Problem-Statement

Let an object have a reflectance given by r. Then the complex-distribution of the object-field is given by $g \sim CN(0, D(r))$ [1]. The problem statement is given

$$y = Ag + w$$

we would like to estimate r.

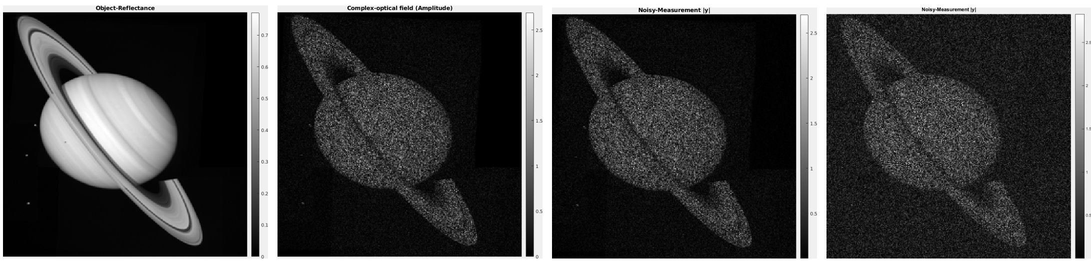
1 Case I: Linear measurements $(A = I)$

1.1 Forward-model:

Let the reflectance of the object that is imaged be given by $r \in [0, 1]$. Under the assumed model of the object-filed given by $g \sim CN(0, D(r))$, and the additive white-gaussian noise model we have that the measured complex-signal is given as

$$y = g + w.$$

A single-realization of this is given in Figure with std. deviations of the noise at $\sigma_w = \{0.001, 0.1\}$.



(a) Reflectance (r) (b) Optical-Field $(|g|)$ (c) $(|y|)$ $\sigma = 0.001$ (d) $(|y|)$ $\sigma = 0.1$

Figure 1: An example-realization of the forward-model $y = g + w$. Two realizations of the noisy observations are shown at std. deviations of $\{0.001, 0.1\}$.

1.2 ML-estimate

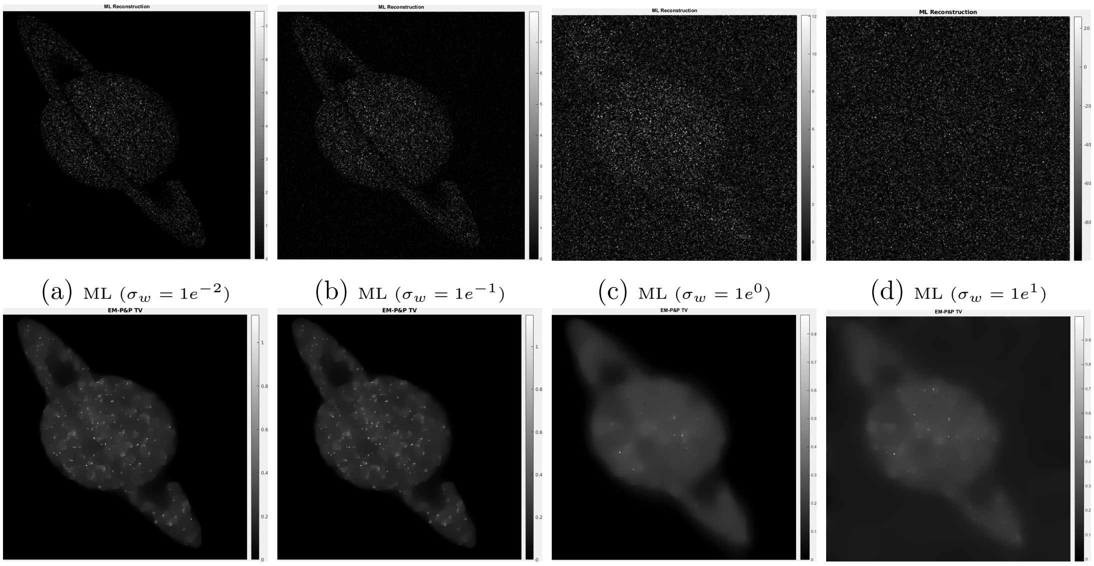
The likelihood of the measurements y is given as $p(y|r) \sim CN(0, D(r) + \sigma_w^2 I)$. Applying the maximum-likelihood estimator to the above problem we have

$$\begin{aligned}
\hat{r} &= \arg\max \log p(y|r) \\
&= \arg\max \frac{-1}{2}\left(\log\left((2\pi)^N |D(r) + \sigma_w^2 I|\right) + y^H (D(r) + \sigma_w^2 I)^{-1} y\right) \\
&= \arg\max \frac{-1}{2}\left(\sum_{i=1}^{N} \log\left(2\pi(r_i + \sigma_w^2)\right) + \frac{|y_i|^2}{r_i + \sigma_w^2}\right) \\
\implies \hat{r_i} &= \arg\max \frac{-1}{2}\left(\log\left(2\pi(r_i + \sigma_w^2)\right) + \frac{|y_i|^2}{r_i + \sigma_w^2}\right) \\
&= \arg\min \log\left(2\pi(r_i + \sigma_w^2)\right) + \frac{|y_i|^2}{r_i + \sigma_w^2}.
\end{aligned}$$

Taking the differential of the above equation and equating to zero we get the ML estimate

$$\hat{r}_i = |y_i|^2 - \sigma_w^2.$$

The reconstructions for varying degrees of noise is shown in Figure 3 (a-d).



(a) ML ($\sigma_w = 1e^{-2}$) (b) ML ($\sigma_w = 1e^{-1}$) (c) ML ($\sigma_w = 1e^{0}$) (d) ML ($\sigma_w = 1e^{1}$)

(e) PnP-TV ($\sigma_w = 1e^{-4}$) (f) PnP-TV ($\sigma_w = 1e^{-3}$) (g) PnP-TV ($\sigma_w = 1e^{-2}$) (h) PnP-TV ($\sigma_w = 1e^{-1}$)

Figure 2: An example-realization for different inversion-models. The inversions are shown for different std. deviations of $\{1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$.

1.3 Plug and play (PnP) estimate (MAP)

The MAP estiamate for the above problem is given as

$$\begin{aligned} \hat{r} &= \arg\min -\log p(r|y) \\ &= \arg\min -\log p(y|r) - \log p(r). \end{aligned} \tag{1}$$

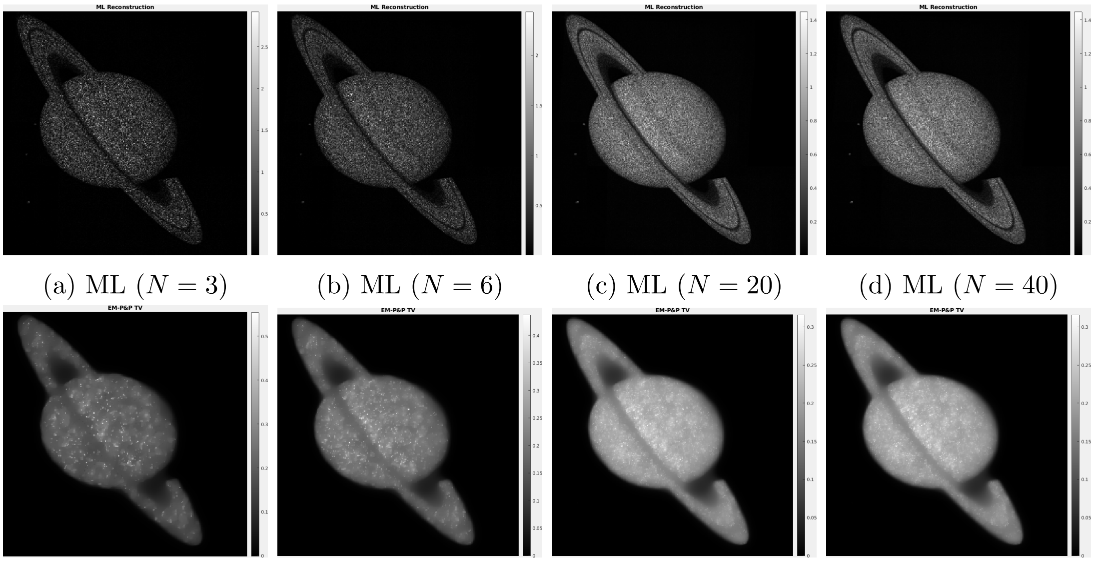
The PnP algorithm [2] decouples the above problem into one where we can apply the forward and prior models in the form of denoisers. Specifically, the PnP algorithm operates by splitting the unknown variable r into two variables r, v and converts the unconstrained optimization problem in (1) into a constrained optimization problem

$$\hat{r} = \underset{r=v}{\arg\min}\ \ell(r) + \beta s(v) \tag{2}$$

where $\beta s(v) = p(v)$ is the prior term and $\ell(r)$ enforces the meausrement constraints on r. Adopting an ADMM approach to solve the above problem results in an algorithm containing two primary operations:

(1) **Inversion operator**

$$F(\tilde{r}, \sigma_\lambda) = \arg\min_r \left\{ \ell(r) + \frac{1}{2\sigma_\lambda^2} \|r - \tilde{r}\|^2 \right\},$$



(a) ML ($N = 3$) (b) ML ($N = 6$) (c) ML ($N = 20$) (d) ML ($N = 40$)

(e) PnP-TV ($N = 3$) (f) PnP-TV ($N = 6$) (g) PnP-TV ($N = 20$) (h) PnP-TV ($N = 40$)

Figure 3: Multi-image plug and play algorithm for: For the image with $\sigma_w = 1e^{-3}$.

where $\tilde{r} = v - u$. The operator F is the proximal mapping of $\ell(r)$ and is equivalent to a MAP estimate of r with a Gaussian prior having a distribution $p(r) \sim N(\tilde{r}, \sigma_\lambda^2 I)$.
Specifically when $\ell(r) = -\log p(y|r)$, the inversion operator is given as

$$\begin{aligned}
F(\tilde{r_i}, \sigma_\lambda) \quad &= \quad \arg\min_{r_i} \left\{ \log(r_i + \sigma_w^2) + \frac{|y_i|^2}{r_i + \sigma_w^2} + \frac{1}{2\sigma_\lambda^2}(r_i - \tilde{r_i})^2 \right\}, \\
&\implies \quad \frac{1}{r_i + \sigma_w^2} - \frac{|y_i|^2}{(r_i + \sigma_w^2)^2} + \frac{1}{\sigma_\lambda^2}(r_i - \tilde{r_i}) = 0, \\
&\implies \quad \sigma_\lambda^2(r_i + \sigma_w^2) - |y_i|^2 + (r_i - \tilde{r_i})(r_i + \sigma_w^2)^2 = 0, \\
&\implies \quad r_i^3 + (-\tilde{r}_i + 2\sigma_w^2)r_i^2 + (-2\tilde{r}_i\sigma_w^2 + \sigma_w^4 + \sigma_\lambda^2)r_i + (\sigma_\lambda^2\sigma_w^2 - \tilde{r}_i\sigma_w^4) = 0.
\end{aligned}$$

Thus the solution of the inversion-operator is simply the root of the cubic-polynomial. We are constrained that the root of the cubic-polynomial is real.

(2) **Denoising operator**

$$H(\tilde{v}, \sigma_n) = \arg\min_{v} \left\{ \frac{1}{2\sigma_n^2} \|v - \tilde{v}\|^2 + s(v) \right\}$$

where $\sigma_n^2 = \beta\sigma_\lambda^2$ and $\tilde{v} = r + u$. The denosing operator H is a proximal mapping of $s(v)$. Mathematically, it is equivalent to a Gaussian denosing operation. Thus, we can replace H with a denoiser which removes noise with a variance σ_n^2. We use the following denoising-operators to use for Gaussian-denosing: (1) TV, (2) BM3D denoiser. The complete PnP algorithm is given in Algorithm 1.

1.3.1 Parameter-tuning

In the following analysis we assume that we know the true-value of σ_w. The parameters to be tuned for the problem are σ_λ, σ_n; which are essentially the tuning parameters for each of the proximal-operators (inversion and denoising).

Algorithm 1 Plug and Play $(y, \sigma_w, \sigma_\lambda, \sigma_n)$

```
v = y, u = 0
Repeat
{
r̃ = v − u
r = F(y, r̃, σ_w, σ_λ)    Inversion-Operator
ṽ = r + u
v = H(ṽ, σ_n)    Denoising-Operator
u = u + (r − v)
}
```

1.4 Regularization by denoising

Recently there has been an alternative approach instead of the ADMM or Plug-and-Play (P^3) approach: regularization by denoising [3]. The key idea of the paper is the regularization proposed in the paper:

$$\rho(x) = \frac{1}{2}x^T[x - f(x)],$$

where $f(x)$ is the denosied version of x. The intuition behind this regulaization is:

- Penalty introduced is the inner-product between the candidate image x and its denosing residual.

A small value for $\rho(x)$ implies that

- Residual is very small, image x is at the fixed point or $x \approx f(x)$.
- Cross-correlation between the image x and its residual is very small or alternatively resembles white-noise or contains no image-like features.

1.5 Properties of the denosier $f(x)$

The RED paper assumes that a denoiser can be assumed to be a psuedo-linear filter of the form

$$f(x) = W(x)x.$$

The following properties are further assumed on the denoiser (**to be completed**).

1.6 Solving the regularized optimization problem

A general inverse-problem is of the form

$$\hat{x} = \arg\min_x \quad E(x) \triangleq \ell(x, y) + \frac{\lambda}{2}x^T[x - f(x)].$$

The gradient of the above expression is of the form

$$\begin{aligned} \nabla_x E(x) &= \nabla_x \ell(y, x) + \frac{\lambda}{2}\nabla_x\{x^T(x - f(x))\} \\ &= \nabla_x \ell(y, x) + \lambda(x - f(x)). \end{aligned}$$

With the above formulation, a gradient-descent based iterative approach would consist of the following iterations:

$$\hat{x}_{k+1} = \hat{x}_k - \mu \nabla_x E(x)|_{\hat{x}_k}.$$

1.7 Regularization by denoising for reflectance-estimation

The regularization by denoising approach for the reflectance-reconstruction problem is given as

$$\begin{aligned} \hat{r} &= \arg\min_r \quad \ell(r, y) + \lambda\rho(r) \\ &= \arg\min_r \quad \sum_{i=1}^{N}\left(\log(2\pi(r_i + \sigma_w^2)) + \frac{|y_i|^2}{r_i + \sigma_w^2}\right) + \frac{\lambda}{2}r^T\{r - f(r)\} \\ &= \arg\min_r \quad E(r). \end{aligned}$$

The RED algorithm for using this procedure is of the form

$$\hat{r}_{k+1} = \hat{r}_k - \mu \nabla_r E(r)|_{\hat{r}_k},$$

where

$$\nabla_r E(r) \quad = \quad \left(\sum_{i=1}^{N} \frac{1}{2\pi(r_i + \sigma_w^2)} - \frac{|y_i|^2}{(r_i + \sigma_w^2)^2} \right) + \lambda(r - f(r)).$$

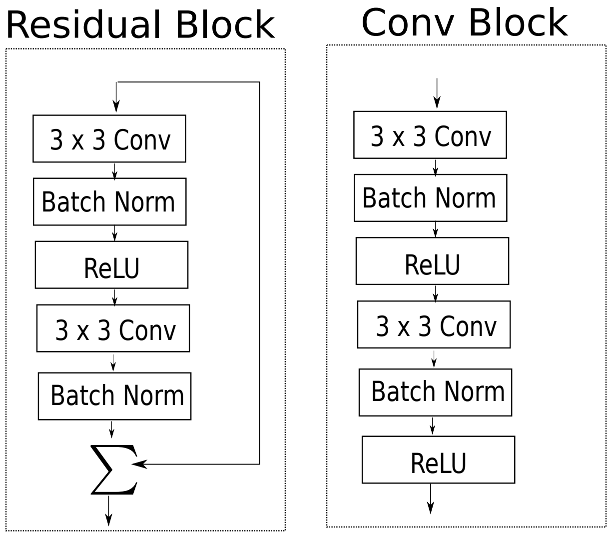


Figure 4: A sample residual and conolutional block.

Layer	Activation-size
Input	$3 \times 224 \times 224$
$64 \times 9 \times 9$ convBlock	$64 \times 224 \times 224$
Residual block 64 filters	$64 \times 224 \times 224$
Residual block 64 filters	$64 \times 224 \times 224$
Residual block 64 filters	$64 \times 224 \times 224$
Residual block 64 filters	$64 \times 224 \times 224$
$64 \times 3 \times 3$ convBlock	$64 \times 224 \times 224$
$3 \times 9 \times 9$ conv2d	$3 \times 224 \times 224$
batchNormalization	3

Table 1: Network architecture used to generate results in Figure 5.

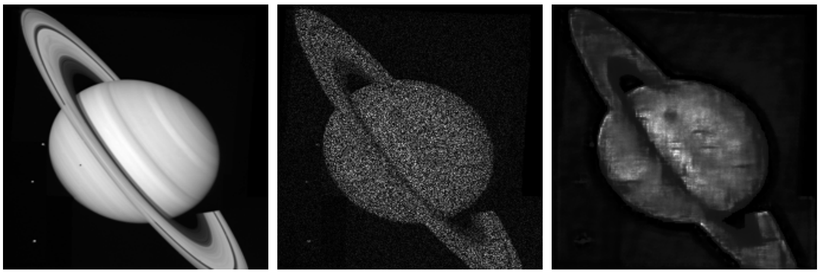
1.8 Direct CNN-approach

Recently we have tried another direct CNN approach to directly learn the mapping from $|y|$ to r. In this approach we learn the mapping from $|y|$ to r by training a convolutional neural network (CNN) on the image pairs from the phase-less measurements to the reflectance of the object. The network architecture that we use follows from the network that has been used for the perceptual-loss extraction paper in SRNET [4].
The architecture that we used is outlined in Tables 1. A residual and convolution block is shown in Figures 4. The results of using testing the trained CNN on the test data is shown in Figure 5.

References

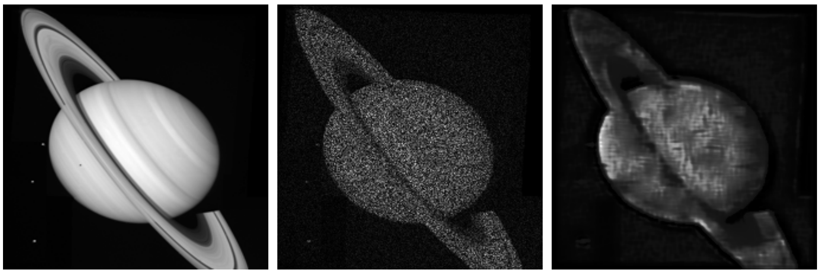
[1] Casey J. Pellizzari et al. *Optically coherent image formation and denoising using a plug and play inversion framework in OSA*.

[2] S. V. Venkatakrishnan et al. *Plug-and-play priors for model based reconstruction in GLOBALSIP*.



(a) GT (b) Input image (c) Output-Img

Figure 5: The learning-rate used to define this network is 0.01 with a step-decay parameter of 0.1 and a step-list being $[10, 25]$. The number of training-epochs is 100.



(a) GT (b) Input image (c) Output-Img

Figure 6: The learning-rate used to define this network is 0.01 with a step-decay parameter of 0.1 and a step-list being $[10, 25]$. The number of training-epochs is 100. We use a deeper-architecture in comparison to the SRNET architecture.

[3] Y. Romano et al. *The little engine that could: Regularization by denoising.*

[4] J. Johnson et al. *Perceptual Losses for Real-Time Style Transfer and Super-Resolution.*