Ptychography-guided processing of XRF

Matthias Ehrhardt, Paul Quinn, Jarrod Williams

March 2020

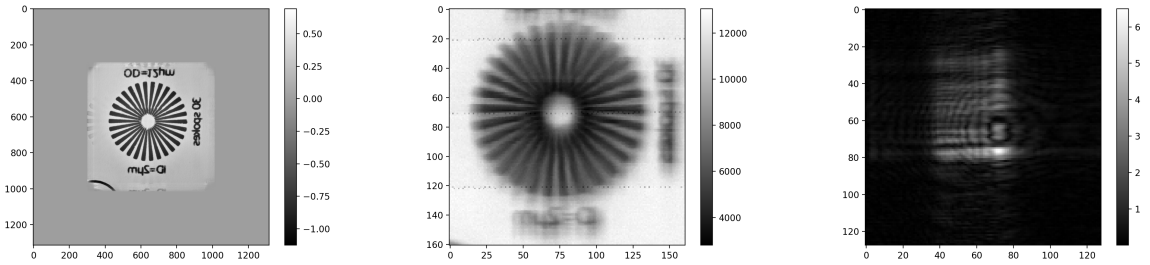
1 Introduction

Aim: to denoise, deblurr (and possibly super-resolve) XRF images using a correlated ptychography image.

What we have:

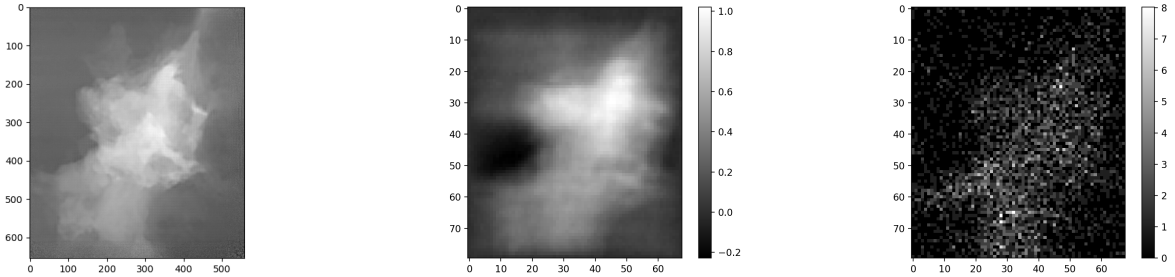
- $u_1, \cdots, u_N$: noisy XRF element maps (to be processed),
- v: high-resolution ptychography image —to be used as guide in above processing, potentially in both kernel estimation and directly in processing of XRF,
- probe modulus. Square this to get estimate of blurring kernel, K_0,
- w: phase contrast image (to be used for improved kernel estimation).

Idea: the (low-resolution) phase contrast image w, when deblurred, should correlate with the high-resolution ptychography image.



(a) Ptycho image (b) XRF image (c) Probe modulus

Figure 1: Dataset 1.



(a) Ptycho image (b) Phase constrast (c) XRF Ca

Figure 2: Dataset 2. Question for Paul: has the XRF data been normalised? Need to know for modelling –re-introduce scale to be able to use KL.

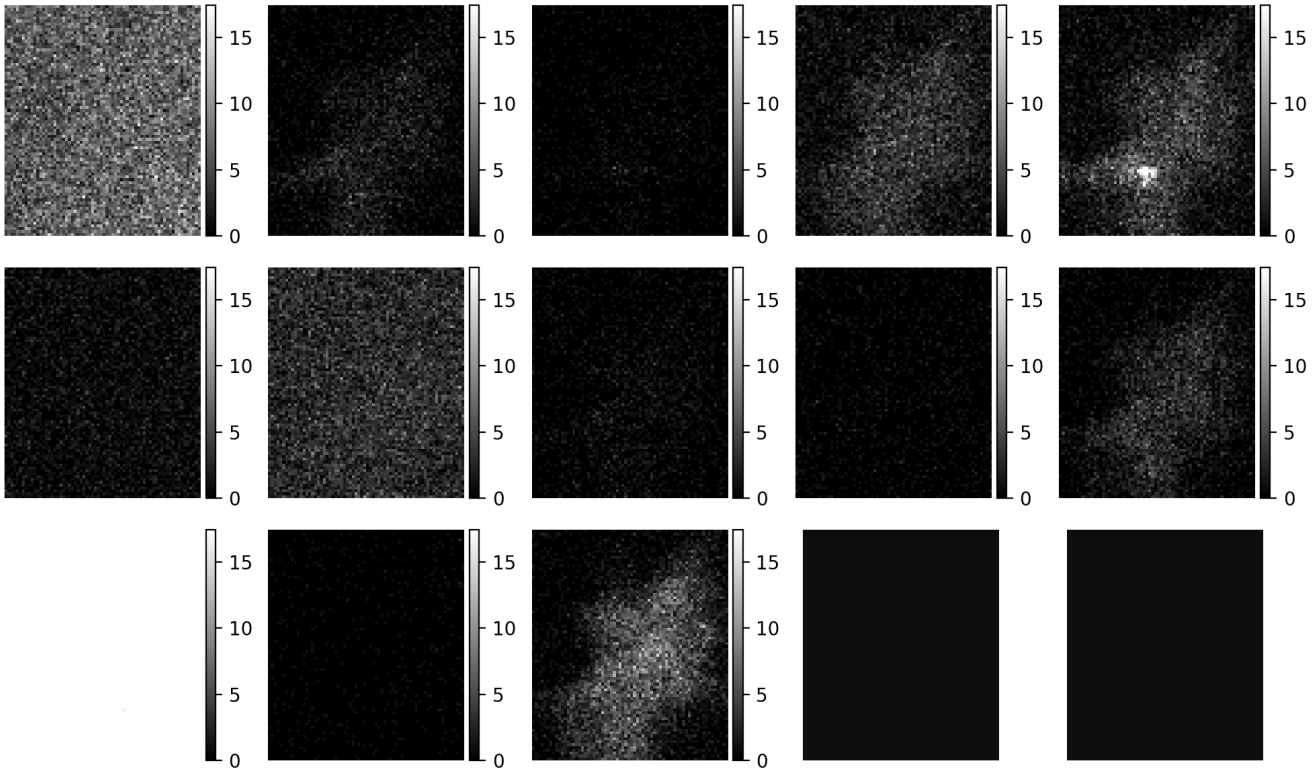


Figure 3: XRF element maps.

2 Deblurring (+upsampling) for dataset 1

2.1 Using K_0 - see "Non blind deconvolution XRF with PDHG" script

TV deblurring model:

$$x^* = \text{argmin}_x \; D(K_0 \star x, u) + \lambda \cdot TVNN(x)$$

dTV deblurring model:

$$x^* = \text{argmin}_x \; D(K_0 \star x, u) + \lambda \cdot dTVNN(x; v)$$

with u denoting the XRF image, v denoting the ptychography image.

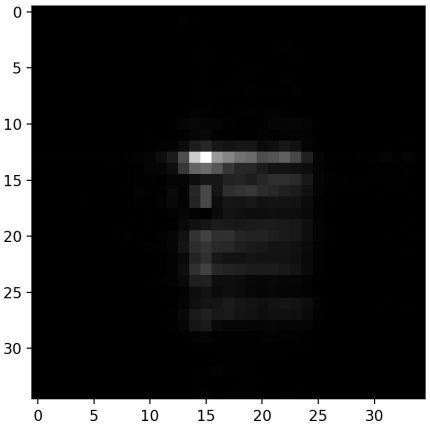
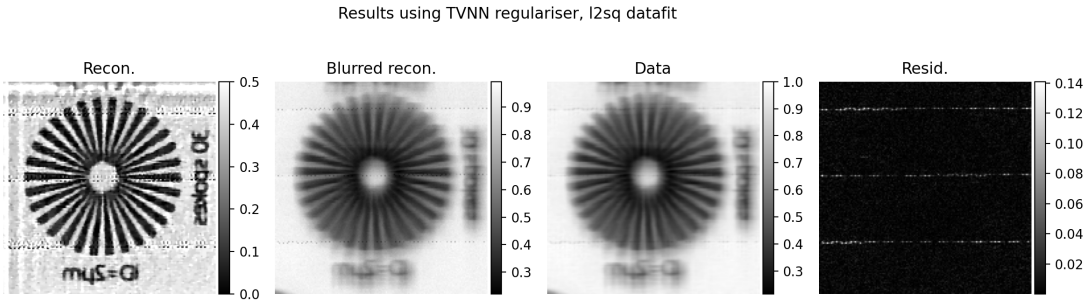


Figure 4: Blurring kernel K_0: square probe modulus, rotate about 180 deg and subsample to match lower resolution.

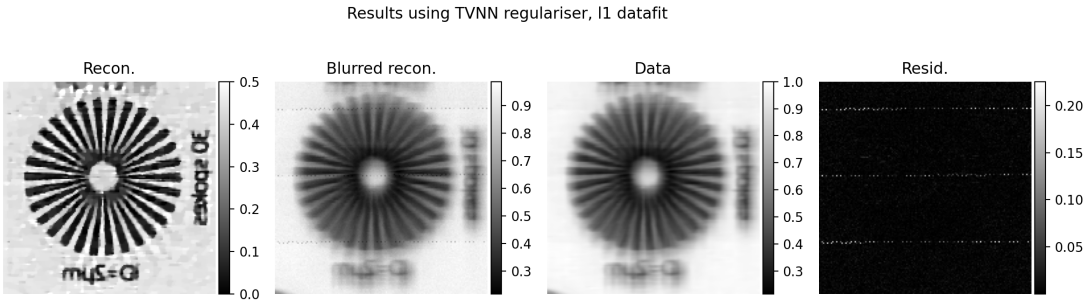
When doing dTV-guided non-blind deblurring we have to guess where the guide image should be. To do this, one approach is to pass the candidate guide image through the forward operator (i.e. blurring or blurring+downsampling) and compare by eye with the blurry data. Another option is perform non-guided deblurring (deblurring+upsampling)

and compare the result with the candidate guide image. Perhaps blind deconvolution is the way to go - then translational mis-registration can be accounted for by the kernel.

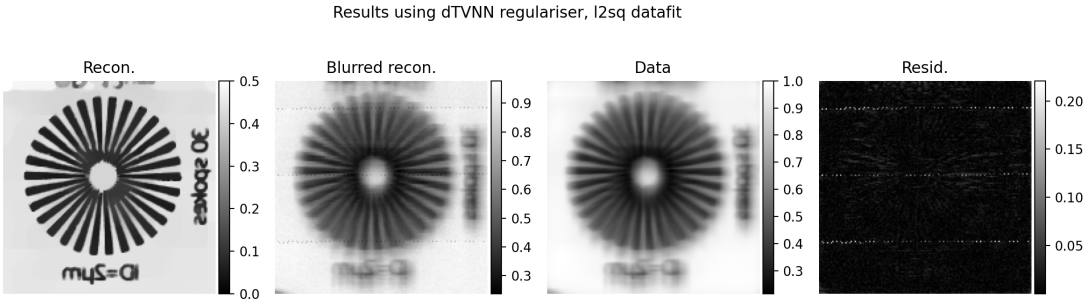
The algorithm I'm using is PDHG, though I'm cheating and computing the proximal of TVNN/dTVNN as a subroutine. I could re-implement this, absorb the derivative into the linear operator.



(a)

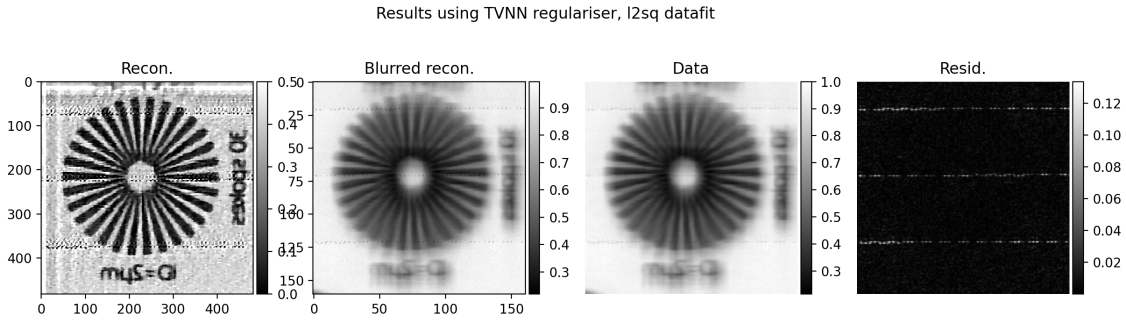


(b)

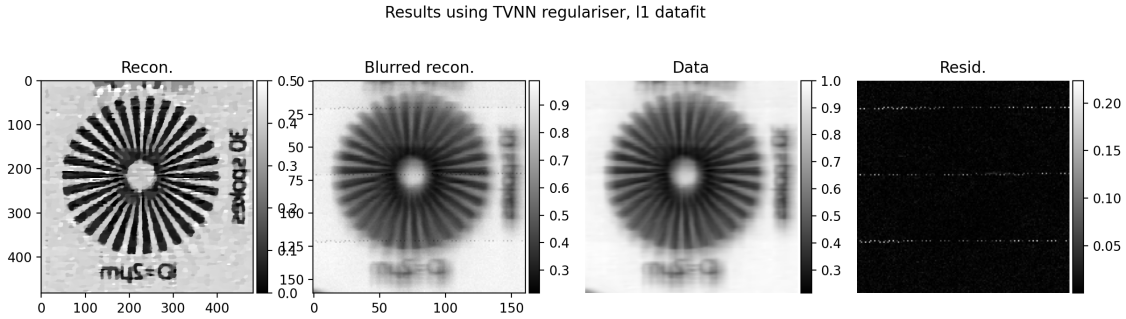


(c)

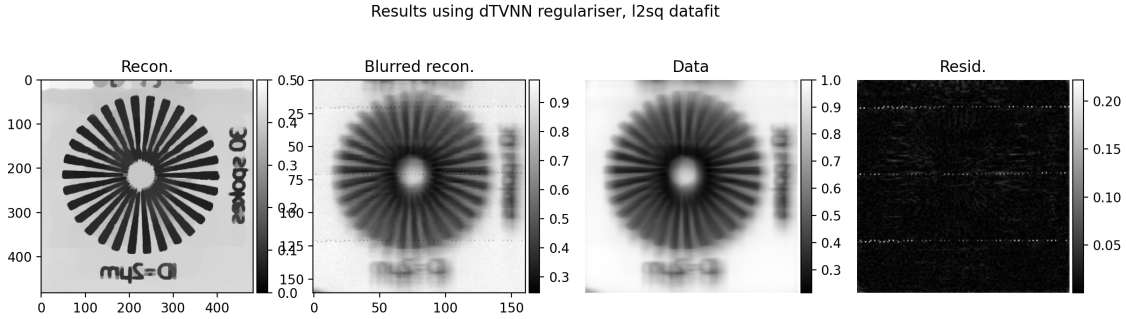
Figure 5: Deblurred images.



(a)



(b)



(c)

Figure 6: Deblurred, upsampled images.

2.2 Blind deblurring

$$K^*, z^* = \text{argmin}_{K,z} \ \|K \star z - w\|_2^2 + \lambda dTVNN(z; v) + \iota_{\mathbb{S}}(K)$$

Algorithm: PALM.

Note: concern that this won't work very well for very noisy data (e.g. dataset 2). An alternative approach would be linearised Bregman.

3 Dataset 2

In what follows, D could either be the L2-squared norm, as before, or the KL divergence (which is adapted to Poisson noise).

3.1 Single-map denoising only (no deblurring) —see "TV and dTV 27042021" script

TV model:

$$x^* = \text{argmin}_x \ D(x, u) + \lambda \cdot TVNN(x)$$

dTV model:

$$x^* = \text{argmin}_x \; D(x, u) + \lambda \cdot dTVNN(x; v)$$

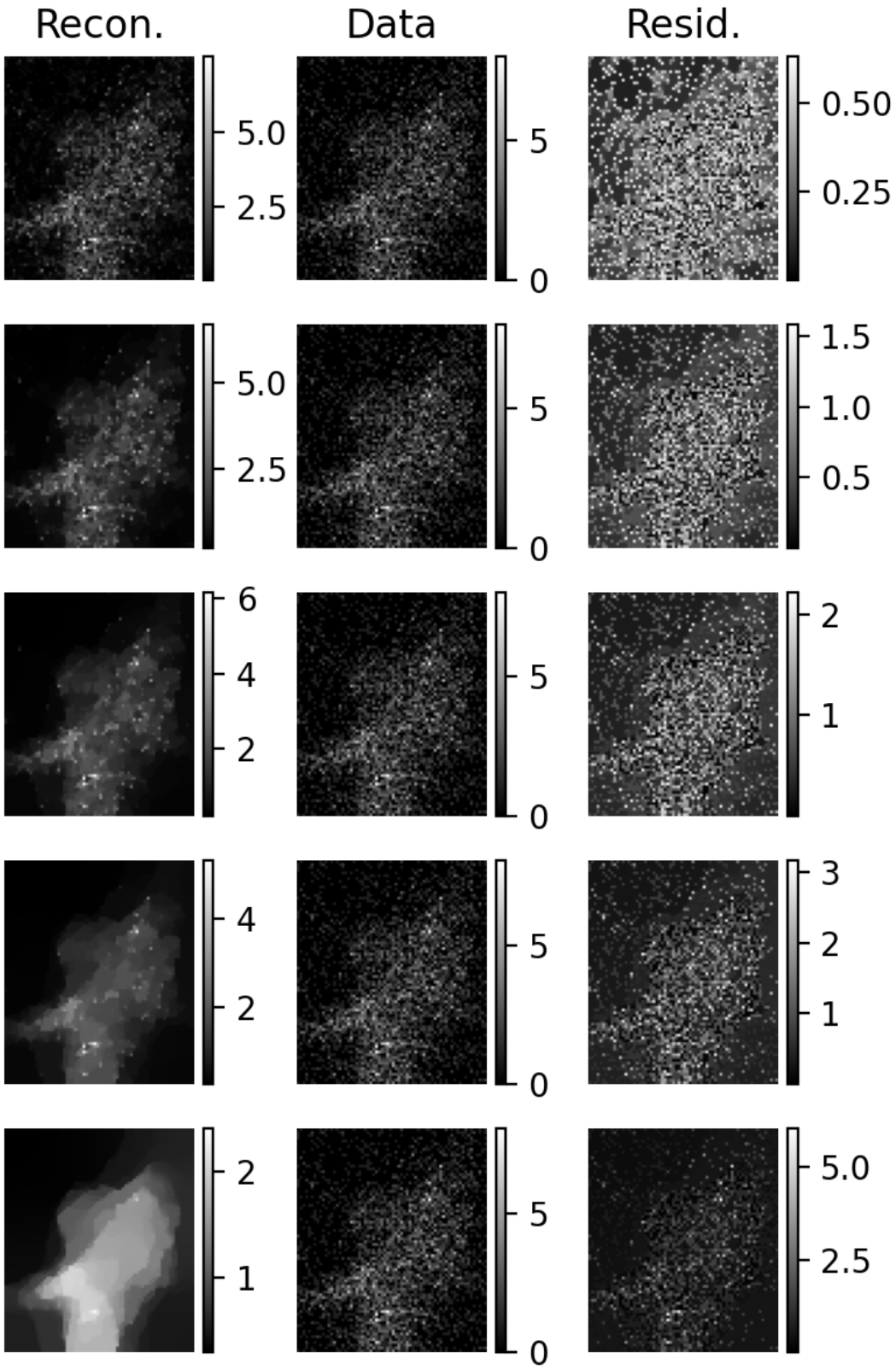


Figure 7: Recons using TV regulariser, l2sq datafit. $\lambda \in \{0.01, 0.025, 0.035, 0.05, 0.1\}$

3.2 Estimating the blur kernel

A kernel that maps ptycho onto phase-contrast:

$$K^* = \text{argmin}_K \ \|K \star v - w\|_2^2 + \iota_{\geq 0}(K)$$

A kernel for which the deblurred image exhibits edge-set correlation with phase-contrast:

$$K^*, z^* = \text{argmin}_{K,z} \ \|K \star z - w\|_2^2 + \lambda dTVNN(z; v) + \iota_{\mathbb{S}}(K)$$

Only keep K^*, throw away z.

Note: A slight mis-registration between v and w shouldn't matter here, since it should be absorbed into the estimated kernel. We have to do some normalisation just to make sure both v and w are non-negative.
Algorithm: PALM - in the first case only update K, in the second update both K, z. PALM is overkill for the first - could use PDHG, Vu–Condat etc.

3.3 Single-map deblurring and denoising

Similar to deblurring of toy dataset, but where we allow D to be more general —note

TV model:

$$x^* = argmin_x \ D(K \star x, u) + \lambda \cdot TVNN(x)$$

dTV model:

$$x^* = \text{argmin}_x \ D(K \star x, u) + \lambda \cdot dTVNN(x; v)$$

Algorithm: PALM.

3.3.1 Results using K_0

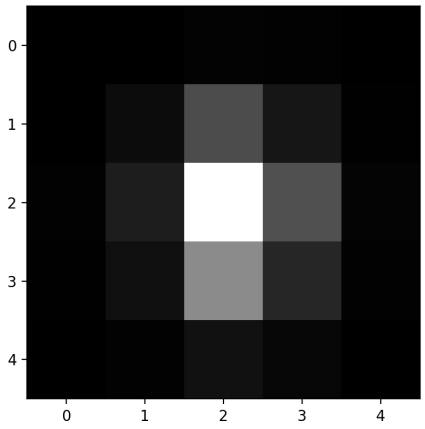


Figure 8: Blurring kernel K_0: square probe modulus, rotate about 180 deg and subsample to match lower resolution.

3.3.2 Results using optimised K^* from previous section

3.4 Multi-map deblurring and denoising

Idea: the appropriate weighted sum of u_i^* should be correlated (in the sense of edge locations) with the ptychography guide.

$$\underline{x}^* = \text{argmin}_{\underline{x}} \sum_{i=1}^{N} \left(D(K \star x_i, u_i) + \lambda_i TVNN(x_i)\right) + \mu dTVNN(\alpha_1 u_1 + \cdots \alpha_N u_N; v)$$

where α_i are the known (?) coefficients, $\lambda_i (i = 1 \cdots N), \mu$ are the regulrisation parameters, to be determined.

3.5 Blind deconvolution and denoising

$$\text{argmin}_{K,x} \ D(K \star x, u)$$

Initialise K at probe modulus squared.