A Study on Convolution Using Half-Precision Floating-Point Numbers on GPU for Radio Astronomy Deconvolution

Mickaël Seznec[1,3], Nicolas Gac[1], André Ferrari[2], François Orieux[1]

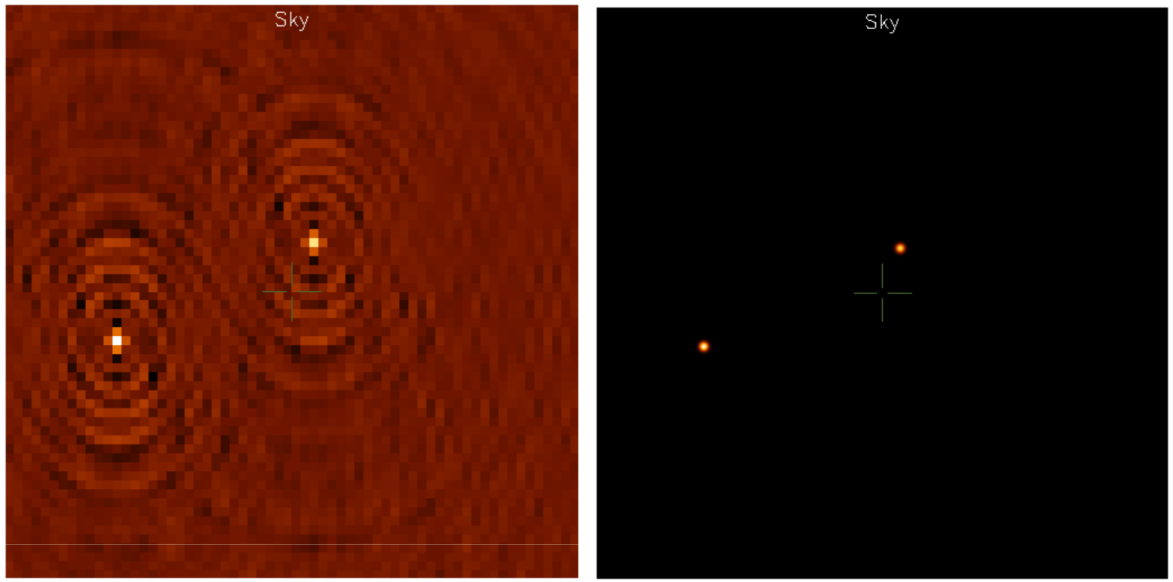
[1] L2S (CentraleSupélec, Univ. Paris-Sud, CNRS) Gif-sur-Yvette, FRANCE
[2] Lab. J.-L. Lagrange (Univ. Nice Sophia Antipolis, CNRS, Observatoire Côte d'Azur) Nice, France
[3] Thales Research & Technology, Palaiseau, France

IEEE SIPS, Special Session on Computing for Radio-Astronomy
2018/10/23

Outline

Deconvolution

Trying to recover data from an acquisition...



Source: I.M. Steward, Bielefeld University

Helps in the analysis in many fields: from radio astronomy to microscope images.

Modelisation

A physical measurement can be modeled as:

$$\boldsymbol{g} = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{n}$$

- $\boldsymbol{g} \in \mathbb{R}^M$ is the measured data
- $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ is the linear observation model (here: a convolution)
- $\boldsymbol{f} \in \mathbb{R}^N$ is the ground truth, unknown
- $\boldsymbol{n} \in \mathbb{R}^M$ is added noise, usually Gaussian

Regularization

$$\boldsymbol{J}(\boldsymbol{f}) = \|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2 + \lambda\|\boldsymbol{D}\boldsymbol{f}\|^2$$

The criterion involves two terms: data fidelity and regularization to compensate the ill-posedness behavior.
In the following, we choose $\boldsymbol{D} = \boldsymbol{I}^{M\times M}$
The solution is:

$$\hat{\boldsymbol{f}} = \arg\min_{f} \boldsymbol{J}(\boldsymbol{f})$$

$$\hat{\boldsymbol{f}} = (\boldsymbol{H}^t\boldsymbol{H} + \lambda\boldsymbol{I})^{-1}\boldsymbol{H}^t\boldsymbol{g}$$

The Hessian matrix's dimensions are too large to inverse it: we fall back on an iterative solver instead.

The gradient descent algorithm

Require: $\boldsymbol{H}$, λ, $\boldsymbol{g}$, ϵ, N, c

1: Set $\boldsymbol{b} = \boldsymbol{H}^t\boldsymbol{g}$ and $\boldsymbol{Q} = \boldsymbol{H}^t\boldsymbol{H} + \lambda\boldsymbol{I}$
2: Set $\boldsymbol{f}^{(0)}$ and $n \leftarrow 0$
3: **repeat**
4: $\quad \boldsymbol{k} \leftarrow \boldsymbol{H}^t\boldsymbol{H}\boldsymbol{f}^{(n)} - \boldsymbol{b} + 2\lambda\boldsymbol{f}^{(n)}$
5: $\quad \alpha \leftarrow \boldsymbol{k}^t\boldsymbol{k}/\boldsymbol{k}^t\boldsymbol{Q}\boldsymbol{k}$ ▷ or $\alpha \leftarrow c$ with $c \leq \frac{2}{\|\boldsymbol{Q}\|}$
6: $\quad \boldsymbol{f}^{(n+1)} \leftarrow \boldsymbol{f}^{(n)} - \alpha\boldsymbol{k}$
7: $\quad n \leftarrow n + 1$
8: **until** Some criterion is met ▷ e.g.: $n \geq N$
return $\boldsymbol{f}^{(n)}$

The gradient descent algorithm

Require: $\boldsymbol{H}$, λ, $\boldsymbol{g}$, ϵ, N, c

Set $\boldsymbol{b} = \boldsymbol{H}^t\boldsymbol{g}$ and $\boldsymbol{Q} = \boldsymbol{H}^t\boldsymbol{H} + \lambda\boldsymbol{I}$

Set $\boldsymbol{f}^{(0)}$ and $n \leftarrow 0$

repeat

  $\boldsymbol{k} \leftarrow \boldsymbol{H}^t\boldsymbol{H}\boldsymbol{f}^{(n)} - \boldsymbol{b} + 2\lambda\boldsymbol{f}^{(n)}$

  $\alpha \leftarrow \boldsymbol{k}^t\boldsymbol{k}/\boldsymbol{k}^t\boldsymbol{Q}\boldsymbol{k}$ ▷ or $\alpha \leftarrow c$ with $c \leq \frac{2}{\|\boldsymbol{Q}\|}$

  $\boldsymbol{f}^{(n+1)} \leftarrow \boldsymbol{f}^{(n)} - \alpha\boldsymbol{k}$

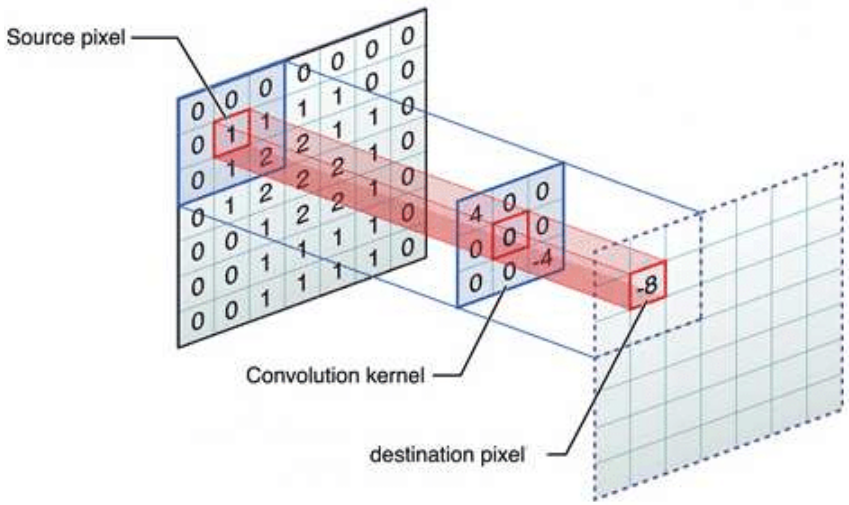
  $n \leftarrow n + 1$

until Some criterion is met ▷ e.g.: $n \geq N$

  return $\boldsymbol{f}^{(n)}$

Up to 4 convolutions per loop!

2D convolution



Convolving a $N \times N$ image with a $K \times K$ kernel can mainly be done in two ways:

- In direct space: complexity $O(N^2 K^2)$
- In Fourier space: complexity $O(N^3 \log(N))$

In special cases, optimization exists: separable kernel, overlap-add...

16-bit floats format

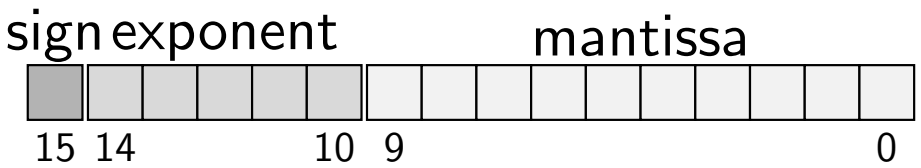
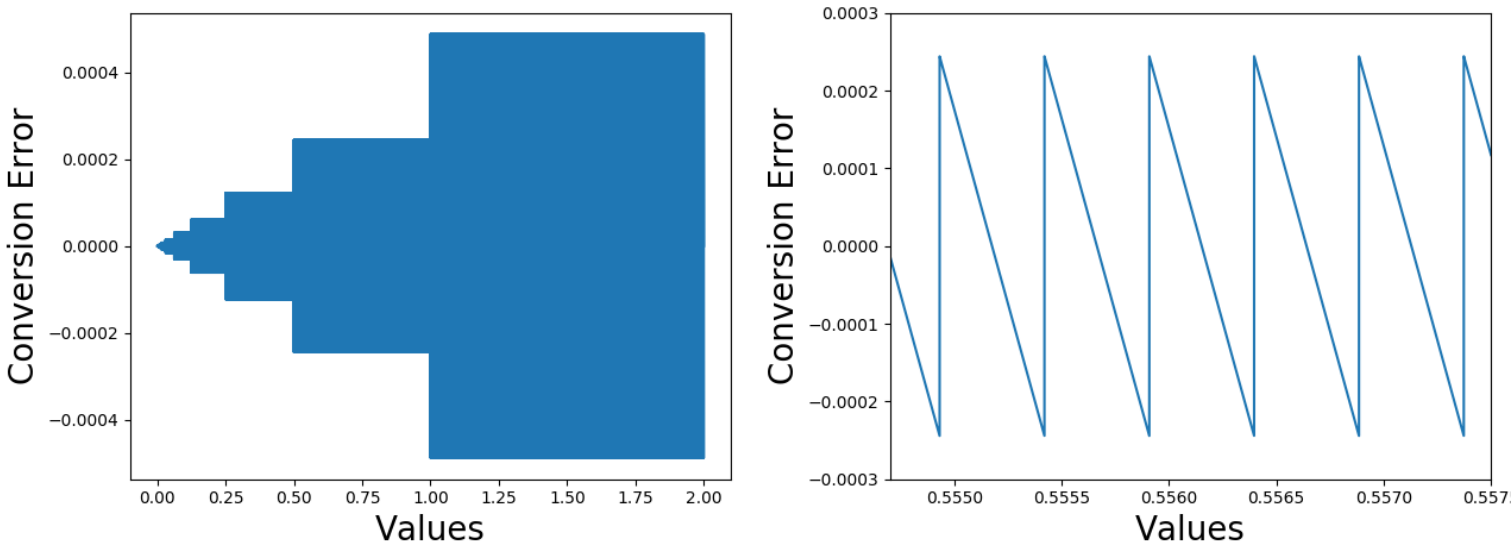


Figure: IEEE 754 binary16 format

- Supported on an increasing number of platforms
- Double the FLOPS on Nvidia GPUs via SIMD instructions
- Decreased memory and bandwidth usage
- Limited range (from 10^{-8} to 65504) and accuracy

From FP32 to FP16



- We suffer from quantization errors in the conversion
- The relative error stays the same

Results on convolutions

Convolution of a 512×512 image with a square kernel of variable size. Only the kernel execution time is measured.

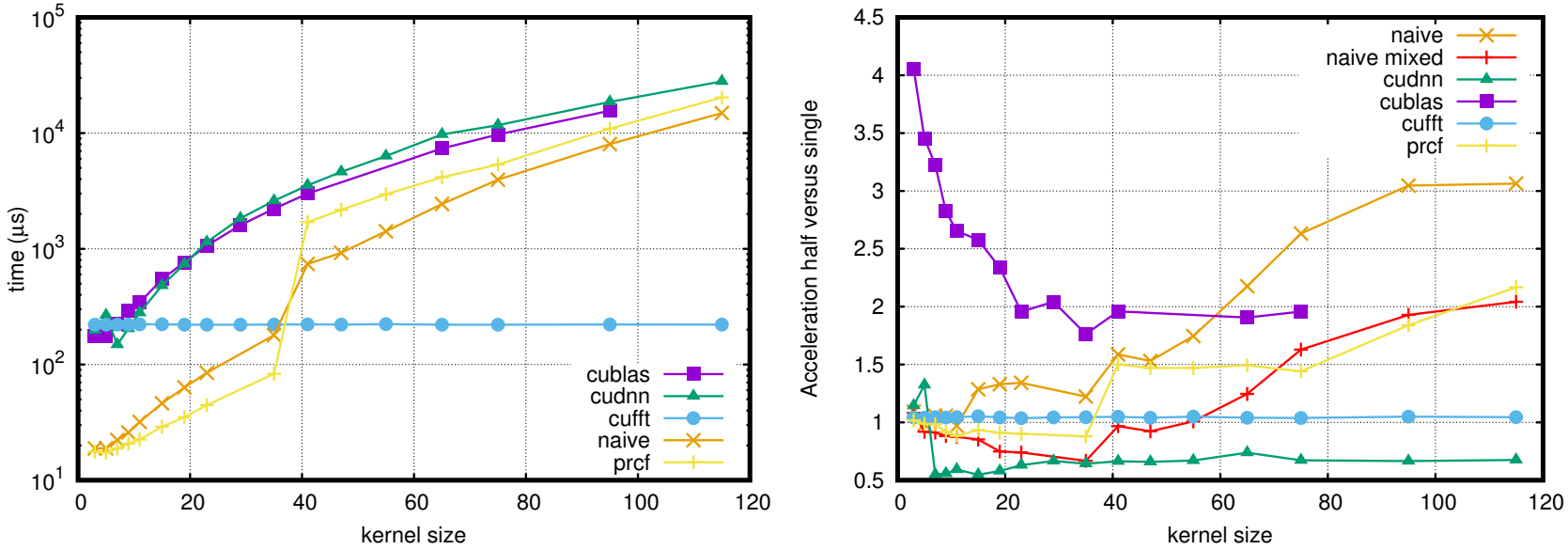


Figure: Left: results with FP32. Right: speed-up with FP16

- Naive and PRCF: own implementations.
- cuFFT looks constant because kernel is padded

Accuracy

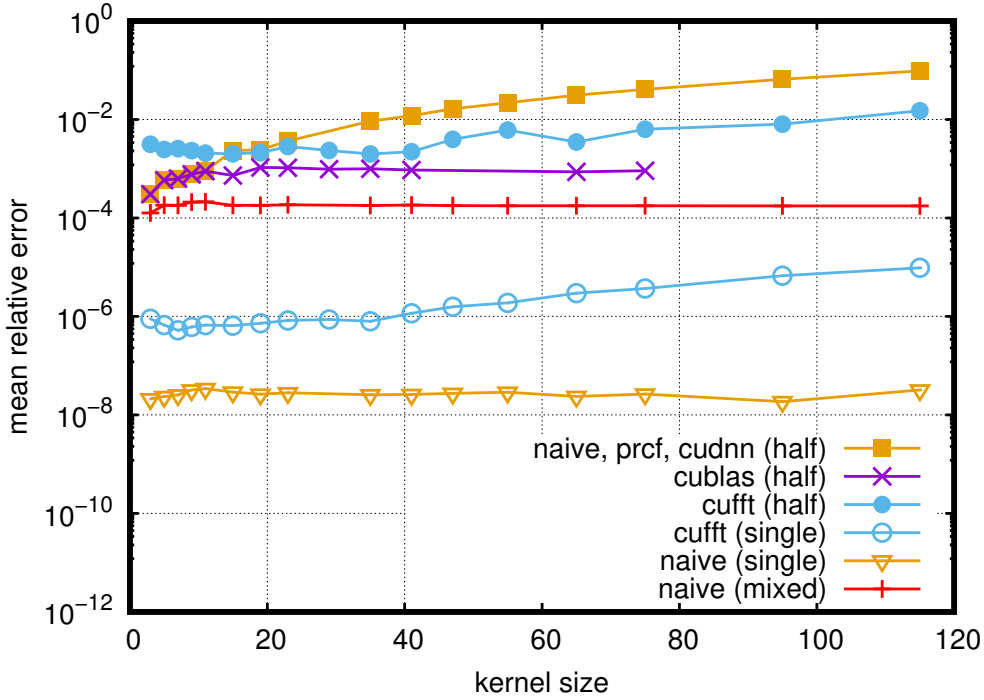
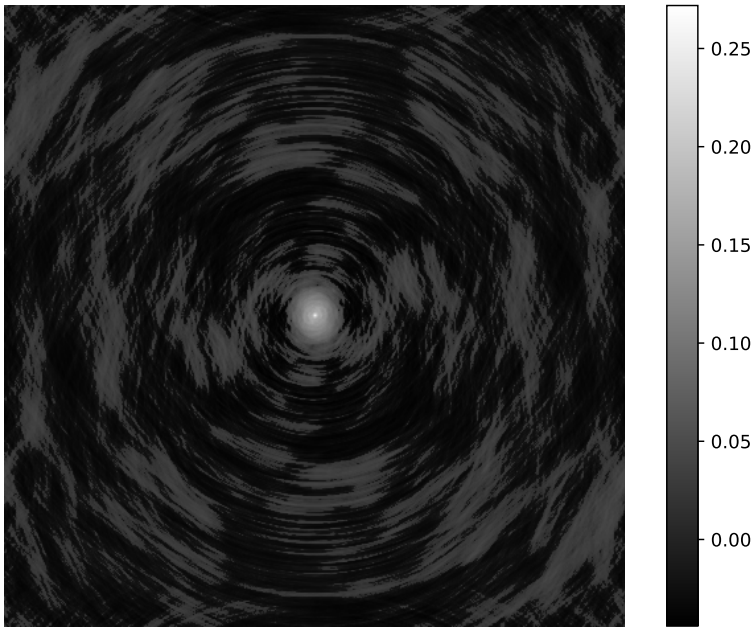


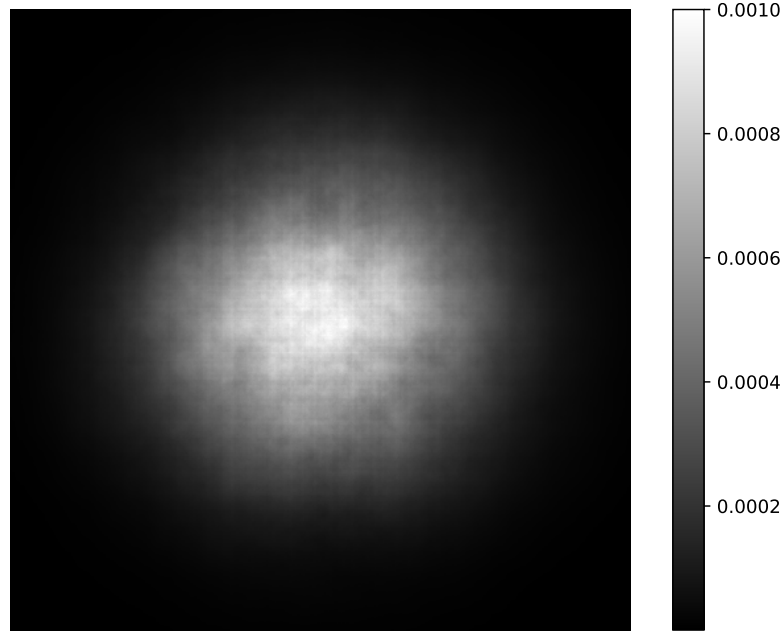
Figure: MRE compared to a reference implementation

$$MRE = \frac{1}{N} \sum \begin{cases} \left| \frac{x[i] - y[i]}{x[i]} \right|, \text{if } x[i] \neq 0 \\ 0, \text{otherwise} \end{cases}$$

Application to radio astronomy



PSF (cubic root)



Sky (clipped to 0.001)

1000:1 ratio between star values and background field.

Errors on "real" data

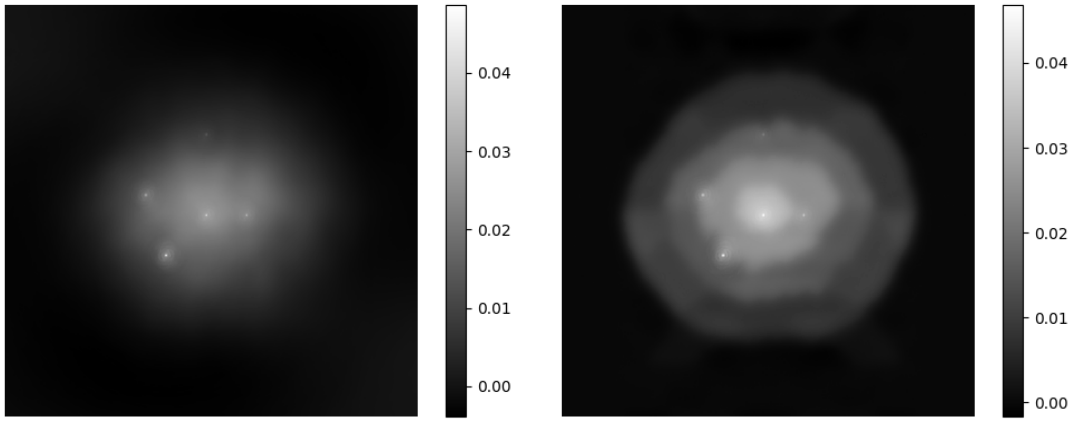


Figure: Left: convolution with FP32, right: with FP16

- Important loss of precision with a single convolution!

Deconvolution Results

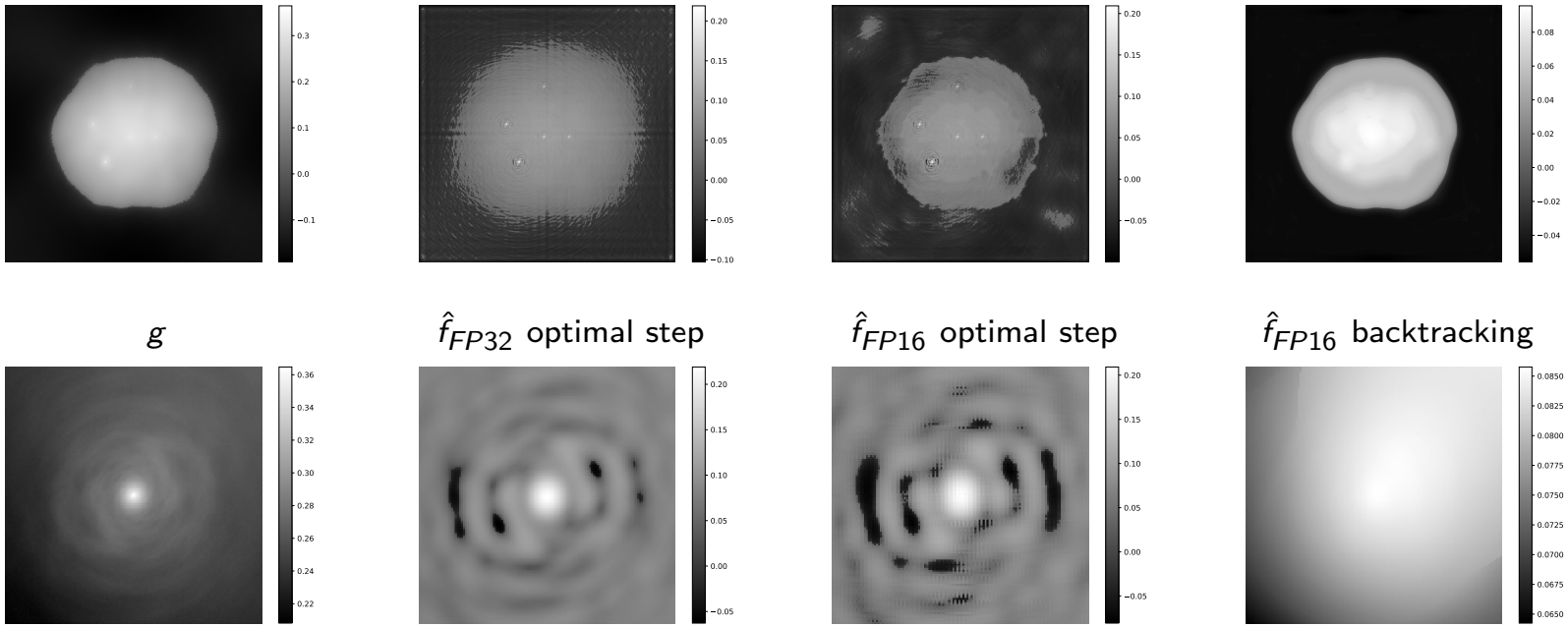
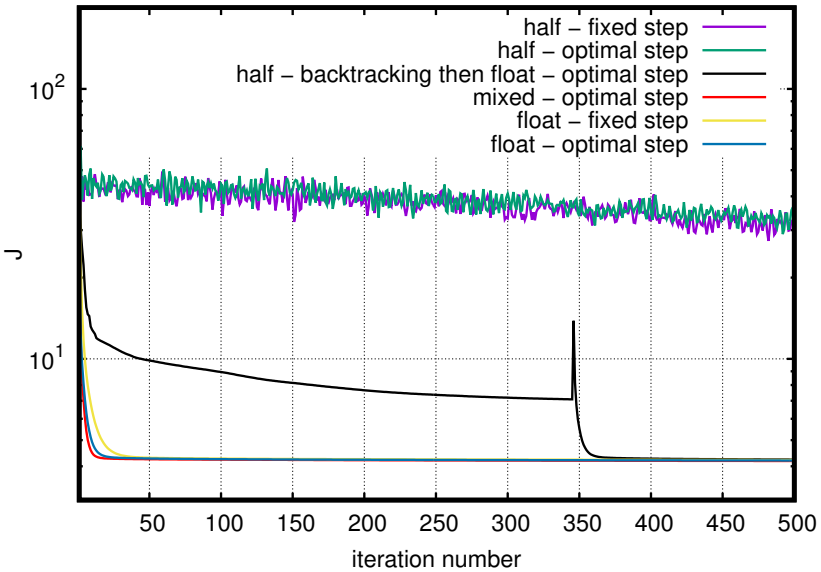


Figure: Reconstructions of f with different precisions (the cubic root is displayed for better contrast)

Minimizing the criterion

$$J(\boldsymbol{f}) = \|\boldsymbol{g} - \boldsymbol{H}\boldsymbol{f}\|^2 + \lambda\|\boldsymbol{f}\|^2$$



- Half-float only implementation does not have a satisfying behavior.
- Mixed strategy performs well.

Summary & Conclusions

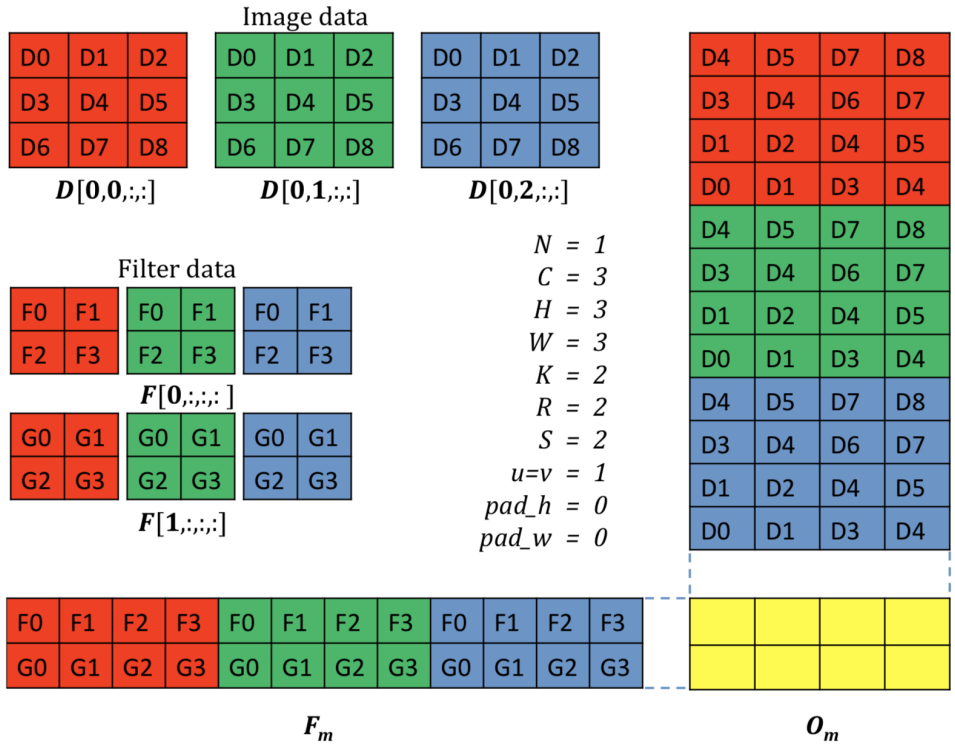
- We developed a GPU convolution benchmark
- We made a CUDA/C++ implementation of a deconvolution algorithm

Summary & Conclusions

- We developed a GPU convolution benchmark
- We made a CUDA/C++ implementation of a deconvolution algorithm

- Half floats can significantly speed-up convolutions (up to $\times 2, \times 3$)
- Loss of precision must be tightly controlled...
- ... but can be mitigated with the use FP32 compute / FP16 storage

Thank you!



Sharan Chetlur et. al. 2015