# RBMRB : A package to download and visualize BMRB data in R

*Kumaran Baskaran*

*November 08, 2016*

## RBMRB

BMRB collects, annotates, archives, and disseminates (worldwide in the public domain) the important spectral and quantitative data derived from NMR spectroscopic investigations of biological macromolecules and metabolites. The goal is to empower scientists in their analysis of the structure, dynamics, and chemistry of biological systems and to support further development of the field of biomolecular NMR spectroscopy

RBMRB is a library to fetch NMR chemical shift data directly from BMRB into R environment as a data frame in R. This facilitates access to BMRB data for statistical analysis and data visualization. It is using the BMRB-API to fetch the data from BMRB database.

### Installation

RBMRB library has been developed and tested in R version 3.3.x. It requires the following R packages preinstalled

- **httr** to import data from BMRB web server(version 1.2.1 or later)
- **data.table** to format the imported data into a data frame in R (version 1.9.6 or later)
- **rjson** to deal with BMRB-API (version 0.2.15 or later)
- **ggplot2** to simulate spectra (version 2.1.0 or later)
- **plotly** for interactive graphics in simulated spectra (version 4.5.2 or later)

Users should make sure that the above packages have beed installed correctly with the required versions, before proceeding to RMBRM insallation.

Here is the instruction to install those packages. Open your R and use the following command

```
installed.packages(c("httr","data.table","rjson","ggplot2","plotly"))
```

#### Method 1

Once the necessary packages have been installed, proceed with RMRBM installation. The source file can be downloaded from GitHub

After downloading the source file, use the following command to install RBMRB library

```
install.packages("~/Downloads/RBMRB_2.0.tar.gz",repos=NULL,type="source")
```

Note: provide the correct path to the downloaded file.

#### Method 2

If you have devtools library in your R, then you can install directly from GitHub.

```
library(devtools)
install_github("uwbmrb/RBMRB/RBMRB")
```

## Usage

RBMRB can be used in a similar way like any other library in R.

```
library(RBMRB)
```

## Data access

BMRB data can be imported in two ways

- **Entry method** Chemical shift data from single or multiple entries
- **Atom method** Chemical shift data from all entries for a given atom

### Entry method

#### fetch_entry_chemical_shifts:

This function will fetch the 'Atom_chem_shift' loop from a NMR-STAR file for a given entry or a list of entries in CSV format. This function works on both macromolecules and metabolites data base. For metabilites entry ids should have right prefix (example 'bmse000034')

Examples:

```
df1<-fetch_entry_chemical_shifts(15060)
df2<-fetch_entry_chemical_shifts(c(17074,17076,17077))
df2<-fetch_entry_chemical_shifts(c('17074','17076','17077'))
df3<-fetch_entry_chemical_shifts(c('bmse000034','bmse000035','bmse000036'))
```

These data frames have the following columns

```
colnames(df1)
```

```
##  [1] "ID"                          "Assembly_atom_ID"
##  [3] "Entity_assembly_ID"          "Entity_ID"
##  [5] "Comp_index_ID"               "Seq_ID"
##  [7] "Comp_ID"                     "Atom_ID"
##  [9] "Atom_type"                   "Atom_isotope_number"
## [11] "Val"                         "Val_err"
## [13] "Assign_fig_of_merit"         "Ambiguity_code"
## [15] "Occupancy"                   "Resonance_ID"
## [17] "Auth_entity_assembly_ID"     "Auth_asym_ID"
## [19] "Auth_seq_ID"                 "Auth_comp_ID"
## [21] "Auth_atom_ID"                "Details"
## [23] "Entry_ID"                    "Assigned_chem_shift_list_ID"
```

Sample data output

```
head(df1)
```

```
##   ID Assembly_atom_ID Entity_assembly_ID Entity_ID Comp_index_ID Seq_ID
## 1  1                .                  1         1            20     20
## 2  2                .                  1         1            20     20
## 3  3                .                  1         1            20     20
## 4  4                .                  1         1            20     20
## 5  5                .                  1         1            21     21
## 6  6                .                  1         1            21     21
##   Comp_ID Atom_ID Atom_type Atom_isotope_number     Val Val_err
```

```
## 1     LEU       H         H                  1   8.149      NA
## 2     LEU      CA         C                 13  56.016      NA
## 3     LEU      CB         C                 13  42.180      NA
## 4     LEU       N         N                 15 122.739      NA
## 5     VAL       H         H                  1   8.048      NA
## 6     VAL      CA         C                 13  63.412      NA
##   Assign_fig_of_merit Ambiguity_code Occupancy Resonance_ID
## 1                   .              1         .            .
## 2                   .              1         .            .
## 3                   .              1         .            .
## 4                   .              1         .            .
## 5                   .              1         .            .
## 6                   .              1         .            .
##   Auth_entity_assembly_ID Auth_asym_ID Auth_seq_ID Auth_comp_ID
## 1                       .            .          20          LEU
## 2                       .            .          20          LEU
## 3                       .            .          20          LEU
## 4                       .            .          20          LEU
## 5                       .            .          21          VAL
## 6                       .            .          21          VAL
##   Auth_atom_ID Details Entry_ID Assigned_chem_shift_list_ID
## 1           HN       .    15060                          1
## 2           CA       .    15060                          1
## 3           CB       .    15060                          1
## 4            N       .    15060                          1
## 5           HN       .    15060                          1
## 6           CA       .    15060                          1
```

### Atom method

#### fetch_atom_chemical_shifts:

This function will fetch the chemical shift data from all the entries for a given atom. The atom name should be in NMR-STAR atom nomenclature.

Examples:

```
df4<-fetch_atom_chemical_shifts('CG2')
df5<-fetch_atom_chemical_shifts('C9')
```

These data frames have the following columns

```
colnames(df4)
```

```
##  [1] "Entry_ID"                   "Entity_ID"
##  [3] "Comp_index_ID"              "Comp_ID"
##  [5] "Atom_ID"                    "Atom_type"
##  [7] "Val"                        "Val_err"
##  [9] "Ambiguity_code"             "Assigned_chem_shift_list_ID"
```

Sample data output

```
head(df4)
```

```
##   Entry_ID Entity_ID Comp_index_ID Comp_ID Atom_ID Atom_type    Val
## 1    10001         1             1     ILE     CG2         C 15.700
## 2    10001         1             6     ILE     CG2         C 17.900
```

```
## 3    10002         1             3     ILE     CG2         C 17.516
## 4    10002         1            18     VAL     CG2         C 22.278
## 5    10002         1            19     THR     CG2         C 21.957
## 6    10002         1            26     THR     CG2         C 21.779
##   Val_err Ambiguity_code Assigned_chem_shift_list_ID
## 1     0.4              1                           1
## 2     0.3              1                           1
## 3     0.4              1                           1
## 4     0.4              1                           1
## 5     0.4              1                           1
## 6     0.4              1                           1
```

## Data manipulation

There are few data manipulation functions are availbale to facilitate plotting.

### convert_cs_to_n15hsqc:

This function will reformat the chemical shift data frame into a data frame which is easy to plot the N15-HSQC spectrum from the data.

### Examples

```
n15hsqc1<-convert_cs_to_n15hsqc(df1)
n15hsqc2<-convert_cs_to_n15hsqc(df2)
```

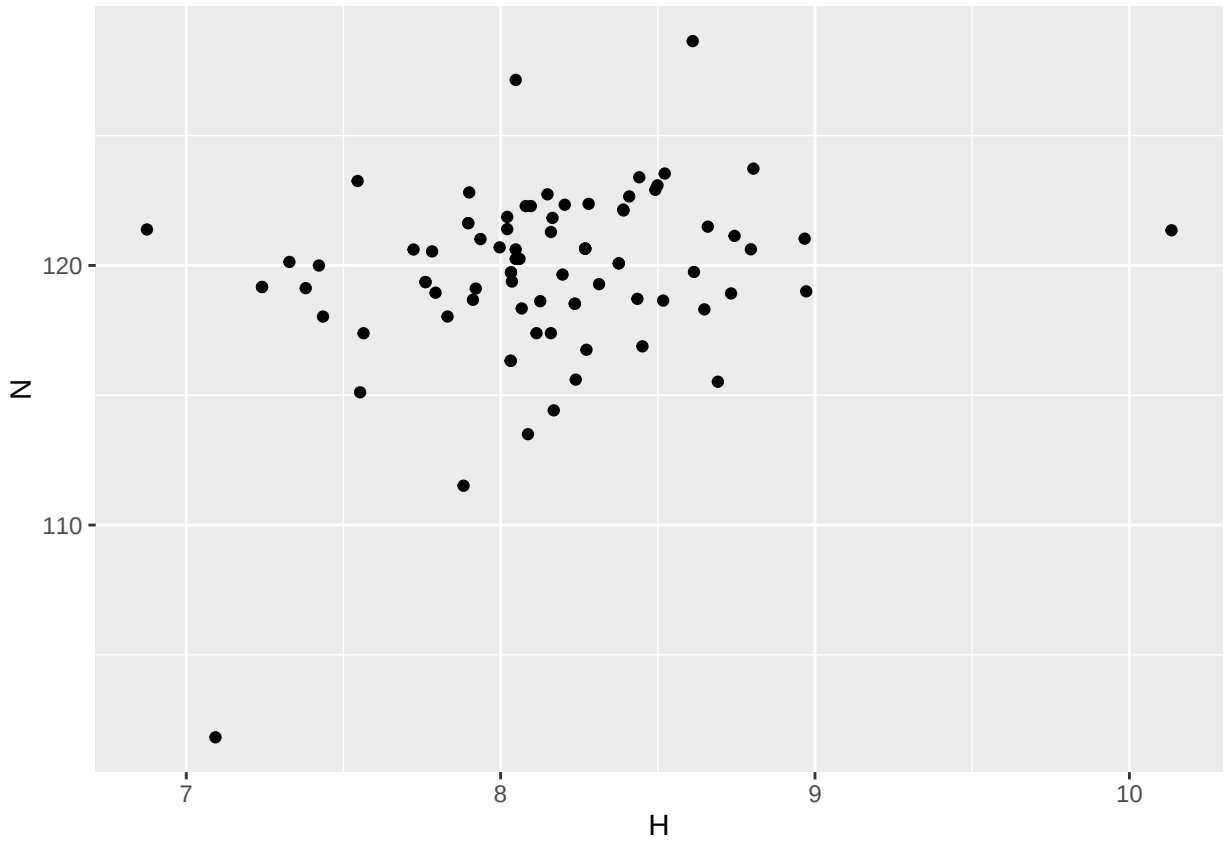
The output data frame will look like

```
head(n15hsqc1)
```

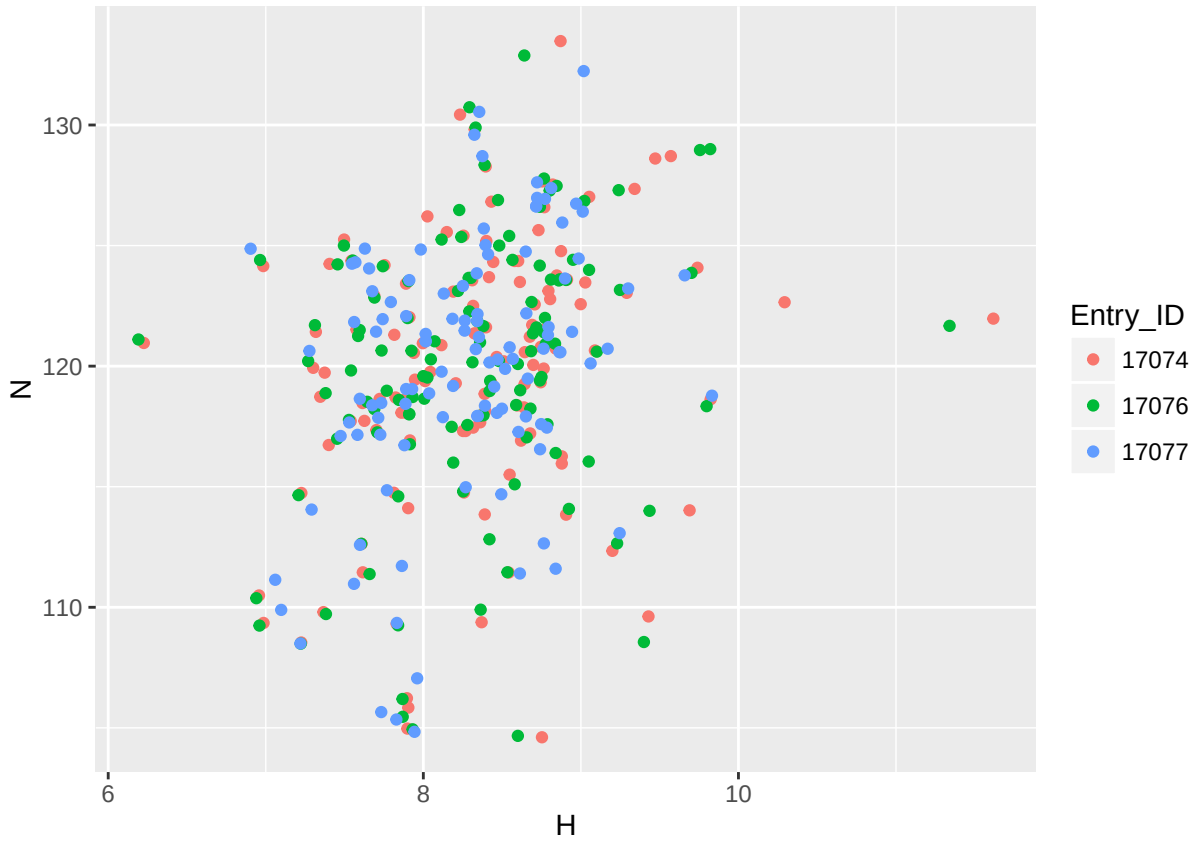
```
##   Entry_ID Comp_index_ID Entity_ID Assigned_chem_shift_list_ID Comp_ID_H
## 1    15060           101         1                           1       ASP
## 2    15060           102         1                           1       ASP
## 3    15060           103         1                           1       SER
## 4    15060           104         1                           1       ASP
## 5    15060           105         1                           1       GLU
## 6    15060           106         1                           1       GLU
##   Comp_ID_N     H       N
## 1       ASP 8.269 120.647
## 2       ASP 8.376 120.080
## 3       SER 8.239 115.602
## 4       ASP 8.409 122.658
## 5       GLU 8.269 120.647
## 6       GLU 8.391 122.119
```

This data frame is easy to plot using any plotting library

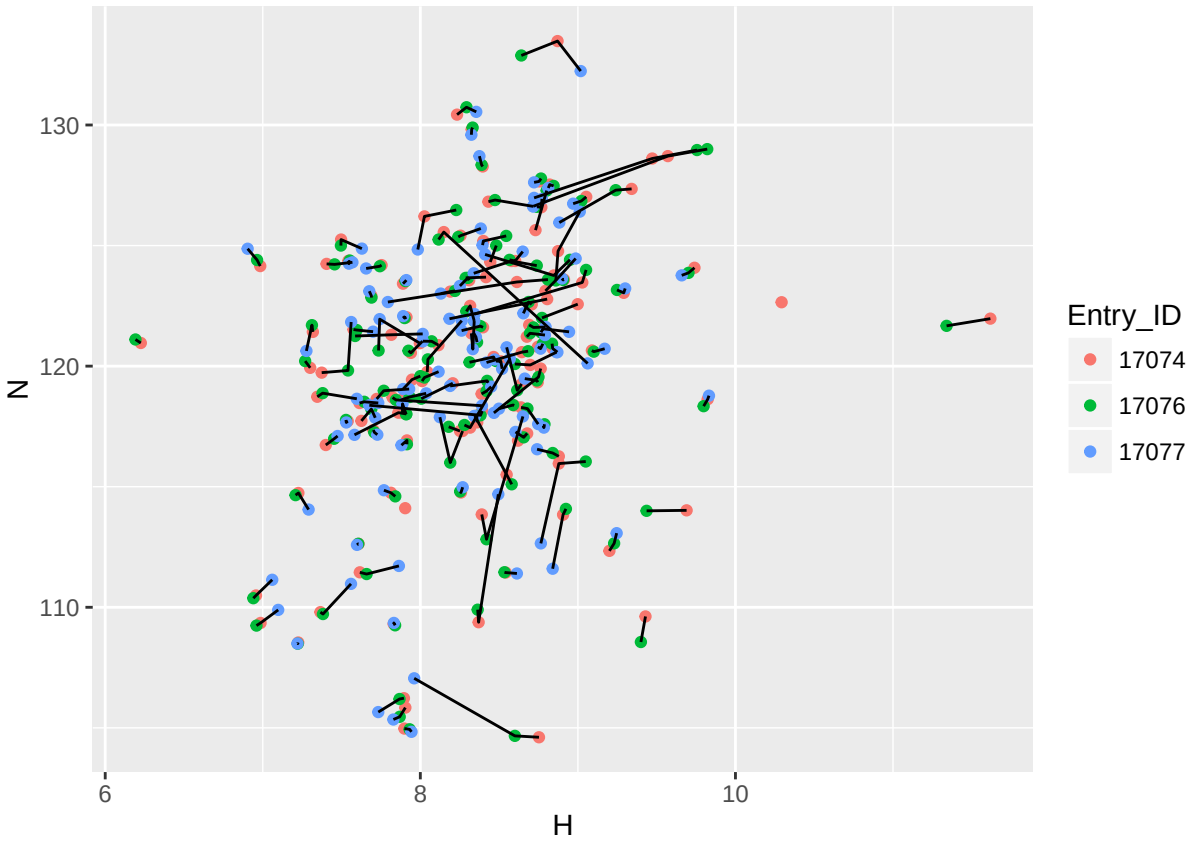
```
library(ggplot2)
plt1<-ggplot(n15hsqc1)+geom_point(aes(x=H,y=N))
plt1
```




```
plt2<-ggplot(n15hsqc2)+geom_point(aes(x=H,y=N,color=Entry_ID))
plt2
```




```
plt3<-ggplot(n15hsqc2)+geom_point(aes(x=H,y=N,color=Entry_ID))+geom_line(aes(x=H,y=N,group=Comp_index_I
plt3
```




**convert_cs_to_c13hsqc:**

This function will reformat the chemical shift data frame into a data frame which is easy to plot the C13-HSQC spectrum from the data.

**Examples**

```
c13hsqc1<-convert_cs_to_c13hsqc(df1)
c13hsqc2<-convert_cs_to_c13hsqc(df2)
```

The output data frame will look like

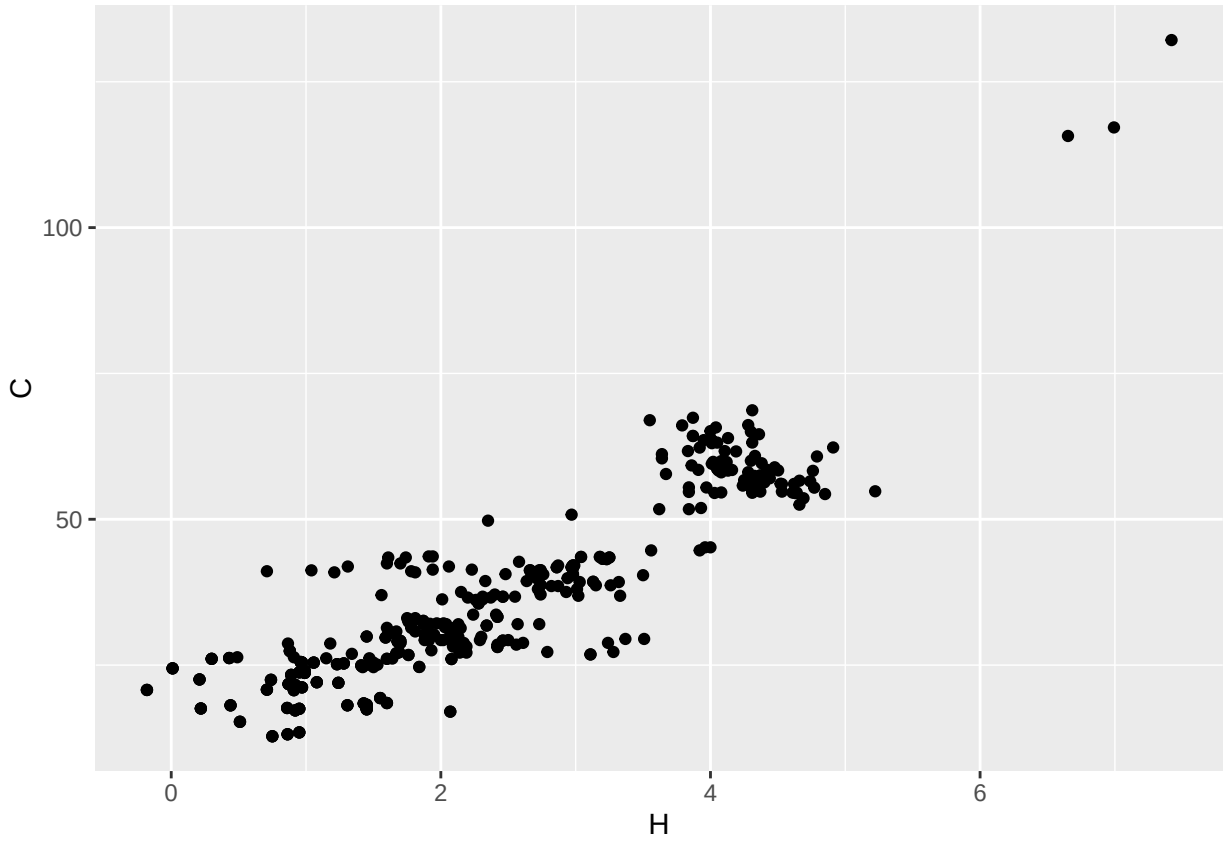
```
head(c13hsqc1)
```

```
##   Entry_ID Comp_index_ID Entity_ID Assigned_chem_shift_list_ID Comp_ID_C
## 1    15060           101         1                           1       ASP
## 2    15060           102         1                           1       ASP
## 3    15060           103         1                           1       SER
## 4    15060           104         1                           1       ASP
## 5    15060           105         1                           1       GLU
## 6    15060           106         1                           1       GLU
##   Comp_ID_H Atom_ID_C Atom_ID_H      C     H
## 1       ASP        CA        HA 54.487 4.630
## 2       ASP        CA        HA 54.572 4.609
## 3       SER        CA        HA 58.470 4.420
## 4       ASP        CA        HA 54.567 4.640
## 5       GLU        CA        HA 56.521 4.271
```

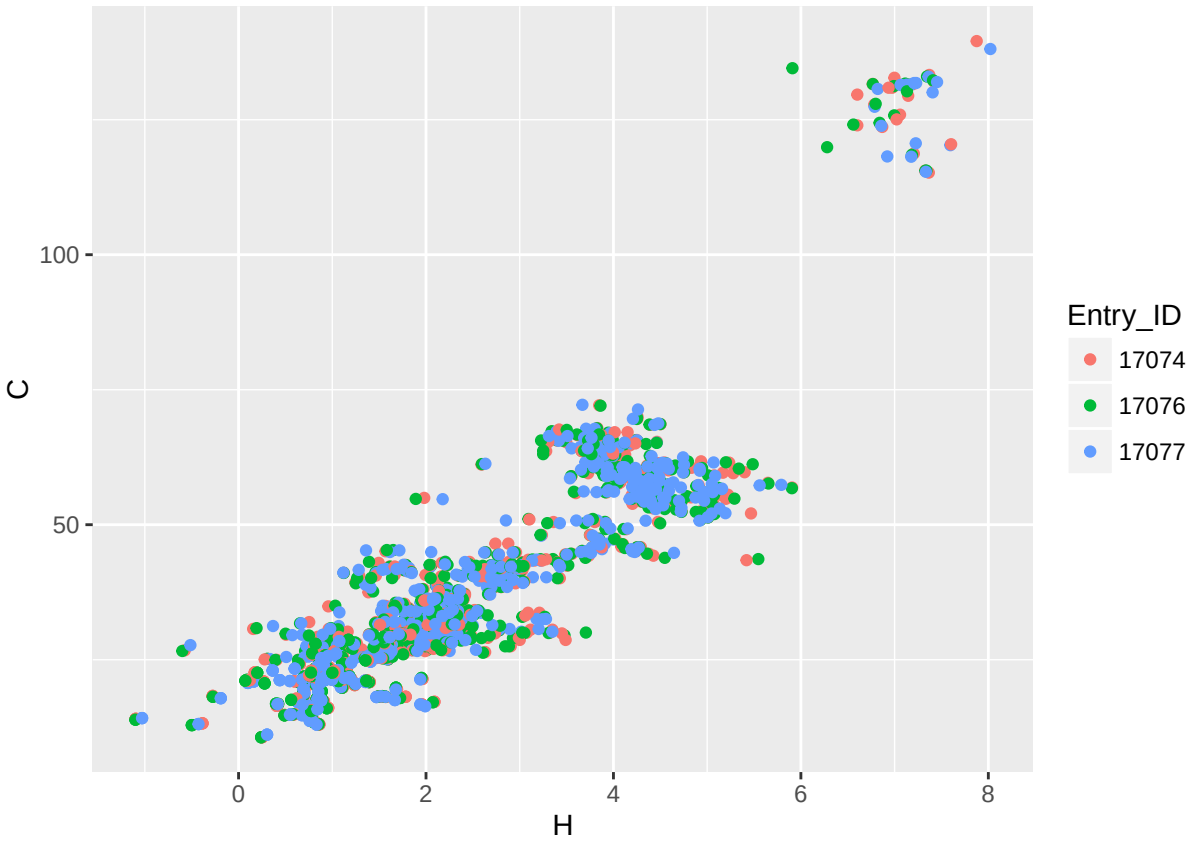
```
## 6       GLU        CA        HA 56.400 4.300
```

and the user may generate a spectrum using the following script

```
library(ggplot2)
plt1<-ggplot(c13hsqc1)+geom_point(aes(x=H,y=C))
plt1
```




```
plt2<-ggplot(c13hsqc2)+geom_point(aes(x=H,y=C,color=Entry_ID))
plt2
```




**convert__cs__to__tocsy:**

This function will reformat the chemical shift data frame into a data frame which is easy to plot the TOCSY spectrum from the data. Note : Since both dimensions have protein chemical shifts, the columns are named as Val.x and Val.y

### Examples

```
tocsy1<-convert_cs_to_tocsy(df1)
tocsy2<-convert_cs_to_tocsy(df2)
```

after conversion the data will look like

```
head(tocsy1)
```

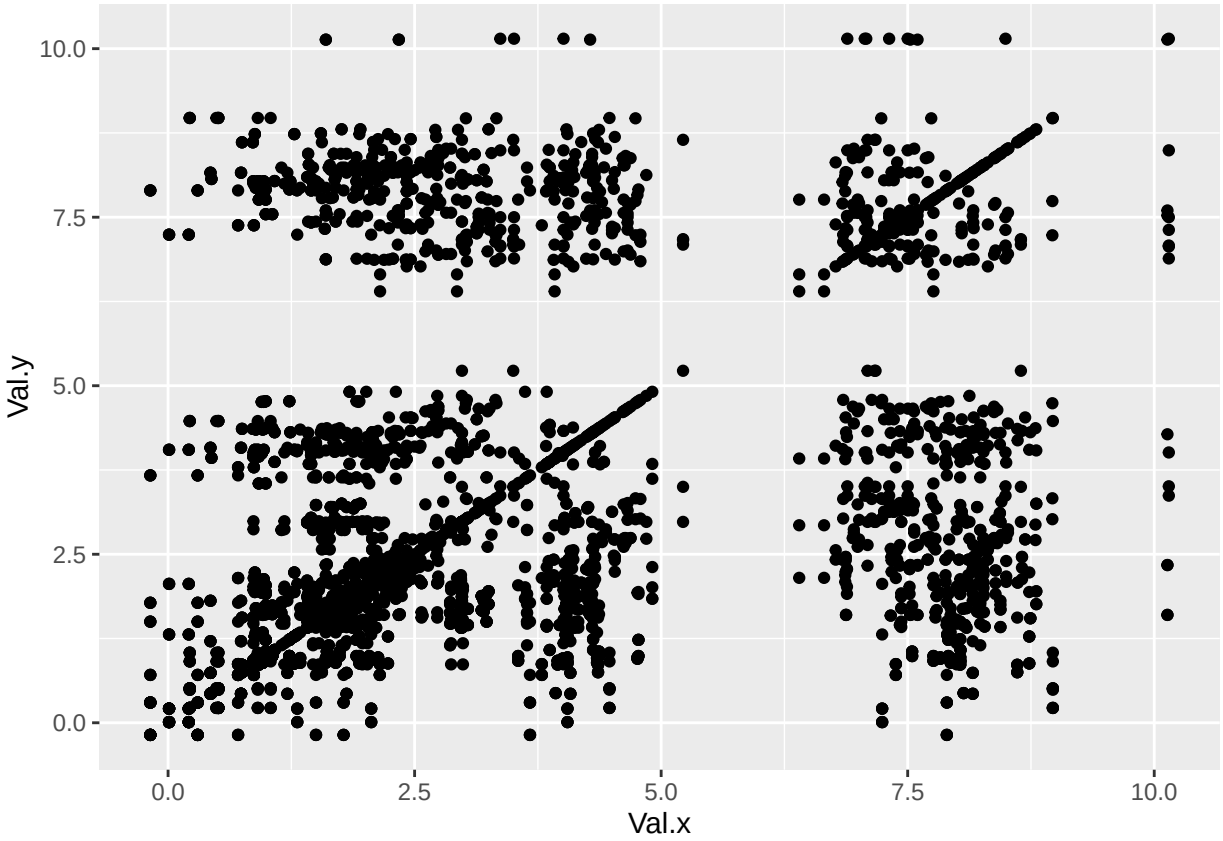
```
##   Entry_ID Entity_ID Comp_index_ID Assigned_chem_shift_list_ID ID.x
## 1    15060         1            100                           1  915
## 2    15060         1            100                           1  915
## 3    15060         1            100                           1  916
## 4    15060         1            100                           1  916
## 5    15060         1            101                           1  919
## 6    15060         1            101                           1  919
##   Assembly_atom_ID.x Entity_assembly_ID.x Seq_ID.x Comp_ID.x Atom_ID.x
## 1                  .                    1      100       GLY       HA2
## 2                  .                    1      100       GLY       HA2
## 3                  .                    1      100       GLY       HA3
## 4                  .                    1      100       GLY       HA3
```

```
## 5                  .                    1      101       ASP         H
## 6                  .                    1      101       ASP         H
##   Atom_type.x Atom_isotope_number.x Val.x Val_err.x Assign_fig_of_merit.x
## 1           H                     1 3.960        NA                     .
## 2           H                     1 3.960        NA                     .
## 3           H                     1 4.000        NA                     .
## 4           H                     1 4.000        NA                     .
## 5           H                     1 8.269        NA                     .
## 6           H                     1 8.269        NA                     .
##   Ambiguity_code.x Occupancy.x Resonance_ID.x Auth_entity_assembly_ID.x
## 1                2           .              .                         .
## 2                2           .              .                         .
## 3                2           .              .                         .
## 4                2           .              .                         .
## 5                1           .              .                         .
## 6                1           .              .                         .
##   Auth_asym_ID.x Auth_seq_ID.x Auth_comp_ID.x Auth_atom_ID.x Details.x
## 1              .           100            GLY            HA1         .
## 2              .           100            GLY            HA1         .
## 3              .           100            GLY            HA2         .
## 4              .           100            GLY            HA2         .
## 5              .           101            ASP             HN         .
## 6              .           101            ASP             HN         .
##   ID.y Assembly_atom_ID.y Entity_assembly_ID.y Seq_ID.y Comp_ID.y
## 1  915                  .                    1      100       GLY
## 2  916                  .                    1      100       GLY
## 3  915                  .                    1      100       GLY
## 4  916                  .                    1      100       GLY
## 5  919                  .                    1      101       ASP
## 6  920                  .                    1      101       ASP
##   Atom_ID.y Atom_type.y Atom_isotope_number.y Val.y Val_err.y
## 1       HA2          H                     1 3.960        NA
## 2       HA3          H                     1 4.000        NA
## 3       HA2          H                     1 3.960        NA
## 4       HA3          H                     1 4.000        NA
## 5         H          H                     1 8.269        NA
## 6        HA          H                     1 4.630        NA
##   Assign_fig_of_merit.y Ambiguity_code.y Occupancy.y Resonance_ID.y
## 1                     .                2           .              .
## 2                     .                2           .              .
## 3                     .                2           .              .
## 4                     .                2           .              .
## 5                     .                1           .              .
## 6                     .                1           .              .
##   Auth_entity_assembly_ID.y Auth_asym_ID.y Auth_seq_ID.y Auth_comp_ID.y
## 1                         .              .           100            GLY
## 2                         .              .           100            GLY
## 3                         .              .           100            GLY
## 4                         .              .           100            GLY
## 5                         .              .           101            ASP
## 6                         .              .           101            ASP
##   Auth_atom_ID.y Details.y
## 1            HA1         .
## 2            HA2         .
```

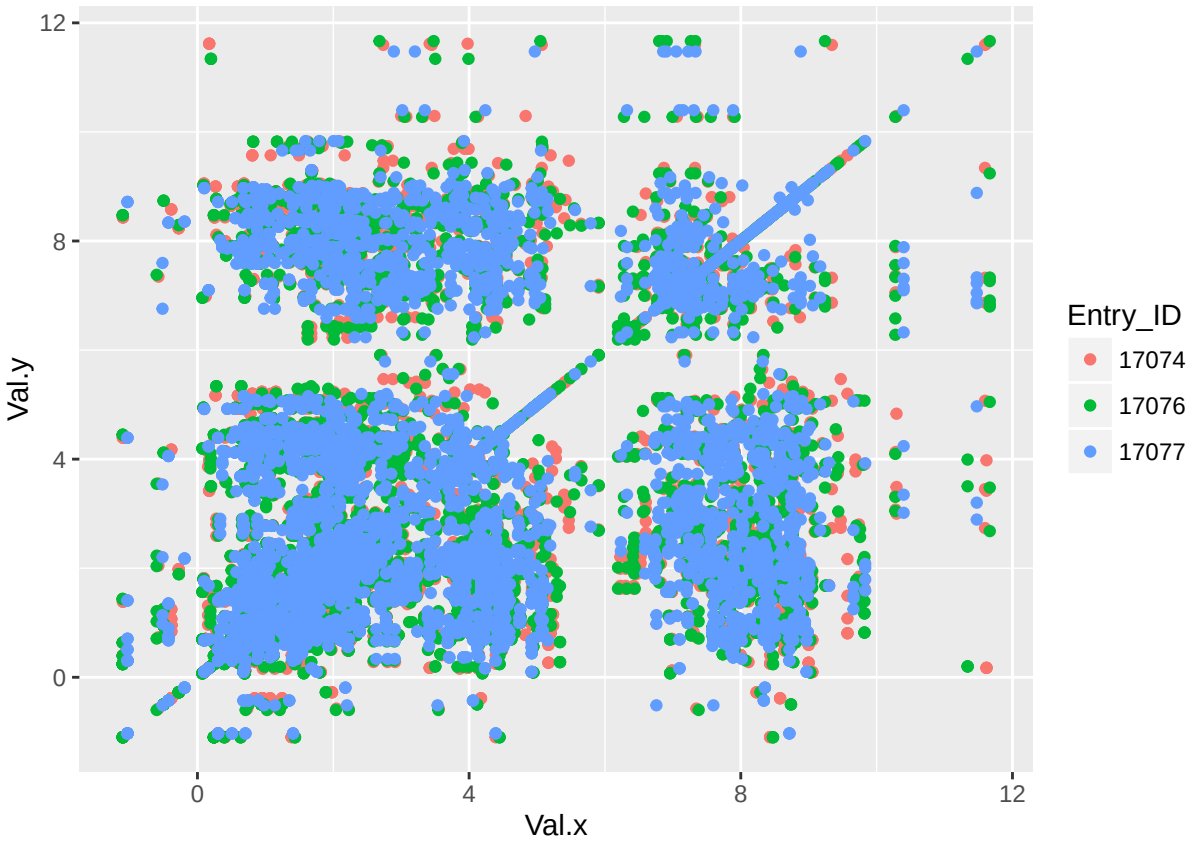
```
## 3            HA1         .
## 4            HA2         .
## 5             HN         .
## 6             HA         .
```

Plotting TOCSY spectrum

```
library(ggplot2)
plt1<-ggplot(tocsy1)+geom_point(aes(x=Val.x,y=Val.y))
plt1
```




```
plt2<-ggplot(tocsy2)+geom_point(aes(x=Val.x,y=Val.y,color=Entry_ID))
plt2
```




**filter_residue:**

This function will filter the data frame and remove all non standard amino acids. The data frame should contain the amino acid information in the Comp_ID column. ####Examples

```
df6<-fetch_atom_chemical_shifts('CG2')
df7<-filter_residue(df6)
```

## Data visualization

RBMRB library contains few functions to generate interactive visualization of BMRB data with out any data manipulation. The interactive visualizations use **plotly** library. If user has problem with plotly, then this feature may be diabled by providing an argument 'interactive=FALSE' for these functions. These interactive plots can be zoomed in and out using a mouse and will show tooltip information when you mouse over. These visualizations can be exported as a stand alone html file

### HSQC_15N

This function will simulae N15-HSQC spectrum for a given entry or list of entries.

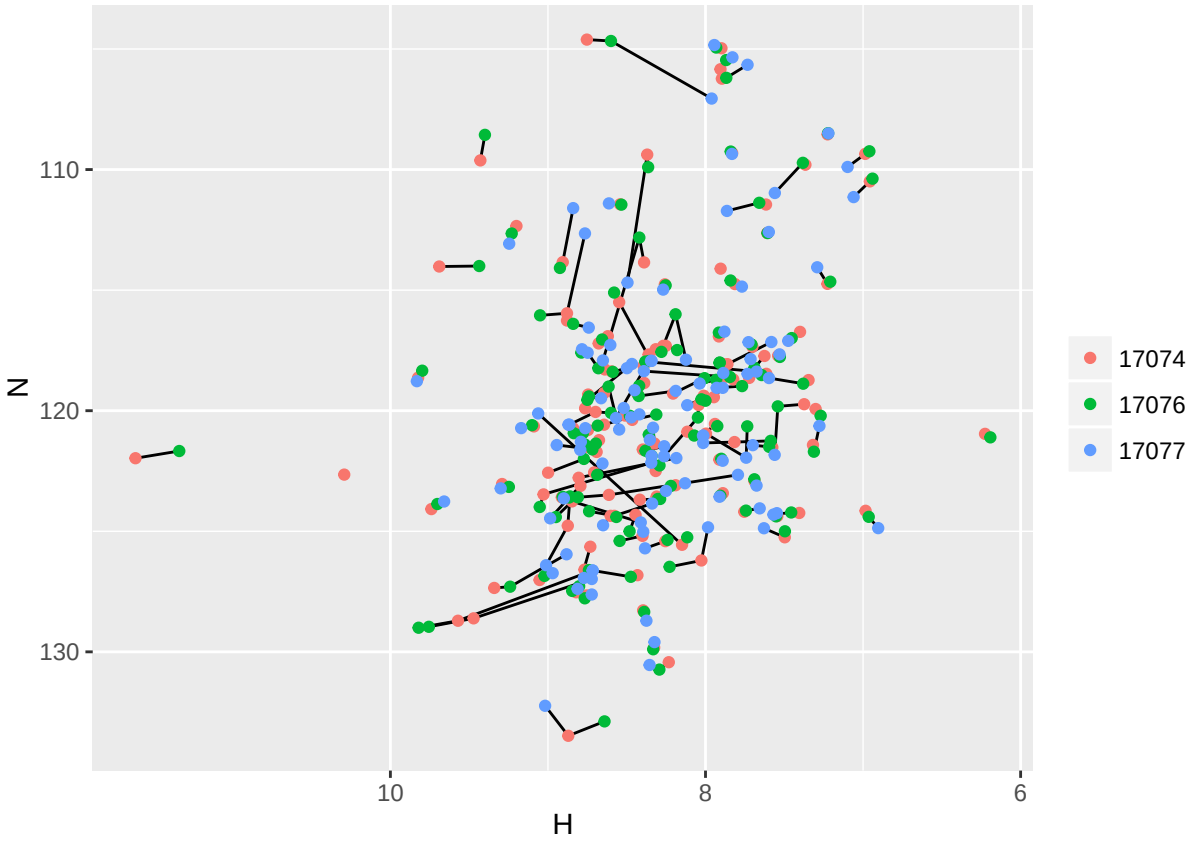
### Examples

These interactive visualization can be exported as single stand alone html file

```
spec1<-HSQC_15N(15060)
spec1
```

```
spec2<-HSQC_15N(c(17074,17076,17077),type='line')
spec2
```

```
spec3<-HSQC_15N(c(17074,17076,17077),type='line',interactive = F)
spec3
```




### HSQC_13C

This function will simulae C13-HSQC spectrum for a given entry or list of entries.

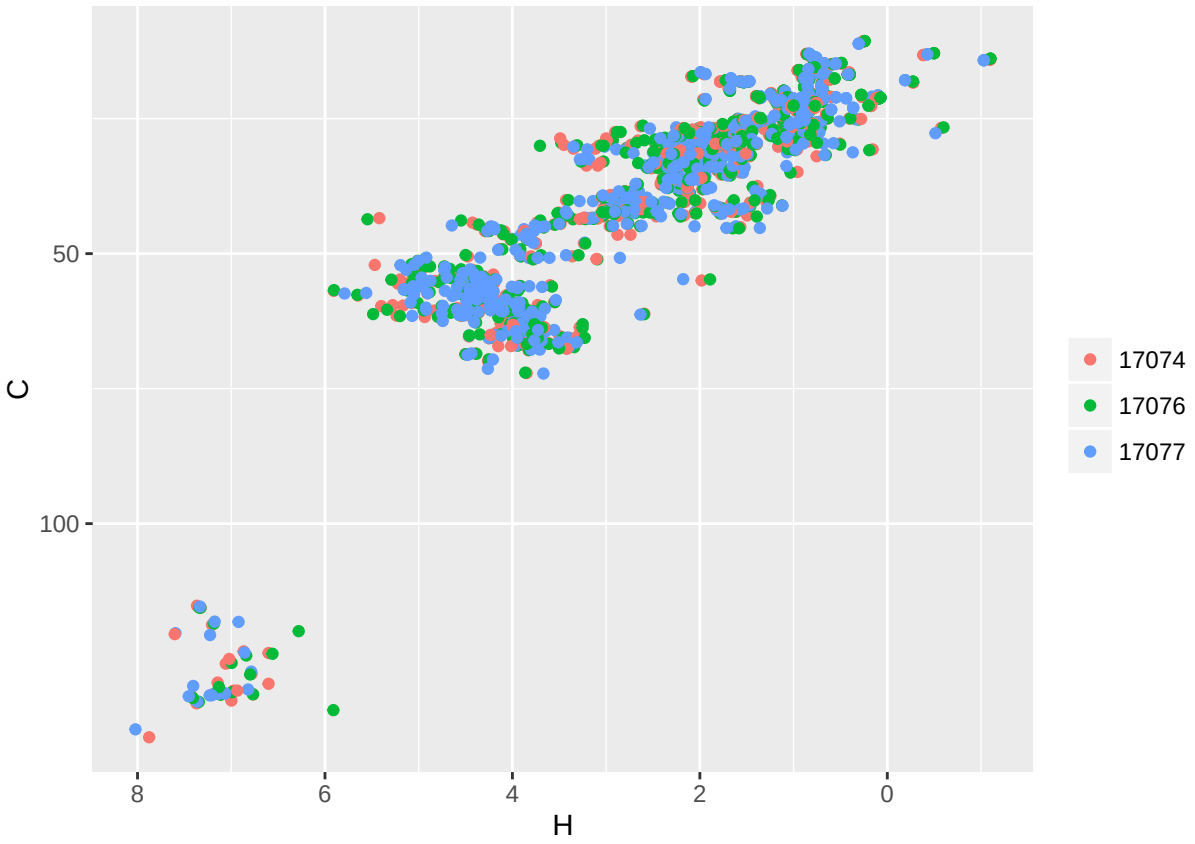
### Examples

These interactive visualization can be exported as single stand alone html file

```
spec1<-HSQC_13C(15060)
spec1
spec2<-HSQC_13C(c(17074,17076,17077))
spec2
```

Non interactive plot

```
spec3<-HSQC_13C(c(17074,17076,17077),interactive = F)
spec3
```




### TOCSY

This function will simulae TOCSY spectrum for a given entry or list of entries.

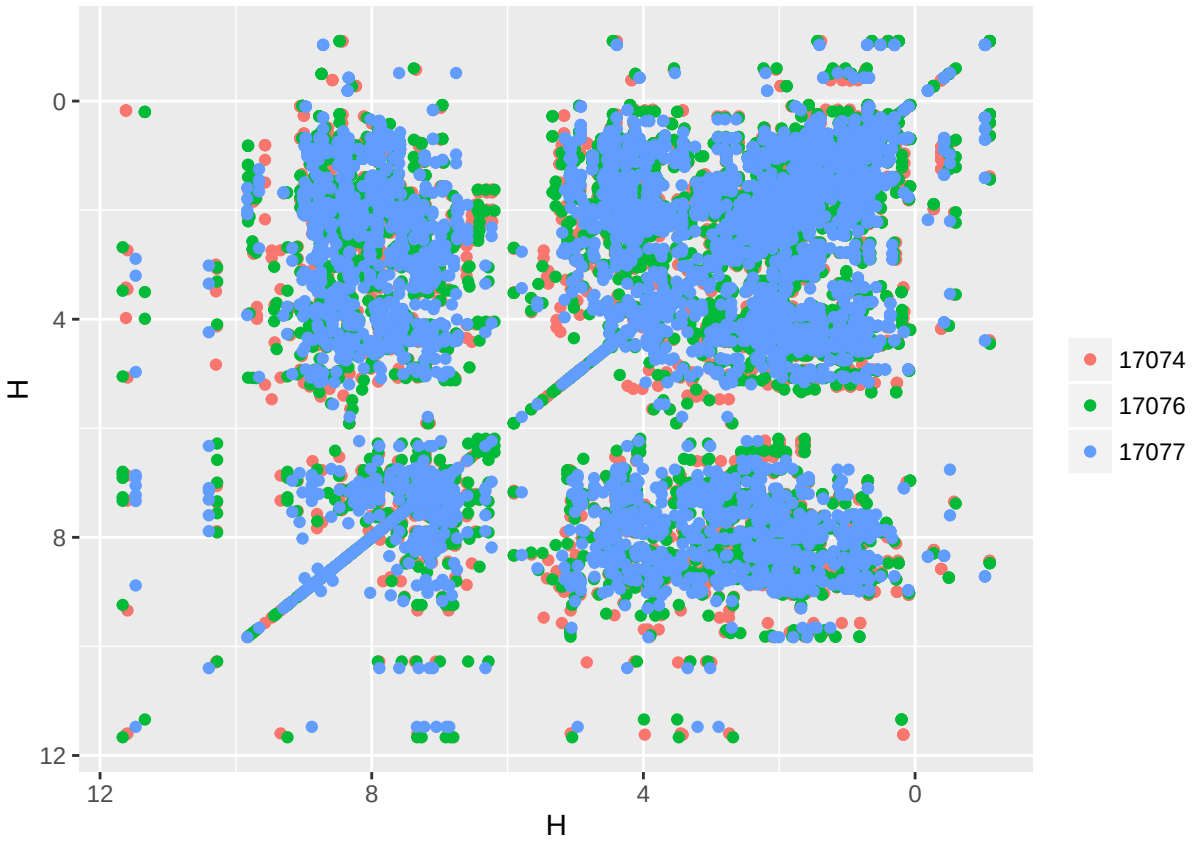
### Examples

These interactive visualization can be exported as single stand alone html file

```
spec1<-TOCSY(15060)
spec1
spec2<-TOCSY(c(17074,17076,17077))
spec2
```

Non interactive plot

```
spec3<-TOCSY(c(17074,17076,17077),interactive = F)
spec3
```



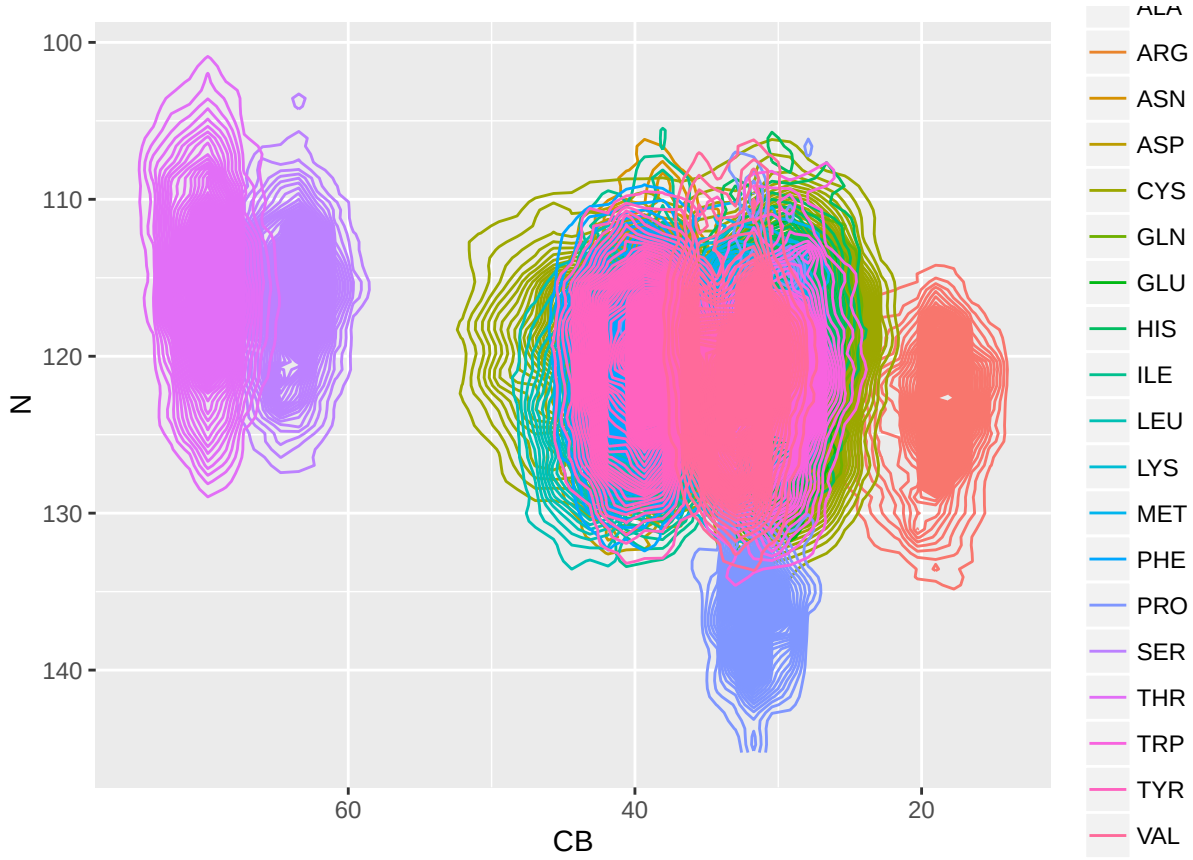

**chemical_shift_corr**

This function will plot the distribution of checmial shift correlation between any two atoms from the 20 standard amino acids. The distribution of a particular residue may turn on and off by clicking the residue name in the legand.

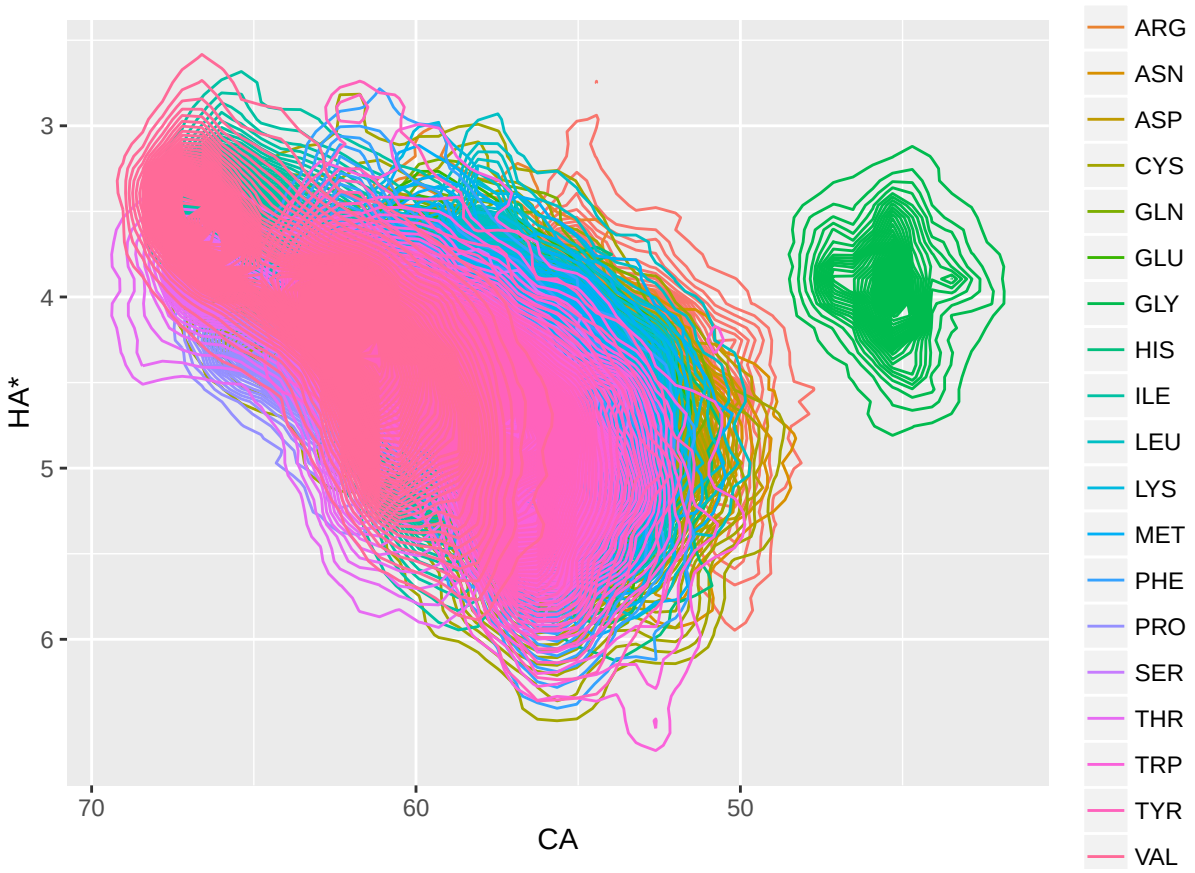
```
corr_plot1<-chemical_shift_corr('CB','N')
corr_plot1
corr_plot2<-chemical_shift_corr('CA','HA*')
corr_plot2
```

Non interactive plot

```
corr_plot1<-chemical_shift_corr('CB','N',interactive = F)
corr_plot1
```




```
corr_plot2<-chemical_shift_corr('CA','HA*',interactive = F)
corr_plot2
```


HA*
3
4
5
6
70
60
50
CA
ARG
ASN
ASP
CYS
GLN
GLU
GLY
HIS
ILE
LEU
LYS
MET
PHE
PRO
SER
THR
TRP
TYR
VAL