EET 109 Coding Task Details

Instructor: Parikshit Pareek

1. DC Power Flow Task

Task

You must write a `.py` file (named exactly as your roll number, e.g. `21241.py`) that implements a **Fast & Accurate DC Power Flow** solver for any `PGLib` data file:

1. **Load .m-file:** Read a .m-file from (PGLib) using `scipy.io.loadmat` or `pypower.case.loadcase` or as given in the code snip `roll_number.py`.
2. **Build Susceptance Matrix:** Assemble the full bus susceptance matrix B_{bus}.
3. **Solve DCPF:** Solve DCPF for angle vector at each node.
4. **Scale & Speed:** Your code will be tested on systems up to several thousand busesuse vectorized routines and sparse solvers to maximize speed. GPU use is allowed, (must have availability condition).

2. Submission Guidelines

1. Submit exactly two files using the official Google Form: Link
   - Your code: `<rollno>.py`
   - Python version $\geq$ 3.11
   - A plain text file listing any additional Python packages you used (excluding `time`, `numpy`, and `load_pglib_opf`)
2. Only pip-installable packages from PyPI are allowed. No system packages or private modules.
3. Your code will be graded based on:
   (a) **Accuracy (40 points)** Based on root mean squared error (RMSE) of voltage angles with respect to the correct solution.
   (b) **Time Efficiency (40 points)** Relative speed (wrt $\mathcal{O}(cN^3)$ runtime) based on growth of runtime with system size.
   (c) **Package Discipline (20 points)** Excessive use of packages will reduce your score, with **very heavy** penalty. Packages in `<rollno>.py` + Three packages are Free.
4. Scripts that fail due to missing or incorrect dependencies may be graded as non-working submissions.
5. **Do not use AI/LLM tools (e.g., ChatGPT, Copilot) beyond asking for syntax corrections**. Automated tools will detect such submissions. Violators will receive zero marks and may face disciplinary action.

2.1 Standing Instruction

At the very end of your `.py` file you **MUST** append exactly the following (so that our autograder can invoke your `test()` automatically):

```
if __name__ == "__main__":
    import sys, json
    case_file = sys.argv[1]
    result = test(case_file)
    sys.stdout.write(json.dumps(result))
```

3. Read or Regret

The following types of submissions will be marked as invalid and may receive **zero marks**:

1. **Submission does not run:** Your `<rollno>.py` file must execute without any runtime errors or import failures when run as: `python3 <rollno>.py <casefile.m>`.
2. **Incorrect or missing output:** Your script must return a list of bus voltage angles in `test()` using the expected format. If the return is missing, malformed, or not interpretable, your code will be considered invalid.
3. **Modification of restricted code:** You are not allowed to change any lines marked as `# Do not modify below/above this line` in the provided template. Any modification to protected sections will result in disqualification.
4. **External Libraries:** You may only use standard libraries such as `numpy`, `scipy`, and built-in Python libraries. Submissions using other third-party packages will be disqualified.
5. **Detected use of LLMs:** Use of Large Language Models (LLMs) such as ChatGPT, Gemini, Copilot, etc., is **strictly prohibited beyond syntax clarification or documentation lookup**. If your code is flagged by our automated checker as LLM-generated, you will be awarded **zero marks** without further discussion.
6. **Plagiarism:** Identical or nearly identical submissions (even if variable names or formatting differ) will be treated as plagiarism. All students involved will receive zero. Remember an automatic code-plag checker will be used to detect such incidents.

Allowed:

- You may use basic online documentation (e.g., numpy/scipy docs, StackOverflow for syntax).
- You may discuss general concepts of DCPF or linear algebra with peers, but code sharing is not permitted.
- You are encouraged to write clean, readable, and well-commented code using the given starter template.

Reminder: The grading script will test accuracy across multiple cases and measure runtime. Submissions should be robust and reasonably efficient. Make sure to test your code thoroughly before submission.