**Tools for selective parity (aka selpar or sp) reconstruction**

./data: the folder contains test data (R2) for recon: haste (DW-HASTE) and sp (SP-DW-HASTE). where *b* represents the strength of diffusion-weighting gradients: b0 (diffusion-weighting gradients set to 0) and b1000 (diffusion-weighting gradients along the phase encoding direction); and *R* represents the acceleration factor along phase encoding (PE) direction.

Folder “matlabStuff” contains all dependencies for selpar recon:

1. FID-A-master: scripts for reading Siemens raw “\*.dat” files.

Available from https://github.com/CIC-methods/FID-A

1. SPIRiT\_v0.3: scripts for estimating the non-acquired k-space lines with SPIRIT-cg (SPIRIT conjugate gradient).

Available from https://people.eecs.berkeley.edu/~mlustig/Software.html

An alternative is to use SPIRIT-POCS implemented in-house.

1. selpar\_recon: scripts for selpar recon.
2. NIfTI\_read\_write: scripts for reading/writing “\*.nii” files, used to generate bias field.

To start with selpar recon:

1. set\_paths.m

Please, check that all paths: (a) to FID-A-master, (b) to SPIRIT-cg, and (c) to NIfTI\_read\_write are set correctly. If they are, running this script will ensure that all tools for selpar recon are available.

1. Please, make sure that all raw “\*.dat” files are housed in separate sub-folders. Otherwise, during “\*.dat” to “data.mat” file conversion, “data.mat” files will be overwriting each other!
2. read\_raw\_data\_selpar\_tasker.mat

Running this script from the top directory will convert all sub-folders’ “\*.dat” files to “data.mat” files.

1. master\_script\_selpar\_recon.m

The master script for performing selpar recon (the recon itself is implemented in selpar\_recon.m). The interface of the master script is explained in the script header.

1. master\_script\_selpar\_recon\_tasker.m

The tasker will run all of the jobs sequentially. It also shows how to set up a recon job.

1. save\_sosimg\_tasker.m

Running this script from the top directory will convert all sub-folders’ “dataRecon.dat” files to “sosimg.mat” files, effectively performing sum-of-squares (SoS) recon.

1. plot\_montage.m

Will create a montage of SoS images and save them if necessary.

1. generate\_bias\_field.m

Script (unlike all others above, this one is operated manually) to generate bias field maps. The bias field maps should be generated only once for the given session per matrix size using the highest quality fully reconstructed data (SoS) available.

1. The bias field correction for now needs to be performed manually. I would recommend creating a brain mask and performing normalization only over the mask, i.e. image(brain mask > 0)./bias(brain mask > 0) and let the background (noise) through unchanged.

**Known limitations**

The reconstruction algorithm #1 is limited to a max R = 2. The issue is solved in algorithm algorithm #2.

**Reconstruction algorithm #1**

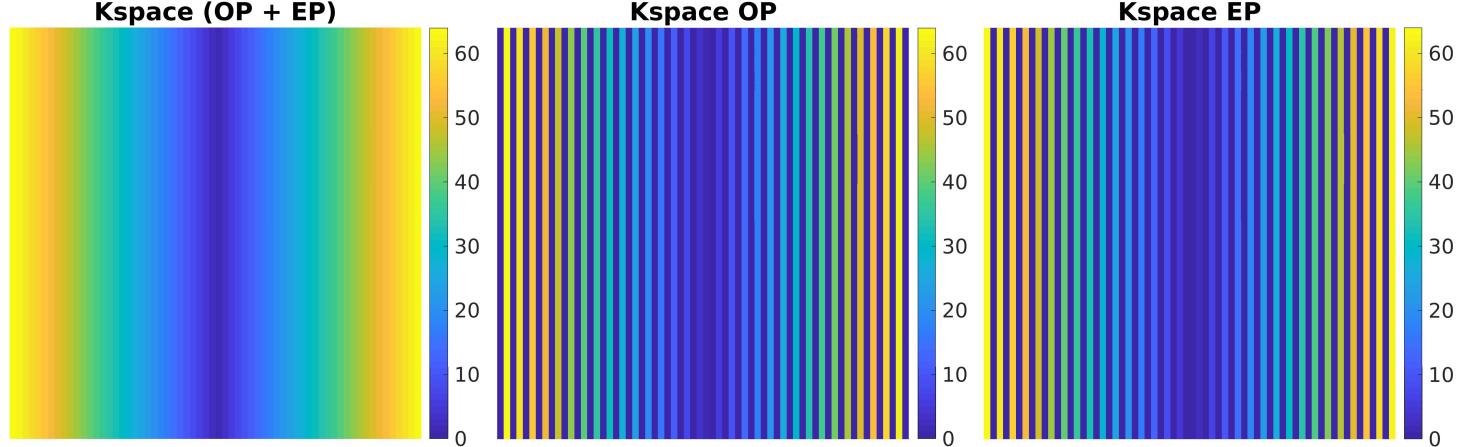
****

Figure 1. Modified centre-out phase encoding scheme (matrix size 64 x 64, acceleration factor R = 1) to sample the k-space while alternating echo parities. Shown is the full k-space matrice as well as k-space matrices on a per parity basis. Odd parity (OP), even parity (EP).

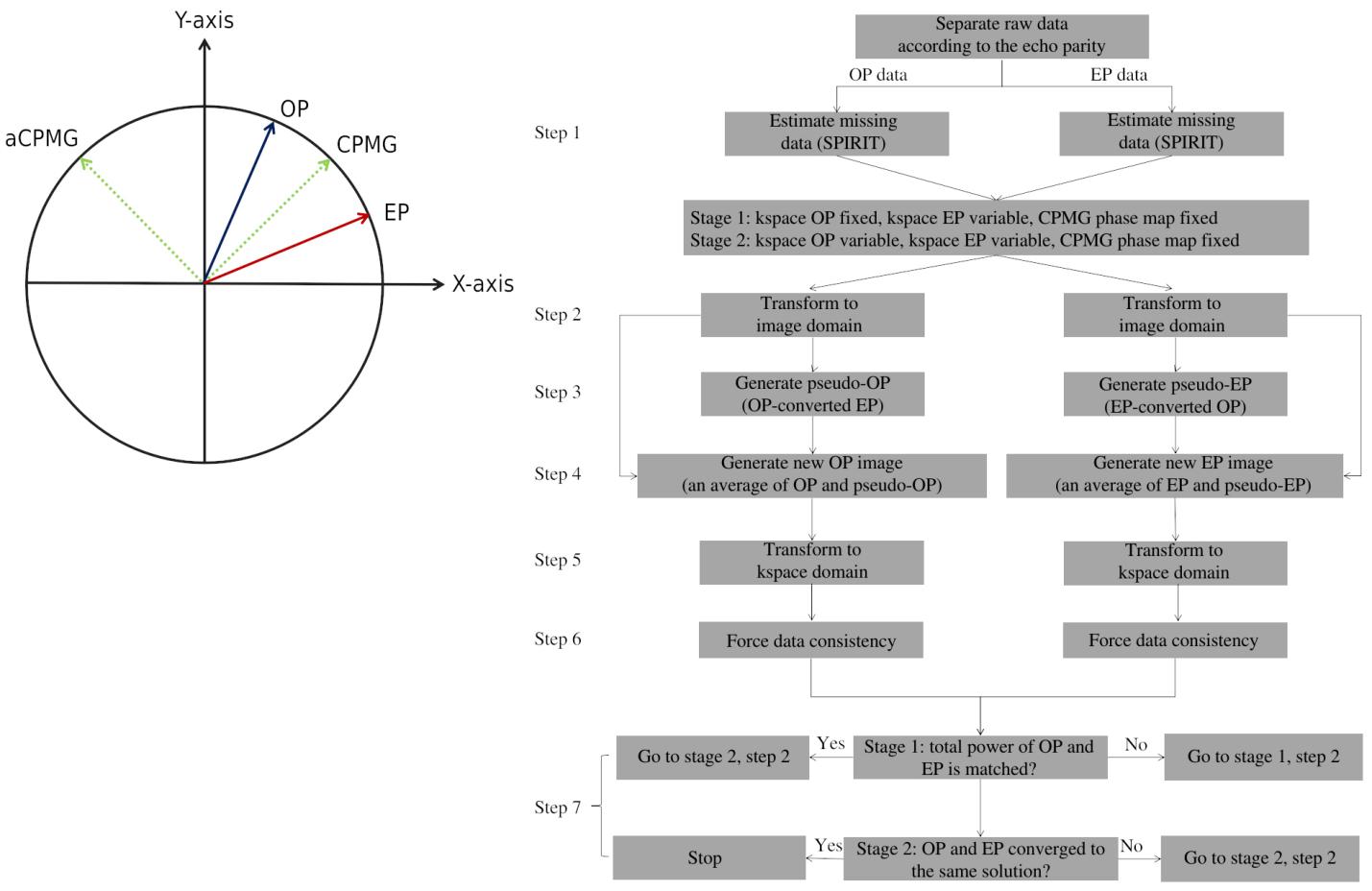
****

Figure 2.Selective parity reconstruction. (A) Orientation of odd parity (OP), even parity (EP), CPMG, and anti-CPMG vectors in an image domain voxel as viewed by a receiver coil. OP and EP data are the complex conjugates of each other in the frame of reference, defined by the CPMG angle in the image domain. (B) A schematic of reconstruction algorithm #1 based on the phase encoding scheme in figure 1. Pseudo-OP and pseudo-EP images can be generated from EP and OP images, respectively, by rotating the image by a negative (-) CPMG angle, taking the complex conjugate, and rotating it back by a positive (+) CPMG angle.

**Appendix**

We define the following operators: ***F*** is the forward FFT, ***iF*** is the inverse FFT, and ***‡*** is the transformation of OP to EP, and vice versa, effectively generating pseudo-EP and pseudo-OP, respectively.

It is assumed that acquired odd parity (OP) and even parity (EP) data are evenly distributed over the entire k-space matrix, each accounting for approximately 50% of the acquired data. Furthermore, OP and EP data are the complex conjugate of each in the frame of reference defined by the CPMG phase angle *φ*(*r*) in the image domain. The orientation of the CPMG vector for each image voxel was fixed from a separate DW-SP-HASTE (b0) acquisition. At the voxel level, rotating the data by (-) *φ*(*r*), taking the complex conjugate and rotating back by (+) *φ*(*r*) will transform OP to EP, and vice versa.

The implementation of the reconstruction algorithm is as follows.

**Stage 1 (iterative)**

1. Use SPIRIT to generate missing lines for *OP*(*k*) and *EP*(*k*) data in k-space.
2. Generate image domain data: *OP*(*r*) = ***iF***[*OP*(*k*)], and *EP*(*r*) = ***iF***[*EP*(*k*)].
3. Generate pseudo *OP*(*r*) and *EP*(*r*) data: *pOP*(*r*) = ***‡****EP*(*r*), and *pEP*(*r*) = ***‡****OP*(*r*).
4. Apply correction term to *OP*(*r*) and *EP*(*r*) data to force information sharing and generate new *OP*(*r*) and *EP*(*r*) data as follows:

*newOP*(*r*) = *OP*(*r*) + (*pOP*(*r*) - *OP*(*r*)) / 2, and *newEP*(*r*) = *EP*(*r*) + (*pEP*(*r*) - *EP*(*r*)) / 2.

1. Generate k-space data: *newOP*(*k*) = ***F***[*newOP*(*r*)], and *newEP*(*k*) = ***F***[*newEP*(*r*)].
2. Force data consistency as follows:

*newOP*(*k*) = *OP*(*k,* from step 1), and *newEP*(*k*) =  *EP*(*k,* acquired lines from step 1).

1. Check whether the total power of *OP*(*k*) and *EP*(*k*) from step 6 match, if so terminate, else return to step 2.

**Stage 2 (iterative,** uses*OP*(*k*) and *EP*(*k*) from **stage 1**: step 6 and step 1, respectively)

1. Generate image domain data: *OP*(*r*) = ***iF***[*OP*(*k*)], and *EP*(*r*) = ***iF***[*EP*(*k*)].
2. Generate pseudo *OP*(*r*) and *EP*(*r*) data: *pOP*(*r*) = ***‡****EP*(*r*), and *pEP*(*r*) = ***‡****OP*(*r*).
3. Apply correction term to *OP*(*r*) and *EP*(*r*) data to force information sharing and generate new *OP*(*r*) and *EP*(*r*) data as follows:

*newOP*(*r*) = *OP*(*r*) + (*pOP*(*r*) - *OP*(*r*)) / 2, and *newEP*(*r*) = *EP*(*r*) + (*pEP*(*r*) - *EP*(*r*)) / 2.

1. Generate k-space data: *newOP*(*k*) = ***F***[*newOP*(*r*)], and *newEP*(*k*) = ***F***[*newEP*(*r*)].
2. Force data consistency as follows:

*newOP*(*k*) = *OP*(*k,* acquired lines from **stage 1**: step 1), and *newEP*(*k*) =  *EP*(*k,* acquired lines from **stage 1**: step 1).

1. Check whether *OP*(*k*) and *EP*(*k*) from step 6 converged to the same solution, if so terminate, else return to **stage 2**: step 2.

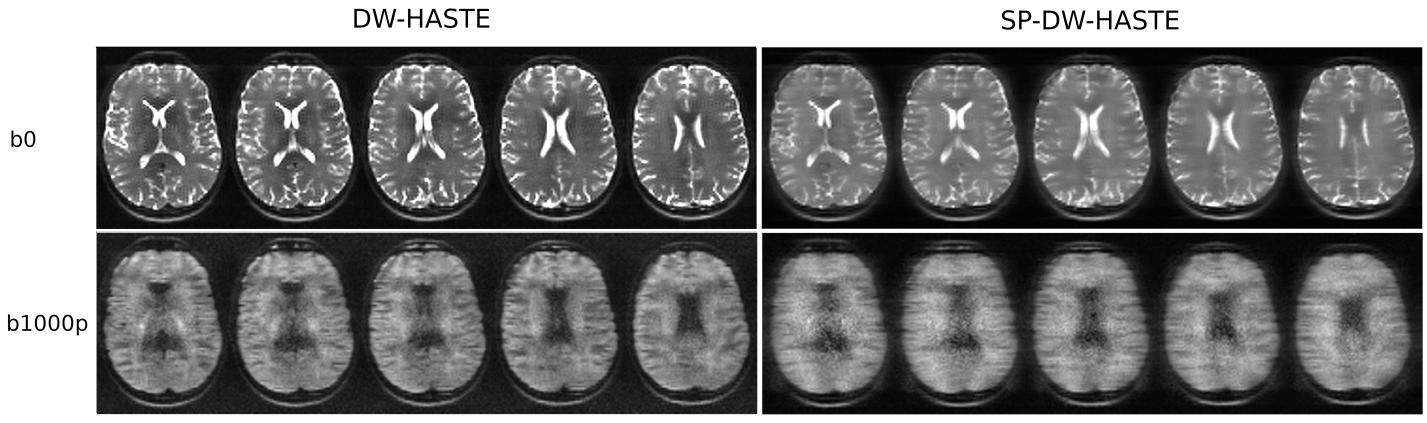


Figure 3. Reconstruction of provided sample data (single repetition bias field normalized SoS images, matrix size 96 x 70, R = 2). (left) DW-HASTE (b0 and b1000p) and (right) SP-DW-HASTE (b0 and b1000p).