Welcome to GENFIRE

Welcome to the MATLAB implementation of GENFIRE.

GENFIRE, for GENeralized Fourier Iterative REconstruction, is a robust, Fourier-based reconstruction algorithm that is capable of using a limited set of input projections to generate a 3D reconstruction while also partially retrieving missing projection information. It does this by iterating between real and reciprocal space and applying simple constraints in each to find an optimal solution that
correlates with the input projections while simultaneously obeying real space conditions such as positivity and support.The result is a more consistent and faithful reconstruction with superior contrast and, in many cases, resolution when compared with more traditional 3D reconstruction algorithms such as Filtered Back-Projection (FBP) or the Algebraic Reconstruction Technique (ART).

Tutorial

Reconstructions are run either with `GENFIRE_Main.m` for a cubic array and 3 Euler angles or with `GENFIRE_Main_Tomo.m` for a rectangular array with a single tilt-axis. Simply edit the parameters in the appropriate file and run the script to execute the reconstruction. The parameters that may be adjusted are

- `filename_Projections` (*char*) - filename containing projection images as an N x N x num_projections array (N must be even)
- `filename_Angles` (*char*) - filename containing the angles as either a num_projections x 3 array of Euler angle triples (phi, theta, psi) or a num_projections x 1 array indicating a single-axis tilt series
- `filename_Support` (*char*) - filename containing an NxNxN binary support
- `filename_InitialModel` (*char*) - filename containing an NxNxN initial model. Comment out to skip providing one, and an empty array will be used instead
- `numIterations` (*int*) - number of GENFIRE iterations to run
- `pixelSize` (*double*) - size of a single pixel, used to display the Fourier Shell Correlation (FSC) with proper units if desired. Setting `pixelSize` to 0.5 will
- `oversamplingRatio` (*int*) - controls the amount of zero padding. Formally, the oversampling ratio in a given direction is the total array size (after padding) divided by the size of the input projection.
- `griddingMethod` (*int*) - controls the gridding method. Choose from the following:

1. FFT - Faster, less accurate
2. DFT - Slower, more accurate

- `constraintEnforcementMode` (*int*) - choose a method of enforcing the Fourier constraint from:

  1. Resolution extension and suppression
  2. Resolution extension only
  3. Enforce all datapoints always (No resolution extension or suppression)

- `interpolationCutoffDistance` (*double*) - maximum tolerable radial distance to consider measured datapoints for gridding to a given Fourier voxel
- `constraintPositivity` (*bool*) - on/off positivity constraint
- `constraintSupport` (*bool*) - on/off support constraint
- `useCustomGridSize` (*bool*) - on/off manually choose the 3D array size of the Fourier grid as opposed to determining it automatically from the size of the projections. If this is turned on, the Fourier grid dimensions must be provided as a 3-element vector in `FourierGridSize`
- `FourierGridSize` (*vector*) - 3-element vector representing the dimensions of the custom Fourier grid size. Only used if `useCustomGridSize` = 1
- `useCustomEulerConvention` (*bool*) - on/off use a custom Euler angle convention rather than the default z-y-z. The rotation vectors should be provided in `Euler_rot_vecs`
- `Euler_rot_vecs` (*cell array*) - cell array containing three 3-element vectors representing the rotation axes to use for assembling the Fourier grid. The Euler angles (phi, theta, psi) will be applied using these Euler rotation vectors such that each projection is rotated by phi degrees about `Euler_rot_vecs{1}` , theta degrees about `Euler_rot_vecs{2}` , and psi degrees about `Euler_rot_vecs{3}` .
- `ComputeFourierShellCorrelation` (*bool*) - the data will be divided into two halves, independently reconstructed, and the FSC will be compared between the two halves. A final reconstruction will then be computed from all of the data
- `numBins` (*int*) - number of bins to use in the calculation of FSC (if applicable)
- `percentValuesForRfree` (*double*) - percentage of data to withold within each shell for calculation of Rfree
- `numBinsRfree` (*int*) - number of bins to divide Fourier space in for Rfree
- `doCTFcorrection` (*bool*) *(experimental)* - turn on CTF correction

- `CTFThrowOutThreshhold` (*double*) *(experimental)* - Fourier values in regions where the corresponding CTF is lower than this value will not be gridded

Author: Alan (AJ) Pryor, Jr.
email: apryor6@gmail.com
Jianwei (John) Miao Coherent Imaging Group
University of California, Los Angeles