Google Summer of Code 2025 Proposal

Project Title:Compiling LAPACK with LFortran

1 Personal Information

1.1 Contact Information

- **University -** Motilal Nehru National Institute of Technology
- **Email IDs -** samimshoaib01@gmail.com, shoaib.20233563@mnnit.ac.in
- **GitHub/Portfolio -** samimshoaib01
- **Timezone -** IST (UTC + 5:30)

1.2 Personal Background

I am a second year undergraduate in the department of Computer Science and Engineering at Motilal Nehru National Institute of Technology, Allahabad.

I have completed the following relevant courses in my academic curriculum in the past years :Compiler Design, Computer Organisation and Architecture, Theory of Computation,Compiler Design, Object Oriented Analysis and Design.

1.3 Programming Background

I currently use Ubuntu 24.04.2 LTS as my primary development system, with Visual Studio Code as my main editor, supplemented by experience with Eclipse IDE and Atom. While I have programming experience across multiple languages including Python 3.x, Java, and web technologies (HTML/CSS/JavaScript) from my undergraduate studies, my recent focus has been on C++ development.My multi-language background helps me approach problems from

different perspectives while maintaining strong C++ fundamentals for this LFortran related work.I have recently learnt fortran which will give me additional advantage.

A list of my ongoing and completed projects is as follows :

- **Campus Rush -** I participated in a university hackathon and secured the position of second runner-up. I developed a 2D RPG game using Phaser.js for the game engine and the MERN stack for backend functionalities. The project required efficient optimization and integration between frontend and backend. Competing under strict time constraints, I gained valuable insights into rapid prototyping and problem-solving.

- **Digital Vernier Caliper Simulator (Physics/Engineering Lab) -** I, along with four team members, developed a Digital Vernier Caliper Simulator to assist students in learning precision measurements in physics and engineering labs. The application, built using Qt, provides an interactive interface where users can adjust the caliper jaws using sliders to measure virtual objects with high accuracy. It displays the main scale reading, vernier scale reading, and the final measurement with least count correction. This tool helps students understand the working of a vernier caliper without needing physical instruments, making it useful for remote learning and virtual labs.

- **AI Powered Coding Assistant** - Developed an AI-powered Chrome extension using OpenAI API to assist with coding challenges, featuring a seamless SPA-integrated UI. Optimized AI prompts for accuracy and added chat history storage, enhancing productivity for developers. Designed for scalability

and performance, aligning with MAANG's high standards for user experience

- **Traffic Helper (ongoing) -** Developed a deep learning-based traffic sign detection system using PyTorch, featuring a custom dataset loader and a CNN with three convolutional layers, max pooling, and dropout regularization. Trained the model with cross-entropy loss and optimized it for high classification accuracy. Evaluated performance using test data and visualized loss curves to track training progress. Designed for real-world applications like autonomous driving and smart traffic management.

  Details about these projects and more are available on my GitHub profile. For all the projects mentioned above, I have used Git as my version control system.

2 The Project

2.1 Information

LAPACK ("Linear Algebra Package") is a standard software library for numerical linear algebra. It provides routines for solving systems of linear equations and linear least squares, eigenvalue problems, and singular value decomposition. It was written in **Fortran 90**.

LFortran is an interactive LLVM-based Fortran compiler designed to be modern, open-source, and user-friendly. A critical step toward its beta release is ensuring that LFortran can compile as much of the LAPACK routines as possible. LAPACK is a widely used numerical library written in Fortran, and successfully compiling it with LFortran

requires implementing missing Fortran semantics. This project aims to enhance LFortran's support for complex Fortran features and enable seamless compilation of LAPACK.

2.2 Issues

I used this as reference.

These are the issues which I was able to deduce from my side. There might be more issues or I might be wrong in this, will work along with mentor to make it work.

Compiling LAPACK with LFortran will require implementing missing Fortran features, including:

- **Unimplemented Intrinsics**: While functions like radix and minexponent are implemented in LFortran, LAPACK routines like snrm2.f90 still fail to compile due to potential edge cases or usage contexts. These need verification and possible test additions
- **Fixed-Form Source Handling:**Fixed-Form Source Handling: LAPACK uses a mix of fixed-form (.f) and free-form (.f90) files. LFortran should infer fixed-form from file extensions when using the --fixed-form-infer flag. This requires improvements to the frontend's file format detection logic and test coverage.
- **Legacy Array Syntax**: Type mismatches occur when passing scalar values to array arguments (e.g., f32 vs f32[:,:]).
- **Runtime Linking**: LFortran's runtime library (liblfortran_runtime) is not linked automatically, leading to undefined symbol errors.

  and there might be more issues which will be identified and work on it in this period.

2.3 Timeline

Community Bonding

- Engage with the LFortran community.
- Study LFortran internals and analyze LAPACK requirements.

Weeks 1-2: Intrinsic Verification & Format Inference

- Validate existing intrinsics (`radix`, `minexponent`, etc.) by compiling test cases from LAPACK.
- Add tests or fixes for edge cases if needed.
- Investigate and enhance `--fixed-form-infer` support:
    - Improve file format detection from `.f` and `.f90` extensions.
    - Add test cases for mixed fixed/free-form sources.

Weeks 3-4: Runtime Linking & Error Debugging

- Modify the LFortran driver to support automatic runtime library linking (`liblfortran_runtime`).
- Fix linking issues causing undefined symbol errors (e.g., `cange`).
- Work on resolving LLVM backend assertion failures (`asr_to_llvm.cpp`) by identifying unsupported constructs.

Weeks 5-7: Interface & Array Compatibility

- Add support or workarounds for:
    - Implicit interfaces (`--implicit-interface` handling),
    - Legacy array syntax and scalar vs. array mismatches.
- Start compiling larger LAPACK files and logging unresolved issues.

Weeks 8-10: LAPACK-Wide Compilation & Patching

- Try compiling complete submodules of LAPACK.
- Apply fixes or submit patches to LFortran for features blocking compilation.
- Collaborate with mentors to prioritize fixes for critical errors.

Final Week: Testing, Validation & Submission

- Conduct final testing and validation.
- Ensure full LAPACK compilation.
- Submit patches and finalize documentation.

This timeline is just for the reference,will work as per the mentors say or the errors occurs while making LAPACK compile .

2.4 Expected Outcomes

- LFortran will be able to compile all LAPACK code.

- New Fortran semantics will be implemented and tested in LFortran.
- Comprehensive documentation and test cases will be contributed to LFortran.

2.5 Why Me?

I am highly interested in compiler development and have prior experience in:

- Working with Fortran and understanding numerical libraries.
- C++ programming, making me comfortable contributing to LFortran.
- Exploring compiler internals, which aligns with my learning and career goals
- Previous Contributions to LFortran :
    - Merged :
        - #6604 : Fix handling of different kinds of arguments in shifta intrinsic

    I am also understanding and working on other issues .

2.6 Mentor

- **Ondřej Čertík**
- **Pranav Goswami**

3 Future Involvement

After GSoC, I plan to continue contributing to LFortran by improving its support for scientific computing libraries like LAPACK. I aim to enhance LFortran's compatibility with modern Fortran features

and address performance optimizations. Additionally, I am interested in exploring compiler optimizations, expanding LFortran's backend support, and contributing to its long-term development.

4 Commitment & Availability

- I confirm that I will be available full-time during GSoC 2025.
- No academic or professional commitments will interfere with my work.

5 References

- LFortran Documentation
- Github
- LAPACK