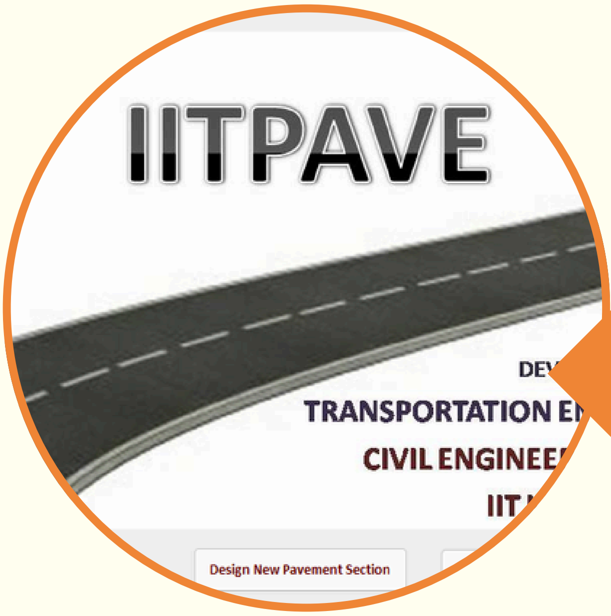
CEC OPEN PROJECT 2025

Project Report

Automated Pavement Response Analysis Using IITPAVE



Prepared by
Ankit Lal

Table of Contents

Project Summary

This project presents a Python-based automation pipeline to evaluate pavement response using IITPAVE as per IRC:37-2018. The script generates multiple input cases by varying bituminous layer modulus (E1), DBM modulus (E2), and BC thickness (h1), while keeping other parameters fixed as per IRC recommendations. For each combination, the tool extracts critical stress and strain responses at two depths (h1 and h1/2) and stores the results in a structured CSV file. Special care was taken to comply with IRC standards notably while storing the values. The output enables detailed mechanistic analysis and supports comparative pavement design studies at scale.

Objective

To develop a Python-based automation pipeline for generating and analyzing pavement layer responses using the IITPAVE software. The aim is to study the impact of varying bituminous layer modulus and thickness on stresses and strains and to complete the table provided in the problem statement.

Tools used:

IITPAVE software and Python (with csv, os, subprocess, re)

Methodology

This project follows a systematic approach to automate the process of generating input cases, running IITPAVE, and extracting critical pavement response values for analysis. The methodology is divided into three key components: parameter variation, file I/O automation, and output parsing.

Parameter Variation:

The analysis focuses on understanding how variations in the bituminous concrete (BC) layer properties influence pavement performance. Three key parameters were varied:

- E1: Modulus of the BC layer (2000 to 4000 MPa)
- E2: Modulus of the DBM layer (300 to 5000 MPa)
- h1: Thickness of the BC layer (40 mm to 150 mm)

All other parameters — such as Poisson's ratio (0.35 for all layers), DBM thickness (100 mm), granular thickness (450 mm), subgrade modulus (76.8 MPa), wheel load (20000 N), and tyre pressure (0.56 MPa) — were kept constant as required by the problem statement.

Input File Generation and Execution:

For each combination, the script dynamically creates an IITPAVE.IN file in the format accepted by the software. This input file contains:

- Number of layers and their moduli
- Poisson's ratios
- Layer thicknesses
- Wheel load and tyre pressure
- Number of points of analysis: 2 points per case
    - At depth = h1 (bottom of BC)
    - At depth = h1/2 (mid-depth of BC)
    - At radial distance = 0 mm for both

After the input file is written, the script triggers IITPAVE.exe via Python's subprocess.run() to execute the analysis and generate an output file IITPAVE.OUT.

Output Parsing and Data Extraction:

The script then reads the output file and parses it line-by-line. It extracts numerical values using regular expressions, but only retains values at radial distance = 0 mm and depth = h1 or h1/2.

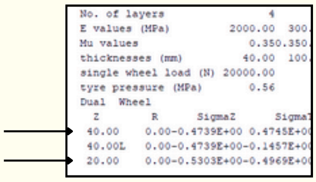
The extracted stress and strain values — including vertical stress (σz), tangential and radial stresses (σt, σr), shear stress (τrz), and the corresponding strains (εz, εt, εr) — are stored for both depths and written into a structured CSV file (automated_iitpave_response.csv).

Some Important Considerations

Radial Distance:

In this study, the critical horizontal tensile strain (εt) at the bottom and mid-depth of the bituminous concrete (BC) layer has been evaluated at a radial distance of 0 mm from the center of the applied wheel load. This follows standard design practice as observed in IRC:37-2018. Although IRC examples also consider r = 155 mm (midpoint between dual wheels) to capture conservative effects due to stress bulb overlap, such cases are usually relevant for deeper layers like the subgrade. For the BC layer specifically, εt is observed to be consistently higher at r = 0 mm in most configurations. Therefore, limiting the strain and other evaluations to r = 0 mm is both technically sound and practical.

Relevant rows:



```
No. of layers                  4
E values (MPa)           2000.00  300.
Mu values                   0.350.350.
thicknesses (mm)           40.00  100.
single wheel load (N) 20000.00
tyre pressure (MPa)         0.56
Dual  Wheel
  Z         R      SigmaZ      SigmaT
 40.00     0.00-0.4739E+00 0.4745E+00
 40.00L    0.00-0.4739E+00-0.1457E+00
 20.00     0.00-0.5303E+00-0.4969E+00
```

As per IRC:37-2018, page 67:
For interface depths, IITPAVE provides two sets of results — one with layer-above properties and another with layer-below properties. The second set (layer-below) is suffixed with L. For computing horizontal tensile strain (εt) in the BC layer, we must use results corresponding to the layer above the interface — i.e., the row without the L suffix.
Hence:

- For bituminous layers, only values from the first set (without suffix L) are meaningful.
- Values with 'L' are relevant only for subgrade strains or other bottom layer evaluations.

Assumptions:

The wheel load and tyre pressure were chosen based on standard recommendations in IRC:37-2018. A single wheel load of 20000 N (representing a standard axle load of 80 kN) and a tyre pressure of 0.56 MPa were assumed. These values are widely used in design examples. Poisson's ratio of 0.35 has also been taken for all the layers following IRC:37-2018.

Automation Code (Steps)

Importing Required Modules

```
import os
import subprocess
import re
import csv
```

Defining Parameters and Paths

```
# Constants
E3, E4 = 240.2, 76.8
h2, h3 = 100, 450
poisson_ratio = 0.35
tyre_pressure = 0.56
wheel_load = 20000

# Parameter sets
E1_list = [2000, 2500, 3000, 3500, 4000]
E2_list = [300, 500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000]
h1_list = [40.0, 50.0, 60.0, 70.0, 80.0, 90.0, 100.0, 110.0, 120.0, 130.0, 140.0, 150.0]

# Paths
base = r"C:\Users\Rajeshwar Lal\Desktop\IITPAVE\IRC_37_IITPAVE\IRC_37_IITPAVE\IRC_37_IITPAVE"
exe = os.path.join(base, "IITPAVE.exe")
inp = os.path.join(base, "IITPAVE.IN")
out = os.path.join(base, "IITPAVE.OUT")
csv_path = os.path.join(base, "automated_iitpave_response.csv")
```

CSV file structure and initialization

```
# Open CSV writer
with open(csv_path, "w", newline="") as file:
    writer = csv.writer(file)
    writer.writerow([
        "E1", "E2", "E1/E2", "h1",
        "sigz_h1", "sigt_h1", "sigr_h1", "tau_h1", "epz_h1", "ept_h1", "epr_h1", "h1/2",
        "sigz_h1/2", "sigt_h1/2", "sigr_h1/2", "tau_h1/2", "epz_h1/2", "ept_h1/2", "epr_h1/2"
    ])
```

Automation Code (Steps)

Generating Input and Running IITPAVE

```python
for E2 in E2_list:
    for E1 in E1_list:
        for h1 in h1_list:
            with open(inp, "w") as f:
                f.write("4\n")
                f.write(f"{E1} {E2} {E3} {E4}\n")
                f.write(f"{poisson_ratio} {poisson_ratio} {poisson_ratio} {poisson_ratio}\n")
                f.write(f"{h1} {h2} {h3}\n")
                f.write(f"{wheel_load} {tyre_pressure}\n")
                f.write("2\n")
                f.write(f"{h1} 0\n{h1/2} 0\n")
                f.write("2\n")

            subprocess.run([exe], cwd=base, check=True)
```

Parsing IITPAVE Output File

```python
            values = {"h1": {}, "h1/2": {}}
            with open(out, "r") as f:
                for line in f:
                    tokens = line.strip().split()
                    if not tokens or 'L' in tokens[0]:
                        continue
                    nums = re.findall(r'[-+]?\d*\.\d+(?:[Ee][-+]?\d+)?', line)
                    if len(nums) >= 10:
                        z, r = float(nums[0]), float(nums[1])
                        if r != 0:
                            continue
                        data = dict(zip(
                            ["sigz", "sigt", "sigr", "tau", "epz", "ept", "epr"],
                            [nums[i] for i in [2, 3, 4, 5, 7, 8, 9]]
                        ))
                        if abs(z - h1) < 0.5 and not values["h1"]:
                            values["h1"] = data
                        elif abs(z - h1 / 2) < 0.5 and not values["h1/2"]:
                            values["h1/2"] = data
```

Storing Valid Results to CSV

```python
            if values["h1"] and values["h1/2"]:
                writer.writerow([
                    E1, E2, round(E1 / E2, 2), h1,
                    *[values["h1"].get(k, "") for k in ["sigz", "sigt", "sigr", "tau", "epz", "ept", "epr"]],
                    h1 / 2,
                    *[values["h1/2"].get(k, "") for k in ["sigz", "sigt", "sigr", "tau", "epz", "ept", "epr"]]
                ])
```

Automation Code

IITPAVE_automation.py | IITPAVE.IN

IITPAVE.IN

```
4
2500 2500 240.2 76.8
0.35 0.35 0.35 0.35
120.0 100 450
20000 0.56
2
120.0 0
60.0 0
2
```

IITPAVE_automation.py | IITPAVE.IN

IITPAVE.IN

```
4
3000 1500 240.2 76.8
0.35 0.35 0.35 0.35
40.0 100 450
20000 0.56
2
40.0 0
20.0 0
2
```

The code dynamically generates a new IITPAVE.IN file for each combination of E1, E2, and h1 values by writing layer moduli, Poisson's ratios, thicknesses, and loading conditions in the format expected by IITPAVE. The points of analysis — specifically at depths h1 and h1/2, both at radial distance 0 mm — are inserted into the file in each loop. This automation allows the input file to be updated case-by-case without manual effort.

The attached screenshot demonstrates how the input file is updated with correct values for each case during iteration.

IITPAVE_automation.py | IITPAVE.IN | iitpave.out

iitpave.out

```
No. of layers              4
E values (MPa)        4000.00 5000.00  240.20   76.80
Mu values                0.350.350.350.35
thicknesses (mm)       150.00  100.00  450.00
single wheel load (N) 20000.00
tyre pressure (MPa)      0.56
Dual  Wheel
  Z        R      SigmaZ      SigmaT     SigmaR     TaoRZ      DispZ      epZ        epT        ep
150.00    0.00-0.2042E+00 0.4433E-02-0.1163E-01-0.4470E-01 0.2427E+00-0.5043E-04 0.2000E-04 0.145
150.00L   0.00-0.2042E+00 0.3303E-01 0.1296E-01-0.4470E-01 0.2427E+00-0.4406E-04 0.1999E-04 0.145
 75.00    0.00-0.4380E+00-0.2263E+00-0.2191E+00-0.3236E-01 0.2473E+00-0.7052E-04 0.9222E-06 0.334
```

IITPAVE_automation.py | IITPAVE.IN | iitpave.out

iitpave.out

```
No. of layers              4
E values (MPa)        4000.00 5000.00  240.20   76.80
Mu values                0.350.350.350.35
thicknesses (mm)       150.00  100.00  450.00
single wheel load (N) 20000.00
tyre pressure (MPa)      0.56
Dual  Wheel
  Z        R      SigmaZ      SigmaT     SigmaR     TaoRZ      DispZ      epZ        epT        ep
150.00    0.00-0.2042E+00 0.4433E-02-0.1163E-01-0.4470E-01 0.2427E+00-0.5043E-04 0.2000E-04 0.145
150.00L   0.00-0.2042E+00 0.3303E-01 0.1296E-01-0.4470E-01 0.2427E+00-0.4406E-04 0.1999E-04 0.145
 75.00    0.00-0.4380E+00-0.2263E+00-0.2191E+00-0.3236E-01 0.2473E+00-0.7052E-04 0.9222E-06 0.334
```

For every new combination of E1, E2, and h1, the code first creates an updated IITPAVE.IN file and then runs the IITPAVE executable. Upon execution, IITPAVE automatically generates a fresh IITPAVE.OUT file, overwriting the previous one. This output contains the calculated stress and strain values at the specified depths and radial distances for that particular case.

The attached screenshot illustrates how IITPAVE.OUT gets regenerated each time with new values corresponding to the input just written, showing different depths (Z), radial locations (R), and associated response parameters.

CSV output structure

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
E1	E2	E1/E2	h1	sigz_h1	sigt_h1	sigr_h1	tau_h1	epz_h1	ept_h1	epr_h1	h1/2	sigz_h1/2	sigt_h1/2	sigr_h1/2	tau_h1/2	epz_h1/2	ept_h1/2	epr_h1/2

The final CSV file automated_iitpave_response.csv stores the critical stress and strain responses at two depths within the bituminous concrete (BC) layer: the bottom (h1) and the mid-depth (h1/2). For each combination of E1, E2, and h1, the following variables are recorded:

General Parameters:

- E1 – Bituminous layer modulus (MPa)
- E2 – DBM layer modulus (MPa)
- E1/E2 – Modulus ratio
- h1 – Bituminous layer thickness (mm)
- h1/2 – Mid-depth of the BC layer (mm)

E1	E2	E1/E2	h1	sigz_h1	sigt_h1	sigr_h1	tau_h1	epz_h1	ept_h1	epr_h1	h1/2	sigz_h1/2	sigt_h1/2	sigr_h1/2	tau_h1/2	epz_h1/2	ept_h1/2	epr_h1/2
2000	300	6.67	40	-4.74E-01	4.75E-01	4.33E-01	-1.02E-02	-3.96E-04	2.45E-04	2.16E-04	20	-5.30E-01	-4.97E-01	-4.63E-01	-7.34E-04	-9.72E-05	-7.47E-05	-5.16E-05
2000	300	6.67	50	-4.16E-01	5.94E-01	5.23E-01	-1.25E-02	-4.03E-04	2.78E-04	2.30E-04	25	-5.04E-01	-4.13E-01	-3.87E-01	-7.72E-04	-1.12E-04	-5.05E-05	-3.31E-05
2000	300	6.67	60	-3.63E-01	6.40E-01	5.47E-01	-1.48E-02	-3.89E-04	2.88E-04	2.25E-04	30	-4.80E-01	-3.50E-01	-3.31E-01	-2.52E-03	-1.21E-04	-3.32E-05	-2.03E-05
2000	300	6.67	70	-3.20E-01	6.50E-01	5.44E-01	-1.68E-02	-3.69E-04	2.86E-04	2.14E-04	35	-4.61E-01	-3.01E-01	-2.88E-01	-5.64E-03	-1.27E-04	-1.95E-05	-1.06E-05
2000	300	6.67	80	-2.83E-01	6.39E-01	5.24E-01	-1.83E-02	-3.45E-04	2.78E-04	1.99E-04	40	-4.46E-01	-2.63E-01	-2.54E-01	-9.42E-03	-1.32E-04	-9.00E-06	-3.05E-06
2000	300	6.67	90	-2.52E-01	6.17E-01	4.98E-01	-1.93E-02	-3.21E-04	2.66E-04	1.85E-04	45	-4.35E-01	-2.31E-01	-2.26E-01	-1.34E-02	-1.38E-04	1.96E-07	3.71E-06
2000	300	6.67	100	-2.26E-01	5.91E-01	4.71E-01	-1.98E-02	-2.99E-04	2.53E-04	1.72E-04	50	-4.25E-01	-2.04E-01	-2.02E-01	-1.71E-02	-1.42E-04	7.94E-06	9.13E-06
2000	300	6.67	110	-2.04E-01	5.62E-01	4.45E-01	-2.00E-02	-2.78E-04	2.39E-04	1.60E-04	55	-4.16E-01	-1.80E-01	-1.81E-01	-2.05E-02	-1.45E-04	1.45E-05	1.37E-05
2000	300	6.67	120	-1.85E-01	5.34E-01	4.20E-01	-1.99E-02	-2.59E-04	2.26E-04	1.49E-04	60	-4.09E-01	-1.60E-01	-1.63E-01	-2.34E-02	-1.48E-04	2.01E-05	1.82E-05
2000	300	6.67	130	-1.68E-01	5.05E-01	3.96E-01	-1.96E-02	-2.42E-04	2.13E-04	1.39E-04	65	-4.01E-01	-1.41E-01	-1.46E-01	-2.59E-02	-1.50E-04	2.51E-05	2.19E-05
2000	300	6.67	140	-1.53E-01	4.77E-01	3.73E-01	-1.91E-02	-2.26E-04	2.00E-04	1.30E-04	70	-3.93E-01	-1.25E-01	-1.31E-01	-2.79E-02	-1.52E-04	2.91E-05	2.52E-05
2000	300	6.67	150	-1.40E-01	4.51E-01	3.51E-01	-1.86E-02	-2.10E-04	1.89E-04	1.21E-04	75	-3.84E-01	-1.11E-01	-1.18E-01	-2.96E-02	-1.52E-04	3.25E-05	2.75E-05
2500	300	8.33	40	-4.58E-01	6.81E-01	6.19E-01	-1.11E-02	-3.65E-04	2.50E-04	2.17E-04	20	-5.22E-01	-5.02E-01	-4.67E-01	-1.46E-05	-7.33E-05	-6.24E-05	-4.32E-05
2500	300	8.33	50	-3.96E-01	7.89E-01	6.93E-01	-1.35E-02	-3.66E-04	2.74E-04	2.22E-04	25	-4.94E-01	-4.13E-01	-3.87E-01	-4.56E-04	-8.55E-05	-4.19E-05	-2.78E-05
2500	300	8.33	60	-3.42E-01	8.17E-01	6.97E-01	-1.59E-02	-3.49E-04	2.77E-04	2.12E-04	30	-4.68E-01	-3.48E-01	-3.29E-01	-3.11E-03	-9.25E-05	-2.75E-05	-1.73E-05

Snap of Output CSV file

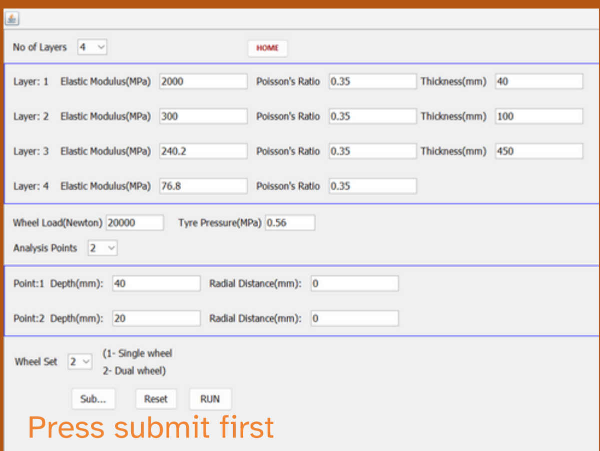
At depth h1:

- sigz_h1 – Vertical stress
- sigt_h1 – Tangential stress
- sigr_h1 – Radial stress
- tau_h1 – Shear stress (τrz)
- epz_h1 – Vertical strain
- ept_h1 – Tangential (horizontal tensile) strain
- epr_h1 – Radial strain

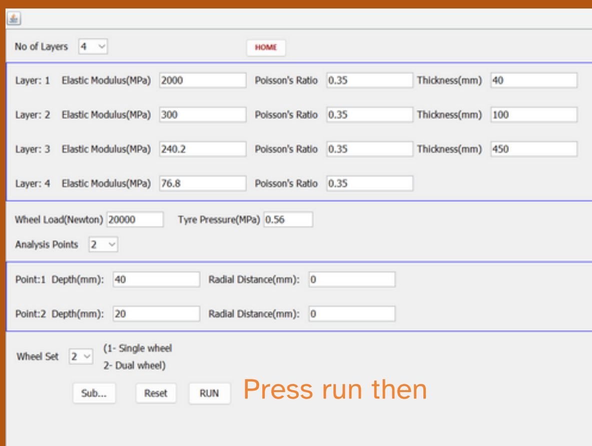
At depth h1/2:

sigz_h1/2 – Vertical stress
sigt_h1/2 – Tangential stress
sigr_h1/2 – Radial stress
tau_h1/2 – Shear stress (τrz)
epz_h1/2 – Vertical strain
ept_h1/2 – Tangential (horizontal tensile) strain
epr_h1/2 – Radial strain

Manual input Case



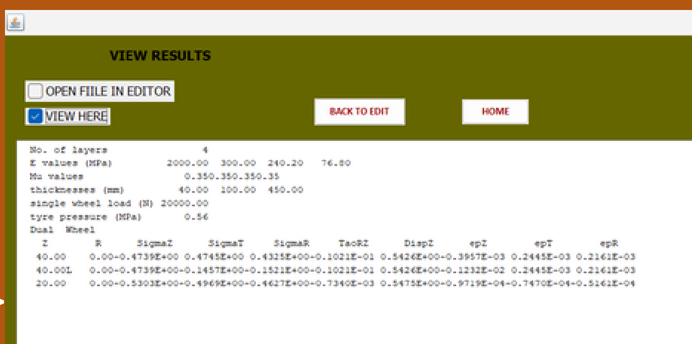
Press submit first



Press run then

Values are entered in GUI interface of the software. Once submitted and run, output is displayed.

OUTPUT(hence verified from the snap above)-->



Steps to Operate the System

1. Place IITPAVE.exe and other files in the same folder as your script.
2. Run the Python file using any Python environment (IITPAVE_automation.py). Make sure to change the path of the file in the script.
3. The script will automatically:
   - Make changes in IITPAVE.IN
   - Execute IITPAVE.exe
   - Read IITPAVE.OUT
   - Append results to automated_iitpave_response.csv
4. The process loops over all E1, E2, h1 combinations and saves the full dataset in the final CSV.

GitHub link for code: https://github.com/ankitlal-iitr/IITPAVE_automation.git

Conclusion

Through this project, I developed a clear understanding of pavement response evaluation and gained hands-on experience with using IITPAVE for mechanistic analysis. I also familiarized myself with key concepts from IRC:37-2018, such as critical strain locations and the importance of evaluating tensile strains at the bottom of the bituminous layer for design purposes. By writing a Python script to automate input generation, execution, and output extraction, I was able to study how variations in bituminous layer properties affect pavement performance. The final CSV output allows for easy comparison and further analysis. Overall, this project helped me strengthen both my technical coding skills and my understanding of pavement design principles.