

Aydin Studio Tutorials

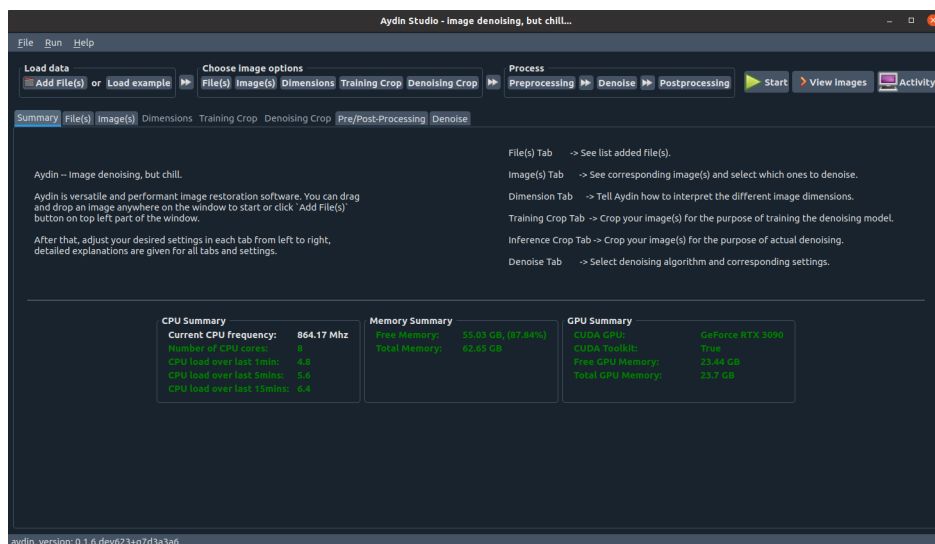
To make Aydin accessible to more users, we designed and implemented a user-friendly GUI. We call our GUI, `Aydin Studio`. We intended to have a self-explanatory GUI but also added detailed explanations on each part and tooltips on most of the elements of the GUI.

Before we dive into tutorials, please make sure to have `Aydin` on your computer.

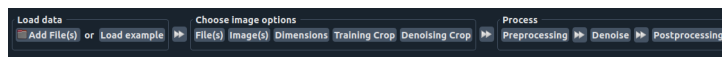
Below we explain how to use fundamental features of the `Aydin Studio`.

Let's start `Aydin Studio`

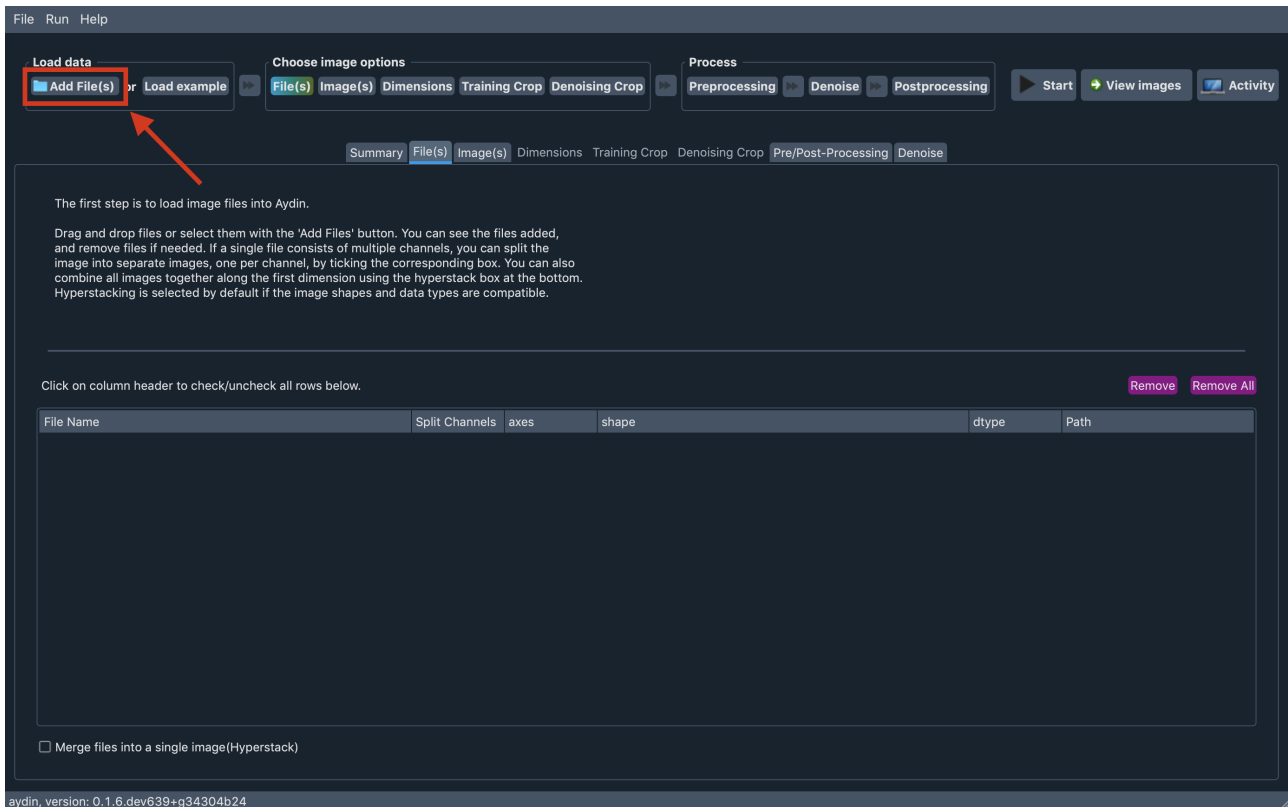
To start `Aydin Studio`, if you have the bundle installation just double-click on its icon. If you have the `pip` installation, you can call `aydin` on your terminal without passing any arguments, this will boot up `Aydin Studio`. Once loading is done, a screen as shown below will be welcoming you:



Using `Aydin Studio` is pretty easy. First you load your image(s). You can hyperstack multiple images if feasible or you can split channels of a multi-channel image into separate images. Then you can tell Aydin how to interpret dimensions with the help of the dimensions tab. You can configure image cropping for training and denoising independently with help of training crop and denoising crop tabs. If you like to use the same cropping for both training and denoising, that is also possible. You just need to select the checkbox on the top right in the `Training Crop` tab. Finally, you have the pre and post processing settings, and then you can choose which algorithm to use for denoising and tune the corresponding parameters. You are set to start denoising! By clicking the `Start` button located on the top right part of the window, you can start denoising your images and results will show up once it is done.



Loading image files



Loading images to Aydin is easy. You can drag and drop image files of your choice to anywhere on the Aydin window(except napari canvases on the cropping tabs) or you can click to `Add File(s)` button on the top left part of the screen and open your files with the help of file dialog.

Hyperstacking

The screenshot shows the Aydin Studio interface. At the top, there are tabs for 'Load data', 'Choose image options', and 'Process'. The 'Load data' tab has 'Add File(s)' and 'Load example' buttons. The 'Choose image options' tab has 'File(s)', 'Image(s)', 'Dimensions', 'Training Crop', and 'Denoising Crop' buttons. The 'Process' tab has 'Preprocessing', 'Denoise', and 'Postprocessing' buttons. Below these are 'Start', 'View images', and 'Activity' buttons. The main area shows the 'File(s)' tab selected, with a table of loaded files. A red arrow points to the 'Merge files into a single image(Hyperstack)' checkbox at the bottom left of the table.

The first step is to load image files into Aydin.

Drag and drop files or select them with the 'Add Files' button. You can see the files added, and remove files if needed. If a single file consists of multiple channels, you can split the image into separate images, one per channel, by ticking the corresponding box. You can also combine all images together along the first dimension using the hyperstack box at the bottom. Hyperstacking is selected by default if the image shapes and data types are compatible.

Click on column header to check/uncheck all rows below.

File Name	Split Channels	axes	shape	dtype	Path
ss1_noisy.png	<input type="checkbox"/>	YX	(1000, 1000)	uint8	/Users/ahmetcan.solak/Data/st...
ss2_noisy.png	<input type="checkbox"/>	YX	(1000, 1000)	uint8	/Users/ahmetcan.solak/Data/st...
ss3_noisy.png	<input type="checkbox"/>	YX	(1000, 1000)	uint8	/Users/ahmetcan.solak/Data/st...
ss4_noisy.png	<input type="checkbox"/>	YX	(1000, 1000)	uint8	/Users/ahmetcan.solak/Data/st...

☒ Merge files into a single image(Hyperstack)

It is common to have a folder full of images where each image file is only a single z-slice or a single time point of your dataset. As explained in the previous section, it is fairly easy to load all those files to Aydin Studio. To make Aydin Studio interpret the dataset correctly, users might like to create a hyperstack out of the files they've loaded. We've considered such a need in our design process and included a checkbox on the File(s) tab that let's you create a hyperstack from the images you loaded if all loaded images have the same shape.

Split channels

FileRunHelp

Load data

Add File(s) or Load example

Choose image options

File(s)Image(s)DimensionsTraining CropDenoising Crop

Process

PreprocessingDenoisePostprocessing

StartView imagesActivity

SummaryFile(s)Image(s)DimensionsTraining CropDenoising CropPre/Post-ProcessingDenoise

The first step is to load image files into Aydin.

Drag and drop files or select them with the 'Add Files' button. You can see the files added, and remove files if needed. If a single file consists of multiple channels, you can split the image into separate images, one per channel, by ticking the corresponding box. You can also combine all images together along the first dimension using the hyperstack box at the bottom. Hyperstacking is selected by default if the image shapes and data types are compatible.

Click on column header to check/uncheck all rows below.

RemoveRemove All

File Name	Split Channels	axes	shape	dtype	Path
Twin-peaks-skyline.jpg	<input checked="" type="checkbox"/>	YXC	(1080, 1920, 3)	uint8	/Users/ahmetcan.solak/Data/T...

☐ Merge files into a single image(Hyperstack)

Having multi-channel images is cool, being able to split channels of a n-dimensional stack and process each channel separately might be cooler. At least we believe so, hence we give our users a checkbox that lets them split each channel of a multi-channel stack into separate images.

Choosing images to denoise

The screenshot shows the Aydin Studio interface. At the top, there are tabs for 'Load data', 'Choose image options', and 'Process'. The 'Choose image options' tab is active, showing sub-tabs for 'File(s)', 'Image(s)', 'Dimensions', 'Training Crop', and 'Denoising Crop'. The 'Image(s)' sub-tab is selected. Below the tabs, there is a table of loaded images. The table has columns: 'file name', 'denoise', 'axes', 'shape', 'dtype', 'size', and 'output folder'. The first row shows 'fountain.png' with a checked 'denoise' checkbox. A red box highlights the 'denoise' checkbox, and a red arrow points to it. The 'Process' tab shows buttons for 'Preprocessing', 'Denoise', and 'Postprocessing'. The 'Denoise' button is highlighted.

Next, we inspect the images contained in the added files.

You can see basic image information such as name, axis label, array shapes, data types, volume in voxels, and size in bytes. When several images are loaded, each images is denoised independently which means that training and denoising are done per image. For bulk denoising of series of images please use the command line interface (CLI).

Click on column header to check/uncheck all rows below.

file name	denoise	axes	shape	dtype	size	output folder
fountain.png	<input checked="" type="checkbox"/>	YX	(588, 402)	uint8	230.84 KB	/Library/Cache...

We provide details on how we interpret each image loaded to Aydin Studio on the Image(s) tab. Also this is the last place to decide, which of the loaded images you want to denoise. For each loaded image, you can decide to denoise or not with the help of checkbox provided on the denoise column.

Setting output folder to a specific image

The screenshot shows the Aydin Studio interface with the 'Image(s)' tab selected. The interface includes a top menu bar (File, Run, Preferences, Help) and a main toolbar with sections for 'Load data', 'Choose image options', and 'Process'. The 'Load data' section has buttons for 'Add File(s)' and 'Load example'. The 'Choose image options' section has buttons for 'File(s)', 'Image(s)', 'Dimensions', 'Training Crop', and 'Denoising Crop'. The 'Process' section has buttons for 'Preprocessing', 'Denoise', and 'Postprocessing'. There are also 'Start', 'View Images', and 'Activity' buttons.

Below the toolbar, there are tabs for 'Summary', 'File(s)', 'Image(s)', 'Dimensions', 'Training Crop', 'Denoising Crop', 'Pre/Post-Processing', and 'Denoise'. The 'Image(s)' tab is active, displaying a table of loaded images.

Text in the interface: "Next, we inspect the images contained in the added files. You can see basic image information such as name, axis label, array shapes, data types, volume in voxels, and size in bytes. When several images are loaded, each images is denoised independently which means that training and denoising are done per image. For bulk denoising of series of images please use the command line interface (CLI)."

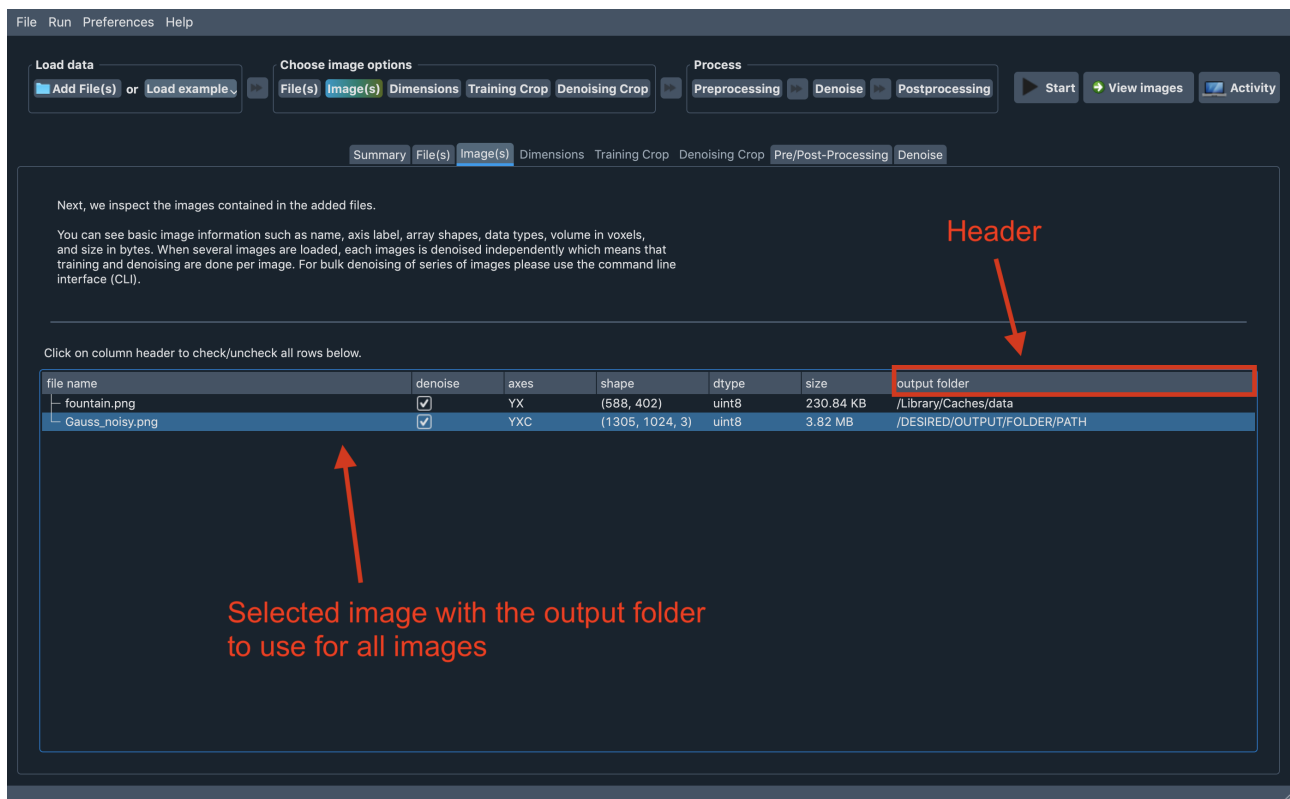
Text below the table: "Click on column header to check/uncheck all rows below."

file name	denoise	axes	shape	dtype	size	output folder
fountain.png	<input checked="" type="checkbox"/>	YX	(588, 402)	uint8	230.84 KB	/Library/Caches/data

A red arrow points to the 'output folder' column header.

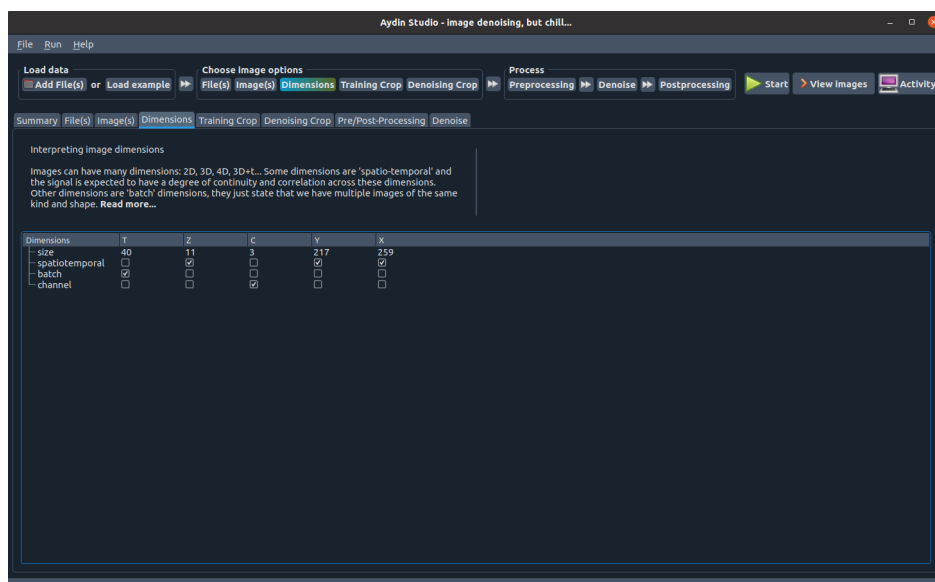
By default, Aydin Studio tries to write the result image and related files to the same exact folder of the input image file. This comes handy, when you like to find the input/output images together, however, we understand that this might not be always the case for all of our users and let our users to specify output folder for each loaded image by double-clicking to the last column of each image entry on the `Image(s)` tab.

Setting output folder to all loaded images

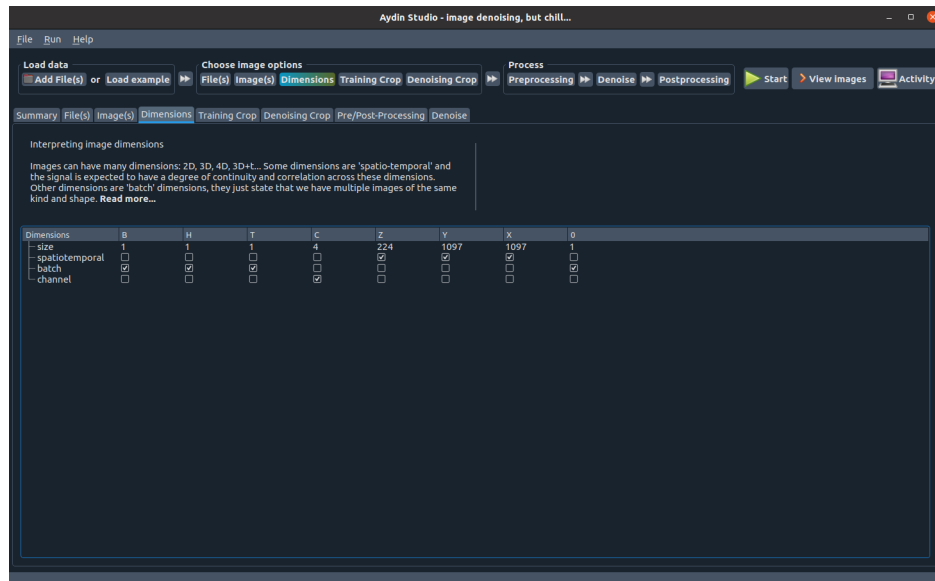


It is also possible to set output folder to all loaded images to Aydin Studio. You can select the image row with your desired output folder for all images then click the `output folder` header (highlighted on the screenshot) to set same output folder to the all loaded images.

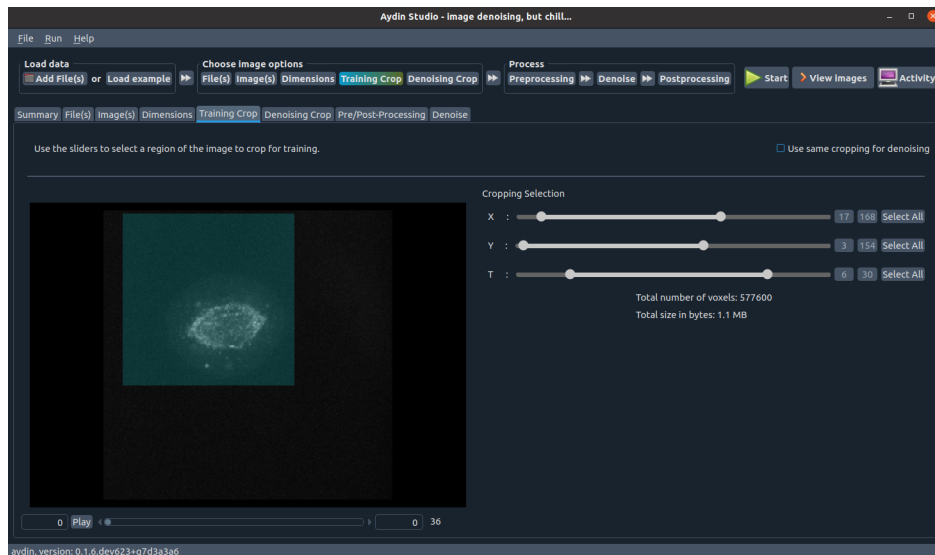
Dimensions tab



Aydin Studio tries to read images and interpret their metadata correctly. There are numerous different file formats out there with specific metadata formats and conventions. Aydin Studio supports most of the common image file formats such as zarr, tif, czi, png, and so on. In the Dimensions tab you can see how Aydin Studio interprets each dimension of your image. If you feel the need to make a correction, you can make adjustments with the help of checkboxes provided for each given dimension. If you observe a mistake about the number of dimensions detected please open an issue and let us know about it in detail.

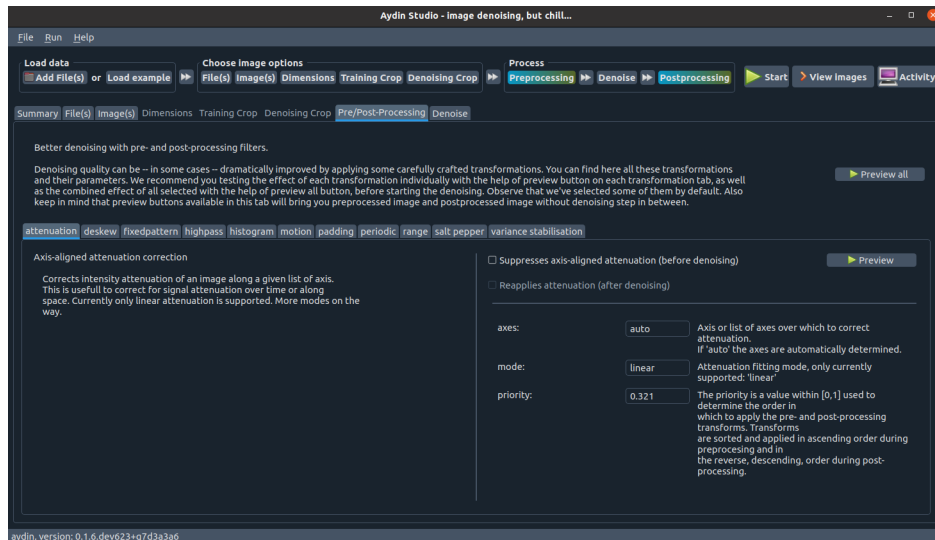


Cropping selection



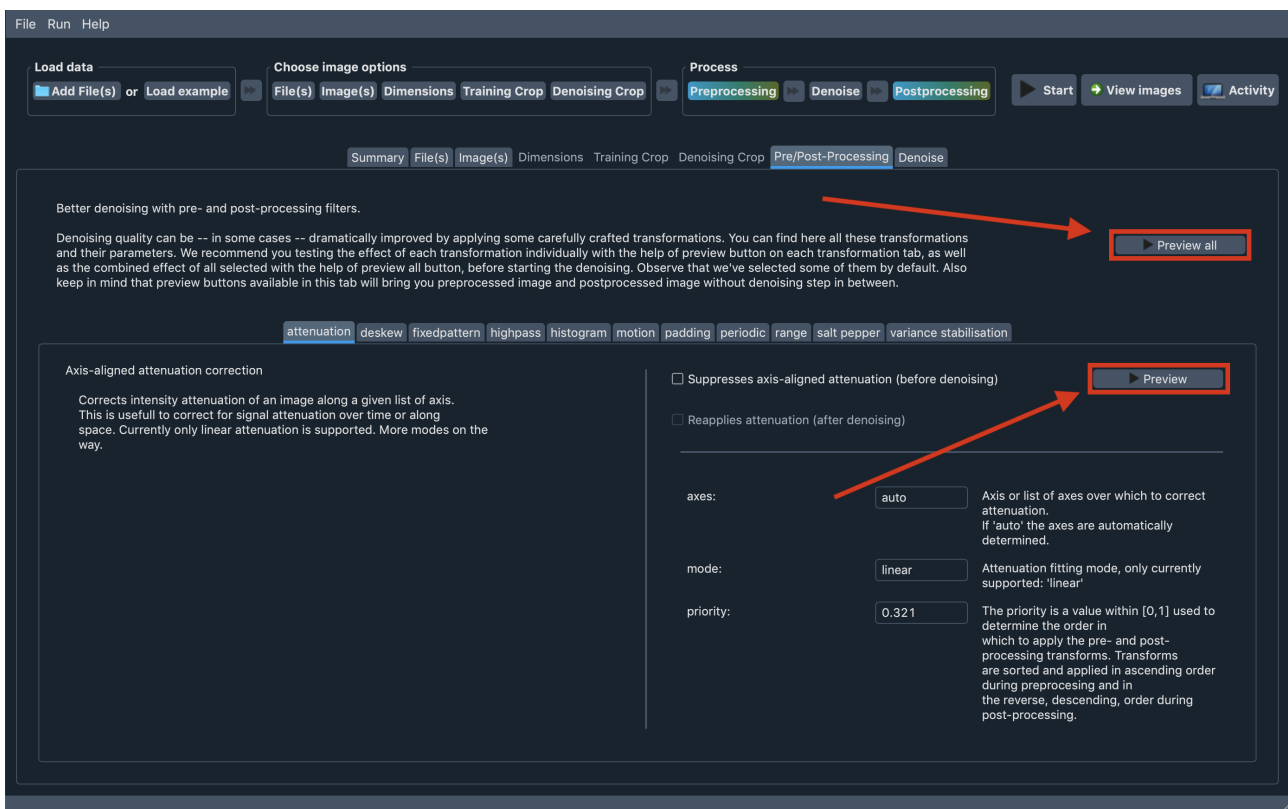
Cropping your image before passing to train a model can shorten your training duration significantly. To address this need, we built two cropping tabs: one for training crop and one for inference crop respectively. Currently, we are only supporting cropping across X, Y, Z and T dimensions. Looking forward to support cropping over Z in near future as well.

Pre and post processing



Different imaging modalities and different imaging setups bring different artifacts and imperfections to the acquired images. Oftentimes, addressing those imperfections before denoising the image helps to improve denoising performance. While handling these imperfections before denoising via preprocessing, inevitably one ends up changing certain characteristics of the image as well. If users want to revert the changes applied by certain pre-processing algorithm back after denoising, we provide a post-processing option.

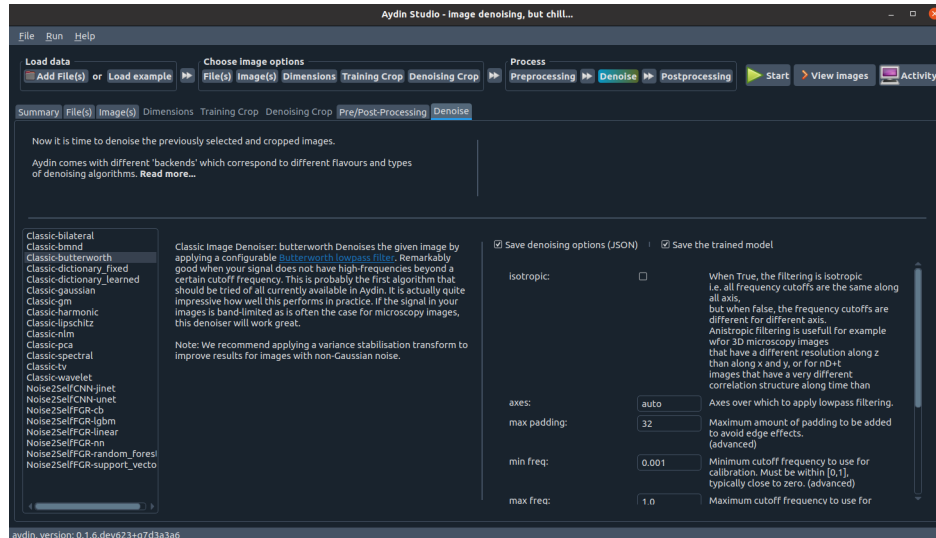
Processing preview and preview all



We provide a Preview button on each subtab of the Pre/Post-Processing tab. With the help of the Preview button one can see the preprocessed(image that would be passed to denoising if it is applied) and postprocessed(resulting image of applying postprocessing to preprocessed image(without denoising)) images.

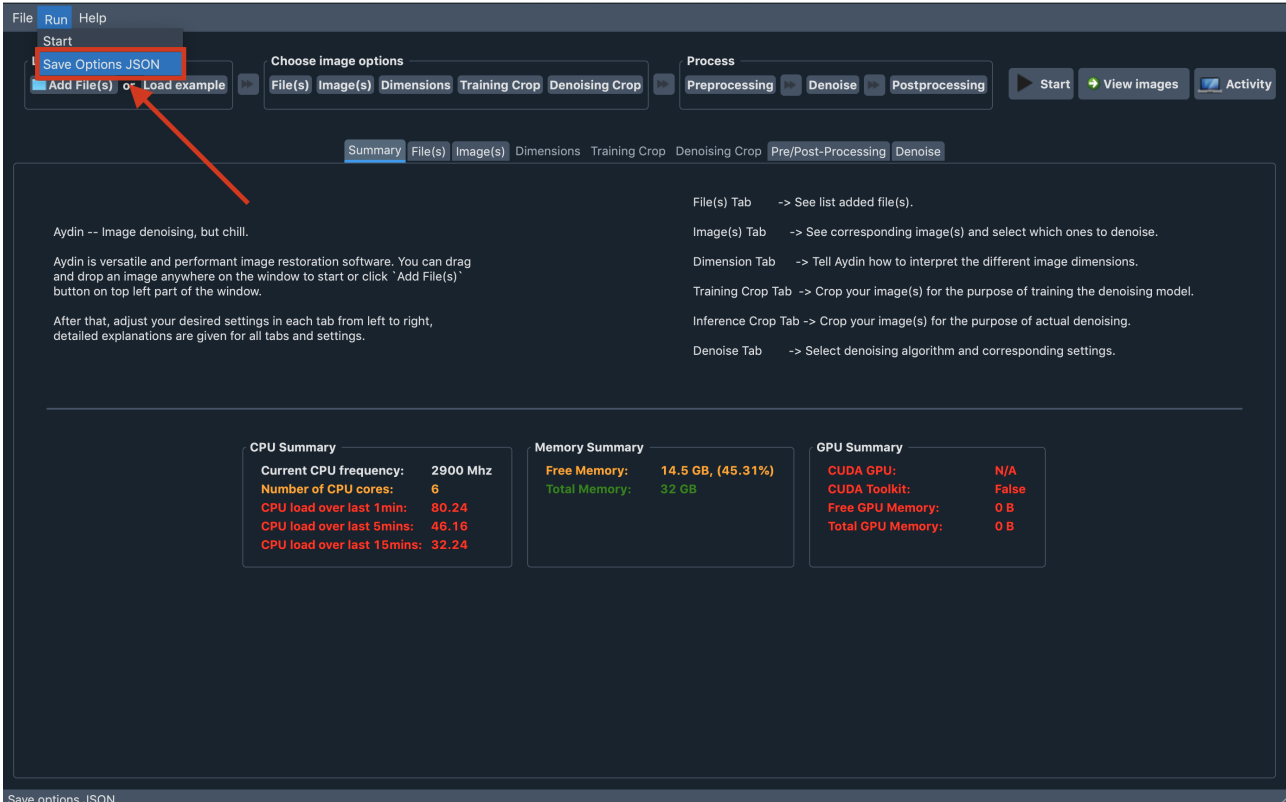
Also, we provide the `Preview all` button in the `Pre/Post-Processing` tab and one can use that button to observe cumulative effect of chosen preprocessing and postprocessing transforms.

Denoising options



We implemented numerous denoising algorithms using different computation libraries and models. We refer to those different implementations as variants. We expose an extensive list of arguments for each provided variant. Users can find default value and a small explanation for each argument in the list.

Saving denoising options and trained model



You can save your option selections for the selected variant with the help of `Save Options JSON` button in `Run` menubar located on the top part of `Aydin Studio` window. You can pass the saved JSON file to `Aydin CLI` to train new models for new images and denoise them. Worth to mention that `Aydin Studio` saves the options JSON next to the resulting image by default on each run.

Aydin Studio also saves the trained model next to the resulting image by default. You can pass such trained model file to Aydin CLI to denoise more images without training a model again. Basically this provides possibility to fine tune parameters and train a model for one of your images from a dataset and rapidly denoise(infer) on the rest of the images in your dataset.