**# 👨‍💻 Guide Programmeur – Application Shiny : 2D NMR Spectra Analysi**s

**## 📅 Dernière mise à jour**2025-05-28

---

**## 📁 Structure du projet**

```

2DNMR-Analyst/

├── Shine.R # Application principale Shiny

├── README.md # Documentation utilisateur

├── Function\_test/

│ ├── Read\_2DNMR\_spectrum.R # Lecture des données Bruker 2D

│ ├── Vizualisation.R # Génération des graphiques 2D

│ └── Integration.R # Détection des pics et intégration

```

---

**Objectif**

Ce document s’adresse aux développeurs souhaitant comprendre, modifier ou étendre l’application Shiny 2D NMR Spectra Analysis. L’application permet de visualiser, analyser et exporter les données RMN 2D issues de spectres Bruker.

**1. Architecture générale**

L'application est définie dans un unique fichier Shine.R, avec des fonctions déportées dans trois fichiers externes localisés dans Function\_test/ :

* Read\_2DNMR\_spectrum.R : fonctions pour lire les dossiers Bruker.
* Vizualisation.R : création des graphes avec ggplot2.
* Integration.R : détection des pics et intégration.

**2. Fonctionnalités principales (structure du serveur)**

**2.1 Chargement des données**

* **shinyDirChoose()** : active le sélecteur de dossiers.
* **parseDirPath()** : transforme la sélection utilisateur en chemin réel.
* **read\_bruker(path)** : lit les fichiers Bruker (acqus + ser/fid), extrait les matrices ppm/intensité.
* **spectra\_list()** : contient tous les spectres chargés dans l'application.
* **bruker\_data()** : spectre actif sélectionné.

**2.2 Calcul de seuils pour les contours**

* **Bouton calculate\_contour** : déclenche le calcul.
* **seuil\_max\_pourcentage()** : renvoie max(mat) \* pourcentage.
* **seuil\_bruit\_multiplicatif()** : estime le bruit (écart-type des valeurs faibles) \* facteur.
* Le résultat est stocké dans calculated\_contour\_value() pour usage ultérieur.

**2.3 Génération des courbes de niveau (plots)**

* **find\_nmr\_peak\_centroids(mat, ...)** :
  + Génère les contours geom\_contour() ou geom\_contour\_filled().
  + Retourne un ggplot enrichi, plus les données de niveau (contour\_data).
  + Stocké dans result\_data\_list().
* **contour\_plot\_base()** : stocke la version de base du plot, utilisée comme fondation pour l'affichage interactif.

**2.4 Détection automatique des centroïdes et boîtes**

* **process\_nmr\_centroids()** :
  + Prend la matrice spectrale et les données de contours.
  + Applique un clustering DBSCAN aux coordonnées des contours.
  + Pour chaque cluster, calcule le centroïde pondéré et les bornes du rectangle (xmin, xmax, ymin, ymax).
  + Filtrage par plage ppm (optionnel).
  + Résultat :
    - centroids\_data() : data.frame avec F2\_ppm, F1\_ppm, stain\_intensity, stain\_id.
    - bounding\_boxes\_data() : data.frame avec coordonnées de la boîte + id.

**2.5 Ajout/Suppression manuelle**

* **add\_manual\_centroid()** :
  + Coordonnées ppm saisies via formulaire (F2/F1).
  + Estimation de l’intensité locale autour des coordonnées à l’aide de contour\_data filtré spatialement (via eps).
* **add\_manual\_bbox()** :
  + Boîte définie manuellement (xmin, xmax, ymin, ymax).
  + Intensité = somme des valeurs de la matrice bruker\_data() à l’intérieur de la boîte.
* **delete\_centroid() / delete\_bbox()** :
  + Supprime la ligne sélectionnée dans le tableau affiché par DTOutput().

**2.6 Visualisation dynamique**

* **refresh\_nmr\_plot()** :
  + Construit le graphique final :
    - Base : contour\_plot\_base() (ggplot)
    - Points des centroïdes : geom\_point() avec couleur par intensité.
    - Boîtes : geom\_rect() avec linetype = dashed.
  + Résultat converti via ggplotly() et affiché dans interactivePlot.

**2.7 Export**

* **export\_centroids** : écrit centroids\_data() dans exported\_centroids.csv.
* **export\_boxes** : déclenche un downloadHandler qui fournit les boîtes courantes.
* **project\_centroids\_with\_intensity()** :
  + Pour chaque spectre et centroïde, extrait l’intensité locale depuis contour\_data.
  + Produit un tableau par spectre, stocké dans centroid\_exports/.
* **Batch run\_batch** :
  + Pour chaque spectre de spectra\_list(), applique les bounding\_boxes\_data().
  + Calcule l’intensité dans chaque boîte et exporte un CSV dans results\_batch/.

**3. Réactifs clés**

* bruker\_data() : spectre courant affiché
* spectra\_list() : tous les spectres chargés
* spectra\_plots() : plots associés
* contour\_plot\_base() : plot de base sans overlay
* nmr\_plot() : plot avec overlay
* centroids\_data() : centroïdes détectés ou ajoutés
* bounding\_boxes\_data() : boîtes calculées ou ajoutées
* result\_data\_list() : données du traitement par spectre (contours, plots, etc.)

**4. Extensions possibles**

**a. UI/UX**

* Prise en charge du double-clic pour placer un centroïde ou une zone
* Repositionnement graphique des boîtes

**b. Intelligence artificielle**

* Ajout d’un module CNN pour la reconnaissance automatique (ex: keras, torch)
* Système d’annotation semi-automatique

**c. Exportation**

* Support de formats mzTab, JSON, NetCDF
* Export SVG/PNG haute résolution

**d. Interopérabilité**

* Connexion à des bases de métabolites ou de référence RMN

## 5. Packages utilisés

Voici la liste des principaux packages R utilisés dans l'application, ainsi que leurs versions recommandées au moment de la rédaction :

|  |  |  |
| --- | --- | --- |
| **Package** | **Utilisation principale** | **Version recommandée** |
| shiny | Application web interactive | ≥ 1.8.0 |
| shinydashboard | Mise en page type tableau de bord | ≥ 0.7.2 |
| shinyFiles | Navigation dans l'arborescence de fichiers | ≥ 0.9.2 |
| shinyjs | Contrôle JS (affichage/masquage, notifications) | ≥ 2.1.0 |
| shinycssloaders | Animation de chargement | ≥ 1.0.0 |
| plotly | Graphiques interactifs à partir de ggplot | ≥ 4.10.0 |
| ggplot2 | Création de graphiques | ≥ 3.4.0 |
| DT | Tableaux interactifs | ≥ 0.25 |
| dplyr | Manipulation de tableaux | ≥ 1.1.0 |
| readr | Lecture/écriture de fichiers CSV | ≥ 2.1.3 |
| magrittr | Opérateur pipe %>% | ≥ 2.0.3 |

Ces packages doivent être installés et chargés dans l'environnement de travail pour que l'application fonctionne correctement.

## 6. Contact

**Auteur original** : Julien Guibert  
**Dépôt GitHub** : https://github.com/JulienGuibertTlse3/2DNMR-Analyst  
**Contact** : julien.guibert@inrae.fr