**Spiral TSE**

This repository contains the MatLab-code to create single shot spiral TSE-sequences using the Pulseq-framework. The software represents the status of the sequence development at the time of submission of the corresponding manuscript to Magnetic Resonance in Medicine. The main program is mySpiralTSE.m. It contains several features, which had been implemented in the course of the development of the sequence, but haven’t been used or checked recently.

It is your own responsibility to set parameters such that they comply with the specifications of your scanner as defined in the structure *system*. Chosing non compatible values for gradient timing or amplitude will lead to an error message, the code contains only very basic error handling.

Even if you don’t have Pulseq installed on the scanner, the software package offers nice tools for visualization of the sequence and its components.

**Setup Environment**

MatLab needs to be installed.  
In order to generate the sequences, the pulseq-package need to be installed. (<https://github.com/pulseq/pulseq>).

In order run the sequences on the scanner (Siemens or GE), the Pulseq-interpreter sequence needs to be installed on the scanner (for details see https://www.opensourceimaging.org/project/pulseq-open-source-pulse-sequences/)

The calculation of the spiral trajectories requires the VDS-package from B. Hargreaves (<https://mrsrl.stanford.edu/~brian/vdspiral/>).

**Sequence setup**

The sequence and related functions should be copied to …\pulseq-git\matlab\spiralTSE

Running mySpiralTSE.m should already yield a reasonable sequence, which can be run on the scanner. mySpiralTSE allows several different options, the available options can be listed with

*help mySpiralTSE*

in MatLab. Some of these options have been used in the initial stage for debugging and other purposes and I am not sure, whether they even still work. I have highlighted the options which are recommended and tested below, others, which may or may not work or are meanwhile useless are dimmed:

The sequence runs in different modes defined by :

scanmode=

'init' : initialization, creates empty k-space corrections dKA and dKE

'trim' : used for experimental trimming,

the first and every 2:decho:acqP.necho echo periods are

readout as a spin echo

**'run' : run sequence**

segmode=

**'fix' : fixed number of points per segment**

**'tan' : tangential transition to and from each segment**

'single': single echo spiral

spmode=

**'cont' : continuous segmentation**

**'alt' : alternate between spiral-out and spiral-in in odd and**

**even segments.**

initmode=

**'no' : no gradient in first half of first segment**

'rev' : first half is reverse of second half

'outin' : first half 180° rotated and out-in

'sym' : first half mirrored at y-axis and out-in;

accmode= acceleration mode

**'dd' : dual density. First seg.n\_full\*acq.nadc/2 points are fully sampled,**

**others undersampled by a factor seg.n\_acc**

**'vd' : variable density with Fcoeff2 = - 2\*Fcoeff/seg.n\_acc**

**(other) : undersampled by a factor seg.n\_acc**

fatsat : 'on' or 'off' : at higher field ‘on’ is mandatory.

Some hints for creating sequences:

System parameters are defined right at the top, they can be used to create sequences for different gradient systems.

FOV and spatial resolution are defined by

acqP.fov=240e-3;

spiral.Nx=120;

The example yields 1 mm resolution, since a spiral with 120 windings covers 240 k-space points.

spiral.kOffset=50;

defines the starting and endpoint of the trajectory in each echo interval. The number defines, how far outside of the k-space covered by the trajectory this point is placed. Spiral.kOffset has to be positive in tangential mode. If it is very large, a long time is required to traverse to the beginning of the actual spiral segment to be read out.

A crucial timing for configuration of the sequence is the time for the spoiler, which allows represents the time for getting to the beginning of each spiral segment:

segP.tSp=1.05e-3;

If you get the error message

‘trajectory not compatible with sequence parameters.

Change parameters and/or buy faster/stronger gradients’

you have to increase segP.tSp (and/or change spiral.kOffset).

The usual acquisition parameters are defined in lines 84 ff. Some explanations:

acq.accfac defines the acquisition bandwidth. Basic timing is given by the 10 s Gradientenrastertime, acq.accfac defines the dwell time in fractions thereof. 4 is a reasonable number, since the execution time is mainly determined by the gradient performance it doesn’t make a lot of sense to use higher bandwidth.

Note that the effective echo time is defined by acqP.nTE in multiples of the echo spacing (there is a commented out line in which TEeff can be explicitly defined).

For refocusing flip angles < 180° acqP.flipflag=2 is recommended to be used. Flip angles are then calculated according to TRAPS to guarantee a smooth transition into the PSS.

The script creates the sequence and also stores all relevant parameters in a .mat-file. This can be used to re-create a specific sequence and also contains all necessary information (and more) for reconstruction

**Sequence Reconstruction**

Some code used for image reconstruction as well as sample datasets will be added later.