A Design Framework for Metasurface Optics-based Convolutional Neural Networks: Supplemental Document

In this Supplemental Document, we show in detail the derivations for the equations we use in the gradient descent algorithm that performs the phase optimization process described in the main text. Additionally, we also derive the expressions for the gradients we need to compute when performing the end-to-end fine-tuning process where the phase profile is trained at the same time as the suffix layers of the hybrid neural network.

1. PHASE OPTIMIZATION PROCESS

In the phase optimization process, we use a gradient descent method to iteratively update the phase profile Φ in order to minimize a loss function L_Φ given by:

$$L_\Phi(\Phi;\, \mathbf{PSF}_{\text{target}}) = \left|\left|\mathbf{PSF}_{\text{target}} - \left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi\right)\right\}\right|^2\right|\right|^2, \tag{S1}$$

where $||\cdot||^2$ denotes the Frobenius norm of the enclosed numerical array.

For simplicity in our derivation, we use $\mathbf{PSF}_{\text{target}} = Y$ onward. Here, both Y and Φ are three-dimensional numerical arrays, where the first two dimensions correspond to spatial positions and the third dimension corresponds to channels. By using the definition of the Frobenius norm and the (discrete) inverse Fourier transform, we have:

$$L_\Phi(\Phi;\, Y) = \sum_{k,l,c}\left(Y^c_{kl} - \left|\sum_{m}^{M-1}\sum_{n}^{N-1}\exp\left(i\Phi^c_{mn}\right)\exp\left(-2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right|^2\right)^2, \tag{S2}$$

where the c superscripts denote the channel index, and M and N are the number of rows and columns in the phase profile numerical array.

In our phase optimization process, we minimize function L_Φ via gradient descent. This means we iteratively calculate the gradient of L_Φ with respect to Φ, denoted $\nabla_\Phi L_\Phi$, and update the values in Φ as $\Phi = \Phi - \alpha\nabla_\Phi L_\Phi$ with α being a "learning rate". We derived an expression for $\nabla_\Phi L_\Phi$ that allows us to compute this gradient in terms of operations between numerical arrays which can be efficiently carried out by libraries such as NumPy or Tensorflow. We will derive it below.

We calculate the partial derivative of L_Φ with respect to an element in the phase profile array, Φ^c_{mn}. After a few straight-forward simplifications, we find:

$$\begin{aligned}
\frac{\partial L_\Phi}{\partial \Phi^c_{mn}} &= -2\sum_{k,l}\left[\left(Y^c_{kl} - \left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right|^2\right)\frac{\partial}{\partial \Phi^c_{mn}}\left|\sum_{m'}^{M-1}\sum_{n'}^{N-1}\exp\left(i\Phi^c_{m'n'}\right)\exp\left(-2\pi i\left(\frac{m'k}{M}+\frac{n'l}{N}\right)\right)\right|^2\right] \\
\frac{\partial L_\Phi}{\partial \Phi^c_{mn}} &= -2\sum_{k,l}\left[\left(Y^c_{kl} - \left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right|^2\right)\frac{\partial}{\partial \Phi^c_{mn}}\left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right|^2\right]
\end{aligned} \tag{S3}$$

Before we continue further, we first calculate separately the result of the partial derivative that appears on the right side of Eq. (S3).

We have:

$$\begin{aligned}
\frac{\partial}{\partial \Phi^c_{mn}} \left| \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \right|^2 &= \frac{\partial}{\partial \Phi^c_{mn}} \left(\mathrm{Re}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right]\right)^2 + \frac{\partial}{\partial \Phi^c_{mn}} \left(\mathrm{Im}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right]\right)^2 \\
\frac{\partial}{\partial \Phi^c_{mn}} \left| \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \right|^2 &= \frac{\partial}{\partial \Phi^c_{mn}} \left(\sum_{m'}^{M-1}\sum_{n'}^{N-1} \cos\left(\Phi^c_{m'n'} - 2\pi\left(\frac{m'k}{M}+\frac{n'l}{N}\right)\right)\right)^2 \\
&+ \frac{\partial}{\partial \Phi^c_{mn}} \left(\sum_{m'}^{M-1}\sum_{n'}^{N-1} \sin\left(\Phi^c_{m'n'} - 2\pi\left(\frac{m'k}{M}+\frac{n'l}{N}\right)\right)\right)^2 \\
\frac{\partial}{\partial \Phi^c_{mn}} \left| \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \right|^2 &= -2\mathrm{Re}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right] \sin\left(\Phi^c_{mn} - 2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)\right) \\
&+ 2\mathrm{Im}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right] \cos\left(\Phi^c_{mn} - 2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)\right) \\
\frac{\partial}{\partial \Phi^c_{mn}} \left| \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \right|^2 &= 2\mathrm{Re}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right] \mathrm{Im}\left[\exp\left(-i\Phi^c_{mn} + 2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
&+ 2\mathrm{Im}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right] \mathrm{Re}\left[\exp\left(-i\Phi^c_{mn} + 2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
\frac{\partial}{\partial \Phi^c_{mn}} \left| \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \right|^2 &= 2\mathrm{Im}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \exp\left(-i\Phi^c_{mn} + 2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right]
\end{aligned} \tag{S4}$$

Substituting Eq. (S4) into Eq. (S3), we have:

$$\begin{aligned}
\frac{\partial L_\Phi}{\partial \Phi^c_{mn}} &= -4\sum_{k,l}\left(Y^c_{kl} - \left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right|^2\right) \mathrm{Im}\left[\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \exp\left(-i\Phi^c_{mn} + 2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
\frac{\partial L_\Phi}{\partial \Phi^c_{mn}} &= -4\mathrm{Im}\left[\exp\left(-i\Phi^c_{mn}\right)\sum_{k,l}\left(Y^c_{kl} - \left|\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right|^2\right) \mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl} \exp\left(+2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
\frac{\partial L_\Phi}{\partial \Phi^c_{mn}} &= -4\mathrm{Im}\left[\exp\left(-i\Phi^c_{mn}\right)\mathcal{F}_{2D}\left\{\left(Y - \left|\mathcal{F}^{-1}_{2D}\left\{\exp\left(i\Phi\right)\right\}\right|^2\right)\mathcal{F}^{-1}_{2D}\left\{\exp\left(i\Phi\right)\right\}\right\}_{mnc}\right].
\end{aligned} \tag{S5}$$

With Eq. (S5), we can finally express the gradient of L_Φ with respect to Φ as:

$$\nabla_\Phi L_\Phi = -4\mathrm{Im}\left[e^{-i\Phi}\mathcal{F}_{2D}\left\{\left(Y - \left|\mathcal{F}^{-1}_{2D}\left\{e^{i\Phi}\right\}\right|^2\right)\mathcal{F}^{-1}_{2D}\left\{e^{i\Phi}\right\}\right\}\right]. \tag{S6}$$

This is the expression we use to efficiently compute $\nabla_\Phi L_\Phi$ during the gradient descent optimization process to find the phase profile Φ whose yielded PSF approaches the target PSF that encodes the convolutional kernel tensor of the replaced convolutional layer.

2. EXPRESSIONS FOR THE NEURAL NETWORK'S GRADIENTS

Recall from the main text that the output signal yielded by the photodetector in our system is given by:

$$I_{out}(x,y) = \sum_{c}^{C_{in}}\left[\sum_{c'}^{C_{in}}\left(\int \mathbf{SSF}(\omega)\mathbf{PCE}^{c'}(\omega)\mathbf{SPD}^{c}(\omega)d\omega\right)\mathbf{PSF}^{c'}(\eta,\xi)\right] \star I^c_{in}(\eta,\xi), \tag{S7}$$

where $\mathbf{SSF}(\omega)$ is the photodetector's spectral sensitivity function, $\mathbf{PCE}^{c'}(\omega)$ is the polarization conversion efficiency spectrum of the metasurface's meta-elements associated with channel c' of the input, and $\mathbf{SPD}^{c}(\omega)$ is the spectral power distribution of the light that composes channel c of the input.

As noted in the main text, Eq. (S7) can be simplified into the form:

$$I_{out}(x,y) = \sum_{c}^{C_{in}}\left[\sum_{c'}^{C_{in}} A_{c'c}\mathbf{PSF}^{c'}(\eta,\xi)\right] \star I^c_{in}(\eta,\xi), \tag{S8}$$

where $A_{c'c} = \int \mathbf{SSF}(\omega)\mathbf{PCE}^{c'}(\omega)\mathbf{SPD}^{c}(\omega)d\omega$, and we can define a *cross-talk matrix* whose elements are $A_{c'c}$ for the different values of c' and c.

In the ideal case, input channels are generated by optical sources whose spectral power distributions $\mathbf{SPD}^{c}(\omega)$ are delta functions centered at a frequency associated with each RGB channel. Additionally, in this ideal case, meta-elements associated with a given channel have a polarization conversion efficiency spectra $\mathbf{PCE}^{c'}(\omega)$ that is equal to 0 almost everywhere and equal to 1 only at the peak frequency of the spectral power distribution associated with the matching channel of the input image. Finally, in the ideal case, the detector has a sensitivity spectrum $\mathbf{SSF}(\omega)$ that is constant in the whole frequency domain of interest. Under these conditions, the intensity profile of the convolutional layers' output is proportional to the intensity profile of the captured image, which is given by:

$$I_{out}(x, y) = \sum_{c} \mathbf{PSF}^{c}(\eta, \xi) \star I_{in}^{c}(\eta, \xi). \tag{S9}$$

We note that Eq. (S8) has the same functional form as Eq. (S9), as we can write it as

$$I_{out}(x, y) = \sum_{c} B^{c}(\eta, \xi) \star I_{in}^{c}(\eta, \xi), \tag{S10}$$

where B is a tensor with the same dimensions as the multi-channel PSF. This tensor B is defined such that channel c of B, denoted as B^c, is given by:

$$B^{c} = \sum_{c'} A_{c'c}\mathbf{PSF}^{c'}. \tag{S11}$$

Finally, we should note that the cross-talk matrix A is reduced to an identity matrix in the ideal case, where the conditions described above hold true. When that happens, Eq. (S8) and Eq. (S10) are reduced to Eq. (S9).

The equations above are necessary for co-training the phase profile along with the trainable parameters of the suffix layers of the neural network using the backpropagation process. In order to perform the backpropagation process, we need to calculate the gradient of the network's loss function L (not to be confused with L_{Φ} from the previous section) with respect to the phase profile, i.e. $\nabla_{\Phi}L$, and we need to express it in terms of the gradients of the loss function with respect to the parameters of the network's layers. In the following sections, we'll derive the expressions for $\nabla_{\Phi}L$ in both the ideal case (with no cross-talk) and the non-ideal case (with cross-talk). We need these to be expressed in terms of operations that can be efficiently performed between numerical arrays, using the built-in optimized algorithms of libraries such as Numpy and Tensorflow.

Ideal case

The math for the ideal case will serve as a baseline for that of the non-ideal case, and because of that, we'll start with the former one. In our pipeline, the phase profile array Φ (three-dimensional, with the third dimension corresponding to channels) is used to yield a $\mathbf{PSF}$ (also a three-dimensional array). Then, from the yielded $\mathbf{PSF}$ we extract the convolutional kernels W of the first convolutional layer. These convolutional kernels will be the ones that are used to process the input of the network as it goes through the first convolutional layer. As such, when the network is trained using a library such as Tensorflow via the backpropagation process, the program will have to calculate the gradient of the loss function with respect to W, i.e. $\nabla_{W}L$.

Under this scheme, L depends on W, which in turn depends on $\mathbf{PSF}$, which itself depends on Φ. Because of this, due to the chain rule of derivatives, the gradient of L with respect to Φ can be expressed as:

$$\nabla_{\Phi}L = \sum_{\mathbf{PSF}} \frac{\partial L}{\partial \mathbf{PSF}} \nabla_{\Phi}\mathbf{PSF}, \tag{S12}$$

where $\frac{\partial L}{\partial \mathbf{PSF}}$ are the elements of $\nabla_{\mathbf{PSF}}L$, which can be expressed as:

$$\nabla_{\mathbf{PSF}}L = \sum_{W} \frac{\partial L}{\partial W} \nabla_{\mathbf{PSF}}W, \tag{S13}$$

where $\frac{\partial L}{\partial W}$ are the elements of $\nabla_{W}L$, which is calculated by the program's backpropagation algorithm. Because of the above, in order to calculate $\nabla_{\Phi}L$, we first need to find a way to express and efficiently compute $\nabla_{\mathbf{PSF}}L$ and the $\nabla_{\Phi}\mathbf{PSF}$.

Let's begin by finding a $\nabla_{\mathbf{PSF}} L$. Since each element of the convolutional kernel tensor W is obtained by adding and subtracting specific elements of the **PSF** tensor, each of the $\nabla_{\mathbf{PSF}} W$ tensors in the sum in Eq. (S13) will be sparse. For a given W (here, W denotes an element of the convolutional kernel tensor), the corresponding $\nabla_{\mathbf{PSF}} W$ will be a tensor with the same dimensions as **PSF** that will be non-zero only in the index positions of the elements of the **PSF** tensor that are involved in the computation of that element W of the conv-weight tensor. The non-zero values will be either 1 or -1. If during the phase optimization process Φ was upscaled by a scale factor n (with respect to the target PSF array that is used to encode the weights obtained from the training pipeline's Step 1), then two tiles of $n \times n \times 1$ pixels of **PSF** are used to calculate one element of tensor W. One tile is subtracted from the other, and then the result is averaged to yield the numerical value of element W. Because of this, for a given element W, the corresponding $\nabla_{\mathbf{PSF}} W$ will be a sparse array where there will be a tile of $n \times n \times 1$ pixels with values $+\frac{1}{n^2}$ and another such tile with values $-\frac{1}{n^2}$. As such, each term $\frac{\partial L}{\partial W} \nabla_{\mathbf{PSF}} W$ in the sum of Eq. (S13) will simply be an sparse array with the same dimensions as **PSF** that contains a tile of $n \times n \times 1$ pixels whose values are all $+\frac{1}{n^2}\frac{\partial L}{\partial W}$ and another such tile whose values are all $-\frac{1}{n^2}\frac{\partial L}{\partial W}$. This way, when performing a sum over elements W of all this terms, the result $\nabla_{\mathbf{PSF}} L$ will simply be a tiled version of $\nabla_W L$. As such, we have now found a way to efficiently compute $\nabla_{\mathbf{PSF}} L$ in terms of $\nabla_W L$: We just need to take $\nabla_W L$ and perform a tiling process similar to the one that we use during step 2 of the training pipeline, where we obtain a target PSF by tiling the convolutional kernel tensor W we obtained from step 1.

Now, let's look at how to calculate the $\nabla_\Phi \mathbf{PSF}$ that appear in Eq. (S12), i.e. the gradients of each pixel in the **PSF** array with respect to the phase profile. First, let's look at the expression for element (k, l, c) of the **PSF** array, taking into account how it depends on Φ as $\mathbf{PSF}(\Phi) = \left|\mathcal{F}^{-1}\left\{e^{i\Phi}\right\}\right|^2$:

$$\mathbf{PSF}_{k,l,c} = \left|\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \exp\left(i\Phi^c_{mn}\right)\exp\left(-2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right|^2, \tag{S14}$$

where M and N are the number of pixels in the vertical (rows) and horizontal (columns) spatial dimensions in the Φ array, respectively. Given this, we can now calculate $\frac{\partial \mathbf{PSF}_{klc}}{\partial \Phi^{c'}_{m'n'}}$:

$$\begin{aligned}\frac{\partial \mathbf{PSF}_{klc}}{\partial \Phi^{c'}_{m'n'}} &= \left(\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \exp\left(i\Phi^c_{mn}\right)\exp\left(-2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right)\frac{\partial}{\partial \Phi^{c'}_{m'n'}}\left(\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \exp\left(-i\Phi^c_{mn}\right)\exp\left(+2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right)\\
&\quad+\left(\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \exp\left(-i\Phi^c_{mn}\right)\exp\left(+2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right)\frac{\partial}{\partial \Phi^{c'}_{m'n'}}\left(\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \exp\left(i\Phi^c_{mn}\right)\exp\left(-2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right)\\
\frac{\partial \mathbf{PSF}_{klc}}{\partial \Phi^{c'}_{m'n'}} &= -i\delta_{cc'}\left(\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right)\left(\exp\left(i\Phi^{c'}_{m'n'}-2\pi i\left(\frac{m'k}{M}+\frac{n'l}{N}\right)\right)\right)^*\\
&\quad+i\delta_{cc'}\left(\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\right)^*\left(\exp\left(i\Phi^{c'}_{m'n'}-2\pi i\left(\frac{m'k}{M}+\frac{n'l}{N}\right)\right)\right),\end{aligned} \tag{S15}$$

where $\delta_{cc'}$ is a Kronecker delta between channel index values c and c' and the * asterisk denotes complex conjugation. The expression in Eq. (S15) can be simplified if we take into account that (it can be shown that) for any two complex numbers x and y, we have that $-xy^* + x^*y = -2i\mathrm{Im}\left[xy^*\right]$, where Im denotes imaginary part of a complex number. This way, we finally have:

$$\frac{\partial \mathbf{PSF}_{klc}}{\partial \Phi^{c'}_{mn}} = 2\mathrm{Im}\left[\exp\left(-i\Phi^{c'}_{mn}\right)\mathcal{F}^{-1}\left\{\exp\left(i\Phi^c\right)\right\}_{kl}\exp\left(2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right]\delta_{cc'}. \tag{S16}$$

With this, we can calculate the (m, n, c') element of $\nabla_\Phi L$ using Eq. (S12) and Eq. (S16) as:

$$\begin{aligned}
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{\partial \mathbf{PSF}_{klc}} \frac{\partial \mathbf{PSF}_{klc}}{\partial \Phi^{c'}_{mn}} \\
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{\partial \mathbf{PSF}_{klc}} 2\mathrm{Im}\left[\exp\left(-i\Phi^{c'}_{mn}\right)\mathcal{F}^{-1}\left\{\exp\left(i\Phi^{c}\right)\right\}_{kl}\exp\left(2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right]\delta_{cc'} \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im}\left[\sum_{k,l} \frac{\partial L}{\partial \mathbf{PSF}_{klc'}} \exp\left(-i\Phi^{c'}_{mn}\right)\mathcal{F}^{-1}\left\{\exp\left(i\Phi^{c'}\right)\right\}_{kl}\exp\left(2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im}\left[\left(\exp\left(-i\Phi\right)\right)_{mnc'}\sum_{k,l} \frac{\partial L}{\partial \mathbf{PSF}_{klc'}} \mathcal{F}^{-1}_{2\mathrm{D}}\left\{\exp\left(i\Phi\right)\right\}_{klc'}\exp\left(2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im}\left[\left(\exp\left(-i\Phi\right)\right)_{mnc'}\sum_{k,l}\left(\left(\nabla_{\mathbf{PSF}} L\right)\left(\mathcal{F}^{-1}_{2\mathrm{D}}\left\{\exp\left(i\Phi\right)\right\}\right)\right)_{klc'}\exp\left(2\pi i\left(\frac{mk}{M}+\frac{nl}{N}\right)\right)\right] \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im}\left[\left(\exp\left(-i\Phi\right)\right)_{mnc'}\mathcal{F}_{2\mathrm{D}}\left\{\left(\nabla_{\mathbf{PSF}} L\right)\left(\mathcal{F}^{-1}_{2\mathrm{D}}\left\{\exp\left(i\Phi\right)\right\}\right)\right\}_{mnc'}\right],
\end{aligned} \tag{S17}$$

thus, we can finally calculate $\nabla_\Phi L$ in terms of operations between known numerical arrays as:

$$\nabla_\Phi L = 2\mathrm{Im}\left[e^{-i\Phi}\mathcal{F}_{2\mathrm{D}}\left\{\left(\nabla_{\mathbf{PSF}} L\right)\left(\mathcal{F}^{-1}_{2\mathrm{D}}\left\{e^{i\Phi}\right\}\right)\right\}\right] \tag{S18}$$

Non-ideal case

The non-ideal case is very similar to the ideal one. However, the main difference mathematically is that we don't extract the elements of W from the **PSF** tensor, but from the B tensor defined in Eq. (S11) instead. As such, we can write:

$$\nabla_\Phi L = \sum_B \frac{\partial L}{\partial B}\nabla_\Phi B, \tag{S19}$$

where $\frac{\partial L}{\partial B}$ are the elements of $\nabla_B L$, which can be expressed as:

$$\nabla_B L = \sum_W \frac{\partial L}{\partial W}\nabla_B W, \tag{S20}$$

where $\frac{\partial L}{\partial W}$ are the elements of $\nabla_W L$, which is calculated by the program's backpropagation algorithm.

In the non-ideal case, $\nabla_B L$ can be calculated in the same way as $\nabla_{\mathbf{PSF}} L$ was calculated in the ideal case. This is because in the non-ideal case, the convolutional kernel tensor W is built from the B tensor in the same way as W was built from the **PSF** tensor in the ideal case. Because of this, we already know what the $\frac{\partial L}{\partial B}$ that appear in the sum of Eq. (S19) look like, and we know how to efficiently compute $\nabla_B L$. As such, we can now proceed to find an expression for the (m, n, c')

element of $\nabla_\Phi L$. We have:

$$\begin{aligned}
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{B_{klc}} \frac{\partial B_{klc}}{\partial \Phi^{c'}_{mn}} \\
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{B_{klc}} \frac{\partial}{\partial \Phi^{c'}_{mn}} \left(\sum_{c''} A_{c''c} \mathbf{PSF}_{klc''} \right) \\
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{B_{klc}} \sum_{c''} A_{c''c} \frac{\partial \mathbf{PSF}_{klc''}}{\partial \Phi^{c'}_{mn}} \\
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{B_{klc}} A_{c'c} \frac{\partial \mathbf{PSF}_{klc'}}{\partial \Phi^{c'}_{mn}} \\
(\nabla_\Phi L)_{mnc'} &= \sum_{k,l,c} \frac{\partial L}{B_{klc}} A_{c'c} 2\mathrm{Im} \left[\left(e^{-i\Phi} \right)_{mnc'} \mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\}_{klc'} \exp\left(2\pi i \left(\frac{mk}{M} + \frac{nl}{N} \right) \right) \right] \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im} \left[\sum_{k,l,c} A_{c'c} \left(\nabla_B L \right)_{klc} \left(e^{-i\Phi} \right)_{mnc'} \mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\}_{klc'} \exp\left(2\pi i \left(\frac{mk}{M} + \frac{nl}{N} \right) \right) \right] \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im} \left[\left(e^{-i\Phi} \right)_{mnc'} \sum_{k,l} \left(\mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\}_{klc'} \exp\left(2\pi i \left(\frac{mk}{M} + \frac{nl}{N} \right) \right) \right) \left(\sum_{c} A_{c'c} \left(\nabla_B L \right)_{klc} \right) \right].
\end{aligned} \tag{S21}$$

Let's define a tensor H such that $H^{c'}_{kl} = \sum_c A_{c'c} \left(\nabla_B L \right)_{klc}$. With that, we can further simplify Eq. (S21) as:

$$\begin{aligned}
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im} \left[\left(e^{-i\Phi} \right)_{mnc'} \sum_{k,l} \left(\mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\} H \right)_{klc'} \exp\left(2\pi i \left(\frac{mk}{M} + \frac{nl}{N} \right) \right) \right], \\
(\nabla_\Phi L)_{mnc'} &= 2\mathrm{Im} \left[\left(e^{-i\Phi} \right)_{mnc'} \mathcal{F}_{\mathrm{2D}} \left\{ H \mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\} \right\}_{mnc'} \right]
\end{aligned} \tag{S22}$$

thus, we can finally calculate $\nabla_\Phi L$ in terms of operations between numerical arrays as:

$$\nabla_\Phi L = 2\mathrm{Im} \left[e^{-i\Phi} \mathcal{F}_{\mathrm{2D}} \left\{ H \mathcal{F}^{-1}_{\mathrm{2D}} \left\{ e^{i\Phi} \right\} \right\} \right], \tag{S23}$$

where H is a tensor defined such that its channel c', denoted $H^{c'}$, is given by $H^{c'} = \sum_c A_{c'c} \left(\nabla_B L \right)^c$, with $\left(\nabla_B L \right)^c$ being channel c of $\nabla_B L$.