• A machine learning based approach is the problem of finding a (non-linear) parametrized mapping $T_D : Y \rightarrow X$ that satisfies the pseudo-inverse property: $T_D^\dagger(y) \approx x$.

• Supervised training:

\[
J(\theta) = E_{(x,y)}[-\ell(T(\theta)(y), x)].
\]

• Training and deploying such methods for 3D CT reconstruction is still a challenge.

• Aim in this work is to explore the idea of invertibility for reducing GPU memory requirements.

### Learned Primal-Dual

This Learned Primal-Dual (LPD) architecture incorporates a forward operator into a deep neural network by unrolling a proximal primal-dual optimization scheme and replacing proximal operators with convolutional neural networks (CNNs).

![Learned Primal-Dual architecture](image)

**Algorithm 1 LPD**

1. Choose initial primal and dual variables $(x_0, u_0) = \text{init}(y)$, where $(x_0, u_0) \in (X, Y)$
2. For $i = 1, 2, \ldots, M$ do:
   3. Dual update: $u_i = u_{i-1} + \Gamma(T_{i-1} u_{i-1}, y)$
   4. Primal update: $x_i = x_{i-1} + \Lambda \left( x_{i-1}, T_{i-1} u_i \right)$
5. Return $x_M$

**Figure 1: Learned Primal-Dual architecture.**

**Figure 2: Invertible Learned Primal-Dual (LPD) architecture.**

• Implementation: [github.com/JevgenijaAksjonova/invertible_learned_primal_dual](https://github.com/JevgenijaAksjonova/invertible_learned_primal_dual)

### Quantitative evaluation

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>Execution time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDK</td>
<td>47.05</td>
<td>0.9997</td>
<td>1.06</td>
</tr>
<tr>
<td>nnFDK</td>
<td>46.10</td>
<td>0.9996</td>
<td>4.69</td>
</tr>
<tr>
<td>U-Net</td>
<td>46.65</td>
<td>0.9996</td>
<td>3.22</td>
</tr>
<tr>
<td>iLPD-10</td>
<td>46.05</td>
<td>0.9996</td>
<td>20.42</td>
</tr>
<tr>
<td>iLPD-20</td>
<td>45.09</td>
<td>0.9996</td>
<td>62.68</td>
</tr>
<tr>
<td>iLPD-30</td>
<td>44.10</td>
<td>0.9996</td>
<td>62.80</td>
</tr>
</tbody>
</table>

• The proposed LPD method requires significantly less GPU memory for training than the original LPD and therefore it is applicable for 3D CT reconstruction.

• Future work: extension to helical geometry.

### References


**Conclusion and future work**

- The proposed LPD method requires significantly less GPU memory for training than the original LPD and therefore it is applicable for 3D CT reconstruction.
- Future work: extension to helical geometry.

**Table 1: Performance metrics for various reconstruction methods in 2D low-dose CT.**

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>Execution time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iLPD-10</td>
<td>47.05</td>
<td>0.9997</td>
<td>1.06</td>
</tr>
<tr>
<td>iLPD-20</td>
<td>46.10</td>
<td>0.9996</td>
<td>4.69</td>
</tr>
<tr>
<td>iLPD-30</td>
<td>46.65</td>
<td>0.9996</td>
<td>3.22</td>
</tr>
<tr>
<td>iLPD-40</td>
<td>45.09</td>
<td>0.9996</td>
<td>20.42</td>
</tr>
<tr>
<td>iLPD-50</td>
<td>44.10</td>
<td>0.9996</td>
<td>62.68</td>
</tr>
<tr>
<td>iLPD-60</td>
<td>43.10</td>
<td>0.9996</td>
<td>62.80</td>
</tr>
</tbody>
</table>

**Table 2: Performance metrics for various reconstruction methods in 3D sparse-angle CBCT.**

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>Execution time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nnFDK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U-Net</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iLPD-10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iLPD-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iLPD-30</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>