Exact sinogram: an analytical approach to the Radon transform of phantoms

Monica Dessole [1], Marta Gatto [1], Davide Poggiali [2], Francesca Tedeschi [1]

[1] Department of Mathematics "Tullio Levi Civita", University of Padova, Italy. [2] PNC - Padova Neuroscience Center, Italy.

Motivations

Tomography produces a projection image of the inaccessible regions of a body. The regions are determined by their **attenuation function**, that quantifies how the intensity of radiations is absorbed due to the mass. It is reconstructed through the Radon transform, whose graphic representation is the **sinogram**, and its inverse. Numerical methods for image reconstruction are tested on the so-called **phantoms**, fictitious images that reproduce a body section, as the Shepp-Logan phantom [3].

In some special cases, it is possible to compute analytically the Radon transform of a phantom.

The *scope of the work* is to compare the discrete Radon transform, that yields an approximate sinogram, and the correspondent analytical sinogram for image reconstruction.

We created `exact-sinogram` , an open-source python library of Analytical Radon transforms for different classes of phantoms (combination of ellipses, squares or rectangles) in order to successively reconstruct the original image and to analyze the error.

Mathematical tools

Let the attenuation function $f(x, y)$ be an integrable continuous function with compact support $\Omega \subset \mathbb{R}^2$. The **Radon transform** of f is instead the functional $Rf : \mathbb{R} \times [0, 2\pi) \to [0, 1]$ defined as

$$Rf(t,\theta) := \int_{l_{t,\theta}} f \, ds =$$

$$= \int_{s=-\infty}^{\infty} f(t\cos\theta - s\sin\theta, t\sin\theta + s\cos\theta)\, ds,$$

where $l_{t,\theta}$ is the line passing through the point $(t\cos(\theta), t\sin(\theta))$ and orthogonal to the unit vector $\mathbf{n} = (\cos(\theta), \sin(\theta))$.

In the Figure, there are the two sinograms that represent data generated by the X-ray emission/detection machine for the given slice of the sample.

The Radon transform is generated with different procedures, according to the phantom type: the method of the bands [2] with polygonal figures, and another method (next section) for phantoms with regions determined by non self-intersecting closed curves.

Phantom of ellipses

Every ellipse in the xy-plane is uniquely determined by six real parameters: semi-axes (a, b), center coordinates (x_0, y_0), angle of orientation (ϕ), and the value of the attenuation (δ) associated to the inside of the ellipse. The coordinates in the roto-translated frame are $\hat{x} = (x - x_0)\cos\phi + (y - y_0)\sin\phi$ and $\hat{y} = (y - y_0)\cos\phi - (x - x_0)\sin\phi$.

Let ϵ be a generic ellipse and for real numbers t and θ, let pose $\hat{\theta} = \theta - \phi$ and $\hat{t} = t - x_0\cos\theta - y_0\sin\theta$, its attenuation function is assumed to be

$$f_\epsilon(x,y) = \begin{cases} \delta & \frac{\hat{x}^2}{a^2} + \frac{\hat{y}^2}{b^2} \le 1, \\ 0 & \text{otherwise.} \end{cases}$$

The Radon transform of f_ϵ is obtained computing analytically the intersections between the line $l_{t,\theta}$ and the ellipse, making the difference between them, i.e. their distance, and is expressed as

$$Rf_\epsilon(t,\theta) = \begin{cases} \delta \cdot \frac{2ab\sqrt{b^2\sin^2\hat{\theta} + a^2\cos^2\hat{\theta} - \hat{t}^2}}{b^2\sin^2\hat{\theta} + a^2\cos^2\hat{\theta}} & \hat{t}^2 \le b^2\sin^2\hat{\theta} + a^2\cos^2\hat{\theta}, \\ 0 & \text{otherwise.} \end{cases}$$

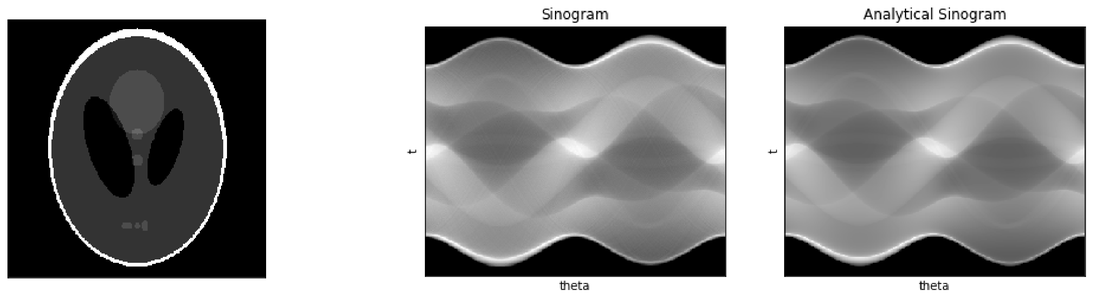

Figure: Phantom of ellipses (left), its approximated (center) and exact (right) sinograms.

The library

A phantom is an instance of the class `Phantom`, initialized with the phantom type (`'ellipses'`, `'squares'`, `'rectangles'`) and a matrix containing the parameters of the figures. We also provide a gallery of default phantoms accessible by the call of the function `my_phantomgallery`. The methods of the class are: `get_phantom`, that returns the image of the phantom, and `get_sinogram`, that generates the analytical Radon transform.

```python
import numpy as np
from skimage.transform import radon, iradon
import exact_sinogram as es

circle = True
phantom_type = 'ellipses'
#  phantom_type : 'ellipses' (or 'shepp_logan') , 'modified_shepp_logan', 'squares', 'rectangles'
n_points = 300;   # number of pixels

# Creation of an instance of the Phantom Class
Phm = es.Phantom(phantom_type = phantom_type, circle = circle)
# Creation of the matrix-image of the phantom, w/ input the number of pixel:
P = Phm.get_phantom(N = n_points)
# Array of projection angles
theta_vec_deg = np.linspace(0, 359, 360)
# Exact sinogram and inversion (exact_sinogram package)
## Calculate the excact Sinogram
analytical_sinogram = Phm.get_sinogram(N = n_points, theta_vec = np.deg2rad(theta_vec_deg))
## invert the Radon transform on the exact sinogram
P_an = iradon(analytical_sinogram, theta = theta_vec_deg, circle=circle)
# Approximated sinogram and inversion (scikit-image)
## Calculate the approximated sinogram
approx_sinogram = radon(analytical_sinogram, theta = theta_vec_deg, circle=circle)
## invert the Radon transform on the approximated sinogram
P_approx = iradon(approx_sinogram, theta = theta_vec_deg, circle=circle)
```

	Ellipses (Shepp-Logan)	Ellipses (Modified Shepp-Logan)	Squares	Rectangles
$\frac{\|P_{an}-P\|_2}{\|P\|_2}$	0.03073	0.02590	0.02703	0.01838
$\frac{\|P_{inv}-P\|_2}{\|P\|_2}$	0.03604	0.03191	0.01997	0.02588

Table: Relative errors

Results and conclusions

For each phantom type, we compute the error between the synthetic data and the reconstructed data obtained by the following procedures:

1. by computing the analytic sinogram and then applying the inverse Radon transform;
2. in the classic way, by using the Radon transform and its inverse.

We are going to evaluate the quality of the reconstructed image by measuring the classical 2-norm and we collect the results in the Table: we show relative errors of the image reconstructed from the analytical sinogram P_{an} and the image reconstructed from the numerical sinogram P_{inv} with respect to the original image P, taking into account Gibbs phenomenon. The Gibbs phenomenon [5] is the concentration of the errors along the discontinuities of the figure. With a mask, we eliminated the influence of boundaries and appreciated our algorithm that gives a smaller relative error with respect to the reconstruction through the `iradon` function.

Info

- `https://github.com/francescat93/Exact_sinogram`
- `pip install exact_sinogram`
- dependencies: numpy, scipy, scikit-image, matplotlib

References

[1] A. M. Cormack. Representation of a function by its line integrals, with some radiological applications.Journal of Applied Physics (U.S.), 34, 9 1963.

[2] T. G. Feeman: The Mathematics of Medical Imaging. UNITEXT 92. Springer, 1 edition, 2016

[3] The Fourier reconstruction of a head section.IEEE Transactions on NuclearScience, 21(3):21âĂŞ43, 1974.

[4] J. Radon. On the determination of functions from their integral values along certain manifolds. IEEE Transactions on Medical Imaging, 5(4):170-176, 1986.

[5] A. J. Jerri.The Gibbs Phenomenon in Fourier Analysis, Splines and Wavelet Approximations. Mathematicsand Its Applications 446. Springer US, 1 edition, 1998.