International Fortran Conference 2021

The State of Fortran 2021

September 2021

Laurence Kedward

Bálint Aradi, Ondřej Čertík, Milan Curcic, Sebastian Ehlert, Philipp Engel, Rohit Goswami, Michael Hirsch, Asdrubal Lozada-Blanco, Vincent Magnin, Arjen Markus, Emanuele Pagone, Ivan Pribec, Brad Richardson, Harris Snyder, Carlos Une, John Urban, Jérémie Vandenplas

Overview

Part 1: Why Fortran?

- 60+ years of high-level, high-performance programming

Part 2: Fortran-Lang

- State of the Fortran ecosystem
- A new online community for Fortran users

Part 3: The Future is Bright

- Fortran-Lang open source software development
- Modern tooling: `stdlib` and `fpm`

Fortran

FORTRAN—FORmula TRANslator

Developed by John Backus' team at IBM in 1954-1957 to ease the translation of mathematical formulas to machine code for scientists and engineers.

- The first **high-level, optimized, cross-platform** programming language
- Highly successful: many scientific applications and libraries developed in Fortran
- Core language is still under active development, 60 years later—latest standard is 2018

What makes Fortran effective for high-performance numerical computing in 202X?

Part 1: Why Fortran?

What makes Fortran effective for high-performance numerical computing in 2021-202X?

Why Fortran? (1)—Performant High-level Programming

Array-Oriented

- Intrinsic multidimensional arrays
- Expressive array slicing
- Non-aliasing arguments
- Compiler-optimized array operations
- `elemental` procedures

Safe

- Strong, static typing
- Simple syntax—easy to learn
- Standardized and portable
- Dynamic allocation with static scoping

Parallel

- **Natively parallel:** `do concurrent`, coarrays, teams, events, and collectives
- **Parallel APIs:** *OpenMP*, *OpenACC*, *MPI*

Why Fortran? (2)—Stability and Reliability

Language Stability

As an actively developed language with a long history, Fortran and its associated compilers have long maintained **excellent backward compatibility** while supporting modern revisions to the language.

Code Longevity

- No breaking changes to the core language
- Supports very large projects where validation and verification are costly and time-consuming
- Fortran programs and libraries can be relied upon to compile and run in the future
- Compilers have kept up-to-date with the latest hardware developments

Why Fortran? (3)—Mature

GFortran	Open source, GPLv3	Full support for F2003, partial support for F2008 and F2018
Classic flang	Open source, Apache-2.0	Full support for F2003, to be superseded by LLVM flang
LLVM flang	Open source, Apache-2.0	Under development, full support for parsing F2018
LFortran	Open source, BSD-3C	Under development, full support for parsing F2018
Intel Classic	Proprietary, Intel	Full support for F2018
Intel LLVM	Proprietary, Intel	Beta development, full support for F95
nvfortran	Proprietary, NVIDIA	Full support for F2003, partial support for F2008

- Multiple freely-available compilers with active support and development
- Many existing libraries for numerical and scientific software
- Standardized interoperability with `C`
- Standardized extensions with *MPI*, *OpenMP*, *OpenACC*

Why Fortran?

What makes Fortran effective for high-performance numerical computing in 2021?

- **Performance**
- **High-level abstraction**
- **Easy-to-learn**
- **Productivity**
- **Portability**
- **Stability**
- **Maturity**
- **Longevity**

The State of Fortran 2021

Part 2: Fortran-Lang

A new open source community for Fortran users

Fortran Ecosystem

The Problem

The ecosystem and tooling around Fortran has stagnated across multiple fronts in comparison to that of modern programming languages.

No Standard Library—*Achieving general-purpose programming tasks in Fortran, such as string handling, is difficult and duplicates effort*

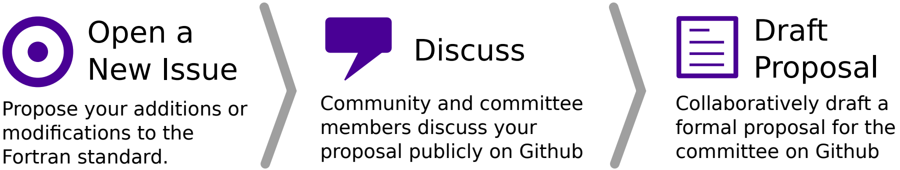
Building and Distributing Fortran Software is Difficult—*This presents a high barrier to entry and discourages software reuse*

No Community-Maintained Compiler—*For prototyping new features and developing new tooling*

No Prominent Dedicated Website—*Essential for new users to discover and learn Fortran*

Formation of Fortran-Lang

- **August 2019** Conversations on Twitter between Ondřej Čertík, Milan Curcic, and Jacob Williams bring out common perceived shortcomings in the Fortran ecosystem
- **October 2019** Ondřej creates the `j3-fortran/fortran_proposals` repository to solicit suggestions and feedback directly from the community
  - Place to publicly suggest and discuss proposals for the standards committee
  - Committee members post meeting updates to the repository
  - Lower communication barrier to the standards committee



Open a New Issue

Propose your additions or modifications to the Fortran standard.

Discuss

Community and committee members discuss your proposal publicly on Github

Draft Proposal

Collaboratively draft a formal proposal for the committee on Github

  - 90 contributors to discussions
  - 8 proposals drafted

Formation of Fortran-Lang

- **August 2019** Conversations on Twitter between Ondřej Čertík, Milan Curcic, and Jacob Williams bring out common perceived shortcomings in the Fortran ecosystem
- **October 2019** Ondřej creates the `j3-fortran/fortran_proposals` repository to solicit suggestions and feedback directly from the community
- **November 2019**
  - Many proposals to the j3 repository are for more intrinsics
  - Wider scope for string handling, filesystem access, sorting and linear algebra
  - Milan proposes the standard library project in response to requests for new intrinsics
  - `stdlib` repo started in December 2019 in `fortran-lang/stdlib`

Formation of Fortran-Lang

- **August 2019** Conversations on Twitter between Ondřej Čertík, Milan Curcic, and Jacob Williams bring out common perceived shortcomings in the Fortran ecosystem
- **October 2019** Ondřej creates the `j3-fortran/fortran_proposals` repository to solicit suggestions and feedback directly from the community
- **November 2019** Milan proposes the standard library project in response to requests for new intrinsics—`fortran-lang/stdlib` repo started in December 2019.
- **December 2019** Discussion in `j3-fortran/fortran_proposals` on the need for a dedicated Fortran package manager and build system

  Motivation & Aims

  - Improve ease-of-use for new users to compile Fortran projects
  - Remove barrier to depending on multiple Fortran libraries
  - Create an interoperable ecosystem of Fortran libraries
  - Support all common compilers with a common front-end

Formation of Fortran-Lang

- **August 2019** Conversations on Twitter between Ondřej Čertík, Milan Curcic, and Jacob Williams bring out common perceived shortcomings in the Fortran ecosystem
- **October 2019** Ondřej creates the `j3-fortran/fortran_proposals` repository to solicit suggestions and feedback directly from the community
- **November 2019** Milan proposes the standard library project in response to requests for new intrinsics—`fortran-lang/stdlib` repo started in December 2019.
- **December 2019** Discussion on the need for a dedicated Fortran package manager and build system—`fpm` repo started in January 2020.

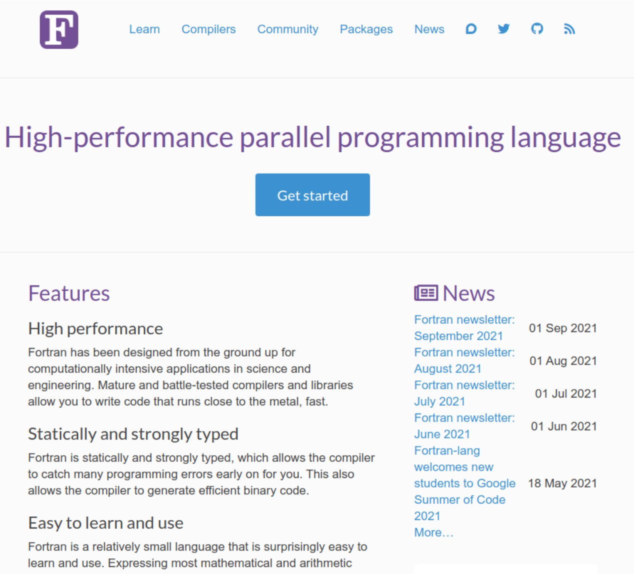
- **April 2020** Milan launches a new modern central website for Fortran: `https://fortran-lang.org`
- **May 2020** Applied for a free Discourse instance: `https://fortran-lang.discourse.group`

Fortran-Lang—Modern Web Presence

`https://fortran-lang.org`

Central community-maintained website

- List of compilers and community projects
- Tutorials and learning resources
- Monthly newsletter
- List of Fortran libraries and programs
- #2 on Google and #1 on most other search engines for "Fortran" queries



High-performance parallel programming language

Features

High performance

Fortran has been designed from the ground up for computationally intensive applications in science and engineering. Mature and battle-tested compilers and libraries allow you to write code that runs close to the metal, fast.

Statically and strongly typed

Fortran is statically and strongly typed, which allows the compiler to catch many programming errors early on for you. This also allows the compiler to generate efficient binary code.

Easy to learn and use

Fortran is a relatively small language that is surprisingly easy to learn and use. Expressing most mathematical and arithmetic

News

Fortran newsletter: September 2021	01 Sep 2021
Fortran newsletter: August 2021	01 Aug 2021
Fortran newsletter: July 2021	01 Jul 2021
Fortran newsletter: June 2021	01 Jun 2021
Fortran-lang welcomes new students to Google Summer of Code 2021	18 May 2021

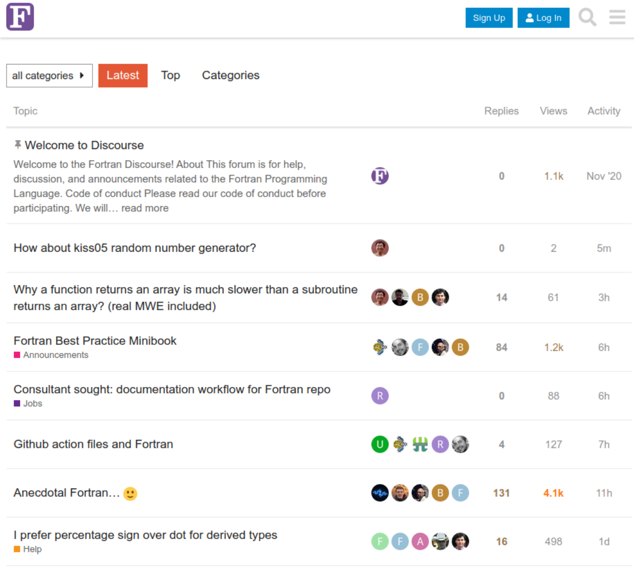
More…

Fortran-Lang—Friendly Discussions

`https://fortran-lang.discourse.group`

Modern friendly Forum

- Friendly and welcoming to all abilities
- Markdown formatting and attachments
- Moderated for respectful discussion
- 450 registered users
- 100,000 page views per month

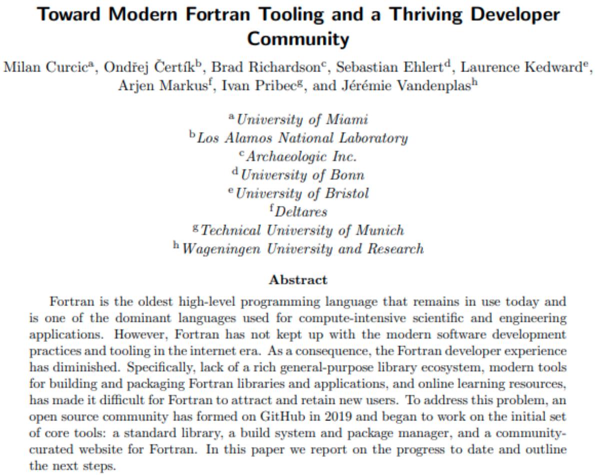


Topic	Replies	Views	Activity
Welcome to Discourse Welcome to the Fortran Discourse! About This forum is for help, discussion, and announcements related to the Fortran Programming Language. Code of conduct Please read our code of conduct before participating. We will… read more	0	1.1k	Nov '20
How about kiss05 random number generator?	0	2	5m
Why a function returns an array is much slower than a subroutine returns an array? (real MWE included)	14	61	3h
Fortran Best Practice Minibook Announcements	84	1.2k	6h
Consultant sought: documentation workflow for Fortran repo Jobs	0	88	6h
Github action files and Fortran	4	127	7h
Anecdotal Fortran… 🙂	131	4.1k	11h
I prefer percentage sign over dot for derived types Help	16	498	1d

Fortran-Lang—Find Out More

Read more about Fortran-Lang at
`https://arxiv.org/abs/2109.07382`

- Submitted to ACM Fortran Forum
- “State of Fortran” paper in revision for CiSE and coming soon!



Toward Modern Fortran Tooling and a Thriving Developer Community

Milan Curcic[a], Ondřej Čertík[b], Brad Richardson[c], Sebastian Ehlert[d], Laurence Kedward[e], Arjen Markus[f], Ivan Pribec[g], and Jérémie Vandenplas[h]

[a] *University of Miami*
[b] *Los Alamos National Laboratory*
[c] *Archaeologic Inc.*
[d] *University of Bonn*
[e] *University of Bristol*
[f] *Deltares*
[g] *Technical University of Munich*
[h] *Wageningen University and Research*

Abstract

Fortran is the oldest high-level programming language that remains in use today and is one of the dominant languages used for compute-intensive scientific and engineering applications. However, Fortran has not kept up with the modern software development practices and tooling in the internet era. As a consequence, the Fortran developer experience has diminished. Specifically, lack of a rich general-purpose library ecosystem, modern tools for building and packaging Fortran libraries and applications, and online learning resources, has made it difficult for Fortran to attract and retain new users. To address this problem, an open source community has formed on GitHub in 2019 and began to work on the initial set of core tools: a standard library, a build system and package manager, and a community-curated website for Fortran. In this paper we report on the progress to date and outline the next steps.

Part 3: The Future is Bright

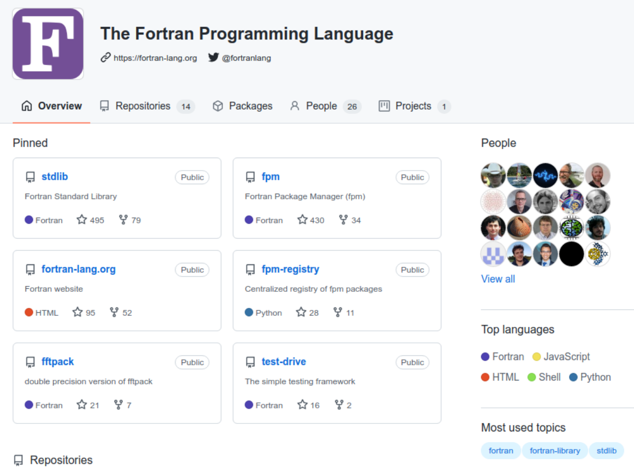
New modern tooling and compilers for Fortran

Fortran-Lang—Open Source Code Development

`https://github.com/fortran-lang`

14 Open Source Projects

- **Collaborative** code development with `git`
- **Transparency:** discussions, contributions and reviews open to everyone
- **Open source** under permissive licenses
- **Over 180 contributors** to code and discussions



The Fortran Programming Language

https://fortran-lang.org @fortranlang

Overview | Repositories 14 | Packages | People 26 | Projects 1

Pinned

stdlib Public
Fortran Standard Library
● Fortran ☆ 495 79

fpm Public
Fortran Package Manager (fpm)
● Fortran ☆ 430 34

fortran-lang.org Public
Fortran website
● HTML ☆ 95 52

fpm-registry Public
Centralized registry of fpm packages
● Python ☆ 28 11

fftpack Public
double precision version of fftpack
● Fortran ☆ 21 7

test-drive Public
The simple testing framework
● Fortran ☆ 16 2

Repositories

People

View all

Top languages

● Fortran ● JavaScript ● HTML ● Shell ● Python

Most used topics

fortran fortran-library stdlib

Fortran-Lang—Google Summer of Code 2021

Google Summer of Code (GSoC)

An international programme by Google to fund students for a 10-week open-source software project during the summer

- **February 2021** Fortran-Lang applied to GSoC as a new mentor organisation
- **March 2021** Fortran-Lang is accepted as a new organisation
- **May 2021** Fortran-Lang and LFortran are awarded 6 student slots for summer 2021
- **June-August 2021** GSoC students work on their projects

LFortran

- *Thirumalai Shaktivel*
- *Gagandeep Singh*
- *Rohit Goswami*

stdlib

- *Aman Godara*
- *Chetan Karwa*

fpm

- *Jakub Jelinek*

GSoC Icon—By Aswinshenoy (Own work), CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=78650424`

More About LFortran Today

- **LFortran**
  - *15:20 UTC* **Finish AST generation in LFortran** —*Thirumalai Shaktivel*
  - *15:25 UTC* **Supporting Arrays and Allocatables in LFortran** —*Gagandeep Singh*
  - *15:30 UTC* **Implementing Fortran Standardese within LFortran** —*Rohit Goswami*
  - *15:35 UTC* **Discussion**
- **`stdlib`**
- **fpm**

Fortran Standard Library (stdlib) Project

Aim

To **develop** and **provide** a **community driven** and agreed-upon de facto standard library for Fortran

- **Open source:** MIT License
- **Github:** `https://github.com/fortran-lang/stdlib`
- **API:** `https://stdlib.fortran-lang.org`

Scope

- **Algorithms**—*Sorting*
- **Programming**—*Strings, containers, file io, testing, logging*
- **Mathematics**—*Linear algebra, statistics, integration, root-finding, special functions*

More About stdlib Today

- LFortran
- **stdlib**
  - *15:40 UTC* **What's new in the Fortran Standard library** —*Nathaniel Shaffer*
  - *16:00 UTC* **Improving Strings Support in Fortran** —*Aman Godara*
  - *16:05 UTC* **Linked lists for stdlib** —*Arjen Markus on behalf of Chetan Karwa*
  - *16:10 UTC* **Discussion**
- fpm

Fortran Package Manager (*fpm*)

- **Open source:** MIT License
- **Github:** `https://github.com/fortran-lang/fpm`

Goal

A Fortran-specific build system and package manager to reduce the learning curve for starting new Fortran projects and depending on other Fortran libraries

Current Status

- *fpm* can scan module/submodule dependencies and build a wide variety of projects
- *fpm* supports Fortran and `C` sources, and incremental and parallel builds
- *fpm*-compatible libraries can easily be specified as project dependencies—*fpm* will automatically download and incorporate the dependency into the local project build
- **Over 170 fpm-compatible packages** available on Github and Gitlab

More About *fpm* Today

- LFortran
- `stdlib`
- **fpm**
    - *16:15 UTC* **Fortran package manager** —*Sebastian Ehlert*
    - *16:35 UTC* **Handling Compiler Flags in fpm (GSoC)** —*Jakub Jelinek*
    - *16:40 UTC* **Discussion**

Fortran-Lang Minisymposium

- **LFortran**
  - *15:20 UTC* **Finish AST generation in LFortran** —*Thirumalai Shaktivel*
  - *15:25 UTC* **Supporting Arrays and Allocatables in LFortran** —*Gagandeep Singh*
  - *15:30 UTC* **Implementing Fortran Standardese within LFortran** —*Rohit Goswami*
  - *15:35 UTC* **Discussion**
- **stdlib**
  - *15:40 UTC* **What's new in the Fortran Standard library** —*Nathaniel Shaffer*
  - *16:00 UTC* **Improving Strings Support in Fortran** —*Aman Godara*
  - *16:05 UTC* **Linked lists for stdlib** —*Arjen Markus on behalf of Chetan Karwa*
  - *16:10 UTC* **Discussion**
- **fpm**
  - *16:15 UTC* **Fortran package manager** —*Sebastian Ehlert*
  - *16:35 UTC* **Handling Compiler Flags in fpm (GSoC)** —*Jakub Jelinek*
  - *16:40 UTC* **Discussion**