Optimization of Hybrid Dealiasing of Convolutions

Robert Joseph George[1,2]

MATH 499 - Honors Thesis
Advised by: John Bowman[1], Noel Murasko[1]
Supervisor: Nicolas Guay[1]

[1]Department of Mathematics and Statistics
[2]Department of Computer Science

Contents

Motivation and Field of Study

1. Fast Fourier Transforms (FFTs) and convolutions are important in signal processing, image processing, computer vision, and applied mathematics.
2. FFTs efficiently compute a signal's discrete Fourier transform (DFT) to analyze the signal's frequency content.
3. FFTs can also be used for efficiently calculating convolutions.

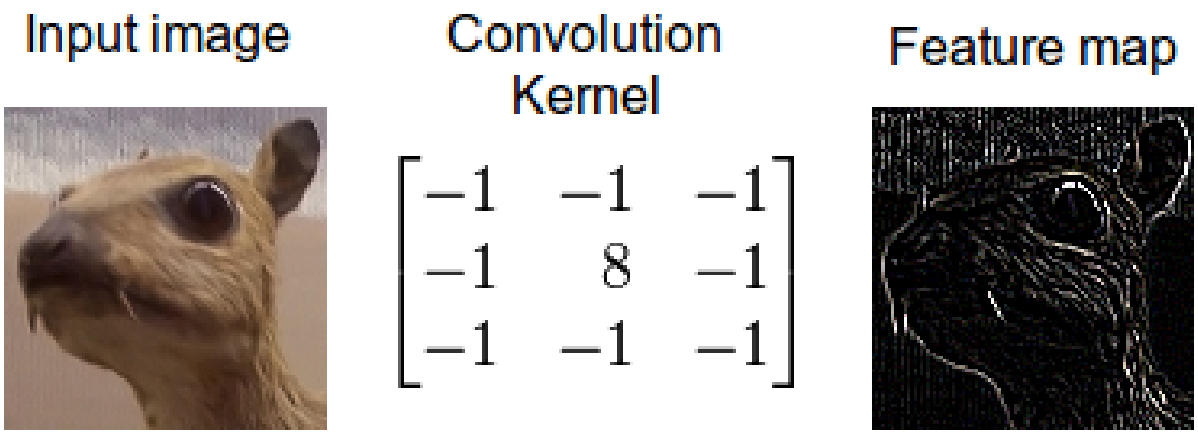


Figure 1: Convolution of 2 images

Discrete Fourier Transform

Consider an array $\{f_j\}_{j=0}^{M-1}$. The forward DFT of f can be written as

$$F_k = \sum_{j=0}^{M-1} \zeta_M^{kj} f_j, \ k = 0, \ldots, M-1$$

The corresponding backward DFT is given by

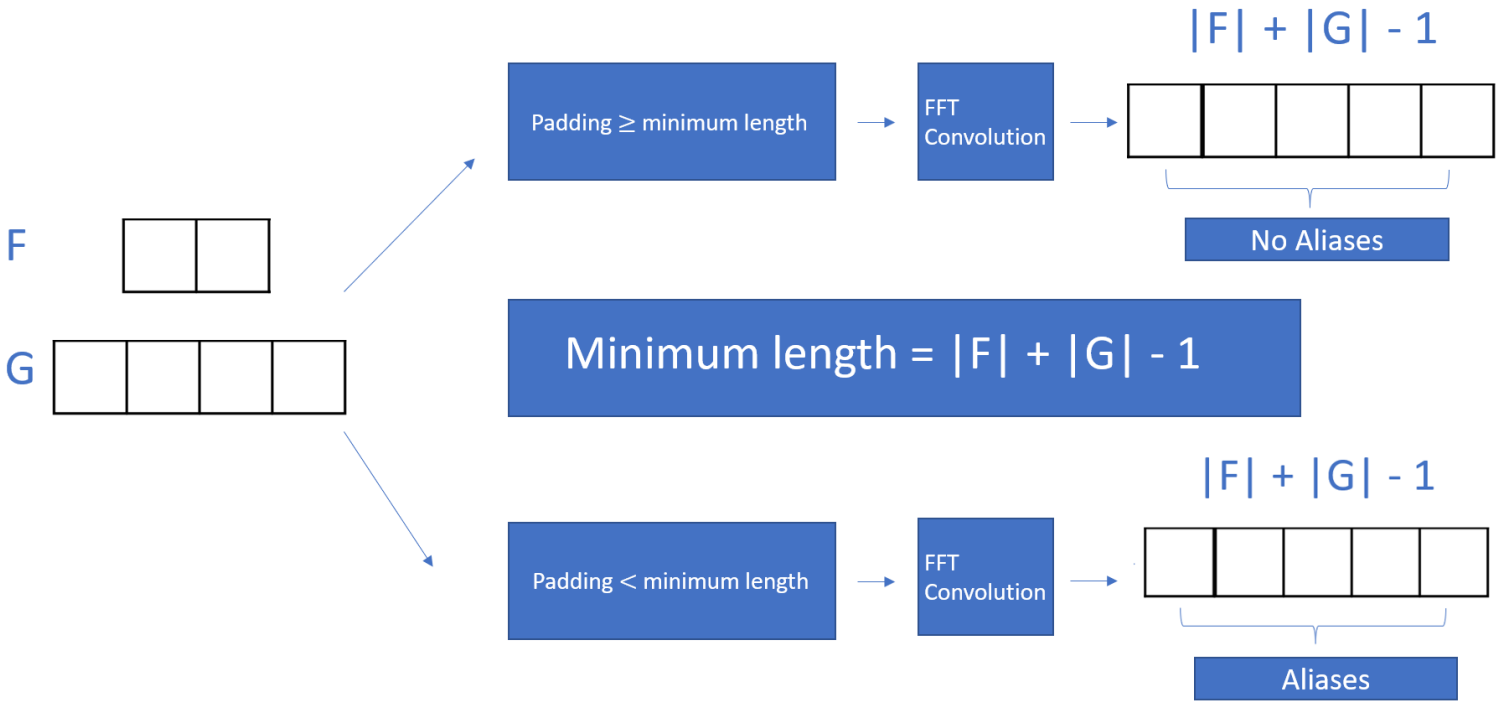
$$f_j = \frac{1}{M} \sum_{k=0}^{M-1} \zeta_M^{-kj} F_k, \ j = 0, \ldots, M-1$$

Convolutions

1. Consider the convolution of two sequences whose inputs are $\{f_j\}_{j=0}^{M-1}$ and $\{g_j\}_{j=0}^{M-1}$ and are of finite length M, yielding a linear convolution with components $\sum_{j=0}^{\ell} f_j g_{\ell-j}$ for $\ell = 0, \ldots, M-1$.
2. Computing such a convolution directly requires $\mathcal{O}\left(M^2\right)$ operations, and round off error is a problem for large M.
3. Preferable to use the convolution theorem, harnessing the FFT to map the convolution to component-wise multiplication and for inputs f_j and g_j.

Dealiasing

Linear convolutions require padding both input arrays with zeros.



Explicit Dealiasing

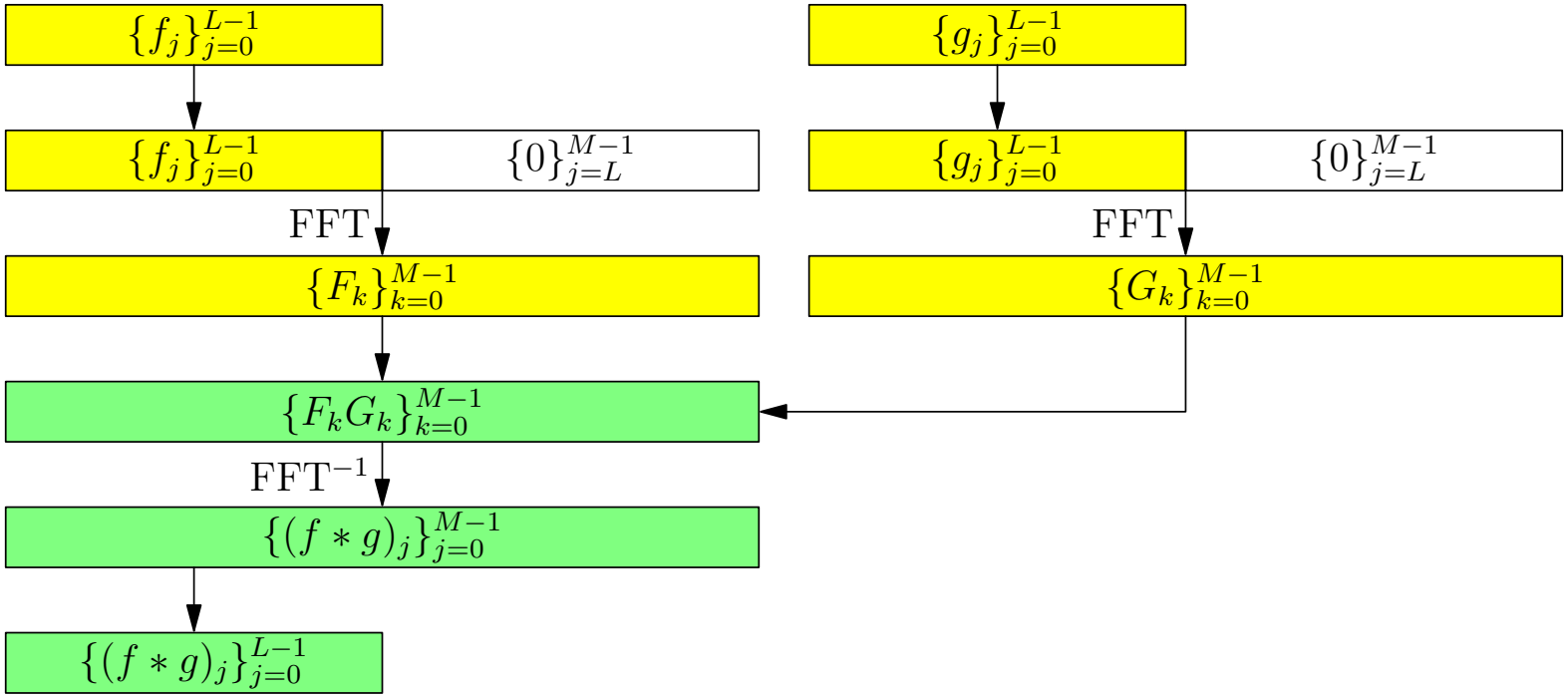


Figure 2: Convolution using explicit padding. Time complexity is $\mathcal{O}(M \log M)$

Implicit/Hybrid Dealiasing

1. **Implicit dealiasing** provides an alternative to explicit dealiasing, where FFTs are formulated to implicitly take account of known zero values, avoiding the need for explicit zero padding.
2. We provide an alternate approach to maximize performance, where we use **hybrid dealiasing:** the padding is chosen to be fully implicit, fully explicit, or a combination of the two.

Tuning Parameters

1. Given some $m \in \mathbb{N}$, we define $p = \left\lceil \frac{L}{m} \right\rceil$, $q = \left\lceil \frac{M}{m} \right\rceil$. These are the smallest positive integers such that $pm \geq L$ and $qm \geq M$.
2. 3 data-dependent parameters: L, M and the number C of FFTs to be computed simultaneously.
3. Other important parameters:
   - Size m of the FFTs.
   - The number D of residues computed at a time.
   - In-place ($I = 1$) or out-of-place ($I = 0$) FFTs.

1D Hybrid Padding

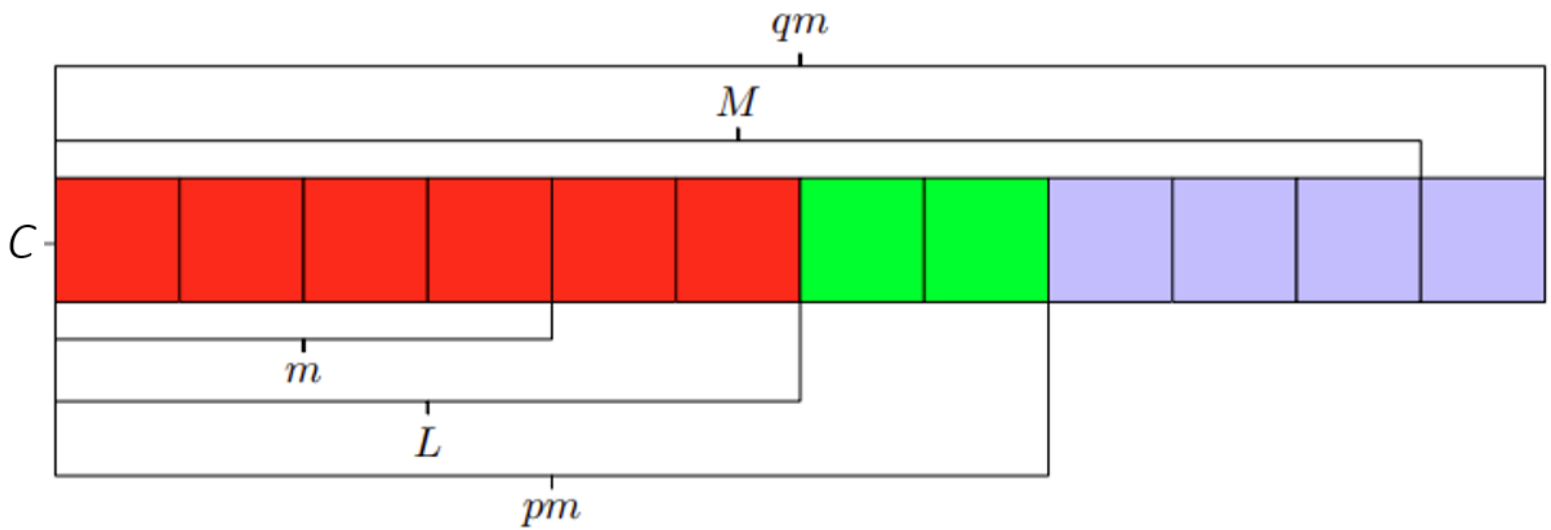
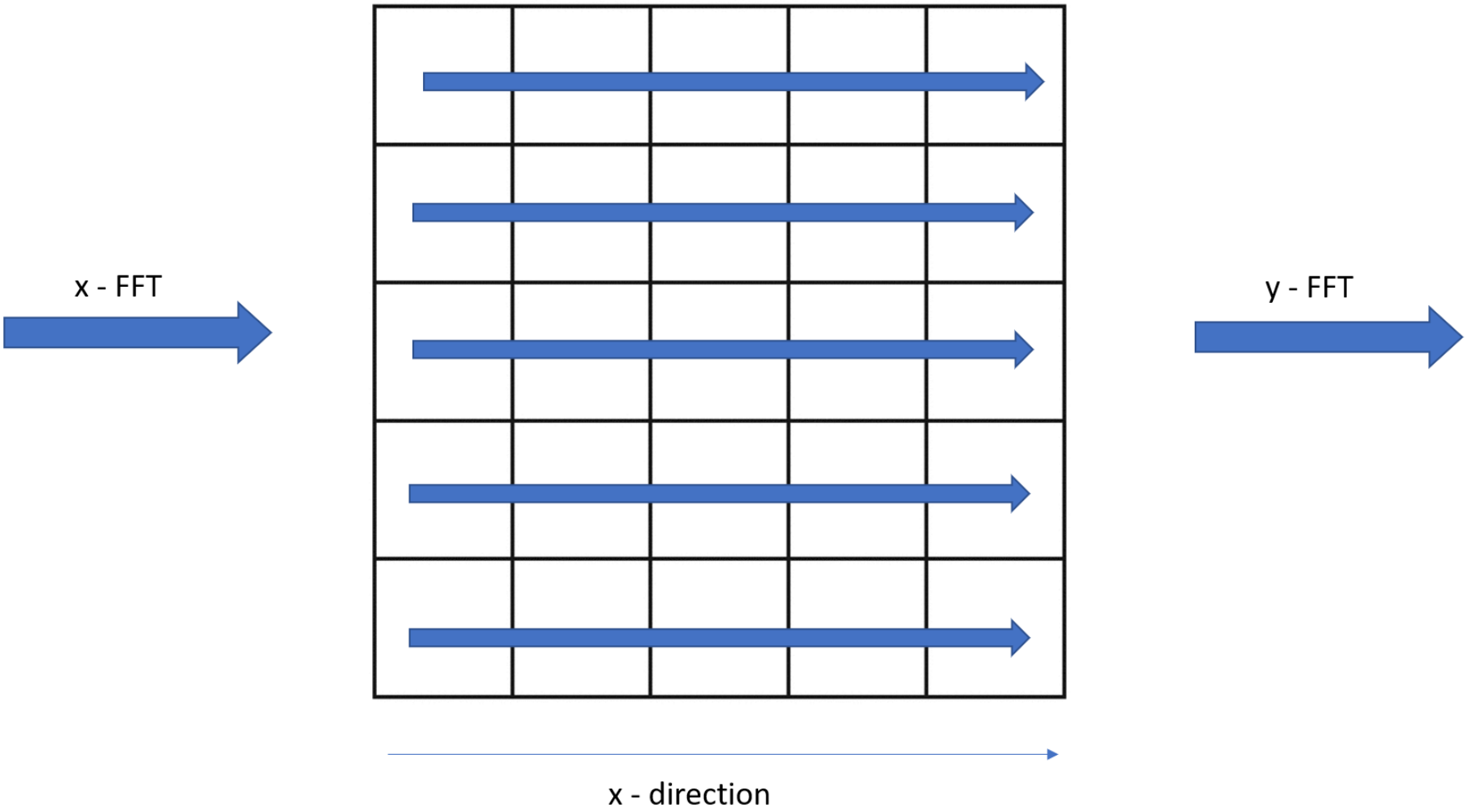
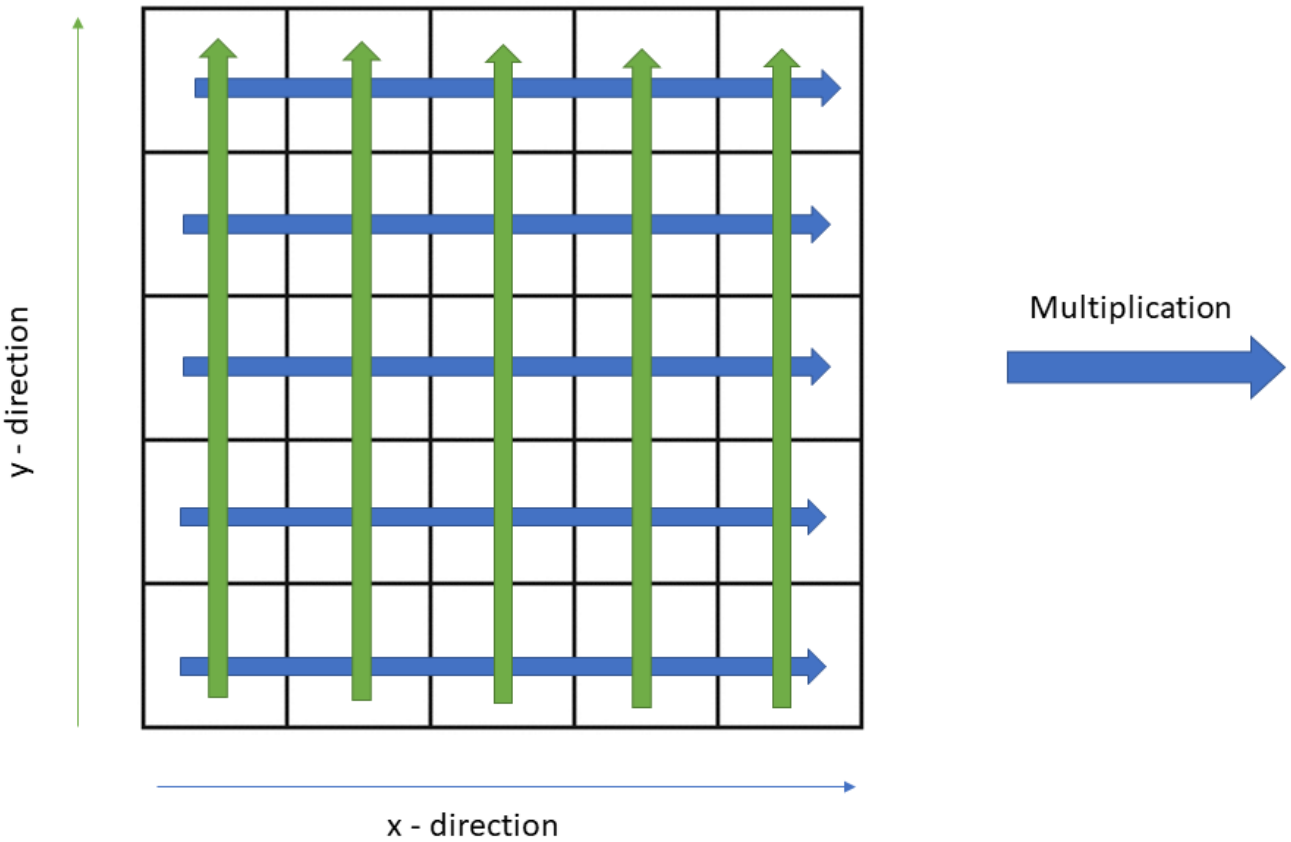


Figure 3: Tuning parameters for a 1D convolution with $L = 6$, $M = 11$, and $m = 4$, so that $p = 2$ and $q = 3$. We explicitly pad from $L = 6$ to $pm = 8$ and implicitly pad up to $qm = 12$.

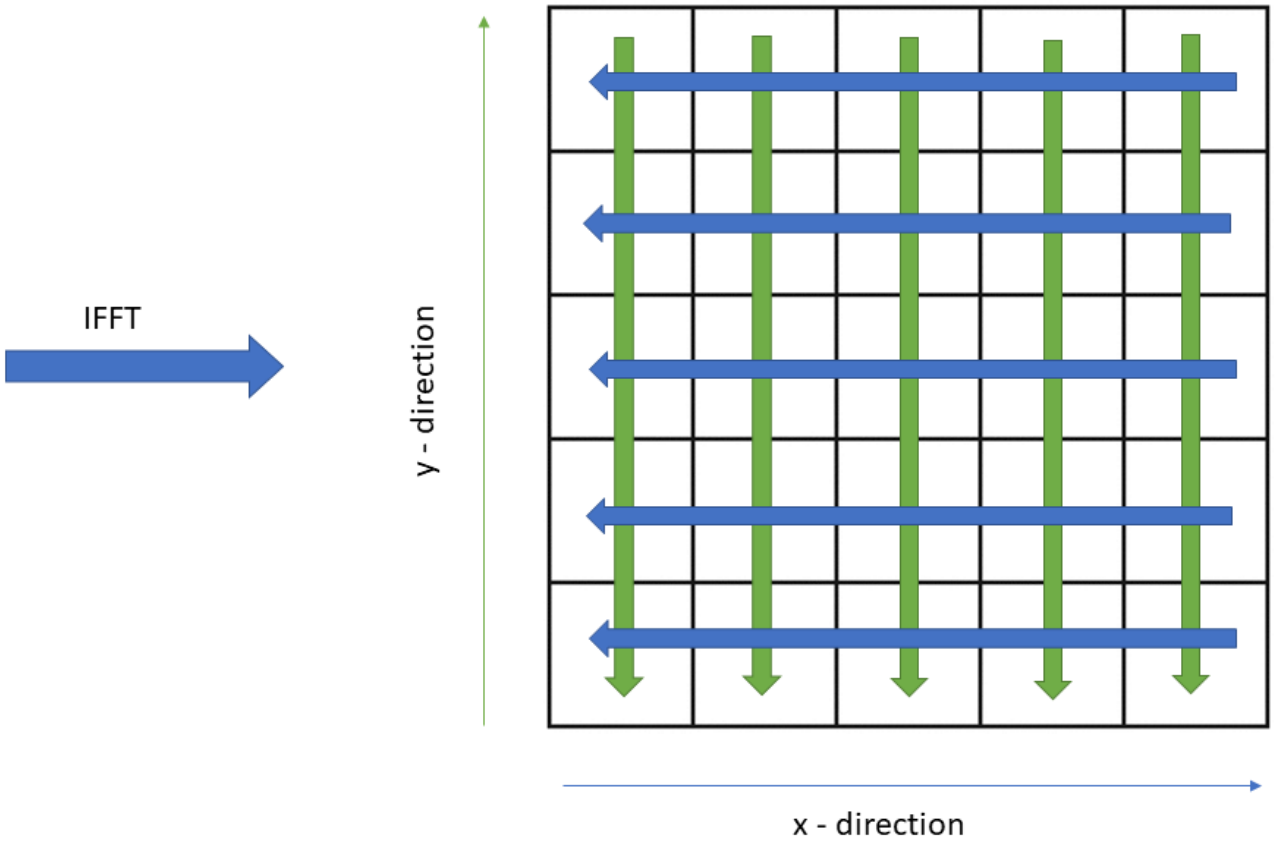
2D Convolution



2D Convolution

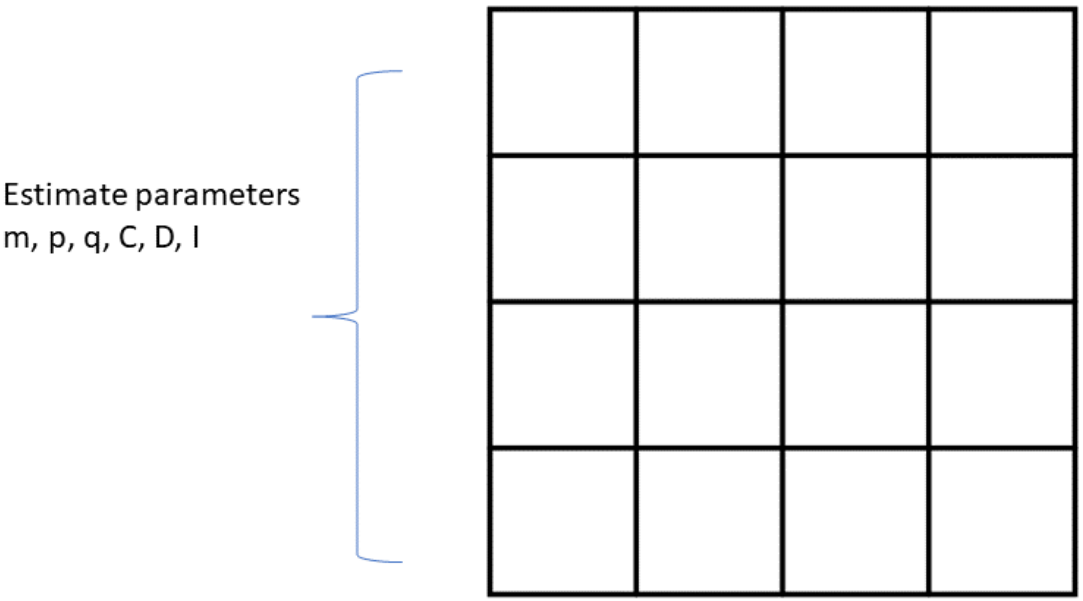


2D Convolution

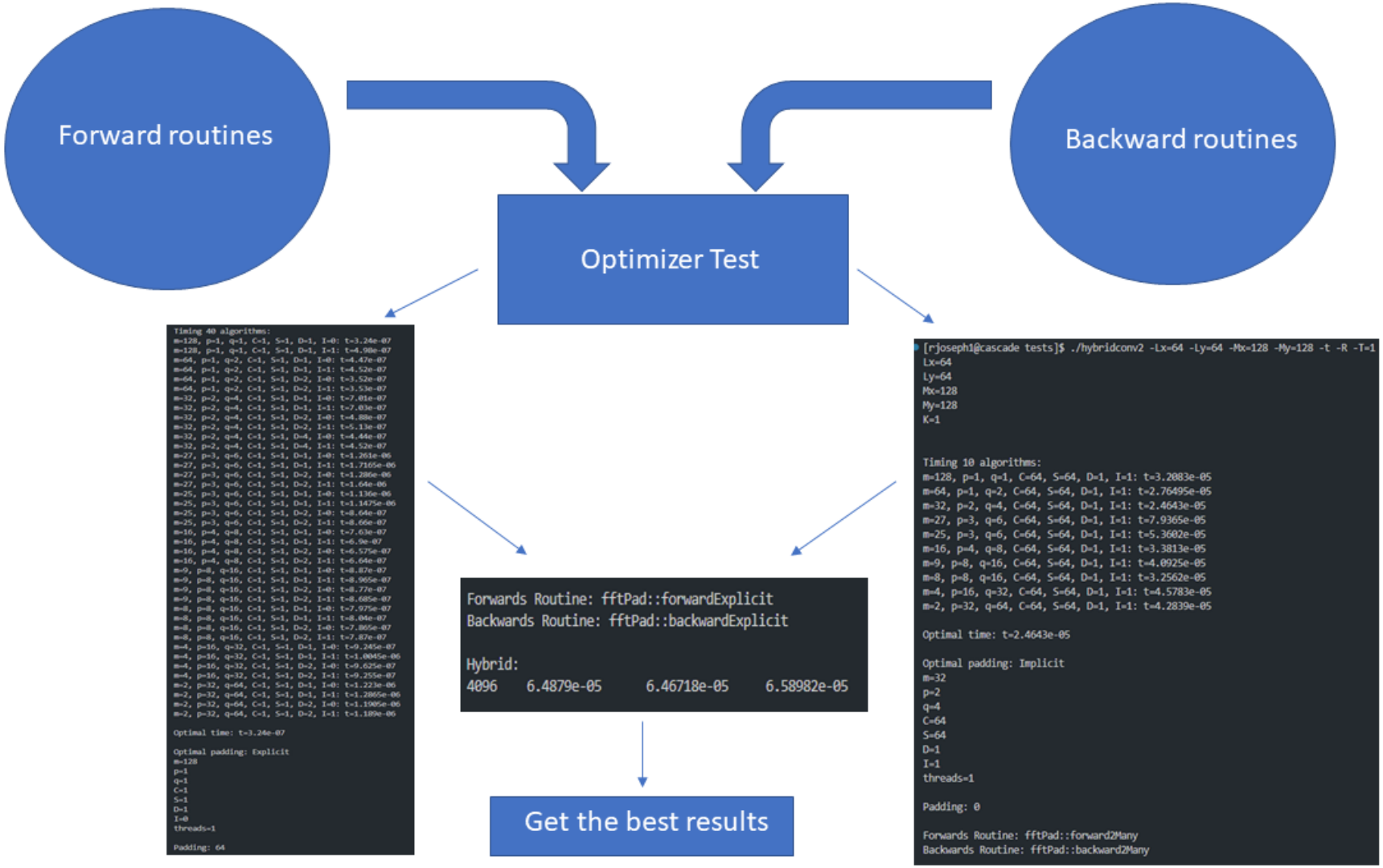


Optimizer Test

Given a square, we want to estimate the best parameters to do a 2D convolution. How do we currently do this?



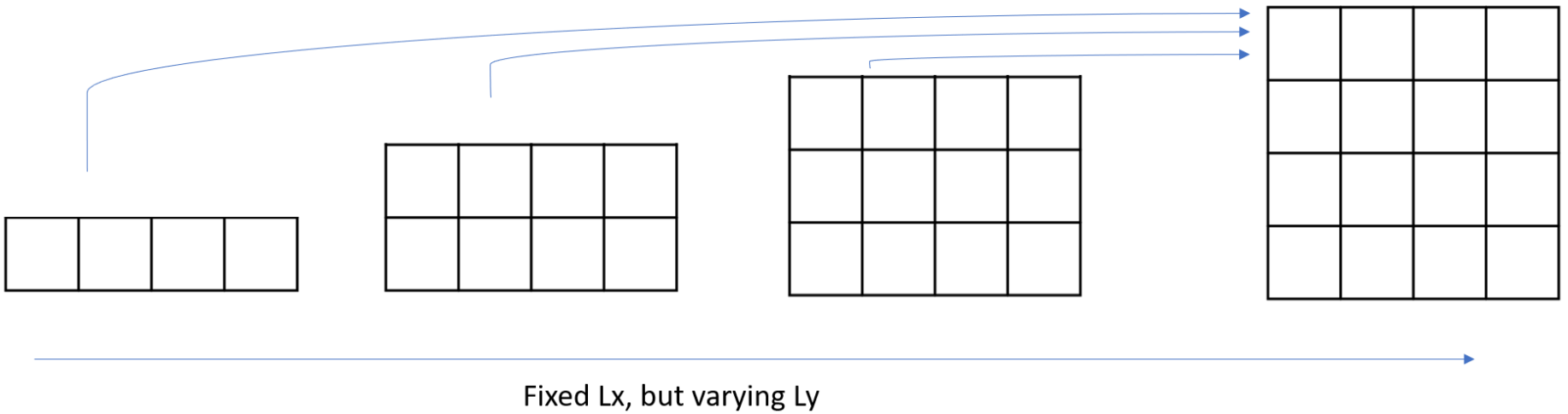
Final Optimizer Pass For Entire Convolution



Problem

Suppose that I am given a square or a large rectangle.

Which rectangle's parameters can be used to estimate the square's parameters?

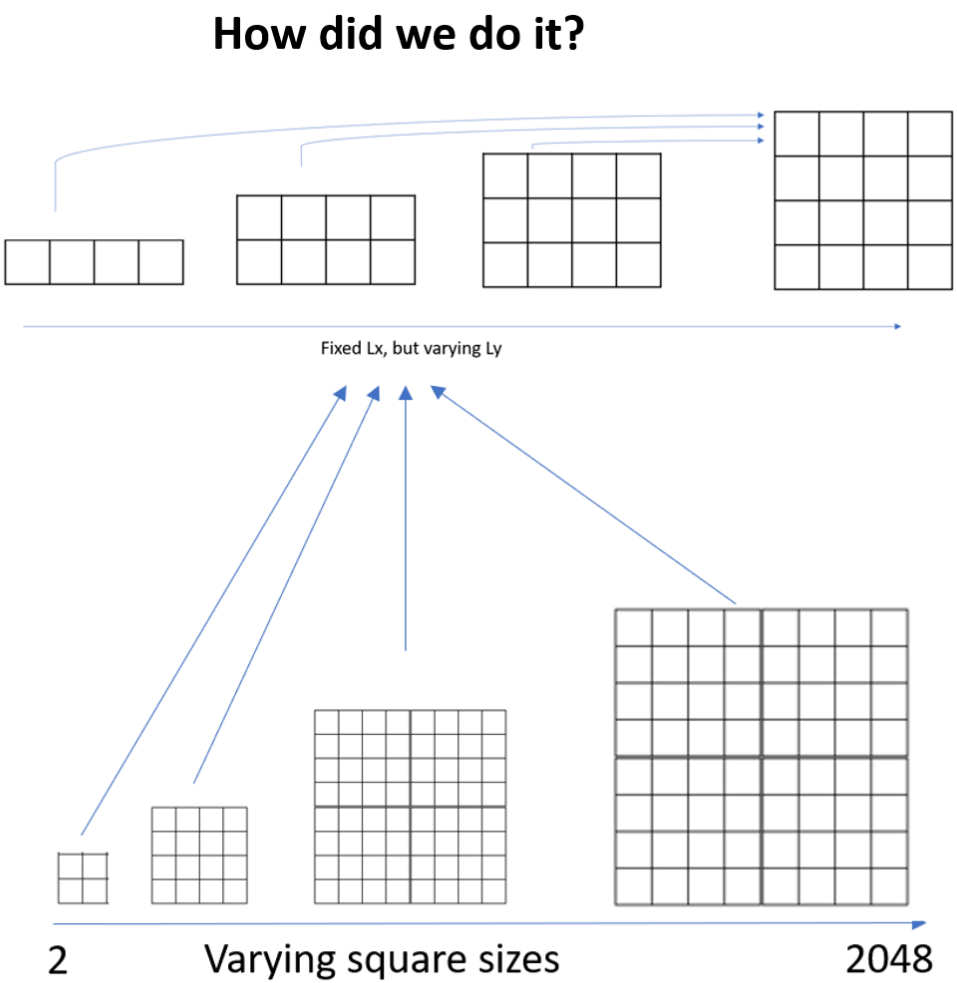


Reminder of Research Goals

1. Create a more optimized and efficient hyperparameter tuning algorithm which we call *experience* to find the optimal parameters.
2. This algorithm would search over a wider set of parameters to empirically determine the fastest algorithm for a given problem.
3. We also want to develop efficient heuristics.

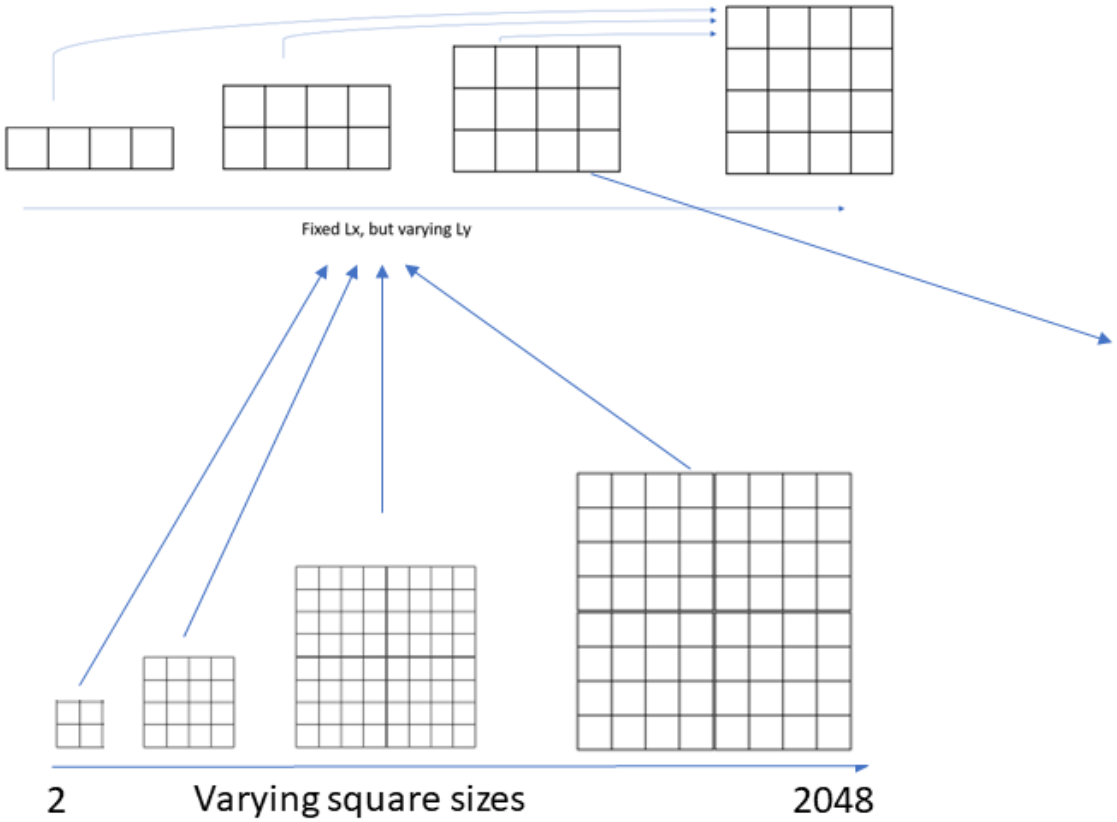
Data Generation

What is the data? How did we generate it? We only search over squares and rectangles of size $2^a 3^b 5^c 7^d$ where $a, b, c, d \in \mathbb{N}$.



Optimizer Test

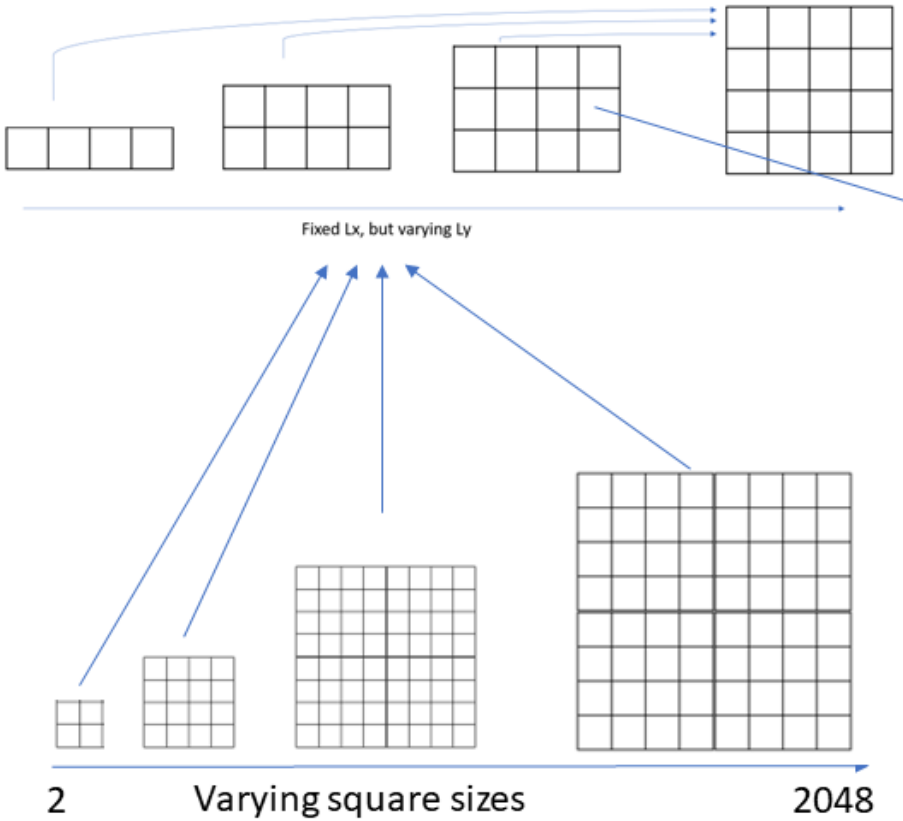
We use the optimizer to get the results.

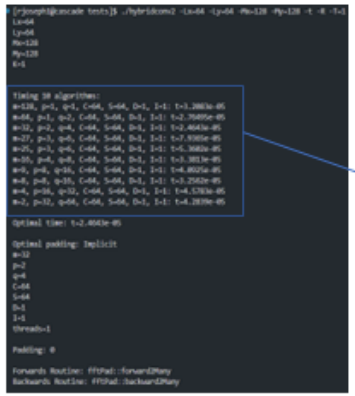



```
[rjoseph1@cascade tests]$ ./hybridconv2 -Lx=64 -Ly=64 -Mx=128 -My=128 -t -R -T=1
Lx=64
Ly=64
Mx=128
My=128
K=1


Timing 10 algorithms:
m=128, p=1, q=1, C=64, S=64, D=1, I=1: t=3.2083e-05
m=64, p=1, q=2, C=64, S=64, D=1, I=1: t=2.76495e-05
m=32, p=2, q=4, C=64, S=64, D=1, I=1: t=2.4643e-05
m=27, p=3, q=6, C=64, S=64, D=1, I=1: t=7.9365e-05
m=25, p=3, q=6, C=64, S=64, D=1, I=1: t=5.3602e-05
m=16, p=4, q=8, C=64, S=64, D=1, I=1: t=3.3813e-05
m=9, p=8, q=16, C=64, S=64, D=1, I=1: t=4.0925e-05
m=8, p=8, q=16, C=64, S=64, D=1, I=1: t=3.2562e-05
m=4, p=16, q=32, C=64, S=64, D=1, I=1: t=4.5783e-05
m=2, p=32, q=64, C=64, S=64, D=1, I=1: t=4.2839e-05

Optimal time: t=2.4643e-05

Optimal padding: Implicit
m=32
p=2
q=4
C=64
S=64
D=1
I=1
threads=1

Padding: 0

Forwards Routine: fftPad::forward2Many
Backwards Routine: fftPad::backward2Many
```

Data Collection

We then cache all the data.

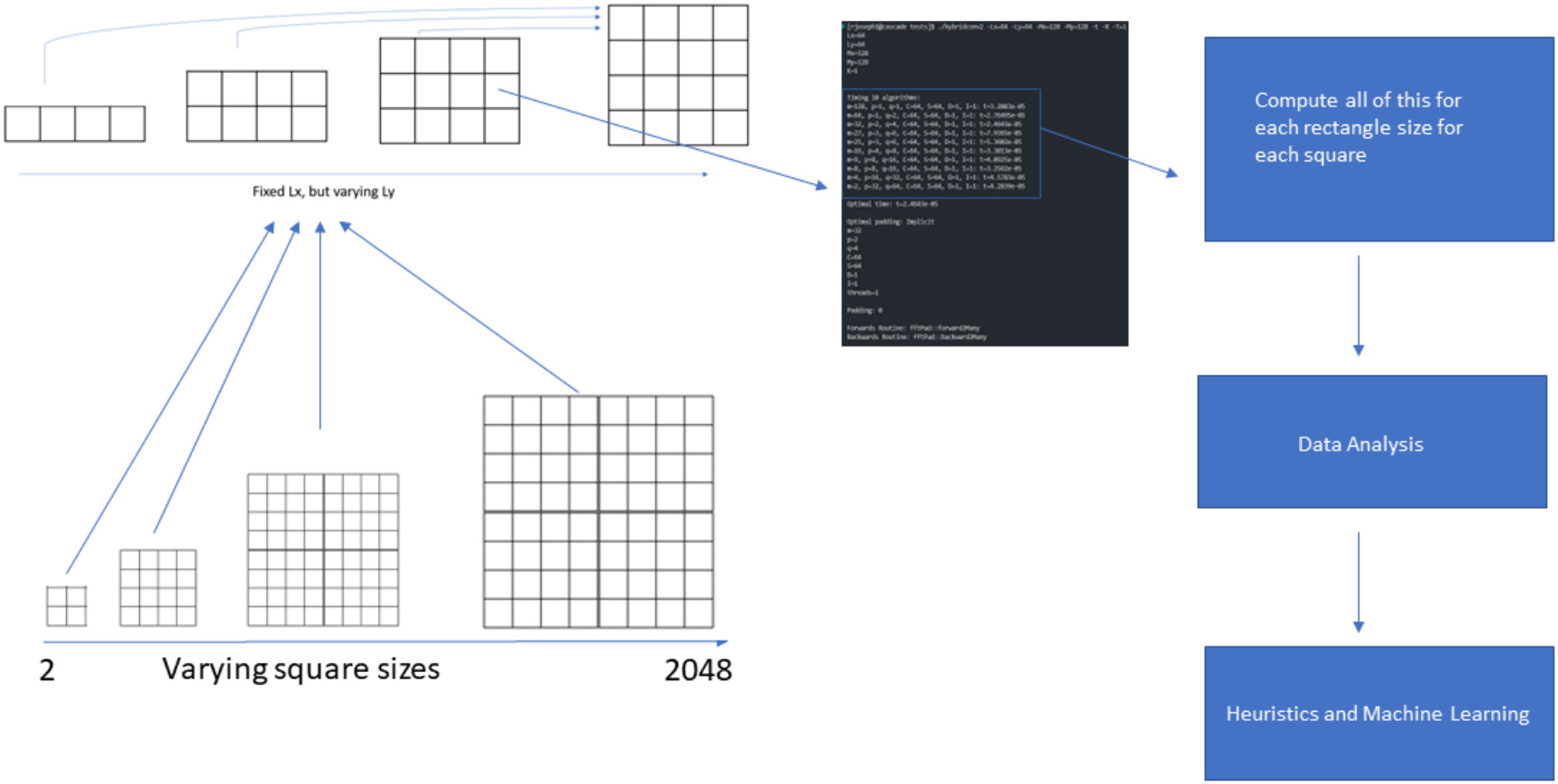




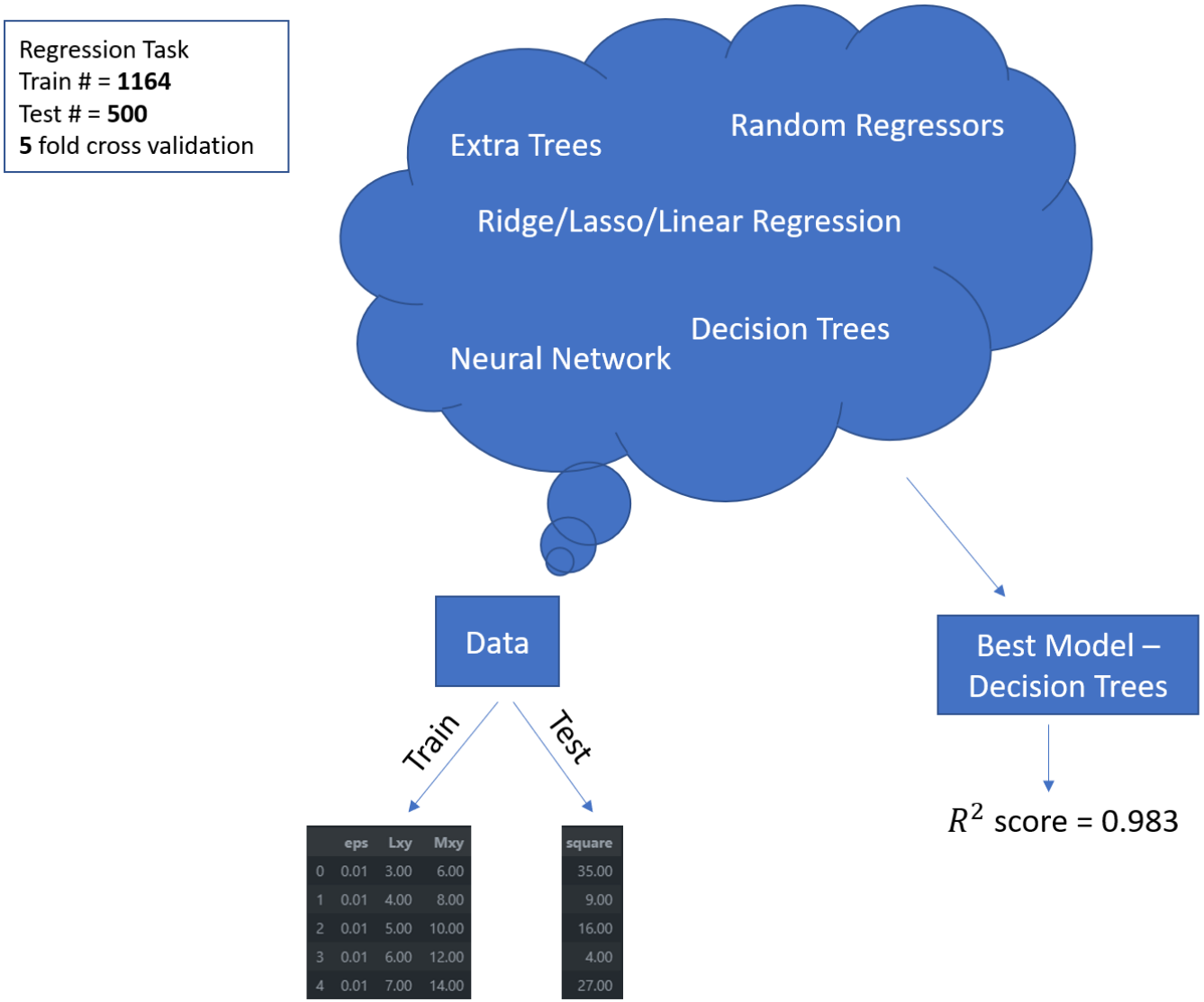
Compute all of this for each rectangle size for each square

Final Steps

Then we follow the standard Machine Learning Pipeline.



Can Machine Learning Save Us?



	eps	Lxy	Mxy
0	0.01	3.00	6.00
1	0.01	4.00	8.00
2	0.01	5.00	10.00
3	0.01	6.00	12.00
4	0.01	7.00	14.00

square
35.00
9.00
16.00
4.00
27.00

Caveats And Accuracy

- Hardware dependent.
- Does not generalize well, although for classification L_2 error penalizes values, one can check how close the predicted value timing is to the actual value timing. My conjecture is that it is almost comparable.

Although a regression task, forcing the output to be an integer is more harder.

R^2_train = 0.879
R^2_test = 0.976

Regressors

eps	Lxy	Mxy	predict	actual	diff
0.00	315.00	630.00	8.55	9	-0.45
0.50	189.00	378.00	9.48	9	0.48

Tried classification, results seem to be promising but more checks needed. Out of distribution again seems worse

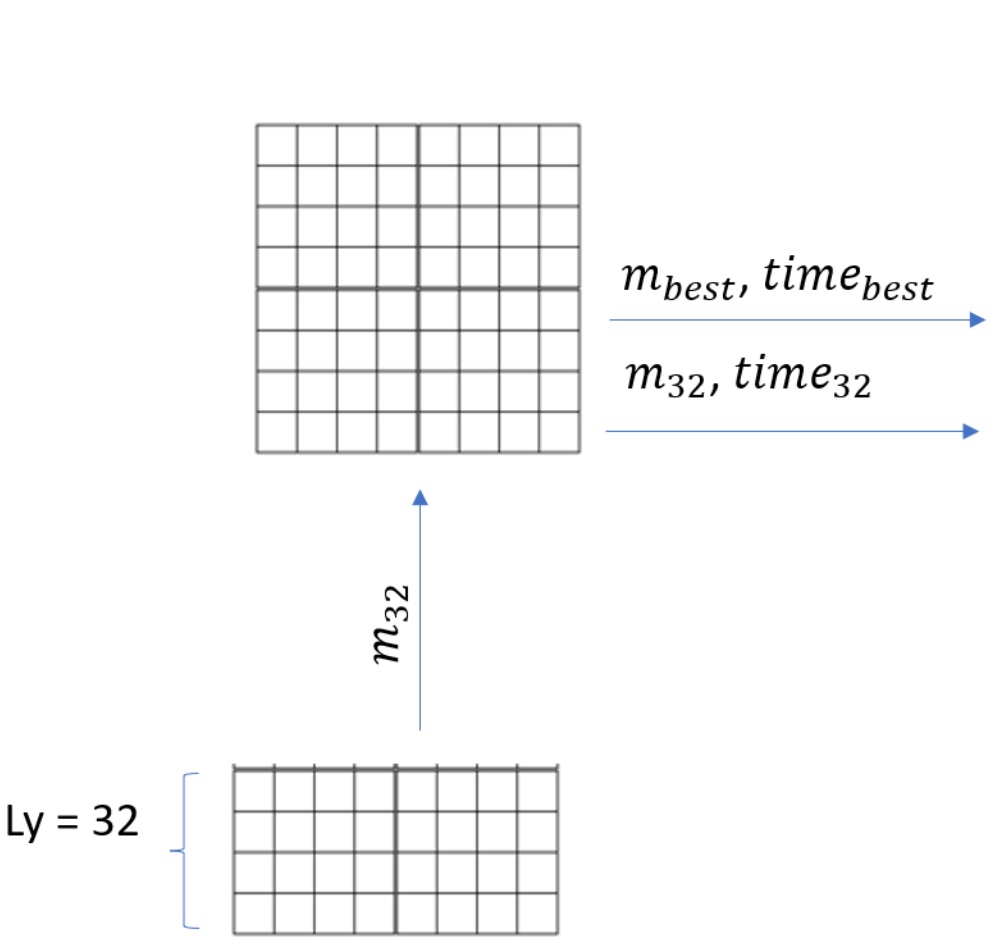
Accuracy_train = 0.971
Accuracy_test = 0.987

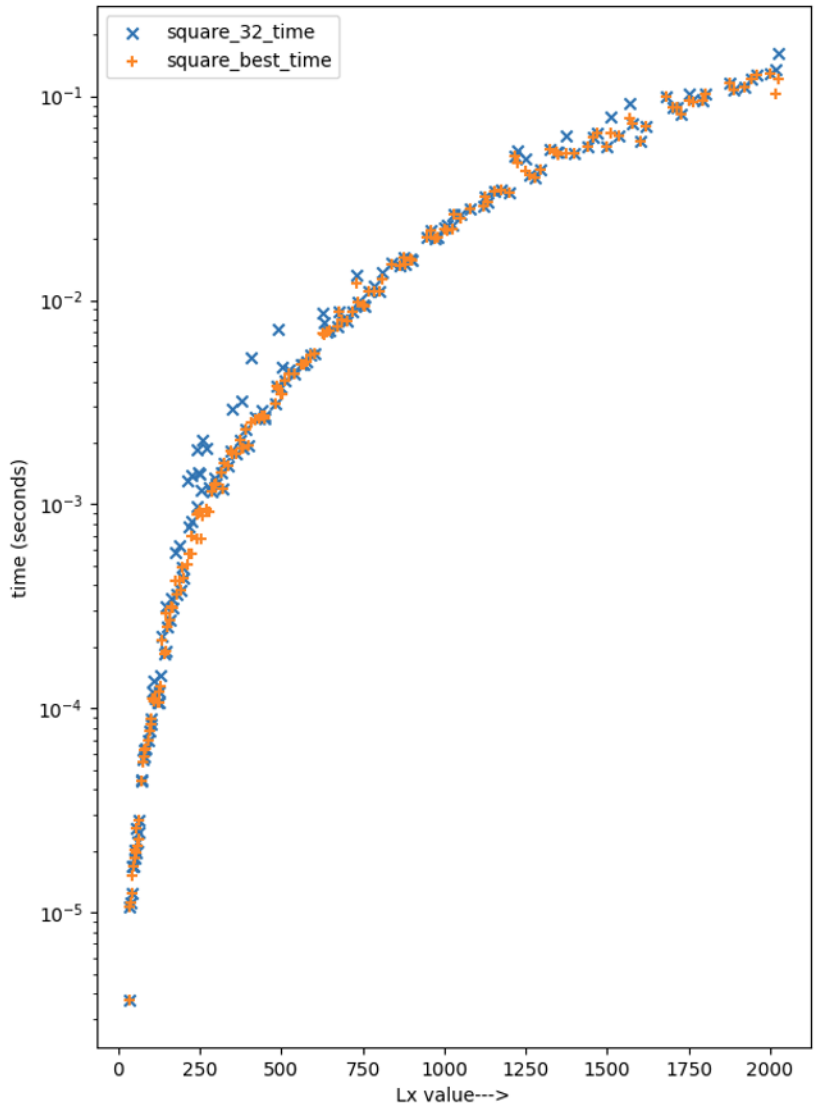
Classifiers

eps	Lxy	Mxy	predict	actual	diff
0.00	3.00	6.00	0.00	0	0.00
0.00	4.00	8.00	4.00	4	0.00

The Story Of 32

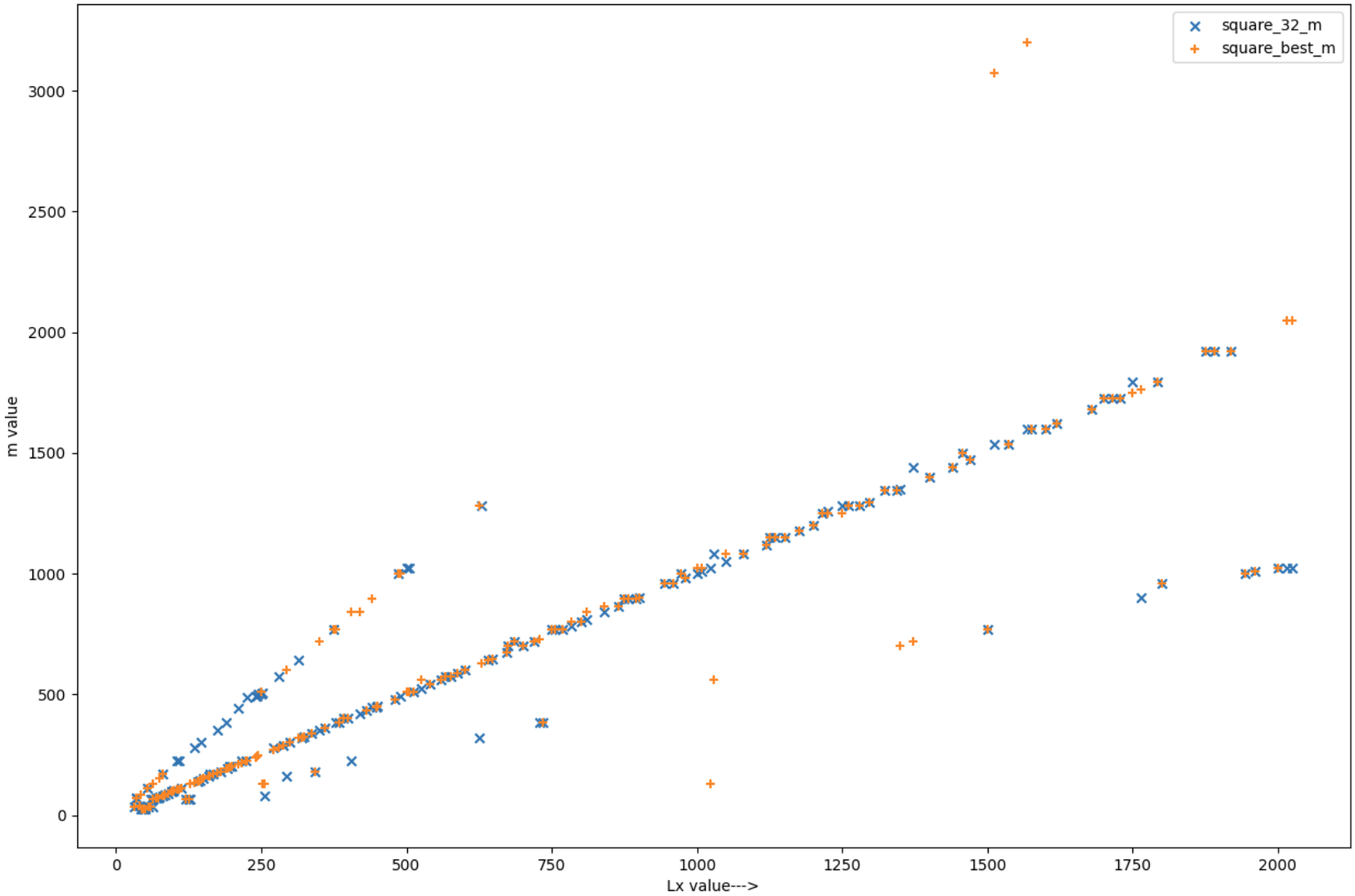
How do we compare?





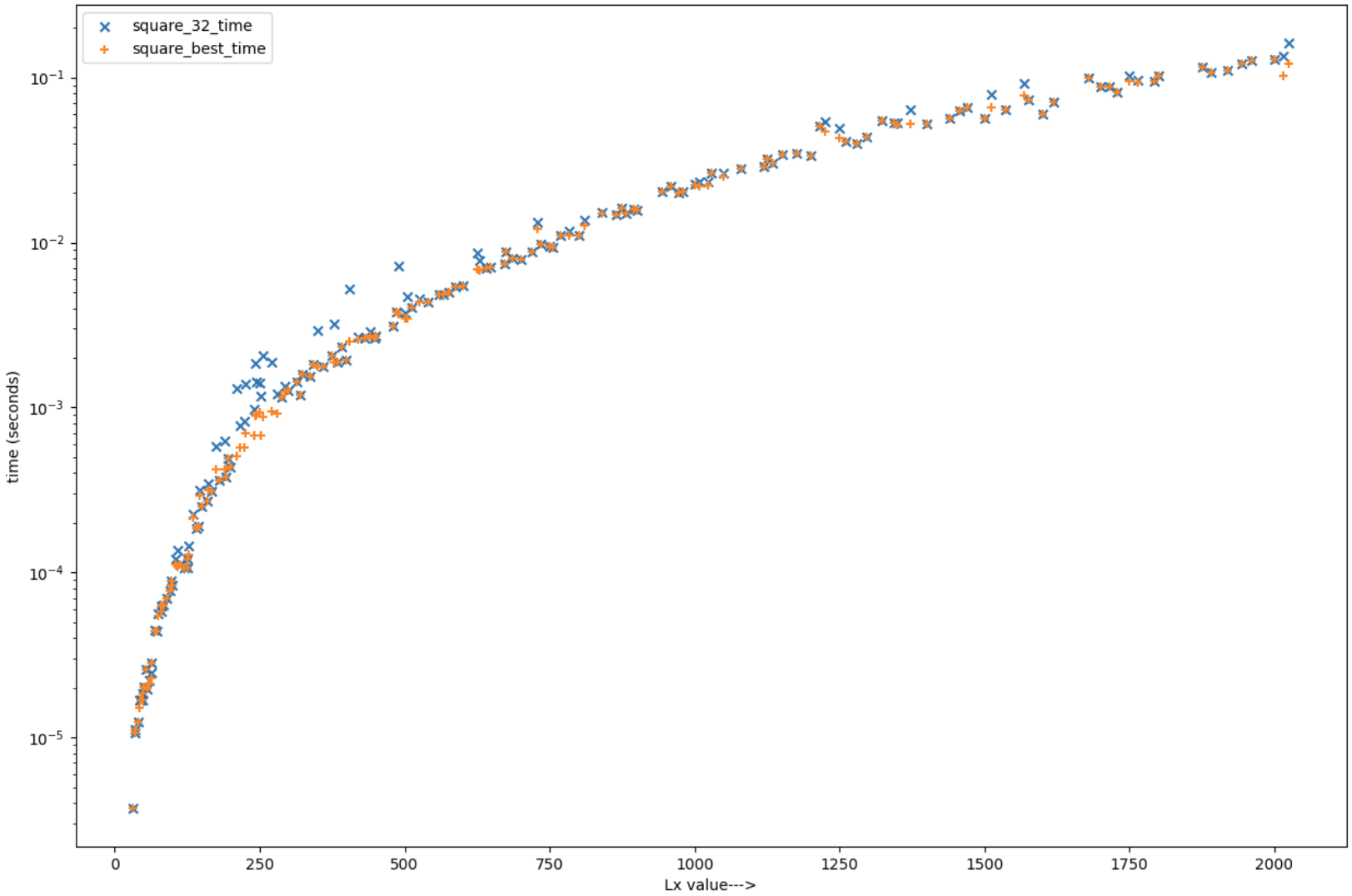
Rays Of Hope

Takeaway result 1: 3 interesting rays of m values.



Is 32 Enough?

Takeaway result 2: One can simply use very *skinny* rectangles instead.



Applications

1. These heuristics will be extremely useful in practice and can contribute to the success of the proposed hybrid dealiasing algorithm.
2. We expect hybrid dealiasing to become the standard method for convolutions.
3. We are developing hybrid dealiasing for real convolutions and convolutions with small kernels, such as in imaging and Convolution Neural Networks.

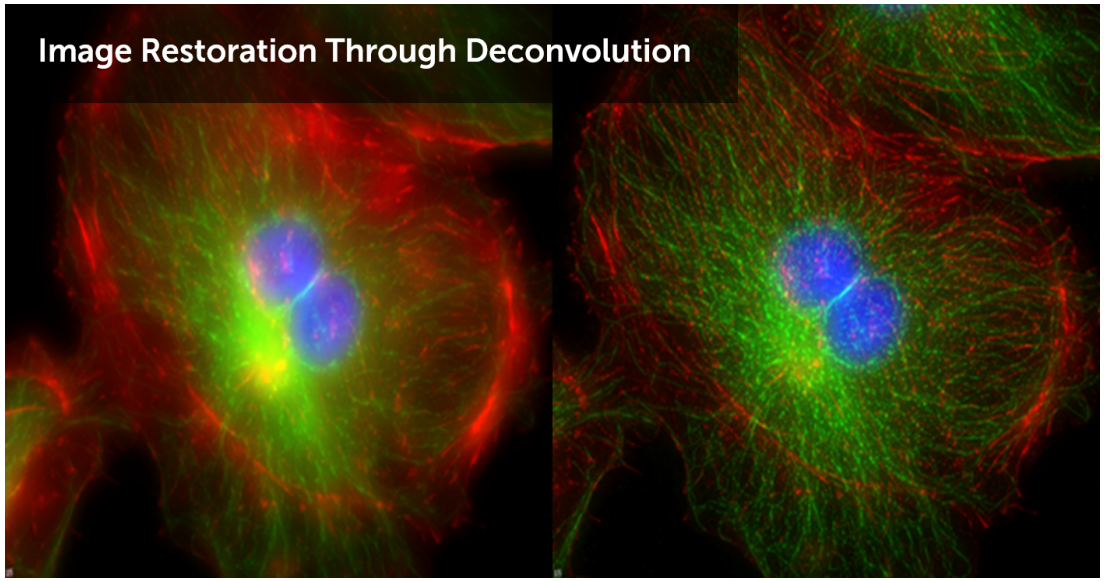


Figure 4: Initial and deconvolved images of fixed cells.

References

[1] George, Robert Joseph, Noel Murasko, and John C. Bowman. Hybrid Dealiasing Convolutions. *Joint Mathematics Meetings*, 2023.

[2] Noel Murasko and John C. Bowman. Hybrid Dealiasing of Complex Convolutions. *Submitted to SIAM J. Sci. Computer*, 2023.

[3] Efficient Dealiased Convolutions without Padding, J. C. Bowman and M. Roberts, *SIAM Journal on Scientific Computing* 33:1, 386-406 (2011).

[4] Multithreaded Implicitly Dealiased Convolutions, M. Roberts and J. C. Bowman, *Journal of Computational Physics* 356, 98-114 (2018).

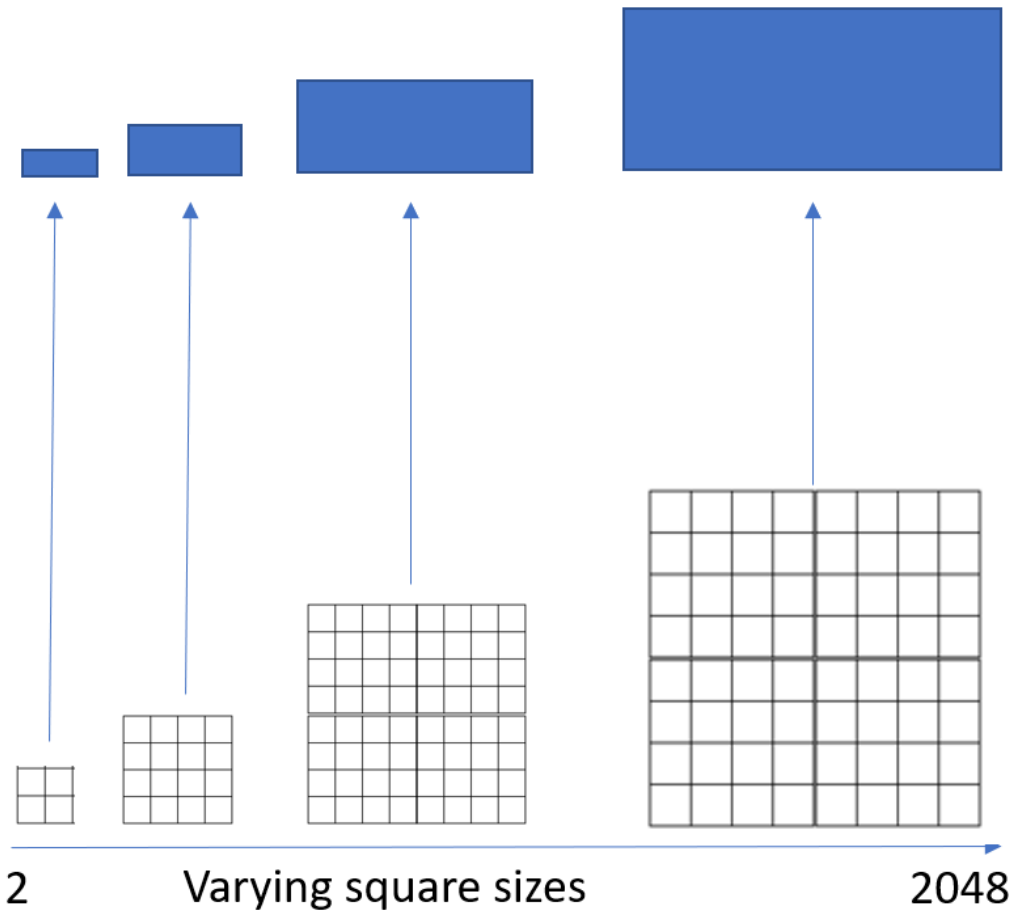
[5] Image Restoration Through Deconvolution, *Teledyne Photometrics*, 2019.

[6] Kernel (Image Processing). *Wikipedia, Wikimedia Foundation*, 12 Jan. 2023.

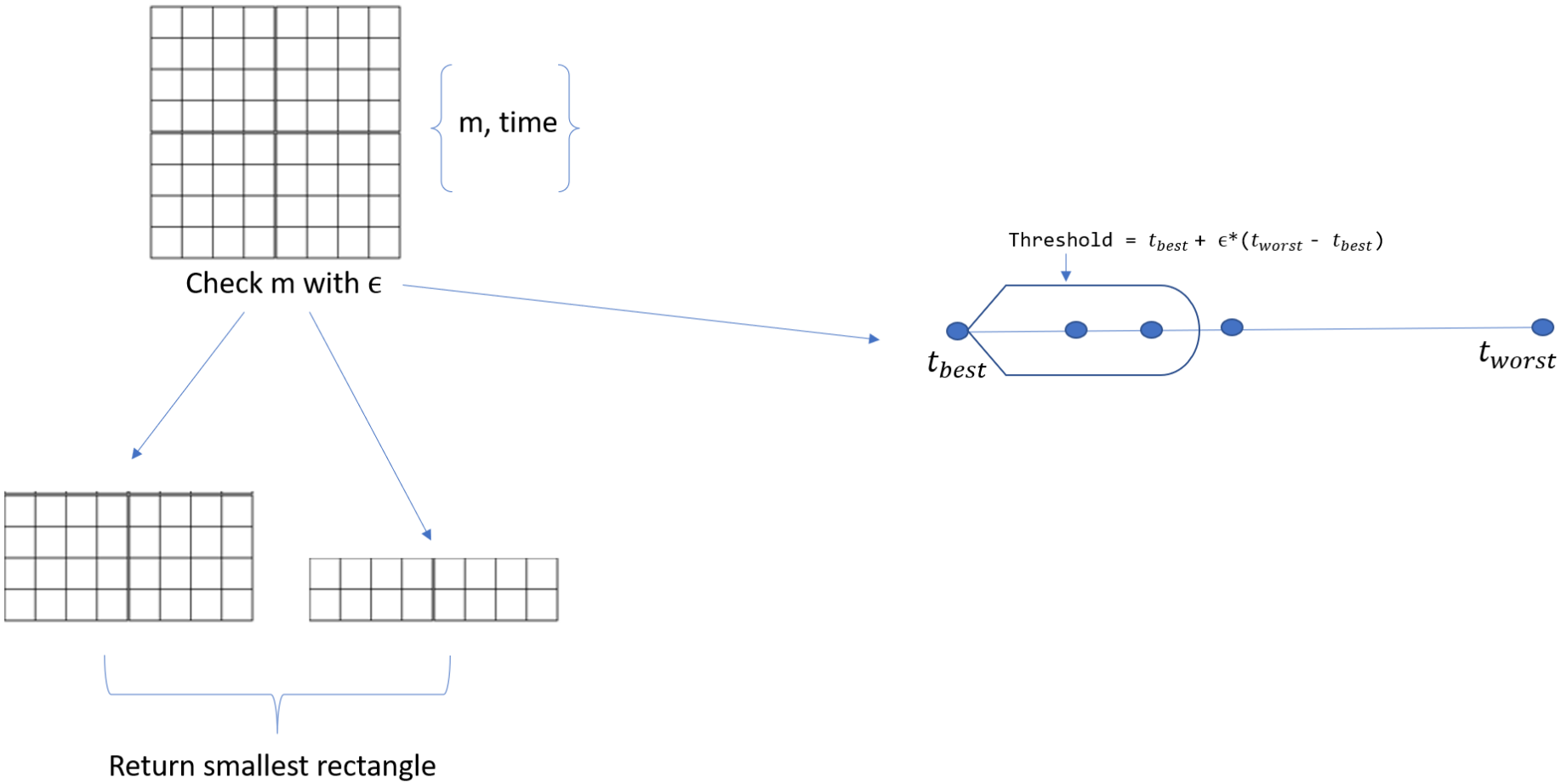
Can We Bound The Maximum Number of Entries?

What does this mean?

For each square, find the smallest rectangle within epsilon factor.
If no such rectangle exists, then return 0.

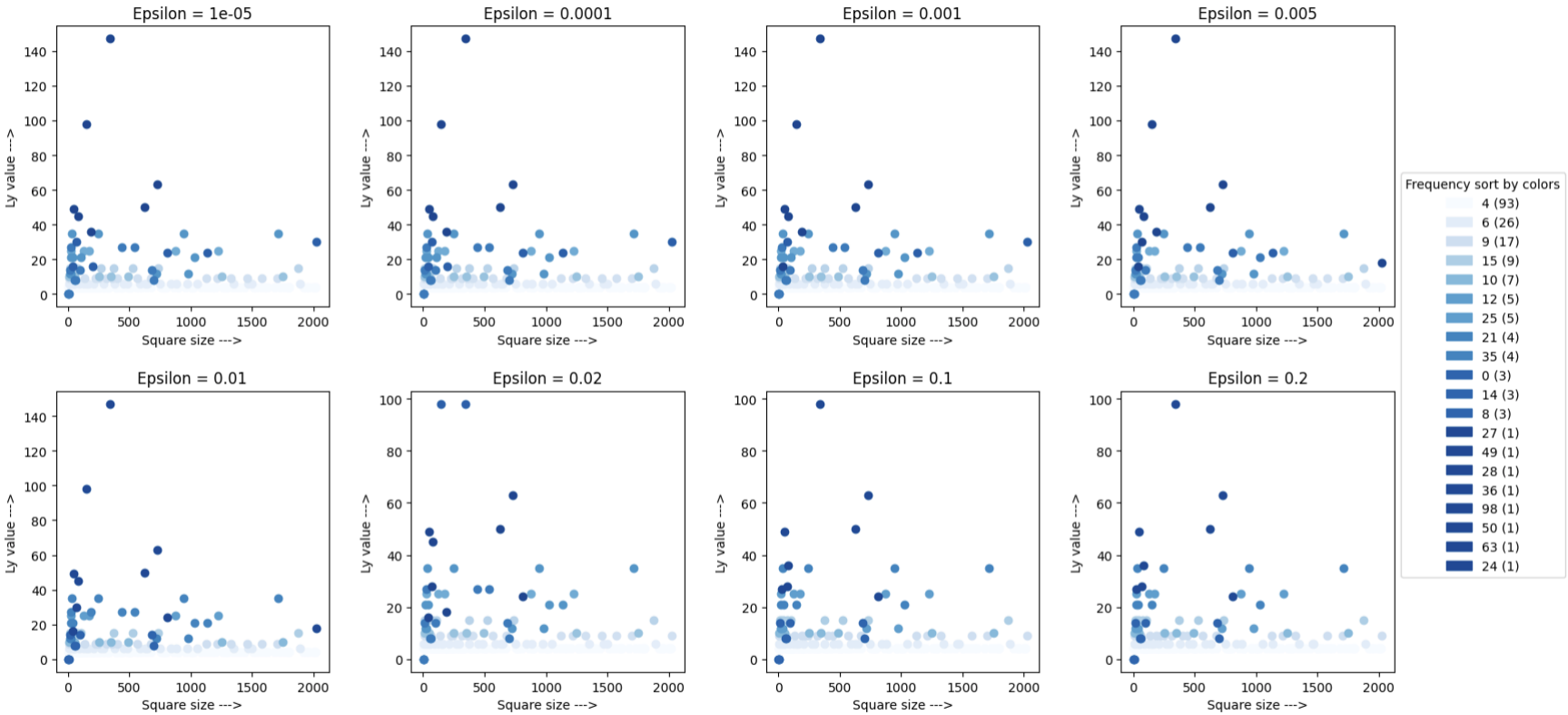


Algorithm Explained



Results for Various ϵ Thresholds

All values are bounded by 140.



Zoomed In Version