# Title: FFT Convolutions are Faster than Winograd on Modern CPUs, Here's Why

JX-Ma

2024/10/5

**ABSTRACT**

- winograd algorithm float operation less than FFT algorithm and direct convolution .
- this article mainly explain why FFT faster than winograd
- This article is based on the roofline model and analyzes the floating-point operation count, memory bandwidth, and cache size of two algorithms on the CPU.

**INTRODUCTION**

This article mainly compares three algorithms, one is the highly optimized Winograd-based convolution, the second is the conventional algorithm based on FFT, and the third is the Gaussian multiplication algorithm based on FFT. Finally, the results show that in some cases, the FFT-based algorithm performs better than Winograd.
Winograd–based approach, in most cases, requires fewer floating point operations than FFT–based approach because it works with real numbers instead of complex numbers. However, due to its numerical instability, the Winograd method can use only very small transform sizes. The FFT–based convolutions do not suffer from such instabilities, allowing for arbitrary large tile sizes. Large tile sizes allow the FFT–based approach to reduce a large number of redundant or unnecessary computations. Thus, in certain scenarios, the FFT–based method requires fewer operations than the Winograd–based one.

**Background**

This section mainly introduces some background information on FFT-based convolution and Winograd convolution, as well as the perspective on convolution in multiple dimensions.

By using the Gauss multiplication FFT compared to the ordinary FFT, the number of operations has been reduced by 25%.

### Implementations

Winograd and FFT both perform calculations in four different stages: input transformation, kernel transformation, element-wise computation, and inverse transformation. This article discusses the optimizations used for these three algorithms, including software prefetching, memory and cache blocking, using aligned vector data access, and interleaving memory access with computation.

In the implementation of Winograd, Wincnn was used to create transformation matrices.

For the FFT–based implementations, the codelets were replaced by C++ codelets generated using "genfft" supplied with FFTW.

For the element–wise stage, where matrix–matrix multiplications are performed, the implementation of provides JIT routines for real– matrix multiplications optimized for AVX512 instruction set.

Implement parallelization of each stage of the algorithm through static scheduling.

### Performance Comparisons

Research implementation has been compared with LIBXSMM and MKL-DNN Winograd implementation as well as MKL-DNN direct convolution. Currently, apart from researchers' own implementation, there is no other publicly available CPU-based FFT method implementation.

Winograd's method has limitations in terms of numerical stability, especially when dealing with larger transform blocks. Most vendors implement the Winograd algorithm with a limit of block size of 6x6.

In AlexNet and VGG, the FFT method outperforms the Winograd method in 6 layers, while the performance of both methods is roughly the same in 3 layers. The advantage of the FFT method is more pronounced in certain layers, resulting in significant overall computational time savings.

The optimal transformation size of FFT is not always a power of 2, but depends on the specific structure of the network and layers

### Performance Analysis

The experimental observations suggested that some of the stages in both Winograd– and FFT–based approach have relatively low utilization of the system's available FLOPS. In most, but not all, cases, these were the transform stages, which have relatively small amount of compute, but access relatively large amount of data, suggesting that their running time is bound by the memory bandwidth, and not the computational capabilities of the CPU.

**Summary**

This article demonstrates that convolution based on FFT is typically faster on modern CPUs than convolution based on Winograd. Analysis based on the Roofline model indicates that the faster method between Winograd and FFT depends on the layer and target hardware. However, with the increasing trend in modern CPU system compute-to-memory ratio, FFT method often outperforms Winograd.