# Deep Neural-Network Based Sinogram Super-resolution and Bandwidth Enhancement for Limited Data Photoacoustic Tomography

This MATLAB code was used as part of the work presented in

**Navchetan Awasthi\*, Gaurav Jain\*,Sandeep Kumar Kalva, Manojit Pramanik, Phaneendra K. Yalavarthy, “Deep Neural-Network Based Sinogram Super-resolution and Bandwidth Enhancement for Limited Data Photoacoustic Tomography,” in IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control (Special issue on Deep Learning in Medical Ultrasound) 2020 (DOI:** [**10.1109/TUFFC.2020.2977210**](https://www.google.com/url?q=https%3A%2F%2Fdoi.org%2F10.1109%2FTUFFC.2020.2977210&sa=D&sntz=1&usg=AFQjCNEA9wZxOOIgNmizraL4ozL5sTsmQg)**).**

**\* Co-first authors with equal contribution**

**\*\*The raw measurement data for the experimental experiments is not provided and can be requested.**

**\*\*\*Please contact if you find any mistakes or if you need any help regarding the codes.**

**The data for the above work can be obtained by using the google drive link and requesting for access:**

<https://drive.google.com/drive/u/3/folders/12egdPGu3muwBCyhYapntNpbAUTFDEJm_>

#Matlab implementation for testing the phantoms: Testing\_script.m

#Matlab implementation for adding noise : addnoise.m

#Matlab implementation for generating the patches : generate\_patches.m

#Matlab implementation for generating the System Matrix : sysBuildPAT\_mod\_Band.m

#Matlab implementation for generating the test data : test\_data\_generation.m

#Python implementation for loading the training and validation data : load\_train\_val\_data.py

#Python implementation for the loss function for our network : loss.py

#Python implementation for testing the network : test.py

#Python implementation for SRCNN network: srcnn\_model.py

#Python implementation for training SRCNN network: srcnn\_train.py

#Python implementation for U-Net-Relu network: unet\_relu.py

#Python implementation for training the U-Net-Relu network: train\_unet\_relu.py

#Python implementation for U-Net-Elu network: unet\_elu\_All.py

#Python implementation for training the U-Net-Elu network: train-unet-elu-all.py

#Python implementation for the U-Net-Hybrid architecture : unet1.py

#Python implementation for training the U-Net-Hybrid network : train.py