????????Pacejka ???? (Magic Formula) ??

????????Pacejka ???? (Magic Formula) ??

這是一個非常核心且內行的問題！在 FSAE 或是任何高階車輛動態控制（ABS、TC、ESC）領域，Pacejka 輪胎模型（俗稱魔術公式 Magic Formula, MF）絕對是繞不開的「聖經級」模型。

既然你正在做牽引力控制（TC），理解輪胎如何把馬達的扭矩轉換成推動車身的力，是最基礎也最重要的一步。以下幫你把這個模型拆解成容易咀嚼的重點：

1. ??? Pacejka ?????

由荷蘭學者 Hans B. Pacejka 提出，它是一個半經驗模型（**Semi-empirical Model**）。

意思是：它並不是從嚴格的物理摩擦學推導出來的，而是拿著大量在測試機台（例如 FSAE 常用的 TTC 輪胎測試數據）上跑出來的數據，用一組巧妙的三角函數去「硬配適（Curve Fitting）」出來的曲線。

因為這組公式不僅能極度精準地擬合出輪胎的真實反應，而且曲線十分平滑（有利於電腦計算與微分），所以業界驚呼這簡直是「魔術」，故得名 Magic Formula。

2. ??????????

無論是算縱向力（加速/煞車）、橫向力（轉向）還是回正力矩，魔術公式的基本原型都是同一條長相奇特的方程式：

$$ y(x) = D \sin(C \arctan(Bx - E(Bx - \arctan(Bx)))) $$

- 輸入值 **x**：通常是縱向滑移率 λ（或稱 κ）或是側偏角 α。
- 輸出值 **y**：通常是縱向力 F_x 或是橫向力 F_y。

3. ??????????B, C, D, E?

這條公式之所以厲害，是因為它把複雜的曲線拆成四個具有「幾何意義」的常數。理解這四個常數，你就看懂了輪胎的靈魂：

- **D (Peak Factor - 峰值因子)**：
  - 意義：代表曲線的最高點，也就是輪胎能提供的最大抓地力（對應 μ_{peak}）。
  - 物理影響：D 越大，車子能承受的極限加速度或過彎 G 值就越高。
- **C (Shape Factor - 形狀因子)**：
  - 意義：控制曲線兩端漸近線的高度，決定了過了極限打滑點之後，抓地力下降的幅度。
  - 物理影響：對於縱向力 F_x，C 通常落在 1.65 左右；對於橫向力 F_y，C 約為 1.3。
- **B (Stiffness Factor - 剛性因子)**：
  - 意義：控制曲線在原點（$x=0$）附近的斜率（Slope）。
  - 物理影響：代表輪胎的縱向剛性或側向剛性。B 越大，曲線爬升越陡，車手會感覺這條輪胎反應非常直接、神經質；B 越小，輪胎反應越遲鈍、漸進。*（註：原點斜率其實是 $B \times C \times D$ 三個乘起來的結果。）*
- **E (Curvature Factor - 曲率因子)**：
  - 意義：控制曲線在達到峰值（D）附近的彎曲程度。
  - 物理影響：決定了輪胎在瀕臨極限時，是「瞬間失去抓地力」（Snap）還是「慢慢滑出去」（Progressive）。

4. ? Simulink ?????

在我們上一篇筆記中，你用的 Simulink QuarterCar 模型裡面的 `roadCoeffs`，其實是 Pacejka 的極度簡化版：

$$ \mu_x = c \sin(b \arctan(a\lambda)) $$

如果你把這個簡化版跟完整的魔術公式對照：

- 你的 **c** 就是 **D**（峰值 μ）。
- 你的 **b** 就是 **C**（形狀因子）。
- 你的 **a** 就是 **B**（剛性因子）。

差異：你的簡化版省略了 E（令 $E=0$）。這在一般控制演算法的初步驗證上已經很夠用了，但在精確的車輛動態模擬（如 CarSim 或圈速模擬器）中，通常會帶入包含 E 以及考慮垂直荷重 F_z 動態變化的完整版公式。

5. Pacejka ??????

痛點 / 特性	說明	對應到你的 **FSAE TC** 專案
精準度極高	穩態（Steady-state）下，對實車輪胎行為的描述幾乎無懈可擊。	用來建構 Simulink 的 Plant（受控體）非常完美。
需要龐大測試數據	模型本身沒有物理意義，你無法用「胎寬、橡膠配方」來算出 B, C, D, E，只能上機台硬測。	必須依賴 FSAE TTC 提供的原始數據來擬合參數。
運算量大、難以解析推導	對微控制器來說，算 $\sin$ 和 $\arctan$ 很耗資源；且在數學上很難直接反推或做線性化。	這就是為什麼 Savaresi 的書在設計控制器時，喜歡改用數學更簡單的 Burckhardt 模型或直接局部線性化。

?? MATLAB ?? TTC ???Curve Fitting ?????????? Pacejka

在實務上，FSAE 車隊通常會花很多時間在處理 TTC（輪胎測試聯盟）給的龐大數據檔。把 FSAE TTC 給的實測數據（CSV/MAT檔），透過 MATLAB 擬合（Curve Fitting）出專屬於你們車隊這套輪胎的 Pacejka B,C,D,E 參數。

這支程式使用了 MATLAB 內建的非線性最小平方法（`lsqcurvefit`），並內建了防呆機制：如果你還沒準備好實際的 TTC CSV 檔案，它會自動生成一組帶有雜訊的模擬數據，讓你立刻就能執行並看到擬合效果與圖表。

```
% =========================================================================
% FSAE TTC 輪胎數據 Pacejka 魔術公式 (Magic Formula) 曲線擬合腳本
% 適用於提取縱向力 (Fx) 或縱向摩擦係數 (Mu_x) 的 B, C, D, E 參數
% =========================================================================

clear; clc; close all;

%% 1. 讀取輪胎數據 (載入 TTC CSV 或產生模擬數據)
filename = 'TTC_tire_data.csv';

if isfile(filename)
    % 如果你有真實的 TTC 數據，讀取 CSV
    fprintf('讀取真實數據: %s\n', filename);
    data = readtable(filename);
    % 假設你的 CSV 欄位名稱為 SlipRatio 和 Mu_x (請依實際情況修改)
    kappa = data.SlipRatio;
```

```
    mu_x = data.Mu_x;
else
    % 若無檔案，產生一組帶有雜訊的模擬實測數據供測試
    fprintf('找不到 %s，產生模擬測試數據...\n', filename);
    kappa = linspace(-0.3, 0.3, 200)'; % 滑移率從 -30% 到 +30%

    % 設定一組虛擬的真實參數 [B=10, C=1.65, D=1.2, E=0.05]
    true_B = 10; true_C = 1.65; true_D = 1.2; true_E = 0.05;

    % 產生理想曲線並加入隨機雜訊 (模擬機台測試數據)
    ideal_mu = true_D * sin(true_C * atan(true_B.*kappa - true_E.*(true_B.*kappa -
atan(true_B.*kappa))));
    noise = 0.05 * randn(size(kappa));
    mu_x = ideal_mu + noise;
end

%% 2. 定義 Pacejka 魔術公式模型
% x(1) = B (Stiffness)
% x(2) = C (Shape)
% x(3) = D (Peak)
% x(4) = E (Curvature)
% xdata = kappa (滑移率)
magic_formula = @(x, xdata) x(3) .* sin(x(2) .* atan(x(1).*xdata - x(4).*(x(1).*xdata -
atan(x(1).*xdata))));

%% 3. 設定初始猜測值與上下界
% 合理的初始猜測值能幫助演算法更快、更準確收斂
x0 = [12.0, 1.5, 1.0, 0.0]; % 初始猜測: [B, C, D, E]

% 設定參數的上下界 (Lower Bound & Upper Bound)，避免擬合出沒有物理意義的負值或極端值
lb = [1.0,  1.0,  0.5, -1.0]; % 下界
ub = [20.0, 2.0,  2.0,  1.0]; % 上界

%% 4. 執行曲線擬合 (Curve Fitting)
% 使用 lsqcurvefit (需要 Optimization Toolbox)
options = optimoptions('lsqcurvefit', 'Display', 'iter', 'StepTolerance', 1e-6);

fprintf('\n開始進行 Pacejka 參數擬合...\n');
[x_opt, resnorm] = lsqcurvefit(magic_formula, x0, kappa, mu_x, lb, ub, options);
```

```
% 提取擬合結果
B_fit = x_opt(1);
C_fit = x_opt(2);
D_fit = x_opt(3);
E_fit = x_opt(4);

fprintf('\n=== 擬合結果 ===\n');
fprintf('B (剛性因子) = %.4f\n', B_fit);
fprintf('C (形狀因子) = %.4f\n', C_fit);
fprintf('D (峰值因子) = %.4f\n', D_fit);
fprintf('E (曲率因子) = %.4f\n', E_fit);
fprintf('殘差平方和   = %.4f\n', resnorm);

%% 5. 繪圖驗證擬合效果
kappa_dense = linspace(min(kappa), max(kappa), 500)'; % 產生高密度 X 軸以繪製平滑曲線
mu_x_fit = magic_formula(x_opt, kappa_dense);          % 計算擬合出的曲線

figure('Name', 'Pacejka Magic Formula Fitting', 'Color', 'w');
plot(kappa, mu_x, 'k.', 'MarkerSize', 8, 'DisplayName', 'TTC 原始測試數據'); % 畫出散點圖
hold on;
plot(kappa_dense, mu_x_fit, 'r-', 'LineWidth', 2, 'DisplayName', 'Pacejka 擬合曲線'); % 畫出擬合線
hold off;

grid on;
title('FSAE TTC 輪胎數據: 縱向滑移率 vs 摩擦係數');
xlabel('縱向滑移率 \lambda (Slip Ratio)');
ylabel('縱向摩擦係數 \mu_x (F_x / F_z)');
legend('Location', 'southeast');
xlim([min(kappa) max(kappa)]);

% 在圖表上標示出計算出的參數
annotation_str = sprintf('B = %.2f\nC = %.2f\nD = %.2f\nE = %.2f', B_fit, C_fit, D_fit,
E_fit);
text(min(kappa)*0.8, max(mu_x)*0.8, annotation_str, 'BackgroundColor', 'w', 'EdgeColor', 'k',
'FontSize', 10);
```

????????

1. Toolbox 需求: 這支腳本使用了 `lsqcurvefit`，你的 MATLAB 需要安裝 `Optimization Toolbox` 才能執行。

2. 導入真實數據: 當你要換成真實 TTC 數據時, 只需把你的檔名填入 `filename = '你的檔名.csv';`, 並確保 `kappa` 和 `mu_x` 抓取的是正確的欄位（TTC 的原始檔通常欄位名是 `SL` 代表 **Slip Ratio**, `FX` 代表 縱向力，記得要先把 F_x 除以垂直荷重 F_z 換算成 μ_x 再丟進來擬合會比較直觀）。

Revision #2
Created 2026-04-01 02:06:35 UTC by TaipeiTechRacing
Updated 2026-04-11 14:33:44 UTC by AI Agent