OBD notes

There is a question about why Eq. (8) in Hirsch et al. (2011) takes the form that it does. It turns out that the precise answer is somewhat involved, but the key motivation is as follows. The overall goal of the OBD procedure is to arrive at an estimate of some underlying image $\boldsymbol{x}$ given a bunch of observed images $\boldsymbol{y}_t$ that have each been convolved with an unknown PSF $\boldsymbol{f}_t$. The mechanism of the procedure is to update $\boldsymbol{x}$ and solve for $\boldsymbol{f}_t$ at every t in an "online" fashion, using a gradient descent method. The different treatments of $\boldsymbol{x}$ and $\boldsymbol{f}_t$ – i.e., the fact that one is simply updated (meaning it experiences only a "small" change, in some appropriately-defined way) while the other is arbitrarily modified – are ultimately what lead to the form of the equation in question.

Now diving a bit more into the specifics. As a reminder, the OBD strategy contains two steps at every t. The first step is to solve for the PSF $\boldsymbol{f}_t$ that results in the minimum loss ℓ_t,

$$\ell_t = \min_{\boldsymbol{f}_t > 0} ||\boldsymbol{y}_t - \boldsymbol{f}_t * \boldsymbol{x}_t||^2 \qquad \longrightarrow \qquad \boldsymbol{f}_t = \operatorname*{argmin}_{\boldsymbol{f} > 0} ||\boldsymbol{y}_t - \boldsymbol{f} * \boldsymbol{x}_t||^2,$$

given the observed $\boldsymbol{y}_t$ and the current best-guess underlying image $\boldsymbol{x}_t$. Since only $\boldsymbol{f}_t$ can vary here, this is a convex problem that can be solved using any number of non-negative least-squares techniques. Note that the authors sometimes use a matrix notation to represent the convolution operation, such that

$$\boldsymbol{f} * \boldsymbol{x} = \boldsymbol{F}\boldsymbol{x},$$

for an appropriately-defined matrix $\boldsymbol{F}$.

The second step at iteration t is to update $\boldsymbol{x}_t$. Because the authors are worried about overfitting, their main priority is simply to ensure that whatever update is applied to $\boldsymbol{x}_t$ produces a loss that is smaller than the loss produced from the previous iteration. For reasons that I haven't been able to track down, the authors choose to use the following function as their effective loss during this second step:

$$g_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}_t) = \boldsymbol{y}_t^\top \boldsymbol{y}_t - 2\boldsymbol{y}_t^\top \boldsymbol{F}_t \boldsymbol{x}_{t+1} + \boldsymbol{x}_t^\top \boldsymbol{F}_t^\top \boldsymbol{F}_t \left(\frac{\boldsymbol{x}_{t+1} \odot \boldsymbol{x}_{t+1}}{\boldsymbol{x}_t} \right).$$

Here, $\odot$ denotes a Hadamard product and the division is understood to be taken element-wise. This function has a couple of properties that are necessary for guaranteeing that it decreases, specifically

$$g_t(\boldsymbol{x}_t, \boldsymbol{x}_t) \geq g_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}_t) \geq g_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}_{t+1})$$

and

$$g_t(\boldsymbol{x}_t, \boldsymbol{x}_t) = ||\boldsymbol{y}_t - \boldsymbol{F}_t \boldsymbol{x}_t||^2.$$

Anyhow, taking this new loss function as a given, the value of $\boldsymbol{x}_{t+1}$ that minimizes $g_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}_t)$ can be determined by taking a gradient and setting it equal to zero; i.e., solving $\nabla_{\boldsymbol{x}_{t+1}} g_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}_t) = 0$ for $\boldsymbol{x}_{t+1}$. Doing so yields the update rule

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t \odot \frac{\boldsymbol{F}_t^\top \boldsymbol{y}_t}{\boldsymbol{F}_t^\top \boldsymbol{F}_t \boldsymbol{x}_t},$$

which is precisely Eq. (8) in the paper.