***NOTE : all the notation and the symbols is as same as the paper [A Fast Algorithm for Convolutional Structured Low-Rank Matrix Recovery]***

# 1. using the same method to derive the $D$

Not involve in this derivation. The one who has read the paper [A Fast Algorithm for Convolutional Structured Low-Rank Matrix Recovery] will easily understand it, 'cause 3d is just a straight extension of 2d optimization.

# 2. solving the least square optimization problem

The problem has the form of

$$\min_x ||SF_t^* x - b||_2^2 + \lambda \sum_i^{x,y,t} ||D^{\frac{1}{2}} F^* M_i x||_2^2 \tag{1}$$

$S$ is the under-sampling factor matrix; $F$ is the 3d Fourier transform; $F^*$ is the inverse 3d Fourier transform; $F_t^*$ represents the inverse Fourier transform of the $t$ dimension; $x$ is the Fourier area data of the 3d MRI image; $b$ is the observation of $k$-space, which results in adding the factor of $F_t^*$ different with the original 2d problem.

NEXT, using ADMM. Let $g = x$ and $Fy_i = M_i g$. The optimization (1) turns to

$$\begin{cases} x^{(k+1)}, g^{(k+1)}, y_i^{(k+1)} = \arg\min_{x,g,y_i} ||SF_t^* x - b||_2^2 + \lambda \sum_i^{x,y,t} ||D^{\frac{1}{2}} y_i||_2^2 + \beta_1 \sum_i^{x,y,t} ||Fy_i - M_i g - l_i^{(k)}||_2^2 + \beta 2 ||g - x - q^{(k)}||_2^2 \\ l_i^{(k+1)} = l_i^{(k)} - (Fy_i^{(k+1)} - M_i g^{(k+1)}) \\ q^{(k+1)} = q^{(k)} - (g^{(k+1)} - x^{(k+1)}) \end{cases}$$

For simple the derivation, we ignore the superscript $^{(k)}$ in the following derivation.

***NOTE: all the symbols, like $x$, $b$, $g$, are treated as vectors, such that, $D$, $M_i$, and so on, can be treated as matrix, but all the operations are 3d tensor's operations.***

## step 1: solve the $y_i$ problem

The problem is

$$\min_{y_i} \lambda \sum_i^{x,y,t} ||D^{\frac{1}{2}} y_i||_2^2 + \beta_1 \sum_i^{x,y,t} ||Fy_i - M_i g - l_i||_2^2$$

We dismiss the $\sum_i^{x,y,t}$, and optimize each component

$$\min_{y_i} \lambda ||D^{\frac{1}{2}} y_i||_2^2 + \beta_1 ||Fy_i - M_i g - l_i||_2^2$$

Take the derivative with respect to $y_i$, and let it equal to zero:

$$(\lambda D + \beta_1) y_i = \beta 1 F^* (M_i g + l_i)$$

$$y_i = (\lambda D + \beta_1)^{-1} \beta 1 F^* (M_i g + l_i)$$

As same as the implement codes of the 2d GIRAF algorithm, define a number of 4d operators and tensors: $y = \{y_x, y_y, y_t\}$, $M(x) = \{M_x x, M_y x, M_t x\}$, $\lambda D + \beta_1 = \{\lambda D + \beta_1, \lambda D + \beta_1, \lambda D + \beta_1\}$, $L = l_x, l_y, l_t$, note that the curly brace includes three 3d tensors and allocates those tensors at the fourth dimension to form a 4d tensor. In that notations, we obtain

$$y = (\lambda D + \beta_1)^{-1} \beta 1 F^* (Mg + L)$$

Moreover, we further define $Y = Fy$ in order to simplify the solution followed.

## step 2: solve the $g$ problem

The problem is

$$\min_g \beta_1 \sum_i^{x,y,t} ||Fy_i - M_i g - l_i||_2^2 + \beta 2||g - x - q||_2^2$$

Take the derivative with respect to g, and let it equal to zero:

$$\beta_1 \sum_i^{x,y,t} [M_i^* M_i - M_i^* (Fy_i - l_i)] + \beta_2 g - \beta_2 (x + q) = 0$$

As same as the implement codes of the 2d GIRAF algorithm, define two 4d operators: $M^*(x) = \sum_i^{x,y,t} M_i^* x$; $M^T M = \sum_i^{x,y,t} M_i^* M_i$ is a diagonal matrix, because that $M_i$ is a diagonal matrix. Then

$$(\beta_1 M^* M + \beta_2) g = \beta_1 M^* (Y - L) + \beta_2 (x + q)$$

$$g = (\beta_1 M^* M + \beta_2)^{-1} [\beta_1 M^* (Y - L) + \beta_2 (x + q)]$$

## step 3: solve the x problem

The problem is

$$\min_x ||SF_t^* x - b||_2^2 + \beta_2 ||g - x - q||_2^2$$

Take the derivative with respect to $x$, and let it equal to zero:

$$F_t (S^* S + \beta_2) F_t^* x = F_t S^* b + \beta_2 (g - q)$$

Because $S^* S + \beta_2$ is a diagonal matrix, then

$$x = F_t (S^* S + \beta_2)^{-1} [S^* b + \beta_2 F_t^* (g - q)]$$