An example session for analyzing quantitative MRI data in the Multi-Parameter Mapping framework

Jörg Polzehl and Karsten Tabelow

September 13, 2023

This document illustrates the workflow of analyzing quantitative Magnetic Resonance Imaging (qMRI) data in the framework of Multi-Parameter Mapping (MPM) experiments (Weiskopf et al. 2013). The example uses artificial noisy MPM data based on the analysis results of a publicly available MPM dataset, see `https://hmri.info`: The original data was modeled using the ESTATICS model (Weiskopf et al. 2014). Its four adaptively smoothed parameter maps were used to generate the artificial data on a selected (small) sub cube with Rician noise by creating corresponding signal echos following the ESTATICS model equation. The artificial data is supplied with this package together with the smoothed quantitative maps (relaxation rates R_1 and $R_2^\star$, proton density PD, and magnetization transfer MT) from the original data for ground truth comparison.

Specifically, the data consist of, in total, 22 image volumes in NIfTI format that correspond to multiple echos of three different imaging modalities, i.e., 8 echos of T_1 weighted images, 8 echos of proton density (PD) weighted images and 6 images from a dual excitation FLASH magnetization transfer (MT) weighted sequence. Each echo in the image volumes is "acquired" at an echo time, repetition time and flip angle, that equal those of the original experiment. The quantitative maps suffer from a B_1 transmit (and receive) bias, that can be corrected for, if the smooth B_1 correction field that affects the local flip angle is known (Tabelow et al. 2019). An estimate for this field is supplied with the package, too.

For an more extended introduction we refer to Polzehl and Tabelow 2019 Chapter 6.

1 Reading the MPM data

First, we specify the directory where the data are stored within the package

```
> dataDir <- system.file("extdata", package = "qMRI")
```

The filenames of the data correspond to the weighting of the imaging. We create the paths of the data files including the B_1 field map a mask file (which in this cases describes the whole data cube, only):

```
> t1Names <- paste0("t1w_", 1:8, ".nii.gz")
> mtNames <- paste0("mtw_", 1:6, ".nii.gz")
> pdNames <- paste0("pdw_", 1:8, ".nii.gz")
> t1Files <- file.path(dataDir, t1Names)
> mtFiles <- file.path(dataDir, mtNames)
> pdFiles <- file.path(dataDir, pdNames)
> B1File <- file.path(dataDir, "B1map.nii.gz")
> maskFile <- file.path(dataDir, "mask.nii.gz")
```

The acquisition parameters (echo time (TE) in milliseconds, repetition time (TR) in milliseconds and flip angle (FA) in degree) for each volume are replicated from the original MPM data:

```
> TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
+         2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
+         2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
> TR <- rep(25, 22)
> FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
```

The package **pkg** provides a function `readMPMData` to read the data into an R session

```
> library(qMRI)
> mpm <- readMPMData(t1Files, pdFiles, mtFiles,
+                    maskFile,
+                    TR = TR, TE = TE, FA = FA,
+                    verbose = FALSE)
```

which creates an object `mpm` of class `"MPMdata"`.

2 Parameter estimation (ESTATICS)

The ESTATICS model (Weiskopf et al. 2014) describes the data from an MPM sequence by the relaxation rate $R_2^\star$ for the signal decay with the echo time TE and the three absolute values for $TE = 0$. Parameters in this model can be estimated by

```
> setCores(2)

  Total CPU cores available: 20  CPU cores in use: 2.

> modelMPM <- estimateESTATICS(mpm,
+                              method = "NLR",
+                              verbose = FALSE)

  Total CPU cores available: 20  CPU cores in use: 2.
```

using nonlinear least-squares regression. However, the low low signal-to-noise ratio (SNR) in the data leads to a bias for the parameter estimates, see Polzehl and Tabelow 2016 and Tabelow et al. 2017. Therefore the parameter estimation can alternatively be performed by

```
> sigma <- array(50, mpm$sdim)
> modelMPMQL <- estimateESTATICS(mpm,
+                                method = "QL",
+                                sigma = sigma,
+                                L = 1,
+                                verbose = FALSE)

 Total CPU cores available: 20  CPU cores in use: 2.
```

using a quasi-likelihood formulation (Polzehl and Tabelow 2016; Tabelow et al. 2017). It avoids, in case of low SNR, the bias caused by the skewness of the Rician signal distribution. The application of this method requires to specify the scale parameter σ of the Rician distribution, which in this case of artificial data is explicitly known. Alternatively, a map of scale parameters could be estimated using function `awslsigmc` from package **dti**

```
> ddata <- extract(mpm,"ddata")
> mask <- extract(mpm,"mask")
> if(require(dti)) sigma <- awslsigmc(ddata[1,,,],16,mask)$sigma
```

see Tabelow, Voss, and Polzehl 2015 for details of the method.

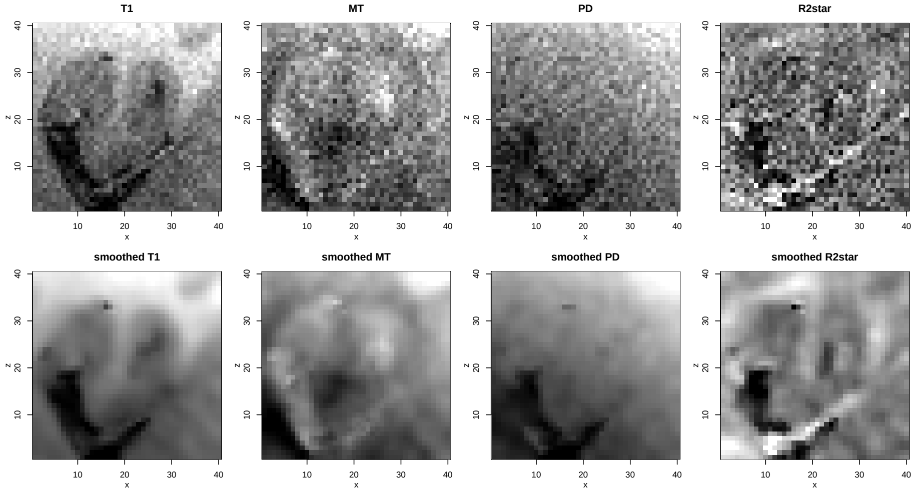
3 Structural adaptive smoothing

Data from MPM sequences suffer from noise hindering the quality of the estimated quantitative maps. Mohammadi et al. 2017 introduced a structural adaptive smoothing method that can be used to reduce the variability of the estimated parameter maps and, if `mpmData` is specified, the observed image data:

```
> # use setCores(ncores) to enable openMP parallelization
> setCores(2,reprt=FALSE)
> modelMPMQLsp1 <- smoothESTATICS(modelMPMQL,
+                                 mpmData = extract(mpm, "ddata"),
+                                 kstar = 16,
+                                 alpha = 0.004,
+                                 patchsize = 1,
+                                 verbose = FALSE)
```

The package **qMRI** implements an extension of the method that uses the comparison of local patches for adaptive smoothing (Polzehl, Papafitsoros, and Tabelow 2018). The resulting ESTATICS unsmoothed and smoothed parameter maps for the central coronal slice can be illustrated by

```
> library(adimpro)
> rimage.options(zquantiles = c(.01, .99), ylab = "z")
> par(mfrow = c(2, 4),
+     mar = c(3, 3, 3, 1), mgp = c(2, 1, 0))
> pnames <- c("T1", "MT", "PD", "R2star")
> for (i in 1:4) {
+   modelCoeff <- extract(modelMPMQL,"modelCoeff")
+   rimage(modelCoeff[i, , 11, ])
+   title(pnames[i])
+ }
> for (i in 1:4) {
+   modelCoeff <- extract(modelMPMQLsp1,"modelCoeff")
+   rimage(modelCoeff[i, , 11, ])
+   title(paste("smoothed", pnames[i]))
+ }
```



The preceding code also applies the adaptive smoothing not only to the parameter maps of the ESTATICS model, but also to the data itself. From this new parameter maps can be estimated by the quasi-likelihood formulation and thus avoiding the bias due to the skewness in the signal distribution:

```
> modelMPM2 <- estimateESTATICS(mpmsp1,
+                                method = "NLR",
+                                L = 1,
+                                verbose = FALSE)

  Total CPU cores available: 20  CPU cores in use: 2.
```

Note, that we again employ the same Rician scale parameter, i.e., of the unprocessed data, cf. the discussion in Polzehl and Tabelow 2016.

4 Calculating quantitative maps

Finally, from the (un)smoothed parameter maps of the ESTATICS model, we can compute the quantitative R_1, $R_2^\star$, PD and MT maps

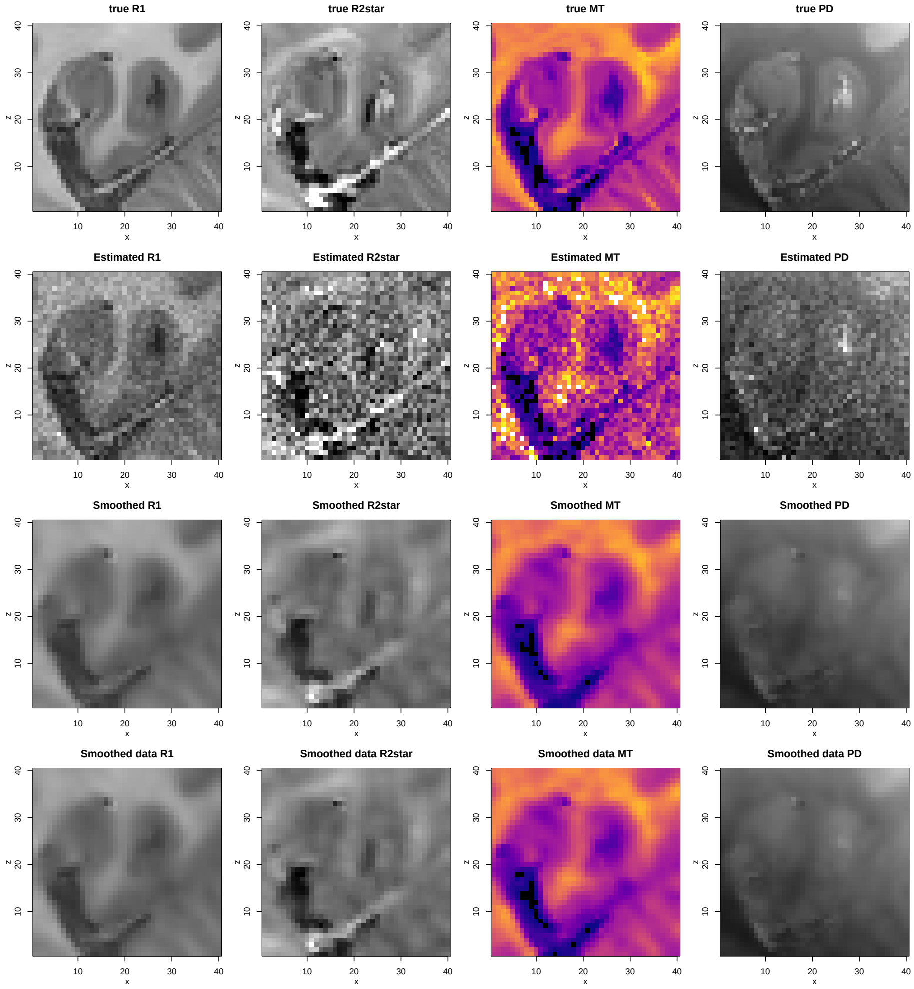
```
> qMRIMaps <- calculateQI(modelMPM,
+                         b1File = B1File,
+                         TR2 = 3.4)

reading B1 correction file
calculating R1 ... Spoiling correction ... v3* 0.8mm R4 protocol
Spoiling correction ... v3* 0.8mm R4 protocol
```

```
done
calculating PD ... done
calculating MT ... done

> qMRIQLMaps <- calculateQI(modelMPMQL,
+                           b1File = B1File,
+                           TR2 = 3.4)

reading B1 correction file
calculating R1 ... Spoiling correction ... v3* 0.8mm R4 protocol
Spoiling correction ... v3* 0.8mm R4 protocol
done
calculating PD ... done
calculating MT ... done

> qMRIQLSmoothedp1Maps <- calculateQI(modelMPMQLsp1,
+                                     b1File = B1File,
+                                     TR2 = 3.4)

reading B1 correction file
calculating R1 ... Spoiling correction ... v3* 0.8mm R4 protocol
Spoiling correction ... v3* 0.8mm R4 protocol
done
calculating PD ... done
calculating MT ... done

> qMRIMaps2 <- calculateQI(modelMPM2,
+                            b1File = B1File,
+                            TR2 = 3.4)

reading B1 correction file
calculating R1 ... Spoiling correction ... v3* 0.8mm R4 protocol
Spoiling correction ... v3* 0.8mm R4 protocol
done
calculating PD ... done
calculating MT ... done
```

The calculation requires the B_1-bias field, if available, and the TR_2 parameter of the MTw sequence (Tabelow et al. 2019).

We show the central coronal slice of the estimated quantitative maps together with the maps for the ground truth used to generate the data

```
> library(oro.nifti)
> zlim <- matrix(c(0, 0, 0, 3000,
+                   1.5, 35, 2, 10000),
+                 4, 2)
> R1 <- readNIfTI(file.path(dataDir, "R1map.nii.gz"))
> R2star <- readNIfTI(file.path(dataDir, "R2starmap.nii.gz"))
> MT <- readNIfTI(file.path(dataDir, "MTmap.nii.gz"))
> PD <- readNIfTI(file.path(dataDir, "PDmap.nii.gz"))
> rimage.options(ylab = "z")
> par(mfrow = c(4, 4),
+     mar = c(3, 3, 3, 1), mgp = c(2, 1, 0))
> nmaps <- c("R1", "R2star", "MT", "PD")
> rimage(R1[, 11, ], zlim = zlim[1, ],
+        main = paste("true", nmaps[1]))
> rimage(R2star[, 11, ], zlim = zlim[2, ],
+        main = paste("true", nmaps[2]))
> rimage(MT[, 11, ], zlim = zlim[3, ],
+        main = paste("true", nmaps[3]),
+        col = colMT)
> rimage(PD[, 11, ], zlim = zlim[4, ],
+        main = paste("true", nmaps[4]))
> qmap1 <- extract(qMRIQLMaps, nmaps)
> for (i in 1:4) rimage(qmap1[[i]][, 11, ], zlim = zlim[i, ],
+                       main = paste("Estimated", nmaps[i]),
+                       col = if(i==3) colMT else grey(0:225/255))
> qmap2 <- extract(qMRIQLSmoothedp1Maps, nmaps)
> for (i in 1:4) rimage(qmap2[[i]][, 11, ], zlim = zlim[i, ],
+                       main = paste("Smoothed", nmaps[i]),
+                       col = if(i==3) colMT else grey(0:225/255))
> qmap3 <- extract(qMRIMaps2, nmaps)
> for (i in 1:4) rimage(qmap3[[i]][, 11, ], zlim = zlim[i, ],
+                       main = paste("Smoothed data", nmaps[i]),
+                       col = if(i==3) colMT else grey(0:225/255))
```



true R1

true R2star

true MT

true PD

Estimated R1

Estimated R2star

Estimated MT

Estimated PD

Smoothed R1

Smoothed R2star

Smoothed MT

Smoothed PD

Smoothed data R1

Smoothed data R2star

Smoothed data MT

Smoothed data PD

The color scheme `colMT` is designed to emphasize the grey matter (magenta) / white matter (yellow) contrast in the MT image with a `zlim = c(0,2)`.

Using the quasi-likelihood estimation method is, for low SNR, supposed to reduce the bias caused by the skewness of the Rician distribution, cf. Polzehl and Tabelow 2016 and Tabelow et al. 2017.

```
> qmap0 <- extract(qMRIMaps,nmaps)
> mask <- extract(mpm,"mask")
> cat("\n",
+     "Bias of NLR estimates\n",
+       "R1", mean((qmap0$R1-R1)[mask]),
+       "R2star", mean((qmap0$R2star-R2star)[mask]),
+       "MT", mean((qmap0$MT-MT)[mask]),
+       "PD", mean((qmap0$PD-PD)[mask]), "\n",
+     "Bias of  QL estimates\n",
+       "R1", mean((qmap1$R1-R1)[mask]),
+       "R2star", mean((qmap1$R2star-R2star)[mask]),
+       "MT", mean((qmap1$MT-MT)[mask]),
+       "PD", mean((qmap1$PD-PD)[mask]), "\n")
```

```
 Bias of NLR estimates
 R1 0.00853 R2star -0.0477 MT -0.00195 PD 64.3
 Bias of  QL estimates
 R1 0.00325 R2star 0.342 MT 0.021 PD 68.5
```

Let's see which estimate performs best with respect to the root mean squared error (RMSE):

```
> cat("\n",
+     "Root mean squared error of NLR estimate\n",
+       "R1", sqrt(mean((qmap0$R1-R1)[mask]^2)),
+       "R2star", sqrt(mean((qmap0$R2star-R2star)[mask]^2)),
+       "MT", sqrt(mean((qmap0$MT-MT)[mask]^2)),
+       "PD", sqrt(mean((qmap0$PD-PD)[mask]^2)), "\n",
+     "Root mean squared error of  QL estimate\n",
+       "R1", sqrt(mean((qmap1$R1-R1)[mask]^2)),
+       "R2star", sqrt(mean((qmap1$R2star-R2star)[mask]^2)),
+       "MT", sqrt(mean((qmap1$MT-MT)[mask]^2)),
+       "PD", sqrt(mean((qmap1$PD-PD)[mask]^2)),"\n",
+     "Root mean squared error of smoothed QL estimate\n",
+       "R1", sqrt(mean((qmap2$R1-R1)[mask]^2)),
+       "R2star", sqrt(mean((qmap2$R2star-R2star)[mask]^2)),
+       "MT", sqrt(mean((qmap2$MT-MT)[mask]^2)),
+       "PD", sqrt(mean((qmap2$PD-PD)[mask]^2)),"\n",
+     "Root mean squared error of estimate from smoothed data \n",
+       "R1", sqrt(mean((qmap3$R1-R1)[mask]^2)),
+       "R2star", sqrt(mean((qmap3$R2star-R2star)[mask]^2)),
+       "MT", sqrt(mean((qmap3$MT-MT)[mask]^2)),
+       "PD", sqrt(mean((qmap3$PD-PD)[mask]^2)),"\n")
```

```
 Root mean squared error of NLR estimate
 R1 0.0919 R2star 6.08 MT 0.26 PD 540
 Root mean squared error of  QL estimate
 R1 0.0932 R2star 6.24 MT 0.273 PD 574
 Root mean squared error of smoothed QL estimate
 R1 0.0747 R2star 3.37 MT 0.144 PD 263
 Root mean squared error of estimate from smoothed data
 R1 0.075 R2star 3.43 MT 0.146 PD 267
```

For interpretation we need to compare this to the mean parameter values

```
> cat("Mean R1", mean(R1[mask]), "Mean R2star",
+     mean(R2star[mask]), "Mean MT", mean(MT[mask]),
+     "Mean PD", mean(PD[mask]),"\n")
```

```
Mean R1 0.769 Mean R2star 18.4 Mean MT 0.895 Mean PD 5706
```

We see here, that using the quasi-likelihood estimation is supposed to reduce the bias of the estimated ESTATICS parameters, but does so at the cost of a slightly increased variance. Structural adaptive smoothing leads to a considerable reduction in RMSE. Modelling of the spatially smoothed data has the additional effect of reduced bias originating from data variability.

References

Mohammadi, S. et al. (2017). *Simultaneous adaptive smoothing of relaxometry and quantitative magnetization transfer mapping.* Preprint 2432. WIAS. DOI: 10.20347/WIAS.PREPRINT.2432.

Polzehl, J., K. Papafitsoros, and K. Tabelow (2018). *Patch-wise adaptive weights smoothing.* Preprint 2520. WIAS. DOI: 10.20347/WIAS.PREPRINT.2520.

Polzehl, J. and K. Tabelow (2016). “Low SNR in diffusion MRI models”. In: *J. Am. Stat. Assoc.* 111.516, pp. 1480–1490. DOI: 10.1080/01621459.2016.1222284.

Polzehl, Jörg and Karsten Tabelow (2019). *Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R.* Use R! Springer. DOI: 10.1007/978-3-030-29184-6.

Tabelow, K., H. U. Voss, and J. Polzehl (2015). “Local estimation of the noise level in MRI using structural adaptation”. In: *Med. Image Anal.* 20, pp. 76–86.

Tabelow, K. et al. (2017). “Removing the estimation bias due to the noise floor in multi-parameter maps”. In: *Proceedings of ISMRM 2017*.

Tabelow, K. et al. (2019). “hMRI – A toolbox for quantitative MRI in neuroscience and clinical research”. In: *NeuroImage* 194, pp. 191 –210. DOI: 10.1016/j.neuroimage.2019.01.029.

Weiskopf, N. et al. (2013). “Quantitative multi-parameter mapping of R1, PD(*), MT, and R2(*) at 3T: a multi-center validation.” eng. In: *Front. Neurosci.* 7, p. 95. DOI: 10.3389/fnins.2013.00095.

Weiskopf, N. et al. (2014). “Estimating the apparent transverse relaxation time (R2*) from images with different contrasts (ESTATICS) reduces motion artifacts”. In: *Front. Neurosci.* 8.278, pp. 1–10.