1 Methods

As we know from the physics of computed tomography, the measured sinogram image is convolved with several sources of noise, notably electronic noise and structural noise**?**. Hence, using an iterative method that handles noise is necessary. In formulating the CT reconstruction as an optimization problem, one of the important considerations is to find a regularization method for the ill-posed problem that preserves the edges in the true image, since identifying boundaries of objects (such as the boundary between normal and abnormal tissues) has high diagnostic value.

We decided to follow the fast gradient-based algorithm proposed by Beck and Teboulle**?**, and the following section will discuss the specific implementation of their method for CT image reconstruction.

Specifically, we chose isotropic TV regularization[1]. The original publication of FISTA with TV**?** describes two types of TV regularization, anisotropic and isotropic TV, and the specific python package of choice, pyunloc**?**, implements isotropic TV functionals. We decided to use the implemented TV function in the package and conducted some research on the differences of those two types of TV regularization functions. As being described in **?**, anisotropic TV is known to favor horizontal and vertical structures for 2D images, and isotropic TV is shown to be a superior choice for regularization for a 2D image that preserves the off-axis edges.

The optimization problem can be formulated as:

$$\min_x \{||A(x) - d||^2 + 2\lambda \mathrm{TV}(x)\} \tag{1}$$

where $x \in \mathbb{R}^{m\times 1}$ is the flattened true image with the size $\sqrt{m \times 1}$. $d \in \mathbb{R}^n$ is the flattened sinogram image where $|rays| \times |views| = n$. $A \in \mathbb{R}^{m\times n}$ is the forward operator that transforms an image to a sinogram image. λ is the regularization parameter. $\mathrm{TV}(x) : \mathbb{R}^{m\times 1} \to \mathbb{R}$ is an istropic total variation functional, where

$$x \in \mathbb{R}^{m\times 1}, \mathrm{TV}(x) = \sum_{i=1}^{m-1} |x_i - x_{i+1}| \tag{2}$$

To solve the problem in eq.**??**, it requires a method that sovles non-differentiable optimization problems. As discussed in **?**, the gradient projection method is particularly suited for this task because of the faster convergence rate compare to other methods such as primal-

[1]In our presentation, we claimed that we were using the anisotropic TV because we falsely assumed that we were optimizing an 1D vector rather than a 2D image. For the 1D case, isotropic and anisotropic TV are essentially the same. The error was resulted from the fact that the majority of the group members are not in the field of imaging science as are unfamiliar with how matlab implicitly handles an image. And the error of assuming a 1D vector resulted in thinking we can use anisotropic TV as well as creating vertical artifacts in reconstructed images, we have corrected the errors in the implementation and rewritten this section. This has been a good learning opportunity for us both in terms of learning new knowledge as well as learning the process of gaining such knowledge from outside of the field. We also attempted to rewrite the method section to increase clarity, we would appreciate any feedback on this version of the description of the method.

dual Newton-based methods**?**. The specific innovation of **?** is a new proximal gradient method called fast iterative shrinkage/thresholding algorithm (FISTA), which has a global convergence rate of $\mathcal{O}(\frac{1}{k^2})$ (The proof of convergence rate is detailed in **?**).

The general principle of the algorithm will be discussed in this paragraph and the pseudocode for the specific implementation will follow. The first key insight is the construction of the new FISTA term.
The general principle of the algorithm will be discussed in this paragraph and the pseudocode for the specific implementation will follow. The first key insight is to apply a orthogonal projection operator P_C, which is a box projection that maps x onto a feasible set which ensures the smoothness of the eq.**??** in the projected space. As discussed in **?**, the solution is

$$x_k = \mathrm{P}_C(x_{k-1} - t_k \nabla f(x_{k-1})) \tag{3}$$

where f denotes the data fidelity term $\min\limits_{x}\{||A(x) - d||^2$.

The next step is to construct the proximal map. Let $\mathcal{P}_1$ denote the set of all pairs of matrices $(\mathbf{p}, \mathbf{q})$. Note that for our special case where $x \in \mathbb{R}^{m\times 1}$, $\mathcal{P}_1$ becomes matrices $\mathbf{p}$ satisfying

$$|p_i| \leq 1, i = 1, \cdots, m-1 \tag{4}$$

Following *Proposition 4.1* in **?**, we can write out the objective function as

$$\max_{\mathbf{p}\in\mathcal{P}} \min_{x\in\mathbb{C}} \{|||x - d|||_{\mathrm{F}}^2 + 2\lambda \mathrm{Tr}(\mathcal{L}(\mathbf{p}^{\mathrm{T}} x))\} \tag{5}$$

and the optimal solution of the inner minimization problem is

$$x = \mathrm{P}_C(d - \lambda \mathcal{L}(\mathbf{p})) \tag{6}$$

where $\mathcal{L} : \mathbb{R}^{(m-1)\times 1} \to \mathbb{R}^{m\times 1}$ is a linear operator defined by

$$\mathcal{L}(\mathbf{p})_i = p_i - p_{i-1}, i = 1, \cdots, m-1 \tag{7}$$

Plugging in the objective function, gradient and step size (which is an upper bound on the Lipschitz constant, and in term depends on the regularization parameter), we can write out the pseudocode for the algorithm, see algorithm.**??**.

We found an existing package in python called pyunlocbox **?** to implement the algorithm. This package has all the required components for constructing the functions for data fidelity and regularization terms, and has implemented the FISTA version.
Now we discuss other implementation choices in this algorithm. First, we decided to use relative tolerance as our stopping criterion since it is easy to implement and measures the change in value relative to the size of the values themselves. Second, we used L-curve criterion to decide the regularization parameter, because it gives us a principled way to decide the

Algorithm 1 TV-FISTA

Input d, A, λ, $maxiter$, tol
$r_0 = p_0 = 0_{m-1}$, $t_1 = 1$
if $i \leq maxiter$ **then**
Forward
Compute $x_f = \frac{2}{\lambda} A^T(A(x) - d)$
Backward
$x_i = x_f - \frac{1}{\lambda} \boldsymbol{\nabla} \cdot x_f$
$obj = \frac{1}{2}||x_f - x_i||^2 + \lambda||\nabla x_i||^2$
$tol_i = \frac{|obj_{i-1} - obj_i|}{obj_i}$
Update projector
$r_i = r - \frac{1}{8\lambda} \nabla x_i$
FISTA
$t_{i+1} = \frac{1+\sqrt{1+4t_i^2}}{2}$
$p_i = \frac{r_1}{\max\{1, p_i\}}$
$r_i = p_i + \frac{t_i - 1}{t_i * (p_i - p_{i-1})}$
if $tol_i \leq tol$ **then**
Stop
$i = i + 1$

proper amount of regularization without relying on the reference image.

In general, we made an iterative plan for implementing this algorithm. We started from a series of simple choices, such as optimizing in the image domain with image represented as a row vector (in contrast to wavelet domain), the anisotropic TV, and relative tolerance. While those choices are unsophisticated compare to many advanced tools, we reasoned that we could adjust our implementation choices if the reconstruction quality is poor. To our surprise and delight, the first try of the reconstruction quality is good. We hence adhered to our implementation choices in regularization function and stopping criterion. The detailed reconstruction result will be presented in the result section.