Report of the second CCP PETMR hackathon

Logistics

Date: 17-19 Dec 2018

Location: St Thomas' Hospital, KCL, London

Attendees: David Atkinson (UCL), James Bland (KCL), Richard Brown (UCL), Sam Ellis (KCL), Abolfazl Mehranian (KCL), Camila Munoz (KCL), Evgueni Ovtchinnikov (STFC), Edoardo Pasca (STFC), Andrew Reader (KCL), Razvan Sencu (Manchester), Kris Thielemans (UCL), Alexander Whitehead (UCL).

Aims

The advertised aims of this hackathon were

- Train new developers in SIRF (Python)
- Port a synergistic algorithm (with existing implementation) to SIRF (Python)
- Resolve remaining issues with processing data from the mMR (MR and static PET)
  - Finish image geometry
  - Test on more data (mMR software VE11 SP3)
- SIRFReg
  - Finalisation and testing
  - Start with MCIR in SIRF

These were somewhat adjusted as indicated below after discussion.

Training

We used the SIRF-Exercises and SIRF/examples/Python as a basis for getting familiar with the code, VM, jupyter notebooks and spyder. These were used as a starting point for developing bits of code, either independently or in small teams. Examples included a minimal code-set to illustrate PET projections and a gradient descent MR reconstruction.

Code-setup and mechanisms

- Created SIRF-Contribs repo on https://github.com/CCPPETMR
- Attendees were given write access
- Project set-up
- Create issues inside SIRF-Contrib:
  - Implement MAPEM
  - Resampling adjoint
  - Image Geometry fill ISMRMRD
  - Abi's synergistic algorithm
  - Implement Gaussian Kernel Prior
- No intention to create "clean" code, but to get something working (Python only)

During the hackathon, we had regular groups sessions to report progress and raise questions.

Topics

Implement MAPEM

Group members: Sam Ellis, Camila Munoz, and James Bland

The aim of this group was to implement MAPEM algorithms to be used both independently and as a part of the overall synergistic PET-MR reconstruction algorithm. There were two main MAPEM algorithms to be explored: the one-step-late algorithm and the modified EM algorithm of De Pierro. In addition, these algorithms had to have the ability to include spatially variant weights, a feature not previously included in STIR/SIRF MAPEM implementations.

The SIRF example Python code for the isotropically weighted quadratically penalised MAPEM algorithm was used as a starting point, both in terms of data to be reconstructed and structure of code. The De Pierro algorithm was written in a series of functions embedded in a script providing example usage. This code was tested for 2 cases: 1) using uniform (i.e. isotropic) weights 2) using Bowsher weights calculated using the Prior class also introduced during the hackathon. Based on these results it seems that the De Pierro MAPEM implementation is working, and ready to be fully incorporated into the SIRF framework in future.

SIRFReg progress

Group members: David, Richard, Evgueni, Alex

It is now possible (with a small amount of work post-Hackathon) to convert STIRImageData objects into NiftiImageData3D objects. As long as an image can be converted into the NiftiImageData format, registration/resampling can be performed. This is therefore now possible with STIR, but requires more work in the MR side (see below).

Resampling of images

Image Geometry from ISMRMRD::Image

- Sorting of 2D images as 3D stack - Prior to the Hackathon, the sorting of acquisition data was done with respect to repetition, position[2] and then slice. This is only correct for axial acquisitions. During the Hackathon, the sorting was altered such that it is now performed with respect to contrast, repetition and then the projection of the position in the through-slice direction.
- Geom info – initial attempts were made at filling in the geometrical information of MR images from the ISMRMRD header. The four pieces of information required are the number of voxels in each direction, the spacing of the voxels, the image offset and the image direction (transformation matrix).
    - Number of voxels – number of voxels in x- and y-directions, number of images in z-direction (only applicable for a set of 2D images?)
    - Spacing - FoV/spacing in x- and y-directions. Distance between voxel centres in z-direction, allowing for under-/over-sampling (and thus why the first bullet point was necessary).
    - Offset and direction are still being checked. Offset has been initially set to 'position'. Direction has initially been set to 'read_dir', 'phase_dir' and 'slice_dir'.

Abi's Synergistic Algorithm

Group members: Abolfazl Mehranian, Camila Munoz

In parallel to the PET MAPEM recon group, we aimed to implement a regularized iterative SENSE MR image reconstruction with a weighted quadratic penalty (wQ-SENSE) optimized using either conjugate gradient or a classic gradient descent (GD). Initially, we tested the basic SIRF example, i.e. FullySampledReconstructor, for reconstruction of our fully-sampled 3D MPRAGE MR scan. The reconstructed image was correct without any noticeable artefacts. Next step was to implement a basic iterative SENSE-GD, which requires an estimation of coil sensitivity maps, however it turned out SIRF's MRAcquisitionModel currently supports only 2D datasets, which was reported to Evgueni for future developments. A 2D example dataset was then used to perform a wQ-SENSE-GD, using the Prior class. For implementation, we noticed the image datatype doesn't act similar to a numpy array for assignment or slicing which are handy for fast implementation and debugging. This was discussed with the core developers, and some tip such as 'fill' method was advised as a solution. The 2D MR data had 2 repeats. It was not straightforward for us to slice this Gadgetron datatype into separate repeats, a hacky solution was followed but the debugging was deferred to a future follow-up session.

Implement Gaussian Kernel Prior

Group members: Sam Ellis, Camila Munoz, Abolfazl Mehranian

A Prior Class was implemented in python in order to i) perform finite forward and backward differences between image voxels over a neighbourhood, which is key part of a penalty function used in regularized PET and MR image reconstruction and ii) calculate similarity kernels such as asymmetric Bowsher and symmetric Gaussians. The class was debugged and used for MAPEM PET and wQ-SENSE MR image reconstructions

Integration with CCPi CIL Optimisation (Python)

Group member: Edoardo Pasca

Started in hackathon 1, additional code modifications to the Python implementation of SIRF classes have been done in order to be able to use the CIL optimisation routines within SIRF.

CIL optimisation package naming conventions wrt methods and functions take from the underlying mathematical objects and it is different from what is normally used in SIRF. Therefore, particular emphasis has been devoted to documenting the new required methods and functions that were not present in SIRF.

Overall evaluation

- Overall very positive feedback.
- 2.5 day format was good.
- Useful to have the initial training at the start

Many thanks to the KCL team (especially Camilla, Abolfazl and Ugwu Joy) for hosting us, and for showing the PET Centre at St Thomas to some of us.