Implicitly Dealiased Convolutions for Distributed Memory Architectures

Malcolm Roberts[*,1], John C. Bowman[2]

[1]University of Strasbourg
[2]University of Alberta

SIAM Conference on Parallel Processing for Scientific Computing, 2016-04-15

[*]malcolm.i.w.roberts@gmail.com, www.malcolmiwroberts.com

Abstract

Implicitly Dealiased Convolutions for Distributed Memory Architectures

Convolutions are a fundamental tool of scientific computing. For multi-dimensional data, implicitly dealiased convolutions [Bowman and Roberts, SIAM J. Sci. Comput. 2011] are faster and require less memory than conventional FFT-based methods. We present a hybrid `OpenMP/MPI` implementation in the open-source software library `FFTW++`. The reduced memory footprint translates to a reduced communication cost, and the separation of input and work arrays allows one to overlap computation and communication.

Outline

- FFT-based convolutions
  - Conventional dealiasing
  - Implicit dealiasing
- Hybrid `OpenMP`/`MPI` parallelism
- 1/2 padded convolutions
  - Performance results
- 2/3 padded convolutions for pseudospectral simulations
  - The Nyquist frequency: compact/non-compact data
  - Performance results

FFT-based convolutions

The convolution of $\{F_k\}_{k=0}^{m-1}$ and $\{G_k\}_{k=0}^{m-1}$ is

$$(F * G)_k = \sum_{\ell=0}^{k-1} F_\ell G_{k-\ell}, \quad k=0,\dots,m-1. \tag{1}$$

Using FFTs improves speed and accuracy.

However, the indices are equivalent modulo m.

To recover a linear convolution, we must remove the aliased terms.

Conventional dealiasing

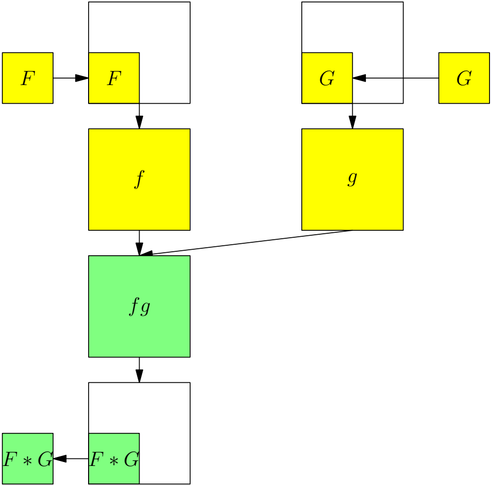
Conventionally, one pads the input with zeros:

$$\widetilde{F} \doteq \{F_0, F_1, \ldots, F_{m-2}, F_{m-1}, \underbrace{0, \ldots, 0}_{m}\} \tag{2}$$

So that

$$\begin{aligned} \left(\widetilde{F} *_{2N} \widetilde{G}\right)_k &= \sum_{\ell=0}^{2N-1} \widetilde{F}_{\ell \bmod 2N} \widetilde{G}_{(k-\ell) \bmod 2N} \\ &= \sum_{\ell=0}^{N-1} F_\ell \widetilde{G}_{(k-\ell) \bmod 2N} \\ &= \sum_{\ell=0}^{k} F_\ell G_{k-\ell}. \end{aligned} \tag{3}$$

Dealiasing with conventional zero-padding



Dealiasing with implicit zero-padding

We modify the FFT to account for the zeros implicitly.

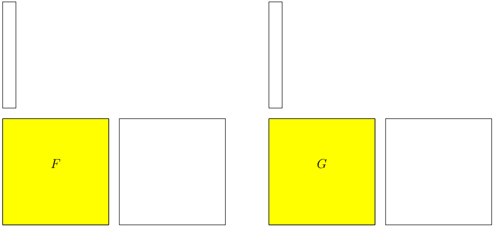
Let $\zeta_n = \exp(-i2\pi/n)$. The Fourier transform of $\widetilde{F}$ is

$$f_x = \sum_{k=0}^{2m-1} \zeta_{2m}^{xk} \widetilde{F}_k = \sum_{k=0}^{m-1} \zeta_{2m}^{xk} \widetilde{F}_k \tag{4}$$

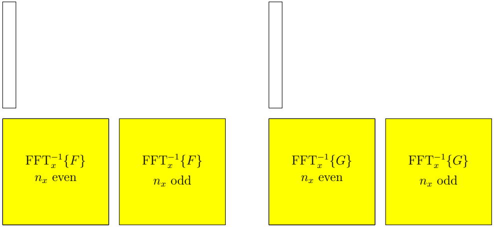
We can compute this using two discontiguous buffers:

$$f_{2x} = \sum_{k=0}^{m-1} \zeta_m^{xk} F_k \quad f_{2x+1} = \sum_{k=0}^{m-1} \zeta_m^{xk} \left(\zeta_{2m}^{k} F_k\right). \tag{5}$$

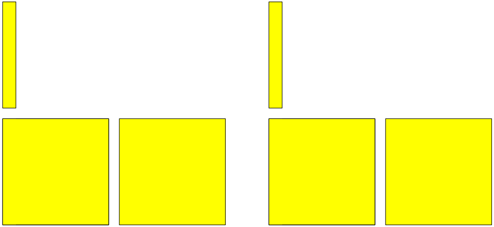
Dealiasing with implicit zero-padding



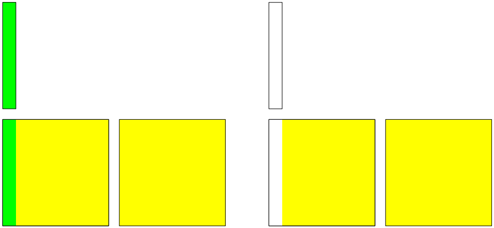
Dealiasing with implicit zero-padding



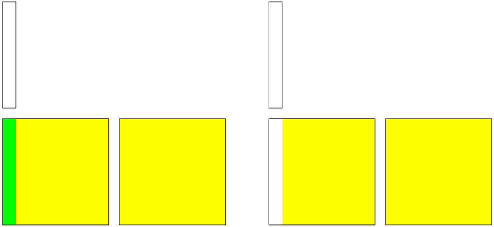
Dealiasing with implicit zero-padding



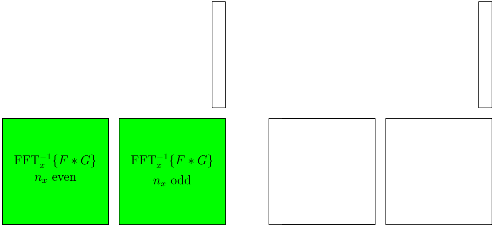
Dealiasing with implicit zero-padding



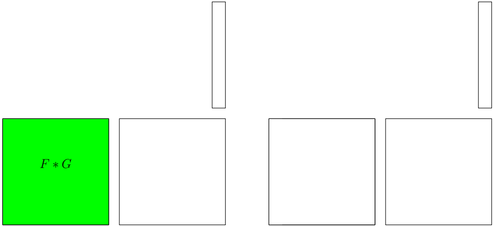
Dealiasing with implicit zero-padding



Dealiasing with implicit zero-padding



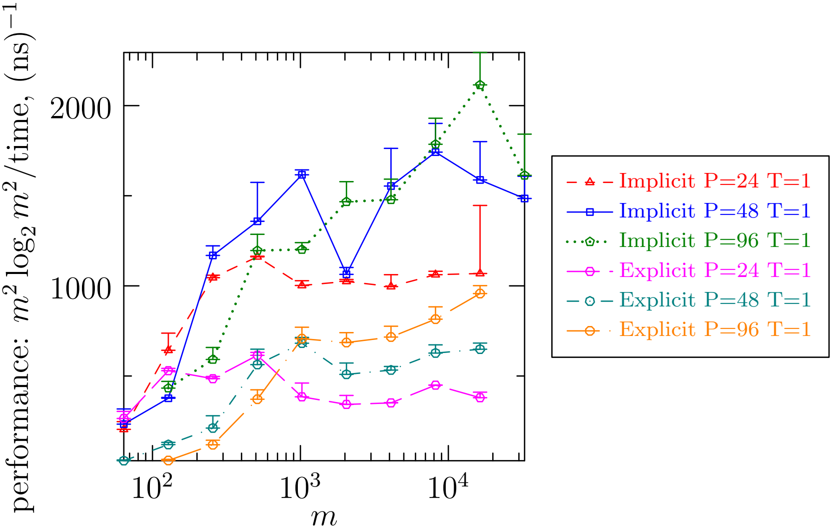
Dealiasing with implicit zero-padding



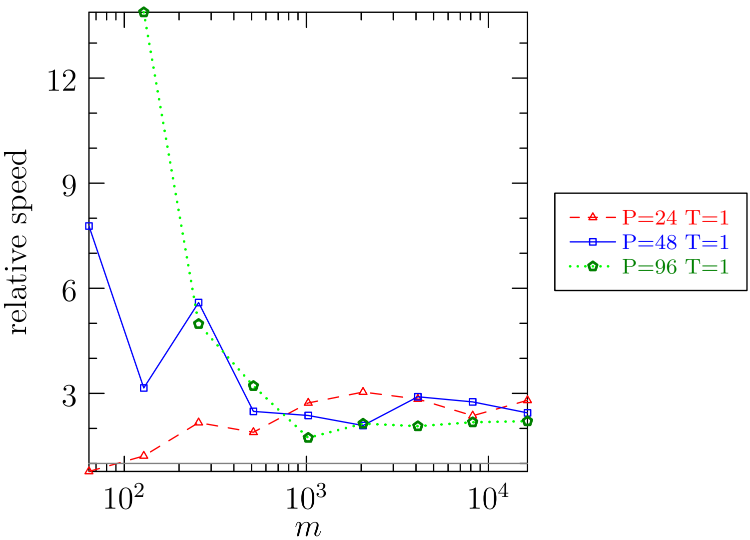
OpenMP/MPI implementation

- Implicit dealiasing requires less communication.
- We avoid FFTs on zero-data.
- By using discontiguous buffers, we can overlap communication and computation.
- We use a hybrid `OpenMP/MPI` parallelization for clusters of multi-core machines.
- 2D `MPI` data decomposition.
- `SSE2` vectorization instructions.
- We make use of the *hybrid transpose* algorithm.
  - See CP13 today at 15:20.

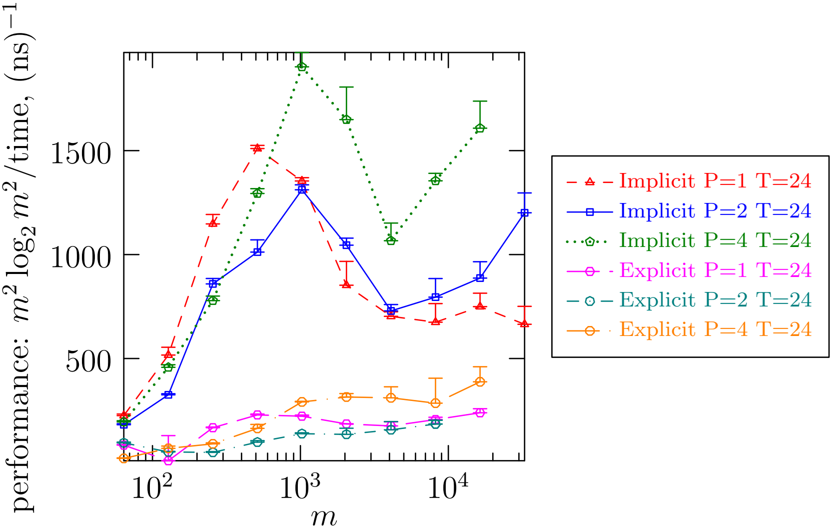
1/2 padding: 2D performance



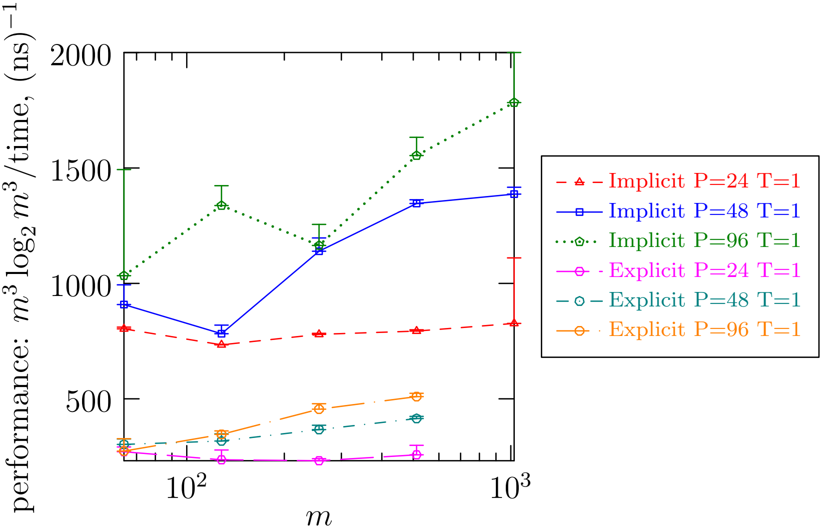
1/2 padding: 2D performance



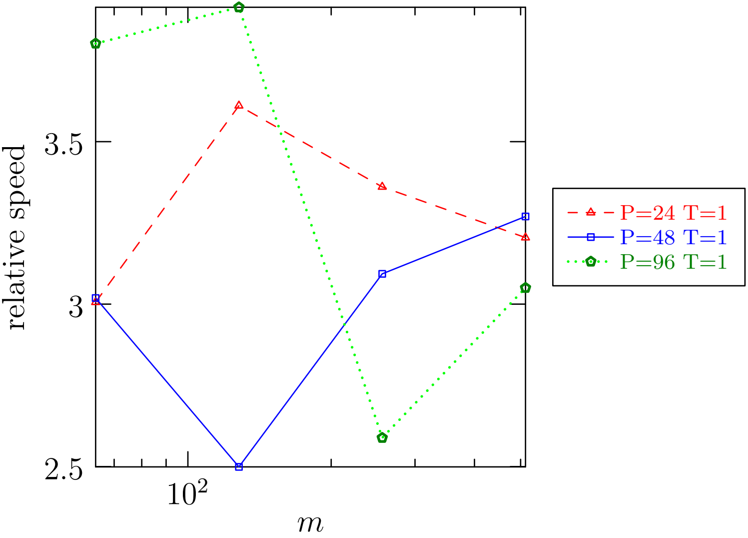
1/2 padding: multithreaded 2D performance



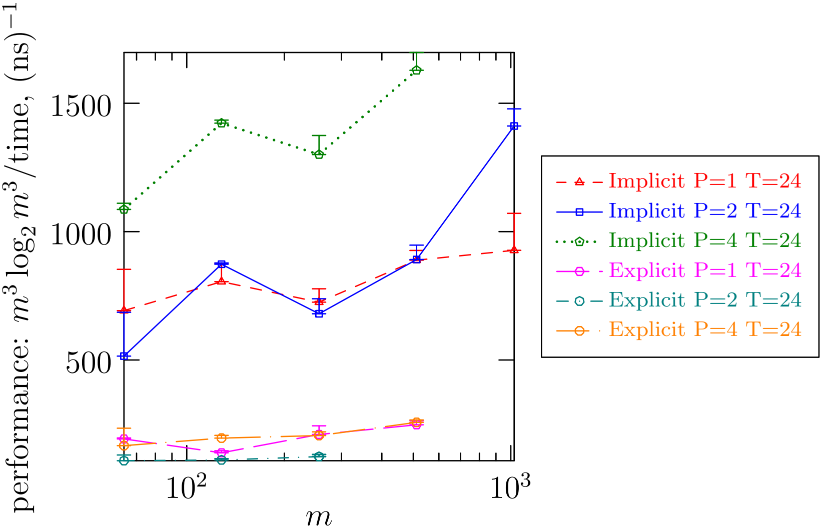
1/2 padding: 3D performance



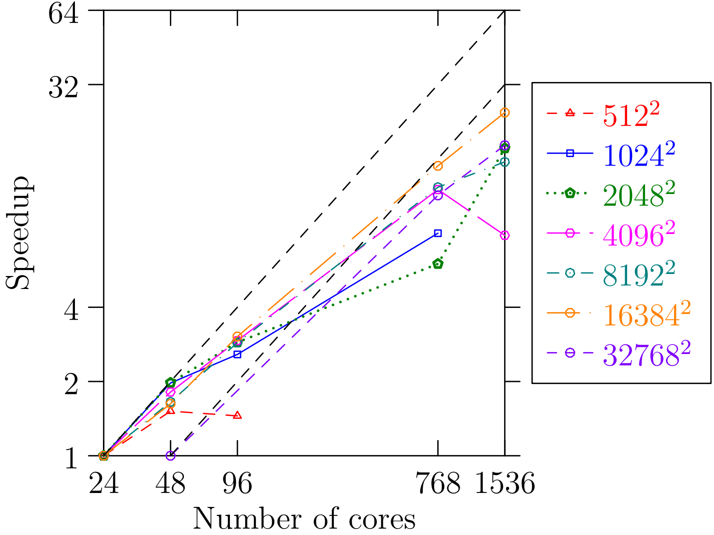
1/2 padding: 3D performance



1/2 padding: multithreaded 3D performance



1/2 padding: 3D scaling



2/3 padding

For pseudospectral simulations of PDEs the input is

$$F = \{F_k\}_{k=-m}^{m-1}. \tag{6}$$

A quadratic nonlinearity is transformed into a convolution:

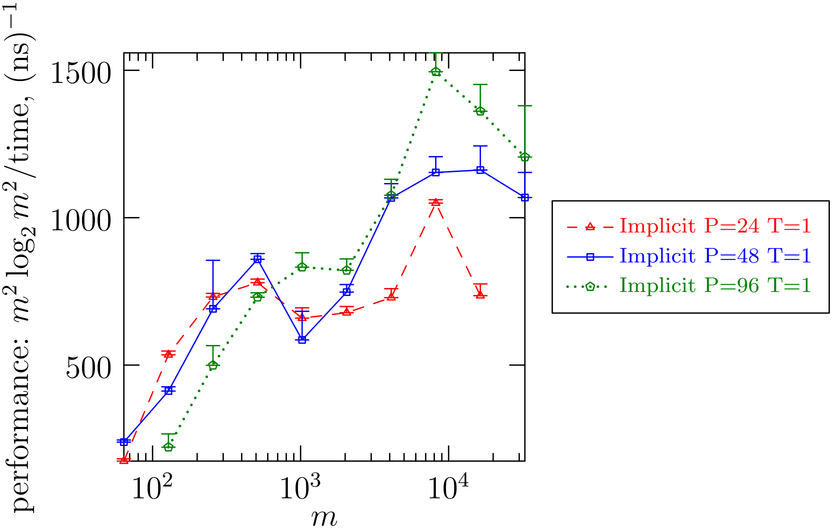
$$(F * G)_k = \sum_{\ell=k-m+1}^{m-1} F_\ell G_{k-\ell} \tag{7}$$

One must pad from length $2m$ to length $3m$ to remove aliases.

The implicitly dealiased convolution routines can either include (non-compact format) or exclude (compact format) the Nyquist mode F_{-m}.

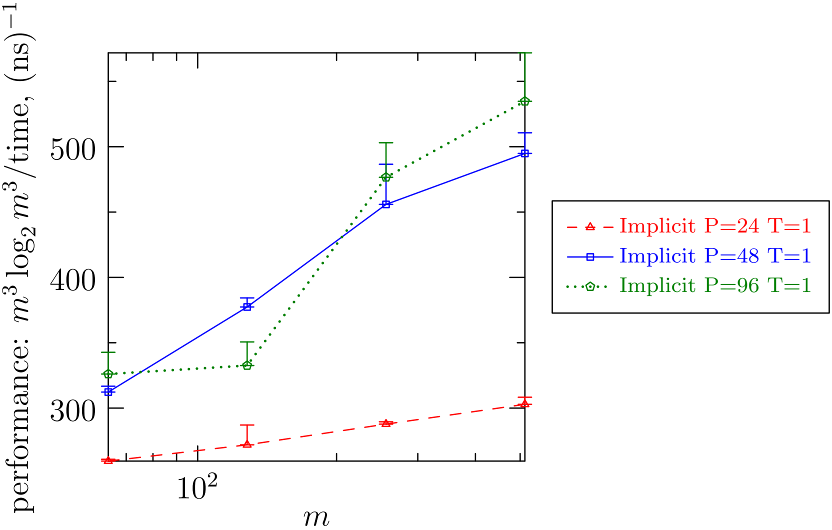
2/3 padding: 2D performance

Here we are non-compact in both directions:



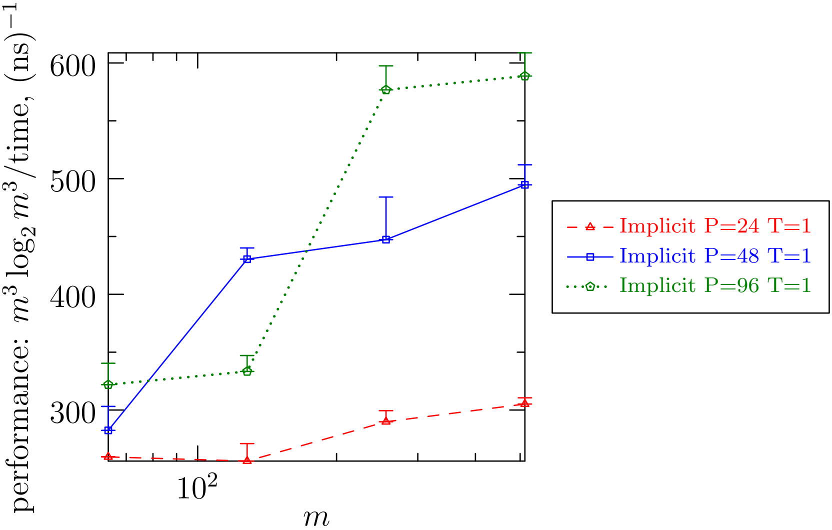
2/3 padding: 3D performance

Here we are non-compact in all directions:



2/3 padding: 3D performance

Here we are compact in the y-direction:



Conclusion

In this talk, I presented the distributed memory version of implicitly dealiased convolutions.

Implicitly dealiased convolutions:

- use less memory
- have less communication costs,
- and are faster than conventional zero-padding techniques.

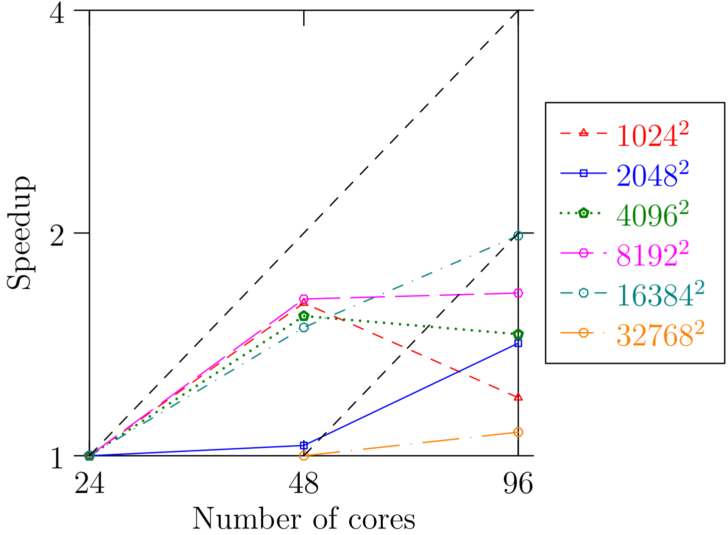
We make use of the hybrid transpose algorithm (CP13, 15:20).

Implementation in the open-source project `FFTW++`:

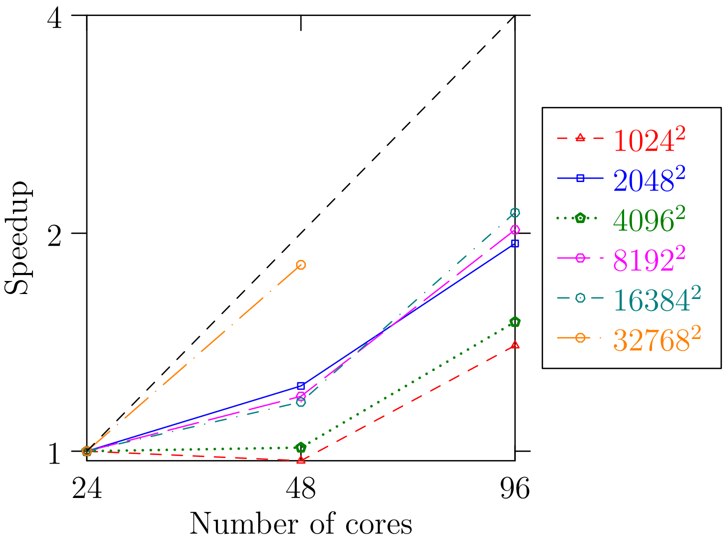
`fftwpp.sf.net`

Thank you for your attention!

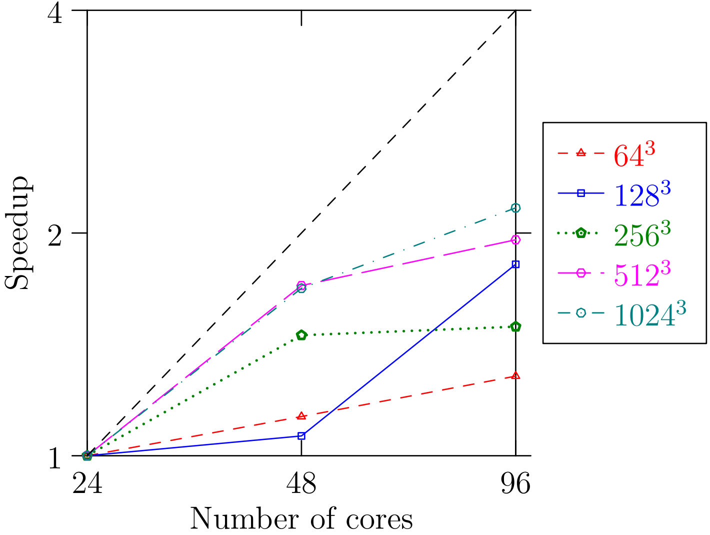
1/2 padding: 2D scaling



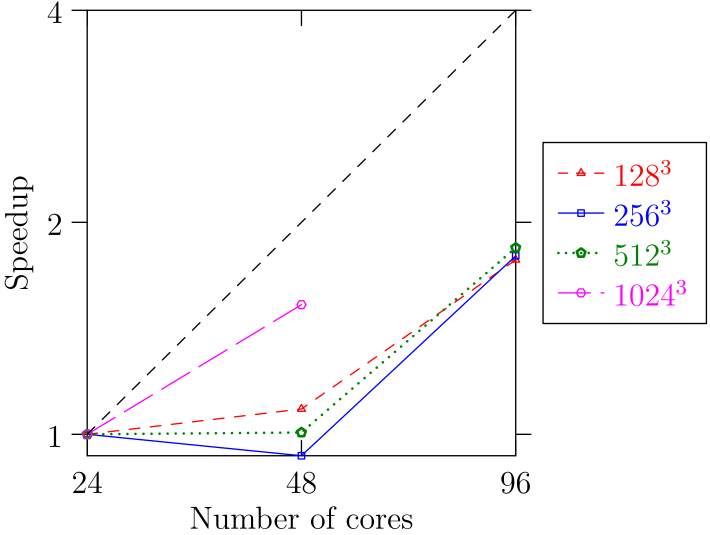
1/2 padding: multithreaded 2D scaling



1/2 padding: 3D scaling



1/2 padding: multithreaded 3D scaling



1/2 padding: explicit 3D scaling