Designing a Platform for Computed Tomography

X. Gao, Q. Li, Suerfu, Y. Zhou

Presentation for APC 524

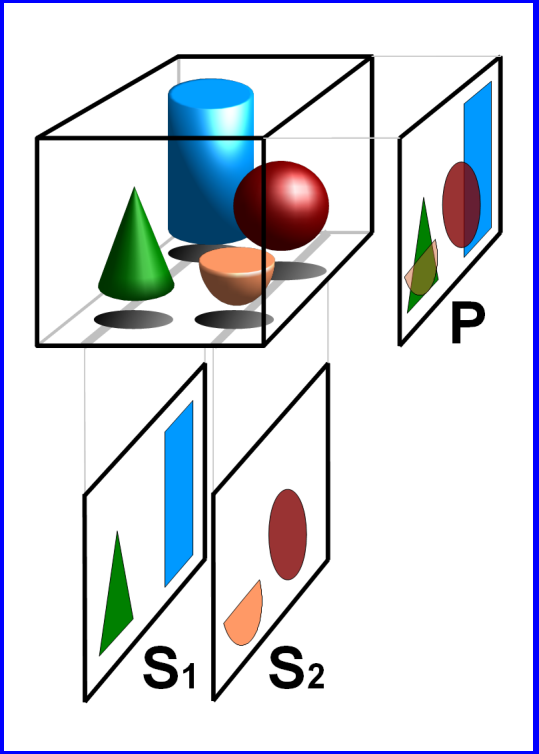
January 6, 2015

What is computed tomography?

Goal of Tomographic Reconstruction:

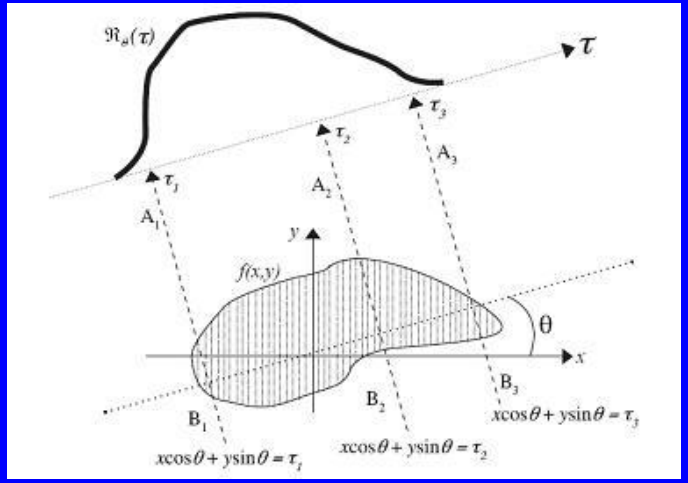
Reconstruct cross-section image of objects from projections along different directions.

In CT: projection is line integral of X-ray absorption coefficient.



In mathematical language

For a 2D function, projection is a 1D function, obtained by doing line integral along parallel lines.



Q: If such projection is known along infinitely many directions, can we reconstruct the original 2D function?

A: Yes!

Filtered Back-Projection

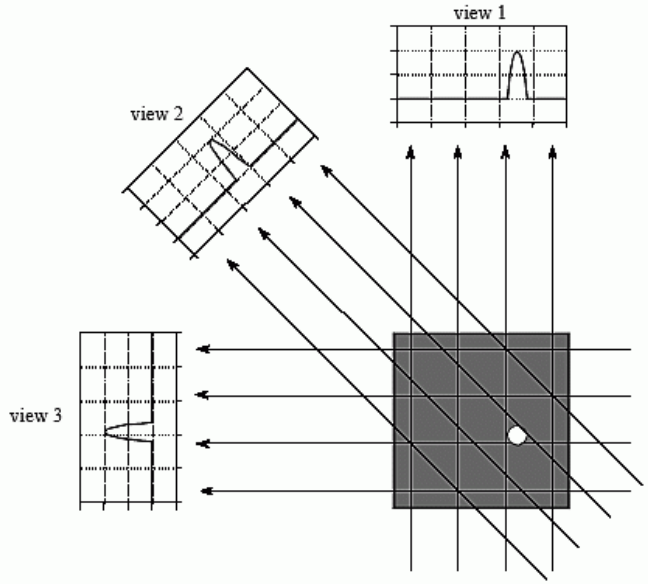


FIGURE 25-15
CT views. Computed tomography acquires a set of views and then reconstructs the corresponding image. Each sample in a view is equal to the sum of the image values along the ray that points to that sample. In this example, the image is a small pillbox surrounded by zeros. While only three views are shown here, a typical CT scan uses hundreds of views at slightly different angles.

An interface for computed tomography

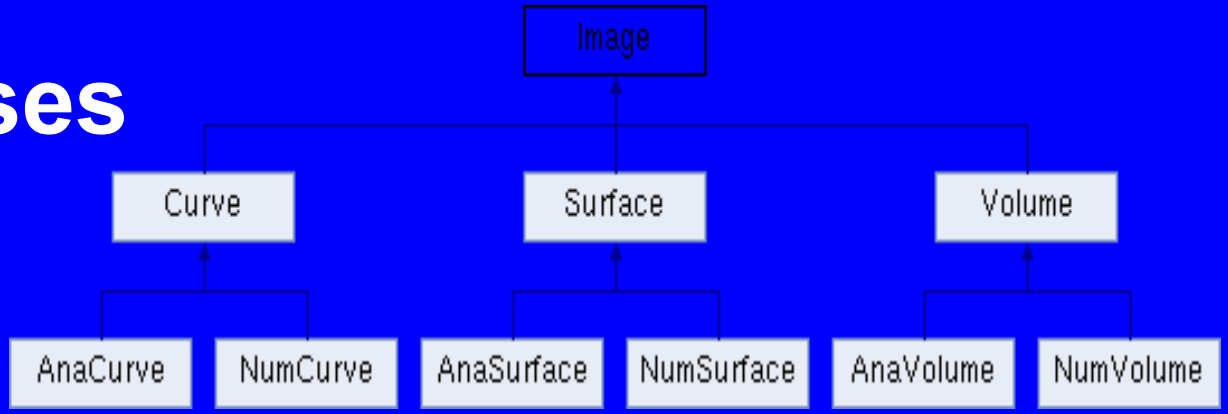
- Be able to simulate CT so that one knows how many X-ray images to take, and how much to pay for.

- Be able to implement different algorithms for reconstructing CT images.

An interface for computed tomography

- Written in C++
- g++ with C++11 required
- HDF5 for input / output
- Python for basic visualization
- ffmpeg for movie
- Mayavi + VTK for 3D rendering

Image Classes



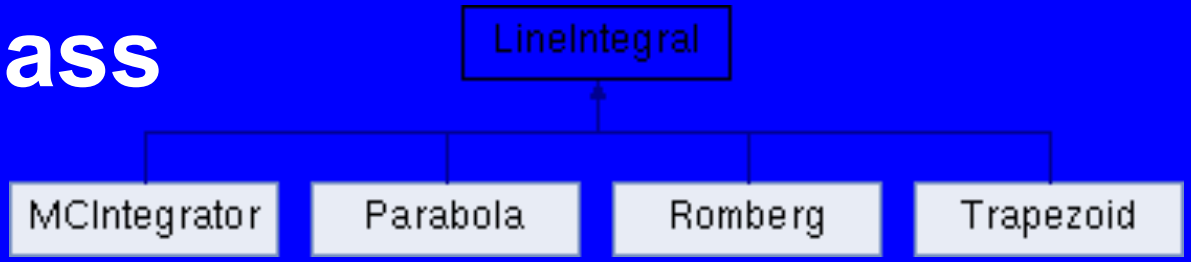
all implements operator( ) and print( ).

Surface :: GetProjection(integrator, angle) -> Curve.

Volume::GetProjection(integrator, angle) -> Surface, or
Volume::GetProjectionAtZ(intg, angle, z) -> Curve.

Each can be defined either analytically or numerically.

Integrator Class

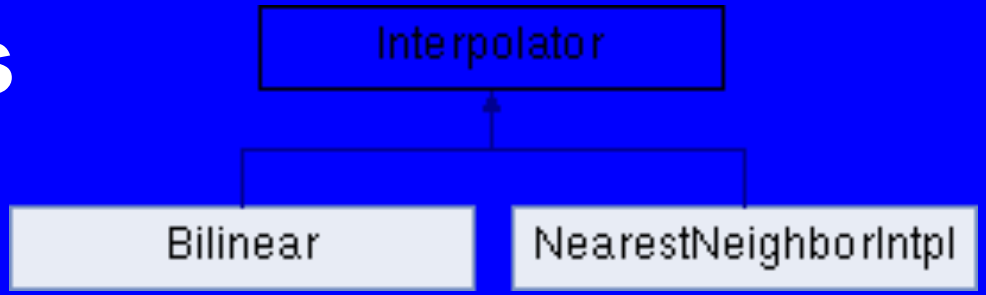


To obtain projection, line integral is necessary.

Implements Integrate(*f, xmin, xmax, step)

This function is passed to Surface::GetProjection( ) to perform the projection/Line-integral.

Interpolator Class



NumCurve, NumSurface and NumVolume needs interpolation when data is requested through operator( ).

Interpolator converts the data to internal 1D-array for unified approach in all dimensions.

ImageArray

ImageArray is a vector containing projection images taken from various angles and heights.

ImageArray:

PushBack (Curve, angle, height) :

GetCurve, Height, Angle (int i) :

ConvolveWithKernal (*kernal) : filters all images stored.

Reconstruction class

Everything needed by reconstruction is

1. each projection curve,
2. the angle at which it was taken,
3. and the height (for 3D),

ImageArray!

Visualization

Export result in HDF5 format.

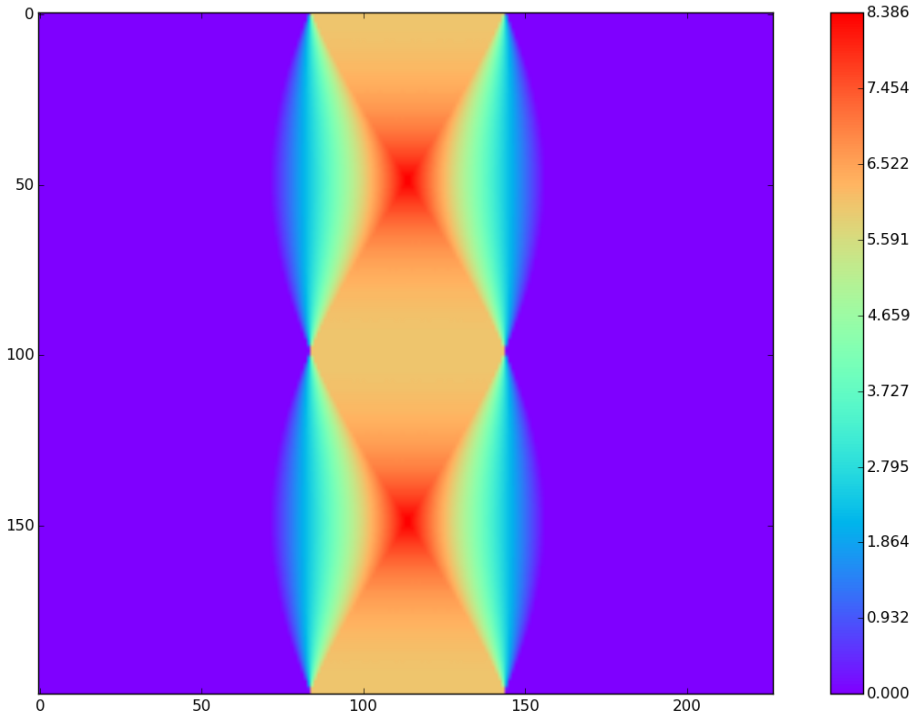
Python script for visualization.

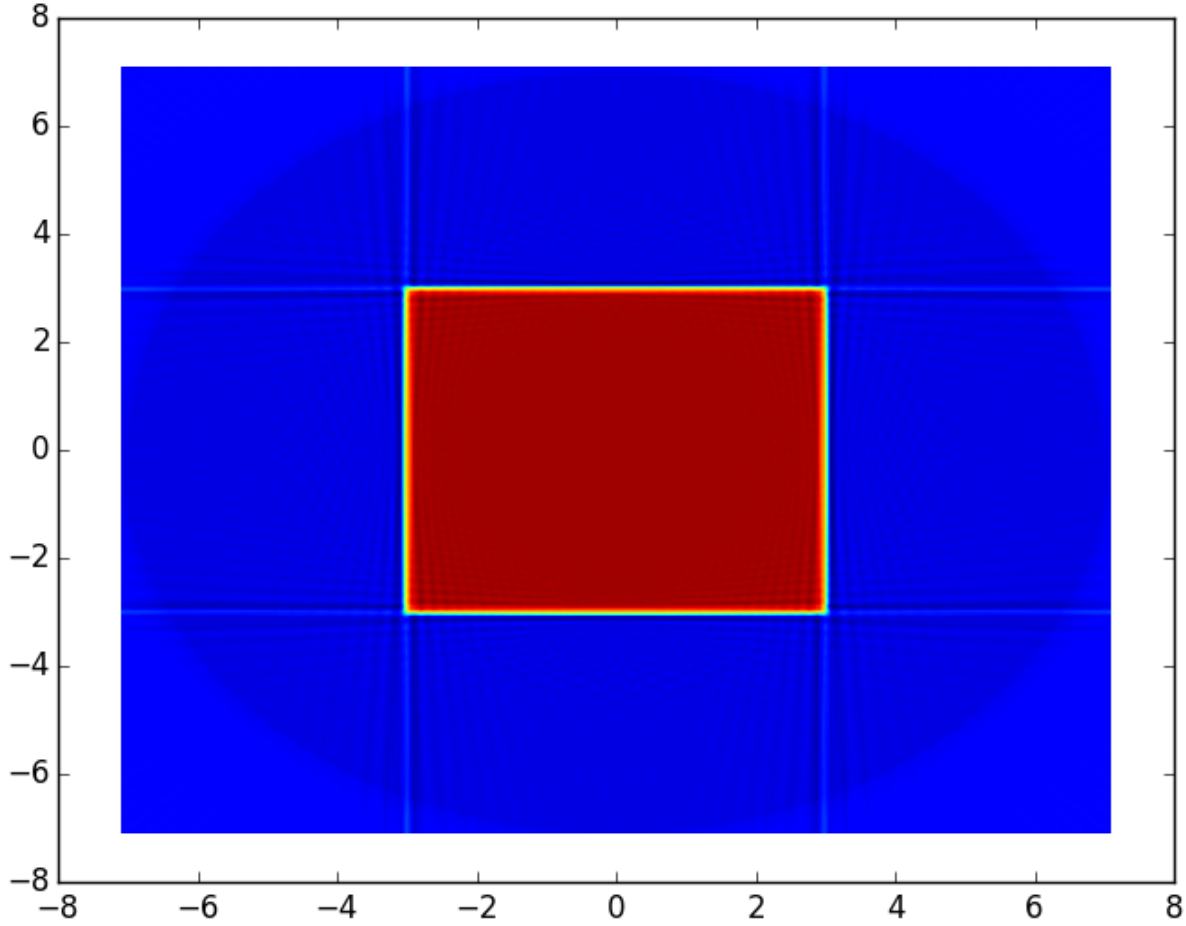
1. simple script for ascii from stdout
2. dedicated script for producing png from hdf5
3. python+ffmpeg for movie!
4. mayavi+VTK for 3D isosurface.

```
ImageArray array;
AnaSurface test_sf (function, range, range);
LineIntegral* l = new Trapezoid();
for(int i=0; i<size; i++){
        array.PushBack(angle[i], test_sf.GetProjection(l,angle[i],0.1));
}
array.Print();                  // inspect array if necessary.
array.PrintSinogram();    // print out the sinogram if necessary.
NumSurface* sf = FilteredBackProjection(array,Nres,Hamming);
sf.Print();                 // print out the result.
#ifdef USE_HDF
    sf->ExportHDF(argv[2]);
#endif
```

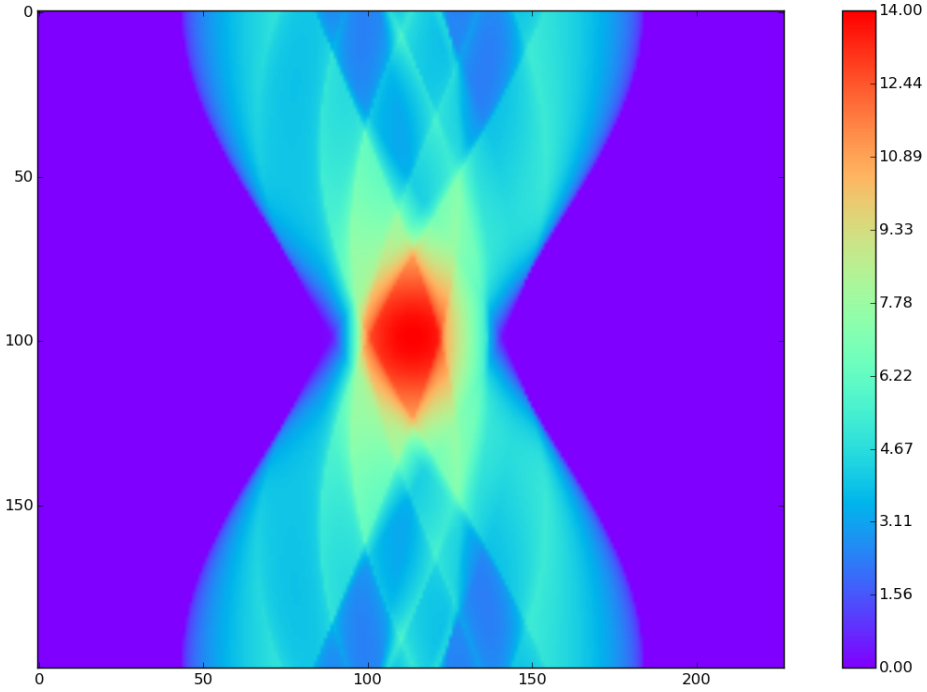
```
NumSurface* FilteredBackProjection( ImageArray& array, int Nres, double
(*kernal)(int,double))
{
   array.ConvolveWithKernal(kernal);
   NumSurface* rec = new NumSurface(Nres,range,Nres,range);
   int Nangle = array.GetSize();
   for(int ll=0; ll<Nangle; ll++)
      for( i , j loop over all x-y points in rec)
            double t = x*cos(angle) + y*sin(angle);
                                 // Distance from (x,y) to origin at angle ll.
            (*rec)(i,j) += (array.GetFilteredCurve(ll))(t,0)*pi/Nangle;
         }
   return rec;
}
```

What's this?


0
50
100
150
0
50
100
150
200
8.386
7.454
6.522
5.591
4.659
3.727
2.795
1.864
0.932
0.000


8
6
4
2
0
−2
−4
−6
−8
−8
−6
−4
−2
0
2
4
6
8

Guess what this is?



intentionally left blank

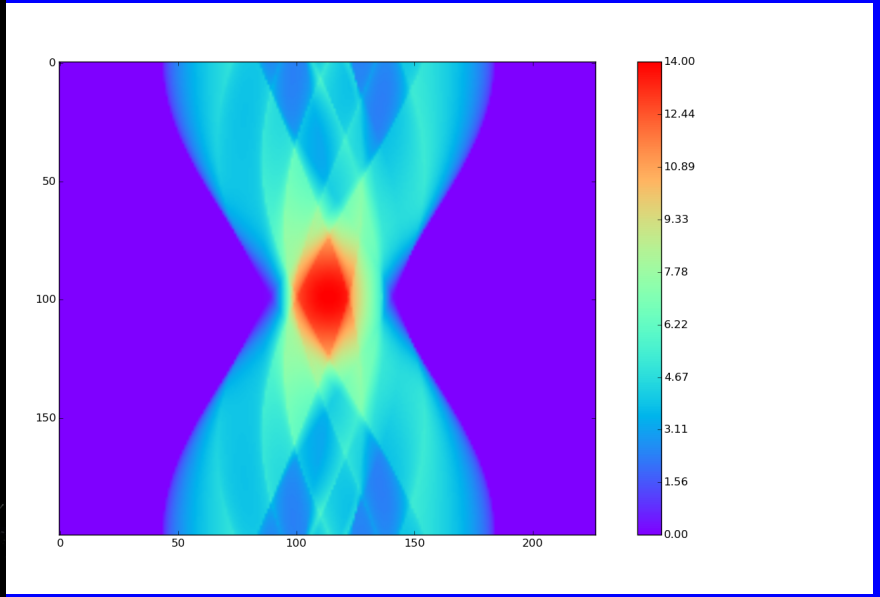
intentionally left blank

intentionally left blank

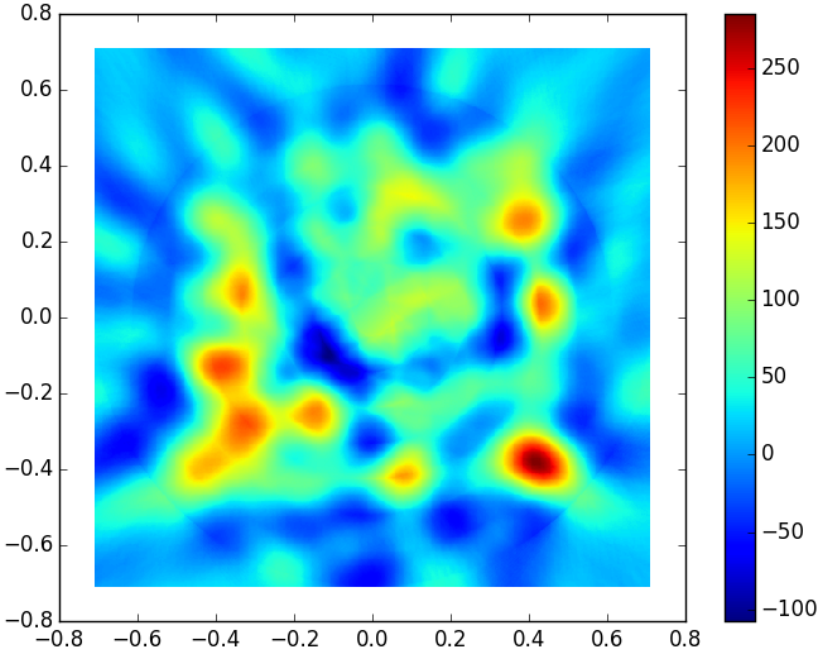
intentionally left blank

intentionally left blank

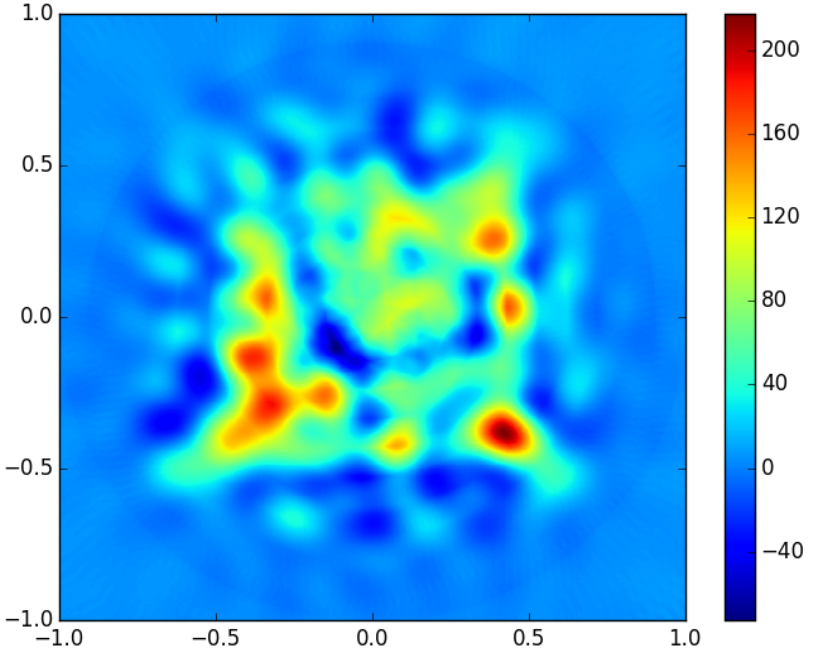
Let’s see how Dark Knight rises!



Input data



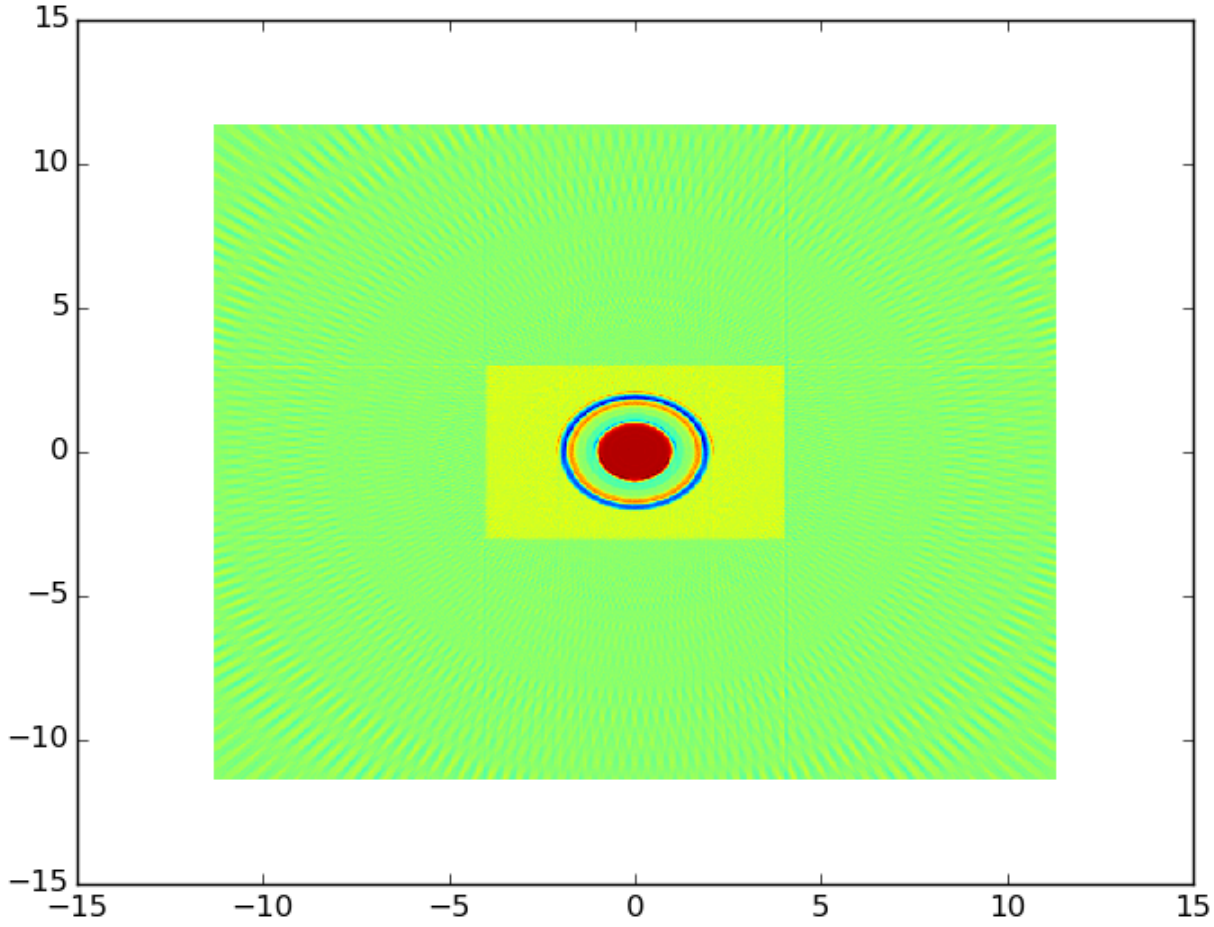
Output data

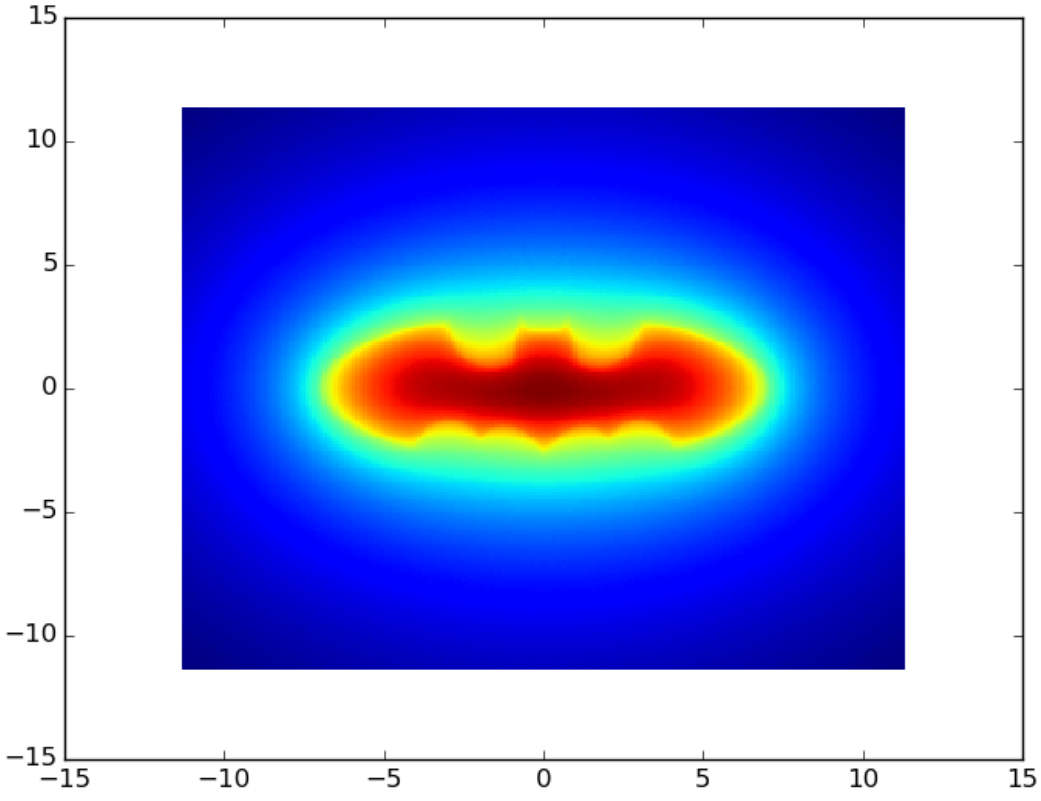


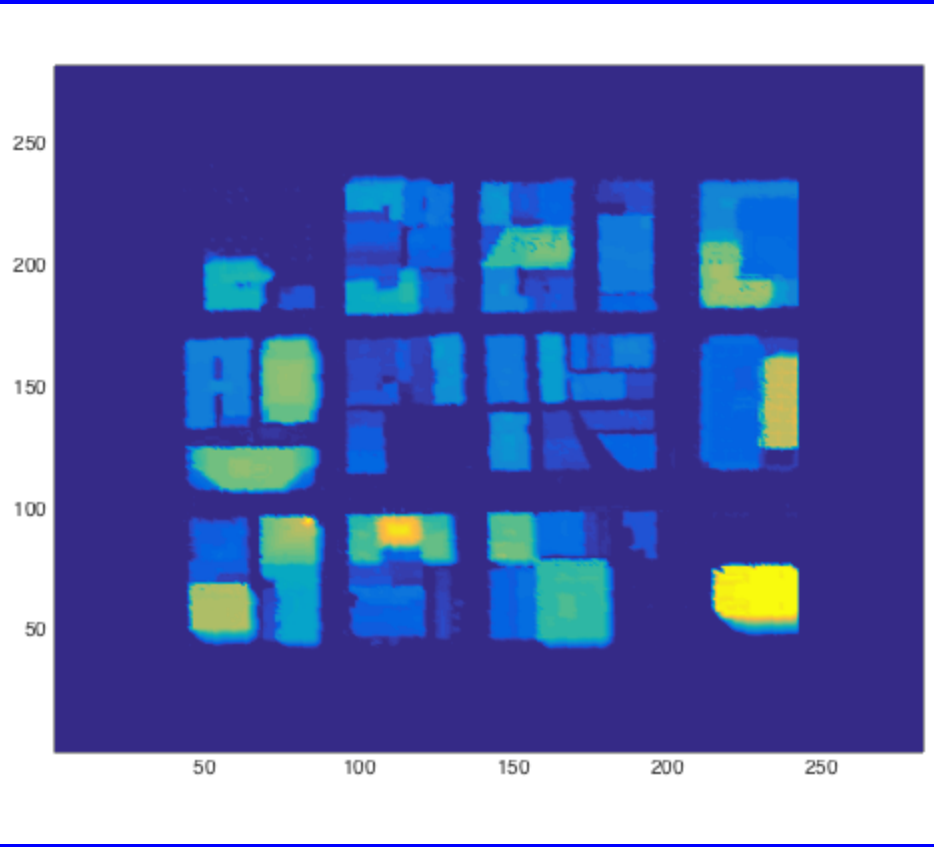
Conclusion

- We have provided an interface for manipulating images for tomographic reconstruction.
- Currently the program is designed to handle numerical and analytical 2D and 3D data.
- Users can write programs to interface their data to ImageArray such that all features will be available.
- In the future, more reconstruction algorithms and compute line integral and projection in parallel.

Back-up slides


15
10
5
0
−5
−10
−15
−15
−10
−5
0
5
10
15


15
10
5
0
−5
−10
−15
−15
−10
−5
0
5
10
15


250
200
150
100
50
50
100
150
200
250