



[Download](#) [GitHub](#) [Mailing List](#)  [Benchmark](#) [Features](#) [Documentation](#) [FAQ](#) [Links](#) [Feedback](#)


## Introduction

FFTW is a C subroutine library for computing the discrete Fourier transform (DFT) in one or more dimensions, of arbitrary input size, and of both real and complex data (as well as of even/odd data, i.e. the discrete cosine/sine transforms or DCT/DST). We believe that FFTW, which is [free software](#), should become the [FFT](#) library of choice for most applications.

The latest official release of FFTW is version **3.3.10**, available from [our download page](#). Version 3.3 introduced support for the AVX x86 extensions, a distributed-memory implementation on top of MPI, and a Fortran 2003 API. Version 3.3.1 introduced support for the ARM Neon extensions. See the [release notes](#) for more information.

The FFTW package was developed at [MIT](#) by [Matteo Frigo](#) and [Steven G. Johnson](#).

Our [benchmarks](#), performed on a variety of platforms, show that FFTW's performance is typically superior to that of other publicly available FFT software, and is even competitive with vendor-tuned codes. In contrast to vendor-tuned codes, however, FFTW's performance is *portable*: the same program will perform well on most architectures without modification. Hence the name, "FFTW," which stands for the somewhat whimsical title of "**Fastest Fourier Transform in the West**."

Subscribe to the [fftw-announce mailing list](#) to receive release announcements (or use the web feed ).

## Features

FFTW 3.3.10 is the latest official version of FFTW (refer to the [release notes](#) to find out what is new). Here is a list of some of FFTW's more interesting features:

- [Speed](#). (Supports SSE/SSE2/Altivec, since version 3.0. Version 3.3.1 supports AVX and ARM Neon.)
- Both one-dimensional and **multi-dimensional** transforms.
- **Arbitrary-size** transforms. (Sizes with small prime factors are best, but FFTW uses  $O(N \log N)$  algorithms even for prime sizes.)
- Fast transforms of **purely real** input or output data.
- Transforms of real even/odd data: the [discrete cosine transform](#) (DCT) and the [discrete sine transform](#) (DST), types I-IV. (Version 3.0 or later.)
- Efficient handling of **multiple, strided** transforms. (This lets you do things like transform multiple arrays at once, transform one dimension of a multi-dimensional array, or transform one field of a multi-component array.)
- [Parallel transforms](#): parallelized code for platforms with **SMP** machines with some flavor of [threads](#) (e.g. POSIX) or [OpenMP](#). An [MPI](#) version for distributed-memory transforms is also available in FFTW 3.3.
- **Portable** to any platform with a C compiler.
- [Documentation](#) in HTML and other formats.
- Both **C** and **Fortran** interfaces.

- [Free](#) software, released under the GNU General Public License (GPL, see [FFTW license](#)). (Non-free licenses may also be [purchased from MIT](#), for users who do not want their programs protected by the GPL. [Contact us](#) for details.) (See also the [FAQ](#).)

If you are still using **FFTW 2.x**, please note that FFTW 2.x was last updated in 1999 and it is obsolete. Please upgrade to FFTW 3.x. The API of FFTW 3.x is **incompatible** with that of FFTW 2.x, for reasons of performance and generality (see the [FAQ](#) or the [manual](#)).

## Documentation

First, read the [FFTW Frequently Asked Questions](#) document.

Manual: [HTML](#) or [PDF](#).

man pages: the [fftw-wisdom](#) and [fftw-wisdom-to-conf](#) utilities.

For general questions about Fourier transforms, see our [links to FFT-related resources](#). People often ask us how to compute a subset of the FFT outputs, so we have posted a short [discussion of pruned FFTs](#).

We benchmarked FFTW against many other FFT programs, in one to three dimensions, on a variety of platforms. You can view the results from this benchmark, or download it to run on your own machine and compiler, at the [benchFFT web page](#).

An audio [interview of the FFTW authors](#) is available from the RCE podcast program.

## Downloading

Versions 3.3.10 and 2.1.5 of FFTW may be [downloaded from this site](#). Feel free to post FFTW on your own site, but be sure to tell us so that we can link to your page and notify you of updates to the software.

## Literature.

- [BibTeX file](#) of FFTW references.
- The most current general paper about FFTW, and the preferred FFTW reference: Matteo Frigo and Steven G. Johnson, "[The Design and Implementation of FFTW3](#)," *Proceedings of the IEEE* **93** (2), 216–231 (2005). Invited paper, Special Issue on Program Generation, Optimization, and Platform Adaptation. [Link is to our preprint of published article; also in [Postscript](#). Official issue is [here](#).]
- [Implementing FFTs in Practice](#), our chapter in the online book *Fast Fourier Transforms* edited by C. S. Burrus.
- "[A Fast Fourier Transform Compiler](#)," by Matteo Frigo, in the Proceedings of the 1999 ACM SIGPLAN Conference on Programming Language Design and Implementation ([PLDI '99](#)), Atlanta, Georgia, May 1999. This paper describes the guts of the FFTW codelet generator. (Also in [Postscript](#). The [slides](#) from the talk are also available.)
- An earlier (and somewhat out of date) paper on FFTW was published in the 1998 ICASSP conference proceedings (vol. 3, pp. 1381–1384) with the title "[FFTW: An Adaptive Software Architecture for the FFT](#)" (also in [Postscript](#)), by M. Frigo and S. G. Johnson.
- An even older technical report is "[The Fastest Fourier Transform in the West](#)," MIT-LCS-TR-728 (September 1997) (also in [Postscript](#)).
- You might also be interested in "[Cache-Oblivious Algorithms](#)," by M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran (FOCS '99).
- The [slides](#) from the 7/28/98 talk "The Fastest Fourier Transform in the West," by M. Frigo, are also available, along with the [slides](#) from a shorter 1/14/98 talk on the same subject by S. G. Johnson.
- A paper on a new FFT algorithm that, following [James Van Buskirk](#), improves upon previous records for the arithmetic complexity of the DFT and related transforms, is: Steven G. Johnson and Matteo Frigo, "[A modified split-radix FFT with fewer arithmetic operations](#)", *IEEE Trans. Signal Processing* **55** (1), 111–119 (2007). Two preprints describing the application of the new algorithm to discrete cosine transforms are "[Type-II/III DCT/DST algorithms with reduced](#)

[number of arithmetic operations](#)" (March 2007) and "[Type-IV DCT, DST, and MDCT algorithms with reduced numbers of arithmetic operations](#)" (August 2007), by X. Shao and S. G. Johnson.

## Awards

FFTW received the [1999 J. H. Wilkinson Prize for Numerical Software](#), which is awarded every four years to the software that "best addresses all phases of the preparation of high quality numerical software." Wilkinson was a seminal figure in modern numerical analysis as well as a key proponent of the notion of reusable, common libraries for scientific computing, and we are especially honored to receive this award in his memory.

Our paper "A Fast Fourier Transform Compiler" (in PLDI 1999) received the Most Influential PLDI Paper award in 2009.

## Acknowledgements

We are grateful for the support of many people and companies, including Sun, Intel, the GNU project, and the Linux community. Please see the [acknowledgements section](#) of our manual for a more complete list of those who helped us. We are especially thankful to all of our users for their continuing support, feedback, and interest in the development of FFTW.

## Related Links

We have put together a [list of links](#) to other interesting sites with FFT-related code or information. This should be helpful if you want to know more about Fourier transforms or if for some reason FFTW doesn't satisfy your needs.

## Feedback

If you have comments, questions, or suggestions regarding FFTW, don't hesitate to email us at [fftw@fftw.org](mailto:fftw@fftw.org). We support encrypted/signed email. Use [our public keys](#).