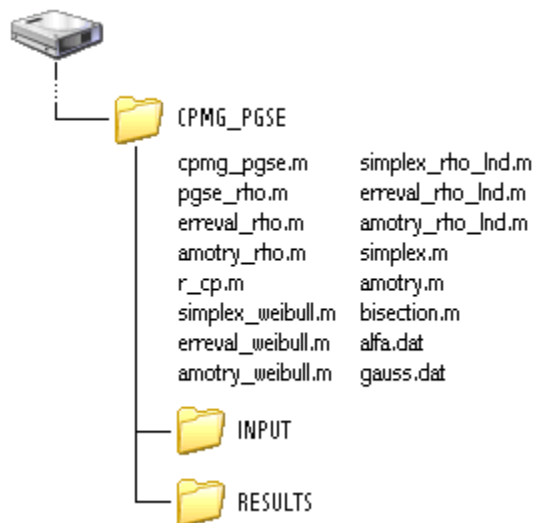


## Appendix D

# MATLAB codes used to process NMR CPMG/PGSE data

### D.1. FILES AND INSTALLATION

Original software was developed to process NMR CPMG/PGSE data. It consists of 14 Matlab (.m) files and two data (.dat) files. The .m files were developed in Matlab, Student Version, Release 12 (The MathWorks, Inc.). Figure D.1 shows the names of these files and the way they must be stored in a memory device so they can be executed.



**Figure D.1.** Files required to process NMR CPMG/PGSE data with Matlab and tree organization of files and sub-directories, as required for execution.

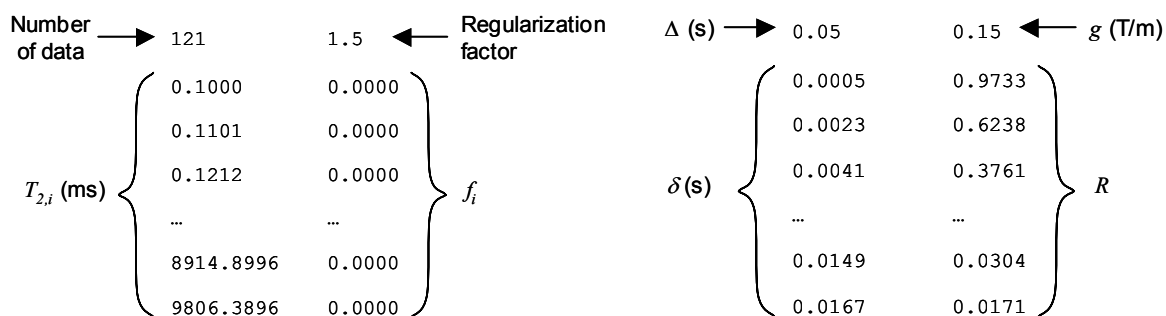
### D.2. EXECUTION AND STRUCTURE OF INPUT FILES

To execute the code, all files must be copied in the same directory (arbitrarily named CPMG\_PGSE in Fig. D.1). Also, two subdirectories must be created: INPUT and RESULTS. Source CPMG and PGSE datafiles must be stored in the directory INPUT. Reports summarizing results are automatically stored in the directory RESULTS.

The main file of the code is *cpmg\_pgse.m*. The rest of the files are accessed as subroutines of such file. The code is executed by simply typing *cpmg\_pgse* in the prompt symbol (>>) of Matlab as shown later.

It is important to clarify how the CPMG and PGSE data must be organized in the input files to successfully run the code. Figure D.2 shows the arrangement that must be observed in the source files for CPMG data. The first datum corresponds to the number  $m$  of  $T_{2,i}f_i$  values that will be considered ( $1 \leq i \leq m$ ). The second datum in the first line is the regularization factor used to invert transverse relaxation data into  $T_2$  distribution, according to the method described by Huang [1]. The code described in this Appendix does not use the regularization factor when processing the data, so an arbitrary number can be assigned for this datum if such information is unknown. The remaining elements in the first column are the  $T_{2,i}$  values (in ms) and the remaining data in the second column are the corresponding  $f_i$  values. This code works for water/oil mixtures exhibiting two separate peaks in the  $T_2$  distribution, with the peak at short  $T_2$  values corresponding to the oil signal and the peak at long  $T_2$  values corresponding to the water signal.

Figure D.3 indicates how the PGSE data must be organized in the corresponding source file. The first datum corresponds to the diffusion time  $\Delta$ , in seconds. The second datum in the first line is the strength of the magnetic field gradient  $g$ , in T/m. The remaining elements of the first column are the chosen durations of the pulsed gradient ( $\delta$ , in s) and the remaining data in the second column are the corresponding attenuation ratios  $R$ . The code is valid for diffusion measurements performed following the sequence shown in Figure 3.2, with  $\tau = \Delta$ . Once the data are organized as indicated in Figures D.2 and D.3, they must be saved in the directory INPUT (see Figure D.1), preferably in text (.txt) or data (.dat) files.



**Figure D.2.** Schematic representation of the ordering required for input files containing CPMG data.

**Figure D.3.** Schematic representation of the ordering required for input files containing PGSE data.

### D.3. ALGORITHM

Figure D.4 shows the algorithm of the code that was developed to process CPMG/PGSE data. The flowchart is self-explanatory, and the nomenclature is fully consistent with the one adopted for Chapter 3.

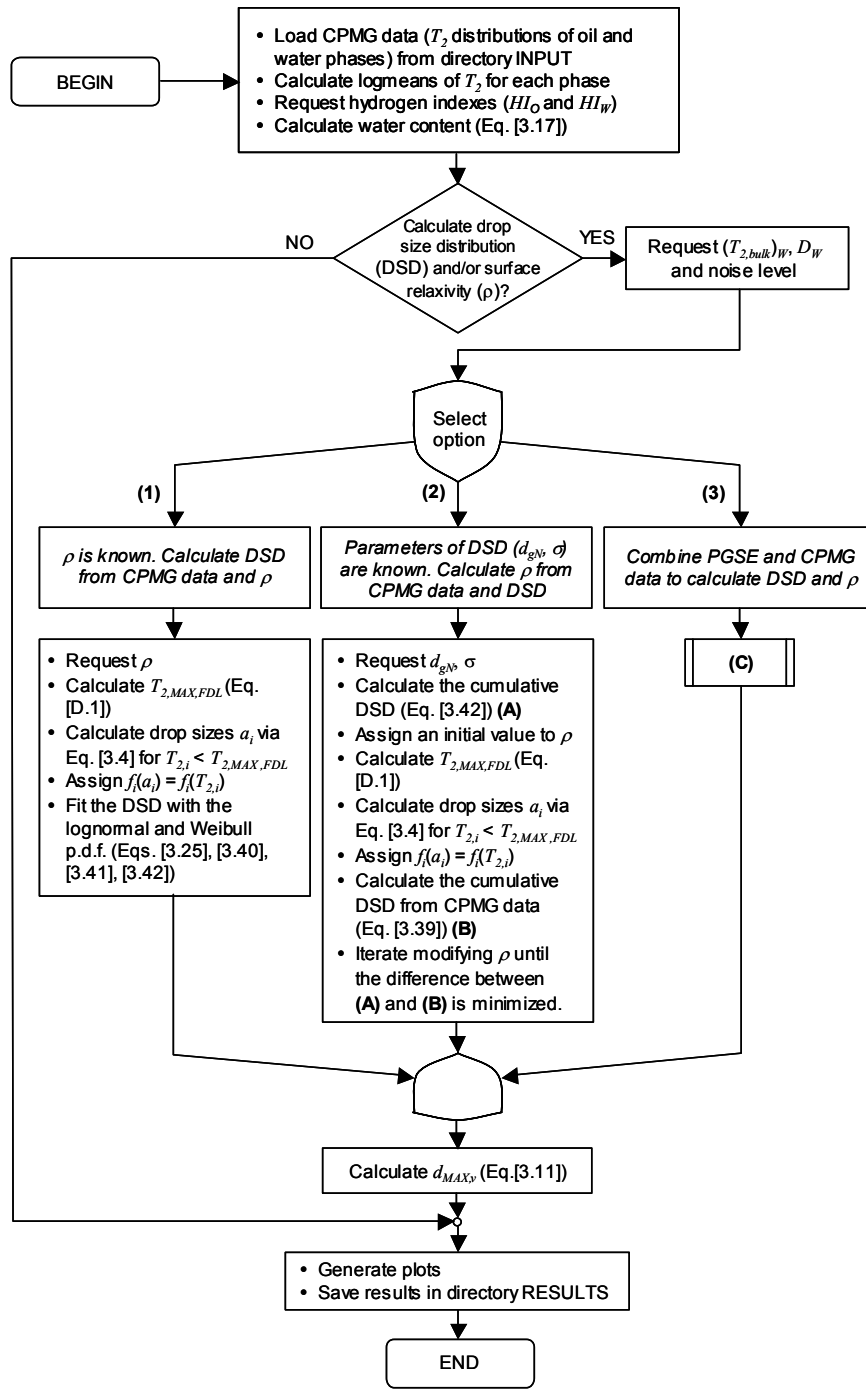


Figure D.4. Flowchart summarizing the algorithm of the Matlab code.

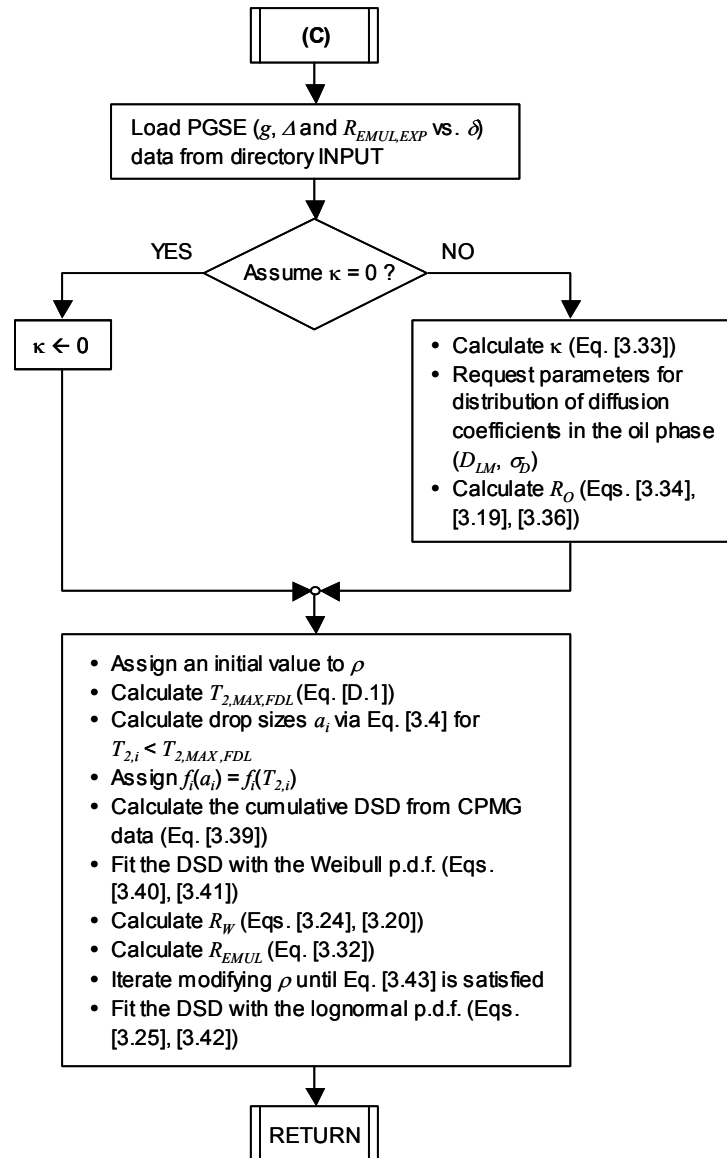


Figure D.4 (cont.) Flowchart summarizing the algorithm of the Matlab codes.

#### D.4. EXAMPLES

In this section we illustrate the usage of the Matlab codes with two sets of synthetic data. These data were created superimposing artificial  $T_2$  distributions for the water phase to the  $T_2$  distribution of the crude oil MP6 shown in Figure 3.3. The  $T_2$  distributions for the water phase were generated with  $\phi_W = 0.30$ ,  $(T_{2,bulk})_W = 2.8$  s in all cases. The properties of the oil phase are listed in Appendix B.

#### D.4.1. Example 1: Water and oil in contact as bulk fluids, not emulsified

Figure D.5 shows the key to process CPMG data from a mixture of oil and water when the fluids are in contact, i.e., not emulsified, as it would appear in the Matlab screen. The file *cpmg\_demo1.dat* was previously stored in the directory INPUT. The texts shown highlighted and in bold characters are the input information provided by the user. The symbol (↵) indicates that the text in the following line would appear in the screen at the end of the line where such symbol is inserted. The code generates the plot shown in Figure D.6 and the file *res\_demo1.dat* shown in Figure D.7, which is stored in the directory RESULTS.

```

EDU>> cpmg_pgse

Characterization of Emulsions via NMR-CPMG/PGSE
*****

Name of file (with extension) where CPMG data are saved ? : cpmg_demo1.dat
File where results will be stored [ Default (press Enter) = results.dat ] ? : ↵
res_demo1.dat

Regularization factor (alpha) = 1

Pre-processing
-----

Please check following results with Figure 1

Number of peaks = 2

Peak 1 : From T2 = 3.481 ms to T2 = 464.2 ms.
        This peak has 2 maxima at T2 = 9.085, and T2 = 195.7 ms.

Peak 2 : From T2 = 2134.1561 ms to T2 = 3780.7877 ms.
        This peak has 1 maximum at T2 = 2840.5618 ms.

Oil Signal   : Peak 1. T2 average = 90.5577 ms.
Water signal : Peak 2. T2 average = 2800.0027 ms.

Parameters
-----

Hydrogen Index - Oil phase   [Default (press Enter) = 1.0] ? : 0.984
Hydrogen Index - Aqueous phase [Default (press Enter) = 1.0] ? : 1

Water content (vol.%) = 30.0011

Calculate (1) Average properties of the phases as bulk fluids
          (2) Drop size distribution {DSD} and/or surface relaxivity
          [ Default (press Enter) = 2 ] ? : 1

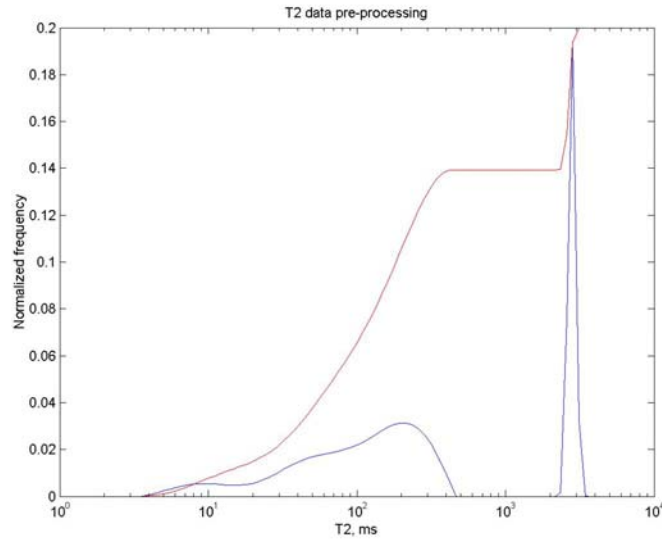
Results
-----

Oil phase   : T2 bulk = 90.5577 ms.
Water phase : T2 bulk = 2800.0027 ms.

Results are stored in file res_demo1.dat

```

**Figure D.5.** Key to run the Matlab code for Example 1.



**Figure D.6.** Plot generated by the code for the  $T_2$  distribution (blue) and the corresponding cumulative curve (red) for Example 1.

```

Analysis CPMG data
-----
Input file: cpmg_demo1.dat
Regularization factor (alpha) = 1

Preprocessing
-----
Number of peaks = 2

Peak 1 : From T2 = 3.481 ms to T2 = 464.2 ms.
        This peak has 2 maxima at T2 = 9.085, and T2 = 195.7 ms.

Peak 2 : From T2 = 2134.1561 ms to T2 = 3780.7877 ms.
        This peak has 1 maximum at T2 = 2840.5618 ms.

Oil Signal   : Peak 1. T2 average = 90.5577 ms.
Water signal : Peak 2. T2 average = 2800.0027 ms.

Parameters
-----
Hydrogen Index - Oil phase      = 0.984
Hydrogen Index - Aqueous phase = 1

Requested calculation : T2 bulk

Results
-----
Oil phase   : T2 bulk = 90.5577 ms.
Water phase : T2 bulk = 2800.0027 ms.
Water content (vol.%) = 30.0011

```

**Figure D.7.** Report generated for the code for Example 1.

If the user attempts to calculate a drop size distribution from the dataset of this example, the code generates the following message:

```
MORE THAN 10 % OF THE SIGNAL OF THE WATER PHASE IS ORIGINATED FROM BULK WATER
AND/OR FROM DROPLETS WITH SIZES LARGER THAN THE MAXIMUM SIZE FOR WHICH THE
FAST DIFFUSION APPROXIMATION IS VALID. IT IS STRONGLY SUGGESTED TO REMOVE
BULK WATER FROM THE SAMPLE AND PERFORM NEW CPMG/PGSE TESTS IN THE REMAINING
EMULSION, SO THE DROP SIZE DISTRIBUTION CAN BE DETERMINED.
```

In general, this message appears whenever the  $T_2$  values below  $T_{2,max,FDL}$  account for 10% or less of the cumulative  $T_2$  distribution of the water phase.  $T_{2,max,FDL}$  is given by:

$$T_{2,max,FDL} = \left[ \frac{1}{(T_{2,bulk})_W} + \rho \frac{6}{d_{MAX,FDL}} \right]^{-1} \quad [D.1]$$

and  $d_{MAX,FDL}$  is given by Eq. [3.7].

#### D.4.2. Example 2: Emulsion of small droplets

This example aims to illustrate the performance of the code when the assumption of fast diffusion is valid for all droplets. In this case, the  $T_2$  distribution of the water phase was generated assuming that the emulsion exhibits a lognormal distribution of sizes with  $d_{gV} = 15 \mu\text{m}$  and  $\sigma = 0.30$ . Also, surface relaxation  $\rho = 0.46 \mu\text{m/s}$  was assumed to generate the synthetic data. The synthetic data is noise-free, but a noise level of 0.5 % was input to the code to illustrate results from Eq. [3.11] for  $d_{MAX,v}$ .

Figure D.8 shows the key to run this second example, as it would appear in the Matlab screen. Figures D.9, D.10 and D.11 show the plots generated by the code for the  $T_2$  distribution, the attenuation profile and the volume-weighted drop size distribution, respectively. The report for this test is omitted for the sake of brevity, but in any case, the same results are also displayed in Figure D.8.

It is seen in Figure D.8 that the code correctly calculated  $d_{gV}$ ,  $\sigma$  and  $\rho$ . Figures D.10 and D.11 show that excellent fits to the synthetic data were achieved. Interestingly, figure D.11 also shows that the bimodal Weibull distribution overlaps the lognormal distribution when a suitable set of parameters for the latter is chosen. Therefore, emulsions exhibiting lognormality in drop sizes can be considered as particular cases of those for which the bimodal Weibull distribution is valid.

```

EDU>> cpmg_pgse

Characterization of Emulsions via NMR-CPMG/PGSE
*****

Name of file (with extension) where CPMG data are saved ? : cpmg_demo2.dat
File where results will be stored [ Default (press Enter) = results.dat ] ? : res_demo2.dat

Regularization factor (alpha) = 1

Pre-processing
-----

Please check following results with Figure 1

Number of peaks = 2

Peak 1 : From T2 = 3.481 ms to T2 = 464.2 ms.
        This peak has 2 maxima at T2 = 9.085, and T2 = 195.7 ms.

Peak 2 : From T2 = 1095.1595 ms to T2 = 2840.5618 ms.
        This peak has 1 maximum at T2 = 1940.1419 ms.

Oil Signal   : Peak 1. T2 average = 90.5577 ms.
Water signal : Peak 2. T2 average = 1918.9377 ms.

Parameters
-----

Hydrogen Index - Oil phase   [Default (press Enter) = 1.0] ? : 0.984
Hydrogen Index - Aqueous phase [Default (press Enter) = 1.0] ? : 1

Water content (vol.%) = 29.9988

Calculate (1) Average properties of the phases as bulk fluids
          (2) Drop size distribution {DSD} and/or surface relaxivity
          [ Default (press Enter) = 2 ] ? : 2

T2 bulk (ms) [ Default (press Enter) = 2800 ] ? : 2800

Diffusion coefficient water (m2/s) [Default (press Enter) = 2.3E-9] ? : 2.3e-9

Noise level (%) [Default (press Enter) = 0 for unknown] ? : 0.5

Please indicate your choice:
-----
(1) Surface relaxivity is known. Calculate DSD.
(2) Parameters of DSD are known. Calculate surface relaxivity
(3) PGSE data are available. Calculate surface relaxivity and DSD
    [ Default (press Enter) = 3 ] ? : 3

Name of file (with extension) where PGSE data are saved ? : pgse_demo2.dat

Time between gradient pulses (s) = 0.05

Strength of magnetic field gradient (T/m) = 0.15

Fractional contribution of continuous phase to R in PGSE tests (kappa):
-----
Calculate kappa from (1) T2 distribution emulsion
                   (2) Data for pure components
                   (3) Assume kappa = 0
                   [ Default (press Enter) = 1 ] ? : 1

kappa = 0.48027

```

**Figure D.8.** Key to run the Matlab code for Example 2.

```

Mean Diffusion coef. continuous phase (m2/s) [Default (press Enter)= 1E-10] ? : 1.78e-10
Standard deviation Diff. coef. continuous phase [Default (press Enter)= 0] ? : 0.27

Iteration # 2
.
.
Iteration # 30

Results
-----
Log-normal distribution fit of cumulative probability P(d), d = drop diameter :
P(d) =  $\int_0^d p(x) dx$ ;  $p(x) = \{1/[(2\pi)^{.5}\sigma x]\} \exp\{-[\ln(x)-\ln(dgV)]^2/(2\sigma^2)\}$ 
dgV (micron) = 14.9682 [ dgN (micron) = 11.4254; d32 (micron) = 13.0774 ]
sigma = 0.30005

Water content (WC, vol.%) = 29.9988

Water in droplets (d =< d_max_FDL) (% of WC) = 100
Bulk water (% of WC) = 0

Surface relaxivity (micron/s) = 0.45869

Contribution of oil phase to R (kappa) = 0.48027

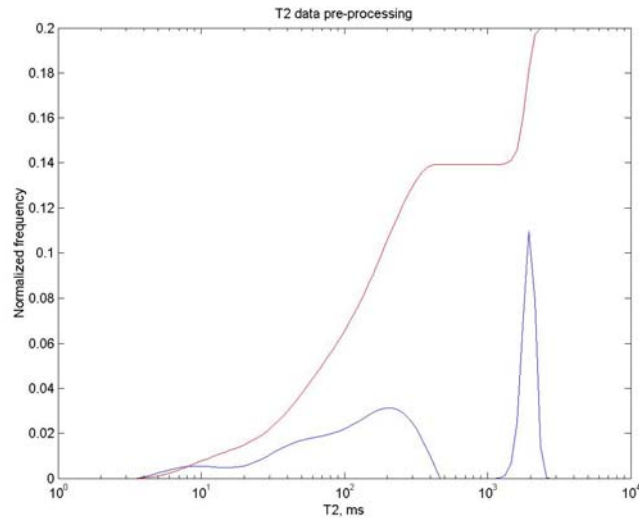
Cutoff drop diameter for fast diffusion limit (d_max_FDL, microns) = 2507.133
Maximum drop size dictated by T2,bulk and SNR (d_max_SNR, microns) = 188.0473

Parameters of the bimodal Weibull distribution
-----
sigma1 = 1.0473
sigma2 = 1.2447
m1 = 5.9157
m2 = 4.1505
omega = 0.1522
a0 (microns) = 2.4751

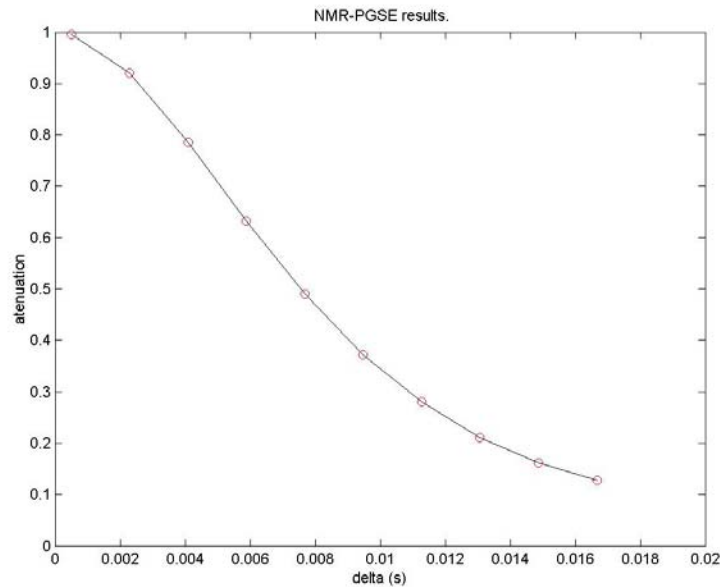
Results are stored in file res_demo2.dat

```

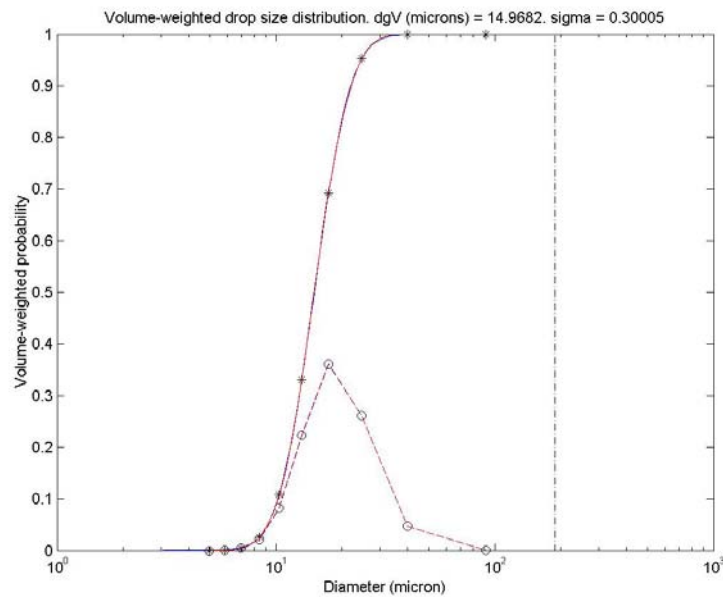
**Figure D.8. (cont.)** Key to run the Matlab code for Example 2.



**Figure D.9.** Plot generated by the code for the  $T_2$  distribution (blue) and the corresponding cumulative curve (red) for Example 2.



**Figure D.10.** Plot generated by the code for the attenuation profile for Example 2. Circles, synthetic data; solid line, best fit to the data using the model reported in Chapter 3.



**Figure D.11.** Plot generated by the code for the volume-weighted drop size distribution for Example 2. Circles, synthetic data from CPMG; stars, cumulative distribution for the synthetic data; red dashed line, best fit using the bimodal Weibull distribution; blue dashed line, best fit using the lognormal distribution; red solid line; best fit to the cumulative distribution using the bimodal Weibull distribution; blue solid line; best fit to the cumulative distribution using the lognormal distribution; dash-dot line,  $d_{MAX}$ , as calculated from Eq. [3.12].

Figure D.11 also shows the position of  $d_{MAX} = d_{MAX,v}$  relative to the drop size distribution. Since all drop sizes are below  $d_{MAX}$ , the NMR-based distribution can be considered fully representative of the actual drop size distribution of the tested emulsion, as was demonstrated in Chapter 3 when comparing NMR and microphotography data.

#### *D.4.3. Example 3: Emulsion containing a small volume of bulk water*

This example aims to illustrate the case in which coalescence has taken place in the emulsion to a significant extent and in consequence, very large drops are and/or a small amount of decanted bulk water is in contact with the emulsion. In these cases, the  $T_2$  distribution typically shows  $T_2$  values beyond  $T_{2MAX,FDL}$  and even beyond  $T_{2,bulk}$  in some cases, due to the inability of the regularization methods to resolve for the  $T_2$  distribution of bulk water with a single  $T_2$  value, but instead with a narrow distribution of  $T_2$  values around  $T_{2,bulk}$ . When  $T_{2,i} \rightarrow T_{2,bulk}$ , Eq. [3.4] predicts that  $a_i \rightarrow \infty$ . For this reason, the results that are generated by the code in these cases should be interpreted as an approximate estimate for the drop size distribution of the drops that remain dispersed in the emulsion. The ability of NMR-based methods to accurately resolve for the drop size distribution in these cases can be improved, for example, by separating the bulk water from the emulsion sample before performing the tests.

The  $T_2$  distribution of this example was generated by shifting the water peak of the former example to higher relaxation times. The distribution is resolved assuming that the surface relaxation  $\rho = 0.46 \mu\text{m/s}$  is known a priori, and that the noise level is 0.5 % as before.

Figure D.12 summarizes the key to run this example, as it would appear in the Matlab window. Results of the analysis are also summarized in this Figure. The program estimates that 95.3 % of the water phase is dispersed as droplet with sizes below  $d_{MAX,FDL} = 2500 \mu\text{m}$  (also below  $d_{MAX} = d_{MAX,v} = 189 \mu\text{m}$ , as shown in Figure D.14), whereas 4.7 % can be considered bulk water. Figure D.13 shows (solid lines) the  $T_2$  distribution for this example and Figure D.14 the corresponding drop size distribution. The cumulative distribution shown in Figure D.14 is truncated to the last drop size for which the fast-diffusion approximation is valid.

The Matlab code allows reprocessing the CPMG data to discard the signal from droplet with sizes greater than  $d_{MAX,FDL}$  and correlate only sizes of drops for which the

```

EDU>> cpmg_pgse

Characterization of Emulsions via NMR-CPMG/PGSE
*****

Name of file (with extension) where CPMG data are saved ? : cpmg_demo3.dat
File where results will be stored [ Default (press Enter) = results.dat ] ? : ↵
res_demo3.dat

Regularization factor (alpha) = 1

Pre-processing
-----
Please check following results with Figure 1

Number of peaks = 2

Peak 1 : From T2 = 3.481 ms to T2 = 464.2 ms.
        This peak has 2 maxima at T2 = 9.085, and T2 = 195.7 ms.

Peak 2 : From T2 = 1325.143 ms to T2 = 3437.0797 ms.
        This peak has 1 maximum at T2 = 2347.5717 ms.

Oil Signal   : Peak 1. T2 average = 90.5577 ms.
Water signal : Peak 2. T2 average = 2321.9137 ms.

Parameters
-----

Hydrogen Index - Oil phase   [Default (press Enter) = 1.0] ? : 0.984
Hydrogen Index - Aqueous phase [Default (press Enter) = 1.0] ? : 1

Water content (vol.%) = 29.9988

Calculate (1) Average properties of the phases as bulk fluids
          (2) Drop size distribution {DSD} and/or surface relaxivity
          [ Default (press Enter) = 2 ] ? : 2

T2 bulk (ms) [ Default (press Enter) = 2800 ] ? : 2800

Diffusion coefficient water (m2/s) [Default (press Enter) = 2.3E-9] ? : 2.3E-9

Noise level (%) [Default (press Enter) = 0 for unknown] ? : 0.5

Please indicate your choice:
-----
(1) Surface relaxivity is known. Calculate DSD.
(2) Parameters of DSD are known. Calculate surface relaxivity
(3) PGSE data are available. Calculate surface relaxivity and DSD
    [ Default (press Enter) = 3 ] ? : 1

Surface relaxivity (micron/s) ? : 0.46

IMPORTANT: T2 distribution surpassed T2_max. Excess computed as bulk water.

4.7277 % of the signal from the water phase is originated in bulk
and/or in droplets with sizes larger than the maximum size for which the
fast diffusion limit is valid. Would you like to discard the signal from
such droplets and reprocess the CPMG data, taking into account the signal
from drops with size d < d_MAX,FDL only ?

(1) YES      (2) NO      [Defaults (press Enter) = 1] ? : 1

```

**Figure D.12.** Key to run the Matlab code for Example 3.

```

File where modified CPMG data will be stored
[ Default (press Enter) = reprocess.dat ] ? : reprocess.dat

Results
-----
Log-normal distribution fit of cumulative probability P(d), d = drop diameter :
P(d) =  $\int_0^d p(x) dx$ ;  $p(x) = \{1/[(2\pi)^{.5}\sigma x]\} \exp\{-[\ln(x)-\ln(dgV)]^2/(2\sigma^2)\}$ 
dgV (micron) = 31.2151 [ dgN (micron) = 15.3084; d32 (micron) = 21.8599 ]
sigma = 0.48734

Water content (WC, vol.%) = 29.9988

Water in droplets (d =< d_max_FDL) (% of WC) = 95.2723
Bulk water (% of WC) = 4.7277

Surface relaxivity (micron/s) = 0.46

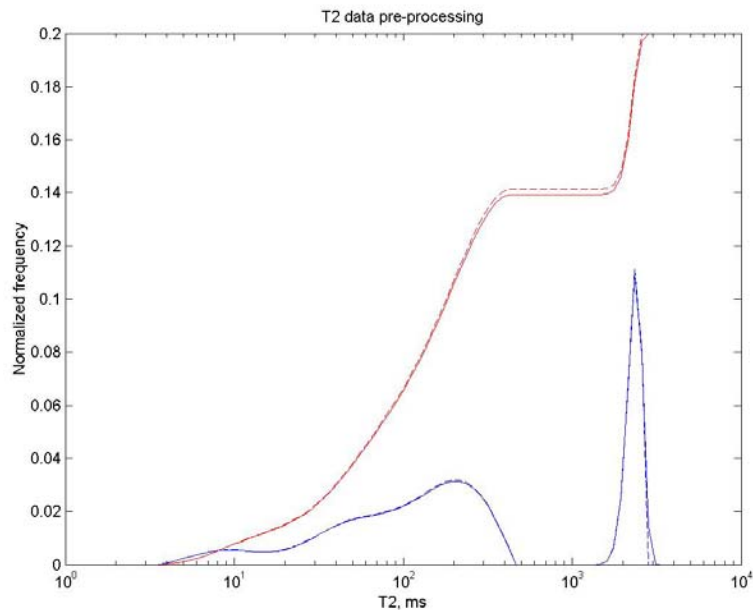
Cutoff drop diameter for fast diffusion limit (d_max_FDL, microns) = 2500
Maximum drop size dictated by T2,bulk and SNR (d_max_SNR, microns) = 188.5838

Parameters of the bimodal Weibull distribution
-----
sigma1 = 2.1827 sigma2 = 1.4303
m1 = 4.2096 m2 = 4.0617
omega = 0.36338 a0 (microns) = 3.4718

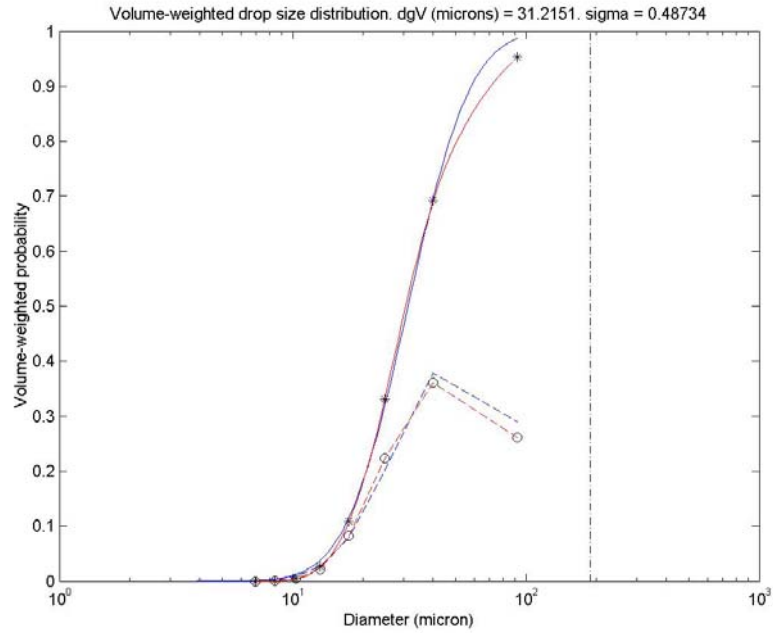
Results are stored in file res_demo3.dat
Data for reprocessing are stored in file reprocess.dat

```

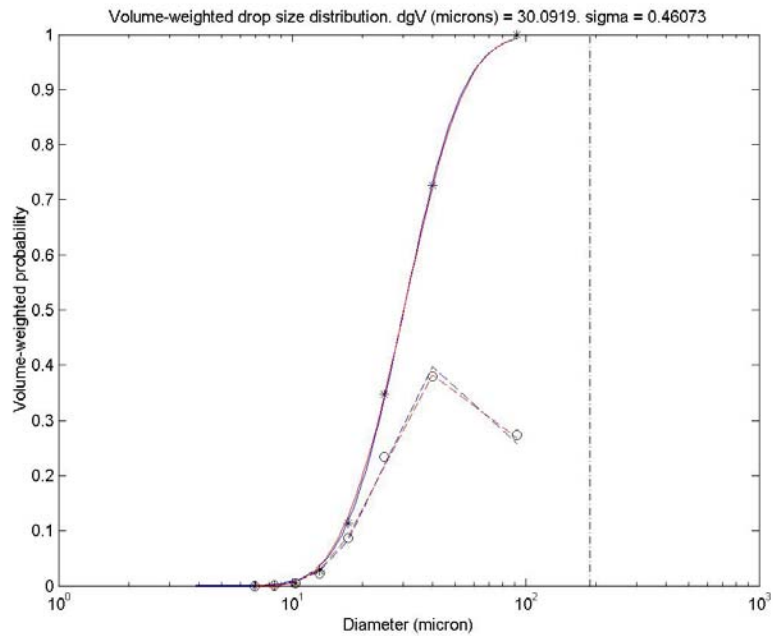
**Figure D.12 (cont.)** Key to run the Matlab code for Example 3.



**Figure D.13.** Plot generated by the code for the  $T_2$  distribution (blue) and the corresponding cumulative curve (red) for Example 3. Solid lines, original dataset; dashed lines, edited CPMG dataset after discarding the contribution of bulk water.



**Figure D.14.** Plot generated by the code for the volume-weighted drop size distribution for Example 3, without reprocessing. The description of the features of Figure D.9 given in the caption of such also applies in this case.



**Figure D.15.** Plot generated by the code for the volume-weighted drop size distribution for Example 3, after reprocessing.

fast-diffusion approximation is valid. It is seen in Figure D.12 that the user is given this option, and if desired, a new file is created with the edited CPMG data. An estimate of the drop size distribution of drops with  $d < d_{MAX,FDL}$  is obtained by re-running the code as shown in the Example 1 and specifying the edited file as the source of CPMG data.

The reprocessed  $T_2$  distribution is shown in dashed lines in Figure D.13. Figure D.15 shows the corresponding drop size distribution, which differ slightly from the one shown in Figure D.14 as might be expected. The water content that is calculated from the reprocessed dataset is slightly below 30 vol.%, i.e.,

$$29.0 \text{ vol.\%} = \frac{30(1 - 4.7/100)}{70 + 30(1 - 4.7/100)} \times 100,$$

since the contribution of bulk water to the  $T_2$  distribution has been discarded.

## D.5. MATLAB CODES

The contents of the files listed in Figure D.1 are provided below. A brief description of the operations performed by such file is provided in each case.

### *File cpmg\_pgse.m*

```

% -----
%   Characterization of O/W Emulsions via NMR-CPMG/PGSE - File CPMG_PGSE.m
% -----
%   Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% This software is provided pursuant to a license agreement containing restrictions
% on its disclosure, duplication and use. This software contains confidential and
% proprietary information, and may not be extracted or distributed, in whole or in
% part, for any purpose whatsoever, without the express written permission of the
% authors. This notice, and the associated author list, must be attached to all
% copies, or extracts, of this software. Any additional restrictions set forth in
% the license agreement also apply to this software.
%
% File CPMG_PGSE.m: Main program for processing of CPMG/PGSE data.
%
% References:
% * Equations for CPMG/PGSE data processing are presented in:
%   Pena AA and Hirasaki GJ. Enhanced Characterization of Oilfield Emulsions via
%   NMR Diffusion and Transverse Relaxation Experiments. Advances in Colloids and
%   Interface Science. In press.
% * This program uses the Simplex method to carry on least square minimization.
%   The code for Simplex method has been adapted from:
%   Press et. al. Numerical recipes in FORTRAN 77. Cambridge Univ. Press (1992)
% -----

```

```

clear data erro P psum Ptry x erro2 P2 psum2 Ptry2 x2 T2_water_old;
clear cumul calc_cumul diam freq freq_oil freq_water T2 T2_oil T2_water tempmax;
clear prob calc_prob_lnd calc_prob_bwd peak_max tempmax ini_peak end_peak pos_max;
clear calc_cumul_lnd calc_cumul_bwd cumul_water cumul_water_old freq_water_old;
clear T2_temp freq_temp cumul_temp;
close all;
minval = 1E-4;
kappa = 0;
flag_end = 0;
flag_reprocess = 0;
resp_rep = 0;

disp(' ');
disp('Characterization of Emulsions via NMR-CPMG/PGSE');
disp('*****');
disp(' ');

% Pre-processing of T2 distribution

nombre = input(' Name of file (with extension) where CPMG data are saved ? : ','s');
cd input;
data = load(nombre);
cd ..;
temp = ' File where results will be stored [ Default (press Enter) = results.dat ] ? : ';
nombre2 = input(temp,'s');
if isempty(nombre2)==1
    nombre2 = 'results.dat';
end
cd results;
fid = fopen(nombre2,'w');
disp(' ');
fprintf(fid,'Analysis CPMG data \n');
fprintf(fid,'----- \n');
fprintf(fid,'\n');
fprintf(fid,' Input file: %s \n',nombre);

% Regularization parameter (alpha)
alpha = data(1,2);
temp = [' Regularization factor (alpha) = ',num2str(alpha)];
disp(temp);
fprintf(fid,'%s \n',temp);
disp(' ');
fprintf(fid,'\n');

T2 = data(2:max(size(data)),1);
freq = data (2:max(size(data)),2);
npeaks = 0;
nmax = 0;
flag = 0;
flag2 = 0;
for i=1:(max(size(T2)));
    if freq(i) < minval
        freq(i) = 0;
    end
    cumul(i)=sum(freq(1:i));
    if ((i==1)+(i==max(size(T2))))== 0; % i <> 1 and i <> max(size(T2))
        if ((freq(i) < minval) + (freq(i+1) >= minval)) == 2 ;
            npeaks = npeaks + 1;
            ini_peak(npeaks) = i;
            flag = 1;
        end
        if ((freq(i)>= freq(i-1)) + (flag == 1)) == 2;
            flag2 = 0;
            pos_max_temp = i;
        else
            if ((flag == 1) + (flag2 == 0)) == 2;
                nmax = nmax + 1;
                pos_max(nmax) = pos_max_temp;
                peak_max(nmax) = npeaks;
            end
        end
    end
end

```

```

        flag2 = 1;
    end
end
end
if ((freq(i) < minval) + (freq(i-1) >= minval) + (flag == 1)) == 3;
    end_peak(npeaks) = i;
    flag = 0;
end
end
end
freq = freq./max(cumul);
cumul = cumul./max(cumul);
if max(freq)>1
    ymax = (fix(max(freq)/10)+1)*10;
else
    ymax = (fix(max(freq)*10)+1)/10;
end
xmin = 10^(fix(log10(T2(ini_peak(1)))-1));
xmax = 10^(fix(log10(T2(end_peak(npeaks))+1)));
figure(1);
semilogx(T2,freq,'-b');
hold on;
semilogx(T2,cumul.*ymax,'-r');
axis([xmin xmax 0 ymax]);
xlabel('T2, ms');
ylabel('Normalized frequency');
title('T2 data pre-processing');
disp('Pre-processing');
disp('-----');
disp(' ');
disp(' Please check following results with Figure 1 ');
disp(' ');
temp = [' Number of peaks = ',num2str(npeaks)];
disp(temp);
disp(' ');
fprintf(fid,'Preprocessing \n');
fprintf(fid,'----- \n');
fprintf(fid,' Number of peaks = %d \n',npeaks);
fprintf(fid,'\n');

for i = 1:npeaks
    temp = [' Peak ',num2str(i),' : From T2 = ',num2str(T2(ini_peak(i))),' ms to T2 = '
    ',num2str(T2(end_peak(i))),' ms.'];
    disp(temp);
    fprintf(fid,'%s \n',temp);
    temp2 = 0;
    for j = 1:nmax
        if peak_max(j) == i
            temp2 = temp2 + 1;
            tempmax(temp2) = pos_max(j);
        end
    end
    if temp2 == 1
        temp = [' This peak has 1 maximum at T2 = ',num2str(T2(tempmax(1))),' ms.'];
        disp(temp);
        fprintf(fid,'%s \n',temp);
        disp(' ');
        fprintf(fid,'\n');
    else
        temp = [' This peak has ',num2str(temp2),' maxima at'];
        for j = 1:(temp2-1)
            temp = [temp,' T2 = ',num2str(T2(tempmax(j))),','];
        end
        temp = [temp,' and T2 = ',num2str(T2(tempmax(temp2))),' ms.'];
        disp(temp);
        fprintf(fid,'%s \n',temp);
        disp(' ');
        fprintf(fid,'\n');
    end
end
end
end

```

```

% Average T2 value - oil peak

T2_oil = T2(ini_peak(1):end_peak(1));
freq_oil = freq(ini_peak(1):end_peak(1));
freq_oil = freq_oil / sum(freq_oil);
temp = size(T2_oil);
if temp(1) == 1
    T2_oil = T2_oil';
end
temp = size(freq_oil);
if temp(1) == 1
    freq_oil = freq_oil';
end
T2_avg_o = 10^(freq_oil'*log10(T2_oil));
temp = [' Oil Signal      : Peak 1. T2 average = ',num2str(T2_avg_o),' ms.'];
disp(temp);
fprintf(fid,'%s \n',temp);

% Average T2 value - water peak

T2_water = T2(ini_peak(npeaks):end_peak(npeaks));
freq_water = freq(ini_peak(npeaks):end_peak(npeaks));
cumul_water = cumul(ini_peak(npeaks):end_peak(npeaks))-cumul(ini_peak(npeaks));
cumul_water = cumul_water./max(cumul_water);
freq_water = freq_water / sum(freq_water);
temp = size(T2_water);
if temp(1) == 1
    T2_water = T2_water';
end
temp = size(freq_water);
if temp(1) == 1
    freq_water = freq_water';
end
temp = size(cumul_water);
if temp(1) == 1
    cumul_water = cumul_water';
end
fid2 = fopen('waterpeak.dat','w');

for i = 1 : max(size(T2_water))
    fprintf(fid2,' %6.4e %6.4e \n ',T2_water(i), freq_water(i));
end
fclose(fid2);

clear data;
data = load('waterpeak.dat');
P0 = [1 1 1 1 .5];
cd ..;
Simplex_weibull;          % Routine to fit T2_water data to the bimodal Weibull ↵
distribution
omega_old = omega;
sigma1_old = sigma1;
sigma2_old = sigma2;
m1_old = m1;
m2_old = m2;
cd results;

T2_avg_w = 10^(freq_water'*log10(T2_water));
temp = [' Water signal : Peak ',num2str(npeaks),'. T2 average = ',num2str(T2_avg_w),' ↵
ms.'];
disp(temp);
fprintf(fid,'%s \n',temp);
disp(' ');
fprintf(fid,'\n');

T2_min = T2_water(1);

% PROCESSING

```

```

disp('Parameters');
disp('-----');
disp(' ');
fprintf(fid,'Parameters \n');
fprintf(fid,'----- \n');

% Water cut

Area_w = (cumul(end_peak(npeaks))-cumul(ini_peak(npeaks)));
Area_o = cumul(ini_peak(npeaks));
HI_o = input(' Hydrogen Index - Oil phase      [Default (press Enter) = 1.0] ? : ');
if isempty(HI_o)==1
    HI_o = 1;
end
HI_w = input(' Hydrogen Index - Aqueous phase [Default (press Enter) = 1.0] ? : ');
if isempty(HI_w)==1
    HI_w = 1;
end
Water_cut = (Area_w/HI_w)/(Area_w/HI_w + Area_o/HI_o)*100;
temp = [' Hydrogen Index - Oil phase      = ',num2str(HI_o)];
fprintf(fid,'%s \n',temp);
temp = [' Hydrogen Index - Aqueous phase = ',num2str(HI_w)];
fprintf(fid,'%s \n',temp);
disp(' ');
fprintf(fid,'\n',temp);
disp([' Water content (vol.%) = ' num2str(Water_cut)]);
disp(' ');
temp = [' Calculate (1) Average properties of the phases as bulk fluids      ';
        ' (2) Drop size distribution {DSD} and/or surface relaxivity '];
disp(temp);
ans = input('                [ Default (press Enter) = 2 ]                ? : ');
if isempty(ans)==1
    ans = 2;
end

if ans == 1 % Calculates T2 bulk as the weighted average from each peak

    fprintf(fid,' Requested calculation : T2 bulk \n');
    fprintf(fid,'\n');
    disp(' ');
    temp = [' Results';' -----'];
    disp(temp)
    disp(' ');
    fprintf(fid,' Results\n',temp(1));
    fprintf(fid,' -----\n');
    fprintf(fid,'\n');
    temp = [' Oil phase      : T2 bulk = ',num2str(T2_avg_o),' ms.'];
    disp(temp);
    fprintf(fid,'%s \n',temp);
    temp = [' Water phase : T2 bulk = ',num2str(T2_avg_w),' ms.'];
    disp(temp);
    fprintf(fid,'%s \n',temp);
    temp = [' Water content (vol.%) = ',num2str(Water_cut)];
    fprintf(fid,'%s \n',temp);

else % Calculates drop size distribution

    fprintf(fid,' Requested calculation : Drop size distribution \n');
    fprintf(fid,'\n');
    disp(' ');
    T2_bulk = input(' T2 bulk (ms) [ Default (press Enter) = 2800 ] ? : ');
    if isempty(T2_bulk)==1
        T2_bulk = 2800;
    end
    temp = [' T2 bulk (ms) = ',num2str(T2_bulk)];
    fprintf(fid,'%s \n',temp);
    disp(' ');
    DC = input(' Diffusion coefficient water (m2/s) [Default (press Enter) = 2.3E-9] ? : ');

```

```

: ');
if isempty(DC)==1
    DC = 2.3E-9;
end
temp = [' Diffusion coefficient water phase (m2/s) = ',num2str(DC)];
fprintf(fid,'%s \n',temp);
disp(' ');
SNR = input(' Noise level (%) [Default (press Enter) = 0 for unknown] ? : ');
if isempty(SNR)==1
    SNR = 0;
end
temp = [' Noise level (%) = ',num2str(SNR)];
fprintf(fid,'%s \n',temp);
disp(' ');
if SNR > 0
    SNR = (100-SNR)/SNR;
end
temp = [' Please indicate your choice:                                     ';
' -----                                                                    ';
' (1) Surface relaxivity is known. Calculate DSD.                            ';
' (2) Parameters of DSD are known. Calculate surface relaxivity              ';
' (3) PGSE data are available. Calculate surface relaxivity and DSD'];
disp(temp);
ans = input('          [ Default (press Enter) = 3 ]                               ? : ');
if isempty(ans)==1
    ans = 3;
end

if ans == 1 % Calculates parameters dgN and sigma
    disp(' ');
    rho = input(' Surface relaxivity (micron/s) ? : ');
    disp(' ');
    T2_max = 1/(1/T2_bulk + 12*(rho*1e-6)^2/(DC*1000));

    for i = 1:max(size(T2_water))

        if T2_water(i) >= T2_max
            temp = ' IMPORTANT: T2 distribution surpassed T2_max. Excess computed as bulk water.';
            flag_reprocess = 1;
            disp(temp);
            disp(' ');
            fprintf(fid,'%s \n',temp);
            fprintf(fid,'\n');
            clear tempx tempy tempz T2_water_old cumul_water_old freq_water_old;
            T2_water_old = T2_water;
            cumul_water_old = cumul_water;
            freq_water_old = freq_water;
            tempx = T2_water(1:i-1);
            tempy = cumul_water(1:i-1);
            tempz = freq_water(1:i-1);
            clear T2_water cumul_water freq_water;
            T2_water = tempx;
            cumul_water = tempy;
            freq_water = tempz;

            if ((cumul_water(i-1) < (1 - eps)) + (cumul_water(i-1) > 0.9)) == 2

                temp1 = [ ' ' num2str(100-max(cumul_water)*100)...
                    ' % of the signal from the water phase is originated in bulk'];
                temp2 = ' and/or in droplets with sizes larger than the maximum size for which the ';
                temp3 = ' fast diffusion limit is valid. Would you like to discard the signal from ';
                temp4 = ' such droplets and reprocess the CPMG data, taking into account the signal ';
                temp5 = ' from drops with size d < d_MAX,FDL only ? ';

                disp(temp1);

```

```

disp(temp2);
disp(temp3);
disp(temp4);
disp(temp5);
disp(' ');

resp_rep = ...
input(' (1) YES      (2) NO      [Defaults (press Enter) = 1] ?  ↵
: ');
if isempty(resp_rep) == 1
    resp_rep = 1;
end

if resp_rep == 1
    disp(' ')
    temp = ' File where modified CPMG data will be stored ';
    disp(temp);
    nombre3 = input(' [ Default (press Enter) = reprocess.dat ] ? : ','s');
    if isempty(nombre3)==1
        nombre3 = 'reprocess.dat';
    end
    disp(' ');
end
end

break;
end

end
if max(cumul_water) < 0.9
    flag_end = 1;
else
    diam = 6E-3*rho./(1./T2_water - 1/T2_bulk);
    temp = size(diam);
    if temp(1) == 1
        diam = diam';
    end
    data = [diam cumul_water];
    cd ..;
    bisection;
    simplex;
    temp = size(data);
    temp2 = size(freq_water);
    if temp(1) ~= temp2(1)
        freq_water = freq_water'
    end
    data = [diam./2 freq_water];
    P0 = [t1 t2 t3 t4 t5];
    Simplex weibull;
    cd results;
end

else
if ans == 3
    cd ..;
    t1 = 1;
    t2 = 1;
    t3 = 1;
    t4 = 1;
    t5 = 0.1;
    PGSE_rho;
    if max(T2_water) >= T2_max
        T2_water_old = T2_water;
        cumul_water_old = cumul_water;
        freq_water_old = freq_water;
        T2_water = T2_temp;
        cumul_water = cumul_temp;
        freq_water = freq_temp;
    end
end

```

```

    if max(cumul_water) < 0.5
        flag_end = 1;
    else
        clear data;
        diam = 6E-3*rho./(1./T2_water - 1/T2_bulk);
        temp = size(diam);
        if temp(1) == 1
            diam = diam';
        end
        data = [diam cumul_water];
        bisection;
        simplex;
    end

else
    disp(' ');
    dgN = input(' Number-based geometric mean diameter (dgN, microns) ? : ');
    sigma = input(' Geometric standard deviation (sigma) ? : ');
    disp(' ');
    fprintf(fid,'\n');
    cd ..;
    simplex_rho_lnd;
    if max(T2_water) >= T2_max
        T2_water_old = T2_water;
        cumul_water_old = cumul_water;
        freq_water_old = freq_water;
        T2_water = T2_temp;
        cumul_water = cumul_temp;
        freq_water = freq_temp;
    end
    if max(cumul_water) < 0.5
        flag_end = 1;
    else
        clear data;
        diam = 6E-3*rho./(1./T2_water - 1/T2_bulk);
        temp = size(diam);
        if temp(1) == 1
            diam = diam';
        end
        data = [diam./2 freq_water];
        P0 = [t1 t2 t3 t4 t5];
        Simplex_weibull;
    end
end
cd results;
end

diam_max = DC*1E12/(2*rho);
if SNR > 0
    diam_max_SNR = 2*rho*SNR*(T2_bulk/1000)/exp(1);
end

if flag_end == 0

    if kappa > 0
        temp = [' Logmean diffusion coefficient oil phase (m2/s) = ' num2str(DC_CP)];
        fprintf(fid,'%s \n',temp);
        temp = [' St. dev. diffusion coeffs. oil phase = ' num2str(SD_CP)];
        fprintf(fid,'%s \n',temp);
    end
    fprintf(fid,'\n');
    temp = [' Results';' -----'];
    disp(temp)
    disp(' ');
    fprintf(fid,'\n');
    fprintf(fid,' Results \n',temp(1));
    fprintf(fid,' ----- \n',temp(1));
    fprintf(fid,'\n');

```

```

figure(4)
calc_prob_lnd(1) = 0;
calc_prob_bwd(1) = 0;
for i = 2:max(size(diam))
    xtempi = log(diam(i)/diam(1));
    xtempi_1 = log(diam(i-1)/diam(1));
    calc_prob_lnd(i) = 0.5*(erf(log(diam(i)/dgV)/(2^.5*sigma))- erf(log(diam(i-1)/dgV)/(2^.5*sigma)));
    calc_prob_bwd(i) = (omega*(exp(-1*(xtempi_1/sigma1)^m1)-exp(-1*(xtempi/sigma1)^m1)) + ...
        (1-omega)*(exp(-1*(xtempi_1/sigma2)^m2)-exp(-1*(xtempi/sigma2)^m2)));
end
semilogx(diam,freq_water,'ok');
hold on;
semilogx(diam,calc_prob_lnd,'--b');
semilogx(diam,calc_prob_bwd,'--r');
semilogx([diam_max diam_max],[0 1],'-g');
if SNR > 0
    semilogx([diam_max_SNR diam_max_SNR],[0 1],'-k');
end
ymax = 1;
clear u_lnd u_bwd T2_temp calc_cumul_lnd calc_cumul_bwd diam_temp;
temp1 = 0.7*min(log10(diam));
temp2 = max(log10(diam));
temp3 = (temp2-temp1)/100;
x = temp1:temp3:temp2;
x = 10.^x;
u_lnd = (log(x./dgV))./(2^.5*sigma);
calc_cumul_lnd = .5.*(1 + erf(u_lnd));
T2_temp = log(T2_min):(log(T2_max/T2_min)/100):log(max(T2_water));
T2_temp = exp(T2_temp);
diam_temp = 6E-3*rho./(1./T2_temp - 1/T2_bulk);
u_bwd = log(diam_temp./diam_temp(1));
calc_cumul_bwd = (omega.*(1-exp(-1.*(u_bwd./sigma1).^m1)) + ...
    (1-omega).*(1-exp(-1.*(u_bwd./sigma2).^m2)));
semilogx(diam,cumul_water.*ymax,'k');
semilogx(x,calc_cumul_lnd.*ymax,'-b');
semilogx(diam_temp,calc_cumul_bwd.*ymax,'-r');
xlabel('Diameter (micron)');
ylabel('Volume-weighted probability');
title(['Volume-weighted drop size distribution. dgV (microns) = ',...
    num2str(dgV),'. sigma = ',num2str(sigma)]);
xmin = 10^(fix(log10(diam(1))-1));
if SNR > 0
    xmax = 10^(fix(log10(diam_max_SNR))+1);
else
    xmax = 10^(fix(log10(diam(max(size(diam)))))+1);
end
axis([xmin xmax 0 ymax]);
fprintf(fid,' Volume-weighted drop size distribution \n');
fprintf(fid,' \n');
fprintf(fid,' d (microns) Prob. (vol. frac.) Cumul. Prob \n');
for i = 1:max(size(diam))
    fprintf(fid,' %6.4e %6.4e %6.4e \n',diam(i), freq_water(i),
cumul_water(i));
end
fprintf(fid,' \n');
temp1 = ' Log-normal distribution fit of cumulative probability P(d), d = drop
diameter : ';
temp2 = ' /d';
temp3 = ' P(d)= | p(x)*dx; p(x) = {1/[(2*pi)^.5*sigma*x]}*exp{-[ln(x)-
ln(dgV)]^2/(2*sigma^2)}';
temp4 = ' /0';
disp(temp1);
disp(temp2);
disp(temp3);
disp(temp4);
disp(' ');
fprintf(fid,' %s \n',temp1);

```

```

fprintf(fid,' %s \n',temp2);
fprintf(fid,' %s \n',temp3);
fprintf(fid,' %s \n',temp4);
fprintf(fid,'\n');

dgN = dgV / exp(3*sigma^2);
d32 = dgN * exp(1.5*sigma^2);

temp = [' dgV (micron) = ' num2str(dgV) ' [ dgN (micron) = ' ...
        num2str(dgN) '; d32 (micron) = ' num2str(d32) ' ]'];

disp(temp)
fprintf(fid,' %s \n',temp);
temp = [' sigma          = ' num2str(sigma)];
disp(temp)
disp(' ');
fprintf(fid,' %s \n',temp);
fprintf(fid,'\n');
temp = [' Water content (WC, vol.%)          = ',num2str(Water_cut)];
disp(temp);
fprintf(fid,'%s \n',temp);
disp(' ');
fprintf(fid,'\n');
temp = [' Water in droplets (d =< d_max_FDL) (% of WC) = ' ⚡
        num2str(100*max(cumul_water))];
disp(temp)
fprintf(fid,' %s \n',temp);
temp = [' Bulk water (% of WC)              = ' num2str(100- ⚡
        max(cumul_water)*100)];
disp(temp);
fprintf(fid,' %s \n',temp);
disp(' ');
fprintf(fid,'\n');
temp = [' Surface relaxivity (micron/s)      = ',num2str(rho)];
disp(temp);
fprintf(fid,'%s \n',temp);
disp(' ');
fprintf(fid,'\n');
if kappa > 0
    temp = [' Contribution of oil phase to R (kappa) = ',num2str(kappa)];
    disp(temp);
    fprintf(fid,'%s \n',temp);
    disp(' ');
    fprintf(fid,'\n');
end

temp = [' Cutoff drop diameter for fast diffusion limit (d_max_FDL, microns) = ⚡
        ',...
        num2str(diam_max)];
disp(temp);
fprintf(fid,'%s \n',temp);

if SNR > 0

    temp = [' Maximum drop size dictated by T2,bulk and SNR (d_max_SNR, microns) = ⚡
            ',...
            num2str(diam_max_SNR)];
    disp(temp);
    disp(' ');
    fprintf(fid,'%s \n',temp);
    fprintf(fid,'\n');

end

temp1 = ' Parameters of the bimodal Weibull distribution ';
temp2 = ' ----- ';
temp3 = [' sigmal          = ' num2str(sigmal)];
temp4 = [' sigma2         = ' num2str(sigma2)];
temp5 = [' m1            = ' num2str(m1)];

```

```

temp6 = [' m2          = ' num2str(m2)];
temp7 = [' omega      = ' num2str(omega)];
temp8 = [' a0 (microns) = ' num2str(diam(1)/2)];
disp(' ');
disp(temp1);
disp(temp2);
disp(temp3);
disp(temp4);
disp(temp5);
disp(temp6);
disp(temp7);
disp(temp8);
fprintf(fid, ' %s \n', temp1);
fprintf(fid, ' %s \n', temp2);
fprintf(fid, ' %s \n', temp3);
fprintf(fid, ' %s \n', temp4);
fprintf(fid, ' %s \n', temp5);
fprintf(fid, ' %s \n', temp6);
fprintf(fid, ' %s \n', temp7);
fprintf(fid, ' %s \n', temp8);

else

fprintf(fid, '\n');
temp = [' Water content (WC, vol.%) = ', num2str(Water_cut)];
fprintf(fid, '%s \n', temp);
fprintf(fid, '\n');

temp = [' MORE THAN 10 % OF THE SIGNAL OF THE WATER PHASE IS ORIGINATED FROM BULK WATER
AND/OR FROM DROPLETS WITH SIZES LARGER THAN THE MAXIMUM SIZE FOR WHICH THE
FAST DIFFUSION APPROXIMATION IS VALID. IT IS STRONGLY SUGGESTED TO REMOVE
BULK WATER FROM THE SAMPLE AND PERFORM NEW CPMG/PGSE TESTS IN THE REMAINING
EMULSION, SO THE DROP SIZE DISTRIBUTION CAN BE DETERMINED.'];
disp(' ');
disp(temp);
temp = temp';
fprintf(fid, '%s \n', temp(1:80));
fprintf(fid, '%s \n', temp(81:160));
fprintf(fid, '%s \n', temp(161:240));
fprintf(fid, '%s \n', temp(241:320));
fprintf(fid, '%s \n', temp(321:400));
disp(' ');
fprintf(fid, '\n');
temp = [' Cutoff drop diameter for fast diffusion limit (d_max_FDL, microns) =
', ...
num2str(diam_max)];
disp(temp);
fprintf(fid, '%s \n', temp);

end

end

disp(' ');
disp([' Results are stored in file ' nombre2]);
disp(' ');
cd ...

if ((flag_reprocess == 1) + (resp_rep == 1)) == 2

cd input;
data = load(nombre);
fid3 = fopen(nombre3, 'w');
fprintf(fid3, '%6.4e %6.4e \n ', data(1,1), data(1,2));
for i = 2 : max(size(data))
if data(i,1) > T2_max
fprintf(fid3, '%6.4e %6.4e \n ', data(i,1), 0);
else

```

```

        fprintf(fid3,' %6.4e %6.4e \n ',data(i,1), data(i,2));
    end
end
fclose(fid3);
cd ..;
cd results;
temp = ([ ' Data for reprocessing are stored in file ', nombre3]);
disp(temp);
disp(' ');
fprintf(fid,'\n');
fprintf(fid,' %s \n',temp);
cd ..;

end

fclose(fid);

```

### File *pgse\_rho.m*

```

% -----
%      Characterization of W/O Emulsions via NMR-CPMG/PGSE - File PGSE_rho.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File PGSE_rho.m: Generates attenuation profile for a PGSE experiment using the
%                  drop size distribution that is calculated from the T2 distri-
%                  bution of the water phase for a given surface relaxivity.
% -----

clear alfa2 alfaR data erro Ldelta P psum Ptry R a x y kappa;
disp(' ');
nombre = input(' Name of file (with extension) where PGSE data are saved ? : ','s');
cd input;
data_PGSE = load(nombre);
cd ..;

Udelta = data_PGSE(1,1);
G = data_PGSE(1,2);
disp(' ');
disp([' Time between gradient pulses (s) = ' num2str(Udelta)]);
disp(' ');
disp([' Strength of magnetic field gradient (T/m) = ' num2str(G)]);
disp(' ');
disp(' Fractional contribution of continuous phase to R in PGSE tests (kappa): ');
disp(' ----- ');
disp(' ');
disp([' Calculate kappa from (1) T2 distribution emulsion      ');
      ' (2) Data for pure components      ');
      ' (3) Assume kappa = 0              ']);
Calckappa = input(' [ Default (press Enter) = 1 ] ? : ');
if isempty(Calckappa)==1
    Calckappa = 1;
end
disp(' ');
if Calckappa == 1
    temp = size(freq);
    if temp(1) == 1
        freq = freq';
    end
    temp1 = freq(ini_peak(npeaks):end_peak(npeaks))*exp(-2*(Udelta*1000)./T2_water);
    temp2 = freq(ini_peak(1):end_peak(1))*exp(-2*(Udelta*1000)./T2_oil);
else
    if Calckappa == 2
        temp = [' Water content (vol.%) [ Default (press Enter) = ' num2str(Water_cut) '
                ] : '];
        fw = input(temp);
    end
end

```

```

    if isempty(fwx)==1
        fwx = Water_cut/100;
    else
        fwx = fwx/100;
    end
    temp = [' Mean T2 water phase [ Default (press Enter) = ' num2str(T2_avg_w) ' ms ↵
    ] : '];
    T2wx = input(temp);
    if isempty(T2wx)==1
        T2wx = T2_avg_w;
    end
    temp = [' Mean T2 oil phase [ Default (press Enter) = ' num2str(T2_avg_o) ' ms ↵
    ] : '];
    T2ox = input(temp);
    if isempty(T2ox)==1
        T2ox = T2_avg_o;
    end
    temp1 = fwx*HI_w*exp(-2*(Udelta*1000)/T2wx);
    temp2 = (1-fwx)*HI_o*exp(-2*(Udelta*1000)/T2ox);
end
end
if Calckappa ~= 3
    kappa = 1/(1+temp1/temp2);
    disp(' ');
    disp([' kappa = ' num2str(kappa)]);
    disp(' ');
else
    kappa = 0;
end

gamma = 2.675e8; % Magnetogyric ratio of the proton, rad/s.T

n = 20; % First n roots of the equation u * J(5/2,u) - J(3/2,u) = 0
temp = load('alfa.dat'); % (to be used in erreval_rho.m)
alfaR = temp(1:n); %

temp = load('gauss.dat'); % Gaussian quadrature points
u = [-1*temp(:,1) ; temp(:,1) ];
w_gc = [ temp(:,2) ; temp(:,2) ];

if (kappa > 0)
    R_CP; % Attenuation profile for continuous phase
end

ndim = 1; % Number of variables: 1 (rho)
ftol = 1e-4; % Permitted tolerance (control)
rtol = 1 + ftol; % Tolerance factor (to be calculated)
ITMAX = 150; % Maximum number of iterations

% Definition of the simplex

P0 = -0.6931; % ln0.50

lambda = P0/2;

P = [P0 ; P0 + lambda];

% Evaluation of errors at the corners of the Simplex

for w = 1:(ndim + 1)
    lnrho = P(w);
    erreval_rho;
    erro(w) = sumerr;
end

% Minimization of error

iter = 0;
psum = sum(P,1);

```

```

disp(' ');

while rtol >= ftol    % Beginning of while loop

    ilo = 1;
    if erro(1) > erro(2)
        ihi = 1;
        inhi = 2;
    else
        ihi = 2;
        inhi = 1;
    end
    for i = 1:(ndim + 1)
        if erro(i) <= erro(ilo)
            ilo = i;
        end
        if erro(i) > erro(ihi)
            inhi = ihi;
            ihi = i;
        else
            if erro(i) > erro(inhi)
                if i ~= ihi
                    inhi = i;
                end
            end
        end
    end
    end

    iter = iter + 2;
    disp([' Iteration # ' num2str(iter)]);
    fac = -1;
    amotry_rho;
    if errotry <= erro(ilo)
        fac = 2;
        amotry_rho;
    elseif errotry >= erro(inhi)
        errosave = erro(ihi);
        fac = 0.5;
        amotry_rho;
        if errotry >= errosave
            for w = 1:(ndim + 1)
                if w ~= ilo
                    for j = 1:ndim
                        psum(j) = 0.5*(P(w,j)+P(ilo,j));
                        P(w,j) = psum(j);
                    end
                    lnrho = psum(1);
                    erreval_rho;
                    erro(w) = sumerr;
                end
            end
            iter = iter + ndim;
            psum = sum(P,1);
        end
    else
        iter = iter - 1;
    end
    end
    rtol = 2*abs(erro(ihi)-erro(ilo))/(abs(erro(ihi))+abs(erro(ilo))+eps);
    if iter >= ITMAX
        disp('WARNING: ITMAX for PGSE_rho exceeded');
        rtol
        rtol = 0;
    end
    end
end %end of while loop

disp(' ');
temp = erro(1);
erro(1) = erro(ilo);
erro(ilo) = temp;

```

```

for i = 1:ndim
    temp = P(1,i);
    P(1,i) = P(i1o,i);
    P(i1o,i) = temp;
end

rho = exp(P(1));

% Averaging and plotting results

figure(3);
plot(data_PGSE(2:max(size(data_PGSE)),1),data_PGSE(2:max(size(data_PGSE)),2),'or');
hold on;
plot(data_PGSE(2:max(size(data_PGSE)),1),R,'-k');
xlabel('delta (s)');
ylabel('atenuation');
temp = 10^fix(log10(max(data_PGSE(2:max(size(data_PGSE)),1))) - 1);
xmax = (fix(max(data_PGSE(2:max(size(data_PGSE)),1))/temp) + 1)*temp;
axis([0 xmax 0 1]);
title([' NMR-PGSE results.']);

```

### File *r\_cp.m*

```

% -----
% Characterization of W/O Emulsions via NMR-CPMG/PGSE - R_CP.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File R_CP.m: Calculates the PGSE attenuation profile for the continuous phase
% of the emulsion. It is assumed that the continuous phase exhibits
% a lognormal distribution of diffusion coefficients.
% -----

clear Diff temp1 temp2 R_CPh DC_CP SD_CP

DC_CP = input(...
' Mean Diffusion coef. continuous phase (m2/s) [Default (press Enter) = 1E-10] ? : ');
if isempty(DC_CP)==1
    DC_CP = 1E-10;
end
SD_CP = input(...
' Standard deviation Diff. coef. continuous phase [Default (press Enter) = 0] ? : ');
if isempty(SD_CP)==1
    SD_CP = 0;
end
SD_CP = SD_CP + eps;

% Correction for restricted diffusion in continuous phase

eta = 1/(1+ Water_cut/200 + (1/4) * (Water_cut/74.02)^9);

% Attenuation profile. Integration is carried

R_CPh = zeros(1,(max(size(data_PGSE))-1));

for k = 1:(max(size(data_PGSE))-1)
    Ldelta = data_PGSE(k + 1, 1);
    for i = 1:max(size(u))
        Diff = eta*DC_CP*((1+u(i))/(1-u(i)))^SD_CP;
        temp1 = exp(-1*gamma^2*Ldelta^2*G^2*(Udelta-Ldelta/3)*Diff);
        temp2 = 2*temp1*exp(-1*(log((Diff/(eta*DC_CP))))^2/(2*SD_CP^2))/((8*atan(1))...
            ^.5*(1-u(i)^2));
        R_CPh(k) = R_CPh(k) + temp2 * w_gc(i);
    end
end
end

```

**File erreval\_rho.m**

```

% -----
% Characterization of O/W Emulsions via NMR-CPMG/PGSE - File erreval_rho.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File erreval_rho.m: Calculates the error in the fit of the PGSE data. Numerical
% integration of the theoretical expression for attenuation
% profile is performed via Gaussian quadrature (24 points)
% -----

clear T2_temp cumul_temp freq_temp xr x_exp y z a a_exp R;
rho = exp(lnrho);
d_min = 6E-9*rho./(1./T2_min - 1/T2_bulk);
d_max = DC/(2*rho*1e-6);
T2_max = 1/(1/T2_bulk + rho*6E-9/d_max);

if max(T2_water) >= T2_max
    for i = 1:max(size(T2_water))
        if T2_water(i) >= T2_max
            clear temp_x temp_y temp_z;
            temp_x = T2_water(1:i-1);
            temp_y = cumul_water(1:i-1);
            temp_z = freq_water(1:i-1);
            clear freq_temp T2_temp cumul_temp;
            T2_temp = temp_x;
            cumul_temp = temp_y;
            freq_temp = temp_z;
            break;
        end
    end
else
    T2_temp = T2_water;
    cumul_temp = cumul_water;
    freq_temp = freq_water;
end

a_exp = 3E-9*rho./(1./T2_temp-1/T2_bulk);
temp = size(a_exp);
if temp(1) == 1
    a_exp = a_exp';
end
data = [a_exp freq_temp];
P0 = [t1 t2 t3 t4 t5];
Simplex_weibull;
ndim = 1;

if 2*max(a_exp) > d_max
    xmax = log(d_max/d_min); % xmax is ksi' in theoretical development
else
    xmax = log(2*max(a_exp)/d_min);
end

xr = (xmax/2).*(u + 1); % Transforming Gaussian points into drop radii a
a = (d_min/2).*exp(xr);

sumerr = 0;

for k = 1:(max(size(data_PGSE)-1))

    Numerator = 0;
    Denominator = 0;
    ldelta = data_PGSE(k+1,1); % Duration of gradient, s (from experimental data)
    for i = 1:max(size(a))
        clear alfa2;
        alfa2 = (alfaR / a(i)).^2;
        alfa2 = alfa2 + eps;
    end
end

```

```

coef = 1./(alfa2.*(alfaR.^2 - 2));
suma = 0;
for j = 2:n
    temp = 2 + exp(-1*alfa2(j)*DC*(Udelta - Ldelta))-2*exp(-1*alfa2(j)*DC*Udelta);
    temp = temp - 2*exp(-1*alfa2(j)*DC*Ldelta) + exp(-1*alfa2(j)*DC*(Udelta + Ldelta));
    temp = 2*Ldelta/(alfa2(j)*DC)- temp/(alfa2(j)*DC)^2;
    suma = suma + coef(j)*temp;
end
y(i) = exp(-2*G^2*gamma^2*suma);
z(i) = ((omega*m1/sigma1^m1)*xr(i)^(m1-1)*exp(-1*(xr(i)/sigma1)^m1)+ ...
        ((1-omega)*m2/sigma2^m2)*xr(i)^(m2-1)*exp(-1*(xr(i)/sigma2)^m2));
y(i) = y(i) * z(i);
y(i) = y(i) * xmax / 2 ;
z(i) = z(i) * xmax / 2 ;
Numerator = Numerator + y(i)*w_gc(i);
Denominator = Denominator + z(i)*w_gc(i);
end

R(k) = Numerator/Denominator;
if kappa > 0
    R(k) = R(k) * (1 - kappa) + R_CPh(k) * kappa;
end
sumerr = sumerr + (data_PGSE(k+1,2) - R(k))^2;

end

lastrho = rho;
figure(5);
hold on;
plot(data_PGSE(2:max(size(data_PGSE)),1),data_PGSE(2:max(size(data_PGSE)),2),'or');
hold on;
plot(data_PGSE(2:max(size(data_PGSE)),1),R,'-g');
xlabel('delta (s)');
ylabel('atenuation');
temp = 10^fix(log10(max(data_PGSE(2:max(size(data_PGSE)),1))) - 1);
xmax = (fix(max(data_PGSE(2:max(size(data_PGSE)),1))/temp) + 1)*temp;
axis([0 xmax 0 1]);
title([' NMR-PGSE results.']);

```

### File amotry\_rho.m

```

% -----
%           Characterization of O/W Emulsions via NMR-CPMG/PGSE - amotry_rho.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File amotry_rho.m: Extrapolates by a factor fac through the face of the Simplex
%                   across from the high point, tries it and replaces the high
%                   point if the new point is better.
% -----

fac1 = (1-fac)/ndim;
fac2 = fac1 - fac;
for j = 1:ndim
    Ptry(j) = psum(j)*fac1-P(ihi,j)*fac2;
end
lnrho = Ptry(1);
erreval_rho;
errotry = sumerr;
if errotry < erro(ihi)
    erro(ihi) = errotry;
    for j = 1:ndim
        psum(j) = psum(j) - P(ihi,j) + Ptry(j);
        P(ihi,j) = Ptry(j);
    end
end
end

```

*File simplex\_weibull.m*

```

% -----
% Characterization of O/W Emulsions via NMR-CPMG/PGSE - File Simplex_weibull.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File Simplex_weibull.m: Applies the downhill Simplex method to determine the
% numerical values of the parameters of the bimodal
% Weibull distribution that render the best fit of the
% drop size distribution calculated from CPMG data
% with a given surface relaxivity.
% -----

clear x2 y yobs erro2 P2 psum2;

x2 = log(data(:,1)./data(1,1));

ndim2 = 5;           % Number of variables: dgN and sigma
ftol2 = 1e-8;       % Permitted tolerance (control)
rtol2 = 1 + ftol2;  % Tolerance factor (to be calculated)
ITMAX2 = 100000;    % Maximum number of iterations

% Definition of the simplex

%P0 = [1 1 1 1 .5];

lambda = P0/2;
e1 = [1 0 0 0 0];
e2 = [0 1 0 0 0];
e3 = [0 0 1 0 0];
e4 = [0 0 0 1 0];
e5 = [0 0 0 0 1];

P2 = [P0 ; P0 + lambda(1)*e1; P0 + lambda(2)*e2; P0 + lambda(3)*e3; P0 + lambda(4)*e4;
P0 + lambda(5)*e5];

% Evaluation of errors at the corners of the Simplex

for w2 = 1:(ndim2 + 1)
    t1 = P2(w2,1);
    t2 = P2(w2,2);
    t3 = P2(w2,3);
    t4 = P2(w2,4);
    t5 = P2(w2,5);
    erreval_weibull;
    erro2(w2) = sumerr2;
end

% Minimization of error
iter2 = 0;
psum2 = sum(P2,1);

while rtol2 >= ftol2    %beginning of while loop

ilo2 = 1;
if erro2(1) > erro2(2)
    ihi2 = 1;
    inhi2 = 2;
else
    ihi2 = 2;
    inhi2 = 1;
end
for i = 1:(ndim2 + 1)
    if erro2(i) <= erro2(ilo2)
        ilo2 = i;
    end
end

```

```

    if erro2(i) > erro2(ihi2)
        inhi2 = ihi2;
        ihi2 = i;
    else
        if erro2(i) > erro2(inhi2)
            if i ~= ihi2
                inhi2 = i;
            end
        end
    end
end
end

if iter2 >= ITMAX2
    ans = 'ITMAX in Simplex_Weibull.m exceeded'
end

iter2 = iter2 + 2;
fac2 = -1;
amotry_weibull;
if errotry2 <= erro2(ilo2)
    fac2 = 2;
    amotry_weibull;
elseif errotry2 >= erro2(inhi2)
    errosave2 = erro2(ihi2);
    fac2 = 0.5;
    amotry_weibull;
    if errotry2 >= errosave2
        for w2 = 1:(ndim2 + 1)
            if w2 ~= ilo2
                for j = 1:ndim2
                    psum2(j) = 0.5*(P2(w2,j)+P2(ilo2,j));
                    P2(w2,j) = psum2(j);
                end
                t1 = P2(w2,1);
                t2 = P2(w2,2);
                t3 = P2(w2,3);
                t4 = P2(w2,4);
                t5 = P2(w2,5);
                erreval_weibull;
                erro2(w2) = sumerr2;
            end
        end
        iter2 = iter2 + ndim2;
        psum2 = sum(P2,1);
    end
else
    iter2 = iter2 - 1;
end
rto12 = 2*abs(erro2(ihi2)-erro2(ilo2))/(abs(erro2(ihi2))+abs(erro2(ilo2))+eps);

end %end of while loop

temp = erro2(1);
erro2(1) = erro2(ilo2);
erro2(ilo2) = temp;
for i = 1:ndim2
    temp = P2(1,i);
    P2(1,i) = P2(ilo2,i);
    P2(ilo2,i) = temp;
end

% Averaging and plotting results
t1 = P2(1,1);
t2 = P2(1,2);
t3 = P2(1,3);
t4 = P2(1,4);
t5 = P2(1,5);

%Plot drop size distribution

```

```

sigma1 = t1^2;
sigma2 = t2^2;
m1 =     t3^2;
m2 =     t4^2;
omega = 1/(1+exp(-1*t5));

erreval_weibull;
y(1) = cum(1);
yobs(1) = data(1,2);
for i = 2:max(size(x2));
    yobs(i) = data(i,2);
end

```

### File erreval\_weibull.m

```

% -----
%   Characterization of Emulsions via NMR-CPMG/PGSE - File erreval_weibull.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File erreval_weibull.m: Calculates the error in the fit of the drop size distri-
%                          bution data with the bimodal Weibull p.d.f.
% -----

clear cum;
sumerr2 = 0;
sigma1 = t1^2;
sigma2 = t2^2;
m1 =     t3^2;
m2 =     t4^2;
omega = 1/(1+exp(-1*t5));
for k = 1:max(size(x2))
    cum(k) = omega*(1-exp(-1*(x2(k)/sigma1)^m1)) + (1-omega)*(1-exp(-
    1*(x2(k)/sigma2)^m2));
    if k == 1
        y(k) = cum(k);
    else
        y(k) = cum(k) - cum(k-1);
    end
    sumerr2 = sumerr2 + (data(k,2) - y(k))^2;
end

```

### File amotry\_weibull.m

```

% -----
%   Characterization of O/W Emulsions via NMR-CPMG/PGSE - amotry_weibull.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File amotry_weibull.m: Extrapolates by a factor fac2 through the face of the
%                          Simplex across from the high point, tries it and
%                          replaces the high point if the new point is better.
% -----

fac12 = (1-fac2)/ndim2;
fac22 = fac12 - fac2;
for j = 1:ndim2
    Ptry2(j) = psum2(j)*fac12-P2(ihi2,j)*fac22;
end
t1 = Ptry2(1);
t2 = Ptry2(2);
t3 = Ptry2(3);

```

```

t4 = Ptry2(4);
t5 = Ptry2(5);
erreval_weibull;
errotry2 = sumerr2;
if errotry2 < erro2(ihi2)
    erro2(ihi2) = errotry2;
    for j = 1:ndim2
        psum2(j) = psum2(j) - P2(ihi2,j) + Ptry2(j);
        P2(ihi2,j) = Ptry2(j);
    end
end
end

```

### File *simplex\_rho\_lnd.m*

```

% -----
% Characterization of W/O Emulsions via NMR-CPMG/PGSE - simplex_rho_lnd.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File simplex_rho_lnd.m: Applies downhill Simplex method to determine surface
% relaxivity when the parameters of the drop size distri-
% bution are known
% -----

clear u_lnd;
dgV = dgN*exp(3*sigma^2);

ndim = 1;           % Number of variables: 1 (rho)
ftol = 1e-8;       % Permitted tolerance (control)
rtol = 1 + ftol;   % Tolerance factor (to be calculated)
ITMAX = 150;       % Maximum number of iterations

% Definition of the simplex

P0 = -0.6931; % ln(0.5)

lambda = P0/2;

P = [P0 ; P0 + lambda];

% Evaluation of errors at the corners of the Simplex

for w = 1:(ndim + 1)
    lnrho = P(w);
    erreval_rho_lnd;
    erro(w) = sumerr;
end

% Minimization of error

iter = 0;
psum = sum(P,1);
disp(' ');

while rtol >= ftol %beginning of while loop

    ilo = 1;
    if erro(1) > erro(2)
        ihi = 1;
        inhi = 2;
    else
        ihi = 2;
        inhi = 1;
    end
    end
    for i = 1:(ndim + 1)

```

```

    if erro(i) <= erro(ilo)
        ilo = i;
    end
    if erro(i) > erro(ihi)
        ihi = i;
    end
    else
        if erro(i) > erro(ihi)
            if i ~= ihi
                ihi = i;
            end
        end
    end
end
end

iter = iter + 2;
disp([' Iteration # ' num2str(iter)]);
fac = -1;
amotry_rho_lnd;
if errotry <= erro(ilo)
    fac = 2;
    amotry_rho_lnd;
elseif errotry >= erro(ihi)
    errosave = erro(ihi);
    fac = 0.5;
    amotry_rho_lnd;
    if errotry >= errosave
        for w = 1:(ndim + 1)
            if w ~= ilo
                for j = 1:ndim
                    psum(j) = 0.5*(P(w,j)+P(ilo,j));
                    P(w,j) = psum(j);
                end
                lnrho = psum(1);
                erreval_rho_lnd;
                erro(w) = sumerr;
            end
        end
        iter = iter + ndim;
        psum = sum(P,1);
    end
else
    iter = iter - 1;
end
rtol = 2*abs(erro(ihi)-erro(ilo))/(abs(erro(ihi))+abs(erro(ilo))+eps);
if iter >= ITMAX
    disp('Warning: ITMAX for PGSE_rho exceeded');
    rtol = 0;
end
end %end of while loop

disp(' ');
temp = erro(1);
erro(1) = erro(ilo);
erro(ilo) = temp;
for i = 1:ndim
    temp = P(1,i);
    P(1,i) = P(ilo,i);
    P(ilo,i) = temp;
end

rho = exp(P(1));

```

**File erreval\_rho\_lnd.m**

```

% -----
%   Characterization of W/O Emulsions via NMR-CPMG/PGSE - erreval_rho_lnd.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File erreval_rho_lnd.m: Calculates the error between the cumulative CPMG data for
%                         the water phase and the cumulative lognormal distribution
%                         for a given surface relaxivity
% -----

clear T2_temp cumul_temp freq_temp x a R;
rho = exp(lnrho);
T2_max = 1/(1/T2_bulk + 12*(rho*1e-6)^2/(DC*1000));

T2_temp = T2_water;
cumul_temp = cumul_water;
freq_temp = freq_water;

if max(T2_water) >= T2_max
    for i = 1:max(size(T2_water))
        if T2_water(i) >= T2_max
            clear tempx tempy tempz;
            tempx = T2_water(1:i-1);
            tempy = cumul_water(1:i-1);
            tempz = freq_water(1:i-1);
            clear freq_temp T2_temp cumul_temp;
            T2_temp = tempx;
            cumul_temp = tempy;
            freq_temp = tempz;
            break;
        end
    end
end

sumerr = 0;
clear diam u_lnd cal_cumul;

diam = 6E-3*rho./(1./T2_temp - 1/T2_bulk);
u_lnd = (log(diam./dgV))./(2^.5*sigma);
calc_cumul = .5.*(1 + erf(u_lnd));
sumerr = sum((cumul_temp-calc_cumul).^2);

```

**File amotry\_rho\_lnd.m**

```

% -----
%   Characterization of O/W Emulsions via NMR-CPMG/PGSE - amotry_rho_lnd.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File amotry_rho_lnd.m: Extrapolates by a factor fac through the face of the
%                         Simplex across from the high point, tries it and
%                         replaces the high point if the new point is better.
% -----

fac1 = (1-fac)/ndim;
fac2 = fac1 - fac;
for j = 1:ndim
    Ptry(j) = psum(j)*fac1-P(ihi,j)*fac2;
end
lnrho = Ptry(1);

```

```

erreval_rho_lnd;
errotry = sumerr;
if errotry < erro(ihi)
    erro(ihi) = errotry;
    for j = 1:ndim
        psum(j) = psum(j) - P(ihi,j) + Ptry(j);
        P(ihi,j) = Ptry(j);
    End
end

```

### File simplex.m

```

% -----
% Characterization of O/W Emulsions via NMR-CPMG/PGSE - File Simplex.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File Simplex.m: Applies the downhill Simplex method to determine the
% numerical values of the parameters of the lognormal
% distribution that render the best fit of the
% drop size distribution calculated from CPMG data
% with a given surface relaxivity.
% -----

clear u_lnd;

ndim = 2;           % Number of variables for simplex : dgV and sigma
ftol = 1e-5;       % Permitted tolerance (control)
rtol = 1 + ftol;   % Tolerance factor (to be calculated)
ITMAX = 150;       % Maximum number of iterations

% Definition of the simplex
dgV_aprox = davg;
sigma_aprox = sigma;
P0 = [davg sigma];

lambda = P0/2;
e1 = [1 0];
e2 = [0 1];
if ndim == 1
    e1 = [0 0]
    e2 = e1;
end

P = [P0 ; P0 + lambda(1) * e1 ; P0 + lambda(2) * e2];

% Evaluation of errors at the corners of the Simplex

for z = 1:(ndim + 1)
    dgV = P(z,1);
    sigma = P(z,2);
    u_lnd = (log(data(:,1)./dgV))./(2^.5*sigma);
    calc_cumul = .5.*(1 + erf(u_lnd));
    erro(z) = sum((data(:,2)-calc_cumul).^2);
end

% Minimization of error

iter = 0;
psum = sum(P,1);

while rtol >= ftol % beginning of while loop

    ilo = 1;
    if erro(1) > erro(2)

```

```

    ihi = 1;
    inhi = 2;
else
    ihi = 2;
    inhi = 1;
end
for i = 1:(ndim + 1)
    if erro(i) <= erro(ilo)
        ilo = i;
    end
    if erro(i) > erro(ihi)
        inhi = ihi;
        ihi = i;
    else
        if erro(i) > erro(inhi)
            if i ~= ihi
                inhi = i;
            end
        end
    end
end
end

iter = iter + 2;
fac = -1;
amotry;
if errotry <= erro(ilo)
    fac = 2;
    amotry;
elseif errotry >= erro(inhi)
    errosave = erro(ihi);
    fac = 0.5;
    amotry;
    if errotry >= errosave
        for z = 1:(ndim + 1)
            if z ~= ilo
                for j = 1:ndim
                    psum(j) = 0.5*(P(z,j)+P(ilo,j));
                    P(z,j) = psum(j);
                end
                dgV = psum(1);
                sigma = psum(2);
                u_lnd = (log(data(:,1)./dgV))./(2^.5*sigma);
                calc_cumul = .5.*(1 + erf(u_lnd));
                erro(z) = sum((data(:,2)-calc_cumul).^2);
            end
        end
        iter = iter + ndim;
        psum = sum(P,1);
    end
else
    iter = iter - 1;
end
rtol = 2*abs(erro(ihi)-erro(ilo))/(abs(erro(ihi))+abs(erro(ilo))+eps);
if iter >= ITMAX
    ans = 'ITMAX exceeded'
    rtol
    rtol = 0;
end
end %end of while loop

temp = erro(1);
erro(1) = erro(ilo);
erro(ilo) = temp;
for i = 1:ndim
    temp = P(1,i);
    P(1,i) = P(ilo,i);
    P(ilo,i) = temp;
end

```

**File amotry.m**

```

% -----
%           Characterization of O/W Emulsions via NMR-CPMG/PGSE - amotry.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File amotry.m: Extrapolates by a factor fac through the face of the Simplex
%                across from the high point, tries it and replaces the high point
%                if the new point is better.
%
% -----

fac1 = (1-fac)/ndim;
fac2 = fac1 - fac;
for j = 1:ndim
    Ptry(j) = psum(j)*fac1-P(ihi,j)*fac2;
end
dgV = Ptry(1);
if ndim == 2
    sigma = Ptry(2);
end
u_lnd = (log(data(:,1)./dgV))./(2^.5*sigma);
calc_cumul = .5.*(1 + erf(u_lnd));
errotry = sum(abs(data(:,2)-calc_cumul));
if errotry < erro(ihi)
    erro(ihi) = errotry;
    for j = 1:ndim
        psum(j) = psum(j) - P(ihi,j) + Ptry(j);
        P(ihi,j) = Ptry(j);
    end
end
end

```

**File bisection.m**

```

% -----
%           Characterization of O/W Emulsions via NMR-CPMG/PGSE - File bisection.m
% -----
% Copyright (C) 2001-2003 Alejandro Pena, Clarence Miller and George Hirasaki
%
% File bisection.m: Applies the bisection method to determine the
%                  numerical values of the parameters of the lognormal
%                  distribution that render the best fit of the
%                  drop size distribution calculated from CPMG data
%                  with a given surface relaxivity.
%
% -----

clear u_lnd c;
k = 1;
while data(k,2) <= 0.5
    k = k + 1;
end

tempy = log10(data((k-1:k),1));
tempx = data((k-1:k),2);
c = polyfit(tempx,tempy,1);
temp = c(1)*0.5 + c(2);

davg = 10^temp;

% Calculating sigma {PgV = .5 * [1 + erf(u)]; u = ln(d/dgV)/2^.5*sigma}
% with bisection method

difer = 1;

```

```

sigma = 10;
delt = -0.045 + eps;
u_lnd = (log(data(:,1)./davg))./(2^.5*sigma);
calc_cumul = .5.*(1 + erf(u_lnd));
sumerr = sum((cumul_water-calc_cumul).^2);

while abs(difer) > 1e-8

    sigma = sigma + delt;
    u_lnd = (log(data(:,1)./davg))./(2^.5*sigma);
    calc_cumul = .5.*(1 + erf(u_lnd));
    sumerr_n = sum((data(:,2)-calc_cumul).^2);
    difer = sumerr_n - sumerr;
    if difer > 0
        sigma = sigma - delt;
        delt = delt / 2;
    else
        sumerr = sumerr_n;
    end
end
end

```

**File alfa.dat ( First 20 roots of the equation  $xJ_{5/2}(x) - J_{3/2}(x) = 0$  )**

```

0.00000000
2.08157597
5.94036999
9.20584014
12.40444502
15.57923641
18.74264558
21.89969648
25.05282528
28.20336100
31.35209173
34.49951492
37.64596032
40.79165523
43.93676147
47.08139741
50.22565165
53.36959182
56.51327046
59.65672900
62.80000056

```

**File gauss.dat (values for 24-point Gaussian quadrature)**

```

0.064056892862605626085  0.127938195346752156974
0.191118867473616309159  0.125837456346828296121
0.315042679696163374387  0.121670472927803391204
0.433793507626045138487  0.115505668053725601353
0.545421471388839535658  0.107444270115965634783
0.648093651936975569252  0.097618652104113888270
0.740124191578554346244  0.086190161531953275917
0.820001985973902921954  0.073346481411080305734
0.886415527004401034213  0.059298584915436780746
0.938274552002732758524  0.044277438817419806169
0.974728555971309498198  0.028531388628933663181
0.995187219997021360180  0.012341229799987199547

```

## References

1. C.C. Huang, "Estimation of rock properties by NMR relaxation methods", MSc Thesis, Rice University, Houston, USA, 1997.

1. C.C. Huang, Estimation of rock properties by NMR relaxation methods, MSc Thesis, Rice University, Houston, USA, 1997.