------------------------------------------------
# Plot
# ------------------------------------------------
fig, ax = plt.subplots(figsize=(10, 8))
ax.streamplot(BETA, R, beta_dot, r_dot, color="blue", density=3.0, linewidth=1)

# Add trajectories
initial_conditions = [[0.1, 0.2], [-0.1, 0.2], [0.1, -0.2], [-0.1, -0.2]]
for x0 in initial_conditions:
   sol = solve_ivp(f, [0, 5], x0, t_eval=np.linspace(0, 5, 200))
   ax.plot(sol.y[0], sol.y[1], 'r-', lw=2)

# Equilibrium point
ax.plot(x_eq[0], x_eq[1], 'ko', markersize=10, label="Equilibrium")
ax.set_xlabel("β [rad]")
ax.set_ylabel("r [rad/s]")
ax.set_title("Phase Plane Portrait (Linear Bicycle Model)")
ax.legend()
ax.grid(True)
plt.show()

# ------------------------------------------------
# Stability analysis
# ------------------------------------------------
eigenvalues = np.linalg.eigvals(A)
print(f"Equilibrium point: β = {x_eq[0,0]:.4f} rad, r = {x_eq[1,0]:.4f} rad/s")
print(f"Eigenvalues: {eigenvalues[0]:.4f}, {eigenvalues[1]:.4f}")

if np.all(np.real(eigenvalues) < 0):
   print("System is stable")
else:
   print("System is unstable")
```

Phase Portrait Generation Nonlinear Pacejka Tyre Model

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
from scipy.optimize import fsolve

# ------------------------------------------------------------------------------------
#
# This is the phase portrait analysis tool which is used, it is recommended that the document
# provided is read and used to understand the equations and theory
#
# ------------------------------------------------------------------------------------

# ------------------------------------------------
# Vehicle Parameters
# ------------------------------------------------
m = 300  # mass in kg
Iz = 150.0  # yaw inertia in kgm^2
lr = 0.8  # CoG to front axle in m
lf = 0.8  # CoG to rear axle in m

# ------------------------------------------------
# Speed and Steering
# ------------------------------------------------
V = 12  # speed in m/s
delta_deg = 1 # steering input angle (degrees)

delta = np.deg2rad(delta_deg)
# ------------------------------------------------
# Tyre Parameters
# ------------------------------------------------------
F_zf = m * 9.81 * lr / (lf + lr)
F_zr = m * 9.81 * lf / (lf + lr)
F_z0 = 1000  # nominal tyre load in N

Gamma = np.deg2rad(-1)  # camber angle in degrees

# -------------------------------
# Front tyre parameters
p_Cy1_f, p_Dy1_f, p_Dy2_f, p_Dy3_f = 1.4, 1.2, -0.2, 10
p_Ey1_f, p_Ey2_f = -0.1, -0.05
p_Ky1_f, p_Ky2_f, p_Ky3_f = 20, 2, -0.50

# Rear tyre parameters
p_Cy1_r, p_Dy1_r, p_Dy2_r, p_Dy3_r = 1.4, 1.2, -0.2, 10
p_Ey1_r, p_Ey2_r = -0.1, -0.05
p_Ky1_r, p_Ky2_r, p_Ky3_r = 20, 2, -0.50
```

```
#=========================Plot control=======================================
#control the analysis region
Horizontal_axis = 1  # range of side slip values / graph axis (radians)
Vertical_axis = 2  # range of yaw rate values /  graph axis(radians)
Plot_Trajectories = False    #red lines, change to False if you want to disable

# ------------------------------------------------
# Simplified Pacejka "pure slip" lateral model, I have used identical tyres front and rear but this can be modified
# ------------------------------------------------
def lateral_pure_slip(alpha, Fz, Fz0, gamma,
                      p_Cy1, p_Dy1, p_Dy2, p_Dy3,
                      p_Ey1, p_Ey2,
                      p_Ky1, p_Ky2, p_Ky3):

    dfz = (Fz - Fz0) / Fz0
    SHy = 0.0001 + 0.0002 * dfz + 0.003 * gamma
    SVy = Fz * ((0.001 + 0.002 * dfz) + (0.003 + 0.004 * dfz) * gamma)

    Cy = p_Cy1
    mu_y = (p_Dy1 + p_Dy2 * dfz) * (1 - p_Dy3 * gamma ** 2)
    Dy = mu_y * Fz

    alpha = np.clip(alpha, -np.pi/2, np.pi/2)
    alpha_y = alpha + SHy

    Ey = p_Ey1 + p_Ey2 * dfz
    Ky = p_Ky1 * Fz0 * np.sin(2 * np.arctan(Fz / (p_Ky2 * Fz0))) * (1 - p_Ky3 * abs(gamma))
    By = Ky / max(Cy * Dy, 1e-6)  # safeguard

    return Dy * np.sin(Cy * np.arctan(By * alpha_y - Ey * (By * alpha_y - np.arctan(By * alpha_y)))) + SVy


# ------------------------------------------------
# System dynamics (β-dot, r-dot)
# ------------------------------------------------
def f_nonlinear(t, x):
    beta, r = x
    alpha_f = delta - beta - (lf / V) * r
    alpha_r = -beta + (lr / V) * r

    # Front tyre force
    Fy_f = 2 * lateral_pure_slip(alpha_f, F_zf, F_z0, Gamma,
                         p_Cy1_f, p_Dy1_f, p_Dy2_f, p_Dy3_f,
                         p_Ey1_f, p_Ey2_f,
                         p_Ky1_f, p_Ky2_f, p_Ky3_f)
```

```
    # Rear tyre force
    Fy_r = 2 * lateral_pure_slip(alpha_r, F_zr, F_z0, Gamma,
                                 p_Cy1_r, p_Dy1_r, p_Dy2_r, p_Dy3_r,
                                 p_Ey1_r, p_Ey2_r,
                                 p_Ky1_r, p_Ky2_r, p_Ky3_r)

    beta_dot = (Fy_f + Fy_r) / (m * V) - r
    r_dot = (lf * Fy_f - lr * Fy_r) / Iz
    return [beta_dot, r_dot]


# ------------------------------------------------
# Find equilibrium positions
# ------------------------------------------------
def find_equilibria():
    def equations(x):
        return f_nonlinear(0, x)

    equilibria = []
    beta_guesses = np.linspace(-Horizontal_axis, Horizontal_axis, 20)
    r_guesses = np.linspace(-Vertical_axis, Vertical_axis, 20)

    for b in beta_guesses:
        for r in r_guesses:
            sol = fsolve(equations, [b, r], full_output=True)
            if sol[2] == 1:  # Converged
                eq_point = np.round(sol[0], decimals=4)


                if (abs(eq_point[0]) <= Horizontal_axis and
                    abs(eq_point[1]) <= Vertical_axis):

                    if not any(np.allclose(eq_point, existing, atol=1e-3) for existing in equilibria):
                        equilibria.append(eq_point)

    return equilibria
# ------------------------------------------------
# Phase portrait with smooth streamlines
# ------------------------------------------------
beta_vals = np.linspace(-Horizontal_axis, Horizontal_axis, 60)
r_vals = np.linspace(-Vertical_axis, Vertical_axis, 60)
B, R = np.meshgrid(beta_vals, r_vals)

# Vector field
dB, dR = np.zeros_like(B), np.zeros_like(R)
for i in range(B.shape[0]):
    for j in range(B.shape[1]):
        dB[i, j], dR[i, j] = f_nonlinear(0, [B[i, j], R[i, j]])
```

```
# ------------------------------------------------
# Compute divergence and curl using numpy.gradient
# ------------------------------------------------
def calculate_divergence_curl(B, R, dB, dR, beta_vals, r_vals, smooth_sigma=None):
    dbeta = beta_vals[1] - beta_vals[0]  # step size in β
    dr = r_vals[1] - r_vals[0]         # step size in r

    # Gradient returns [∂/∂r , ∂/∂β] because arrays are (r, β)
    dB_dr, dB_dbeta = np.gradient(dB, dr, dbeta, edge_order=2)
    dR_dr, dR_dbeta = np.gradient(dR, dr, dbeta, edge_order=2)

    # Divergence: ∂β/∂β + ∂ṙ/∂r
    divergence = dB_dbeta + dR_dr

    # Curl (scalar in 2D): ∂ṙ/∂β - ∂β/∂r
    curl = dR_dbeta - dB_dr

    return divergence, curl

# ------------------------------------------------
# Calculate divergence and curl (replace your loop version with this)
# ------------------------------------------------
divergence, curl = calculate_divergence_curl(B, R, dB, dR, beta_vals, r_vals)

fig = plt.figure(figsize=(16, 12))
gs = fig.add_gridspec(2, 2, height_ratios=[2, 1], width_ratios=[1, 1], hspace=0.3, wspace=0.3)

# -------------------------------
# Top: Phase portrait (spans both columns)
# -------------------------------
ax1 = fig.add_subplot(gs[0, :])
ax1.streamplot(B, R, dB, dR, color="blue", density=3.0, linewidth=0.5)

if Plot_Trajectories == True:
    initial_conditions = [[0.1, 0.2], [-0.1, 0.2], [0.1, -0.2], [-0.1, -0.2]]
    for x0 in initial_conditions:
        sol = solve_ivp(f_nonlinear, [0, 15], x0, t_eval=np.linspace(0, 15, 50))
        ax1.plot(sol.y[0], sol.y[1], 'r-', lw=1.2, alpha=0.8)

# Equilibria
for i, eq in enumerate(find_equilibria()):
    print(f"Equilibrium {i + 1}: β = {eq[0]:.4f}, r = {eq[1]:.4f}")
    ax1.plot(eq[0], eq[1], 'ko', markersize=6)

ax1.set_xlabel("β [rad]")
ax1.set_ylabel("r [rad/s]")
ax1.set_title(f"Phase Portrait (δ={delta_deg}°, V={V} m/s)")
```

```
ax1.text(0.02, 0.98, '● Equilibrium Points', transform=ax1.transAxes,
      verticalalignment='top', fontsize=10, bbox=dict(boxstyle="round,pad=0.3",
      facecolor="white", alpha=0.8))
ax1.grid(True)
ax1.set_xlim(-Horizontal_axis, Horizontal_axis)
ax1.set_ylim(-Vertical_axis, Vertical_axis)

# ------------------------------
# Bottom left: Divergence
# ------------------------------
ax2 = fig.add_subplot(gs[1, 0])
div_plot = ax2.contourf(B, R, divergence, 50, cmap="RdYlBu")
plt.colorbar(div_plot, ax=ax2, label='Divergence')
ax2.set_xlabel("β [rad]")
ax2.set_ylabel("r [rad/s]")
ax2.set_title("Divergence (∇·F)")
ax2.grid(True)
ax2.set_xlim(-Horizontal_axis, Horizontal_axis)
ax2.set_ylim(-Vertical_axis, Vertical_axis)

# ------------------------------
# Bottom right: Curl
# ------------------------------
ax3 = fig.add_subplot(gs[1, 1])
curl_plot = ax3.contourf(B, R, curl, 50, cmap='RdYlBu')
plt.colorbar(curl_plot, ax=ax3, label='Curl')
ax3.set_xlabel("β [rad]")
ax3.set_ylabel("r [rad/s]")
ax3.set_title("Curl (∇×F)")
ax3.grid(True)
ax3.set_xlim(-Horizontal_axis, Horizontal_axis)
ax3.set_ylim(-Vertical_axis, Vertical_axis)

plt.tight_layout()
plt.show()

# Print some statistics

print(f"Divergence range: [{divergence.min():.3f}, {divergence.max():.3f}]")
print(f"Curl range: [{curl.min():.3f}, {curl.max():.3f}]")
```