```
> restart;
kernelopts(numcpus=6);
> with(LinearAlgebra):
with(CurveFitting):
with(Optimization):
with(plots):
with(Statistics):
with(ListTools):
with(plottools):
with(ArrayTools):
with(FileTools):
with(StringTools):
with(SignalProcessing):
> #Get the spectral parameters

downsample:=proc(myVector,samplingrate)
local i, returnArray;
returnArray:=[];
for i from 1 to Dimension(myVector) by samplingrate do
returnArray:=[op(returnArray),myVector[i]];
end do;

return Vector[row](nops(returnArray),returnArray);
end proc:

installAcquParameter:=proc(searchDirectory,parameterToSearch,destinationName,p
arameterType)
local filenameForParameters, acquParametersString, stringToSearch, startIndex,
endIndex, parameterString,foundParameter;
description "retrieve the value of the parameter s and store it into the
destination variable";
filenameForParameters:=cat(searchDirectory,"/acqus");
acquParametersString:=FileTools[Text][ReadFile](filenameForParameters):
stringToSearch:=cat("$",parameterToSearch,"= ");
startIndex:=Search(stringToSearch,acquParametersString);
endIndex:=SearchText("\n",acquParametersString,startIndex..startIndex+50);
parameterString:=SubString(acquParametersString,startIndex..startIndex+endInde
x-2);
foundParameter:=sscanf(parameterString,cat(stringToSearch,parameterType))[1];
parse(cat(destinationName,":=",foundParameter),statement);
end proc:


installAcqu2Parameter:=proc(searchDirectory,parameterToSearch,destinationName,
parameterType)
local filenameForParameters, acquParametersString, stringToSearch, startIndex,
endIndex, parameterString,foundParameter;
description "retrieve the value of the parameter s and store it into the
destination variable";
filenameForParameters:=cat(searchDirectory,"/acqu2s");
acquParametersString:=FileTools[Text][ReadFile](filenameForParameters):
stringToSearch:=cat("$",parameterToSearch,"= ");
startIndex:=Search(stringToSearch,acquParametersString);
endIndex:=SearchText("\n",acquParametersString,startIndex..startIndex+10);
parameterString:=SubString(acquParametersString,startIndex..startIndex+endInde
x-2);
foundParameter:=sscanf(parameterString,cat(stringToSearch,parameterType))[1];
```

```
parse(cat(destinationName,":=",foundParameter),statement);
end proc:

installProcParameter:=proc(searchDirectory,parameterToSearch,destinationName,p
arameterType)
local filenameForParameters, procParametersString, stringToSearch, startIndex,
endIndex, parameterString,foundParameter;
description "retrieve the value of the parameter s and store it into the
destination variable";
filenameForParameters:=cat(searchDirectory,"/procs");
procParametersString:=FileTools[Text][ReadFile](filenameForParameters):
stringToSearch:=cat("$",parameterToSearch,"= ");
startIndex:=Search(stringToSearch,procParametersString);
endIndex:=SearchText("\n",procParametersString,startIndex..startIndex+50);
parameterString:=SubString(procParametersString,startIndex..startIndex+endInde
x-2);
foundParameter:=sscanf(parameterString,cat(stringToSearch,parameterType))[1];
parse(cat(destinationName,":=",foundParameter),statement);
end proc:


installProc2Parameter:=proc(searchDirectory,parameterToSearch,destinationName,
parameterType)
local filenameForParameters, procParametersString, stringToSearch, startIndex,
endIndex, parameterString,foundParameter;
description "retrieve the value of the parameter 2s and store it into the
destination variable";
filenameForParameters:=cat(searchDirectory,"/proc2s");
procParametersString:=FileTools[Text][ReadFile](filenameForParameters):
stringToSearch:=cat("$",parameterToSearch,"= ");
startIndex:=Search(stringToSearch,procParametersString);
endIndex:=SearchText("\n",procParametersString,startIndex..startIndex+50);
parameterString:=SubString(procParametersString,startIndex..startIndex+endInde
x-2);
foundParameter:=sscanf(parameterString,cat(stringToSearch,parameterType))[1];
parse(cat(destinationName,":=",foundParameter),statement);
end proc:


createNormalized1DData:=proc(commonDirectory,sampleName,experimentNumber,proce
ssedDataNumber,sampleMass, adjustableFactor,sampling,range_min, range_max)
local directoryForAcqu, directoryForProc, filenameForData,
importedDataFile,myO1p,mySR,windowCenter,windowStep,
normalizedDataMatrix,chemicalshiftvector,chemicalshiftvector2,tempvector,i,sta
rtindex,endindex;

#acq parameters directory
directoryForAcqu:=cat(commonDirectory,sampleName,"/",experimentNumber);

#processed data directory
directoryForProc:=cat(directoryForAcqu,"/pdata/",processedDataNumber);

#filename
filenameForData:=cat(directoryForProc,"/1r");
importedDataFile:=FileTools:-Binary:-ReadFile(filenameForData,'datatype' =
'integer[4]',byteorder=little);
```

```
installAcquParameter(directoryForAcqu,"SW","mySW","%f");
installAcquParameter(directoryForAcqu,"SW_h","mySWH","%f");
installAcquParameter(directoryForAcqu,"O1","myO1","%f");
installAcquParameter(directoryForAcqu,"SFO1","mySFO1","%f");
installAcquParameter(directoryForAcqu,"BF1","myBF1","%f");
installAcquParameter(directoryForAcqu,"RG","myRG","%f");
installAcquParameter(directoryForAcqu,"NS","myNS","%d");

installAcquParameter(directoryForAcqu,"TD","myTD","%d");

installAcquParameter(directoryForAcqu,"NC","my_NC","%d");
installProcParameter(directoryForProc,"SI","mySI","%d");
installProcParameter(directoryForProc,"SF","mySF","%f");

installProcParameter(directoryForProc,"NC_proc","my_NC_PROC","%d");

myO1p:=round(myO1/mySFO1*1000.)/1000.;
mySR:=(mySF-myBF1)*1e6;

windowCenter:=myO1p-mySR/mySFO1;
windowStep:=mySW/(mySI-1);

chemicalshiftvector:=Vector([seq(windowCenter+mySW/2-(i-
1)*windowStep,i=1..mySI)]);

startindex:=0;
endindex:=0;
for  i from 1 to mySI do
if chemicalshiftvector[i]>range_min then
endindex:=endindex+1;
end if;
if chemicalshiftvector[i]>range_max then
startindex:=startindex+1;
end if;
end do;

chemicalshiftvector2:=downsample(chemicalshiftvector[startindex..endindex],sam
pling);

tempvector:=Vector(downsample(Vector[row](endindex-
startindex+1,[seq(importedDataFile[i]*adjustableFactor*2^(my_NC_PROC+my_NC)/(m
yNS*sampleMass*myRG),i=startindex..endindex)]),sampling));
normalizedDataMatrix:=[chemicalshiftvector2,tempvector]:
return normalizedDataMatrix;
end proc:


createNormalized2DTimeData:=proc(commonDirectory,sampleName,experimentNumber,p
rocessedDataNumber,sampleMass, adjustableFactor,sampling,range_min, range_max)
local directoryForAcqu, directoryForProc, filenameForData,
importedDataFile,myO1p,mySR,windowCenter,windowStep,
normalizedDataArray,importedVDList,filenameForVDList,i,j,k,tempArray,tempArray
2,tempMatrix,tempMatrix2,chemicalshiftvector2,chemicalshiftvector,templine,myl
ine,myresult,myfactor,startindex, endindex;


#acq parameters directory
```

```
directoryForAcqu:=cat(commonDirectory,sampleName,"/",experimentNumber);

#processed data directory
directoryForProc:=cat(directoryForAcqu,"/pdata/",processedDataNumber);

#filename
filenameForData:=cat(directoryForProc,"/2rr");

importedDataFile:=FileTools:-Binary:-ReadFile(filenameForData,'datatype' =
'integer[4]',byteorder=little);

installAcquParameter(directoryForAcqu,"SW","mySW","%f");
installAcquParameter(directoryForAcqu,"SW_h","mySWH","%f");
installAcquParameter(directoryForAcqu,"O1","myO1","%f");
installAcquParameter(directoryForAcqu,"SFO1","mySFO1","%f");
installAcquParameter(directoryForAcqu,"BF1","myBF1","%f");
installAcquParameter(directoryForAcqu,"RG","myRG","%f");
installAcquParameter(directoryForAcqu,"NS","myNS","%d");

installAcquParameter(directoryForAcqu,"NC","my_NC","%d");

installProcParameter(directoryForProc,"SI","mySI","%d");

installProc2Parameter(directoryForProc,"SI","my2ndSI","%d");

installAcqu2Parameter(directoryForAcqu,"TD","my2ndTD","%d");

installProcParameter(directoryForProc,"SF","mySF","%f");
installProcParameter(directoryForProc,"NC_proc","my_NC_PROC","%d");

installProcParameter(directoryForProc,"XDIM","XDIMF2","%d");
installProc2Parameter(directoryForProc,"XDIM","XDIMF1","%d");


filenameForVDList:=cat(directoryForAcqu,"/vdlist");

fclose(filenameForVDList);

importedVDList:=Vector(my2ndTD);

for i from 1 to my2ndTD do
myline:=readline(filenameForVDList);
myresult:=sscanf(myline, "%f%c");

if nops(myresult)=1 then
importedVDList[i]:=op(myresult);

elif myresult[2]="u" then
importedVDList[i]:=myresult[1]*1e-6;

elif myresult[2]="m" then
importedVDList[i]:=myresult[1]*1e-3;

elif myresult[2]="s" then
importedVDList[i]:=myresult[1]*1.0;
end if;
end do;
```

```
myO1p:=round(myO1/mySFO1*1000.)/1000.;
mySR:=(mySF-myBF1)*1e6;
windowCenter:=myO1p-mySR/mySFO1;
windowStep:=mySW/(mySI-1);


chemicalshiftvector:=Vector([seq(windowCenter+mySW/2-(i-
1)*windowStep,i=1..mySI)]);

startindex:=0;
endindex:=0;
for  i from 1 to mySI do
if chemicalshiftvector[i]>range_min then
endindex:=endindex+1;
end if;
if chemicalshiftvector[i]>range_max then
startindex:=startindex+1;
end if;
end do;

chemicalshiftvector2:=downsample(chemicalshiftvector[startindex..endindex],sam
pling);



tempArray:=Reshape(importedDataFile,[XDIMF2,XDIMF1,mySI/XDIMF2,my2ndSI/XDIMF1]
);


tempMatrix:=Matrix(my2ndSI,Dimension(chemicalshiftvector2));

for i from 1 to my2ndSI/XDIMF1 by 1 do
for j from 1 to XDIMF1 by 1 do
tempArray2:=Vector[row](tempArray[1..,j,1,i]);

if mySI/XDIMF2 > 1 then
for k from 2 to mySI/XDIMF2 by 1 do
tempArray2:=Vector[row](Concatenate(2, tempArray2, tempArray[1..,j,k,i]));
end do;
end if;

templine:=Vector[row](downsample(Vector[row](tempArray2[startindex..endindex])
,sampling));
tempMatrix(j+(i-1)*XDIMF1,1..):=templine;

end do;
end do;

myfactor:=adjustableFactor*2^(my_NC_PROC+my_NC)/(myNS*sampleMass*myRG);
tempMatrix:=tempMatrix*myfactor;


tempMatrix2:=Matrix(my2ndTD,Dimension(chemicalshiftvector2),tempMatrix(1..my2n
dTD,1..));
```

```
normalizedDataArray:=[importedVDList,chemicalshiftvector2,tempMatrix2]:

return normalizedDataArray;

#return importedVDList;

end proc:

myReflect:=(a)->reflect(a, [[0,0], [0,1]]);

displayRMN:=proc(myPlotList,tickDistance,minTick,maxTick)
display(map(myReflect,myPlotList),tickmarks=[[seq(-
tickDistance*i=convert(tickDistance*i,string),i=round(minTick/tickDistance)..r
ound(maxTick/tickDistance))],default]);
end proc:

> #definition of pre-factors to have an area equal to 1

pref2 := evalf(2*sqrt(ln(2)/Pi));
pref := evalf(4*ln(2));


#definition of peak shapes

Gaussienne := (w, L)-> pref2*exp(-pref*(delta-w)^2/L^2)/abs(L);
Lorentzienne := (w, L)-> 1/(evalf(Pi)*(1+(delta-w)^2/L^2))/abs(L);
int(Lorentzienne(0,2),delta=-infinity..infinity);
int(Gaussienne(0,2),delta=-infinity..infinity);


#definition of build up functions

#simple HOESY transfer
#buildUpHOESY := proc (t, M_0, R12, R11, R22) options operator, arrow;
#M_0*abs(R12)*sinh(((1/2)*(R11-R22)^2+2*R12^2)^(1/2)*t)*exp(-
(1/2)*(R11+R22)*t)/((1/2)*(R11-R22)^2+2*R12^2)^(1/2) end proc;


buildUpHOESY :=(t,R12,R11,R22,M10,M20)->exp((1/2*(-R11-R22+sqrt(R11^2-
2*R11*R22+4*R12^2+R22^2)))*t)*R12*M20/sqrt(R11^2-2*R11*R22+4*R12^2+R22^2)-
exp(-(1/2*(R11+R22+sqrt(R11^2-
2*R11*R22+4*R12^2+R22^2)))*t)*R12*M20/sqrt(R11^2-2*R11*R22+4*R12^2+R22^2);

#with diffusion attenuation


attGradient := proc (t, DiffConstant, ldelta, gyro, gradient) options
operator, arrow;
exp((-1)*DiffConstant*gyro^2*0.1e7^2*ldelta^2*0.1e-2^2*gradient^2*0.1e-1^2*(t-
(1/3000)*ldelta)) end proc;


#HOESY buildup with diffusion attenuation
buildUpHOESY_diff := proc (t,R12,R11,R22,M10,M20, DiffConstant, ldelta, gyro,
gradient) options operator, arrow;
buildUpHOESY(t,R12,R11,R22,M10,M20)*attGradient(t, DiffConstant, ldelta, gyro,
gradient) end proc;
```

```
#T1 relaxation function for inversion-recovery, with imperfect inversion (A<1)

relaxation := proc (t, T, A) options operator, arrow; 1-2*A*exp(-t/T) end
proc;
> #Retrieve the data and parameters
myCommonDirectory:="C:/Program Files/Maple 2015/dataPyr13/";


#main directory name
mySampleName_1:="C3PyrFSILi1M50C";



#experiment number for quantitative 1D spectrum for 1H
experimentNumber_1D1H:="201";
processedDataNumber_1D1H:="1";


#experiment number for 1H T1 measurement

experimentNumber_1HT1:="100";
processedDataNumber_1HT1:="1";

#experiment number for 1H HOESY
experimentNumber_1HHOESY:="21";
processedDataNumber_1HHOESY:="1";


sampling_1HHOESY:=4;

# sample mass if needed
mySampleMass_1:=1;


#set up for 1H experiments (display and sampling)
min_shift:=1;
max_shift:=5;
range_extension:=0.1;
tick_spacing:=1;

sampling_1D1H:=8;
sampling_1HT1:=16;


#limits between which the peaks are located for 1H

limitleft := 1.0;
limitright := 4.0;


#intensity cutoff for 1H T1
intensity_cutoff:=10;

#intensity cutoff for 1H 1D
intensity_cutoff_1D:=10;
```

```
#intensity cutoff for 1H HOESY
intensity_cutoff_HOESY:=1e-5;

#delay to add in the VDList
DelayToAdd := 0.1254e-2;

#experiment number for 7Li T1 measurement
experimentNumber_7LiT1:="103";
processedDataNumber_7LiT1:="1";


#range to display for 7Li
min_shift_7Li:=-3;
max_shift_7Li:=3;
range_extension:=0.1;

sampling_7LiT1:=4;

#limits between which the peak is located

limitleft_7Li := -1.5;
limitright_7Li := 1.5;
intensity_cutoff_7Li:=0.2;


#if the first HOESY slice is bad, skip it
RowToStart:=2;

#for the fit of the HOESY build-up
g_Li := 103.962;
g_H := 267.513;

M_LiFSI:=187.07;
M_P13FSI:=308.37;

mass_LiFSI:=0.187*1;
mass_P13FSI:=1;

Li_natural_abundance:=0.9241;

Ratio_Li_Cation := (mass_LiFSI/M_LiFSI)/(mass_P13FSI/M_P13FSI);

Max_Intensity:=g_Li*Ratio_Li_Cation/g_H*Li_natural_abundance;
I_0 := [1, 1, 1, 1, 1, 1];

#for gradient selection in HOESY
ldelta:=1;
gyro:=g_Li;

#SINE pulse
shape_factor:=evalf(int(sin(x), x=0..Pi)/Pi);

gradient:=54*shape_factor;
```

```
#Fit with diffusion
DLi := 2.50E-11;
D1H := 3.95E-11;
myDiffConstant := (D1H+DLi)/2;

> #plot the data without adjustment

myNormalizedDataMatrix_1H1D:=createNormalized1DData(myCommonDirectory,mySample
Name_1,experimentNumber_1D1H,processedDataNumber_1D1H,mySampleMass_1,1,samplin
g_1D1H,min_shift-range_extension,max_shift+range_extension):
plot_1H1D:=plot(myNormalizedDataMatrix_1H1D[1],myNormalizedDataMatrix_1H1D[2],
color=red,axes=boxed,style=point):
displayRMN([plot_1H1D],tick_spacing,min_shift,max_shift);

>
myNormalizedDataMatrix_1HT1:=createNormalized2DTimeData(myCommonDirectory,mySa
mpleName_1,experimentNumber_1HT1,processedDataNumber_1HT1,mySampleMass_1,1,sam
pling_1HT1,min_shift-range_extension,max_shift+range_extension):

Dimension(myNormalizedDataMatrix_1HT1[3]);

displayRMN([plot(myNormalizedDataMatrix_1HT1[2],Vector(myNormalizedDataMatrix_
1HT1[3][8,1..]),style=line,axes=boxed)],tick_spacing,min_shift,max_shift);
> surfdata(myNormalizedDataMatrix_1HT1[3], colorscheme = ["zgradient",
["Blue", "Red"]], style = patchnogrid, orientation = [50, 80], labels =
[Slice,Chemical*shift*{ppm}, Intensity]);

rowToPlot:=8;
displayRMN([plot([seq([myNormalizedDataMatrix_1HT1[2][i],myNormalizedDataMatri
x_1HT1[3][rowToPlot,i]],i=1...Dimension(myNormalizedDataMatrix_1HT1[2]))])],ti
ck_spacing,min_shift,max_shift);


> #definition of the peaks

nPeaks_T1_H := 6;


#array of peaks parameters
peak_T1_H := Matrix(5, nPeaks_T1_H, [[A_T1_H_1, A_T1_H_2, A_T1_H_3, A_T1_H_4,
A_T1_H_5, A_T1_H_6], [I_T1_H_1, I_T1_H_2, I_T1_H_3, I_T1_H_4, I_T1_H_5,
I_T1_H_6], [T1_H_1, T1_H_2, T1_H_3, T1_H_4, T1_H_5, T1_H_6], [x_T1_H_1,
x_T1_H_2, x_T1_H_3, x_T1_H_4, x_T1_H_5, x_T1_H_6], [w_T1_H_1, w_T1_H_2,
w_T1_H_3, w_T1_H_4, w_T1_H_5, w_T1_H_6]]);


#collection of T1 build-up curves
intensityMatrix := Matrix(Dimension(myNormalizedDataMatrix_1HT1[3]));

MonSpectre_T1_H := add(peak_T1_H[2, i]*relaxation(t, peak_T1_H[3, i],
peak_T1_H[1, i])*Lorentzienne(peak_T1_H[4, i], peak_T1_H[5, i]), i = 1 ..
nPeaks_T1_H);

> #initial values for the fit


#inversion efficiency
```

```
Ai_T1_H_1 := 0.96;
Ai_T1_H_2 := 0.96;
Ai_T1_H_3 := 0.96;
Ai_T1_H_4 := 0.96;
Ai_T1_H_5 := 0.96;
Ai_T1_H_6 := 0.96;

#peak intensities
Ii_T1_H_1 := 160;
Ii_T1_H_2 := 140;
Ii_T1_H_3 := 250;
Ii_T1_H_4 := 180;
Ii_T1_H_5 := 140;
Ii_T1_H_6 := 250;


#T1 values
T1i_H_1 := 1.12;
T1i_H_2 := .87;
T1i_H_3 := 1.25;
T1i_H_4 := .90;
T1i_H_5 := .789;
T1i_H_6 := 1.08;

#width values
wi_T1_H_1 := 1.4e-2;
wi_T1_H_2 := 2.1e-2;
wi_T1_H_4 := 1.4e-2;
wi_T1_H_3 := 0.7e-2;
wi_T1_H_5 := 1.8e-2;
wi_T1_H_6 := 1.3e-2;


#peak positions in ppm
xi_T1_H_1 := 1.26;
xi_T1_H_2 := 2.07;
xi_T1_H_3 := 2.48;
xi_T1_H_4 := 3.27;
xi_T1_H_5 := 3.51;
xi_T1_H_6 := 3.72;

InitialGuesses := Matrix(5, nPeaks_T1_H, [[Ai_T1_H_1, Ai_T1_H_2, Ai_T1_H_3,
Ai_T1_H_4, Ai_T1_H_5, Ai_T1_H_6], [Ii_T1_H_1, Ii_T1_H_2, Ii_T1_H_3, Ii_T1_H_4,
Ii_T1_H_5, Ii_T1_H_6], [T1i_H_1, T1i_H_2, T1i_H_3, T1i_H_4, T1i_H_5, T1i_H_6],
[xi_T1_H_1, xi_T1_H_2, xi_T1_H_3, xi_T1_H_4, xi_T1_H_5, xi_T1_H_6],
[wi_T1_H_1, wi_T1_H_2, wi_T1_H_3, wi_T1_H_4, wi_T1_H_5, wi_T1_H_6]]);
> #fit parameters: set to true for gain of time


fixedPositions := false;
fixedWidths := false;
fixedA := true;



if fixedPositions then
x_T1_H_1 := xi_T1_H_1;
```

```
x_T1_H_2 := xi_T1_H_2;
x_T1_H_3 := xi_T1_H_3;
x_T1_H_4 := xi_T1_H_4;
x_T1_H_5 := xi_T1_H_5;
x_T1_H_6 := xi_T1_H_6;

else
unassign('x_T1_H_1','x_T1_H_2','x_T1_H_3','x_T1_H_4','x_T1_H_5','x_T1_H_6');

end if;

if fixedWidths then
w_T1_H_1 := wi_T1_H_1;
w_T1_H_2 := wi_T1_H_2;
w_T1_H_3 := wi_T1_H_3;
w_T1_H_4 := wi_T1_H_4;
w_T1_H_5 := wi_T1_H_5;
w_T1_H_6 := wi_T1_H_6;

else
unassign('w_T1_H_1','w_T1_H_2','w_T1_H_3','w_T1_H_4','w_T1_H_5','w_T1_H_6');

end if;

if fixedA then
A_T1_H_1:=Ai_T1_H_1;
A_T1_H_2:=Ai_T1_H_2;
A_T1_H_3:=Ai_T1_H_3;
A_T1_H_4:=Ai_T1_H_4;
A_T1_H_5:=Ai_T1_H_5;
A_T1_H_6:=Ai_T1_H_6;

else
unassign('A_T1_H_1','A_T1_H_2','A_T1_H_3','A_T1_H_4','A_T1_H_5','A_T1_H_6');

end if;
> #Initial values display spectra

MonSpectreInitial_T1_H := (delta, t) -> add(InitialGuesses[2, i]*relaxation(t,
InitialGuesses[3, i], InitialGuesses[1, i])*Lorentzienne(InitialGuesses[4, i],
InitialGuesses[5, i]), i = 1 .. nPeaks_T1_H):


plotinitial_T1_H := plot3d(MonSpectreInitial_T1_H(delta, t), delta = min_shift
.. max_shift, t = myNormalizedDataMatrix_1HT1[1][1] ..
myNormalizedDataMatrix_1HT1[1][Dimension(myNormalizedDataMatrix_1HT1[1])]);

displayedRow_H_1 := 2:
displayedRow_H_2 := 7:

plotinitial_H_1 := plot(MonSpectreInitial_T1_H(delta,
myNormalizedDataMatrix_1HT1[1][displayedRow_H_1]), delta = min_shift ..
max_shift, color = "DarkBlue"):
plotinitial_H_2 := plot(MonSpectreInitial_T1_H(delta,
myNormalizedDataMatrix_1HT1[1][displayedRow_H_2]), delta = min_shift ..
max_shift, color = "DarkBlue"):
```

```
oneRow_H_1 :=
plot([seq([myNormalizedDataMatrix_1HT1[2][i],myNormalizedDataMatrix_1HT1[3][di
splayedRow_H_1,i]],i=1..Dimension(myNormalizedDataMatrix_1HT1[2]))],style=poin
t,color=red):
oneRow_H_2 :=
plot([seq([myNormalizedDataMatrix_1HT1[2][i],myNormalizedDataMatrix_1HT1[3][di
splayedRow_H_2,i]],i=1..Dimension(myNormalizedDataMatrix_1HT1[2]))],style=poin
t,color=red):


displayedRow_H_1;
myNormalizedDataMatrix_1HT1[1][displayedRow_H_1];

displayRMN([oneRow_H_1, plotinitial_H_1],tick_spacing,min_shift,max_shift);

displayedRow_H_2;
myNormalizedDataMatrix_1HT1[1][displayedRow_H_2];


displayRMN([oneRow_H_2, plotinitial_H_2],tick_spacing,min_shift,max_shift);


> #preparing the fit

#building the residues (X-Xexp)
residu_T1_H := List();

for j to Dimension(myNormalizedDataMatrix_1HT1[1]) do
for i by 1 to Dimension(myNormalizedDataMatrix_1HT1[2]) do

if myNormalizedDataMatrix_1HT1[2][i] < limitleft or
myNormalizedDataMatrix_1HT1[2][i] > limitright or
abs(myNormalizedDataMatrix_1HT1[3][j,i])<intensity_cutoff then


else residu_T1_H := [op(residu_T1_H), eval(MonSpectre_T1_H, [delta =
myNormalizedDataMatrix_1HT1[2][i], t = myNormalizedDataMatrix_1HT1[1][j]])-
myNormalizedDataMatrix_1HT1[3][j, i]]
end if;
end do;
end do;

nops(residu_T1_H);
>
> startingValues_T1:={I_T1_H_1 = Ii_T1_H_1, I_T1_H_2 = Ii_T1_H_2, I_T1_H_3 =
Ii_T1_H_3, I_T1_H_4 = Ii_T1_H_4, I_T1_H_5 = Ii_T1_H_5, I_T1_H_6 = Ii_T1_H_6,
T1_H_1 = T1i_H_1, T1_H_2 = T1i_H_2, T1_H_3 = T1i_H_3, T1_H_4 = T1i_H_4, T1_H_5
= T1i_H_5, T1_H_6 = T1i_H_6};

if fixedPositions then
else
startingValues_T1:={op(startingValues_T1), x_T1_H_1 = xi_T1_H_1, x_T1_H_2 =
xi_T1_H_2, x_T1_H_3 = xi_T1_H_3, x_T1_H_4 = xi_T1_H_4, x_T1_H_5 = xi_T1_H_5,
x_T1_H_6 = xi_T1_H_6};
end if;

if fixedWidths then
```

```
else
startingValues_T1:={op(startingValues_T1),w_T1_H_1 = wi_T1_H_1, w_T1_H_2 =
wi_T1_H_2, w_T1_H_3 = wi_T1_H_3, w_T1_H_4 = wi_T1_H_4, w_T1_H_5 = wi_T1_H_5,
w_T1_H_6 = wi_T1_H_6};
end if;

if fixedA then
else
startingValues_T1:={op(startingValues_T1),A_T1_H_1=1,
A_T1_H_2=1,A_T1_H_3=1,A_T1_H_4=1,A_T1_H_5=1,A_T1_H_6=1};
end if;

solution_T1_H := LSSolve(residu_T1_H, initialpoint = startingValues_T1);

> for displayedRow from 1 to Dimension(myNormalizedDataMatrix_1HT1[1]) do


displayRMN([plot([seq([myNormalizedDataMatrix_1HT1[2][i],myNormalizedDataMatri
x_1HT1[3][displayedRow,i]],i=1..Dimension(myNormalizedDataMatrix_1HT1[2]))],st
yle=point,color=red,legend = [spectrum]),plot(eval(MonSpectre_T1_H,
[op(solution_T1_H[2]), t = myNormalizedDataMatrix_1HT1[1][displayedRow]]),
delta = min_shift .. max_shift, color = "DarkBlue", legend = [fit], labels =
[Chemical*shift*{ppm}, Intensity])],tick_spacing,min_shift,max_shift);
end do;


> T1_H_1_sol := subs(solution_T1_H[2], T1_H_1);
T1_H_2_sol := subs(solution_T1_H[2], T1_H_2);
T1_H_3_sol := subs(solution_T1_H[2], T1_H_3);
T1_H_4_sol := subs(solution_T1_H[2], T1_H_4);
T1_H_5_sol := subs(solution_T1_H[2], T1_H_5);
T1_H_6_sol := subs(solution_T1_H[2], T1_H_6);

T1_H := [T1_H_1_sol, T1_H_2_sol, T1_H_3_sol, T1_H_4_sol, T1_H_5_sol,
T1_H_6_sol];

x_T1_H_1_sol := subs(solution_T1_H[2], x_T1_H_1);
x_T1_H_2_sol := subs(solution_T1_H[2], x_T1_H_2);
x_T1_H_3_sol := subs(solution_T1_H[2], x_T1_H_3);
x_T1_H_4_sol := subs(solution_T1_H[2], x_T1_H_4);
x_T1_H_5_sol := subs(solution_T1_H[2], x_T1_H_5);
x_T1_H_6_sol := subs(solution_T1_H[2], x_T1_H_6);

w_T1_H_1_sol := subs(solution_T1_H[2], w_T1_H_1);
w_T1_H_2_sol := subs(solution_T1_H[2], w_T1_H_2);
w_T1_H_3_sol := subs(solution_T1_H[2], w_T1_H_3);
w_T1_H_4_sol := subs(solution_T1_H[2], w_T1_H_4);
w_T1_H_5_sol := subs(solution_T1_H[2], w_T1_H_5);
w_T1_H_6_sol := subs(solution_T1_H[2], w_T1_H_6);
> #calculation of T1 for lithium - INDICATE HOW TO PREPARE THE DATAFILE


#Retrieve the data and parameters

#plot the data without adjustment
```

```
myNormalizedDataMatrix_7LiT1:=createNormalized2DTimeData(myCommonDirectory,mySampleName_1,experimentNumber_7LiT1,processedDataNumber_7LiT1,mySampleMass_1,1,sampling_7LiT1,min_shift_7Li-range_extension,max_shift_7Li+range_extension):

Dimension(myNormalizedDataMatrix_7LiT1[3]);


#first plot
surfdata(myNormalizedDataMatrix_7LiT1[3], colorscheme = ["zgradient", ["Blue", "Red"]], style = patchnogrid, orientation = [50, 80], labels = [Slice,Chemical*shift*{ppm}, Intensity]);

> #making the fit function
nPeaks_Li := 1;

peak_T1_Li := Matrix(5, nPeaks_Li, [[A_T1_Li_1], [I_T1_Li_1], [T1_Li_1], [x_T1_Li_1], [w_T1_Li_1]]);

fixedPositions := false;
fixedWidths := false;
fixedA := false;

Ai_T1_Li_1 := 1.0;
Ii_T1_Li_1 := 10;
T1i_Li_1 := 1;
wi_T1_Li_1 := 0.04;
xi_T1_Li_1 := -0.075;

if fixedPositions then
x_T1_Li_1 := xi_T1_Li_1
else
unassign('x_T1_Li_1');
end if;

if fixedWidths then
w_T1_Li_1 := wi_T1_Li_1

else
unassign('w_T1_Li_1');
end if;

if fixedA then
A_T1_Li_1 := Ai_T1_Li_1

else
unassign('A_T1_Li_1');
end if;


InitialGuesses := Matrix(5, nPeaks_Li, [[Ai_T1_Li_1], [Ii_T1_Li_1], [T1i_Li_1], [xi_T1_Li_1], [wi_T1_Li_1]]);
MonSpectreInitial_T1_Li :=  (delta,t)->add(InitialGuesses[2, i]*relaxation(t, InitialGuesses[3, i], InitialGuesses[1, i])*Lorentzienne(InitialGuesses[4, i], InitialGuesses[5, i]), i = 1 .. nPeaks_Li);
```

```
MonSpectre_T1_Li :=add(peak_T1_Li[2, i]*relaxation(t, peak_T1_Li[3, i],
peak_T1_Li[1, i])*Lorentzienne(peak_T1_Li[4, i], peak_T1_Li[5, i]), i = 1 ..
nPeaks_Li);


plotinitialLi := plot3d(MonSpectreInitial_T1_Li(delta,t), delta =
min_shift_7Li .. max_shift_7Li, t = myNormalizedDataMatrix_7LiT1[1][1] ..
myNormalizedDataMatrix_7LiT1[1][Dimension(myNormalizedDataMatrix_7LiT1[1])],
color = blue);

displayedRow_Li_1 := 2;
displayedRow_Li_2 := 7;

myNormalizedDataMatrix_7LiT1[1][displayedRow_Li_1];
myNormalizedDataMatrix_7LiT1[1][displayedRow_Li_2];

plotinitial_Li_1 := plot(MonSpectreInitial_T1_Li(delta,
myNormalizedDataMatrix_7LiT1[1][displayedRow_Li_1]), delta = min_shift_7Li ..
max_shift_7Li, color = "DarkBlue"):
plotinitial_Li_2 := plot(MonSpectreInitial_T1_Li(delta,
myNormalizedDataMatrix_7LiT1[1][displayedRow_Li_2]), delta = min_shift_7Li ..
max_shift_7Li, color = "DarkBlue"):

oneRow_Li_1 := plot([seq([myNormalizedDataMatrix_7LiT1[2][i],
myNormalizedDataMatrix_7LiT1[3][displayedRow_Li_1, i]], i = 1 ..
Dimension(myNormalizedDataMatrix_7LiT1[2]))], color =
red,style=point,axes=boxed):
oneRow_Li_2 := plot([seq([myNormalizedDataMatrix_7LiT1[2][i],
myNormalizedDataMatrix_7LiT1[3][displayedRow_Li_2, i]], i = 1 ..
Dimension(myNormalizedDataMatrix_7LiT1[2]))], color =
red,style=point,axes=boxed):

displayRMN([oneRow_Li_1,
plotinitial_Li_1],tick_spacing,min_shift_7Li,max_shift_7Li);
displayRMN([oneRow_Li_2,
plotinitial_Li_2],tick_spacing,min_shift_7Li,max_shift_7Li);

> #preparing the residues for the fit

#building the residues (X-Xexp)
residu_T1_Li := List();

for j to Dimension(myNormalizedDataMatrix_7LiT1[1]) do
for i by 1 to Dimension(myNormalizedDataMatrix_7LiT1[2]) do

if myNormalizedDataMatrix_7LiT1[2][i] < limitleft_7Li or
myNormalizedDataMatrix_7LiT1[2][i] > limitright_7Li or
abs(myNormalizedDataMatrix_7LiT1[3][j,i])<intensity_cutoff_7Li then

else residu_T1_Li := [op(residu_T1_Li), eval(MonSpectre_T1_Li, [delta =
myNormalizedDataMatrix_7LiT1[2][i], t = myNormalizedDataMatrix_7LiT1[1][j]])-
myNormalizedDataMatrix_7LiT1[3][j, i]]
end if;
end do;
end do;
nops(residu_T1_Li);
```

```
>
> #Least Squares minimization


startingValues_T1Li:={I_T1_Li_1 = Ii_T1_Li_1, T1_Li_1 = T1i_Li_1};

if fixedWidths then
else
startingValues_T1Li:={op(startingValues_T1Li),w_T1_Li_1 = wi_T1_Li_1};
end if;

if fixedPositions then
else
startingValues_T1Li:={op(startingValues_T1Li),x_T1_Li_1 = xi_T1_Li_1};
end if;

if fixedA then
else
startingValues_T1Li:={op(startingValues_T1Li), A_T1_Li_1 = Ai_T1_Li_1 };
end if;

solution_T1_Li := LSSolve(residu_T1_Li, initialpoint = startingValues_T1Li );

T1_Li_1_sol := subs(solution_T1_Li[2], T1_Li_1);
> plot3d(eval(MonSpectre_T1_Li,[op(solution_T1_Li[2])]),
delta=min_shift_7Li..max_shift_7Li,t = myNormalizedDataMatrix_7LiT1[1][1] ..
myNormalizedDataMatrix_7LiT1[1][Dimension(myNormalizedDataMatrix_7LiT1[1])]);
> for displayedRow from 1 to Dimension(myNormalizedDataMatrix_7LiT1[1]) do


displayRMN([plot([seq([myNormalizedDataMatrix_7LiT1[2][i],myNormalizedDataMatr
ix_7LiT1[3][displayedRow,i]],i=1..Dimension(myNormalizedDataMatrix_7LiT1[2]))]
,style=point,color=red,legend =
[spectrum],axes=boxed),plot(eval(MonSpectre_T1_Li, [op(solution_T1_Li[2]), t =
myNormalizedDataMatrix_7LiT1[1][displayedRow]]), delta = min_shift_7Li ..
max_shift_7Li, color = "DarkBlue", legend = [fit], labels =
[Chemical*shift*{ppm}, Intensity])],tick_spacing,min_shift_7Li,max_shift_7Li);
end do;
>
> Npeaks_ZG_H:=6;
peak_ZG_H := Matrix(3, Npeaks_ZG_H, [[I_ZG_H_1, I_ZG_H_2, I_ZG_H_3, I_ZG_H_4,
I_ZG_H_5, I_ZG_H_6], [x_ZG_H_1, x_ZG_H_2, x_ZG_H_3, x_ZG_H_4, x_ZG_H_5,
x_ZG_H_6], [w_ZG_H_1, w_ZG_H_2, w_ZG_H_3, w_ZG_H_4, w_ZG_H_5, w_ZG_H_6]]);
> fixedPositions := false;
fixedWidth := false;

xi_ZG_H_1:=x_T1_H_1_sol;
xi_ZG_H_2:=x_T1_H_2_sol;
xi_ZG_H_3:=x_T1_H_3_sol;
xi_ZG_H_4:=x_T1_H_4_sol;
xi_ZG_H_5:=x_T1_H_5_sol;
xi_ZG_H_6:=x_T1_H_6_sol;

wi_ZG_H_1:=w_T1_H_1_sol;
wi_ZG_H_2:=w_T1_H_2_sol;
wi_ZG_H_3:=w_T1_H_3_sol;
wi_ZG_H_4:=w_T1_H_4_sol;
```

```
wi_ZG_H_5:=w_T1_H_5_sol;
wi_ZG_H_6:=w_T1_H_6_sol;

Ii_ZG_H_1 := 10;
Ii_ZG_H_2 := 6;
Ii_ZG_H_3 := 9;
Ii_ZG_H_4 := 15;
Ii_ZG_H_5 := 10;
Ii_ZG_H_6 := 12;


if fixedPositions then
x_ZG_H_1:=xi_ZG_H_1;
x_ZG_H_2:=xi_ZG_H_2;
x_ZG_H_3:=xi_ZG_H_3;
x_ZG_H_4:=xi_ZG_H_4;
x_ZG_H_5:=xi_ZG_H_5;
x_ZG_H_6:=xi_ZG_H_6;
end if;

if fixedWidth then
w_ZG_H_1:=wi_ZG_H_1;
w_ZG_H_2:=wi_ZG_H_2;
w_ZG_H_3:=wi_ZG_H_3;
w_ZG_H_4:=w_T1_H_4_sol;
w_ZG_H_5:=w_T1_H_5_sol;
w_ZG_H_6:=w_T1_H_6_sol;
end if;


InitialGuesses := Matrix(3, Npeaks_ZG_H, [[Ii_ZG_H_1, Ii_ZG_H_2, Ii_ZG_H_3,
Ii_ZG_H_4, Ii_ZG_H_5, Ii_ZG_H_6], [xi_ZG_H_1, xi_ZG_H_2, xi_ZG_H_3, xi_ZG_H_4,
xi_ZG_H_5, xi_ZG_H_6], [wi_ZG_H_1, wi_ZG_H_2, wi_ZG_H_3, wi_ZG_H_4, wi_ZG_H_5,
wi_ZG_H_6]]);

MonSpectre_ZG := add(peak_ZG_H[1, i]^2*Lorentzienne(peak_ZG_H[2, i],
peak_ZG_H[3, i]), i = 1 .. Npeaks_ZG_H);
MonSpectreInitial_ZG_H := add(InitialGuesses[1,
i]^2*Lorentzienne(InitialGuesses[2, i], InitialGuesses[3, i]), i = 1 ..
Npeaks_ZG_H);

plotinitial_ZG_H := plot(MonSpectreInitial_ZG_H, delta = min_shift ..
max_shift, color = blue):


displayRMN([plot_1H1D,plotinitial_ZG_H],tick_spacing,min_shift,max_shift);
> residu_ZG := List();
for i to Dimension(myNormalizedDataMatrix_1H1D[1]) do


if myNormalizedDataMatrix_1H1D[1][i] < limitleft or
myNormalizedDataMatrix_1H1D[1][i] > limitright or
abs(myNormalizedDataMatrix_1H1D[2][i])<intensity_cutoff_1D then

else
```

```
residu_ZG := [op(residu_ZG), eval(MonSpectre_ZG, delta =
myNormalizedDataMatrix_1H1D[1][i])-myNormalizedDataMatrix_1H1D[2][i]];
end if;

end do:
nops(residu_ZG);
> startingValues_ZG:={I_ZG_H_1 = Ii_ZG_H_1, I_ZG_H_2 = Ii_ZG_H_2, I_ZG_H_3 =
Ii_ZG_H_3, I_ZG_H_4 = Ii_ZG_H_4, I_ZG_H_5 = Ii_ZG_H_5, I_ZG_H_6 = Ii_ZG_H_6};

if fixedPositions then
else
startingValues_ZG:={op(startingValues_ZG), x_ZG_H_1=xi_ZG_H_1,
x_ZG_H_2=xi_ZG_H_2, x_ZG_H_3=xi_ZG_H_3, x_ZG_H_4=xi_ZG_H_4,
x_ZG_H_5=xi_ZG_H_5, x_ZG_H_6=xi_ZG_H_6};
end if;

if fixedWidth then
else
startingValues_ZG:={op(startingValues_ZG), w_ZG_H_1=wi_ZG_H_1,
w_ZG_H_2=wi_ZG_H_2, w_ZG_H_3=wi_ZG_H_3, w_ZG_H_4=wi_ZG_H_4,
w_ZG_H_5=wi_ZG_H_5, w_ZG_H_6=wi_ZG_H_6};
end if;


solution_ZG := LSSolve(residu_ZG, initialpoint = startingValues_ZG);


> I_ZG_H_sol_1 := eval(I_ZG_H_1, solution_ZG[2]);
I_ZG_H_sol_2 := eval(I_ZG_H_2, solution_ZG[2]);
I_ZG_H_sol_3 := eval(I_ZG_H_3, solution_ZG[2]);
I_ZG_H_sol_4 := eval(I_ZG_H_4, solution_ZG[2]);
I_ZG_H_sol_5 := eval(I_ZG_H_5, solution_ZG[2]);
I_ZG_H_sol_6 := eval(I_ZG_H_6, solution_ZG[2]);

I_ZG_H_sol := Vector([I_ZG_H_sol_1^2, I_ZG_H_sol_2^2, I_ZG_H_sol_3^2,
I_ZG_H_sol_4^2, I_ZG_H_sol_5^2, I_ZG_H_sol_6^2]);
> Spectre_solution_ZG := eval(MonSpectre_ZG, solution_ZG[2]);

plotfinal_ZG := plot(Spectre_solution_ZG, delta = min_shift .. max_shift,
color = blue,legend = [fit]):
displayRMN([plot_1H1D, plotfinal_ZG],tick_spacing,min_shift,max_shift);
>
> #import HOESY dataplot
myNormalizedDataMatrix_1HHOESY:=createNormalized2DTimeData(myCommonDirectory,m
ySampleName_1,experimentNumber_1HHOESY,processedDataNumber_1HHOESY,mySampleMas
s_1,1,sampling_1HHOESY,min_shift-range_extension,max_shift+range_extension):


Dimension(myNormalizedDataMatrix_1HHOESY[3]);


> for i to Dimension(myNormalizedDataMatrix_1HHOESY[1]) do
myNormalizedDataMatrix_1HHOESY[1][i] :=
myNormalizedDataMatrix_1HHOESY[1][i]+DelayToAdd
end do:
myNormalizedDataMatrix_1HHOESY[1];
```

```
surfdata(myNormalizedDataMatrix_1HHOESY[3], colorscheme = ["zgradient",
["Blue", "Red"]], style = patchnogrid, orientation = [70, 70], labels =
[Mixing*Time*{s}, Chemical*shift*{ppm}, Intensity*{a . u}]);
> nPeaks_HOESY := 6;
peak_HOESY := Matrix(3, nPeaks_HOESY, [[I_HOESY_1, I_HOESY_2, I_HOESY_3,
I_HOESY_4, I_HOESY_5, I_HOESY_6], [x_HOESY_1, x_HOESY_2, x_HOESY_3, x_HOESY_4,
x_HOESY_5, x_HOESY_6], [w_HOESY_1, w_HOESY_2, w_HOESY_3, w_HOESY_4, w_HOESY_5,
w_HOESY_6]]);

HOESY_intensity_Matrix := Matrix(Dimension(myNormalizedDataMatrix_1HHOESY[1]),
nPeaks_HOESY);
> betterValues:=true;

fixedPositions := false;
fixedWidths := false;
Ii_HOESY_1 := 100;
Ii_HOESY_2 := 100;
Ii_HOESY_3 := 105;
Ii_HOESY_4 := 108;
Ii_HOESY_5 := 107;
Ii_HOESY_6 := 106;

xi_HOESY_1:=x_T1_H_1_sol;
xi_HOESY_2:=x_T1_H_2_sol;
xi_HOESY_3:=x_T1_H_3_sol;
xi_HOESY_4:=x_T1_H_4_sol;
xi_HOESY_5:=x_T1_H_5_sol;
xi_HOESY_6:=x_T1_H_6_sol;

wi_HOESY_1:=w_T1_H_1_sol;
wi_HOESY_2:=w_T1_H_2_sol;
wi_HOESY_3:=w_T1_H_3_sol;
wi_HOESY_4:=w_T1_H_4_sol;
wi_HOESY_5:=w_T1_H_5_sol;
wi_HOESY_6:=w_T1_H_6_sol;



if betterValues then


Ii_HOESY_1 := 669;
Ii_HOESY_2 := 340;
Ii_HOESY_3 := 698;
Ii_HOESY_4 := 582;
Ii_HOESY_5 := 340;
Ii_HOESY_6 := 614;

wi_HOESY_1 := 0.187e-1;
wi_HOESY_2 := 0.242e-1;
wi_HOESY_3 := 0.187e-1;
wi_HOESY_4 := 0.117e-1;
wi_HOESY_5 := 0.214e-1;
wi_HOESY_6 := 0.188e-1;

xi_HOESY_1 := 1.269;
xi_HOESY_2 := 2.08;
```

```
xi_HOESY_3 := 2.49;
xi_HOESY_4 := 3.285;
xi_HOESY_5 := 3.527;
xi_HOESY_6 := 3.71;

end if;



if fixedPositions then
x_HOESY_1 := xi_HOESY_1;
x_HOESY_2 := xi_HOESY_2;
x_HOESY_3 := xi_HOESY_3;
x_HOESY_4 := xi_HOESY_4;
x_HOESY_5 := xi_HOESY_5;
x_HOESY_6 := xi_HOESY_6;

else
unassign('x_HOESY_1','x_HOESY_2','x_HOESY_3','x_HOESY_4','x_HOESY_5','x_HOESY_
6');
end if;

if fixedWidths then
w_HOESY_1 := wi_HOESY_1;
w_HOESY_2 := wi_HOESY_2;
w_HOESY_3 := wi_HOESY_3;
w_HOESY_4 := wi_HOESY_4;
w_HOESY_5 := wi_HOESY_5;
w_HOESY_6 := wi_HOESY_6;
else
unassign('w_HOESY_1','w_HOESY_2','w_HOESY_3','w_HOESY_4','w_HOESY_5','w_HOESY_
6');
end if;

InitialGuesses_HOESY := Matrix(3, nPeaks_HOESY, [[Ii_HOESY_1, Ii_HOESY_2,
Ii_HOESY_3, Ii_HOESY_4, Ii_HOESY_5, Ii_HOESY_6], [xi_HOESY_1, xi_HOESY_2,
xi_HOESY_3, xi_HOESY_4, xi_HOESY_5, xi_HOESY_6], [wi_HOESY_1, wi_HOESY_2,
wi_HOESY_3, wi_HOESY_4, wi_HOESY_5, wi_HOESY_6]]);

MonSpectre_HOESY := add(peak_HOESY[1, i]^2*Lorentzienne(peak_HOESY[2, i],
peak_HOESY[3, i]), i = 1 .. nPeaks_HOESY);
MonSpectreInitial_HOESY := add(InitialGuesses_HOESY[1,
i]^2*Lorentzienne(InitialGuesses_HOESY[2, i], InitialGuesses_HOESY[3, i]), i =
1 .. nPeaks_HOESY);

plotinitial := plot(MonSpectreInitial_HOESY, delta = min_shift .. max_shift,
color = blue, axes=boxed):
displayRMN([plotinitial],tick_spacing,min_shift,max_shift);
>
> for j from 1 to Dimension(myNormalizedDataMatrix_1HHOESY[1]) by 1 do


residu_HOESY := List();



for i from 1 to Dimension(myNormalizedDataMatrix_1HHOESY[2])  by 1 do
```

```
if myNormalizedDataMatrix_1HHOESY[2][i] < limitleft or
myNormalizedDataMatrix_1HHOESY[2][i] > limitright or
abs(myNormalizedDataMatrix_1HHOESY[3][j,i])<intensity_cutoff_HOESY then
else
residu_HOESY := [op(residu_HOESY), eval(MonSpectre_HOESY, [delta =
myNormalizedDataMatrix_1HHOESY[2][i]])-myNormalizedDataMatrix_1HHOESY[3][j,
i]]:
end if:


end do:
nops(residu_HOESY);

startingValues :=
{I_HOESY_1 = Ii_HOESY_1,
 I_HOESY_2 = Ii_HOESY_2,
 I_HOESY_3 = Ii_HOESY_3,
 I_HOESY_4 = Ii_HOESY_4,
 I_HOESY_5 = Ii_HOESY_5,
 I_HOESY_6 = Ii_HOESY_6};

if fixedPositions then
else
startingValues:={op(startingValues),
 x_HOESY_1 = xi_HOESY_1,
 x_HOESY_2 = xi_HOESY_2,
 x_HOESY_3 = xi_HOESY_3,
 x_HOESY_4 = xi_HOESY_4,
 x_HOESY_5 = xi_HOESY_5,
 x_HOESY_6 = xi_HOESY_6};
end if;

if fixedWidth then
else
startingValues:={op(startingValues),
 w_HOESY_1 = wi_HOESY_1,
 w_HOESY_2 = wi_HOESY_2,
 w_HOESY_3 = wi_HOESY_3,
 w_HOESY_4 = wi_HOESY_4,
 w_HOESY_5 = wi_HOESY_5,
 w_HOESY_6 = wi_HOESY_6};
end if;
solution_HOESY := LSSolve(residu_HOESY, initialpoint = startingValues,
iterationlimit = 1000000000);

display(plot([seq([myNormalizedDataMatrix_1HHOESY[2][i],myNormalizedDataMatrix
_1HHOESY[3][j,i]], i = 1 .. Dimension(myNormalizedDataMatrix_1HHOESY[2]))],
color = red, style=point, legend = [spectrum]), plot(eval(MonSpectre_HOESY,
[op(solution_HOESY[2])]), delta = min_shift .. max_shift, color = "DarkBlue",
legend = [fit]), labels = [Chemical*shift*{ppm}, Intensity]);

HOESY_intensity_Matrix[j, 1] := eval(I_HOESY_1^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[1]);
HOESY_intensity_Matrix[j, 2] := eval(I_HOESY_2^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[2]);
```

```
HOESY_intensity_Matrix[j, 3] := eval(I_HOESY_3^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[3]);
HOESY_intensity_Matrix[j, 4] := eval(I_HOESY_4^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[4]);
HOESY_intensity_Matrix[j, 5] := eval(I_HOESY_5^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[5]);
HOESY_intensity_Matrix[j, 6] := eval(I_HOESY_6^2,
[op(solution_HOESY[2])])/(I_ZG_H_sol[6]);
end do;

>
> colorTable:=["Orange","Purple","Red","Black","Green","Blue"];

Plot_Hoesy_BuildUp:=List(nPeaks_HOESY);

for displayedPeak from 1 to nPeaks_HOESY do


Plot_Hoesy_BuildUp[displayedPeak]:=
pointplot([seq([myNormalizedDataMatrix_1HHOESY[1][i],
HOESY_intensity_Matrix[i, displayedPeak]], i = RowToStart ..
Dimension(myNormalizedDataMatrix_1HHOESY[1]))], color =
colorTable[displayedPeak], symbol = circle,labels = ["Mixing time in (s)", "H
magnetization"]);

end do:

display([seq(Plot_Hoesy_BuildUp[i],i=1..nPeaks_HOESY)]);
> T1_H;


bruker:=false;
if bruker then
T1_H:=[1.12,0.791,1.05,0.74,0.707,0.910];
T1_Li_1_sol:=1.086;
end if;

R11 := 1/T1_Li_1_sol;
R22 := map(proc (x) options operator, arrow; 1/x end proc, T1_H);

R12 := [0.5e-6, 0.5e-6, 0.5e-6, 0.5e-6, 0.5e-6, 0.5e-6];




>
>
> #test of the gradient attenuation
plot(attGradient(t, myDiffConstant, ldelta, g_Li, gradient), t=0..20,
view=0.4..1);

for j from 1 to nPeaks_HOESY do

residu_buildup := List();

for i from RowToStart to Dimension(myNormalizedDataMatrix_1HHOESY[1]) do
```

```
residu_buildup := [op(residu_buildup),
(eval(buildUpHOESY_diff(t,R12diff_u,R11,R22[j],1,Max_Intensity*I_0[j],myDiffCo
nstant, ldelta, gyro, gradient), t = myNormalizedDataMatrix_1HHOESY[1][i])-
HOESY_intensity_Matrix[i, j])];
end do;
solution_buildup := LSSolve(residu_buildup, initialpoint = {R12diff_u = 1e-
6});
R12diff[j] := abs(eval(R12diff_u, solution_buildup[2]));

Magnitude;
I_0[j]*Max_Intensity;
HOESYrate;
R12diff[j];

display(Plot_Hoesy_BuildUp[j],
plot(buildUpHOESY_diff(t,R12diff[j],R11,R22[j],1,Max_Intensity*I_0[j],myDiffCo
nstant, ldelta, gyro, gradient), t = 0 .. 5, color = colorTable[j],legend =
FitPeak[j]));

end do: #display fit with diff one by one

for displayedPeak from 1 to nPeaks_HOESY do


Buildup_fit[displayedPeak]:=plot(buildUpHOESY_diff(t,R12diff[displayedPeak],R1
1,R22[displayedPeak],1,Max_Intensity*I_0[displayedPeak],myDiffConstant,
ldelta, gyro, gradient), t = 0 .. 5, color = colorTable[displayedPeak],legend
= FitPeakWithDiffusion[displayedPeak]):
end do:
display(seq(Plot_Hoesy_BuildUp[i], i=1..nPeaks_HOESY),seq(Buildup_fit[i],
i=1...nPeaks_HOESY)); #display fit with diff all in one

for j from 1 to nPeaks_HOESY do

residu_buildup := List();

for i from RowToStart to Dimension(myNormalizedDataMatrix_1HHOESY[1]) do
residu_buildup := [op(residu_buildup),
eval(buildUpHOESY(t,R12_u,R11,R22[j],1,Max_Intensity*I_0[j]), t =
myNormalizedDataMatrix_1HHOESY[1][i])-HOESY_intensity_Matrix[i, j]];
end do;
solution_buildup := LSSolve(residu_buildup, initialpoint = {R12_u = 1e-6});
R12[j] := abs(eval(R12_u, solution_buildup[2]));
Magnitude;
I_0[j]*Max_Intensity;
HOESYrate;
R12[j];

display(Plot_Hoesy_BuildUp[j],
plot(buildUpHOESY(t,R12[j],R11,R22[j],1,Max_Intensity*I_0[j]), t = 0 ... 5,
color = colorTable[j],linestyle = dot,thickness=3,symbolsize=12,legend =
peak[j]));

end do: #display fit without diff one by one
```

```
for j from 1 to nPeaks_HOESY do
display(Plot_Hoesy_BuildUp[j],
plot(buildUpHOESY(t,R12[j],R11,R22[j],1,Max_Intensity*I_0[j]), t = 0 .. 5,
color = colorTable[j],linestyle = dot,legend = "Fit without
diffusion"),plot(buildUpHOESY_diff(t,R12diff[j],R11,R22[j],1,Max_Intensity*I_0
[j],myDiffConstant, ldelta, gyro, gradient), t = 0 .. 5, color =
colorTable[j],legend = "Fit with diffusion"));
end do; #display fit with and without diff one by one

for displayedPeak from 1 to nPeaks_HOESY do


Buildup_fit_without_diff[displayedPeak]:=plot(buildUpHOESY(t,R12[displayedPeak
],R11,R22[displayedPeak],1,Max_Intensity*I_0[displayedPeak]), t = 0 .. 5,
color = colorTable[displayedPeak],linestyle = dot,legend =
FitPeakWithoutDiffusion[displayedPeak]):
end do:
display(seq(Plot_Hoesy_BuildUp[i],
i=1..nPeaks_HOESY),seq(Buildup_fit_without_diff[i],
i=1...nPeaks_HOESY));#display fit without diff all in one

display(seq(Plot_Hoesy_BuildUp[i], i=1..nPeaks_HOESY),seq(Buildup_fit[i],
i=1...nPeaks_HOESY),seq(Plot_Hoesy_BuildUp[i],
i=1..nPeaks_HOESY),seq(Buildup_fit_without_diff[i],
i=1...nPeaks_HOESY));#display fit with and without diff all in one


seq(R12diff[i],i=1..nPeaks_HOESY);

maximumHOESYrate:=max(seq(R12diff[i],i=1..nPeaks_HOESY));

pointplot({seq([j,R12diff[j]],j=1..6)},color=red,view=0..maximumHOESYrate*1.1)
;

pointplot({seq([j,maximumHOESYrate/(R12diff[j])],j=1..6)},color=blue,view=1..3
);

>

> #methyl
C_1:=[-1.58401,-0.82776,-1.0134];


#butyl ring
Cb_1:=[-1.49791,1.67573,-0.7344];
Cb_2:=[-2.93403,2.14171,-0.51603];

Cb_3:=[-3.63965,1.058,0.27381];
Cb_4:=[-2.5235,0.33293,0.99329];

#propyl chain
Cp_1:=[-0.03626,0.17116,0.723];
Cp_2:=[1.18053,0.15549,-0.20395];
Cp_3:=[2.47148,0.03334,0.59344];


#Nitrogen
```

```
N1:=[-1.37009,0.3205,-0.02589];



#Hydrogen
Hm_1:=[-0.78435,-0.84051,-1.72423];
Hm_2:=[-2.51368,-0.68981,-1.52485];
Hm_3:=[-1.60365,-1.75624,-0.48195];


Hb_1:=[-0.81199,2.38019,-0.31233];
Hb_2:=[-1.27151,1.59154,-1.77678];


Hb_3:=[-3.42059,2.28486,-1.45819];
Hb_4:=[-2.95607,3.07249,0.01129];


Hp_3:=[1.10133,-0.67701,-0.87146];
Hp_4:=[1.20372,1.07417,-0.75203];


Hb_5:=[-4.16404,0.39289,-0.38005];
Hb_6:=[-4.37105,1.45345,0.94729];


Hp_5:=[2.46310,-0.88188,1.14770];
Hp_6:=[2.55271,0.85888,1.26930];
Hp_7:=[3.30603,0.03716,-0.07619];


Hp_1:=[0.06521,0.99140,1.40256];
Hp_2:=[-0.06102,-0.77137,1.22889];

Hb_7:=[-2.24018,0.85720,1.88198];
Hb_8:=[-2.80621,-0.65008,1.30740];


barycenter:=(C_1+Cb_1+Cb_2+Cb_3+Cb_4+N1+Cp_1+Cp_2+Cp_3)/9;

C_1:=C_1-barycenter;

Cb_1:=Cb_1-barycenter;
Cb_2:=Cb_2-barycenter;
Cb_3:=Cb_3-barycenter;
Cb_4:=Cb_4-barycenter;

Cp_1:=Cp_1-barycenter;
Cp_2:=Cp_2-barycenter;
Cp_3:=Cp_3-barycenter;


N1:=N1-barycenter;

Hm_1:=Hm_1-barycenter;
Hm_2:=Hm_2-barycenter;
Hm_3:=Hm_3-barycenter;
```

```
Hb_1:=Hb_1-barycenter;
Hb_2:=Hb_2-barycenter;
Hb_3:=Hb_3-barycenter;
Hb_4:=Hb_4-barycenter;
Hp_3:=Hp_3-barycenter;
Hp_4:=Hp_4-barycenter;
Hb_5:=Hb_5-barycenter;
Hb_6:=Hb_6-barycenter;
Hp_5:=Hp_5-barycenter;
Hp_6:=Hp_6-barycenter;
Hp_7:=Hp_7-barycenter;
Hp_1:=Hp_1-barycenter;
Hp_2:=Hp_2-barycenter;
Hb_7:=Hb_7-barycenter;
Hb_8:=Hb_8-barycenter;
> rayon:=0.3;
rayon_H:=0.2;
vr:=5;
display({
sphere(C_1,rayon,color="DimGray"),

sphere(Cb_1,rayon,color="DimGray"),
sphere(Cb_2,rayon,color="DimGray"),
sphere(Cb_3,rayon,color="DimGray"),
sphere(Cb_4,rayon,color="DimGray"),

sphere(Cp_1,rayon,color="DimGray"),
sphere(Cp_2,rayon,color="DimGray"),
sphere(Cp_3,rayon,color="DimGray"),

sphere(N1,rayon,color=grey),


sphere(Hm_1,rayon_H,color=black),
sphere(Hm_2,rayon_H,color=black),
sphere(Hm_3,rayon_H,color=black),

sphere(Hb_1,rayon_H,color=blue),
sphere(Hb_2,rayon_H,color=blue),

sphere(Hb_3,rayon_H,color=red),
sphere(Hb_4,rayon_H,color=red),

sphere(Hp_3,rayon_H,color=purple),
sphere(Hp_4,rayon_H,color=purple),

sphere(Hb_5,rayon_H,color=red),
sphere(Hb_6,rayon_H,color=red),

sphere(Hp_5,rayon_H,color=yellow),
sphere(Hp_6,rayon_H,color=yellow),
sphere(Hp_7,rayon_H,color=yellow),

sphere(Hp_1,rayon_H,color=green),
sphere(Hp_2,rayon_H,color=green),

sphere(Hb_7,rayon_H,color=blue),
```

```
sphere(Hb_8,rayon_H,color=blue),
line(C_1,N1,color=black),
line(C_1,Hm_1,color=black),
line(C_1,Hm_2,color=black),
line(C_1,Hm_3,color=black),

line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),


line(Cb_1,Cb_2,color=black),

line(Cb_1,Hb_1,color=black),
line(Cb_1,Hb_2,color=black),

line(Cb_2,Cb_3,color=black),

line(Cb_2,Hb_3,color=black),
line(Cb_2,Hb_4,color=black),

line(Cb_3,Cb_4,color=black),

line(Cb_3,Hb_5,color=black),
line(Cb_3,Hb_6,color=black),
line(Cb_4,Hb_7,color=black),
line(Cb_4,Hb_8,color=black),

line(Cp_1,Cp_2,color=black),

line(Cp_1,Hp_1,color=black),
line(Cp_1,Hp_2,color=black),

line(Cp_2,Hp_3,color=black),
line(Cp_2,Hp_4,color=black),

line(Cp_2,Cp_3,color=black),
line(Cp_3,Hp_5,color=black),
line(Cp_3,Hp_6,color=black),
line(Cp_3,Hp_7,color=black)
},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr], axes=none);
>
> #color coding of the atoms

R12_max:=max(seq(R12[j],j=1..nPeaks_HOESY));
R12_min:=min(seq(R12[j],j=1..nPeaks_HOESY));
#R12_max:=0.25;
#R12_min:=0.01;
R12_color:=[seq((R12[j]-R12_min)/(R12_max-R12_min), j=1..nPeaks_HOESY)];
logR12diff:= [seq(sqrt(R12diff[i]),i=1..nPeaks_HOESY)];
R12_maxbis:=max(seq(logR12diff[j],j=1..nPeaks_HOESY));
R12_minbis:=min(seq(logR12diff[j],j=1..nPeaks_HOESY));
R12_logcolor:=[seq(((logR12diff[j]-R12_minbis)/(R12_maxbis-R12_minbis)),
j=1..nPeaks_HOESY)];
```

```
rayon:=0.3;
rayon_H:=0.3;
vr:=6;


display({
sphere(C_1,rayon,color="DimGray"),

sphere(Cb_1,rayon,color="DimGray"),
sphere(Cb_2,rayon,color="DimGray"),
sphere(Cb_3,rayon,color="DimGray"),
sphere(Cb_4,rayon,color="DimGray"),

sphere(Cp_1,rayon,color="DimGray"),
sphere(Cp_2,rayon,color="DimGray"),
sphere(Cp_3,rayon,color="DimGray"),

sphere(N1,rayon,color=grey),


sphere(Hm_1,rayon_H,color=RGB(R12_color[4],0,0.55-R12_color[4])),
sphere(Hm_2,rayon_H,color=RGB(R12_color[4],0,0.55-R12_color[4])),
sphere(Hm_3,rayon_H,color=RGB(R12_color[4],0,0.55-R12_color[4])),

sphere(Hb_1,rayon_H,color=RGB(R12_color[6],0,0.55-R12_color[6])),
sphere(Hb_2,rayon_H,color=RGB(R12_color[6],0,0.55-R12_color[6])),

sphere(Hb_3,rayon_H,color=RGB(R12_color[3],0,0.55-R12_color[3])),
sphere(Hb_4,rayon_H,color=RGB(R12_color[3],0,0.55-R12_color[3])),

sphere(Hp_3,rayon_H,color=RGB(R12_color[2],0,0.55-R12_color[2])),
sphere(Hp_4,rayon_H,color=RGB(R12_color[2],0,0.55-R12_color[2])),

sphere(Hb_5,rayon_H,color=RGB(R12_color[3],0,0.55-R12_color[3])),
sphere(Hb_6,rayon_H,color=RGB(R12_color[3],0,0.55-R12_color[3])),

sphere(Hp_5,rayon_H,color=RGB(R12_color[1],0,0.55-R12_color[1])),
sphere(Hp_6,rayon_H,color=RGB(R12_color[1],0,0.55-R12_color[1])),
sphere(Hp_7,rayon_H,color=RGB(R12_color[1],0,0.55-R12_color[1])),

sphere(Hp_1,rayon_H,color=RGB(R12_color[5],0,0.55-R12_color[5])),
sphere(Hp_2,rayon_H,color=RGB(R12_color[5],0,0.55-R12_color[5])),

sphere(Hb_7,rayon_H,color=RGB(R12_color[6],0,0.55-R12_color[6])),
sphere(Hb_8,rayon_H,color=RGB(R12_color[6],0,0.55-R12_color[6])),

line(C_1,N1,color=black),
line(C_1,Hm_1,color=black),
line(C_1,Hm_2,color=black),
line(C_1,Hm_3,color=black),

line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),


line(Cb_1,Cb_2,color=black),
```

```
line(Cb_1,Hb_1,color=black),
line(Cb_1,Hb_2,color=black),

line(Cb_2,Cb_3,color=black),

line(Cb_2,Hb_3,color=black),
line(Cb_2,Hb_4,color=black),

line(Cb_3,Cb_4,color=black),

line(Cb_3,Hb_5,color=black),
line(Cb_3,Hb_6,color=black),
line(Cb_4,Hb_7,color=black),
line(Cb_4,Hb_8,color=black),

line(Cp_1,Cp_2,color=black),

line(Cp_1,Hp_1,color=black),
line(Cp_1,Hp_2,color=black),

line(Cp_2,Hp_3,color=black),
line(Cp_2,Hp_4,color=black),

line(Cp_2,Cp_3,color=black),
line(Cp_3,Hp_5,color=black),
line(Cp_3,Hp_6,color=black),
line(Cp_3,Hp_7,color=black)

},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr], axes=none);


display({
sphere(C_1,rayon,color=RGB(R12_color[4],0,0.6-R12_color[4])),

sphere(Cb_1,rayon,color=RGB(R12_color[6],0,0.6-R12_color[6])),
sphere(Cb_2,rayon,color=RGB(R12_color[3],0,0.6-R12_color[3])),
sphere(Cb_3,rayon,color=RGB(R12_color[3],0,0.6-R12_color[3])),
sphere(Cb_4,rayon,color=RGB(R12_color[6],0,0.6-R12_color[6])),

sphere(Cp_1,rayon,color=RGB(R12_color[5],0,0.6-R12_color[5])),
sphere(Cp_2,rayon,color=RGB(R12_color[2],0,0.6-R12_color[2])),
sphere(Cp_3,rayon,color=RGB(R12_color[1],0,0.6-R12_color[1])),

sphere(N1,rayon,color=white),

line(C_1,N1,color=black),
line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),
line(Cb_1,Cb_2,color=black),
line(Cb_2,Cb_3,color=black),
line(Cb_3,Cb_4,color=black),
line(Cp_1,Cp_2,color=black),
line(Cp_2,Cp_3,color=black)

},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr], axes=none);
```

```
> vr:=7;

#color coding of the atoms

R12_max:=max(seq(R12[j],j=1..nPeaks_HOESY));
R12_min:=min(seq(R12[j],j=1..nPeaks_HOESY));
#R12_max:=0.25;
#R12_min:=0.15;

myt:=0;

min_exclusion_radius:=0.9+0.38;
radiusfactor:=min_exclusion_radius*R12[1];


R12_color:=[seq((R12[j]-R12_min)/(R12_max-R12_min), j=1..nPeaks_HOESY)];

display({
sphere(C_1,rayon,color=black),

sphere(Cb_1,rayon,color=black),
sphere(Cb_2,rayon,color=black),
sphere(Cb_3,rayon,color=black),
sphere(Cb_4,rayon,color=black),

sphere(Cp_1,rayon,color=black),
sphere(Cp_2,rayon,color=black),
sphere(Cp_3,rayon,color=black),


sphere(C_1,radiusfactor/R12[4],color=red,transparency=myt),

sphere(Cb_1,radiusfactor/R12[6],color=red,transparency=myt),
sphere(Cb_2,radiusfactor/R12[3],color=red,transparency=myt),
sphere(Cb_3,radiusfactor/R12[3],color=red,transparency=myt),
sphere(Cb_4,radiusfactor/R12[6],color=red,transparency=myt),

sphere(Cp_1,radiusfactor/R12[5],color=red,transparency=myt),
sphere(Cp_2,radiusfactor/R12[2],color=red,transparency=myt),
sphere(Cp_3,radiusfactor/R12[1],color=red,transparency=myt),

sphere(N1,rayon,color=white),

line(C_1,N1,color=black),
line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),
line(Cb_1,Cb_2,color=black),
line(Cb_2,Cb_3,color=black),
line(Cb_3,Cb_4,color=black),
line(Cp_1,Cp_2,color=black),
line(Cp_2,Cp_3,color=black)

},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr], axes=none);
> rayon_1:=radiusfactor/R12[4];
rayon_b1:=radiusfactor/R12[6];
rayon_b4:=rayon_b1;
```

```
rayon_b2:=radiusfactor/R12[3];
rayon_b3:=rayon_b2;
rayon_p1:=radiusfactor/R12[5];
rayon_p2:=radiusfactor/R12[2];
rayon_p3:=radiusfactor/R12[1];

display({
sphere(C_1,rayon,color="DimGray"),

sphere(Cb_1,rayon,color="DimGray"),
sphere(Cb_2,rayon,color="DimGray"),
sphere(Cb_3,rayon,color="DimGray"),
sphere(Cb_4,rayon,color="DimGray"),

sphere(Cp_1,rayon,color="DimGray"),
sphere(Cp_2,rayon,color="DimGray"),
sphere(Cp_3,rayon,color="DimGray"),

sphere(N1,rayon,color=grey),


sphere(Hm_1,rayon_1,color=RGB(R12_color[4],0,0.55-
R12_logcolor[4]),transparency=myt),
sphere(Hm_2,rayon_1,color=RGB(R12_color[4],0,0.55-
R12_logcolor[4]),transparency=myt),
sphere(Hm_3,rayon_1,color=RGB(R12_color[4],0,0.55-
R12_logcolor[4]),transparency=myt),

sphere(Hb_1,rayon_b1,color=RGB(R12_color[6],0,0.55-
R12_logcolor[6]),transparency=myt),
sphere(Hb_2,rayon_b1,color=RGB(R12_color[6],0,0.55-
R12_logcolor[6]),transparency=myt),

sphere(Hb_3,rayon_b2,color=RGB(R12_color[3],0,0.55-
R12_logcolor[3]),transparency=myt),
sphere(Hb_4,rayon_b2,color=RGB(R12_color[3],0,0.55-
R12_logcolor[3]),transparency=myt),


sphere(Hb_5,rayon_b3,color=RGB(R12_color[3],0,0.55-
R12_logcolor[3]),transparency=myt),
sphere(Hb_6,rayon_b3,color=RGB(R12_color[3],0,0.55-
R12_logcolor[3]),transparency=myt),


sphere(Hb_7,rayon_b4,color=RGB(R12_color[6],0,0.55-
R12_logcolor[6]),transparency=myt),
sphere(Hb_8,rayon_b4,color=RGB(R12_color[6],0,0.55-
R12_logcolor[6]),transparency=myt),

sphere(Hp_1,rayon_p1,color=RGB(R12_color[5],0,0.55-
R12_logcolor[5]),transparency=myt),
sphere(Hp_2,rayon_p1,color=RGB(R12_color[5],0,0.55-
R12_logcolor[5]),transparency=myt),
```

```
sphere(Hp_3,rayon_p2,color=RGB(R12_color[2],0,0.55-
R12_logcolor[2]),transparency=myt),
sphere(Hp_4,rayon_p2,color=RGB(R12_color[2],0,0.55-
R12_logcolor[2]),transparency=myt),

sphere(Hp_5,rayon_p3,color=RGB(R12_color[1],0,0.55-
R12_logcolor[1]),transparency=myt),
sphere(Hp_6,rayon_p3,color=RGB(R12_color[1],0,0.55-
R12_logcolor[1]),transparency=myt),
sphere(Hp_7,rayon_p3,color=RGB(R12_color[1],0,0.55-
R12_logcolor[1]),transparency=myt),

line(C_1,N1,color=black),
line(C_1,Hm_1,color=black),
line(C_1,Hm_2,color=black),
line(C_1,Hm_3,color=black),

line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),


line(Cb_1,Cb_2,color=black),

line(Cb_1,Hb_1,color=black),
line(Cb_1,Hb_2,color=black),

line(Cb_2,Cb_3,color=black),

line(Cb_2,Hb_3,color=black),
line(Cb_2,Hb_4,color=black),

line(Cb_3,Cb_4,color=black),

line(Cb_3,Hb_5,color=black),
line(Cb_3,Hb_6,color=black),
line(Cb_4,Hb_7,color=black),
line(Cb_4,Hb_8,color=black),

line(Cp_1,Cp_2,color=black),

line(Cp_1,Hp_1,color=black),
line(Cp_1,Hp_2,color=black),

line(Cp_2,Hp_3,color=black),
line(Cp_2,Hp_4,color=black),

line(Cp_2,Cp_3,color=black),
line(Cp_3,Hp_5,color=black),
line(Cp_3,Hp_6,color=black),
line(Cp_3,Hp_7,color=black)

},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr], axes=none);
> distance3d:=(A,B)->evalf(sqrt((A[1]-B[1])^2+(A[2]-B[2])^2+(A[3]-
B[3])^2));rayonexC:=1.1;
rayonexN:=1.8;
myt:=0.8;
```

```
plotResolution:=60;
myAtomRadiusList:=[[Hm_1,rayon_1],[Hm_2,rayon_1],[Hm_3,rayon_1],[Hb_1,rayon_b1
],[Hb_2,rayon_b1],[Hb_3,rayon_b2],[Hb_4,rayon_b2],[Hb_5,rayon_b3],[Hb_6,rayon_
b3],[Hb_7,rayon_b4],[Hb_8,rayon_b4],[Hp_1,rayon_p1],[Hp_2,rayon_p1],[Hp_3,rayo
n_p2],[Hp_4,rayon_p2],[Hp_5,rayon_p3],[Hp_6,rayon_p3],[Hp_7,rayon_p3],[Cb_1,ra
yonexC],[Cb_2,rayonexC],[Cb_3,rayonexC],[Cb_4,rayonexC],[N1,rayonexN],[Cp_1,ra
yonexC],[Cp_2,rayonexC],[Cp_3,rayonexC]];

surfaceFunction:=(x,y,z)->min({seq(distance3d([x,y,z],myAtomRadiusList[i][1])-
myAtomRadiusList[i][2],i=1..nops(myAtomRadiusList))});
mySurface:=implicitplot3d(surfaceFunction(x,y,z)=0,x=-vr..vr, y=-vr..vr,z=-
vr..vr,grid=[plotResolution,plotResolution,plotResolution],style=wireframe,col
or=grey,transparency=myt):

plotsetup(png, plotoutput = `plot.png`, plotoptions = ` width=4000,
height=4000`);

display({
sphere(C_1,rayon,color="DimGray"),

sphere(Cb_1,rayon,color="DimGray"),
sphere(Cb_2,rayon,color="DimGray"),
sphere(Cb_3,rayon,color="DimGray"),
sphere(Cb_4,rayon,color="DimGray"),

sphere(Cp_1,rayon,color="DimGray"),
sphere(Cp_2,rayon,color="DimGray"),
sphere(Cp_3,rayon,color="DimGray"),

sphere(N1,rayon,color=grey),


sphere(Hm_1,rayon_H,color=RGB(1,0.8,0)),
sphere(Hm_2,rayon_H,color=RGB(1,0.8,0)),
sphere(Hm_3,rayon_H,color=RGB(1,0.8,0)),

sphere(Hb_1,rayon_H,color=RGB(1,0.4,0)),
sphere(Hb_2,rayon_H,color=RGB(1,0.4,0)),

sphere(Hb_3,rayon_H,color=RGB(1,0.2,0)),
sphere(Hb_4,rayon_H,color=RGB(1,0.2,0)),

sphere(Hp_3,rayon_H,color=RGB(1,0.6,0)),
sphere(Hp_4,rayon_H,color=RGB(1,0.6,0)),

sphere(Hb_5,rayon_H,color=RGB(1,0.2,0)),
sphere(Hb_6,rayon_H,color=RGB(1,0.2,0)),

sphere(Hp_5,rayon_H,color=RGB(1,0,0)),
sphere(Hp_6,rayon_H,color=RGB(1,0,0)),
sphere(Hp_7,rayon_H,color=RGB(1,0,0)),

sphere(Hp_1,rayon_H,color=RGB(1,1,0)),
sphere(Hp_2,rayon_H,color=RGB(1,1,0)),

sphere(Hb_7,rayon_H,color=RGB(1,0.4,0)),
sphere(Hb_8,rayon_H,color=RGB(1,0.4,0)),
```

```
line(C_1,N1,color=black),
line(C_1,Hm_1,color=black),
line(C_1,Hm_2,color=black),
line(C_1,Hm_3,color=black),

line(N1,Cb_1,color=black),
line(N1,Cp_1,color=black),
line(N1,Cb_4,color=black),


line(Cb_1,Cb_2,color=black),

line(Cb_1,Hb_1,color=black),
line(Cb_1,Hb_2,color=black),

line(Cb_2,Cb_3,color=black),

line(Cb_2,Hb_3,color=black),
line(Cb_2,Hb_4,color=black),

line(Cb_3,Cb_4,color=black),

line(Cb_3,Hb_5,color=black),
line(Cb_3,Hb_6,color=black),
line(Cb_4,Hb_7,color=black),
line(Cb_4,Hb_8,color=black),

line(Cp_1,Cp_2,color=black),

line(Cp_1,Hp_1,color=black),
line(Cp_1,Hp_2,color=black),

line(Cp_2,Hp_3,color=black),
line(Cp_2,Hp_4,color=black),

line(Cp_2,Cp_3,color=black),
line(Cp_3,Hp_5,color=black),
line(Cp_3,Hp_6,color=black),
line(Cp_3,Hp_7,color=black),mySurface

},style=patchnogrid,view=[-vr..vr,-vr..vr,-vr..vr],
axes=none,orientation=[80,290]);


>
²Hydrogens:=[Hm_1,Hm_2,Hm_3,Hb_1,Hb_2,Hb_3,Hb_4,Hb_5,Hb_6,Hb_7,Hb_8,Hp_1,Hp_2,
Hp_3,Hp_4,Hp_5,Hp_6,Hp_7];
MinDistances:=[rayon_1,rayon_1,rayon_1,rayon_b1,rayon_b1,rayon_b2,rayon_b2,ray
on_b3,rayon_b3,rayon_b4,rayon_b4,rayon_p1,rayon_p1,rayon_p2,rayon_p2,rayon_p3,
rayon_p3,rayon_p3];

> #spectral density calculations
assume(w,positive);
JDDinter:=(w,D1,D2,distance)->72*1/distance^3*int( (u^2)/(81+9*u^2-
2*u^4+u^6)*(u^2*distance^2/(D1+D2))/(u^4+(w*distance^2/(D1+D2))^2),u=0..infini
ty);
```

```
> assume(d,positive);
assume(w,positive);

plot(JDDinter(w,1e-10,1e-10,1e-10),w=0..1e10);
> JDDinter(0,DLi,D1H,d);
series(Re(JDDinter(w,DLi,D1H,d)),d=0);

series(Re(0.1*JDDinter(w0-g_Li*w0/g_H,DLi,D1H,d)-
0.6*JDDinter(w0+g_Li*w0/g_H,DLi,D1H,d)),d=0);
freq1:=0;

w0:=500e6;

freq2:=2*Pi*w0;
freq3:=2*Pi*g_Li*w0/g_H;
freq4:=2*Pi*(w0-g_Li*w0/g_H);
freq5:=2*Pi*(w0+g_Li*w0/g_H);

dmin:=0.5e-10;
dmax:=1e-9;

plotJ0:=plot(Re(JDDinter(0,DLi,D1H,d)),d=dmin..dmax,color=blue):
plotJ1H:=plot(Re(JDDinter(freq2,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ7Li:=plot(Re(JDDinter(freq3,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ0HmLi:=plot(Re(JDDinter(freq4,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ0HpLi:=plot(Re(JDDinter(freq5,DLi,D1H,d)),d=dmin..dmax,color=red):
plotSigma:=plot(Re(-
0.1*JDDinter(freq4,DLi,D1H,d)+0.9*JDDinter(freq5,DLi,D1H,d)),d=dmin..dmax,colo
r=red):
display(plotJ0,plotJ1H,plotJ7Li,plotJ0HmLi,plotJ0HpLi);
display(plotSigma);
> JDDinter(0,DLi,D1H,d);
series(Re(JDDinter(w,DLi,D1H,d)),d=0);


freq1:=0;

w0:=500e6;

freq2:=2*Pi*w0;
freq3:=2*Pi*g_Li*w0/g_H;
freq4:=2*Pi*(w0-g_Li*w0/g_H);
freq5:=2*Pi*(w0+g_Li*w0/g_H);

dmin:=1.2e-10;
dmax:=8e-10;

plotJ0:=plot(Re(JDDinter(0,DLi,D1H,d)),d=dmin..dmax,color=blue):
plotJ1H:=plot(Re(JDDinter(freq2,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ7Li:=plot(Re(JDDinter(freq3,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ0HmLi:=plot(Re(JDDinter(freq4,DLi,D1H,d)),d=dmin..dmax,color=purple):
plotJ0HpLi:=plot(Re(JDDinter(freq5,DLi,D1H,d)),d=dmin..dmax,color=red):

plotSigma1M:=plot(Re(-
0.1*JDDinter(freq4,DLi,D1H,d)+0.9*JDDinter(freq5,DLi,D1H,d)),d=dmin..dmax,colo
r=red);
```

```
plotPho1M:=plot(Re(0.1*JDDinter(freq4,DLi,D1H,d)+0.3*JDDinter(freq2,DLi,D1H,d)
+0.6*JDDinter(freq5,DLi,D1H,d)),d=dmin..dmax,color=red):
plotNOE1M:=plot(Re((-
0.1*JDDinter(freq4,DLi,D1H,d)+0.6*JDDinter(freq5,DLi,D1H,d))/(0.1*JDDinter(fre
q4,DLi,D1H,d)+0.3*JDDinter(freq2,DLi,D1H,d)+0.6*JDDinter(freq5,DLi,D1H,d))),d=
dmin..dmax,color=grey);
DLi3M2:=9.58E-12;
D1H3M2:=1.36E-11;
plotSigma3M2:=plot(Re(-
0.1*JDDinter(freq4,DLi3M2,D1H3M2,d)+0.9*JDDinter(freq5,DLi3M2,D1H3M2,d)),d=dmi
n..dmax,color=purple):
plotNOE3M2:=plot(Re((-
0.1*JDDinter(freq4,DLi3M2,D1H3M2,d)+0.6*JDDinter(freq5,DLi3M2,D1H3M2,d))/(0.1*
JDDinter(freq4,DLi,D1H,d)+0.3*JDDinter(freq2,DLi3M2,D1H3M2,d)+0.6*JDDinter(fre
q5,DLi3M2,D1H3M2,d))),d=dmin..dmax,color=black);
plotRapport:=plot((Re(-
0.1*JDDinter(freq4,DLi,D1H,d)+0.6*JDDinter(freq5,DLi3M2,D1H3M2,d)))/(Re(-
0.1*JDDinter(freq4,DLi,D1H,d)+0.9*JDDinter(freq5,DLi,D1H,d))),d=dmin..dmax,col
or=red):
display(plotSigma3M2,plotSigma1M);
display(plotNOE3M2,plotNOE1M):
display(plotRapport):

series(Re(0.6*JDDinter(w0+g_Li*w0/g_H,DLi,D1H,d)-0.1*JDDinter(w0-
g_Li*w0/g_H,DLi,D1H,d)),d=0);

series(Re(0.6*JDDinter(w0+g_Li*w0/g_H,DLi3M2,D1H3M2,d)-
0.1*JDDinter(w0+g_Li*w0/g_H,DLi3M2,D1H3M2,d)),d=0);
```