1. **Test installation**
   Run the ASTRA command `astra_test`, and the ASTRA sample `s004_cpu_reconstruction` to test ASTRA is working.

2. **Basic operations**
   Define a parallel beam geometry (not too large, up to $256 \times 256$), and your favourite phantom image. Use ASTRA to perform a forward projection and a backprojection, and display the results.

3. **Basic reconstructions**
   Using the sinogram $p$ from the previous exercise, run reconstructions with the **FBP** (Filtered Backprojection) and **SIRT** algorithms in ASTRA, and display the results.

   Define the residual of a reconstruction $\bar{x}$ as $p - A\bar{x}$ (where $A$ is the system matrix). For the two reconstructions just computed, also compute their residuals and the (Euclidean) norm of the residuals.

4. **Convergence plot**
   While running **SIRT**, compute the norm of the residual after each iteration. Also compute the norm of the difference with the phantom image after iteration. Plot these two series to visualize the convergence. Do the same thing for the **SART** algorithm. (Note that for SART, each iteration processes only a single projection angle.)
5. SPOT operator
Again define your favourite 2D geometry and define a opTomo SPOT operator. Take a phantom image, and use the SPOT operator to compute a forward projection and a backprojection, and display the results. Use the Matlab function lsqr to compute a reconstruction from the forward projection and the SPOT operator.

6. Block algorithms
Use ASTRA to implement the block algebraic algorithm discussed this morning for a 2D reconstruction problem. (This will require defining multiple projection geometries and projectors.) It is possible to vary the number of subsets used; the two extremes are one projection per subset, and only a single subset. Compare convergence speed for different numbers of subsets.

7. Null space
In this exercise you will visually inspect the null space of the tomography forward projection map.

The SIRT algorithm discussed in the lectures, when started from initial vector \( x^{(0)} = 0 \), converges to the shortest (weighted) least-squares solution to the problem \( Ax = b \).

In general, when started from any vector \( x^{(0)} \), it converges to the (weighted) least-squares solution closest to \( x^{(0)} \). When applied to the system \( Ax = 0 \), this can be used to compute an element of the null space of \( A \) closest to a given \( x^{(0)} \). (Recall: \( x \) is in the null space of \( A \) if and only if we have \( Ax = 0 \).)

Take various images (of people, objects, geometric phantoms), and, using the above method, compute the element \( y \) in the null space of \( A \) closest to that image.

Depending on the geometry (different numbers of projection angles, and different angular ranges), \( y \) might be very close to your image, or it might not resemble it at all. Experiment with these parameters of the geometry, especially the number of projection angles, going all the way down to 8, 4, 2, or even 1 angle(s).

8. Fan beam geometry
Define a volume geometry of 64 \( \times \) 64, and two fanflat_vec projection geometries: one for a regular fan beam geometry, and one for a fan beam geometry with the detector shifted by 4 pixels along the direction of the detector. Perform a forward projection and a reconstruction for both of them and verify that they behave as expected.

Study what happens if you create a sinogram of a phantom using the shifted geometry, but perform the reconstruction using the original non-shifted geometry. This models misalignment in an experimental setup.

NB: When using fan beam, you’ll have to use a line_fanflat projector. Also, you can use astra_mex_data2d(’change_geometry’, data_id, new_proj_geom) to change the geometry attached to a sinogram data object.

9. Discretizations
Study the effect of using a different projector (line, strip, linear) for the forward projection than for the reconstruction, by plotting how the error relative to the ground truth behaves while performing (very many) SIRT iterations. Use a very small phantom to better see these effects.

For example, try strip for forward projection and line for reconstruction, compared to using line for both.

10. GPU usage
Test GPU usage using the s003_gpu_reconstruction.m sample script. Try several of the other exercises using ASTRA’s GPU functions.