Contents

- Prepping the script
- Reading the data (LOW noise)
- MRF prior: Quadratic function (LOW noise)
- MRF prior: Discontinuity-adaptive Huber function (LOW noise)
- MRF prior: Discontinuity-adaptive Log function (LOW noise)
- Colormap plot of all three MRFs (LOW noise)
- Reading the data (MEDIUM noise)
- MRF prior: Quadratic function (MEDIUM noise)
- MRF prior: Discontinuity-adaptive Huber function (MEDIUM noise)
- MRF prior: Discontinuity-adaptive Log function (MEDIUM noise)
- Colormap plot of all three MRFs (MEDIUM noise)
- Reading the data (HIGH noise)
- MRF prior: Quadratic function (HIGH noise)
- MRF prior: Discontinuity-adaptive Huber function (HIGH noise)
- MRF prior: Discontinuity-adaptive Log function (HIGH noise)
- Colormap plot of all three MRFs (HIGH noise)

Prepping the script

```
clc; clear; close all;
addpath('../functions/');
actual = abs(double(imread('../../data/mri_image_noiseless.png'))/255);
```

Reading the data (LOW noise)

```
fprintf('Analysis for LOW noise level image\n');
img = abs(double(imread('../../data/mri_image_noise_level_low.png'))/255);
```

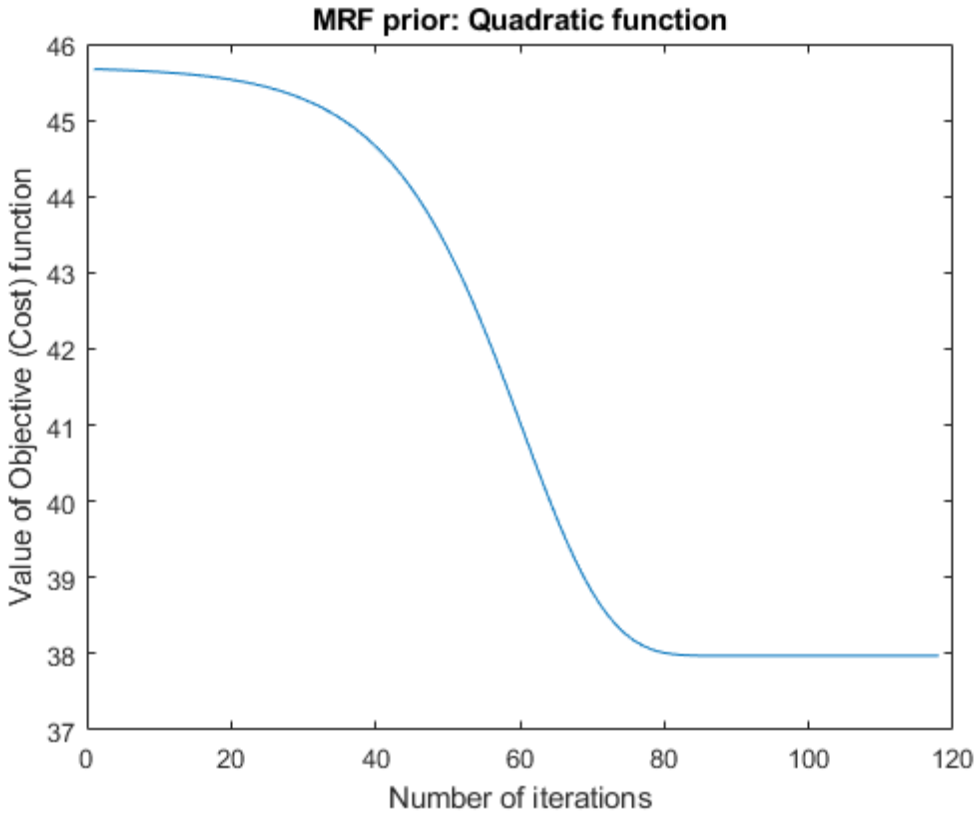
```
Analysis for LOW noise level image
```

MRF prior: Quadratic function (LOW noise)

```
fprintf('MRF prior: Quadratic function (LOW noise)\n');
alpha_opt = 0.085; gamma_opt = 1; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
quad_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters Testing
fprintf('alpha(a) = %f\n', alpha_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
```

```
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Quadratic function');
```

```
MRF prior: Quadratic function (LOW noise)
RRMSE(noiseless, noisy) : 0.051898
alpha(a) = 0.085000
RRMSE(a) = 0.046735
RRMSE(0.8a) = 0.046941
RRMSE(1.2a) = 0.046842
```

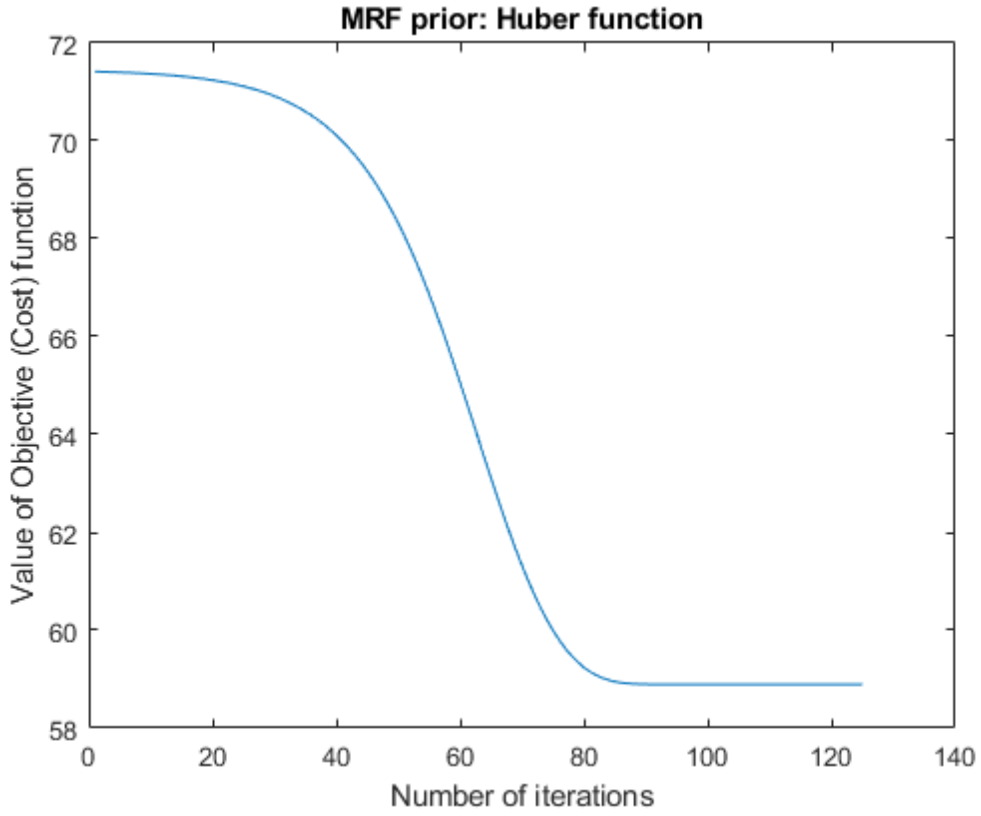


MRF prior: Discontinuity-adaptive Huber function (LOW noise)

```
fprintf('MRF prior: Discontinuity-adaptive Huber function (LOW noise)\n');
alpha_opt = 0.373248; gamma_opt = 0.026214; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
huber_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);
```

```
% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);

alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Huber function');
```

```
MRF prior: Discontinuity-adaptive Huber function (LOW noise)
RRMSE(noiseless, noisy) : 0.051898
alpha(a) = 0.373248, gamma(b) = 0.026214
RRMSE(a, b) = 0.043781
RRMSE(0.8a, b) = 0.044381
RRMSE(a, 0.8b) = 0.043818
RRMSE(1.2a, b) = 0.043991
RRMSE(a, 1.2b) = 0.043986
```

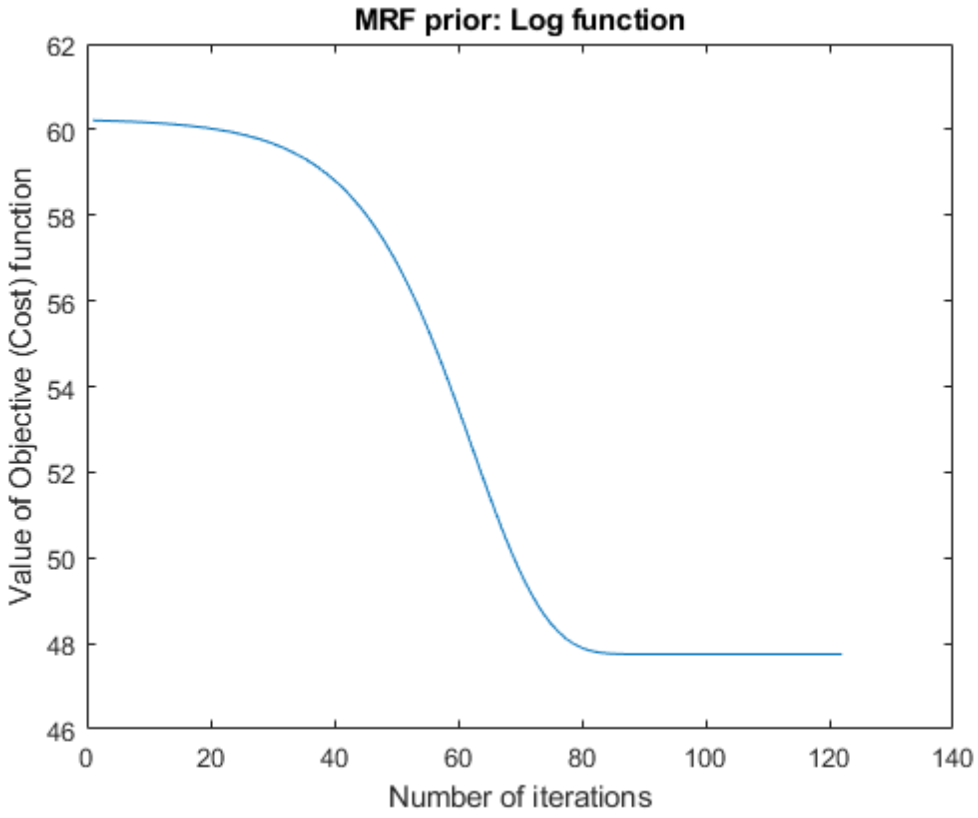


MRF prior: Discontinuity-adaptive Log function (LOW noise)

```
fprintf('MRF prior: Discontinuity-adaptive Log function (LOW noise)\n');
alpha_opt = 0.210; gamma_opt = 0.502; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
log_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);

alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);
```

```
% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Log function');
```

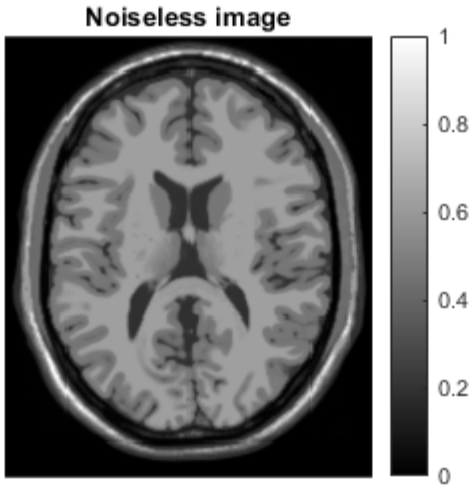
```
MRF prior: Discontinuity-adaptive Log function (LOW noise)
RRMSE(noiseless, noisy) : 0.051898
alpha(a) = 0.210000, gamma(b) = 0.502000
RRMSE(a, b) = 0.046154
RRMSE(0.8a, b) = 0.046264
RRMSE(a, 0.8b) = 0.046001
RRMSE(1.2a, b) = 0.046569
RRMSE(a, 1.2b) = 0.046263
```

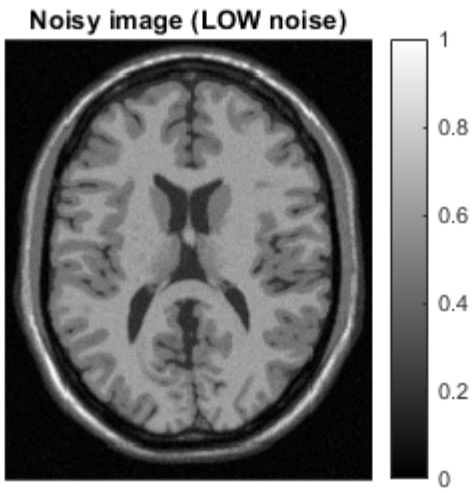


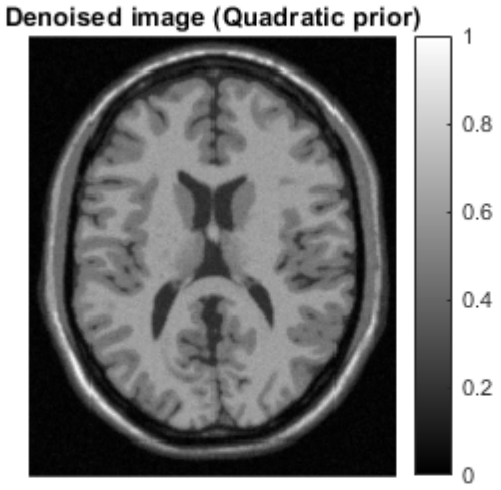
Colormap plot of all three MRFs (LOW noise)

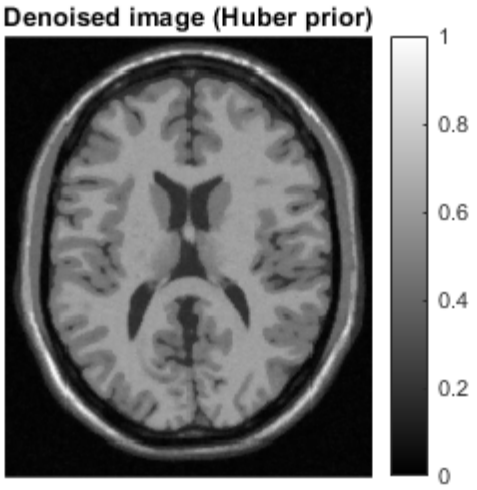
```
mx = max([max(img), max(actual), max(log_denoised), max(huber_denoised), max(quad_denoised)]);
mn = min([min(img), min(actual), min(log_denoised), min(huber_denoised), min(quad_denoised)]);
magnification = 200;

figure;
imshow(actual,[mn, mx], 'InitialMagnification', magnification);
title('Noiseless image');
colorbar

figure;
imshow(img,[mn, mx], 'InitialMagnification', magnification);
title('Noisy image (LOW noise)');
colorbar

figure;
imshow(quad_denoised,[mn, mx], 'InitialMagnification', magnification);
```

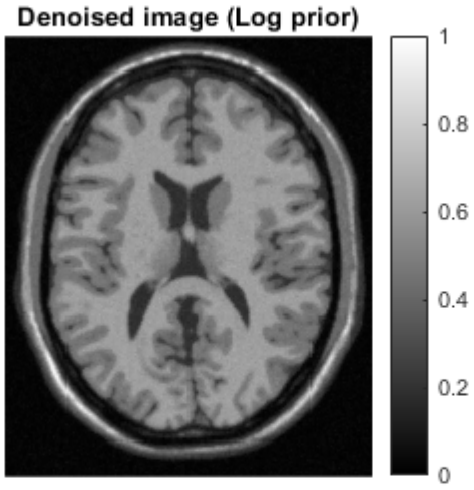
```
title('Denoised image (Quadratic prior)');
colorbar

figure;
imshow(huber_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Huber prior)');
colorbar

figure;
imshow(log_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Log prior)');
colorbar
```






Denoised image (Quadratic prior)
1
0.8
0.6
0.4
0.2
0


Denoised image (Huber prior)
1
0.8
0.6
0.4
0.2
0



Reading the data (MEDIUM noise)

```
fprintf('Analysis for MEDIUM noise level image\n');
img = abs(double(imread('../../data/mri_image_noise_level_medium.png'))/255);
```

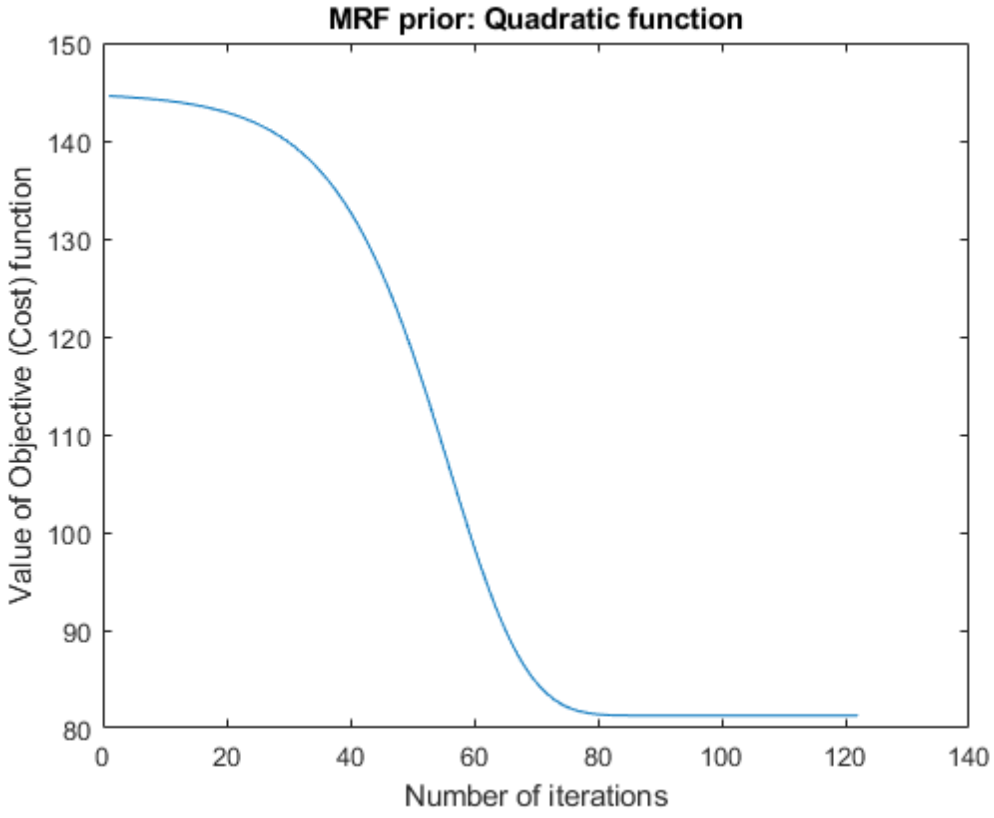
```
Analysis for MEDIUM noise level image
```

MRF prior: Quadratic function (MEDIUM noise)

```
fprintf('MRF prior: Quadratic function (MEDIUM noise)\n');
alpha_opt = 0.18; gamma_opt = 1; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
quad_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters Testing
fprintf('alpha(a) = %f\n', alpha_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a) = %f\n', error);

% Objective function plot
figure;
```

```
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Quadratic function');
```

```
MRF prior: Quadratic function (MEDIUM noise)
RRMSE(noiseless, noisy) : 0.131247
alpha(a) = 0.180000
RRMSE(a) = 0.116103
RRMSE(0.8a) = 0.116441
RRMSE(1.2a) = 0.116499
```

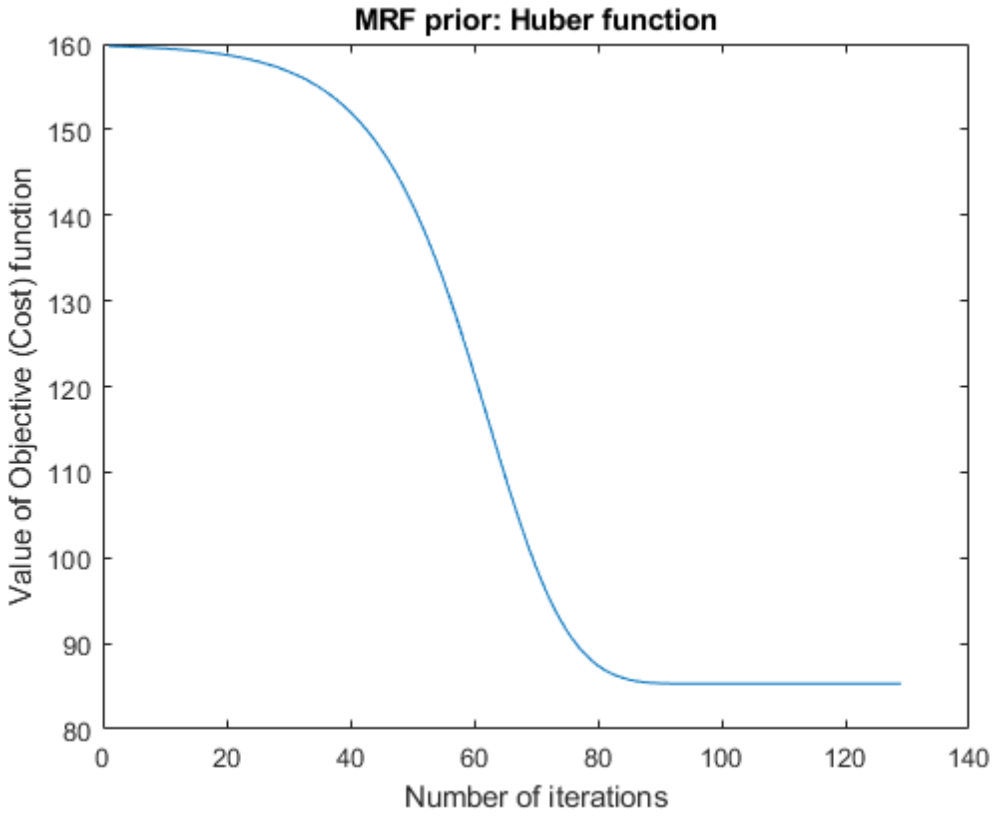


MRF prior: Discontinuity-adaptive Huber function (MEDIUM noise)

```
fprintf('MRF prior: Discontinuity-adaptive Huber function (MEDIUM noise)\n');
alpha_opt = 0.5184; gamma_opt = 0.036; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
huber_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);
```

```
alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Huber function');
```

```
MRF prior: Discontinuity-adaptive Huber function (MEDIUM noise)
RRMSE(noiseless, noisy) : 0.131247
alpha(a) = 0.518400, gamma(b) = 0.036000
RRMSE(a, b) = 0.112744
RRMSE(0.8a, b) = 0.113579
RRMSE(a, 0.8b) = 0.112542
RRMSE(1.2a, b) = 0.115040
RRMSE(a, 1.2b) = 0.113376
```

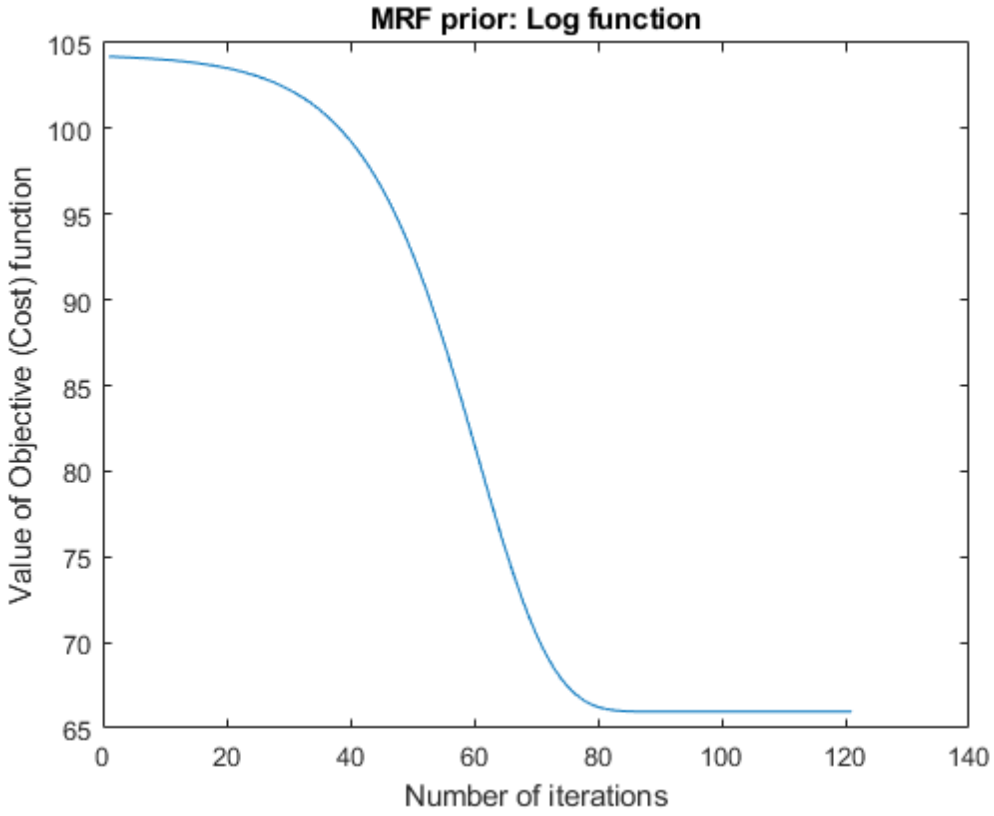


MRF prior: Discontinuity-adaptive Log function (MEDIUM noise)

```
fprintf('MRF prior: Discontinuity-adaptive Log function (MEDIUM noise)\n');
alpha_opt = 0.24; gamma_opt = 0.448; eps = 1e-8; epochs = 1e9;
```

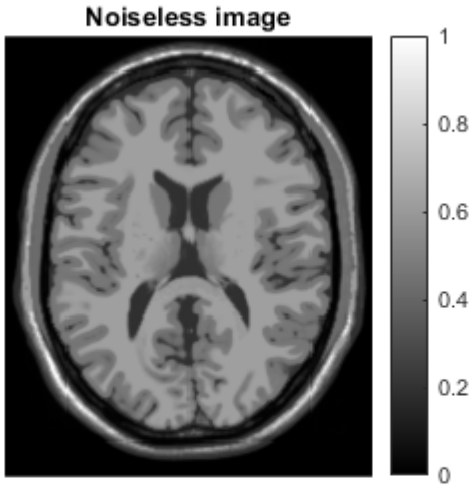
```
alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
log_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);

alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Log function');
```

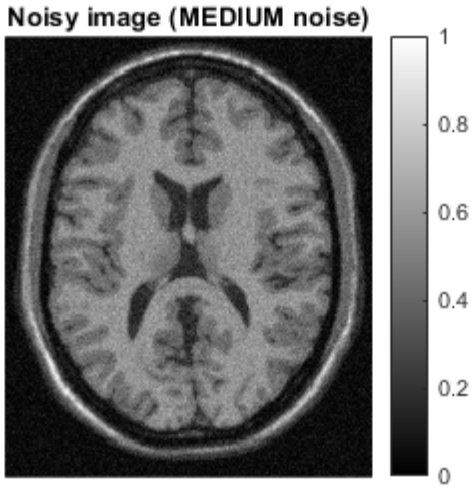
```
MRF prior: Discontinuity-adaptive Log function (MEDIUM noise)
RRMSE(noiseless, noisy) : 0.131247
alpha(a) = 0.240000, gamma(b) = 0.448000
RRMSE(a, b) = 0.116804
RRMSE(0.8a, b) = 0.118418
RRMSE(a, 0.8b) = 0.116908
RRMSE(1.2a, b) = 0.115774
RRMSE(a, 1.2b) = 0.116740
```

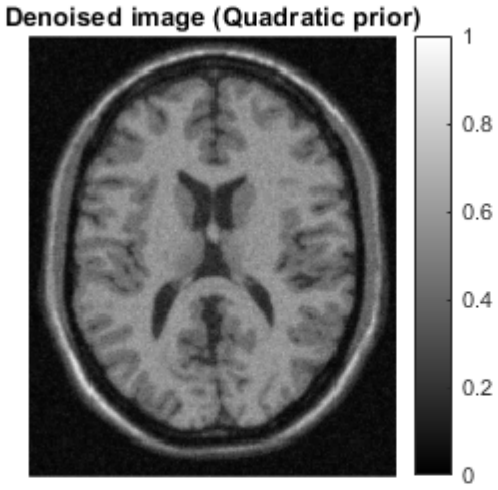


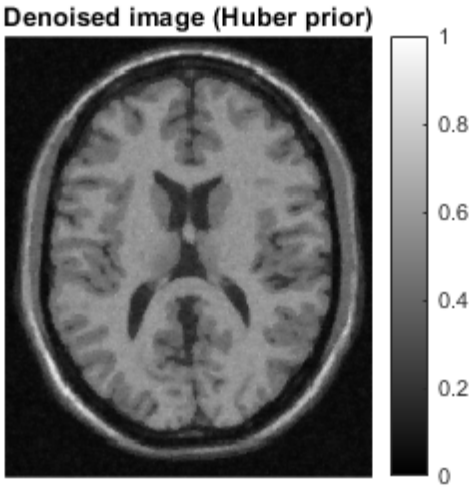
Colormap plot of all three MRFs (MEDIUM noise)

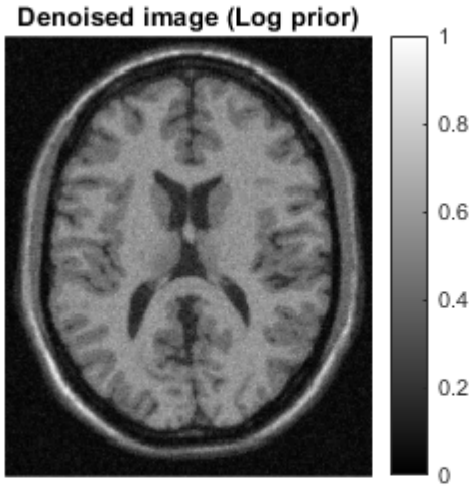
```
mx = max([max(img), max(actual), max(log_denoised), max(huber_denoised), max(quad_denoised)]);
mn = min([min(img), min(actual), min(log_denoised), min(huber_denoised), min(quad_denoised)]);
magnification = 200;

figure;
imshow(actual,[mn, mx], 'InitialMagnification', magnification);
title('Noiseless image');
colorbar

figure;
imshow(img,[mn, mx], 'InitialMagnification', magnification);
title('Noisy image (MEDIUM noise)');
colorbar

figure;
imshow(quad_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Quadratic prior)');
colorbar

figure;
imshow(huber_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Huber prior)');
colorbar

figure;
imshow(log_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Log prior)');
colorbar
```


Noiseless image
1
0.8
0.6
0.4
0.2
0


Noisy image (MEDIUM noise)
1
0.8
0.6
0.4
0.2
0


Denoised image (Quadratic prior)
1
0.8
0.6
0.4
0.2
0


Denoised image (Huber prior)
1
0.8
0.6
0.4
0.2
0



Reading the data (HIGH noise)

```
fprintf('Analysis for HIGH noise level image\n');
img = abs(double(imread('../../data/mri_image_noise_level_high.png'))/255);
```

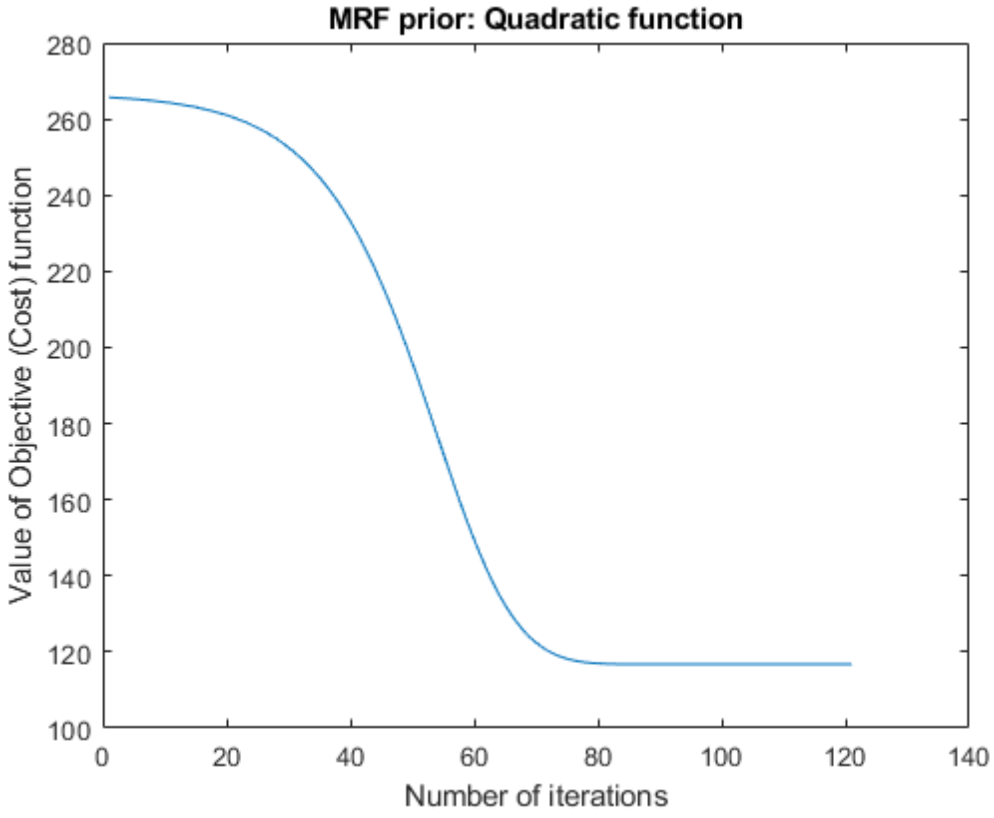
```
Analysis for HIGH noise level image
```

MRF prior: Quadratic function (HIGH noise)

```
fprintf('MRF prior: Quadratic function (HIGH noise)\n');
alpha_opt = 0.24; gamma_opt = 1; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
quad_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters Testing
fprintf('alpha(a) = %f\n', alpha_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @quad_grad, @quad_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a) = %f\n', error);

% Objective function plot
figure;
```

```
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Quadratic function');
```

```
MRF prior: Quadratic function (HIGH noise)
RRMSE(noiseless, noisy) : 0.155534
alpha(a) = 0.240000
RRMSE(a) = 0.127077
RRMSE(0.8a) = 0.127766
RRMSE(1.2a) = 0.127648
```

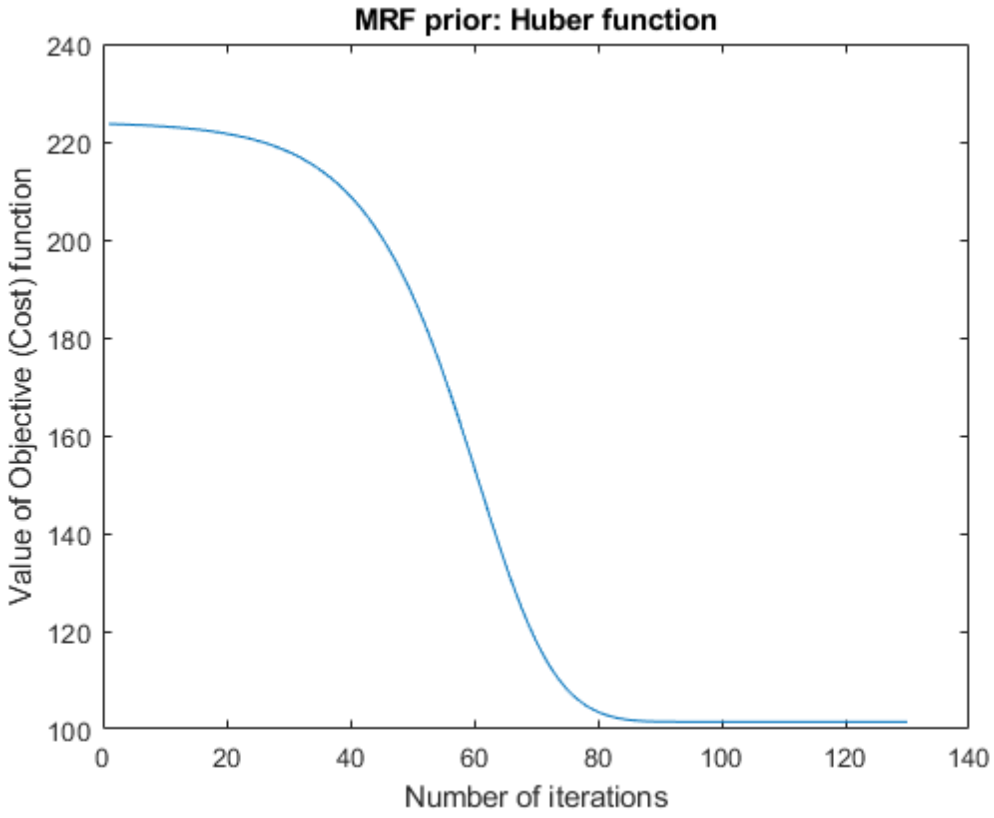


MRF prior: Discontinuity-adaptive Huber function (HIGH noise)

```
fprintf('MRF prior: Discontinuity-adaptive Huber function (HIGH noise)\n');
alpha_opt = 0.48; gamma_opt = 0.06; eps = 1e-8; epochs = 1e9;

alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
huber_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);
```

```
alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @huber_grad, @huber_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Huber function');
```

```
MRF prior: Discontinuity-adaptive Huber function (HIGH noise)
RRMSE(noiseless, noisy) : 0.155534
alpha(a) = 0.480000, gamma(b) = 0.060000
RRMSE(a, b) = 0.124114
RRMSE(0.8a, b) = 0.126240
RRMSE(a, 0.8b) = 0.124157
RRMSE(1.2a, b) = 0.125333
RRMSE(a, 1.2b) = 0.124559
```

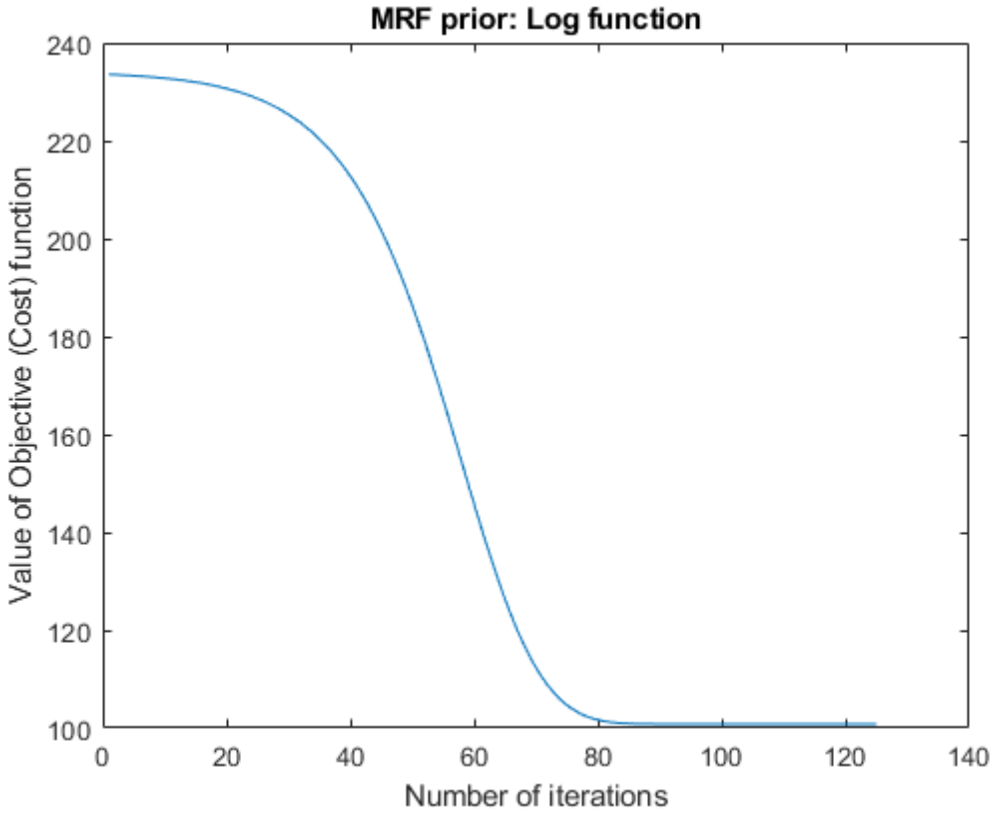


MRF prior: Discontinuity-adaptive Log function (HIGH noise)

```
fprintf('MRF prior: Discontinuity-adaptive Log function (HIGH noise)\n');
alpha_opt = 0.4; gamma_opt = 0.5632; eps = 1e-8; epochs = 1e9;
```

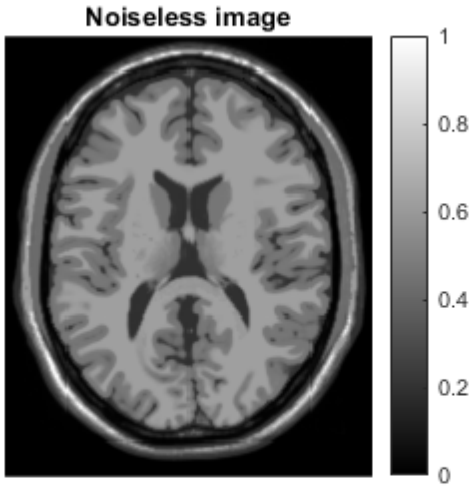
```
alpha = alpha_opt; gamma = gamma_opt;
[noisy, costs] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
log_denoised = abs(noisy);

% RRMSE
error = rrmse(actual, img);
fprintf('RRMSE(noiseless, noisy) : %f\n',error);

% Optimal Parameters
fprintf('alpha(a) = %f, gamma(b) = %f\n', alpha_opt, gamma_opt);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, b) = %f\n', error);

alpha = 0.8 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(0.8a, b) = %f\n', error);

alpha = alpha_opt; gamma = 0.8 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 0.8b) = %f\n', error);

alpha = 1.2 * alpha_opt; gamma = gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(1.2a, b) = %f\n', error);

alpha = alpha_opt; gamma = 1.2 * gamma_opt;
[noisy, ~] = descent(img, eps, alpha, gamma, epochs, @log_grad, @log_cost);
error = rrmse(actual, noisy);
fprintf('RRMSE(a, 1.2b) = %f\n', error);

% Objective function plot
figure;
plot(costs);
xlabel('Number of iterations')
ylabel('Value of Objective (Cost) function');
title('MRF prior: Log function');
```

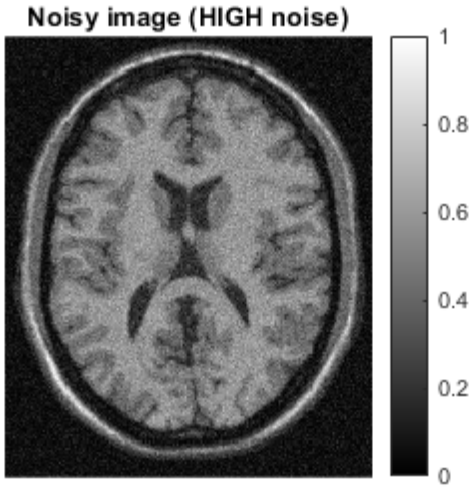
```
MRF prior: Discontinuity-adaptive Log function (HIGH noise)
RRMSE(noiseless, noisy) : 0.155534
alpha(a) = 0.400000, gamma(b) = 0.563200
RRMSE(a, b) = 0.126402
RRMSE(0.8a, b) = 0.128116
RRMSE(a, 0.8b) = 0.155406
RRMSE(1.2a, b) = 0.126733
RRMSE(a, 1.2b) = 0.126466
```

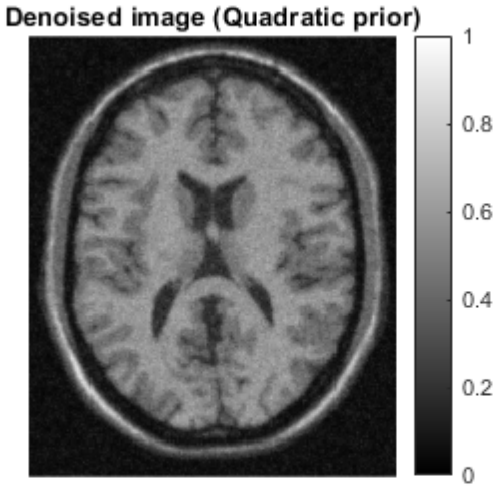


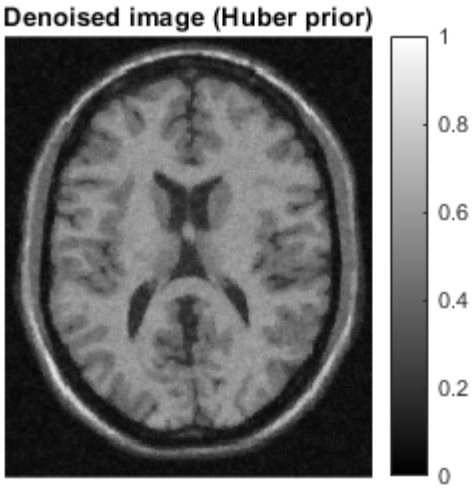
Colormap plot of all three MRFs (HIGH noise)

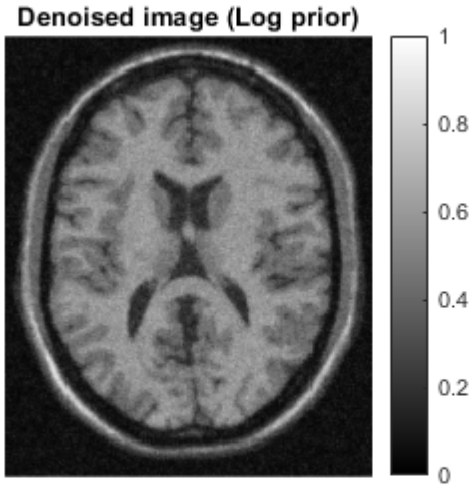
```
mx = max([max(img), max(actual), max(log_denoised), max(huber_denoised), max(quad_denoised)]);
mn = min([min(img), min(actual), min(log_denoised), min(huber_denoised), min(quad_denoised)]);
magnification = 200;

figure;
imshow(actual,[mn, mx], 'InitialMagnification', magnification);
title('Noiseless image');
colorbar

figure;
imshow(img,[mn, mx], 'InitialMagnification', magnification);
title('Noisy image (HIGH noise)');
colorbar

figure;
imshow(quad_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Quadratic prior)');
colorbar

figure;
imshow(huber_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Huber prior)');
colorbar

figure;
imshow(log_denoised,[mn, mx], 'InitialMagnification', magnification);
title('Denoised image (Log prior)');
colorbar
```


Noiseless image
1
0.8
0.6
0.4
0.2
0


Noisy image (HIGH noise)
1
0.8
0.6
0.4
0.2
0


Denoised image (Quadratic prior)
1
0.8
0.6
0.4
0.2
0


Denoised image (Huber prior)
1
0.8
0.6
0.4
0.2
0



Published with MATLAB® R2019a