Chapter 41

Analysis Protocol for Renal Sodium (^{23}Na) MR Imaging

James T. Grist, Esben Søvsø Szocska Hansen, Frank G. Zöllner,
and Christoffer Laustsen

Abstract

The signal acquired in sodium (^{23}Na) MR imaging is proportional to the concentration of sodium in a voxel, and it is possible to convert between the two using external calibration phantoms. Postprocessing, and subsequent analysis, of sodium renal images is a simple task that can be performed with readily available software. Here we describe the process of conversion between sodium signal and concentration, estimation of the corticomedullary sodium gradient and the procedure used for quadrupolar relaxation analysis.

This chapter is based upon work from the COST Action PARENCHIMA, a community-driven network funded by the European Cooperation in Science and Technology (COST) program of the European Union, which aims to improve the reproducibility and standardization of renal MRI biomarkers. This analysis protocol chapter is complemented by two separate chapters describing the basic concept and experimental procedure.

Key words Magnetic resonance imaging (MRI), Kidney, Rats, ^{23}Na , Sodium

1 Introduction

The signal acquired in sodium (^{23}Na) MR imaging is proportional to the concentration of sodium in a voxel, and it is possible to convert between the two using external calibration phantoms. To calculate sodium concentration maps, an imaging volume including the tissue of interest and external calibration phantoms is required, using a gradient echo (GRE) sequence for data acquisition. T_2^* relaxation maps can be calculated from multiecho GRE data acquired in the same acquisition. Theoretical considerations and acquisition protocols are detailed in the chapters by Grist JT et al. “Sodium (^{23}Na) MRI of the Kidney: Basic Concept” and “Sodium (^{23}Na) MRI of the Kidney: Experimental Protocol.” Creation of sodium concentration maps can be performed with several commercially available software packages (*see Note 1*). In general, a linear fit is performed between the mean signal intensity of each

calibration phantoms and a region of noise. The fitting coefficients are used to convert sodium signal to sodium concentration. This chapter focuses on how to do this, conceptually.

This analysis protocol chapter is complemented by two separate chapters describing the basic concept and experimental procedure, which are part of this book.

2 Materials

2.1 Software Requirements

1. To calculate parameter maps: A programming environment capable of applying fitting models, such as Python (www.python.org), Octave (www.gnu.org/software/octave) or MATLAB (The MathWorks, Natick, MA, USA) with a curve fitting capability. The method described in this chapter provides a detailed description for a solution in MATLAB, but can be adopted to other platforms.
2. Optional: A image processing software such as Fiji (www.fiji.sc) or Horos (www.horosproject.org).

2.2 Source Data: Format Requirements and Quality Check

To be able to calculate sodium concentration maps, images including sodium calibration phantoms acquired during an experiment are required, as well as the known absolute concentration of sodium in the calibration phantoms.

2.2.1 Input Requirements

2.2.2 Data Exclusion

When sodium imaging is acquired, it commonly has a low signal-to-noise ratio (SNR). In order to construct accurate sodium T_2^* maps, it is important to ensure that the signal of kidneys is greater than that of the background noise, else a poor fit will occur. Therefore, data with SNR lower than a threshold (e.g., $\text{SNR} < 5$) should be discarded.

This step should be avoided for concentration mapping, as a region of noise is required to calculate an assumed 0 mmol/L signal value.

2.2.3 SNR Check for T_2^* Mapping

To check SNR in renal sodium imaging in MATLAB (NB: example code for SNR measurements assumes a single slice and absolute value data):

1. Draw an ROI, using the “roipoly” command, over a region of background noise free of artefacts.

```
Region = poly2mat(roipoly(mat2gray(image_slice)));
Masked_Region = Region .* image_slice;
```

2. Calculate the standard deviation of the noise using the “std” command.

```
NoiseSTD = std(Masked_Region(Masked_Region > 0));
```

3. Divide the original image by 0.66 multiplied by the standard deviation of the noise to produce SNR maps.

```
Image_SNR = image_slice ./ (sqrt(2)*NoiseSTD);
```

4. SNR maps can then be used to form a mask to remove signal less than a predefined SNR value (e.g., 5).

```
Image_SNR(Image_SNR < 5) = 0;
Image_SNR(Image_SNR > 1) = 1;
Masked_Image_Slice = Image_slice .* Image_SNR
```

2.2.4 Dual Flip Angle B_1 Mapping for Sodium Concentration Mapping

Furthermore, if local send and receive RF coils are used, B_1 mapping to correct for signal inhomogeneity due to coil profiles can be performed (*see Note 2*).

3 Methods

3.1 Sodium Concentration Mapping

The mean signal from phantoms in the images can be derived using regions of interest (ROIs). Furthermore, a mean image noise value should be defined on a slice by slice basis using signal outside of the body.

Once the mean signal of each phantom, as well as noise, has been calculated, a linear fit should be performed between the known concentration values and the phantom/noise signal (assuming noise represents 0 mmol/L sodium).

The coefficients derived from the fit (offset and slope) can then be used to convert each voxel in the image from signal intensity to concentration using Eq. 1, below.

$$\text{Sodium concentration (mmolL}^{-1}\text{)} = \frac{\text{signal} - \text{offset}}{\text{slope}} \quad (1)$$

3.1.1 Algorithm for Sodium Concentration Mapping

1. Define region of noise on a slice-by-slice basis using either automated (selecting voxels in a specific region of each image) or via ROI placement on a slice-by-slice basis.
2. Calculate the mean of each noise region and store this value in a vector (e.g., *meannoise*).
3. Segment sodium calibration phantoms using ROIs and calculate the mean signal for each phantom.
4. Perform concentration mapping by calculating a linear fit between the noise, phantom signals, and the known

Fig. 1 Example images of sodium signal in the healthy porcine kidney, acquired at 3T, demonstrating increase sodium signal in the medulla, in comparison to the cortex

concentrations of the phantoms (example code presented below) on a slice by slice basis

3.1.2 Example Matlab Code

```
for zslice = 1:Number_Of_Z_Slices
    fitting = polyfit([0,conc1,conc2],[meannoise(zslice),
    Phantom1, Phantom2],1);
    slope = fitting(1);
    offset = fitting(2);
    Image(:,:,zslice) = (Image(:,:,zslice) - offset)./slope;
End
```

Example sodium concentration mapping data of the porcine kidney can be seen in Fig. 1.

3.1.3 Biexponential T_2^* Mapping

In order to map the biexponential T_2^* of the kidney (separating the restricted quadrupolar spins and the freely moving spins), a more complicated process of signal fitting is required. Utilizing multi-echo GRE data, acquired in the same imaging acquisition, a non-linear fitting routine is employed to determine the behavior of the sodium signal, as described in Eq. 2.

$$s(\text{TE}) = \left(a e^{-\frac{\text{TE}}{T_{2,\text{Short}}^*}} + b e^{-\frac{\text{TE}}{T_{2,\text{Long}}^*}} \right) \quad (2)$$

where a and b are the relative pool sizes, and $T_{2,\text{Short}}^*$ and $T_{2,\text{Long}}^*$ are the pool T_2^* constants (ms), with the constraints that $a + b = 1$, and $T_{2,\text{Short}}^* > T_{2,\text{Long}}^*$. Assuming five or more data sets are acquired, the following MATLAB code can be used to fit the above curve. An example fit is shown in Fig. 2.

Fig. 2 Example fit of biexponential data acquired from an agar phantom

3.1.4 Algorithm for $T2^*$ Mapping

1. Perform SNR thresholding as detailed in Subheading 2.2.3.
2. Loop biexponential curve-fitting code (example Matlab code presented below) over all SNR masked regions in the image. The fit algorithm used here is trust region reflective.
3. Store data either as DICOM, proprietary (e.g., .mat), or text files.

3.1.5 Example Matlab Fitting Code

```
%% List of echo times acquired during imaging
TE_List = [TE1, TE2, TE3,...];
%% Setting up fit options
fopts = fitoptions('Method','NonlinearLeastSquares',...
    'Lower',[0,0,0,5],...
    'Upper',[1,1,5,55]
    'StartPoint',[0.2 1,10]);
% Fit curve equation
ft = fittype('a*exp(-x/b) + (1-a)*exp(-x/c)','coefficients',
{'a', 'T21','T22'},'independent','x','options',fopts);
%% Pre-allocation of memory to speed up fitting
ShortPoolSize(xvoxel,yvoxel,zvoxel) = ...
    zeros(size(Imaging_Data,1),size(Imaging_Data,2),size(Imaging_Data,3));
LongPoolSize(xvoxel,yvoxel,zvoxel) = ...
    zeros(size(Imaging_Data,1),size(Imaging_Data,2),size(Imaging_Data,3));
ShortPoolRelaxation(xvoxel,yvoxel,zvoxel) = ...
    zeros(size(Imaging_Data,1),size(Imaging_Data,2),size(Imaging_Data,3));
LongPoolRelaxation(xvoxel,yvoxel,zvoxel) = ...
    zeros(size(Imaging_Data,1),size(Imaging_Data,2),size(Imaging_Data,3));
```

```

%% Fitting loop core
for zslice = 1:Number_Of_Z_Slices
    for yvoxel = 1:Number_Of_Y_Voxels
        for xvoxel = 1:Number_Of_X_Voxels
            if (Masked_Image_Slice ~= 0)
                % Extract data vector for one pixel
                Data_Vector = Imaging_Data (xvoxel,yvoxel,zlice,:);
                % Do curve fitting for one voxel
                [Fit,gof] = fit(TE_List,Data_Vector, ft);
                % Store results
                ShortPoolSize(xvoxel,yvoxel,zvoxel) = Fit.a;
                LongPoolSize(xvoxel,yvoxel,zvoxel) = 1 - Fit.a;
                ShortPoolRelaxation(xvoxel,yvoxel,zvoxel) = Fit.b;
                LongPoolRelaxation(xvoxel,yvoxel,zvoxel) = Fit.c
            end
        end
    end
end
end

```

If data are acquired at higher magnetic field strengths ($B_0 \geq 3T$), it is advisable to alter the pool constraints for the parameter f_{opts} to reflect the shorter $T_{2,Short}^*$ and $T_{2,Long}^*$ components.

3.2 Visual Display

There is no current consensus on the best colour map to use when displaying sodium concentration images. T_1 - or T_2 -weighted images acquired in the same slice position can provide guidance for the interpretation as well as base layer for presenting the images as a set of fused images, *see* Fig. 1.

An example display option for MATLAB is described below:

1. Display the parameter map, which is a matrix with floating point numbers, as an image (in MATLAB: `imagesc (ShortPoolRelaxation)`).
2. Remove axis labels and ensure the axis are scaled equally, so that pixels are square and not rectangular (in MATLAB: `axis off; axis equal;`).
3. Select the color map and display a color bar (in MATLAB: `colormap(hot(256)); colorbar;`).
4. Set the display range for the color coding, for example, for short and long T_2^* `[0 5]` and `[5 60]`, respectively; (in MATLAB: `caxis([0 5]);`).

3.3 Quantification

In order to obtain quantitative medulla and cortex values from sodium concentration maps, the kidney can be segmented using either concentric objects or semiautomated ROI placement [1, 2]. To assess the corticomedullary sodium gradient, a linear fit can be performed across concentration values derived from the

Fig. 3 Example corticomedullary sodium gradient, healthy (black) and impaired rodent kidney (orange)

Table 1
Example sodium concentration values from renal compartments, derived from post-sacrifice kidneys [5]

Tissue type	Total sodium concentration (mmol/L)
Cortex	29 ± 7
Outer-medullar	75 ± 20
Inner-medullar	161 ± 29

concentric objects method, providing the slope of the gradient. An example corticomedullary sodium gradient in both healthy and fibrosis impaired rodent kidney can be seen in Fig. 3.

3.4 Results
Validation

3.4.1 Comparison
with Tissue Values from
Biopsy

Further validation of imaging derived results can be performed using biopsy derived tissue samples, using methods such as flame photometry, mass spectrometry, or *ex vivo* spectroscopy.

3.4.2 Comparison
with Reference Values from
the Literature

If biopsy derived results cannot be obtained, for example due to longitudinal studies, obtained values for both tissue segmentation can be compared against reference values, shown in Table 1.

4 Notes

1. Processing is typically performed in software packages such as MATLAB (The MathWorks, MA) or open source platforms as python programming language (python.org). The data can either

be processed as DICOM images or if available as the raw data format from the scanner (dicomstandard.org/using/cds/). Finalizing the data in the DICOM format pipelines the data for further analysis and comparison to conventional MRI images.

2. The linear proportionality of the NMR signal to the spin density allows for the absolute quantification of the Total Sodium Concentration (TSC) on the basis of a known concentration reference. However, TSC measurements are distorted by hardware-dependent influences and by typical ^{23}Na -NMR properties. This can be determined and corrected with the help of B_1 mapping methods [3]. Mapping methods which allow short TE are the double-angle and the phase-sensitive method [4].

Acknowledgments

This chapter is based upon work from COST Action PARENCHIMA, supported by European Cooperation in Science and Technology (COST). COST (www.cost.eu) is a funding agency for research and innovation networks. COST Actions help connect research initiatives across Europe and enable scientists to enrich their ideas by sharing them with their peers. This boosts their research, career, and innovation.

PARENCHIMA (renalMRI.org) is a community-driven Action in the COST program of the European Union, which unites more than 200 experts in renal MRI from 30 countries with the aim to improve the reproducibility and standardization of renal MRI biomarkers.

References

1. Milani B, Ansaloni A, Sousa-Guimaraes S et al (2017) Reduction of cortical oxygenation in chronic kidney disease: evidence obtained with a new analysis method of blood oxygenation level-dependent magnetic resonance imaging. *Nephrol Dial Transplant* 32(12):2097–2105
2. Mikheev A, Lim RP (2016) A semi-automated “blanket” method for renal segmentation from non-contrast T1-weighted MR images. *MAGMA* 29:197–206
3. Gatzeva-topalova PZ, Warner LR, Pardi A, Carlos M (2011) Quantitative sodium imaging with a flexible twisted projection pulse sequence. *Magn Reson Med* 18:1492–1501
4. Morrell GR, Schabel MC (2010) An analysis of the accuracy of magnetic resonance flip angle measurement methods. *Phys Med Biol* 55:6157–6174
5. Saikia TC (1964) Composition of the renal cortex and medulla of rats during water diuresis and antidiuresis. *J Clin Invest* 51:1145–1151

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.