I. Introduction

```
---
Title : I. Introduction
format:
    html:
        code_fold:true
---
```

```
using Pkg, Images, TestImages, CoordinateTransformations, Rotat
using BenchmarkTools, Images, Plots
#Pkg.activate("//Users/jiyong/External/GPUserver/development/Pr
# Pkg.activate("/home/jiyong/development/Project/nTomo.jl")
using nTomo
```

1. 설치

우선 julia 언어를 설치하고 난 뒤

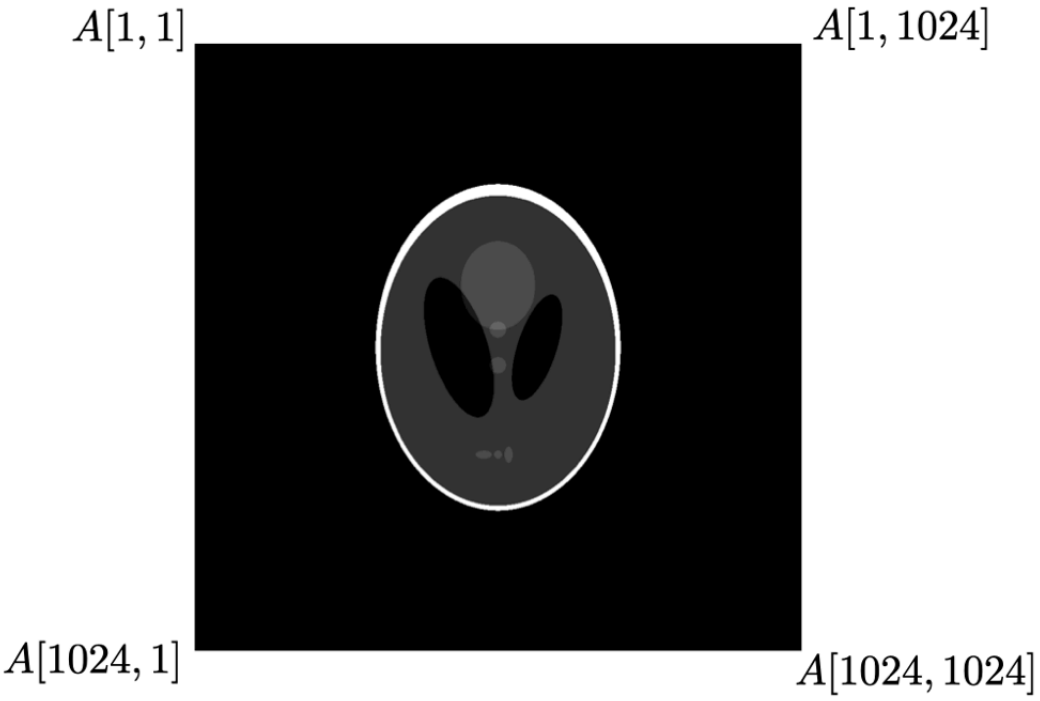
```
using Pkg
Pkg.add("https://github.com/Julia-KAERI/nTomo.jl.git")
```

를 이용해 `nTomo.jl` 을 설치한다. 이 과정은 `nTomo.jl` 뿐만 아니라 `nTomo.jl` 이

2. nTomo.jl 의 좌표계

이미지의 좌표계

- `nTomo` 는 julia 언어 위에서 동작하는 패키지이다. 모든 이미지 데이터는 julia 의 `Array` 타입(혹은 `Matrix` 타입) 이며 `A` 가 이 이미지의 값을 저장하는 `Array` 변수라면 `A[i, j]` 형식으로 접근한다.
- 기본적으로 행렬에서의 인덱스와 같은 방식을 사용하며, 따라서 i 는 세로축의 인덱스, j 는 가로축의 인덱스이다.
- Julia 는 C 나 Python 과는 다르게 인덱스가 1 부터 시작한다. 2차원 배열 `A` 를 이미지로 표현하면 좌상단에 `A[1, 1]` 이 오며 $m \times n$ 2차원 배열일경우 `A[m, n]` 이 우하단이다. 아래의 그림을 보라.

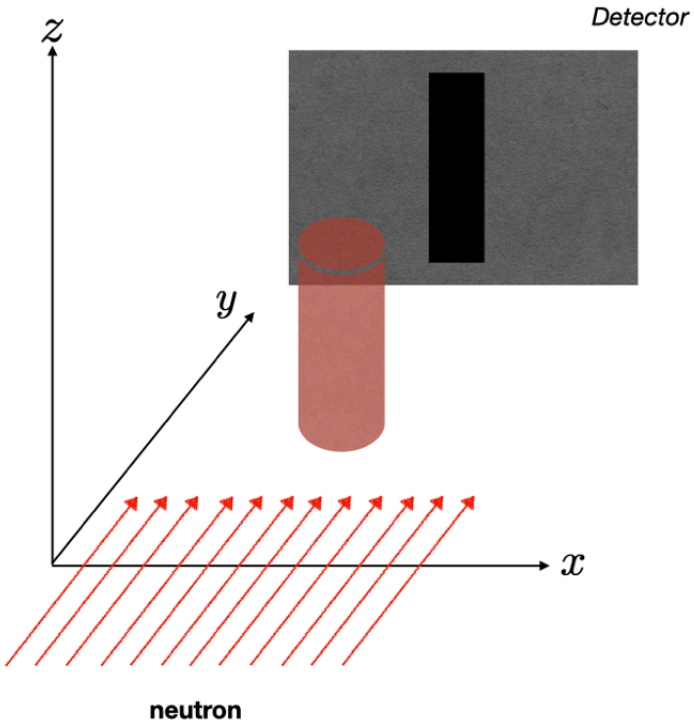


Shepp-Logan phantom

- 이 좌표계는 의외로 혼란을 줄 수 있는데 보통 2차원의 좌표를 표현할 때 (x,y) 형식으로 표현하며 이 때 먼저 오는 x 가 수평축이기 때문이다. 그러나 배열 인덱싱은 수직축을 먼저 표현한다. 또한 xy 평면상에 데이터를 그림으로 표현할 때는 오른쪽이 x 가 증가하는 방향, 위쪽이 y 가 증가하는 방향이 선호되는데 이미지를 그림으로 표현할때는 아래쪽이 첫번째 인덱스가 증가하는 방향이다.

토모그래피 좌표계

중성자 토모그래피는 평행빔 토모그래피이다. 여기서는 아래 그림과 같은 좌표계를 사용한다.

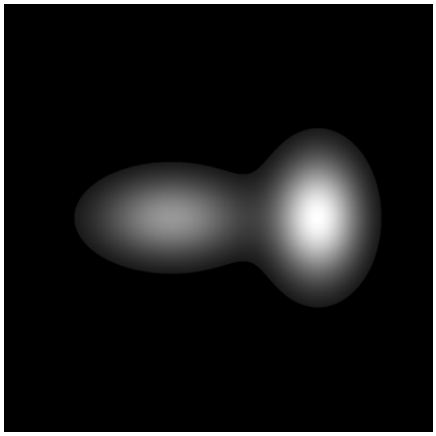


Geometry of tomography

- 중성자 빔은 y 축과 평행하다.
- 중성자 검출기는 $y = 0$ 에 의해 결정되는 xz 평면과 평행하다. 실제의 검출기는 렌즈나 광학에 의해 다른 위치에 있을지라도 검출기가 관측하는 평면이 xz 평면과 평행하다는 의미이다.
- 토모그래피의 회전축은 z 축과 평행하다.
- 회전각은 반시계방향의 각이 양수이다.

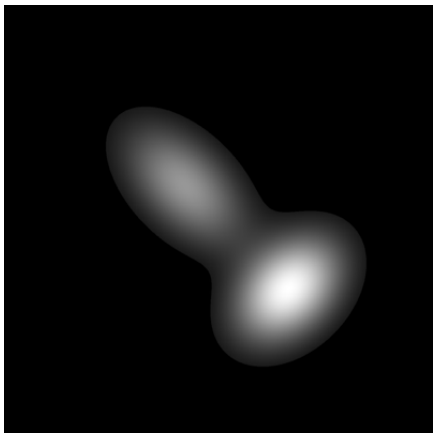
3. 라돈 변환과 역 라돈 변환

다음의 그림을 보자. 이미지가 0 에서 1 사이의 값을 가지며 검은색이 0, 흰색이 1 이라고 하자. 이 그림은 실제로 회색조 1024×1024 크기의 이미지이다. 이 이미지는 `f` 라는 배열의 변수로 저장되었다고 하자. $f(x, y)$ 라는 함수 나 `f[i, j]` 라는 형식의 배열로 이해해도 상관 없다.



object

이 이미지를 θ 만큼 회전 시킨 이미지를 $\mathfrak{R}_\theta[f]$ 라고 하자. 아래 그림은 $\mathfrak{R}_{\pi/4}[f]$ 이다. 회전방향은 시계방향이다.



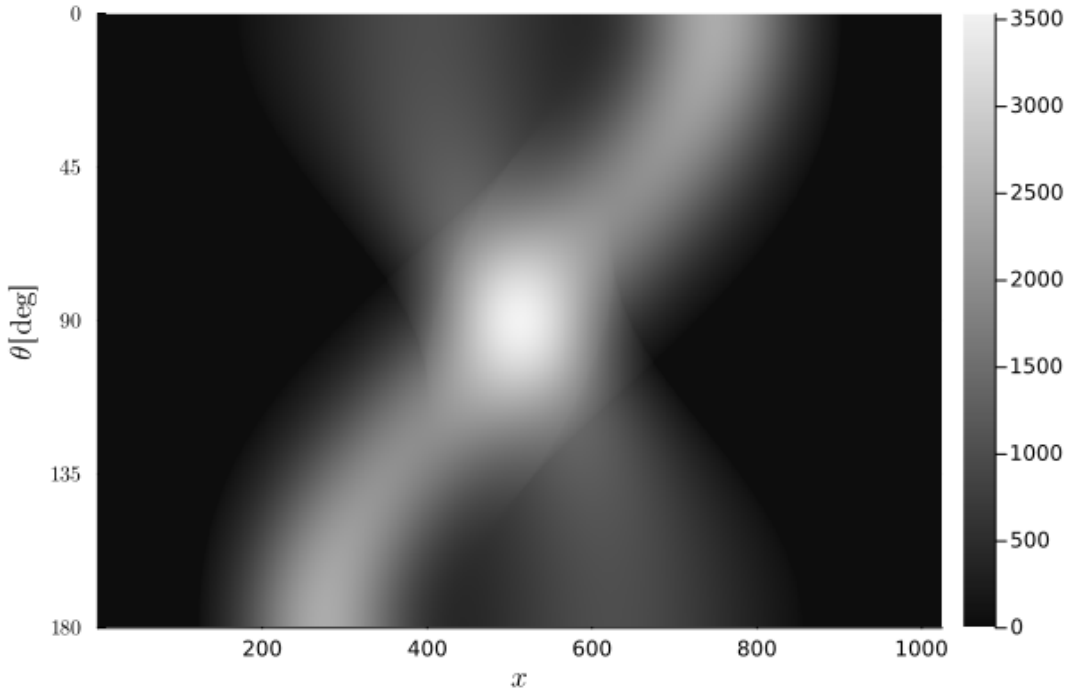
rotated object

Sinogram

아래와 같이 정의된 $S(x,\,\theta)$ 를 f 에 대한 **sinogram** 이라고 한다.

$$S(x,\theta) = \int \mathfrak{R}_\theta[f](x,y)\,dy$$

즉 sinogram 은 2 차원 이미지, 혹은 함수를 회전시켜가며 한 방향(여기서는 y 방향)에 대한 선적분을 구하였을때 나오는 다른 방향과 회전각도에 대한 2차원 함수를 의미한다. 위의 그림에 대한 Sinogram 은 다음과 같다.



sinogram

이 때 고정된 θ 에 대한 $S(x, \theta)$ 를 **projection** 이라고 하고 $P_\theta(t)$ 라고 표기한다. 즉 $P_\theta(t) = S(t, \theta)$ 이다. 이미지로부터 sinogram 을 얻는 것을 **라돈 변환 (Randon transformation)** 이라고 한다. Radon 은 오스트리아의 수학자 [Johann Karl August Radon](#) 을 의미한다.

Reconstruction 과 역 라돈 변환

Reconstruction 은 sinogram 으로부터 원래의 이미지를 구성하는 것을 말한다. Fourier slice theorem (projection slice theorem 혹은 central slice theorem) 은 수학적으로 simogram 으로부터 원래의 이미지를 구성할 수 있다는 것을 보장한다. 이미지로부터 sinogram 을 얻는 것을 라돈 변환이라고 하듯이 sinogram 으로부터 이미지를 얻는 것을 역 라돈 변환 (inverse Radon transformation) 이라고 한다.

4. 'nTomo.jl` 을 이용한 라돈 변환과 역 라돈 변환

Shepp-Logan Phantom 과 sinogram

널리 사용되는 *Shepp-Logan* 팬텀을 이용하여 라돈변환과 역 라돈 변환을 수행해보자. `nTomo.jl` 의 `phantom_shepp_logan` 함수를 이용하여 Shepp-Logan 펜텀 에 대한 2차원 배열을 생성한다. `mat2gray` 함수는 2차원 배열을 2차원 이미지로 변환한다.

`phantom_shepp_logan` 함수는 2개의 정수를 인자로 받는다. 앞의 정수는 phantom 의 크기이고, 뒤의 정수는 이미지에 주어진 정수만큼 크기의 경계를 추가한다. 아래의 `phantom_shepp_logan(600, 212)` 함수는 크기 600 의 Shepp-Logan 팬텀의 위/아래/오른쪽/왼쪽 에 각각 크기 212 만큼의 0 의 값을 갖는 경계를 추가하여 이미지 크기가 $1024 \times 1024 (= 600 + 2 \times 212)$ 가 된다.

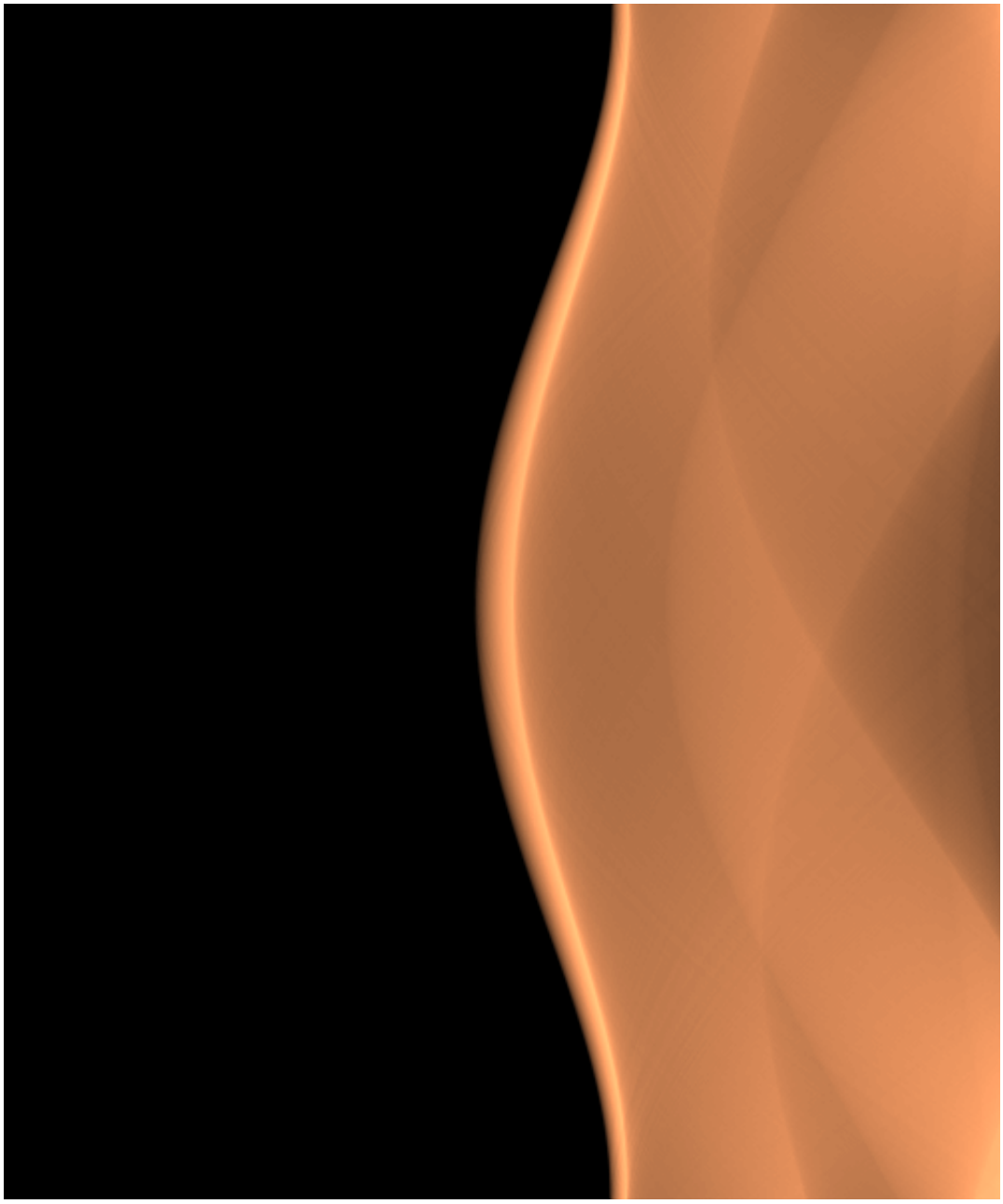
```
img = phantom_shepp_logan(600, 212)
s = mat2gray(img)
# save("shepp_logan.png", s)
```

`nTomo.jl` 은 jupyter notebook 에서 이미지를 출력할 때 `Images.jl` 에서 지원하는 이미지 출력을 사용한다. `Plots.jl` 등에서 사용하는 heatmap 보다 출력이 훨씬 빠르기 때문이다. 대신에 숫자의 배열을 픽셀의 배열로 바꾸는 작업이 필요한데 이것은 `nTomo.jl` 의 `mat2gray` 함수로 처리한다.

라돈 변환

이제 라돈 변환을 얻어보자. Julia 에서 `1:100` 은 1 부터 1 씩 증가하여 100 이 넘지 않는 수까지의 배열을 의미한다. 따라서 $1, 2, \ldots, 99, 100$ 이다. 이와 비슷하게 `0:0.3:179.9` 는 0 부터 0.3 씩 증가하여 179.9 보다 크지 않은 수까지의 배열이다. 즉 $0.0, 0.3, 0.6, \ldots, 179.1, 179.4, 179.7$ 이다. 정확히는 배열은 아니지만 배열과 같은 기능을 한다. `nTomo.jl` 에서 라돈변환은 `radon(img, ths, (cx, cy))` 로 수행한다. `img` 는 라돈변환을 수행하고자 하는 2차원 배열이며 `ths` 는 라돈 변환을 수행하는 각도이다. 각도는 라디안이 아닌 $^\circ$ 로 입력되어야 한다. `(cx, cy)` 는 회전 중심을 의미한다. `colorize` 함수는 이미지에 특정한 스펙트럼의 색을 칠하는 함수이다. 시노그램은 세로축이 각이고 가로축이 검출기 픽셀이며 좌상단이 시작 각도와 시작 픽셀을 의미한다.

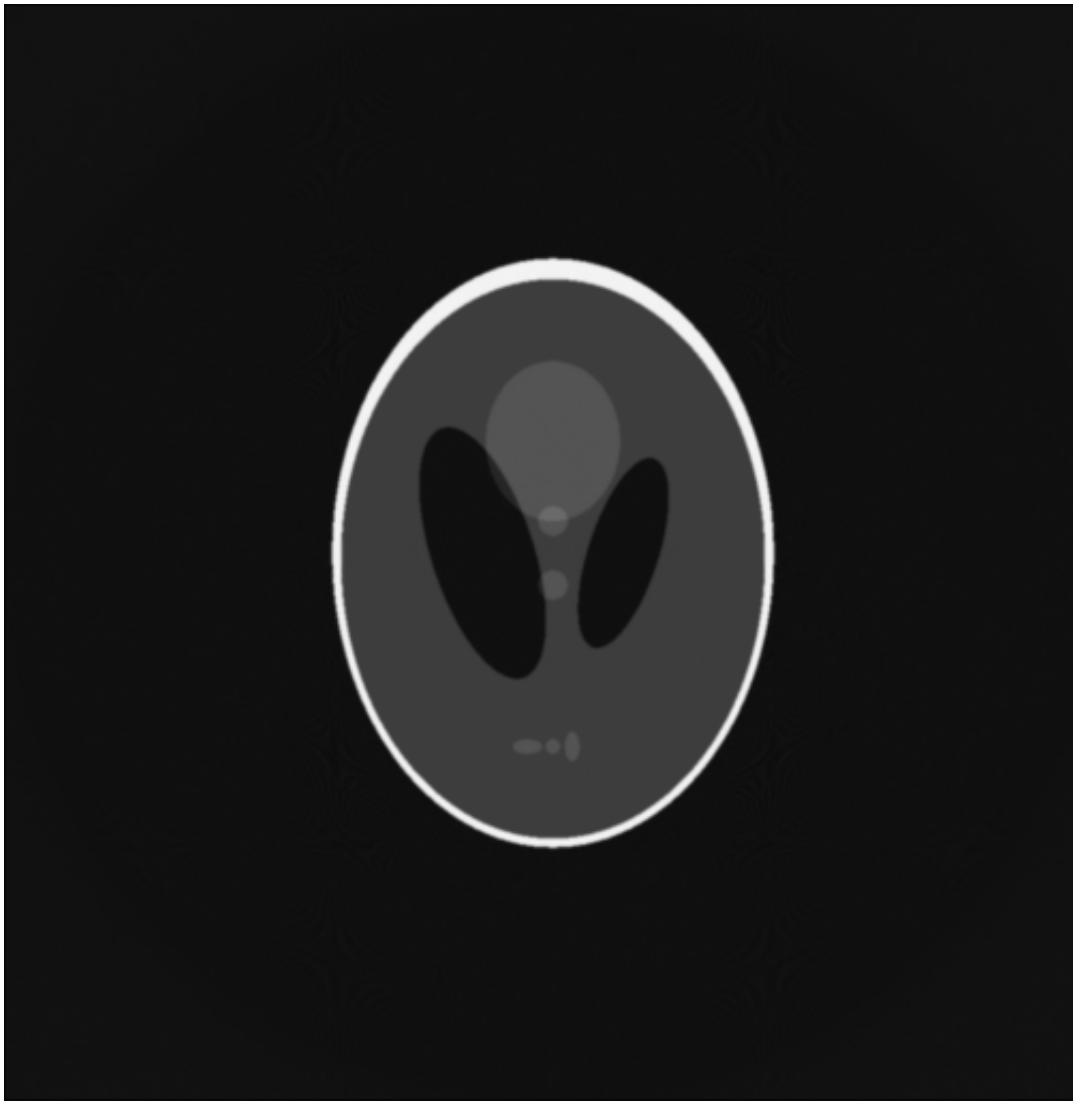
```
ths = 0.0:0.3:179.9
sino1 = radon(img, ths, (512, 512));
colorize(sino1, :copper)
```



5. Tomography reconstruction

위에서 생성한 Shepp-Logan 팬텀에 대한 시노그램을 이용하여 reconstruction, 즉 역 라돈 변환을 수행하도록 하자. 현재 `nTomo.jl` 은 filtered back projection(이하 FBP) 방법과 reconstruction 과 simultaneous algebraic reconstruction technique(이하 SART) 방법을 지원한다. FBP 를 사용하는 역 라돈 변환은 `iradon_fbp` 함수를 사용하며 SART 를 사용하는 역 라돈 변환은 `iradon_sart` 를 사용한다.

```
# Filtered back projection
rec1=iradon_fbp(sino1, ths, 512, "hann")
mat2gray(rec1)
```



```
# Simultaneous Algebraic Reconstriction
rec1=iradon_sart(sino1, ths, image = nothing, center = 512)
mat2gray(rec1)
```

일반적으로 FBP 가 SART 에 비해 훨씬 빠르지만 노이즈에 취약하고 여러 artifact 가 생길 수 있다.