Best place to start: longform [tutorial](https://docs.google.com/document/d/1exxVxgPkdJzQFXlw-mH6ecuxrKeMhWO0UUNTq-qEo-0/edit#heading=h.7wpjvrd6g5ee) with more image examples and details on using the GUI application.

**Download and Installation**

Download a release from the CTS github page. Unzip the archive into your matlab/toolbox folder, and add the resulting cryotomosim folder to your Matlab path. It is not necessary to add subfolders.

Requirements:

Matlab 2019b or later and toolboxes: image processing, statistics and machine learning

[EMIOD toolbox](https://github.com/rbehrouzi/emtoolbox) for matlab

IMOD installed for running simulations

[UIpickfiles](https://www.mathworks.com/matlabcentral/fileexchange/10867-uipickfiles-uigetfile-on-steroids) is an optional but recommended toolbox. It allows easier navigation and selection of input structures, including duplicate files and reading from multiple directories.

Download at least one structure file from a database such as the RCSB for testing. Cif files are preferred as they have additional features, but pdb files are perfectly serviceable.

WIP GUI tool: execute the command ‘ctsgui’ on the matlab command line for GUI controls. Incomplete features, but may be more accessible to those with no command line experience.

**Note for Mac users**: start matlab from the command line, not from a shortcut. Simulations fail if started from shortcut, as matlab fails to run IMOD commands in that state.

**Tomogram size**: Don’t make anything the size of a full tomogram. As examples from the preprint article show, you don’t need something that large and the synthetic models are often far more data-dense than actual tomograms. It also causes silent errors from IMOD and/or matlab trying to create files of many gb.

**Minimal commands for a basic workflow**

WARNING: PASTING INTO MATLAB MIGHT BREAK CHARACTERS. ‘ and “ (single or double quotes) may be converted to improper special characters. If it’s red, it’s broken. If the argument is purple, it’s working. Replace the broken instance with a manually typed one when necessary.

1. set up the model parameters. This command readies parameters, with most using the defaults but specifying a pixel size of 12 (required input), and a single layer of particles (optional, defaults to 1). A GUI will appear to select the input structures for each layer. This command can be run inside the next command as an argument, or you can provide the arguments as a cell array (in curly brackets). Creating a variable makes it fast and easy to run many model generations with the same parameters programmatically, and avoids needing to use a GUI to select files repeatedly. Estimated runtime: <1 minute (will be longer if large numbers of structures are input).

[parammodel] = param\_model(12,'layers',1);

help param\_model for a manual of model parameter arguments and options.

1. Generate the model with the provided parameters. The model size is 300x400x50 voxels, and is output with the suffix \_model\_1. The files are in the /tomosim directory, in a folder named with the input structure names following a timestamp. The second command demonstrates how to include the model parameters inside the same argument, if you do not wish to store them as a variable. Estimated runtime: <2 minutes

[cts] = cts\_model(zeros(300,400,50),parammodel,'suffix','model\_1');

Or [cts] = cts\_model(zeros(300,400,50),{12});

1. View the model if desired. The default model parameters include a grid, so the edge of a carbon hole should be visible along the left side. Subcomponents of the model are also viewable, stored inside the cts struct variable. The particle types are stored across a number of models in the splitmodel field, according to their filename. cts.splitmodel run in the command window will display its contents for viewing with the second example command.

sliceViewer(cts.vol);

sliceViewer(cts.splitmodel.X);

1. Set up simulation parameters. This functions similarly to the param\_model function, but has no required inputs – all options have a default value. The following command is the easiest to use, as it provides a GUI to input all the parameters with default values already filled in. To skip the GUI, name-value pairs are used to change parameters from default values. If you do not want to change any parameters, you can simply not provide this argument to the simulator function – it will automatically supply defaults.

paramsim = param\_simulate('gui');

1. Run the simulation. This uses input (or default) parameters to project an initial tiltseries, detect electron scattering, CTF convolve, and reconstruct a tomogram, along with generating an atlas of object identities. The input model is selected with a GUI (change ‘gui’ to a full path to use a file programmatically) – you can use either a model .mrc or .mat file, but the .mat file is required to generate the atlas. Estimated runtime: <1 minute

cts\_simulate('gui',paramsim,'suffix','tutorial1');

cts\_simulate('gui','suffix','tutorial1'); %using default parameters only

The simulation steps will be output into a subfolder inside the model folder that was used, and in this example the simulation folder will have \_tutorial1 as a suffix. The easiest way to view all the steps in series is with IMOD’s 3dmod command in that folder, with 3dmod \*.mrc. the 5\_recon\_X is the reconstructed tomogram, and atlas\_X is the atlas of particle classes.

For more detailed information on each command, including a list of all input/output options, use matlab’s built-in help functionality. The following is the syntax for retrieving documentation:

help commandname

For instructions on how to use more complex particle handling options in CTS model generation, see the readme\_structurefiles word document.