UNIVERSITY OF GOTHENBURG

Department of Medical Radiation Sciences

SAHLGRENSKA ACADEMY

Deep Neural Networks for Noise Reduction and Bias Removal in MR Diffusion Signal

Beqir Berisha

Essay/Thesis:	30 hp
Program:	Medical physics
Level:	Second cycle
Term/year:	Fall/2021
Supervisors:	Stefan Kuczera, Stephan Maier, Rolf A. Heckemann
Examiner:	Magnus Båth

Abstract

Essay/Thesis:	30 hp
Program:	Medical physics
Level:	Second cycle
Term/year:	Fall/2021
Supervisors:	Stefan Kuczera, Stephan Maier, Rolf A. Heckemann
Examiner:	Magnus Båth
Keywords:	Deep Neural network, Prostate cancer, noise, Diffusion-weighted imaging, Rician bias correction

The purpose of this study is to assess the potential of deep neural networks, trained by unsupervised learning, for diffusion weighted imaging (DWI) data modeling and denoising. DWI data were modeled by a biexponential model and Rician bias was corrected for. Deep neural networks that estimate the magnetic resonance (MR) diffusion-weighted signal decay were trained on simulated signal data. Results for simulated data with known σ_g and estimated σ_g were compared, where known σ_g was the most suitable method. Furthermore, a deep neural network trained directly on patient prostate data was used to denoise images. The method of using deep neural networks was compared with OBSIDIAN, which is a model-based, iterative fitting procedure. The deep neural network showed an improvement of image quality with respect to the raw data, but did not have the same quality as OBSIDIAN. Using the trained deep neural network on the same patient data resulted in a runtime of 1.9 ms. The results showed that there is some potential in using deep neural networks for DWI data modelling and denoising, but further optimization is needed.

Contents

List of Figures

List of Tables

1 INTRODUCTION

Diffusion-weighted imaging (DWI) is a magnetic resonance imaging (MRI) technique that uses strong gradient pulses with a long duration to attain diffusion sensitivity to the signal measured in tissues (McRobbie, Moore, Graves, & Prince, 2017, p. 39). In particular, the diffusion-weighted (DW) image is sensitive to the self-diffusion of water molecules in tissues.

Cancer tissue has structural properties that differ from those of normal tissue in ways that affect diffusion. For example, higher cell density in cancer tissue means that more cell membranes block the diffusion path. DWI is a non-invasive method that yields useful information in the assessment of prostate cancer. Elsewhere in the body, DWI is useful for the diagnosis of acute stroke since diffusion abnormalities become visible within the first hours of the stroke event, whereas stroke-related changes that are visible on T_1, T_2, or proton density (PD) images take longer to develop (McRobbie et al., 2017, p. 303-306).

The data from the DW image are derived from the magnitude of the complex magnetic resonance (MR) signal, consisting of a real and an imaginary part. The strong gradient pulses used in DWI to achieve high diffusion sensitivity attenuate the signal. At the same time, the standard deviation of the underlying noise stays the same, irrespective of the attenuation.

The image intensity of magnitude MR images has been shown to be governed by a Rician distribution (Gudbjartsson & Patz, 1995), meaning that the real and imaginary parts of the noise signal are independent and follow a Gaussian distribution. This means that a Rician bias is introduced when taking the magnitude of the signal, M, received from two channels of the quadrature detector (Equation 1):

$$M = \sqrt{(\nu + n_r)^2 + n_i^2} \tag{1}$$

where ν is the signal intensity which is assumed real, n_r and n_i are independent Gaussian distributed random variables with the same standard deviation σ_g and zero mean, representing the real and imaginary channel (Koay & Basser, 2006).

The Rician bias is visualized by the horizontal lines in Figure 1. This indicates that signal with low signal-to-noise ratio (SNR) is more biased than signal with higher SNR, where SNR is defined as ν/σ_g.

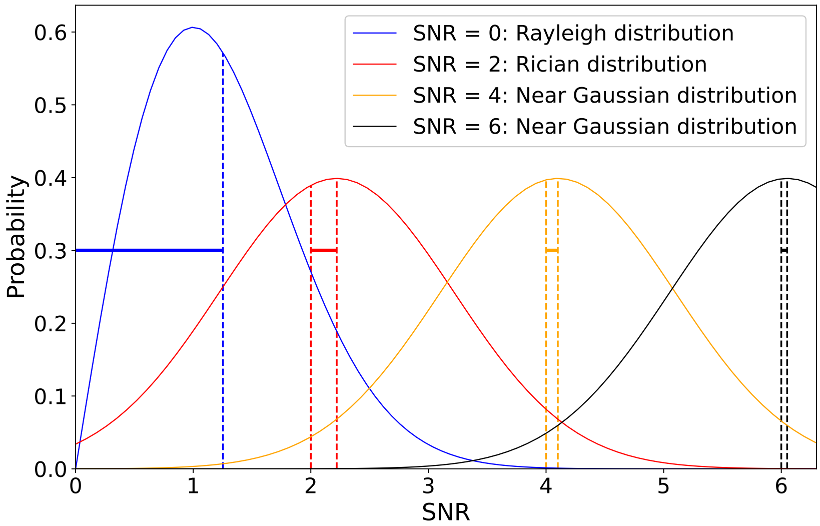


Figure 1: Rician probability distribution function for different SNR. The Rician bias is indicated by the shift of the distributions away from the mean (value of SNR). The Rician bias is especially noticeable for SNRs < 2, where the shift of the mean of the distribution is the greatest. For higher SNRs, the distributions become near Gaussian.

The expectation value of M can be calculated as (Koay & Basser, 2006):

$$\langle M \rangle = \frac{1}{2\sigma_g^2}\left(exp\left(-\frac{\nu^2}{4\sigma_g^2}\right)\sqrt{\frac{\pi}{2}}\sigma_g\left[(\nu^2 + 2\sigma_g^2)I_0\left(\frac{\nu^2}{4\sigma_g^2}\right) + I_1\left(\frac{\nu^2}{4\sigma_g^2}\right)\right]\right) \quad (2)$$

where I_0 and I_1 are the modified Bessel functions of order 0 and 1 respectively.

The Rician bias, RB, can be calculated as:

$$RB = \langle M \rangle - \nu \quad (3)$$

The diffusion-weighted signal attenuation in DW images can be described with a biexponential model that assumes two diffusion compartments in tissues. These compartments are the fast diffusion compartment, describing the extracellular diffusion process, and the slow diffusion compartment, describing the intracellular diffusion process (Riches, Hawtin, Charles-Edwards, & de Souza, 2009). In the context of this model, the diffusion-weighted signal can be expressed mathematically as a biexponential function (see Equation 4):

$$S(b) = S_0(fe^{-bD_1} + (1-f)e^{-bD_2}) \tag{4}$$

where S_0 is the signal without any diffusion-sensitizing gradients, D_1 and D_2 are the fast and slow diffusion components respectively, f is the relative fraction between D_1 and D_2, and b is a measure of the amount of diffusion weighting used in the DW image, called the b-value (see Equation 5):

$$b = \gamma^2 G^2 \delta^2 (\Delta - \delta/3) \tag{5}$$

where γ is the gyromagnetic ratio, G is the amplitude of the gradients, δ is the gradient duration and Δ is the diffusion time.

Theoretically, a higher b-value reduces the amount of T_2 shine-through while increasing the contrast due to a higher sensitivity to tissue diffusivity (Seo, Chang, Na, Kwon, & Lee, 2008). T_2 shine-through refers to a high signal on DWI images that is not due to restricted diffusion but rather a high T_2 signal, which shines through to the DW image. Higher b-values might be beneficial for detecting less prominent changes in diffusion. However, the Rician bias is more pronounced for higher b-values (see Figure 2).

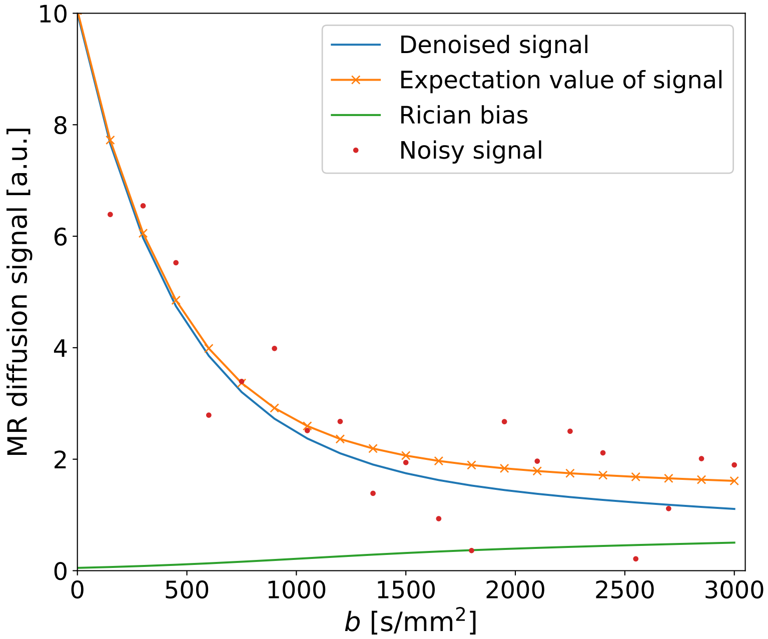


Figure 2: Simulated signal decay with the biexponential function, for SNR = 10 at $b = 0$, showing an increasing Rician bias with higher b-values. This implies that the measured signal (noisy signal) will be overestimated as the b-value increases.

One way to reduce the noise in DW images is to use deep neural networks (DNN) trained on simulated or patient data. (Barbieri, Gurney-Champion, Klaassen, & Thoeny, 2020) used a DNN to estimate diffusion-weighted related parameters but did not correct for the Rician bias in the DW images. The general architecture of the DNN from (Barbieri et al., 2020) was used when creating the DNN for this study.

A neural network (NN) consists of an input layer, hidden layers, and an output layer. The input layer is where the input data are inserted to be forwarded to the first hidden layer. The output data from one layer become the input data to the next through an activation function. The activation function performs nonlinear transformations of the data in the NN.

The optimization process of the NN uses a loss function that compares the input parameters of the network with the estimated parameters. The aim is to minimize the loss function, which is done by updating the weights associated with the trainable parameters in the NN. The process of minimizing the loss function is called optimization. The most common way to optimize a NN is by using gradient descent (Ruder, 2016). However, the optimizer used in this study was an Adam-optimizer (Kingma & Ba, 2017), as used by (Barbieri et al., 2020). The process of optimizing all the weights can be time-consuming and computationally demanding. However, once the weights have been fixed, the time it takes for the trained NN to process new data is extremely short (Bishop, 1994).

A NN has parameters that require manual intervention, called hyperparameters. Among these hyperparameters are epochs, batch size, and learning rate. The number of epochs describes how many times the NN is trained on the entire dataset. Batch size refers to how many training samples are trained on before updating the trainable parameters during the optimization. Learning rate determines how slow or fast the NN trains by updating the weights by a certain amount each time an optimization occurs. Finding the optimal learning rate can be challenging. A learning rate that is too slow can lead to very slow convergence. A very fast learning rate can instead lead to divergence or just fluctuations of the loss function around the minimum (Ruder, 2016).

There are two main ways a NN can learn: supervised learning and unsupervised learning. When using supervised learning, the training data include both the input and the output attributes (Kotsiantis, Zaharakis, Pintelas, et al., 2007). This method is typically fast, and the NN gives accurate results when tested on new data from the same distribution. However, it is not suitable for denoising of MR patient data, where the target attributes are unknown prior to training.

Unlike supervised learning, unsupervised learning is done without the use of target attributes (Alloghani, Al-Jumeily Obe, Mustafina, Hussain, & Aljaaf, 2020). This study used an MR diffusion signal model for network training, eliminating the need for target attributes.

This study aims to assess the potential of DNNs, using unsupervised learning for DWI data modeling and denoising of patient data. The DNN was trained on simulated data, using σ_g as a known value and trainable parameter to see which method performed better. To train the DNN directly on patient data, a predefined diffusion model is used. The results obtained by the DNN are compared with the results from OBSIDIAN (Kuczera, Alipoor, Langkilde, & Maier, 2021), which is a model-based, iterative fitting procedure used to simultaneously estimate true signal and underlying Gaussian noise on a pixel-by-pixel basis in magnitude MR images.

2 METHODS

2.1 Data generation and programming language

The open source machine learning framework $PyTorch$ (Paszke et al., 2019) version 1.11.0, based on the $Torch$-library was used to create the DNN. The programming language used was $Python$ version 3.9.7. All computations in this work were performed on a laptop equipped with a quad-core Intel(R) Core(TM) i5-1035G1 CPU @ 3.6 GHz, 8 GB DDR4 RAM and Microsoft Windows 10 (Microsoft Corp., Redmond, WA, USA).

2.2 Deep neural network architecture

The DNN consisted of an input layer and three hidden layers with 20 nodes each. Input data are DW signals at 20 different b-values, as defined in section 2.3. The output layer consisted of 3 nodes corresponding to the parameters D_1, D_2, and f. The activation function used in this study, and by (Barbieri et al., 2020), was the Exponential Linear Unit (ELU). With ELU, the inputs, x, that have a value of 0 or greater will output the same value. For x $<$ 0, the output value is $f(x)$ = α(e^x-1), where α is a positive constant (see Figure 3). The DNN is visualized in Figure 4:

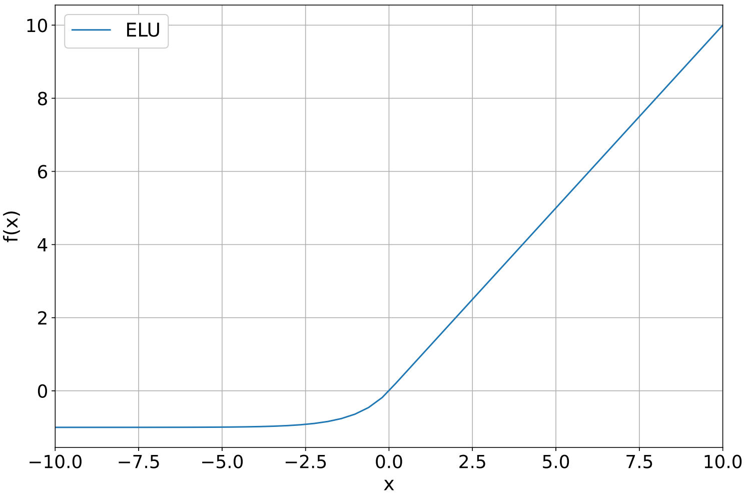


Figure 3: ELU activation function. Output values are: $f(x) = x$ if $x \geq 0$, and $f(x) = \alpha(e^x - 1)$ for $x < 0$, where α is a positive constant.

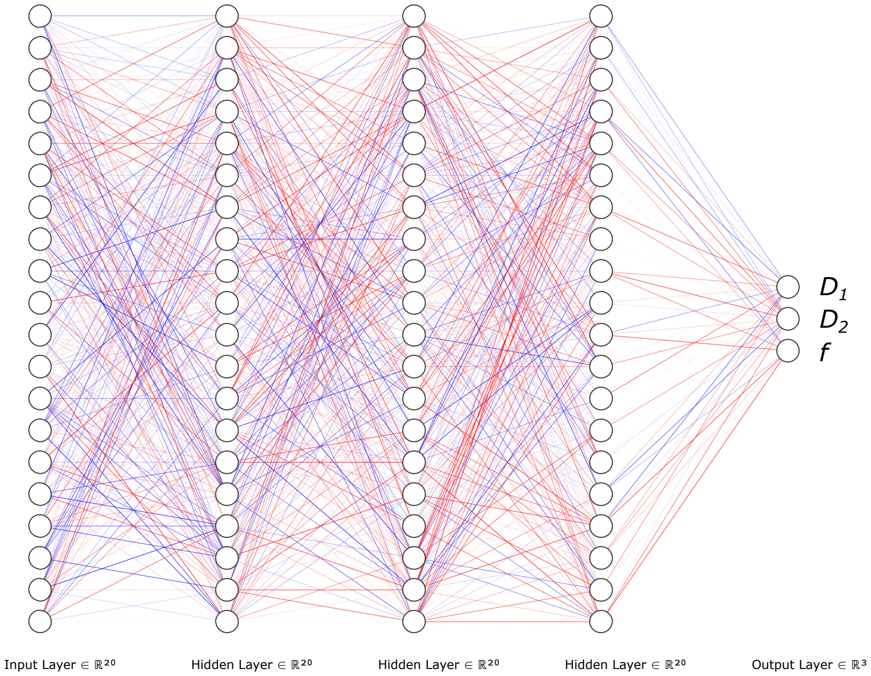


Figure 4: A visual representation of the trained DNN. The blue and red lines correspond to positive and negative weights respectively. Transparent lines indicate weaker weights whereas the more visible lines indicate stronger weights. The weights in this figure are randomly generated for visualization purposes.

An Adam-optimizer (Kingma & Ba, 2017) was used for training with the mean squared error (MSE) loss function between the noisy diffusion-weighted signals and the denoised diffusion-weighted signals. For simulated data, 20 DNNs were trained independently. The results from all DNNs were collected to achieve better statistics.

The idea of using DNNs in this study is outlined in the following. A visual representation is found in Figure 5. The simulated MR diffusion signal decay, referred to as noisy signal, serves as input to the DNN. The initial network weights are set to random values from a continuous uniform distribution ranging from $(-\sqrt{k}, \sqrt{k})$, where $k = 1/input\_nodes = 1/20$. After the first forward pass, estimates of D_1, D_2, and f are used to obtain the denoised signal with Eq.(4). Then, if not stated otherwise, Rician bias is added to the denoised signal with Eq.(3). This way, the DNN is trained to account for Rician bias when estimating D_1, D_2, and f. Next, the loss function is computed as the mean squared error (MSE) between the noisy and denoised signals. An Adam-optimizer (Kingma & Ba, 2017) was used for updating the weights of the DNN. In this case, as long as the predefined number of epochs has not been reached, the process is iterated, using the updated DNN. DNNs with and without Rician bias correction were trained to evaluate the effect of the correction.

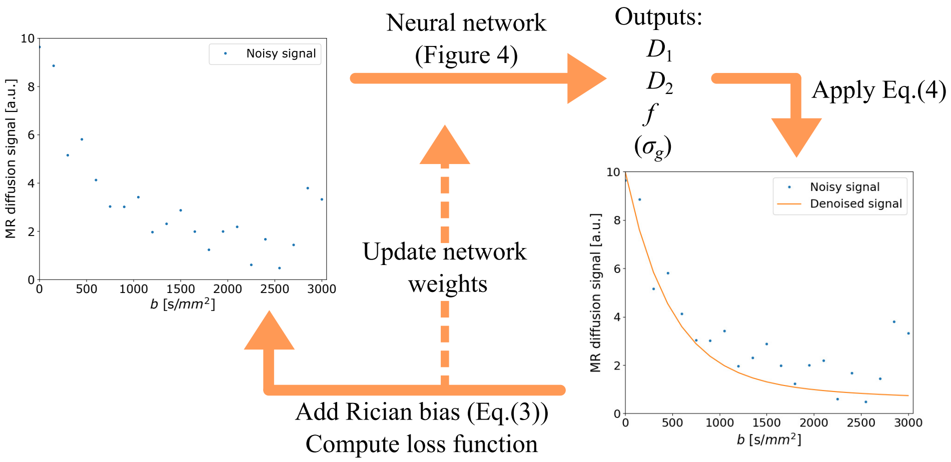


Figure 5: General idea behind the process of using DNNs in this work. The noisy signals are used as inputs to the DNN, which gives estimations of D_1, D_2, and f. With the estimations, Eq.(4) can be used to obtain a denoised signal. Rician bias is added to the denoised signal using Eq.(3), the loss function is computed, and the weights are updated.

2.3 Simulated data

A set of 21 b-values, linearly spaced from 0 to 3000 $\mathrm{s/mm^2}$, was used. For each noisy signal, values for D_1, D_2, and f were taken from a random uniform distribution (see Table 1). The range of the distributions include typical parameter values for normal and cancerous tissue (see section 2.3.4). The values were used to calculate signals at these b-values, with a biexponential function (see Equation 4).

Table 1: The lower and upper limits of the random, uniform distribution of the parameters D_1, D_2, and f.

	D_1 [µm^2/ms]	D_2 [µm^2/ms]	f
Range of random uniform distribution	[2, 2.4]	[0.1, 0.5]	[0.5, 0.9]

The noisy signals were normalized, and the $b = 0$ value was excluded (resulting in 20 b-values used) to avoid a signal decay component that would be caused by blood perfusion in patient data. At lower b-values, the perfusion effects that arise within the capillary networks are more noticeable (Shinmoto et al., 2012). The training data for all simulated data consisted of 100 000 simulated noisy signals.

Training data was generated for SNRs of 10, 20, 30, and 50 in all cases.

2.3.1 Training data - σ_g as single fixed value

The standard deviation σ_g used to generate the noise, that follows a Rician distribution, was defined as 1/SNR meaning $S_0 = 1$. This noise was added to the signals (see section 2.3) to create the training data.

2.3.2 Training data - σ_g as input parameter

The training data was created the same way as in section 2.3.1 with the exception that σ_g was not a single fixed value. SNR-values were chosen from a random uniform distribution ranging from SNR $= 10$ to SNR $= 50$. Values for σ_g, for each noisy signal, were calculated as S_0/SNR where $S_0 = 1$.

2.3.3 Training data - σ_g as trainable parameter

Instead of using a fixed value for σ_g, the DNN architecture was modified to estimate σ_g at the same time as D_1, D_2, and f. The σ_g values were randomly chosen, as in section 2.3.2. Rather than using a known value for σ_g for generating the noise (and Rician bias), estimates of σ_g were used. The fitting procedure used these σ_g values throughout the training process. As a result, the DNN's output layer consisted of 4 nodes (instead of 3, as in Figure 4), corresponding to estimations of D_1, D_2, f, and σ_g.

2.3.4 Test data

The test data used to assess the DNN's performance was created the same way as the training data, except fixed values for D_1, D_2, f, and σ_g were chosen. The values were: $D_1 = 2.2$ µm^2/ms, $D_2 = 0.4$ µm^2/ms, and $f = 0.8$ for normal prostate tissue and $D_1 = 2.2$ µm^2/ms, $D_2 = 0.2$ µm^2/ms, and $f = 0.6$ for cancerous prostate tissue. These values were chosen in approximate accordance with literature reference values (Langkilde et al., 2018). The value for σ_g was always defined as $1/\text{SNR}$. The test data consisted of 10 000 simulated noisy signals calculated with these values. A flowchart of the training process is shown in Figure 6.

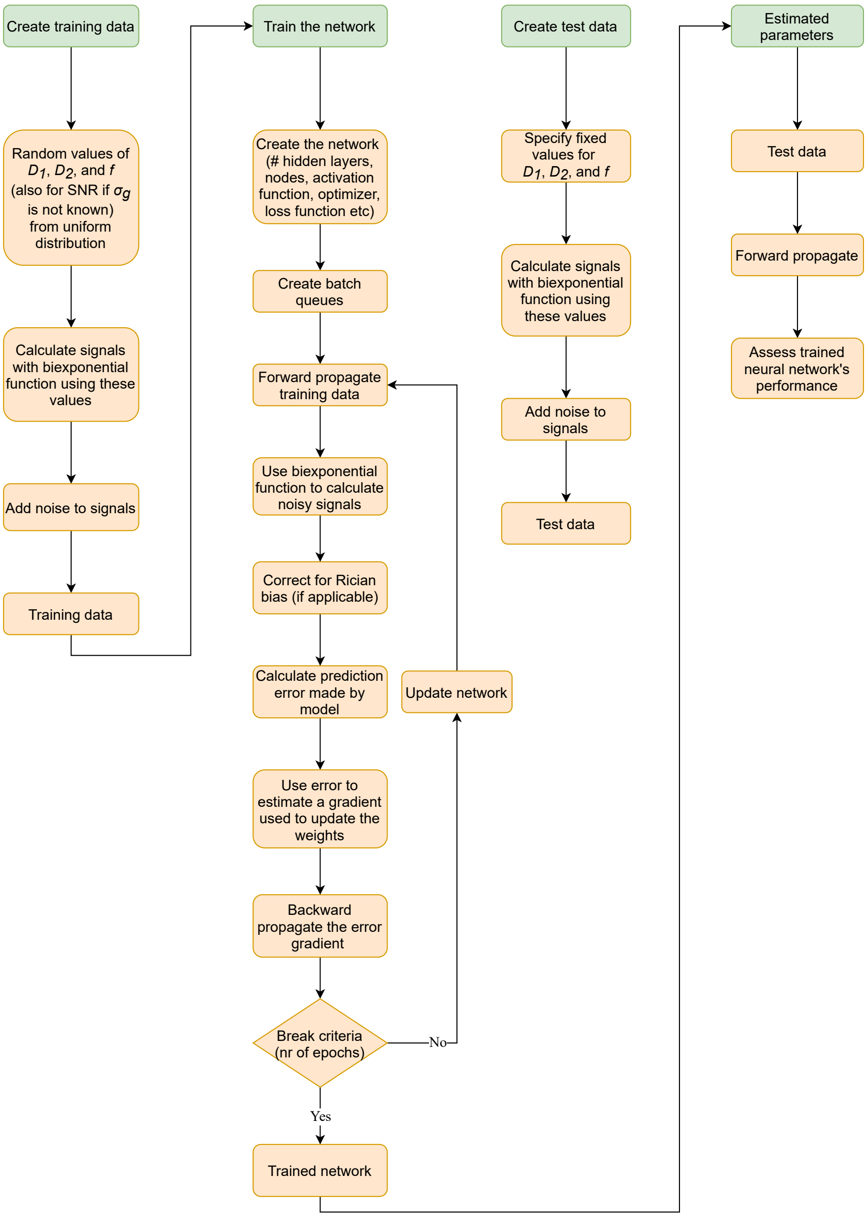

Create training data
Random values of D_1, D_2, and f (also for SNR if σ_g is not known) from uniform distribution
Calculate signals with biexponential function using these values
Add noise to signals
Training data
Train the network
Create the network (# hidden layers, nodes, activation function, optimizer, loss function etc)
Create batch queues
Forward propagate training data
Use biexponential function to calculate noisy signals
Correct for Rician bias (if applicable)
Calculate prediction error made by model
Use error to estimate a gradient used to update the weights
Backward propagate the error gradient
Break criteria (nr of epochs)
No
Yes
Update network
Trained network
Create test data
Specify fixed values for D_1, D_2, and f
Calculate signals with biexponential function using these values
Add noise to signals
Test data
Estimated parameters
Test data
Forward propagate
Assess trained neural network's performance

Figure 6: Flowchart of the method used for yielding estimations of the diffusion-weighted related parameters. The green boxes represent the general workflow whereas the orange boxes describe the workflow more in-depth. The break criterion is fulfilled when a predefined number of epochs has been reached.

2.4 Patient data - σ_g as input parameter

2.4.1 Patient data

The patient data was selected from the Göteborg-2 screening trial (Grenabo Bergdahl et al., 2016) and have been used in a previous publication (Kuczera et al., 2021). The data consisted of diffusion-weighted patient prostate data. The imaging protocol was performed on a Philips Ingenia CX 3T. Diffusion measurements were taken with three encoding directions for 21 linearly spaced b-values ranging from 0 - 3000 s/mm^2. Data from each patient consisted of 22 slices. The slice thickness was 3 mm. A single-shot echo-planar imaging (single-shot EPI) sequence was used. The echo time was 70.8 ms, the gradient strength was 80 mTm^{-1}, diffusion times were 34 ms, $\delta = 20$ ms, and the slew rate was 100 Tm^{-1}s^{-1} ms. No averaging was performed.

2.4.2 Patient training data

The patient training data consisted of diffusion-weighted signals gathered from three encoding directions. The patient training data used a σ_g-map as input to the DNN. The pixel-wise σ_g-map was computed by the OBSIDIAN algorithm. The σ_g-map and the diffusion-weighted signals were pixel-wise normalized by the signal value at $b = 0$. Data used was from one slice from one patient.

In the analysis, pixels with SNR < 5 were excluded to avoid unrealistic diffusion coefficients. The SNR values in every pixel of the image were computed by the OBSIDIAN algorithm. A Region of interest (ROI) was placed in the peripheral and transition zones of the prostate, respectively, by a certified radiologist. For patient data, one DNN was trained (as opposed to the simulated data, where 20 DNNs were trained independently).

2.4.3 Estimated parameters

The data in each pixel of the image were averaged from three encoding directions. The DNN estimated D_1, D_2, and f in each pixel of the image. The denoised image of a given b-value was obtained by calculating the diffusion-weighted signal in each pixel using Eq.(4). Furthermore, estimated D_1, D_2, and f values, in the ROIs mentioned in section 2.4.2, were compared between the DNN and OBSIDIAN. A flowchart of the training process using patient data is shown in Figure 7.

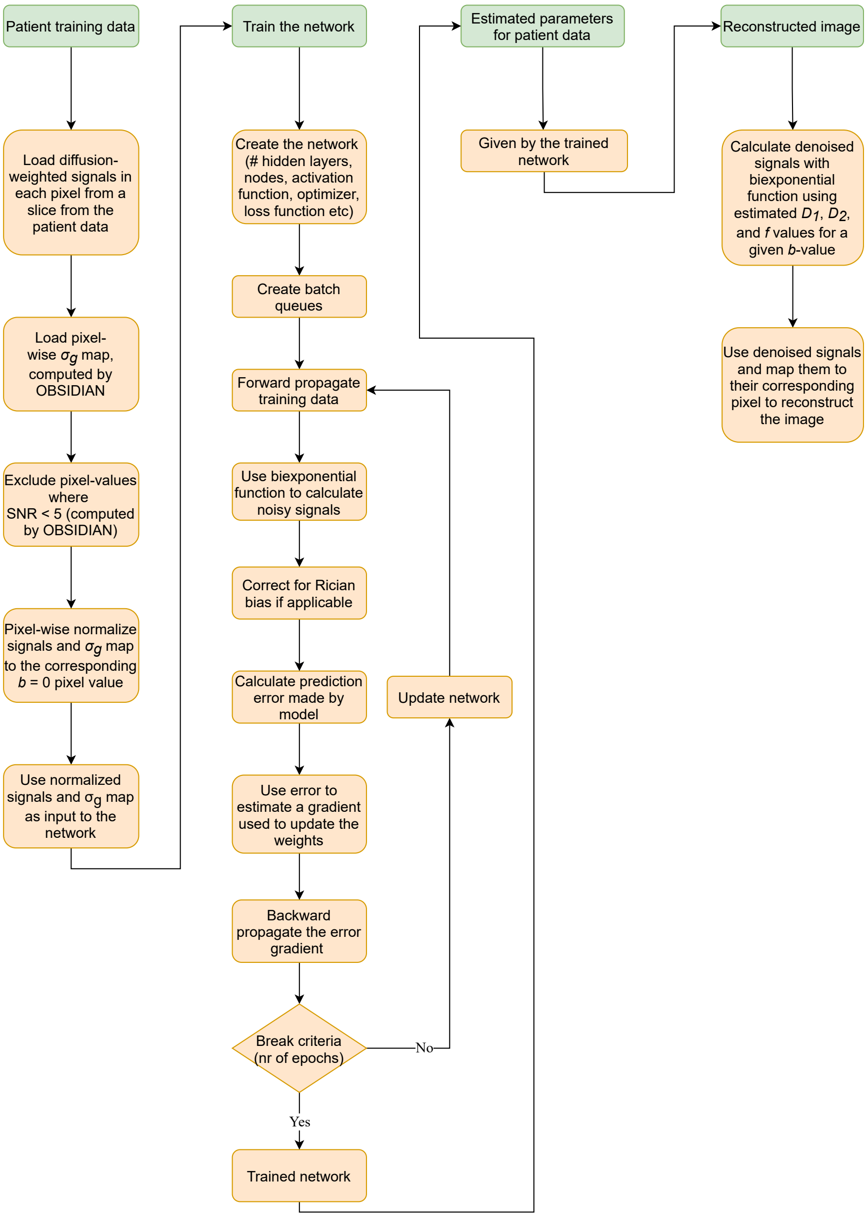

Patient training data
Load diffusion-weighted signals in each pixel from a slice from the patient data
Load pixel-wise σ_g map, computed by OBSIDIAN
Exclude pixel-values where SNR < 5 (computed by OBSIDIAN)
Pixel-wise normalize signals and σ_g map to the corresponding b = 0 pixel value
Use normalized signals and σ_g map as input to the network
Train the network
Create the network (# hidden layers, nodes, activation function, optimizer, loss function etc)
Create batch queues
Forward propagate training data
Use biexponential function to calculate noisy signals
Correct for Rician bias if applicable
Calculate prediction error made by model
Use error to estimate a gradient used to update the weights
Backward propagate the error gradient
Break criteria (nr of epochs)
No
Yes
Update network
Trained network
Estimated parameters for patient data
Given by the trained network
Reconstructed image
Calculate denoised signals with biexponential function using estimated D_1, D_2, and f values for a given b-value
Use denoised signals and map them to their corresponding pixel to reconstruct the image

Figure 7: Flowchart of the method used for reconstructing images of the prostate from patient prostate data. The DNN was trained using patient prostate data. The green boxes represent the general workflow whereas the orange boxes describe the workflow more in-depth. The break criterion is fulfilled when a predefined number of epochs has been reached.

3 RESULTS

3.1 Simulated data - σ_g as single fixed value

Estimations of D_1, D_2, and f from 20 trained DNNs with and without Rician bias correction and with $\sigma_g = 1/\mathrm{SNR}$ were collected. Rician bias correction greatly improved estimations of D_2 for low SNR (SNR $=$ 10, 20). Results for higher SNR (SNR $=$ 30, 50) were not affected as much (see Figures 8 and 9). Rician bias correction was used for all results in the following sections.

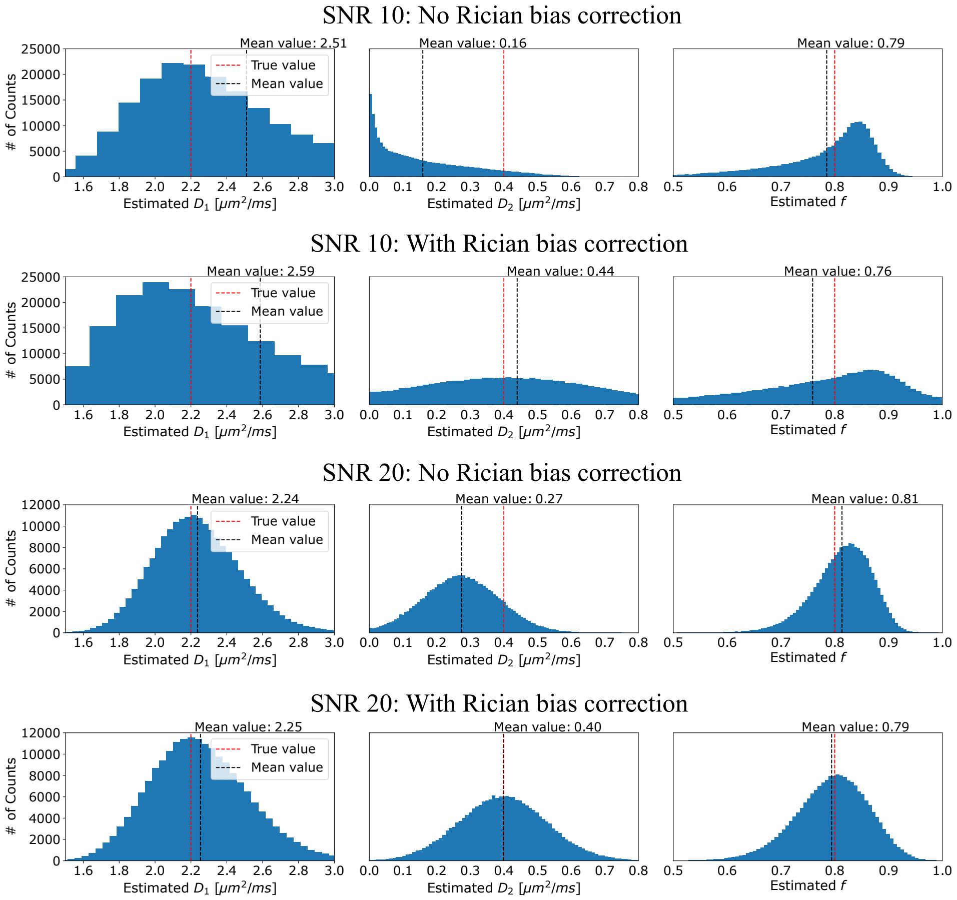
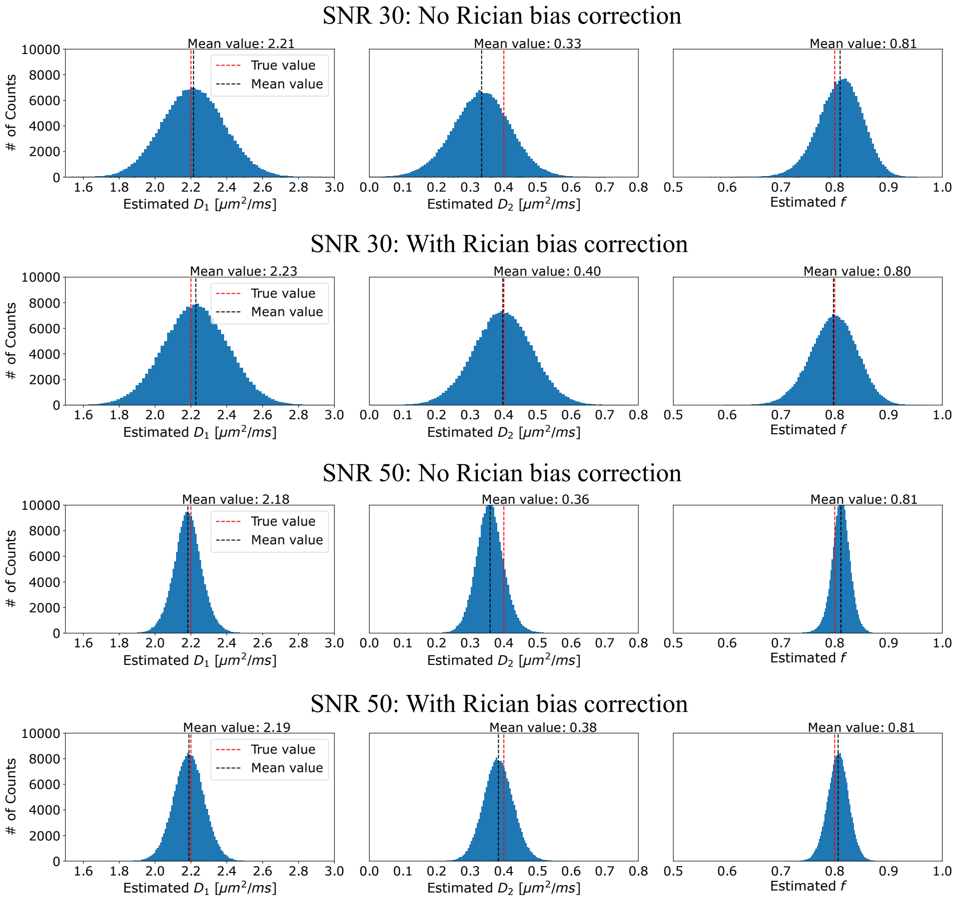


Figure 8: Histograms of the estimated parameters D_1, D_2, and f for normal tissue, with and without Rician bias correction applied for SNR = 10, 20. True values: $D_1 = 2.2$ $\mu\mathrm{m}^2/\mathrm{ms}$, $D_2 = 0.4$ $\mu\mathrm{m}^2/\mathrm{ms}$, and $f = 0.8$ represented by the red dashed line and the mean of the estimations represented by the black dashed line.



SNR 30: No Rician bias correction

SNR 30: With Rician bias correction

SNR 50: No Rician bias correction

SNR 50: With Rician bias correction

Figure 9: Histograms of the estimated parameters D_1, D_2, and f for normal tissue, with and without Rician bias correction applied for SNR = 30, 50. True values: $D_1 = 2.2$ µm^2/ms, $D_2 = 0.4$ µm^2/ms, and $f = 0.8$ represented by the red dashed line and the mean of the estimations represented by the black dashed line.

The mean and standard deviation of D_1, D_2, and f for normal and cancerous tissue were summarized (see Table 2).

Table 2: Estimated values of D_1, D_2, and f for normal and cancerous tissue for different SNR with and without Rician bias correction. The effect of Rician bias correction is especially noticeable in estimations of D_2 for low SNR (SNR = 10, 20).

Normal prostate tissue						
	No Rician bias correction			**With Rician bias correction**		
SNR	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f
10	2.51 ± 0.76	0.16 ± 0.14	0.79 ± 0.09	2.59 ± 1.09	0.44 ± 0.24	0.76 ± 0.15
20	2.24 ± 0.25	0.28 ± 0.11	0.81 ± 0.06	2.25 ± 0.28	0.40 ± 0.13	0.79 ± 0.07
30	2.21 ± 0.17	0.33 ± 0.09	0.81 ± 0.04	2.23 ± 0.18	0.40 ± 0.09	0.80 ± 0.04
50	2.18 ± 0.08	0.36 ± 0.04	0.81 ± 0.02	2.19 ± 0.09	0.38 ± 0.04	0.81 ± 0.02
True values	2.2	0.4	0.8	2.2	0.4	0.8
Cancerous prostate tissue						
	No Rician bias correction			**With Rician bias correction**		
SNR	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f
10	2.72 ± 1.37	0.18 ± 0.14	0.58 ± 0.14	3.06 ± 1.75	0.23 ± 0.16	0.56 ± 0.15
20	2.26 ± 0.35	0.19 ± 0.08	0.59 ± 0.07	2.30 ± 0.37	0.20 ± 0.08	0.59 ± 0.07
30	2.20 ± 0.18	0.20 ± 0.05	0.60 ± 0.04	2.21 ± 0.18	0.20 ± 0.05	0.60 ± 0.05
50	2.20 ± 0.08	0.20 ± 0.03	0.60 ± 0.02	2.20 ± 0.08	0.20 ± 0.03	0.60 ± 0.02
True values	2.2	0.2	0.6	2.2	0.2	0.6

3.2 Simulated data - σ_g as input parameter

Estimations of D_1, D_2, and f from 20 trained DNNs and different SNR, for normal and cancerous tissue (see Figures 10 and 11), generally yielded better results for cancerous tissue.

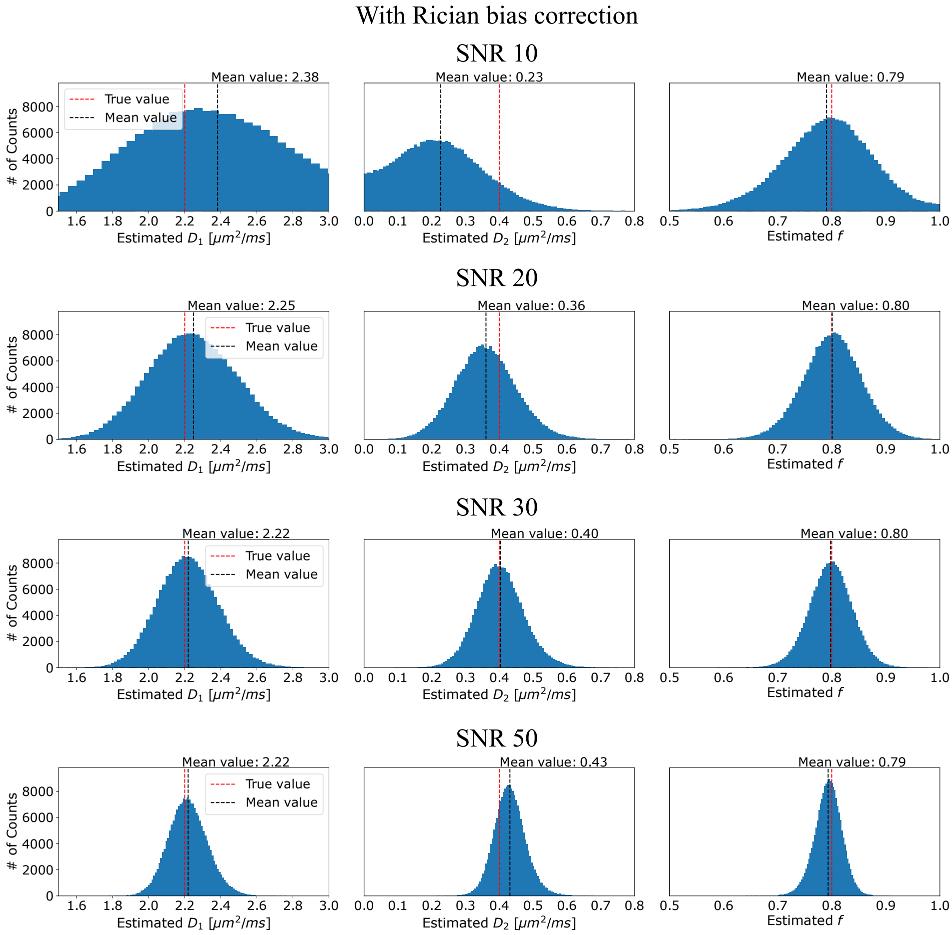


Figure 10: Histograms of the estimated parameters D_1, D_2, and f for normal tissue with true values: $D_1 = 2.2$ µm^2/ms, $D_2 = 0.4$ µm^2/ms, and $f = 0.8$ represented by the red dashed line and the mean of the estimations represented by the black dashed line. The results were a collection of 20 independently trained DNNs for each SNR (SNR = 10, 20, 30, 50).

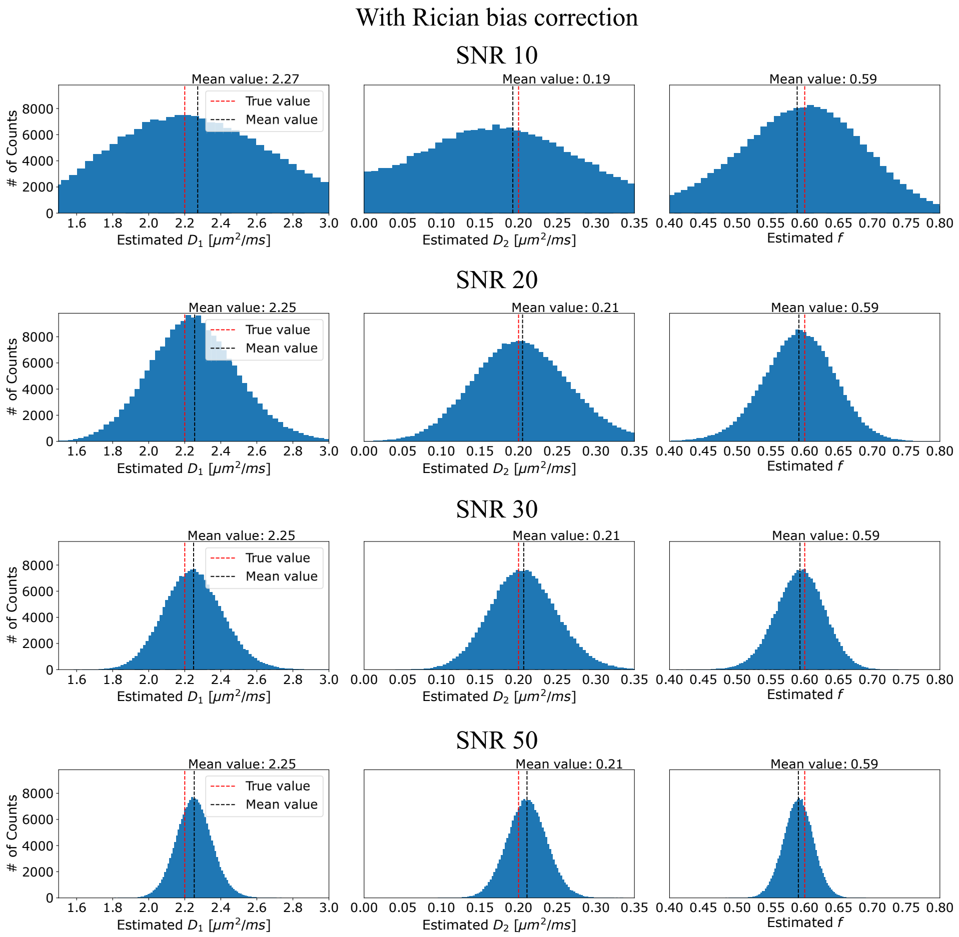


Figure 11: Histograms of the estimated parameters D_1, D_2, and f for cancerous tissue with true values: $D_1 = 2.2\ \mu\text{m}^2/\text{ms}$, $D_2 = 0.2\ \mu\text{m}^2/\text{ms}$, and $f = 0.6$ represented by the red dashed line and the mean of the estimations represented by the black dashed line. The results were a collection of 20 independently trained DNNs for each SNR (SNR = 10, 20, 30, 50).

Estimations of D_1, D_2, and f for normal and cancerous tissue were summarized (see Table 3). Results showed that the DNN performed better for cancerous prostate tissue, specifically when estimating D_2.

Table 3: The DNN's estimated values for D_1, D_2, and f for different SNR and for normal and cancerous tissue with Rician bias correction.

	Normal prostate tissue			**Cancerous prostate tissue**		
SNR	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f	D_1 [μm^2/ms]	D_2 [μm^2/ms]	f
10	2.36 ± 0.46	0.23 ± 0.13	0.79 ± 0.08	2.25 ± 0.46	0.19 ± 0.11	0.59 ± 0.10
20	2.25 ± 0.24	0.36 ± 0.09	0.80 ± 0.05	2.26 ± 0.25	0.21 ± 0.06	0.59 ± 0.05
30	2.22 ± 0.17	0.40 ± 0.07	0.80 ± 0.04	2.24 ± 0.16	0.20 ± 0.04	0.60 ± 0.03
50	2.22 ± 0.11	0.43 ± 0.05	0.80 ± 0.03	2.26 ± 0.10	0.21 ± 0.03	0.59 ± 0.02
True values	2.2	0.4	0.8	2.2	0.2	0.6

3.3 Simulated data - σ_g as trainable parameter

When training a single DNN, the results varied greatly each time it was run (see Figure 12).

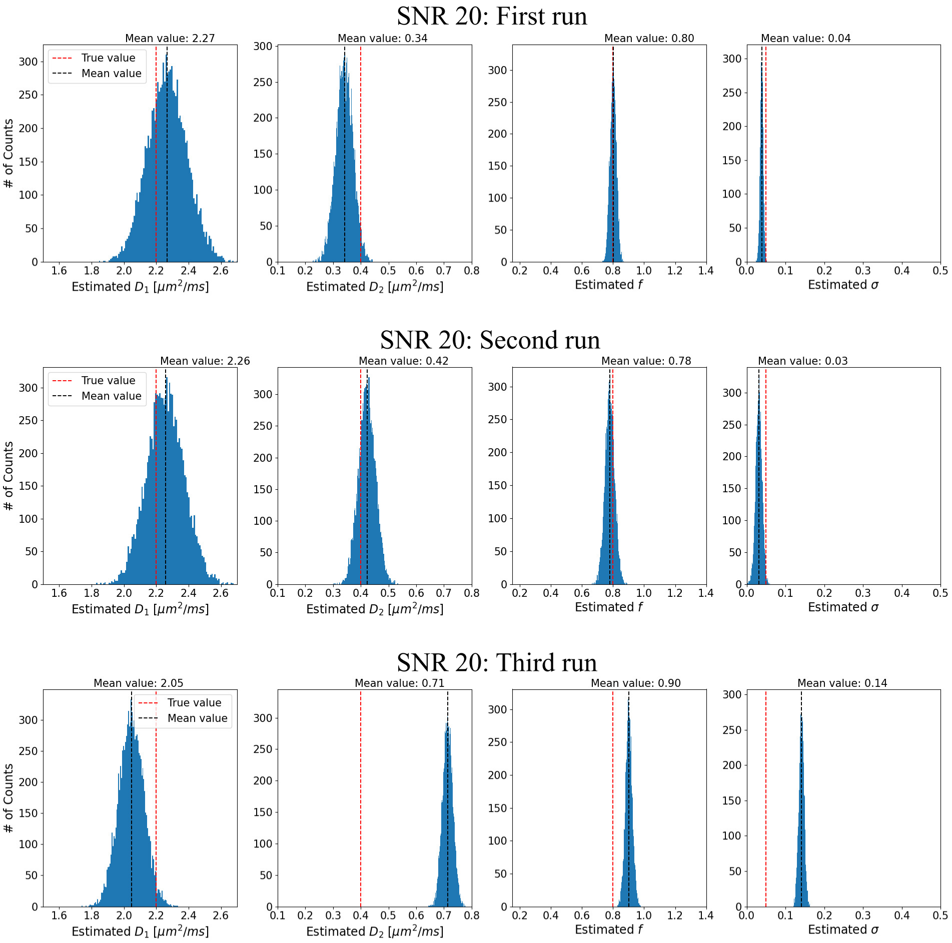


Figure 12: Histograms of D_1, D_2, f, and σ_g for normal prostate tissue. Rician bias correction was applied. True values: $D_1 = 2.2$ µm^2/ms, $D_2 = 0.4$ µm^2/ms, $f = 0.8$, and $\sigma_g = 1/20$ represented by the red dashed line. The mean of the estimations are represented by the black dashed line. Each run shows the result from a single trained DNN for SNR $= 20$. There is a clear variance in the results between each run.

By training 20 DNNs instead of 1, the distributions of the results changed (see Figure 13). The histograms show the superposition of the results from the 20 trained DNNs.

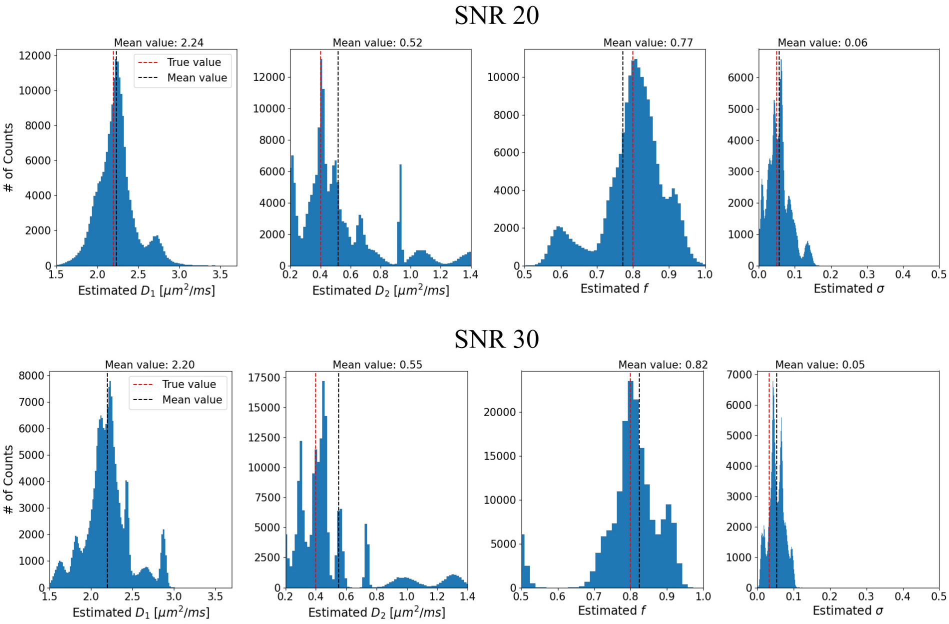


Figure 13: Histograms of D_1, D_2, f, and σ_g for normal prostate tissue with Rician bias correction applied. True values: $D_1 = 2.2$ µm^2/ms, $D_2 = 0.4$ µm^2/ms, $f = 0.8$, and $\sigma_g = 1/\text{SNR}$ represented by the red dashed line. The mean of the estimations are represented by the black dashed line. Results were a collection of 20 different trained DNNs for each SNR.

3.4 Learning rate

As the training sample size changed, so did the distribution of the results. A larger training sample size required a slower learning rate to achieve similar distributions (see Figure 14).

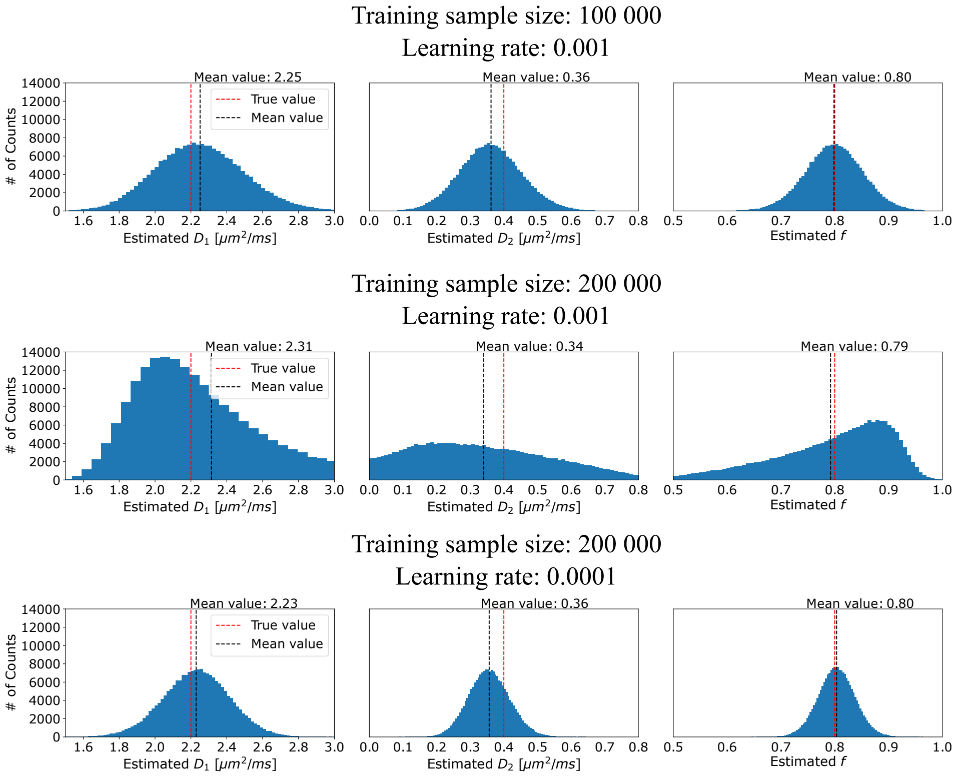


Figure 14: Histograms D_1, D_2, and f for normal prostate tissue and SNR $= 20$. True values are represented by the red dashed line and the mean of the estimations are represented by the black dashed line. Results clearly show the distribution of the estimations, and also the means, change when the training sample size is increased from 100 000 to 200 000 while keeping the learning rate constant. Lowering the learning rate to 0.0001 resolves this issue for a training sample size of 200 000.

3.5 Patient data - σ_g as input parameter

The signal map by the DNN was compared with the raw scan data and the signal map resulting from OBSIDIAN. The DNN reduces the noise in the image; however, introducing artifacts in the reconstructed image (see Figure 15).

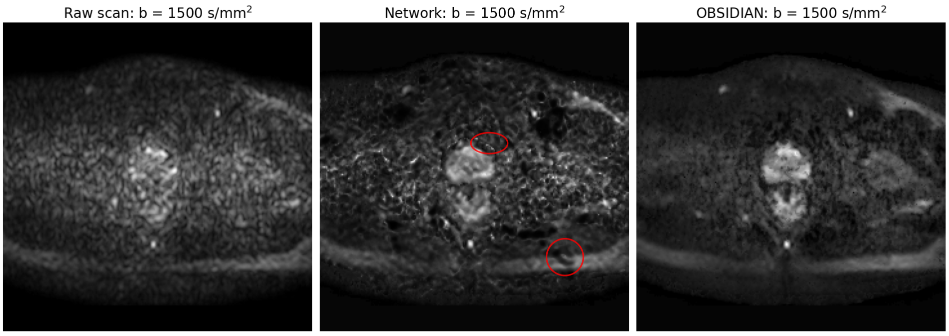

Raw scan: b = 1500 s/mm^2
Network: b = 1500 s/mm^2
OBSIDIAN: b = 1500 s/mm^2

Figure 15: The signal map, for $b = 1500\ \mathrm{s/mm}^2$, from the raw scan data, the DNN, and OBSIDIAN. The pixel signal intensity for all images were averaged from three encoding directions. Artifacts introduced by the DNN are marked with a red circle and a red ellipsoid. The artifact in the red circle shows extreme signal loss in the fat of the patient, which is not noticeable in either the raw scan data or the OBSIDIAN image. The artifact in the red ellipsoid shows a sharp difference in contrast in pixels around the prostate tissue.

Using the trained DNN on the same patient data resulted in a computation time of 1.9 ms. Estimations of D_1, D_2, and f, by the DNN and OBSIDIAN, in the peripheral and transition zone of the prostate were compared in Table 4.

Table 4: Estimations of D_1, D_2, and f by the DNN trained directly on patient data vs the results obtained from OBSIDIAN. Results were gathered in the peripheral and transition zone of the prostate.

Method	**Peripheral zone**			**Transition zone**		
	D_1 [µm^2/ms]	D_2 [µm^2/ms]	f	D_1 [µm^2/ms]	D_2 [µm^2/ms]	f
DNN	2.54 ± 0.75	0.61 ± 0.07	0.76 ± 0.10	2.74 ± 0.73	0.57 ± 0.06	0.66 ± 0.09
OBSIDIAN	2.78 ± 0.54	0.48 ± 0.21	0.77 ± 0.11	2.76 ± 0.55	0.46 ± 0.16	0.68 ± 0.10

3.6 Runtimes

The average time to train the DNNs was summarized in Table 5.

Table 5: The average runtimes when training the DNN with simulated and patient training data, with Rician bias correction applied.

Method	**Runtime [s]**
Simulated data - σ_g as single fixed value	715
Simulated data - σ_g as input parameter	715
Simulated data - σ_g as trainable parameter	715
Patient data - σ_g as input parameter	157

4 DISCUSSION

Using σ_g as a single fixed value gave very accurate and precise results; however, this method does not well represent patient data. Using σ_g as a trainable parameter proved an unsuitable method with poor results. Using σ_g as input was the most suitable method from this study and resulted in both accurate and precise estimations.

For simulated data, the average runtime was 488 s without Rician bias correction and 715 s with the correction. Correcting for the Rician bias increased runtimes by 47%. Despite slower runtimes, Rician bias correction is always recommended. This can prevent results that falsely imply a slower diffusion of D_2. For patient data, training the DNN on one slice took, on average, 157 s. Increasing the training sample size from 100 000 to 200 000 (see Figure 14) increased the runtime from 722 s to 1424 s. The learning rate did not affect the runtime in this case. When using the trained DNN on the same patient data, the runtime was 1.9 ms.

The effect of using Rician bias correction is noticeable for estimations of D_2 (see Table 2). D_2 is underestimated at low SNR due to the Rician bias. The Rician bias is not as prominent for SNR > 30 but should still be corrected since the SNR varies across the image. Some pixels have a low SNR, while others have a high SNR.

The DNN performed better for cancerous tissue since the signal intensity is less attenuated at higher b-values. Molecules that diffuse faster lead to stronger attenuation of the signal intensity (Woodhams et al., 2011). Molecules diffuse faster in less restricted areas (i.e., normal tissue), meaning the signal intensity will be lower. Therefore, the signal intensity is less attenuated in areas with more restricted diffusion (i.e., tumors).

For σ_g as a single fixed value, Rician bias correction greatly improved results (see Table 2). This was especially noticeable for low SNR (SNR $=$ 10, 20). Although these are great results, σ_g is not a single fixed value in clinical data that consists of many different σ_g values across the image. Therefore, this method is not suitable when the aim is to use the DNN on patient data.

Using σ_g as input to the DNN for simulated data, D_1, D_2, and f were especially accurate for SNR > 10. The Rician bias correction barely affected the results for SNR > 30. This is expected because the signals are less affected by the noise for higher SNR.

Using σ_g as input parameter generated better results than using σ_g as a trainable parameter. This is expected since using σ_g as a trainable parameter increases the complexity of the DNN. If σ_g is known, it is better to use it as input to the DNN.

The performance of the DNN for σ_g as a trainable parameter varied greatly (see Figure 12). When the noise is estimated at the same time as the parameters, it becomes increasingly harder to find unique solutions for the fitting procedure. Lacking noise knowledge prior to fitting will therefore lack consistency in case of isotropic diffusion (Veraart et al., 2013). This method is therefore not recommended. Instead, it is better to use σ_g as an input parameter, which showed promising results. However, if σ_g is unknown, it might be needed to optimize the DNN further or use another approach for estimating σ_g.

The performance of the DNN on patient data is to some degree dependent on the accuracy and variance of the noise estimates from OBSIDIAN. Application of the OBSIDIAN method to patient data leads to σ_g maps with high-frequency random fluctuations in the image space (Kuczera et al., 2021). Therefore, a low-pass filtered σ_g map (obtained from OBSIDIAN) was used as input to reconstruct the images (see Figure 15).

Using the DNN trained on patient data reduces the noise in the image. The image quality is not as good as OBSIDIAN's, and some artifacts are introduced in the reconstructed image. The results might improve in the future by using more slices from the same patient data or using data from more patients to train the DNN. Finding the optimal hyperparameters for the problem is time-consuming but can also improve results.

It is unclear why the same learning rate can not be used for different training sample sizes. Increasing the training sample size from 100 000 to 200 000 but keeping the same learning rate of 0.001 changes the distribution of the estimated parameters (see Figure 14). The first thought was that the issue originates from the optimizer. An Adam optimizer automatically updates the learning rate as training progresses. Therefore, a specific learning rate is not needed (Kingma & Ba, 2017). Removing the constant value for the learning rate and letting the optimizer automatically update it was tested, but the issue remained. No literature explaining this issue has been found. The mean of the estimations is still accurate, but the results become less precise. Also, results become less comparable if the distributions change for different training sample sizes. The number of epochs can be adjusted to achieve comparable distributions as an alternative to changing the learning rate.

The DNN does not restrict the range of the estimated parameters. This is due to the ELU activation function that was used. ELU allows for infinitely large values to be estimated, as shown in Figure 3. The DNN sometimes estimated D_1 to be higher than the self-diffusion coefficient of free water at 37°C (see Figure 10), which is 3.0 μm^2/ms (Le Bihan & Iima, 2015), (Woodhams et al., 2011). This might be avoided by using another activation function, e.g., sigmoid, where output values can only assume values in the range of 0 to 1. However, the estimations would need to be scaled, in this case, to use the sigmoid function.

5 CONCLUSION

The performance of DNNs trained by unsupervised learning was assessed. Results showed that having a known σ_g gave more accurate estimations of D_1, D_2, and f compared to σ_g estimated by the DNN. For simulated data, the DNN performed better for cancerous prostate tissue. The average runtime for training the DNN was 715 s with Rician bias correction. The most suitable method from this study is to use σ_g as input to the DNN. This more accurately (compared to using σ_g as a single fixed value) represents σ_g from clinical data and yields promising results for normal and cancerous prostate tissue. Using σ_g as a trainable parameter by the DNN is unsuitable due to the large errors associated with this method.

Using the DNN trained directly on patient data reduces the noise and bias in the image. The image quality is an improvement with respect to the raw data but does not quite match the image quality obtained by OBSIDIAN. Still, the use of DNNs look promising for DWI data modelling and denoising. Results might improve with further optimization of the DNN. However, some artifacts are introduced in the reconstructed image by the DNN. The real advantage of using a trained NN on patient data is not necessarily to achieve better quality of the reconstructed image, but rather to take advantage of the extremely short runtime required by a trained NN on new patient data. Using the trained DNN on the same patient data resulted in a runtime of 1.9 ms. This further proves the potential of DNNs in DWI data modeling and denoising regarding runtimes.

ACKNOWLEDGEMENTS

I thank my supervisors Stefan Kuczera, Stephan Maier, and Rolf A. Heckemann for providing a lot of help and guidance throughout this study.

References

Alloghani, M., Al-Jumeily Obe, D., Mustafina, J., Hussain, A., & Aljaaf, A. (2020, 01). Supervised and unsupervised learning for data science. In (chap. A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science). doi: 10.1007/978-3-030-22475-2_1

Barbieri, S., Gurney-Champion, O. J., Klaassen, R., & Thoeny, H. C. (2020). Deep learning how to fit an intravoxel incoherent motion model to diffusion-weighted MRI. *Magnetic Resonance in Medicine*, *83*(1), 312-321. doi: https://doi.org/10.1002/mrm.27910

Bishop, C. M. (1994). Neural networks and their applications. *Review of Scientific Instruments*, *65*(6), 1803-1832. doi: 10.1063/1.1144830

Grenabo Bergdahl, A., Wilderäng, U., Aus, G., Carlsson, S., Damber, J.-E., Frånlund, M., ... Hugosson, J. (2016). Role of magnetic resonance imaging in prostate cancer screening: A pilot study within the Göteborg randomised screening trial. *European Urology*, *70*(4), 566-573. doi: https://doi.org/10.1016/j.eururo.2015.12.006

Gudbjartsson, H., & Patz, S. (1995). The rician distribution of noisy MRI data. *Magnetic Resonance in Medicine*, *34*(6), 910-914. doi: https://doi.org/10.1002/mrm.1910340618

Kingma, D. P., & Ba, J. (2017). *Adam: A method for stochastic optimization.* (arXiv:1412.6980)

Koay, C. G., & Basser, P. J. (2006). Analytically exact correction scheme for signal extraction from noisy magnitude MR signals. *Journal of Magnetic Resonance*, *179*(2), 317-322. doi: https://doi.org/10.1016/j.jmr.2006.01.016

Kotsiantis, S. B., Zaharakis, I., Pintelas, P., et al. (2007). Emerging artificial intelligence applications in computer engineering : real word AI systems with applications in eHealth, HCI, information retrieval and pervasive technologies. In (Vol. 160, chap. Supervised machine learning: A review of classification techniques). Amsterdam.

Kuczera, S., Alipoor, M., Langkilde, F., & Maier, S. E. (2021). Optimized bias and signal inference in diffusion-weighted image analysis (OBSIDIAN). *Magnetic Resonance in Medicine*, *86*(5), 2716-2732. doi: https://doi.org/10.1002/mrm.28773

Langkilde, F., Kobus, T., Fedorov, A., Dunne, R., Tempany, C., Mulkern, R. V., & Maier, S. E. (2018). Evaluation of fitting models for prostate tissue characterization using extended-range b-factor diffusion-weighted imaging. *Magnetic Resonance in Medicine*, *79*(4), 2346-2358. doi: https://doi.org/10.1002/mrm.26831

Le Bihan, D., & Iima, M. (2015, 07). Diffusion magnetic resonance imaging: What water tells us about biological tissues. *PLOS Biology*, *13*(7), 1-13. doi: 10.1371/journal.pbio.1002203

McRobbie, D., Moore, E., Graves, M., & Prince, M. (2017). *MRI from Picture to Proton* (3rd ed.). Cambridge University Press.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. Retrieved from `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`

Riches, S. F., Hawtin, K., Charles-Edwards, E. M., & de Souza, N. M. (2009). Diffusion-weighted imaging of the prostate and rectal wall: comparison of biexponential and monoexponential modelled diffusion and associated perfusion coefficients. *NMR in Biomedicine*, *22*(3), 318-325. doi: https://doi.org/10.1002/nbm.1328

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, *abs/1609.04747*. (http://arxiv.org/abs/1609.04747)

Seo, H., Chang, K.-H., Na, D., Kwon, B., & Lee, D. (2008). High b-value diffusion (b = 3000 s/mm^2) MR imaging in cerebral gliomas at 3T: Visual and quantitative comparisons with b = 1000 s/mm^2. *American Journal of Neuroradiology*, *29*(3), 458–463. doi: 10.3174/ajnr.A0842

Shinmoto, H., Tamura, C., Soga, S., Shiomi, E., Yoshihara, N., Kaji, T., & Mulkern, R. V. (2012). An intravoxel incoherent motion diffusion-weighted imaging study of prostate cancer. *American Journal of Roentgenology*, *199*(4), W496-W500. (PMID: 22997399) doi: 10.2214/AJR.11.8347

Veraart, J., Rajan, J., Peeters, R. R., Leemans, A., Sunaert, S., & Sijbers, J. (2013). Comprehensive framework for accurate diffusion MRI parameter estimation. *Magnetic Resonance in Medicine*, *70*(4), 972-984. doi: https://doi.org/10.1002/mrm.24529

Woodhams, R., Ramadan, S., Stanwell, P., Sakamoto, S., Hata, H., Ozaki, M., … Inoue, Y. (2011). Diffusion-weighted imaging of the breast: Principles and clinical applications. *RadioGraphics*, *31*(4), 1059-1084. (PMID: 21768239) doi: 10.1148/rg.314105160