GroupCDL: Interpretable Denoising and Compressed Sensing MRI via Learned Group-Sparsity and Circulant Attention

Nikola Janjušević, Amirhossein Khalilian-Gourtani, Adeen Flinker, Li Feng, Yao Wang

I. Supplementary Material

A. Notation in Detail

In this manuscript, we consider 2D multi-channel images as vectors. Specifically, image $\boldsymbol{x} \in \mathbb{R}^{NC}$, with N pixels and C channels is formed by stacking the vectorized 2D channels in a column vector,

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_C \end{bmatrix}. \tag{1}$$

Clearly as convolution is a linear operator, it may be expressed as a matrix multiplication. For images, this matrix representation has the form of a block circulant/toeplitz matrix with toeplitz blocks (BCCB/BTTB) for circular/zero padding, respectively. Consider an array of 2D filters $\boldsymbol{d}_{ij} \in \mathbb{R}^{N_f \times N_f}$, and (with some abuse of notation) their corresponding convolution matrices $\boldsymbol{D}_{ij}$, $1 \leq i \leq M$, $1 \leq j \leq C$. With this notation, standard convolution layer of deep-learning (modulo bias vector) is expressed simply as multiplication with a block matrix,

$$\boldsymbol{y}_i = \sum_{j=1}^{C} \boldsymbol{d}_{ij} * \boldsymbol{x}_j = \sum_{j=1}^{C} \boldsymbol{D}_{ij} \boldsymbol{x}_c, \quad 1 \leq i \leq M \tag{2}$$

$$\begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \\ \vdots \\ \boldsymbol{y}_M \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_{11} & \boldsymbol{D}_{12} & \cdots & \boldsymbol{D}_{1C} \\ \boldsymbol{D}_{21} & \boldsymbol{D}_{22} & \cdots & \boldsymbol{D}_{2C} \\ \vdots & \vdots & \ddots & \cdots \\ \boldsymbol{D}_{M1} & \boldsymbol{D}_{M2} & \cdots & \boldsymbol{D}_{MC} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_C \end{bmatrix} \tag{3}$$

$$\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}. \tag{4}$$

Clearly, operators which we can think of in terms of being sperable or having sub-operators over channels can be represented as block matrices. For example, we can express applying the same matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ to every channel as,

$$(I_C \otimes \boldsymbol{A})\boldsymbol{x} = \begin{bmatrix} \boldsymbol{A} & & & \\ & \boldsymbol{A} & & \\ & & \ddots & \\ & & & \boldsymbol{A} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_C \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}\boldsymbol{x}_1 \\ \boldsymbol{A}\boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{A}\boldsymbol{x}_C \end{bmatrix} \tag{5}$$

and we say that $(I_C \otimes \boldsymbol{A})$ is seperable over channels. Now suppose we have a matrix $(w_{ij}) = \boldsymbol{W} \in \mathbb{R}^{M \times C}$ which operates on image pixels $\boldsymbol{x}[i] \in \mathbb{R}^C$, $1 \leq i \leq N$, i.e. $\boldsymbol{y}[i] = \boldsymbol{W}\boldsymbol{x}[i] \in \mathbb{R}^M$. We can convieniently express this as,

$$(\boldsymbol{W} \otimes I_N)\boldsymbol{x} = \begin{bmatrix} w_{11}I_N & w_{12}I_N & \cdots & w_{1C}I_N \\ w_{21}I_N & w_{22}I_N & \cdots & w_{2C}I_N \\ \vdots & \vdots & \ddots & \cdots \\ w_{M1}I_N & w_{M2}I_N & \cdots & w_{MC}I_N \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_C \end{bmatrix} \tag{6}$$

Note that $(\boldsymbol{W} \otimes I_N)$ also corresponds to a 1×1 kernel C to M channel conv operator.

B. Circulant-Sparse Attention Back-Propagation Rules

We derive the reverse-mode back-propagation rules of Circulant-Sparse Attention (Algorithms 1, 2) in accordance with [1] and as extended for complex arithmetic in [2]. Consider a computation with intermediate variable $\boldsymbol{X}$. We denote the derivate of $\boldsymbol{X}$ with respect to some input (preceeding) variable t as $\frac{d\boldsymbol{X}}{dt} = \dot{\boldsymbol{X}}$. We denote the derivative of some output variable s with respect to the elements of $\boldsymbol{X}$ as $\mathrm{d}\boldsymbol{X}$. For example, given a function $\boldsymbol{C} = f(\boldsymbol{A}, \boldsymbol{B})$, calculus's total derivate rule says,

$$\dot{\boldsymbol{C}} = \frac{\partial f}{\partial \boldsymbol{A}} \dot{\boldsymbol{A}} + \frac{\partial f}{\partial \boldsymbol{B}} \dot{\boldsymbol{B}}.$$

For reverse-mode automatic differentiation, we wish to work backwards from some output variable (s). Consider this intermediate computation is used to produce s. We can then express the derivative of s with respect to t in terms of the intermediate computation as,

$$\frac{ds}{dt} = \sum_{ij} \frac{ds}{d\boldsymbol{C}_{ij}} \frac{d\boldsymbol{C}_{ij}}{dt} = \mathbf{tr}\left(\mathrm{d}\boldsymbol{C}^T \dot{\boldsymbol{C}}\right) = \langle \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{C}} \rangle.$$

where $\mathbf{tr}(\boldsymbol{X}) = \sum_{ii} \boldsymbol{X}_{ii}$ for square matrices. Combining the previous two expressions, we arrive at the identity

$$\begin{aligned} \langle \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{C}} \rangle &= \left\langle \mathrm{d}\boldsymbol{C}, \frac{\partial f}{\partial \boldsymbol{A}} \dot{\boldsymbol{A}} \right\rangle + \left\langle \mathrm{d}\boldsymbol{C}, \frac{\partial f}{\partial \boldsymbol{A}} \dot{\boldsymbol{B}} \right\rangle \\ &= \left\langle \left(\frac{\partial f}{\partial \boldsymbol{A}}\right)^T \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{A}} \right\rangle + \left\langle \left(\frac{\partial f}{\partial \boldsymbol{B}}\right)^T \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{B}} \right\rangle \\ \langle \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{C}} \rangle &= \left\langle \mathrm{d}\boldsymbol{A}, \dot{\boldsymbol{A}} \right\rangle + \left\langle \mathrm{d}\boldsymbol{B}, \dot{\boldsymbol{B}} \right\rangle. \end{aligned} \tag{7}$$

This allows us to implement reverse-mode differentiation by computing an operation's input perturbations ($\mathrm{d}\boldsymbol{A}$, $\mathrm{d}\boldsymbol{B}$) given output perturbation $\mathrm{d}\boldsymbol{C}$. As we go backwards along the chain of computations (which define a neural network), ($\mathrm{d}\boldsymbol{A}$, $\mathrm{d}\boldsymbol{B}$) may then become the output perturbation for the next operation/layer backwards.

For differentiation purposes, we consider complex variables as numbers in $\mathbb{R}^2$. Therefore, the trace innner product above may be generalized with the real trace-inner product, which is equivalent to computing said inner product over real and imaginary components sepearately (ex. $\mathrm{Re}\langle \mathrm{d}\boldsymbol{C}, \dot{\boldsymbol{C}}\rangle$).

Now, we derive the reverse-mode rules for the Circulant-sparse similarity/attention algorithms by first computing the forward mode rules (ex. $\dot{\boldsymbol{S}} = \dots$) and then manipulating the expression to be in the form of the identity (7) to determine input perturbations given an output perturbation. For ease of derivation, we reshape our vectorized signals into matrices channel-wise, i.e. $\boldsymbol{x} \in \mathbb{C}^{NC} \Leftrightarrow \boldsymbol{X} \in \mathbb{C}^{N\times C}$.

Algorithm 1: Circulant-Sparse Dot-Sim $\mathcal{O}(QW^2M)$

```
function CircDotSim($\boldsymbol{k} \in \mathbb{C}^{QM}$, $\boldsymbol{q} \in \mathbb{C}^{QM}$; $W \in \mathbb{Z}_+$):
    $\boldsymbol{S}_{ij} = \begin{cases} \mathrm{Re}\{\boldsymbol{q}[j]^H \boldsymbol{k}[i]\}, & \|\vec{i} - \vec{j}\|_\infty \leq \frac{1}{2}W \\ -\infty, & \text{otherwise} \end{cases} \forall\ (i,j) \in [1,Q]^2$
    function bwd($\mathrm{d}\boldsymbol{S} \in \mathbb{B}_W^{Q\times Q}$):
        $\mathrm{d}\boldsymbol{k} = \underline{\mathrm{d}\boldsymbol{S}}\boldsymbol{q}$
        $\mathrm{d}\boldsymbol{q} = \underline{\mathrm{d}\boldsymbol{S}}^T\boldsymbol{k}$
        return $\mathrm{d}\boldsymbol{k}$, $\mathrm{d}\boldsymbol{q}$
    return $\boldsymbol{S}$, bwd
```

As a warm-up, we begin with deriving rules for Circulant-sparse dot similarity (Supp Alg. 1), as follows. Let $\boldsymbol{M} \in \{0,1\}^{N\times N}$ be the BCCB sparsity mask for window-size W. Then,

$$\boldsymbol{S} = \mathbf{CircDotSim}(\boldsymbol{k},\, \boldsymbol{q};\, W) \tag{8}$$

$$\boldsymbol{S} = \boldsymbol{M} \circ \mathrm{Re}\{\boldsymbol{K}\boldsymbol{Q}^H\} - \infty(1 - \boldsymbol{M}) \tag{9}$$

$$\dot{\boldsymbol{S}} = \boldsymbol{M} \circ \mathrm{Re}\{\dot{\boldsymbol{K}}\boldsymbol{Q}^H + \boldsymbol{K}\dot{\boldsymbol{Q}}^H\} \tag{10}$$

$$\begin{aligned} \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,, \dot{\boldsymbol{S}}\rangle &= \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,, \boldsymbol{M} \circ (\dot{\boldsymbol{K}}\boldsymbol{Q}^H + \boldsymbol{K}\dot{\boldsymbol{Q}}^H)\rangle & (11)\\ &= \mathrm{Re}\langle \boldsymbol{M} \circ \mathrm{d}\boldsymbol{S}\,, \dot{\boldsymbol{K}}\boldsymbol{Q}^H + \boldsymbol{K}\dot{\boldsymbol{Q}}^H\rangle & (12)\\ &= \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,\boldsymbol{Q}, \dot{\boldsymbol{K}}\rangle + \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}^H\,\boldsymbol{K}, \dot{\boldsymbol{Q}}\rangle & (13) \end{aligned}$$

$$\mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,, \dot{\boldsymbol{S}}\rangle = \mathrm{Re}\langle \mathrm{d}\boldsymbol{K}\,, \dot{\boldsymbol{K}}\rangle + \mathrm{Re}\langle \mathrm{d}\boldsymbol{Q}\,, \dot{\boldsymbol{Q}}\rangle. \tag{14}$$

As $\mathrm{d}\boldsymbol{S} \in \mathbb{B}_W^{N\times N}$ is real, we simplify the rules as $\mathrm{d}\boldsymbol{K} = \mathrm{d}\boldsymbol{S}\,\boldsymbol{Q}$, $\mathrm{d}\boldsymbol{Q} = \mathrm{d}\boldsymbol{S}^T\,\boldsymbol{K}$.

Circulant-sparse distance similarity (Alg. 1) rules are similarly derived as follows, where $\mathbf{1}$ is the $N \times C$ matrix of all ones, and $|\cdot|^2$ is taken element-wise,

$$\boldsymbol{S} = \mathbf{CircDistSim}(\boldsymbol{k},\, \boldsymbol{q};\, W) \tag{15}$$

$$\begin{aligned} \boldsymbol{S} = &-\tfrac{1}{2}\boldsymbol{M} \circ (|\boldsymbol{K}|^2\mathbf{1}^T - 2\,\mathrm{Re}\{\boldsymbol{K}\boldsymbol{Q}^H\} + \mathbf{1}(|\boldsymbol{Q}|^2)^T)\\ &- \infty(1 - \boldsymbol{M}) \end{aligned} \tag{16}$$

$$\begin{aligned} \dot{\boldsymbol{S}} = &\ \boldsymbol{M} \circ (\mathrm{Re}\{\dot{\boldsymbol{K}}\boldsymbol{Q}^H\} - (\boldsymbol{K} \circ \dot{\boldsymbol{K}})\mathbf{1}^T) +\\ &\ \boldsymbol{M} \circ (\mathrm{Re}\{\boldsymbol{K}\dot{\boldsymbol{Q}}^H\} - \mathbf{1}(\boldsymbol{Q} \circ \dot{\boldsymbol{Q}})^T) \end{aligned} \tag{17}$$

$$\begin{aligned} \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,, \dot{\boldsymbol{S}}\rangle = &\ \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,\boldsymbol{Q} - (\mathrm{d}\boldsymbol{S}\,\mathbf{1}) \circ \boldsymbol{K}, \dot{\boldsymbol{K}}\rangle +\\ &\ \mathrm{Re}\langle \mathrm{d}\boldsymbol{S}^T\,\boldsymbol{K} - (\mathrm{d}\boldsymbol{S}^T\,\mathbf{1}) \circ \boldsymbol{Q}, \dot{\boldsymbol{Q}}\rangle \end{aligned} \tag{18}$$

$$\mathrm{Re}\langle \mathrm{d}\boldsymbol{S}\,, \dot{\boldsymbol{S}}\rangle = \mathrm{Re}\langle \mathrm{d}\boldsymbol{K}\,, \dot{\boldsymbol{K}}\rangle + \mathrm{Re}\langle \mathrm{d}\boldsymbol{Q}\,, \dot{\boldsymbol{Q}}\rangle. \tag{19}$$

Circulant-sparse attention (Alg. 2) rules are derived as follows,

$$\boldsymbol{Y} = \boldsymbol{\Gamma}\boldsymbol{X} \tag{20}$$

$$\dot{\boldsymbol{Y}} = \dot{\boldsymbol{\Gamma}}\boldsymbol{X} + \boldsymbol{\Gamma}\dot{\boldsymbol{X}} \tag{21}$$

$$\begin{aligned} \mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{Y}}\rangle &= \mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{\Gamma}}\boldsymbol{X}\rangle + \mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}, \boldsymbol{\Gamma}\dot{\boldsymbol{X}}\rangle & (22)\\ &= \mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}\boldsymbol{X}^H, \dot{\boldsymbol{\Gamma}}\rangle + \mathrm{Re}\langle \boldsymbol{\Gamma}^H\mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{X}}\rangle & (23) \end{aligned}$$

As shown, $\mathrm{d}\boldsymbol{Y}\boldsymbol{X}^H$ is a massive dense attention matrix. However, we may make use of the fact that $\dot{\boldsymbol{\Gamma}} \in \mathbb{B}_W^{N\times N}$ is real and has a BCCB sparsity pattern, i.e. $\dot{\boldsymbol{\Gamma}} = \boldsymbol{M} \circ \dot{\boldsymbol{\Gamma}}$. Therefore we rewrite

$$\mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{Y}}\rangle = \mathrm{Re}\langle \boldsymbol{M} \circ (\mathrm{d}\boldsymbol{Y}\boldsymbol{X}^H), \dot{\boldsymbol{\Gamma}}\rangle + \mathrm{Re}\langle \boldsymbol{\Gamma}^T\mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{X}}\rangle \tag{24}$$

$$\mathrm{Re}\langle \mathrm{d}\boldsymbol{Y}, \dot{\boldsymbol{Y}}\rangle = \mathrm{Re}\langle \mathrm{d}\boldsymbol{\Gamma}, \dot{\boldsymbol{\Gamma}}\rangle + \mathrm{Re}\langle \mathrm{d}\boldsymbol{X}, \dot{\boldsymbol{X}}\rangle \tag{25}$$

which is equivalent to calling the circulant dot simiarity function, $\mathrm{d}\boldsymbol{\Gamma} = \mathbf{CircDotSim}(\mathrm{d}\boldsymbol{Y},\, \boldsymbol{X};\, W)$ and $\mathrm{d}\boldsymbol{X} = \boldsymbol{\Gamma}^T\mathrm{d}\boldsymbol{Y}$.

C. Sliding vs. Overlapping Window Nonlocal Processing

From Figure 1a, it is clear that the patch-based dense attention strategy (used by NLRN [3] and GroupSC [4]) is able to achieve a denoising to speed trade-off by reducing the amount of overlap between windows, i.e. increasing window-stride (s_w). CircAtt can also achieve a similar performance-speed trade-off by instead reducing the nonlocal window-size (W) during inference.

Figure 1 plots the denoising performance vs. inference time trade-off attainable by GroupCDL under the CircAtt strategy and the patch-based dense attention strategy. The CircAtt: $W^{\text{train}} = 35$, and PbDA curves show a single GroupCDL model (trained with a nonlocal window-size $W^{\text{train}} = 35$) under CircAtt inference with varying nonlocal window-size W^{test}, and patch-based dense attention inference with varying window-stride s_w, respectively. Each point in the CircAtt: $W^{\text{train}} = W^{\text{test}}$ curve shows the performance of a GroupCDL model trained with a different non-local window-size and performing CircAtt inference with their respective training window-size.

First, we observe that CircAtt consistently out-performs patch-based dense attention across the PSNR-speed and SSIM-speed trade-offs – in both parallel and serial window processing forms of patch-based dense attention. This agrees well with the burden-factor analysis in Section III-D. The curves further highlight that competitive denoising performance in patch-based dense attention inference is predicated on using a small window-stride, in order to compensate for the neglect of dependencies between overlapping window regions by window-averaging. Second, we observe that increasing the training window-size W^{train} has diminishing returns on denoising performance. This is consistent with the intuition that non-local similarities of natural images are generally located close to the pixel of interest. Lastly, we observe that the patch-based dense attention curves of Figure 1 are not monotonically increasing with smaller window-stride, and in-fact have significant drops in the SSIM curves (Fig. 1 (b), blue triangle and pentagon curves). The source of this

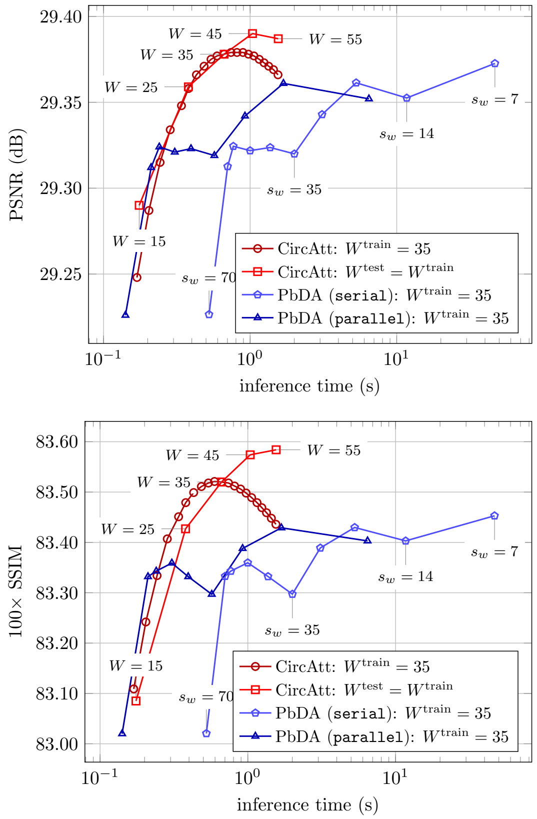


Fig. 1: Inference-time vs. denoising-performance trade-off for GroupCDL. Circle, pentagon, and triangle markers are generated by the same trained GroupCDL model ($W^{\text{train}} = 35$) under different inference strategies (CircAtt w/ window-size W, PbDA w/ window-stride s_w). The square markers are each generated by a GroupCDL with a different training window-size W^{train}. Performance (a) PSNR, (b) $100\times$SSIM, is evaluated on BSD68 [5] with $\sigma^{\text{train}} = \sigma^{\text{test}} = 25$. PbDA (`serial,parallel`) curves are generated via processing independent overlapping windows either sequentially or all at once, respectively. The same noise-realization for the dataset was used across all evaluations plotted. Note that W corresponds between CircAtt curve markers vertically (the same inference time), and s_w corresponds between PbDA curve markers horizontally (the same PSNR/SSIM).

behavior is windowing artifacts, highlighted visually in Figure 2. These visualizations show that the denoising-speed trade-off exhibited by patch-based dense attention processing comes with the penalty of unnatural artifacts in the form of grid-lines corresponding to the spatial pattern of window overlaps. In contrast, the proposed CircAtt does not exhibit windowing artifacts. Instead, as the window-size decreases, CircAtt processing transitions to fully convolutional CDLNet processing, and artifacts associated with FCNNs (such as hallucinated edges) are observed.

References

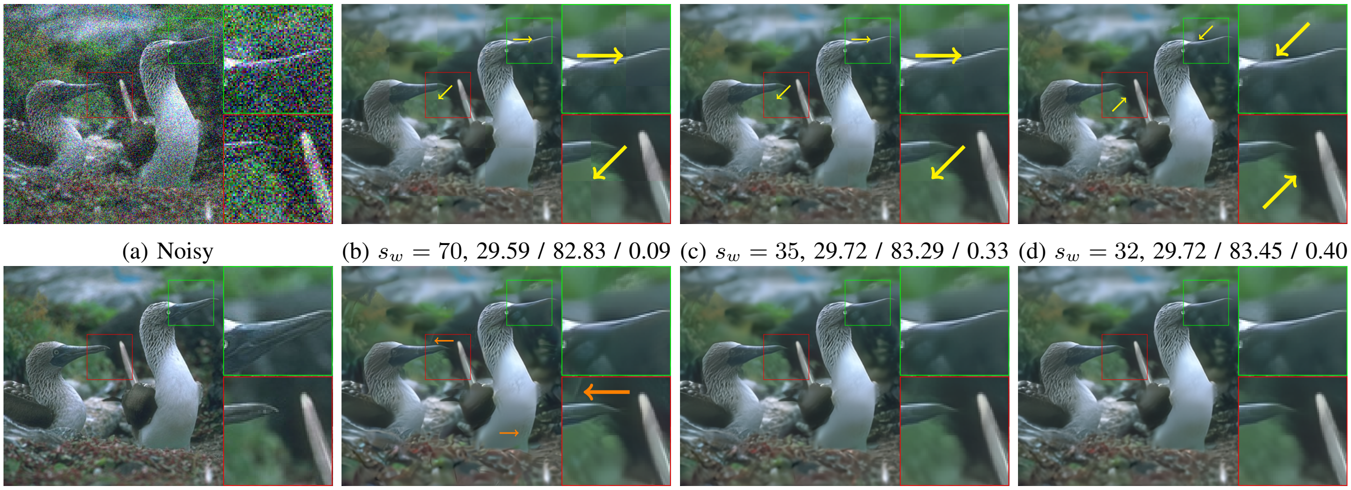
[1] M. B. Giles, *Collected Matrix Derivative Results for Forward and Reverse Mode Algorithmic Differentiation.* Springer Berlin Heidelberg, 2008, p. 35–44. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-68942-3_4

[2] F. White, M. Zgubic, M. Abbott, J. Revels, N. Robinson, A. Arslan, D. Widmann, S. D. Schaub, Y. Ma, W. Tebbutt, S. Axen, C. Rackauckas, P. Vertechi, BSnelling, K. Fischer, st, Y. Horikawa, B. Cottier, H. Ranocha, N. Schmitz, M. Besançon, Marco, Jutho, G. Dalle, B. Chen, A. B. PhD, dreivmeister, cormullion, V. B. Shah, and T. Wright, "Juliadiff/chainrulescore.jl," 2024. [Online]. Available: https://zenodo.org/doi/10.5281/zenodo.4754916

[3] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, "Non-local recurrent network for image restoration," in *Advances in Neural Information Processing Systems*, 2018, pp. 1680–1689.

[4] B. Lecouat, J. Ponce, and J. Mairal, "Fully trainable and interpretable non-local sparse models for image restoration," in *European Conference on Computer Vision (ECCV)*, 2020.

[5] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings of 8th IEEE International Conference on Computer Vision (ICCV)*, vol. 2, 2001, pp. 416–423.



(a) Noisy (b) $s_w = 70$, 29.59 / 82.83 / 0.09 (c) $s_w = 35$, 29.72 / 83.29 / 0.33 (d) $s_w = 32$, 29.72 / 83.45 / 0.40

(e) Ground Truth (f) $W = 15$, 29.57 / 82.92 / 0.08 (g) $W = 25$, 29.79 / 83.69 / 0.16 (h) $W = 35$, 29.83 / 83.75 / 0.29

Fig. 2: Comparison of inference-speed/denoising trade-off between patch-based dense attention (b-d) and CircAtt (f-h) processing by a color GroupCDL model ($\sigma^{\text{train}} = \sigma^{\text{test}} = 50$, $W^{\text{train}} = 35$). PSNR (dB) / $100\times$SSIM / GPU inference-time (s) shown in respective captions. Zoomed-in regions highlight blocking artifacts exist across shown patch window-strides (s_w), whereas CircAtt processing exhibits no blocking artifacts across inference window-sizes (W). Yellow arrows (b-d) point to specific blocking artifact boundaries of patch attention. Orange arrows (f) point to edge/texture hallucination artifacts of CircAtt with a small windowsize. The inclusion of (d) ($s_w = 32$), demonstrates that blocking artifacts are not merly a result of the effective spatial windowsize ($s_c W = 70$) being divisible by the overlapping window-stride. The same noise-realization is used across all methods (b-d, f-h).