

Trabajo Fin de Grado

Facultad de Ciencias Experimentales



Grado en Química

Aplicación de nuevos algoritmos a experimentos de difusión molecular mediante RMN.

Application of new algorithms to NMR diffusion experiments.

Alejandro Cortés Villena
Curso académico 2016-2017
06/07/2017

Tutor/es
Dr. Ignacio Fernández de las Nieves
Departamento de Química y Física
Ing. Francisco Manuel Arrabal Campos
Departamento de Química y Física

Aplicación de nuevos algoritmos a experimentos de difusión molecular mediante RMN

Memoria del Trabajo Fin de Grado en Química presentada por
Alejandro Cortés Villena

Almería, 6 de julio de 2017

Fdo.: Alejandro Cortés Villena

Fdo.: Ignacio Fernández de las Nieves

Fdo.: Francisco Manuel Arrabal Campos

Life is a relationship between molecules

Linus Pauling

*Asegurémonos, pues, merced a una investigación bibliográfica cuidadosa,
de la originalidad del hecho o idea que deseamos exponer,
y guardémonos además de dar a luz prematuramente el fruto de la observación
Cuando nuestro pensamiento fluctúa todavía entre conclusiones diversas
y no tenemos plena conciencia de haber dado en el blanco,
ello es señal de haber abandonado harto temprano el laboratorio.
Conducta prudente será volver a él y esperar a que,
bajo el influjo de nuevas observaciones, acaben de cristalizar nuestras ideas.*

Santiago Ramón y Cajal “Reglas y consejos”

Agradecimientos

Quiero agradecer al Ministerio de Educación, Cultura y Deporte, la beca de colaboración concedida durante el curso académico 2016/2017.

Este TFG ha podido desarrollarse gracias a la financiación recibida por Bruker Biospin y por el Plan Propio de Investigación y Transferencia 2016 de la Universidad de Almería, gracias a la cual se han podido realizar todas las medidas de RMN necesarias para el desarrollo del trabajo.

MEMORIA

ÍNDICE

1. RESUMEN	3
2. ABSTRACT.....	3
3. INTRODUCCIÓN	5
3.1. FUNDAMENTOS FÍSICOS DE RMN	5
3.2. DIFUSIÓN MOLECULAR	8
3.3. INTRODUCCIÓN A LA PROGRAMACIÓN	12
3.4. TECNICAS DE OPTIMIZACIÓN	13
4. OBJETIVOS	17
5. MÉTODOS EXPERIMENTALES	18
5. 1. MATERIALES	18
5. 2. PREPARACIÓN DE LA MUESTRA	18
5. 3. TÉCNICAS ESPECTROSCÓPICAS	19
5. 4. SOFTWARE	20
6. RESULTADOS Y DISCUSIÓN	20
6.1. PROGRAMACIÓN DE UN VISOR DE RMN PARA LA VISUALIZACIÓN DE UN ESPECTRO DE ¹ H-RMN	21
6.2. IMPLEMENTACIÓN DEL ALGORITMO FISTA WITH BACKTRACKING.....	31
7. CONCLUSIONES	45
8. BIBLIOGRAFÍA	46
9. ABREVIATURAS Y ACRÓNIMOS.....	47
ANEXO	51

1. RESUMEN

Este TFG está constituido por dos bloques conectados entre sí. El primero comprende la creación de una aplicación “Visor de ^1H -RMN” con la herramienta de programación MATLAB, que permite además observar espectros superpuestos. El segundo bloque consiste en la implementación del algoritmo “FISTA *with backtracking*” a experimentos de difusión molecular mediante la técnica PGSE (pulsos de gradiente de campo magnético). El algoritmo aquí implementado ha sido probado en muestras reales de polímeros de poli(propilenglicol) de distinto peso molecular (450 y 5000 Da), y finalmente evaluado en una mezcla de ellos. El primero de los bloques ha sido aceptado para su publicación en la revista americana *Journal of Chemical Education*.

Palabras clave: *Polímeros, Resonancia Magnética Nuclear, MATLAB, FISTA with backtracking, ITAMeD.*

2. ABSTRACT

This TFG consists of two blocks connected to one another. The first comprises the creation of an application “ ^1H -NMR Viewer” with the MATLAB programming tool, which also allows to observe overlapping spectra. The second block consists of the implementation of the algorithm “FISTA *with backtracking*” to molecular diffusion experiments by the technique PGSE (magnetic field gradient pulses). The algorithm implemented here has been tested on real samples of poly(propylene glycol) polymers of distinctive molecular weight (450 and 5000 Da), and finally evaluated in a mixture of them. The first of the block has been accepted for publication in the *American Journal of Chemical Education*.

Keywords: *Polymers, Nuclear Magnetic Resonance, MATLAB, FISTA with backtracking, ITAMeD.*

3. INTRODUCCIÓN

3.1. FUNDAMENTOS FÍSICOS DE RMN

La Resonancia Magnética Nuclear (RMN) es una técnica espectroscópica de absorción que se basa en las propiedades que presentan los núcleos atómicos de absorber energía ($h\nu$) cuando son sometidos a la acción de una radiación no ionizante como lo es la radiofrecuencia (rf).¹ La gran mayoría de los núcleos atómicos poseen un momento angular intrínseco simbolizado por P . El espín (ver más adelante) es otra de las propiedades importantes en la RMN, gracias a la cual tienen momento angular. El núcleo de los átomos está formado por protones y neutrones, como resultado de la unión de *quarks*, por lo cual, el núcleo atómico adquiere una carga positiva. Este núcleo presenta un movimiento de giro alrededor de su propio eje, generando a su vez, un campo magnético, como lo hace toda carga en movimiento. El espín resultante de los protones y neutrones cuando no se encuentran totalmente apareados genera un dipolo magnético alineado en la dirección resultante del eje de giro nuclear. La magnitud intrínseca de este dipolo magnético es una propiedad fundamental del núcleo atómico, que se denomina momento magnético nuclear μ . Atendiendo a la mecánica cuántica, se sabe que el momento angular P , al igual que otras magnitudes atómicas, se encuentran cuantizadas, y que por tanto toman una serie de valores determinados como puede apreciarse en la siguiente ecuación:

$$P = \sqrt{I(I + 1)}\hbar \quad (1)$$

donde $\hbar = h/2\pi$, siendo h la constante de Planck, universalmente conocida con un valor de $6,6256 \cdot 10^{-34}$ J·s, e I el número cuántico del momento de espín angular, conocido como *espín*. El momento angular P y el momento magnético μ (ambas magnitudes vectoriales) se encuentran relacionados de la siguiente forma:

$$\mu = \gamma P \quad (2)$$

El factor de proporcionalidad γ es la constante giromagnética o magnetogírica, que es característica para cada isótopo de cada elemento. Este parámetro informa sobre el grado de sensibilidad de respuesta de un núcleo al experimento de RMN. A continuación, se representa en la Figura 1 una analogía que ilustra el momento magnético de un núcleo atómico positivo.²

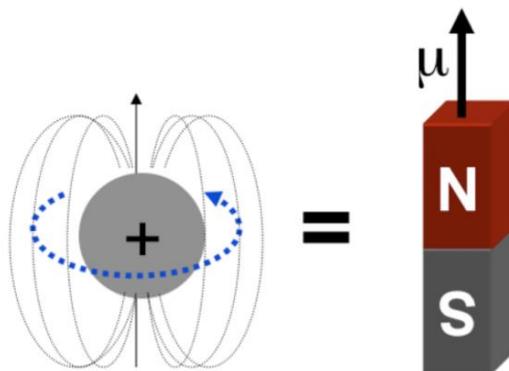


Figura 1. Representación del momento magnético de un núcleo positivo.³

Se pueden apreciar las líneas de fuerza del campo magnético cómo emergen del núcleo en cuestión al estar en movimiento. Este vector momento magnético se representa como un imán.

La variación del campo magnético en un determinado punto del espacio con respecto al tiempo origina un campo eléctrico que se puede desplazar y generar una corriente eléctrica, según una de las Leyes de Maxwell.

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \quad (3)$$

el signo negativo explica que el sentido de la corriente inducida es tal que su flujo se opone a la causa que lo produce, compensando así la variación de flujo magnético.⁴

La RMN por tanto sólo puede utilizarse para estudiar núcleos atómicos que presentan un número impar de protones o neutrones (o ambos), es decir, que al menos el número másico o atómico sea impar, ya que sólo estos presentan momentos magnéticos y, por tanto, son magnéticamente activos.

Los isótopos más abundantes en compuestos orgánicos son el protón (¹H) y el carbono (¹²C). El problema radica en que el ¹²C no es magnéticamente activo (A y Z par) y, por tanto, tiene un número cuántico *I* de momento de espín angular igual a 0. Como puede observarse en la siguiente ecuación:

$$\mu = \gamma \sqrt{I(I+1)} \hbar \quad (4)$$

al ser 0 el valor del espín *I*, el valor del momento magnético μ se hace 0.

Estos núcleos, por tanto, no poseen momento magnético y, por ende, no dan señal en RMN. Para elucidar la estructura del esqueleto carbonado se tiene en cuenta el isótopo ¹³C, que es mucho menos abundante que el ¹²C, pero como tiene un número másico impar, sí produce señal en los espectros de RMN.

Cuando un núcleo atómico se introduce en el interior de un campo magnético B_0 , el momento angular *P* se alinea con la dirección del campo magnético externo en una determinada dirección, por estar cuantizado.² De acuerdo con la teoría clásica de electromagnetismo, el núcleo atómico realiza un movimiento de precesión alrededor del campo magnético externo B_0 con una frecuencia angular ν_L , denominada frecuencia de Larmor. Además, es proporcional a la intensidad del campo magnético.⁵

Como se acaba de comentar anteriormente, la espectroscopia de ¹³C-RMN es mucho menos sensible que la de ¹H-RMN ya que sólo el 1.1 % de los átomos de carbono son de carbono-13 y presentan espín (el mismo espín que el de ¹H). La constante giromagnética del ¹³C es cuatro veces más pequeña que la del ¹H.⁶ Esto se puede observar fácilmente mediante la siguiente ecuación que emplea las frecuencias de Larmor (ν_L) de ambos núcleos:

$$\nu_{L^{13}C} = \frac{1}{4} \frac{\gamma_{1H}}{2\pi} B_0 \quad (5)$$

En el presente trabajo fin de grado sólo se tendrá en cuenta la resonancia magnética de ¹H. El fenómeno de RMN se observa cuando se irradia un núcleo

atómico, en el interior de un campo magnético externo B_0 , con un pulso de radiofrecuencia de una determinada duración que produce un giro de $\pi/2$ del vector momento magnético con una frecuencia de radiación ν_0 igual a la frecuencia de Larmor ν_L . Esto provoca que se desvíe el vector magnetización neto \vec{M}_0 (suma de las componentes Z de todos los momentos magnéticos nucleares en la muestra) de su posición de equilibrio. Esta situación crea una componente de la magnetización en el plano XY, llamada magnetización transversal.² Al volver a la situación de equilibrio produce una variación del campo magnético que induce una corriente eléctrica sobre una bobina receptora que se encuentra en el eje Y. Esta variación del campo magnético es lo que detecta el conjunto instrumental del equipo de RMN. La relación que hay entre campos eléctricos y magnéticos se puede expresar según la Ley de Ampère generalizada, que en su forma diferencial es:

$$\vec{\nabla} \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \quad (6)$$

siendo:

$$\vec{H} = \mu_0 \cdot \vec{B} \text{ (campo magnético)}$$

$$\vec{J} = \frac{\partial Q}{\partial A \cdot t} \text{ (flujo de corriente eléctrica)}^7$$

$$\vec{D} = \epsilon_0 \cdot \frac{\partial \phi}{\partial t} \text{ (corriente de desplazamiento)}^8$$

En el instante que se desconecta el pulso de radiofrecuencia (rf), el vector magnetización tiende a volver a su posición de equilibrio (paralelo a la dirección del campo magnético externo). Este proceso se conoce como relajación, en el cuál, disminuye la intensidad de la señal conforme aumenta el tiempo hasta que se hace nulo. Este proceso de relajación puede ser descrito mediante dos tiempos distintos de relajación: T_1 (tiempo de relajación longitudinal) y T_2 (tiempo de relajación transversal). Cabe destacar, que realmente lo que se mide es la componente de la magnetización M_y que se encuentra en el plano XY. Esta atenuación de la magnetización transversal detectada en la bobina receptora se denomina *Free Induction Decay* (FID).

Dada la dificultad para obtener información con la atenuación de la señal se recurre a un cambio de plano en frecuencia. Para ello se emplea una herramienta denominada Transformada de Fourier, que transforma las frecuencias encerradas en la FID en el dominio del tiempo a dominio de frecuencias. La Transformada de Fourier queda definida por la siguiente expresión:

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) \cdot e^{i\omega t} dt \quad (7)$$

donde $f(t)$ representa una señal oscilatoria y estacionaria y $F(\omega)$ representa el conjunto de frecuencias que definen el espectro.²

A continuación, se representa una imagen de la transformación de la FID a un espectro de ^1H -RMN convencional (Figura 2):

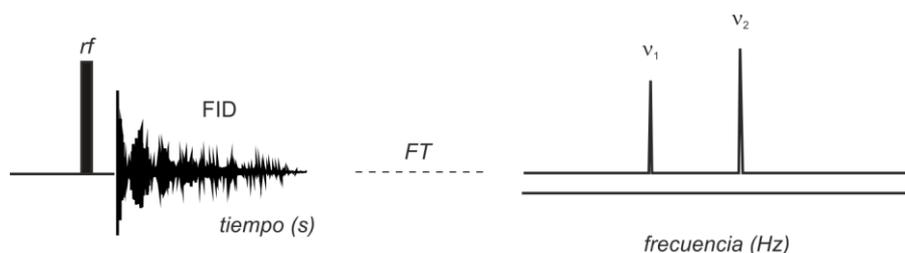


Figura 2. Transformación de la FID a dos señales espectrales ν_1 y ν_2 .

Se puede observar en la figura 2 que al aplicar la Transformada de Fourier a la FID se obtiene las frecuencias y sus amplitudes que la componen. Esto corresponde a la dimensión de las frecuencias (ν) en Hz. Para facilitar el tratamiento práctico de estas frecuencias se expresan como desplazamiento químico (δ), en unidades de ppm.² Esta operación se ilustra en la ecuación 8:

$$\delta = \frac{\nu_{\text{muestra}} - \nu_{\text{referencia}}}{\nu_{\text{referencia}}} \cdot 10^6 \quad (8)$$

siendo ν_{muestra} la frecuencia de resonancia de la muestra y $\nu_{\text{referencia}}$ la frecuencia de resonancia de un compuesto de referencia, generalmente TMS (tetrametilsilano).

La RMN es una de las herramientas de elucidación estructural más importante en Química Orgánica, aunque también se utiliza en Química Inorgánica, Organometálica, Bioquímica y en otras muchas. Esta misma técnica también se ha extendido a otros campos científicos, como por ejemplo en la medicina, en el cuál el objetivo es obtener imágenes del cuerpo por resonancia magnética que sirve para diagnosticar enfermedades o determinados tipos de traumatismos. Esta técnica, se conoce específicamente como Resonancia Magnética de Imagen (RMI).^{9a}

3.2. DIFUSIÓN MOLECULAR

En la frontera de la RMN, se desarrollaron métodos capaces de obtener coeficientes de difusión (D). Uno de los principales objetivos de este trabajo fin de grado consiste en medir coeficientes de difusión de unos polímeros y una mezcla de polímeros. La difusión molecular es el resultado del movimiento Browniano de las moléculas en disolución que ocurre en ausencia de cualquier fuerza externa como pudieran ser los gradientes de concentración, variaciones en el campo eléctrico, o convección inducida por cambios de temperatura.

La primera metodología que surgió para su estudio fue la aplicación de la secuencia de pulsos de radiofrecuencia y gradientes de campo magnéticos denominada PGSE (Pulsed Field Gradient Spin Echo). A partir de esta metodología, surgieron otras que son modificaciones de esta, tales como, la secuencia estimulada, STE, entre otras.^{9a}

La técnica está basada en la medida de la atenuación de la señal que se produce durante un tiempo de codificación y decodificación del momento magnético nuclear referenciado a un sistema de referencia rotatorio. El movimiento browniano o de difusión provoca un desfase en el momento magnético nuclear obteniendo una atenuación en la señal detectada. El momento magnético nuclear rota en el plano XY en función de la codificación y rota en sentido contrario en la decodificación. En los experimentos STE, la codificación y decodificación dependen de la fuerza del

gradiente.^{9a} Para hallar la relación entre la fuerza del gradiente de campo magnético con el coeficiente de difusión, se parte de la relación de Abragam y se opera el resultado para el caso de un pulso de 90° seguido de uno de 180° (secuencia SE). La ecuación 9, ecuación diferencial en derivadas parciales, relaciona el coeficiente de difusión y la atenuación de la señal^{9b}:

$$\frac{\partial \varphi(z,t)}{\partial t} = -i \cdot \gamma \cdot z \cdot G \cdot \varphi(z,t) + D \cdot \Delta \cdot \varphi(z,t) \quad (9)$$

Aquí, $\varphi(z,t)$ está definido como^{9b}:

$$M_x + i \cdot M_y = \varphi(z,t) \cdot e^{[-(i \cdot \omega_0 + \frac{1}{T_2}) \cdot t]} \quad (10)$$

en donde $M_x + i \cdot M_y$ representan el comportamiento de los componentes de la magnetización nuclear en el plano perpendicular al campo magnético B_0 aplicado. Haciendo una serie de operaciones se obtiene^{9b}:

$$\frac{-1}{D \cdot \gamma^2} \cdot \ln \left(\frac{\varphi(2\tau)}{\varphi(0)} \right) = \int_0^{2\tau} F(\tau)^2 dt - 4 \cdot f \cdot \int_{\tau}^{2\tau} F(\tau) dt + 4 \cdot f^2 \cdot \tau \quad (11)$$

siendo:^{9b}

$$f = F(\tau) = \int_0^{\tau} G(t') dt'$$

Finalmente, la atenuación de la señal resultante fue descrita por Stejskal y Tanner en 1965¹⁰, que resuelve la ecuación 11, según la siguiente expresión^{9c,10}:

$$\Psi(G) = \frac{I}{I_0} = \int_0^{\infty} A(D) \cdot e^{-[D \cdot \gamma_{eff}^2 \cdot \delta^2 \cdot \sigma^2 \cdot G^2 \cdot \Delta']} \quad (12)$$

donde $A(D)$ es la distribución de coeficientes de difusión, D es el coeficiente de difusión, γ es la constante giromagnética, δ es la duración del gradiente, σ es la forma del gradiente, G es la intensidad del gradiente y Δ' es el tiempo de difusión aparente.

La atenuación de la señal puede ser descrita además como la Transformada de Laplace Inversa de una distribución de coeficientes de difusión $A(D)$. Siendo la Transformada de Laplace:

$$\mathcal{L}\{f(t)\} = \int_0^{\infty} f(t) \cdot e^{-s \cdot t} dt \quad (13)$$

Y la Transformada de Laplace Inversa (ILT):

$$\mathcal{L}\{f(t)\}^{-1} = f(t) \quad (14)$$

La secuencia estimulada (STE) emplea tres pulsos de radiofrecuencias de una cierta duración que provoca un desplazamiento de la magnetización de $\pi/2$ y dos gradientes de campo magnético (que rompen la homogeneidad del campo magnético externo), que funcionan de forma similar al experimento de eco de espín (SE).^{9a} A continuación se representan las dos secuencias (Figura 3):

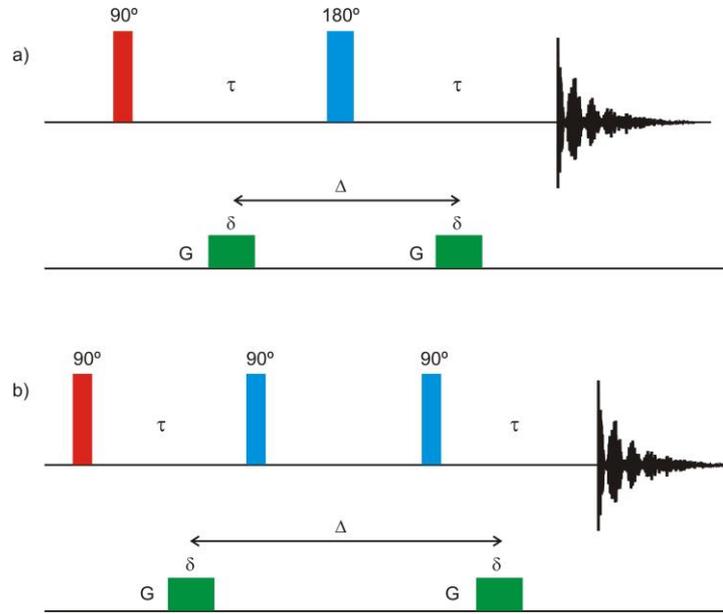


Figura 3. Secuencias de pulsos para medidas de difusión mediante RMN: a) Stejskal-Tanner y b) Eco estimulada (STE).

La atenuación de la señal para esta secuencia STE (Figura 3b) viene dada por una expresión donde intervienen los tiempos de relajación transversal (T_2) y longitudinal (T_1). Dicha expresión es la siguiente^{9a}:

$$\Psi(G^2) = \frac{I}{I_0} = \frac{1}{2} \cdot e^{-\left[\left(\frac{2 \cdot \tau_2}{T_2} + \frac{\tau_1}{T_1}\right) + (\gamma \cdot G \cdot \delta)^2 \cdot D \cdot \Delta^2\right]} \quad (15)$$

En esta secuencia de pulsos, la mitad de la magnetización se encuentra en el plano XY y la otra mitad en el eje y. La magnetización se encuentra afectada por los tiempos de relajación T_1 y T_2 . Por tanto, cuando se mide difusión, interesa que los núcleos tengan un T_2 largo ya que, si estos núcleos se relajan rápidamente (T_2 pequeño), sus espines se desfasan completamente por todo el plano XY sin llegar a observar la atenuación producida por difusión. Por lo que se observaría en la dimensión de coeficientes de difusión una banda ancha o simplemente no habría señal.^{9a} Una descripción matemática de este fenómeno se resume a continuación:

$$\text{Cuando } T_2 \sim \infty \Rightarrow \left(\frac{2 \cdot \tau_2}{T_2}\right) \approx 0 \quad (16)$$

Sustituyendo en la ecuación 15 el valor de T_2 , ésta se puede simplificar asegurando que la atenuación de la señal detectada es debida al proceso difusivo. Por tanto, sustituyendo en la ecuación 15, queda:

$$\Psi(G^2) = \frac{I}{I_0} = e^0 \cdot e^{\frac{\tau}{T_1} \cdot \gamma^2 \cdot G^2 \cdot \delta^2 \cdot D \cdot \Delta'} \equiv \Psi(G^2) = e^{\frac{\tau}{T_1} \cdot \gamma^2 \cdot G^2 \cdot \delta^2 \cdot D \cdot \Delta'} \quad (17)$$

Cabe destacar también la aplicación de estos métodos basados en experimentos de difusión molecular para predecir pesos moleculares promedios en polímeros. Para ello, se han propuesto un número de métodos empíricos que relacionan el coeficiente de difusión con el peso molecular promedio. Así, por ejemplo, el grupo de Williard aplicó esta técnica para correlacionar el coeficiente de difusión con el peso molecular promedio de especies organolíticas reactivas.¹¹

Se han predicho pesos moleculares fiables usando curvas de calibración construidas a partir de estándares adecuados. El grupo de investigación en el que se encuadra este trabajo fin de grado ha contribuido recientemente en este contexto desarrollando las denominadas curvas de calibrado universales, en el que la predicción del peso molecular ya no depende de la viscosidad del medio.^{12a} Para ello, emplearon el algoritmo ITAMeD¹³ y el método de mínimos cuadrados para la determinación cuantitativa del coeficiente de difusión de un conjunto de polímeros de poliestireno monodispersos.

El tamaño de una molécula, representado por su radio hidrodinámico, está relacionado con el coeficiente de difusión según la ecuación de Stokes-Einstein.^{12b} Para una molécula con una forma y tamaño no esférica y teniendo en cuenta los efectos de microfricción^{12c}, la ecuación se presenta como:^{14a,14b}

$$D = \frac{k_B \cdot T}{c \cdot (r_{S1}, r_H) \cdot f_s(a, b) \cdot \eta \cdot r_H} \quad (18)$$

siendo k_B la constante de Boltzmann, T la temperatura, c el factor de microficción, r_{S1} el radio del disolvente, r_H el radio hidrodinámico, $f_s(a, b)$ el factor de forma y η la viscosidad.

En sistemas poliméricos, las características de la masa son quizás más importantes que las geométricas, donde la relación empírica de Flory es probablemente la relación más directa entre el peso molecular promedio y el coeficiente de difusión^{14c,14d}.

$$D = A \cdot M_w^{-\alpha} \quad (19)$$

Li y colaboradores emplearon la ecuación 20, que procedía de la anteriormente aludida y desarrollada por Williard, para relacionar linealmente el coeficiente de difusión con el peso molecular promedio, incorporando la densidad de la molécula (ρ) y el número de Avogadro (N_A).

$$\log(D) = -\frac{1}{3} \cdot \log(M_w) + \frac{1}{3} \cdot \log \rho - \frac{1}{3} \cdot \log \eta - \frac{1}{3} \cdot \log\left(\frac{162 \cdot \pi^2}{k_B^3 \cdot T^3 \cdot N_A}\right) \quad (20)$$

La sucesiva aplicación de esta ecuación a distintos sistemas y por distintos autores, siempre llevaba a un incumplimiento del término $\frac{1}{3}$, cuando se representaba

$\log(D)$ frente al $\log(M_w)$, en el que las rectas que siempre se obtenían se alejaban significativamente de ese factor $\frac{1}{3}$.^{12a} Los resultados experimentales obtenidos por Pollack^{14e} mostraron que era posible tener una relación no lineal entre el coeficiente de difusión y la viscosidad. Estos resultados podrían relacionar la ecuación de Stokes-Einstein (ecuación 18) de una forma geométrica y asignar la interacción entre el medio y la difusión de una forma anisotrópica. Sobre el trabajo de Pollack, que es una relación empírica entre el coeficiente de difusión y la viscosidad. Por tanto, se reconsideró la relación del radio hidrodinámico con el coeficiente de difusión por la ecuación de Stokes-Einstein como una relación linealizada independiente de la viscosidad. A continuación, se muestra dicha ecuación^{12a}:

$$\log(D\eta) = -\frac{1}{3\beta} \cdot \log(M_w) - \frac{1}{\beta} \cdot \log\left(\frac{c \cdot f_s \cdot K^{1-\beta} \cdot \eta^{\beta \cdot (\beta-1)} \cdot \left(\frac{3}{4 \cdot \rho \cdot N_A \cdot \pi}\right)^{\frac{1}{3}}}{k_B \cdot T}\right) \quad (21)$$

De la pendiente del ajuste lineal, se determina β . El término β describe una interacción molecular entre el disolvente y el soluto.

Este nuevo enfoque representa la primera curva de calibrado universal para predicciones de pesos moleculares promedios de polímeros con independencia de la viscosidad del disolvente utilizado.^{12a}

3.3. INTRODUCCIÓN A LA PROGRAMACIÓN

Para representar el conjunto de espectros de ^1H -RMN y poder observar gráficamente la atenuación de la señal con respecto al incremento del gradiente (que es el parámetro que varía en los experimentos de difusión molecular por RMN), se va hacer uso de una herramienta de cálculo numérico, MATLAB.¹⁵

La programación es el proceso de diseñar, codificar y depurar un código, en un determinado lenguaje de programación (normalmente en lenguaje natural), conformando un programa.¹⁶ El ordenador representa la información utilizando únicamente ceros y unos. Este programa implementa una serie de algoritmos, que son secuencias de instrucciones en pseudocódigo que permiten realizar unas operaciones de forma secuencial. El pseudocódigo es una notación textual que puede llegar a ser muy parecida al lenguaje natural. El pseudocódigo dispone de un conjunto de palabras clave reservadas que expresan tanto las acciones elementales como las diferentes estructuras lógicas del algoritmo.¹⁷ El diagrama de flujo es la representación gráfica del algoritmo. En la Figura 4 se muestra el diagrama de flujo, que ha sido extraído del artículo recientemente aceptado en el que se ha descrito parte de este trabajo fin de grado.

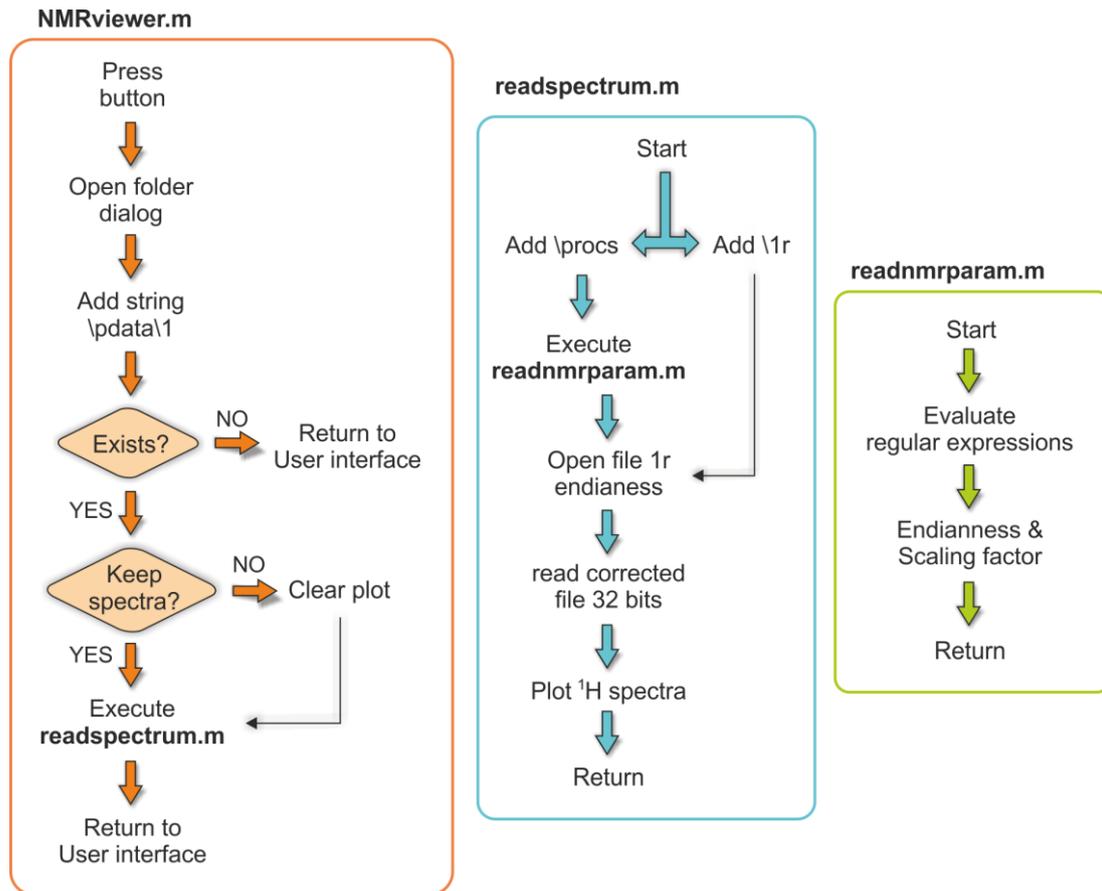


Figura 4. Diagrama de flujo usado para leer un espectro ^1H -RMN monodimensional. Los tres bloques corresponden con los archivos que han sido programados.¹⁸

3.4. TÉCNICAS DE OPTIMIZACIÓN

Se van a utilizar técnicas de optimización avanzada para resolver la Transformada de Laplace Inversa numérica. Para el caso del presente trabajo, decaimientos multiexponenciales aplicados a la difusión molecular por RMN, se pueden diferenciar dos tipos de métodos. Los métodos de canal simple y los métodos multivariantes^{19a}. Hay que mencionar que estos métodos son técnicas de optimización aplicados a programación no lineal, es decir, consistente en maximizar o minimizar funciones objetivo no lineales con restricciones, como por ejemplo función objetivo no lineal no negativa.

Los métodos multivariantes analizan la totalidad de la información disponible en el espectro. Teóricamente, estos métodos son preferibles que los de canal simple, ya que proporcionan un espectro más interpretable para todos los componentes de la mezcla. También, estos métodos pueden manipular regiones solapadas en el espectro de ^1H -RMN, que son tratados por los métodos simples con un enfoque diferente. Sin embargo, sólo resuelven un número limitado de componentes. Desde un punto de vista práctico, éstos no terminan de resolver mezclas complejas de forma adecuada^{19a}, dando resultados erróneos o no interpretables. Dentro de este grupo, se pueden comentar dos tipos de algoritmos: DECRA (*Direct Exponential Curve Resolution Algorithm*)^{19b} y MCR (*Multivariate Curve Resolution*).^{19c} La ventaja de DECRA es que un conjunto de datos experimentales se divide en dos, de tal manera que, el método GRAM (*Generalized Range Annihilation Method*)^{19a} puede ser

aplicado para analizar las contribuciones de los decaimientos multiexponenciales de una mezcla. DECRA usa una solución exacta y se obtiene un perfil de decaimiento de cada componente, incluso con la existencia de picos solapados. MCR se basa en la regresión de mínimos cuadrados alternante. Depende de la varianza de intensidad de cada componente con los niveles de gradiente y, por tanto, puede tolerar desviación del decaimiento exponencial ideal. En la práctica, los métodos multivariantes son difíciles de tratar y no generan buenos resultados.

Los métodos de canal simple emplean sólo una parte del espectro. Son simples de tratar y proporcionan coeficientes de difusión razonables de un largo número de componentes en una mezcla, a partir de picos de RMN. Sin embargo, en el caso de picos solapados (muy común en las muestras reales) requieren de algoritmos matemáticos más sofisticados desde un punto de vista matemático. Dentro de este grupo, se pueden comentar dos tipos de algoritmos: SPLMOD (*SPLine MODel*)^{19d} y CONTIN (*CONstrained regularizaTion program for INverting noisy linear algebraic and integral equations*).²⁰ SPLMOD es aplicado a componentes con coeficientes de difusión discretos (valores únicos) mientras que CONTIN es aplicado a distribuciones de difusión (el coeficiente de difusión es considerado como un rango con una distribución normal, log-normal o distribución gamma generalmente).^{19a}

En este trabajo fin de grado se va a utilizar la metodología de un canal (canal simple).

Desde el punto de vista matemático, los problemas de difusión pueden ser considerados como débilmente formulados debido a que pequeñas modificaciones en x producen grandes modificaciones en $f(x)$. También presenta no unicidad en la solución debido al ruido instrumental.

Para solucionar este problema, han sido propuestas numerosas técnicas matemáticas, de las cuales, una de ellas es la técnica de regularización. La regularización es una técnica que proporciona estabilidad numérica y unicidad a la solución de un problema débilmente formulado mediante la adición de un funcional penalizador en su formulación. A continuación, se presenta el método de regularización de Tikhonov más conocido y frecuente.

$$\mathbf{x}_\lambda = \operatorname{argmin} \{ \|\mathbf{Ax} - \mathbf{b}\|^2 + \|\Gamma\mathbf{x}\|_2 \} \quad (22)$$

siendo el término $\|\Gamma\mathbf{x}\|_2$ un funcional penalizador del método de regularización de Tikhonov.²¹

El ejemplo más simple de los métodos de un canal es el algoritmo Levenberg-Marquardt, en donde el número de componentes se asume directamente. SPLMOD, que es un método más sofisticado, requiere sólo el número máximo de componentes para ser ajustado y busca la solución óptima automáticamente. El primer algoritmo usado ampliamente de este grupo de métodos fue el CONTIN,²⁰ que fue empleado en señales con un número múltiple de componentes. Delsuc y Malliavin²² implementaron el algoritmo MaxEnt (*MAXimum ENTropy*) para procesar espectros de difusión.

Ambos métodos, CONTIN y MaxEnt se pueden considerar como la regularización de Tikhonov del siguiente problema de minimización de mínimos cuadrados no negativos:

$$\|\Phi\mathbf{A} - \Psi\|_{l_2}^2 + \tau\Theta(\mathbf{A}) \quad (23)$$

donde Φ es la matriz del espacio de Hilbert asociada al kernel, para este caso asociado a una función exponencial.

El término $\Theta(\mathbf{A})$ es definido como sigue para el método MaxEnt:

$$\Theta(\mathbf{A}) = - \sum_i \frac{A_i}{\sum_j A_j} \cdot \log\left(\frac{A_i}{\sum_j A_j}\right) \quad (24)$$

mientras que para el CONTIN es:

$$\Theta(\mathbf{A}) = \|\mathbf{A}\|_{l_2} \quad (25)$$

Hay muchas más posibilidades para la elección de $\Theta(\mathbf{A})$ que no han sido ampliamente explotadas en el procesamiento de señales de difusión. Por ejemplo, la regularización usada en CONTIN puede ser generalizada para una norma arbitraria l_p :

$$\Theta(\mathbf{A}) = \|\mathbf{A}\|_{l_p} \quad (26)$$

(usualmente $0 < p \leq 2$). En 2004, Daubechies y sus colaboradores²³ discutieron las ventajas de ajustar $p = 1$. En tal caso, el problema de optimización puede ser descrito como sigue:

$$\min_{\mathbf{A}} \|\Phi\mathbf{A} - \Psi\|_{l_2}^2 + \tau\|\mathbf{A}\|_{l_1} \quad (27)$$

El mínimo puede ser encontrado usando algoritmos muy robustos como el ISTA.²⁴ El enfoque basado en la norma l_1 no promueve la dispersión de los coeficientes de difusión, es decir, hay muchas soluciones \mathbf{A} que están de acuerdo con los datos experimentales Ψ . En este sentido, la regularización de la norma l_1 puede considerarse como un híbrido de los dos algoritmos antes mencionados.

La desventaja del ISTA es que presenta una velocidad de convergencia muy lenta, que puede ser un inconveniente en el caso de problemas débilmente formulados, tales como los problemas de difusión molecular con decaimiento multiexponencial. La solución a este problema fue la mejora del algoritmo que condujo al denominado FISTA.²⁴

En la segunda parte de este trabajo fin de grado se va a tratar el ITAMeD aplicado a datos de PGSE de RMN, que usa un FISTA con retroseguimiento (*Fast Iterative Shrinkage-Thresholding Algorithm with backtracking*) aplicado a decaimientos multiexponenciales.

El ITAMeD emplea la minimización de la norma l_1 . Este método es el más adecuado para muestras con un bajo grado de polidispersión y que posean distribuciones de coeficientes de difusión discretos.¹³

Como se ha comentado anteriormente, el algoritmo empleado en este trabajo de fin de grado es el “FISTA with backtracking”,²⁴ que viene descrito de la siguiente forma:

Paso 0. Toma $L_0 > 0$, algún $\eta > 1$ y $x_0 \in \mathbb{R}^n$. Ajusta $y_1 = x_0$, $t_1 = 1$.

Paso k. ($k \geq 1$) Encuentra el entero no negativo más pequeño i_k tal que con $\bar{L} = \eta^{i_k} \cdot L_{k-1}$

$$F(p_L(y_k)) \leq Q_L(p_L(y_k), y_k)$$

Ajusta $\bar{L} = \eta^{i_k} \cdot L_{k-1}$ y computa

$$\begin{aligned} x_k &= p_{L_k}(y_k), \\ t_{k+1} &= \frac{\left(1 + \sqrt{(1 + 4 \cdot t_k^2)}\right)}{2}, \\ y_{k+1} &= x_k + \left(\frac{t_k - 1}{t_{k+1}}\right) \cdot (x_k - x_{k-1}). \end{aligned}$$

Algoritmo 1. FISTA with backtracking.

A continuación, se describen las variables utilizadas:

$t = \frac{1}{L(f)}$, siendo $L(f)$ la constante de Lipschitz

$$F(x) = \|Ax - b\|^2 + \lambda \|x\|_1$$

$$Q_L(x, y) = f(y) + \langle x - y, \nabla f(y) \rangle + \frac{L}{2} \|x - y\|^2 + g(x)$$

$c = x - 2 \cdot t \cdot A' \cdot (A \cdot x - b)$, siendo c el movimiento en la dirección del gradiente del espacio de Hilbert.

El operador *Thresholding* (p_L) se define como:²⁴

$$p_L(c) = \text{soft_thresholding}\left(c, \lambda \cdot \frac{1}{L(f)}\right)$$

A continuación, se muestra en la Figura 5 un ejemplo de la función *soft_thresholding*, en donde la función es 0 en un tramo definido adecuadamente.

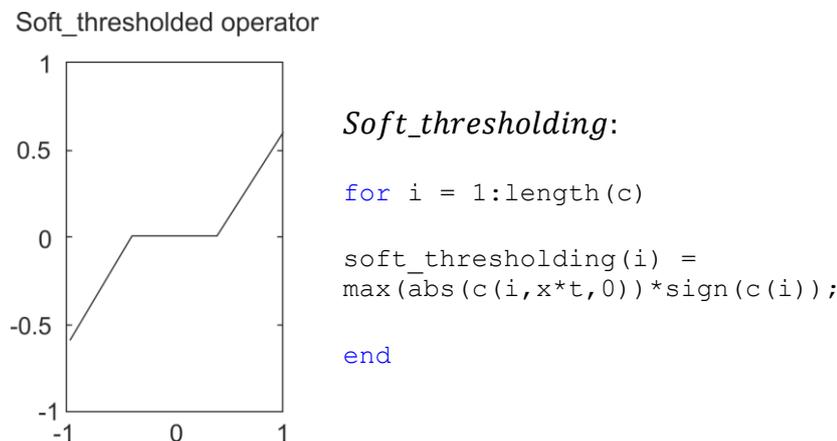


Figura 5. Ejemplo de función *soft_thresholding*.²⁵

Este algoritmo computa el tamaño del paso mediante el *backtracking* empezando en cualquier valor mayor que uno. Este método emplea la misma velocidad de convergencia mejorada como el método del gradiente óptimo de Nesterov para funciones convexas compuestas.^{26a,26b} Esta técnica de optimización se define para revolver problemas de la forma:

$$\min \{F(x) \equiv f(x) + g(x) : x \in \mathbb{R}^n\} \quad (26)$$

El número de iteraciones requeridas del FISTA para obtener una solución óptima ε , es que para un \tilde{x} tal que $F(\tilde{x}) - F^* \leq \varepsilon$, es como mucho $\lceil \frac{C}{\sqrt{\varepsilon}} - 1 \rceil$, donde $C = \sqrt{2 \cdot \alpha \cdot L(f) \cdot \|x_0 - x^*\|^2}$.

En estos métodos, se utiliza una combinación de puntos previos $\{x_{k-1}, x_{k-2}\}$ para computar el nuevo punto en cada iteración.²⁴

4. OBJETIVOS

Las competencias requeridas para el cumplimiento de cualquier TFG son las siguientes:

E-Q1. Capacidad para demostrar el conocimiento y comprensión de los hechos esenciales, conceptos, principios y teorías relacionadas con la Química.

E-Q2. Capacidad para aplicar dichos conocimientos a la resolución de problemas cualitativos según modelos previamente desarrollados.

E-Q3. Competencias para evaluar, interpretar y sintetizar datos e información Química.

E-Q4. Capacidad para reconocer y llevar a cabo buenas prácticas en el trabajo científico.

E-Q5. Competencia para presentar, tanto en forma escrita como oral, material y argumentación científica a una audiencia especializada.

E-Q6. Destreza en el manejo y procesado informático de datos e información química.

En el presente TFG se pretenden adquirir dichas competencias mediante la creación de un visor de espectros de RMN, y mediante la implementación de un algoritmo para la identificación de coeficientes de difusión en mezclas poliméricas. Para ello, se ha empleado la RMN como herramienta espectroscópica, y se han aplicado algoritmos descritos para otras disciplinas para su aplicación a los problemas de polímeros. Los objetivos concretos se describen a continuación:

- Exposición oral de un seminario al grupo de investigación FQM-376 Advanced NMR Methods and Metal-based Catalysts (www2.ual.es/NMRMBC/).
- Creación de un visor de RMN, específicamente programado para la visualización de espectros de ¹H-RMN monodimensionales, incluyendo la posibilidad de solapar varios espectros simultáneamente.

- Realización de medidas de difusión molecular mediante el uso de un espectrómetro Bruker Avance HD 500 MHz ubicado en los Servicios Técnicos de la Universidad de Almería.
- Implementación del algoritmo “FISTA *with backtracking*” a decaimientos multiexponenciales obtenidos por ^1H -RMN, para la obtención de coeficientes de difusión en tres muestras problema de naturaleza polimérica.

En el apartado de conclusiones se abordará la correlación entre los objetivos cumplidos con sus correspondientes competencias adquiridas.

5. MÉTODOS EXPERIMENTALES

5.1. MATERIALES

A continuación, se nombran los materiales y reactivos empleados en la preparación de las muestras de RMN:

- ✓ Jeringas de plástico (NORM-JECT de 1 mL)
- ✓ Microbalanza analítica (METLER TOLEDO, modelo XP6)
- ✓ Viales de muestreo (Sigma-Aldrich, modelo SU860012 SUPELCO, de 1.5 mL, O.D. x H x I.D. 11.6 mm x 32 mm x 6 mm)
- ✓ Tubos de RMN de 5mm (NORELL, ST500-7 de 100 mm de longitud)
- ✓ Parafilm (Sigma-Aldrich, 4 in x 125 ft.)
- ✓ Poli(propilenglicol)-450 (PPG450, American Polymer Standards Corporation, Pico peso molecular promedio: 450 Da e índice de polidispersión de 1,12.
- ✓ Poli(propilenglicol)-5000 (PPG5000, American Polymer Standards Corporation, Pico peso molecular promedio: 4850 Da e índice de polidispersión de 1,10.
- ✓ Benceno deuterado (C_6D_6), 99,6 %, ampollas de 0,75 mL, Euriso-top.

5.2. PREPARACIÓN DE LA MUESTRA

Para la preparación de las muestras de RMN se empleó benceno- d_6 y el correspondiente polímero de polipropilenglicol (PPG). Se han preparado las siguientes tres muestras:

- PPG-450: Se pesan 0,743 mg del PPG-450 Da en un vial de muestreo en la microbalanza analítica con ayuda de una jeringa. Seguidamente, se añaden 0,5 mL de benceno- d_6 con otra jeringa, se homogeniza bien la disolución en el vial y se añade al tubo de RMN. Se etiqueta y se mide.
- PPG-5000: Se pesan 1,228 mg del PPG-5000 Da en un vial de muestreo en la microbalanza analítica con ayuda de una jeringa. Seguidamente, se añaden 0,5 mL de benceno- d_6 con otra jeringa, se homogeniza bien la disolución en el vial y se añade al tubo de RMN. Se etiqueta y se mide.
- PPG-Mix: Se pesan 1,303 mg del PPG-450 Da y 1,302 mg del PPG-5000 Da en el mismo vial de muestreo en la microbalanza analítica con ayuda de una jeringa. Seguidamente, se añaden 0,5 mL de benceno- d_6 con otra

jeringa, se homogeniza bien la disolución en el vial y se añade al tubo de RMN. Se etiqueta y se mide.

El benceno- d_6 es un compuesto orgánico altamente inflamable y altamente tóxico, por lo que su uso debe ser lo más rápido posible dentro de la campana de ventilación encendida, y empleando guantes. Después de su manipulación, se recomienda dejar aproximadamente 5 minutos la campana encendida para poder eliminar los restos de benceno- d_6 que pudiera haber en la atmósfera. Después de haber utilizado las jeringas y los viales con benceno- d_6 , se recomienda depositarlos dentro de algún frasco y cerrar el tapón correctamente con parafilm. La información adicional de este compuesto se puede encontrar en literatura.²⁷

5. 3. TÉCNICAS ESPECTROSCÓPICAS

A continuación, se describe las características y la configuración del equipo de RMN utilizado para obtener los espectros de ^1H -RMN y difusión correspondiente.

Los espectros de ^1H -RMN obtenidos se han registrado en un espectrómetro Bruker Avance III HD con frecuencia de Larmor de ^1H de 500 MHz. La sonda empleada para la realización de los experimentos de difusión ha sido una triple inversa de tipo TBI $^1\text{H}/^{31}\text{P}/\text{BB}$ (^{31}P - ^{109}Ag) de 5 mm y con gradientes en el eje Z. En la Figura 6 se muestra el equipo y una sonda similar a la empleada.



Figura 6. Izquierda) Imán superconductor de 11.7 T empleado en este TFG; Derecha) Sonda de RMN similar a la empleada en este TFG.

Los experimentos realizados en este TFG son de ^1H -RMN y de difusión molecular. Todos los experimentos se realizaron sin control de temperatura

utilizando un pulso de radiofrecuencias de 9,7 o 10,5 μs , y una potencia de 9 W, todos los experimentos con 32 scans. El espectro resultante de un experimento de ^1H -RMN es una representación de la intensidad de la señal frente al desplazamiento químico (ppm). En los experimentos de difusión molecular, se realizan en total 69 experimentos para cada una de las tres muestras preparadas. Estos 69 experimentos se encuentran divididos en tres grupos de 23 experimentos cada uno de ellos que se diferencian sólo en el tiempo de difusión (Δ). Para los primeros 23 experimentos, el Δ es 75 ms, para los segundos 23 experimentos es 100 ms y para los terceros 23 experimentos es 125 ms. Esto se hace así para corroborar que las medidas no se encuentran afectas por convección, y en consecuencia es de esperar que los tres valores de difusión obtenidos para cada uno de estos tres grupos de 23 experimentos, sea el mismo. Para las medidas de difusión se empleó la secuencia de Bruker *stegp1s1d* (Eco de Espín Estimulada, STE), con una duración del pulso de gradiente de desfase/reenfoco (δ) de 2,2 ms para el polímero de 450 Da, de 4,0 ms para el polímero de 5000 Da y de 3,5 ms para la mezcla de ambos. La intensidad del gradiente (G) se varía desde un 8% hasta un 96% en incrementos de 4%. También se introduce un pulso de homospoil que elimina la magnetización residual del plano transversal. Se utiliza la forma de gradiente (σ) *squared smooth* (SMSQ). El tiempo transcurrido (τ) después del gradiente de campo magnético es de 1 ms. El output resultante de un experimento de difusión molecular es una representación de la intensidad de la señal frente al G^2 . El factor de difusión es un parámetro corrector del tiempo de difusión obtenido de la resolución de la ecuación 9 y que tiene en cuenta la duración del gradiente (δ), el tiempo de difusión (Δ), la forma del gradiente (σ), la correlación con el disolvente y la secuencia empleada.

5. 4. SOFTWARE

El software utilizado para generar los tres espectros superpuestos en el apartado de implementación del algoritmo “FISTA *with backtracking*” (ver resultados y discusión) es el TopSpin de Bruker Biospin versión 3.2.

El software empleado para generar los espectros correspondientes es el programa MATLAB versión R2015a. Este programa es la abreviatura de MATrix LABoratory (laboratorio de matrices). Fue creado en 1984 por *The MathWorks*.¹⁵ Se trata de un software de análisis numérico y representación gráfica de fácil manejo muy usado en universidades y centros de investigación. La Universidad de Almería dispone de licencia para su uso, y ha sido el software seleccionado para implementar técnicas de optimización avanzada mediante algoritmos en este trabajo fin de grado. Permite crear funciones propias y programas especiales (denominados archivos-m). En los últimos años ha incluido muchas más capacidades, como la de programar directamente procesadores digitales de señal, crear código, etc. MATLAB tiene herramientas para crear interfaces de usuario gráfico mediante un entorno específico de desarrollo, que en MATLAB se denomina GUIDE. Esta herramienta es un kit de desarrollo de software para crear aplicaciones con interfaces gráficas de usuario, que son programas independientes que automatizan una tarea específica.¹⁵

6. RESULTADOS Y DISCUSIÓN

Los resultados de este trabajo se van a dividir en dos grandes bloques. El primer bloque corresponde con la programación de un visor de RMN para la correcta visualización de un espectro de ^1H -RMN monodimensional. El segundo bloque

corresponde a la implementación del algoritmo “FISTA *with backtracking*.” lo que nos debe proporcionar soluciones gráficas en las que se representen los coeficientes de difusión de los componentes de la muestra. En el caso de las muestras PPG-450 y PPG-500, deberá mostrar una única solución (un único coeficiente de difusión), mientras que, en la muestra compuesta por dos polímeros, deberá mostrarnos una solución con dos coeficientes de difusión.

6.1. PROGRAMACIÓN DE UN VISOR DE RMN PARA LA VISUALIZACIÓN DE UN ESPECTRO DE ^1H -RMN

En primer lugar, se procede a crear un entorno de desarrollo en el programa MATLAB, que es un área de trabajo que contiene una interfaz gráfica de usuario (GUIDE), que posteriormente permitirá visualizar el espectro de ^1H -RMN correspondiente de un determinado número de experimentos. Para ello, se tecldea en la ventana de comandos “guide” y se elige la opción de crear una interfaz gráfica de usuario en blanco (predeterminada), en el que proporciona un formulario nuevo para diseñar un programa. Este programa será, en este caso, un visor de ^1H -RMN. Se utiliza una plantilla predefinida por MATLAB (Figura 7).

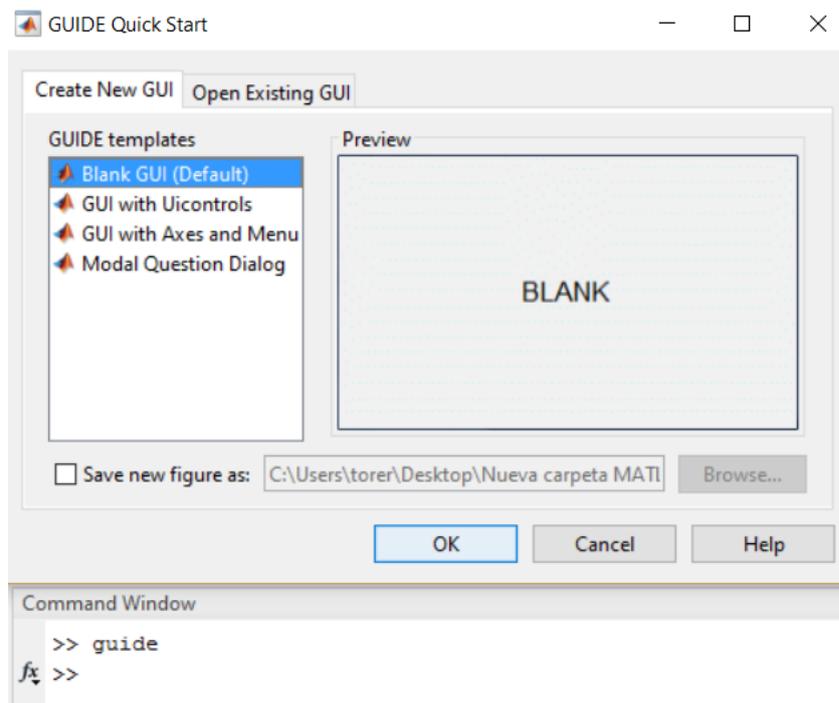


Figura 7. Creación de un entorno de desarrollo en el programa MATLAB.

El entorno de GUIDE presenta un área de diseño y un conjunto de componentes o controles “edit text” (crea un campo de texto que se modifica), “static text” (crea un letrero), “push button” (invoca a un evento inmediatamente), “check box” (indica el estado de una opción), “axes” (gráfica donde se representa el espectro), una barra de menú, etc. Esto se ilustra en la Figura 8.

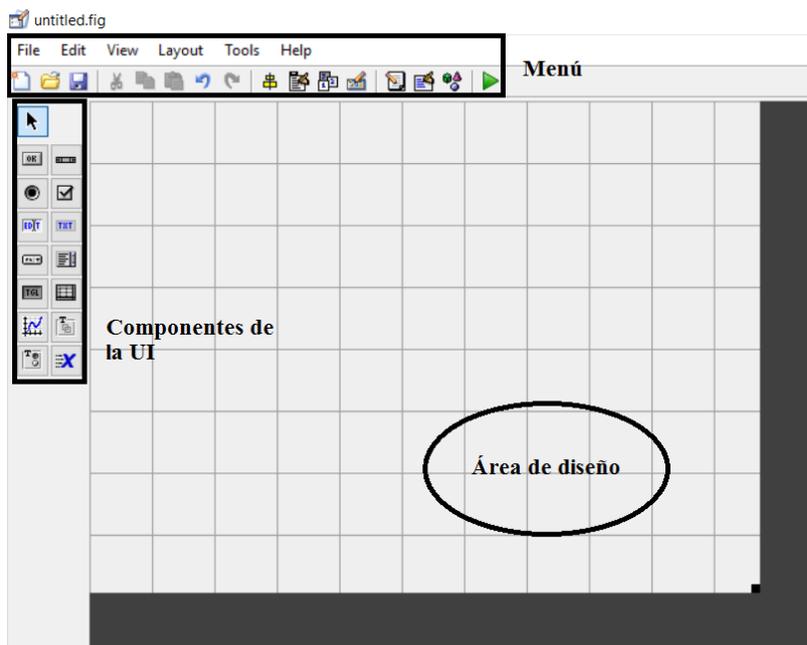


Figura 8. Graphical User Interface Development Environment (GUIDE).

De todas ellas se seleccionan “axes”, “check box”, “static text”, “edit text” y “push button”, como se observa en la Figura 9.

El componente “axes” permite visualizar el espectro en una representación, el componente “push button” permite establecer un botón para seleccionar la carpeta del experimento deseado, el componente “static text” muestra una cadena de texto dentro de una caja para que el usuario tenga mayor facilidad en interpretar la aplicación, el componente “edit text” establece un cuadro de texto para editar los números de las carpetas de los experimentos y el componente “check box”, es una opción que permite la superposición de varios espectros ^1H -RMN sin que se elimine el anterior.

Después de arrastrar estos elementos al área de diseño se guarda el archivo como NMRViewer, que se guardará automáticamente como una extensión .m y .fig en el directorio de trabajo y se abrirá un editor de texto (NMRViewer.m) con nueve funciones.

Estas nueve funciones son las siguientes:

- *varargout*: es el número variable de argumentos de salida y *varargin* es el número variable de argumentos de entrada. Esta función acepta un número indeterminado de argumentos tanto de entrada como de salida.
- *NMRViewer_OpeningFcn*: se encarga de leer y cargar la imagen. Todos los valores de las propiedades de los elementos (hObject) se almacenan en una estructura, los cuales son accedidos mediante un único y mismo identificador para todos. El identificador se asigna en `handles.output = hObject`; siendo el identificador a los datos *handles*. La sentencia `guidata(hObject, handles)`; se aplica para salvar los datos.
- *varargout = NMRViewer_OutputFcn(hObject, eventdata, handles)*: devuelve en una matriz los argumentos de salida.

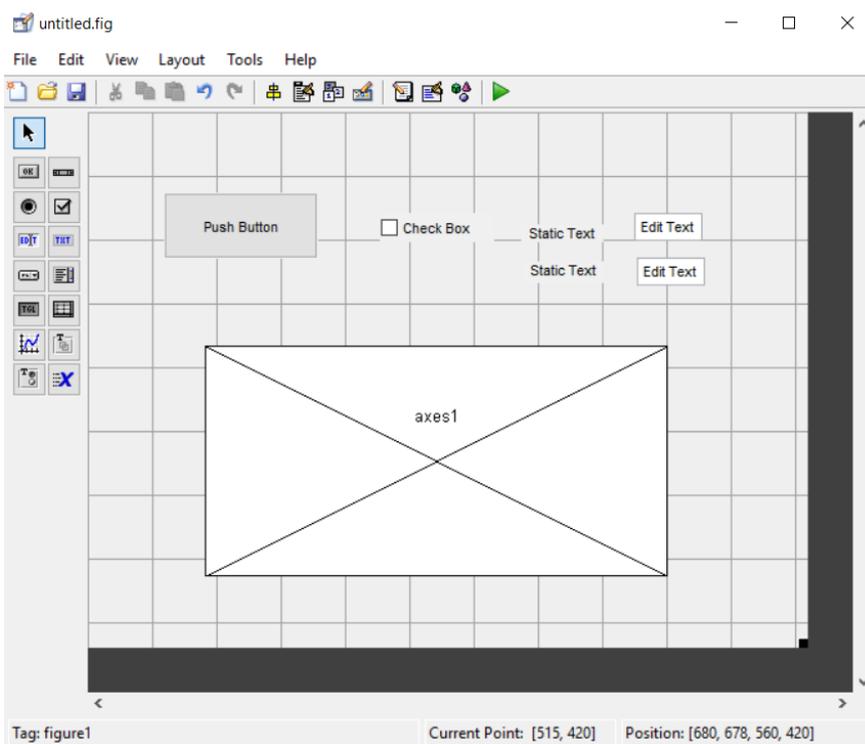


Figura 9. Proceso de creación de la aplicación VisorNMRFig.

- *edit1_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número del experimento donde empieza.
- *edit1_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.
- *edit2_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número en el que acaba el experimento.
- *edit2_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.
- *pushbutton1_Callback*: se ejecuta el evento al presionar el botón de selección de carpetas de experimentos.
- *checkbox1_Callback*: permite indicar el estado de una opción o atributo para ver el espectro con una superposición de los experimentos anteriores o poder borrar todas las representaciones que estaban antes y ver una nueva.

Se puede observar que cuatro de estas funciones generan directamente el *callback*, como puede apreciarse en el nombre de la función. Un *callback* es una función que se ejecuta como respuesta de un evento o acción predefinido por el usuario. Con el *callback* del botón, permite posicionarse en la parte del código que corresponda a esa subrutina, que se ejecutará cuando se realice una determinada acción (pulsar la tecla). El *callback* de la casilla de verificación se ejecutará cuando el usuario haga clic en esta casilla.

Una vez creadas estas funciones, se procede a modificar el nombre y ajustar el tamaño de cada uno de los elementos creados en la GUIDE. Para ello, hay que entrar dentro del GUIDE de la siguiente manera: se clikea con el botón derecho en el archivo de NMRViewer.fig, que se encuentra en el directorio de trabajo y se accede a Open in GUIDE. Ver Figura A.1 (sección de anexo).

Seguidamente, se selecciona cada elemento de la interfaz y se entra en el inspector de propiedades (que se encuentra en la barra del menú) y se cambian los siguientes controles. La imagen correspondiente a estos cambios puede apreciarse en la Figura A.2 (sección de anexo).

- “String” del botón: Select folder 1H NMR experiment
- “Tag” del botón: Select_Folder_NMR
- “String” de la caja de verificación: Keep the current 1H spectrum
- “Tag” de la caja de verificación: CheckBox_Keep_Spectrum
- “String” del primer texto estático: Experiment start at:
- “String” del segundo texto estático: Experiment finish at:
- “Tag” del primer texto editable: etExpStart
- “Tag” del segundo texto editable: etExpFinish

Se eliminan el “string” de los textos “edit text” para que no haya ningún texto que aparezca ya que en esta casilla se introducirán los números de los experimentos. También se cambia un control en todos los componentes para que se agranden cuando se maximice la ventana de la aplicación.

- Units: normalized

Después realizar estos cambios, se guarda el archivo .fig y se pulsa el botón *RUN* (ejecutar) o la tecla F5 y se observa la aplicación creada. Para contemplar la ejecución de la aplicación, ver Figura A.3 (sección de anexo). El resultado de la ejecución de la aplicación se observa en la Figura 10.

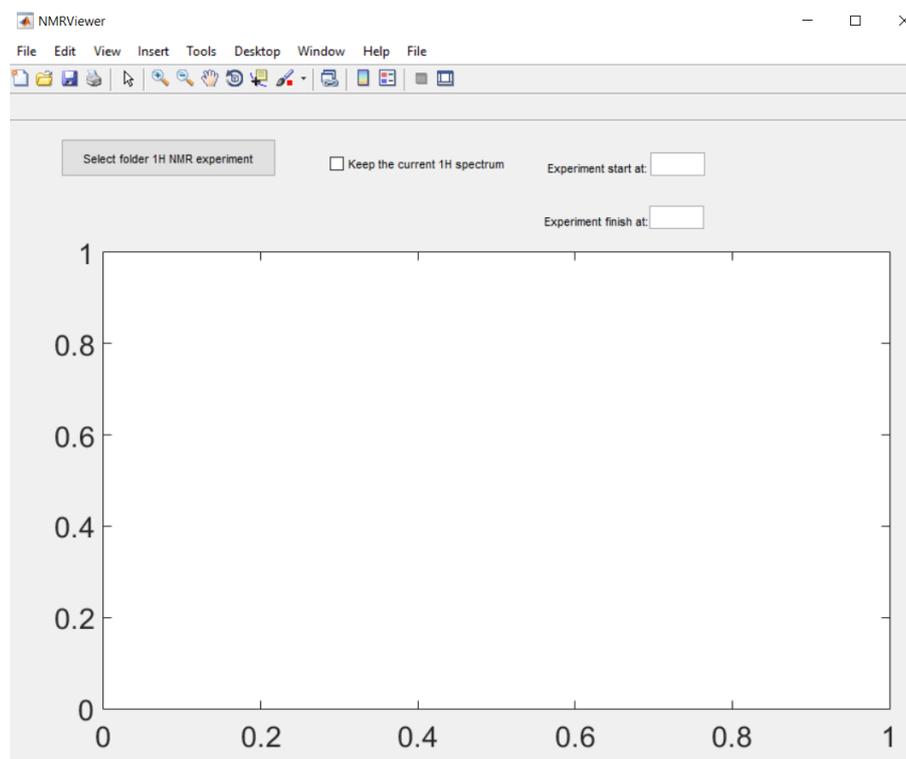


Figura 10. Creación de la aplicación NMRViewer.

Una vez creada la aplicación NMRViewer, se procede a implementar un código en un archivo nuevo (*readspectrum.m*) que, posteriormente se llamará en el archivo actual *NMRViewer.m* para que el programa pueda buscar los parámetros de procesado “procs” de los archivos de los experimentos a seleccionar, y el archivo “1r” que posteriormente permita visualizar el espectro. En el Código 1 se puede observar el código implementado.

```
function A = readspectrum(CurrentWorkingDirectory)
st = strcat(CurrentWorkingDirectory, 'procs');
if exist(st, 'file')==2;
    A.Procs = readnmrpar(st);
end
if A.Procs.BYTORDP == 0
    endian = 'l';
else
    endian = 'b';
end
st = strcat(CurrentWorkingDirectory, '1r');
[FID, MESSAGE] = fopen(st, 'r', endian);
if FID == -1
    disp(MESSAGE); error('NMRViewer: error to open file');
end
A.Data = fread(FID, 'int32');
fclose(FID);
A.Data = A.Data / (2^-A.Procs.NC_proc);
A.XAxis = linspace( A.Procs.OFFSET, ...
                    A.Procs.OFFSET-A.Procs.SW_p./A.Procs.SF, ...
                    A.Procs.SI)';
plot(A.XAxis, A.Data, 'LineWidth', 1.5);
xlabel('ppm');
ylabel('Intensity (a.u)');
set(gca, 'Xdir', 'reverse')
```

Código 1. Implementación de la función readspectrum.

En esta función, se localiza el archivo “procs” que se encuentra en el directorio de trabajo actual que, posteriormente se comentará cual es y se guarda en la variable “st”. Se introduce una condición, si existe esta variable y, si es de tipo archivo, devuelve un “2” y le pasa por parámetro la variable “st” a la función *readnmrpar*, que se describirá a continuación, y se guarda en “A.Procs”. La función *readnmrpar* es una función que contiene parámetros experimentales de RMN. Esta función se puede observar en el Código 2.

```
function P = readnmrpar(FileName)
A = textread(FileName, '%s', 'whitespace', '\n');
TypeOfRow = cell(length(A), 2);
R = {
    ...
    '^#\$*(.+)=\ \(\d\.\.\d+\) (.+)', 'ParVecVal' ; ...
    '^#\$*(.+)=\ \(\d\.\.\d+\)$', 'ParVec' ; ...
    '^#\$*(.+)=\ (.+)', 'ParVal' ; ...
    '^([\^$#].*)', 'Val' ; ...
    '^$\$$(.*)', 'Stamp' ; ...
    '^#\$*(.+)=', 'EmptyPar' ; ...
    '^(.+)', 'Anything' ...
};
```

```

for i = 1:length(A)
    for j=1:size(R,1)
        [s,t]=regexp(A{i},R{j,1}, 'start', 'tokens');
        if (~isempty(s))
            TypeOfRow{i,1}=R{j,2};
            TypeOfRow{i,2}=t{1};
            break;
        end
    end
end
end
i=0;
while i < length(TypeOfRow)
    i=i+1;
    switch TypeOfRow{i,1}
        case 'ParVal'
            LastParameterName = TypeOfRow{i,2}{1};
            P.(LastParameterName)=TypeOfRow{i,2}{2};
        case {'ParVec', 'EmptyPar'}
            LastParameterName = TypeOfRow{i,2}{1};
            P.(LastParameterName)=[];
        case 'ParVecVal'
            LastParameterName = TypeOfRow{i,2}{1};
            P.(LastParameterName)=TypeOfRow{i,2}{2};
        case 'Stamp'
            if ~isfield(P, 'Stamp')
                P.Stamp=TypeOfRow{i,2}{1};
            else
                P.Stamp=[P.Stamp ' ## ' TypeOfRow{i,2}{1}];
            end
        case 'Val'
            if isempty(P.(LastParameterName))
                P.(LastParameterName) = TypeOfRow{i,2}{1};
            else
                P.(LastParameterName) = [P.(LastParameterName), '
', TypeOfRow{i,2}{1}];
            end
        case {'Empty', 'Anything'}
    end
end

end
Fields = fieldnames(P);
for i=1:length(Fields);
    trystring = sprintf('P.%s = [%s];', Fields{i}, P.(Fields{i}));
    try
        eval(trystring);
    catch
        % Let the string P.(Fields{i}) be unaltered
    end
end
end

```

Código 2. Implementación de la función readnmrpar.²⁸

Seguidamente se averigua el valor del “endianness”, que es el formato en el que se almacenan los datos (orden de los bits). Una vez encontrado el valor del “endianness”, se localiza el archivo “1r” que se encuentra en el directorio de trabajo actual y se sobreguarda en la variable “st”.

Una vez encontrado el archivo “1r”, se abre y se lee en modo lectura con el formato “little endian” previamente averiguado y se guarda en la matriz “[FID,

MESSAGE]”. Ahora se implementa una condición, si la “FID” es igual a “-1”, se muestra en pantalla el siguiente mensaje de error: “NMRViewer”: error al abrir el archivo”. Una vez obtenido el valor de la “FID” distinto de “-1”, se lee la “FID” en enteros de “32 bits” y se guarda en la variable “A.Data” y luego se cierra la “FID”. Seguidamente, se introduce un factor de escala que depende de los parámetros de procesados específicos y se sobreguarda en la variable “A.Data”. “A.XAxis” es el nuevo eje x del espectro, una vez transformado los 64k (puntos del espectro) a escala de ppm. Finalmente, se representan las variables “A.XAxis” y “A.Data” con un ancho de línea de 1,5. Se etiquetan los ejes x e y, como ppm e Intensidad (a.u.). El comando “set(gca, ‘Xdir’, ‘reverse’)” hace que la dirección del eje x (ppm) esté invertida.

Por otra parte, se implementa la función *readnmrpar* en otro archivo diferente, ya que se hace una llamada de ésta dentro de la función *readspectrum*, que se incluirá como se mencionó en el archivo actual (“NMRViewer.m”). Es como decir que, una serie de funciones se encuentran dentro de otras por acción de llamada. No hace falta tenerlas desarrolladas dentro del archivo “NMRViewer.m”, tan sólo con llamarlas se ejecutan.

Básicamente, en el código 2 se ilustran un conjunto de algoritmos complejos de lectura que buscan una serie de parámetros de los experimentos de RMN usando técnicas de detección regular.

Una vez guardados estos dos archivos, se procede a implementar un algoritmo de lectura en el archivo actual (“NMRViewer.m”), que servirá para que el programa pueda buscar dónde se encuentran las carpetas de los experimentos. Esto se hace justo después de la función *Select_Folder_NMR_Callback(hObject, eventdata, handles)*.

A continuación, se inicializan las variables que se van a emplear para que por ahora no tomen ningún valor pero que el programa reconozca que existan. Esto puede observarse en el Código 3.

```
% --- Executes on button press in Select_Folder_NMR.
function Select_Folder_NMR_Callback(hObject, eventdata, handles)
% hObject      handle to Select_Folder_NMR (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
vSst = [];
vFst = [];
vSnum = [];
vFnum = [];
Z = [];
X = [];
FutureWorkingDirectory = [];
```

Código 3. Inicialización de las variables.

Posteriormente, se escribe la ruta del directorio de trabajo actual (ruta donde se encuentran las carpetas de los experimentos), y se toman como cadenas de carácter los valores que se introducen en las casillas de los textos editables y se guardan en las dos variables. Luego, se introduce una condición; si están vacías estas dos variables muestra en pantalla el mensaje de error: “error: Introduce el experimento inicial y el experimento final”. Si no, se concatena la ruta del directorio de trabajo

actual (donde se encuentran las carpetas de los experimentos) con la ruta “\pdata\1\” (donde se encuentran los archivos “procs” y “1r” de cada experimento) debido a que los experimentos de Bruker se guardan en carpetas con ese formato. Esto se puede apreciar en el Código 4.

```
CurrentWorkingDirectory = uigetdir();
vSst = get(handles.etExpStart, 'String');
vFst = get(handles.etExpFinish, 'String');

if (isempty(vSst) | isempty(vFst))
    fprintf('Error: Introduce the initial experiment and the final
experiment');
    CurrentWorkingDirectory =
strcat(CurrentWorkingDirectory, '\pdata\1\');
```

Código 4. Ruta del directorio de trabajo actual y lectura de las carpetas como cadenas de carácter.

Ahora se vuelve a implementar otra condición; si no existen las carpetas de los experimentos en el directorio de trabajo actual, muestra en pantalla el mensaje de error: “error: No hay experimentos en el directorio de RMN”. Una vez hecho esto, se programa la casilla de verificación. Al manipularla toma el valor “value” y se guarda en la variable “value”. Ahora se implementa otra condición; si la casilla de verificación es distinta de “value”, se borran todas las gráficas anteriores. Esto quiere decir que, si se hace clic en la casilla de verificación se mantienen guardados los espectros anteriores y si no, al seleccionar de nuevo otros experimentos se borran los anteriores. Por tanto, si es igual a “value”, se mantienen los espectros introduciendo el comando “hold on”. Seguidamente, se introduce la función *readspectrum* comentada anteriormente y se le pasa por parámetro el “CurrentWorkingDirectory” y se guarda en la variable estructura “A”, para que se ejecuten los comandos programados en el otro archivo. Esto se observa en el Código 5.

```
if ~exist(CurrentWorkingDirectory, 'dir')
    fprintf('Error: there is not experiments in the NMR
directory\n');
    return
end
value = get(handles.CheckBox_Keep_Spectrum, 'Value');
if ~(value)
    cla;
end
hold on;
A = readspectrum(CurrentWorkingDirectory);
```

Código 5. Verificación de las carpetas de experimentos y del checkbox.

Ahora se procede a transformar las cadenas de carácter de las variables que se crearon anteriormente “vSst” y “vFst” a valores numéricos y se guardan en las variables “vSnum” y “vFnum”. Una vez transformados estas variables, se implementa un algoritmo de lectura que va a ir buscando en las 23 carpetas de experimentos las subcarpetas correspondientes de los archivos “procs” y “1r” y se guarda en la variable “FutureWorkingDirectory” (que contendrá la ruta del

experimento en cuestión más el “\pdata\1”). Se implementa una condición; si no existen carpetas en el directorio de trabajo actual, muestra el mensaje de error: “error: No hay experimentos en el directorio de RMN”. Luego se escribe el comando “hold on” para mantener guardadas todas las rutas que posteriormente se representarán. Una vez obtenidas todas las rutas de las carpetas de los 23 experimentos se aplica la función *readspectrum* y se le pasa por parámetro la variable “FutureWorkingDirectory”.

Finalmente, se van obteniendo los valores de las matrices “Z” y “X” para cada experimento correspondiente a los valores de “A.Data” y “A.XAxis”, respectivamente, y se vuelve a inicializar la variable “A” para liberar memoria RAM del ordenador. Estas líneas de código se pueden observar en el Código 6.

```
else

    vSnum = str2num(vSst);
    vFnum = str2num(vFst);

    for i = 1:(vFnum - vSnum + 1)
        FutureWorkingDirectory =
strcat(CurrentWorkingDirectory, strcat('\', strcat(num2str(vSnum+i-
1), '\pdata\1\')));
        if ~exist(CurrentWorkingDirectory, 'dir')
            fprintf('Error: there is not experiments in the NMR
directory\n');
            return
        end
        hold on;
        A = readspectrum(FutureWorkingDirectory);
        Z(i,:) = A.Data;
        X(i,:) = A.XAxis;
        A = [];

    end
```

Código 6. Algoritmo de lectura e implementación de la función *readspectrum*.

Una vez hecho esto, se termina con la programación para poder visualizar perfectamente un conjunto de espectros de ^1H -RMN. El resultado de este trabajo se aprecia en la Figura 11. Para ello, se pulsa el botón RUN o la tecla F5 que hace que se ejecuten todas las funciones siguiendo el orden lógico programado.

También puede hacerse zoom a una zona concreta del espectro para poder observar con más nitidez la atenuación de la intensidad de la señal con el aumento del gradiente de campo magnético. Esto se observa en la Figura 12.

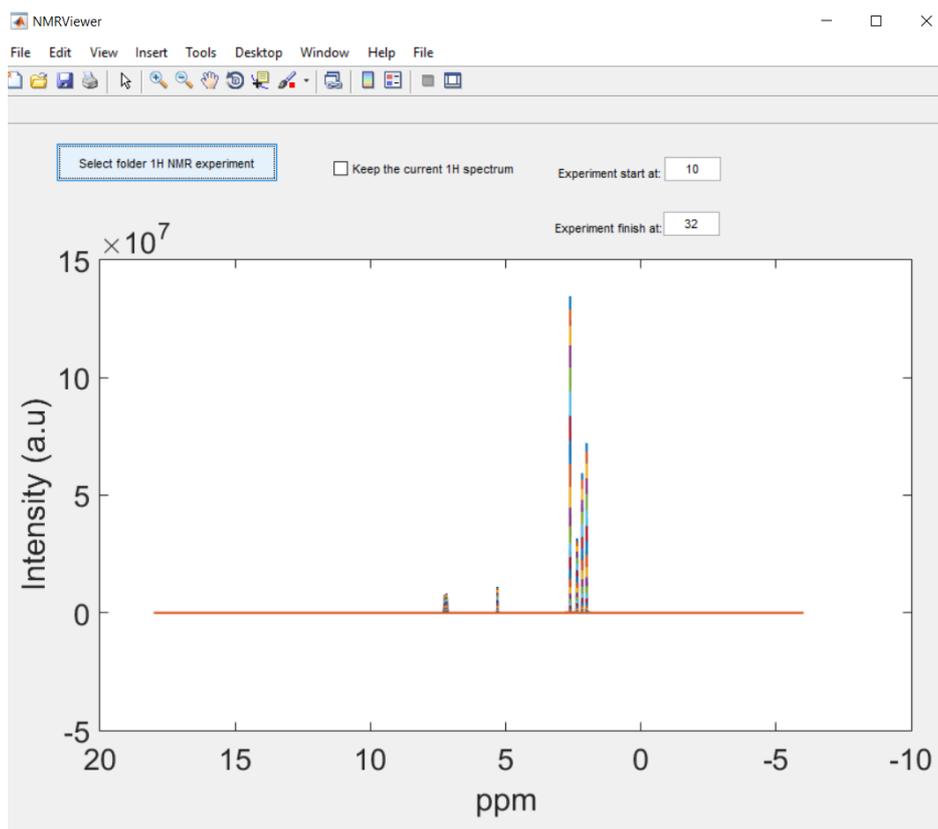


Figura 11. Representación de un conjunto de 23 experimentos de ^1H -RMN a diferentes gradientes.

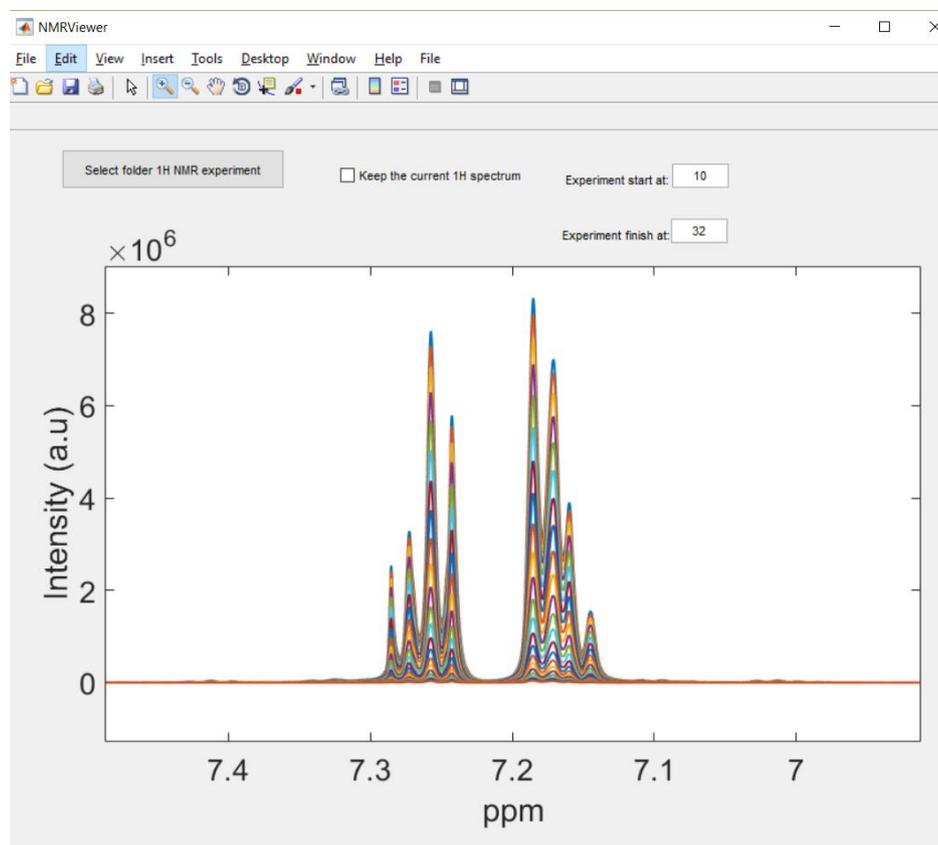


Figura 12. Efecto del zoom a la zona comprendida entre 7.1 y 7.3 ppm.

6.2. IMPLEMENTACIÓN DEL ALGORITMO FISTA WITH BACKTRACKING

Esta segunda parte del TFG consiste en seleccionar un punto del espectro monodimensional de ^1H -RMN del polímero propilenglicol (PPG) y ejecutar el algoritmo “FISTA *with backtracking*” para poder obtener el coeficiente de difusión (D) de las especies existentes en ese punto seleccionado.

Para ello, se han realizado tres experimentos de difusión para cada una de las tres muestras preparadas (PPG-450, PPG-5000 y PPG-Mix). Las dos primeras deberán mostrar coeficientes de difusión únicos y distintos entre sí, mientras que la tercera deberá mostrar los dos coeficientes en una misma medida. Esto se va hacer de esta manera para observar la capacidad que tiene este algoritmo para resolver los coeficientes de difusión de una mezcla de dos polímeros con pesos moleculares distintos. Tal y como se mencionó con anterioridad, el hecho de realizar tres series de experimentos por muestra es para asegurarnos que las medidas están realizadas en ausencia de convección, y que, por tanto, los coeficientes obtenidos son debidos exclusivamente al movimiento molecular del polímero, y no debido a la presencia de gradientes de temperatura.

Para poder ejecutar adecuadamente el algoritmo, primero se necesita programarlo. A continuación, se ilustrará con códigos el desarrollo del mismo junto a su explicación y discusión.

Una vez obtenido las atenuaciones del espectro de ^1H -RMN es necesario elegir un punto (x,y) dentro de la función *Select_Folder_NMR_Callback* para poder aplicar el algoritmo en ese punto. Esto se hace con la función *ginput* y los valores “x” e “y” se guardan en las variables “ppm” y “aux”. Debido a que el espectro está formado por puntos y no es una línea continua, se hace una comprobación por si no se eligió el punto correctamente. Por tanto, se crea un valor mínimo y un valor máximo a partir de este valor seleccionado. Se inicializa la variable “miElemento” que posteriormente se necesitará. Ahora se procede a introducir un ciclo *for* para que recorra todas las posiciones del vector “X(1,:)”. Se introduce una condición, si el valor de “ppm” está dentro del rango de los puntos mínimo y máximo generados anteriormente, se guarda la posición “i” del vector que genera ese valor en la variable “miElemento” y se termina el bucle. Por lo tanto, de una manera general, lo que se ha hecho es elegir la posición del “vector X(1,:)”, que denota todos los valores de la primera fila de la matriz “X”, y que esta posición genere un valor de “ppm” que esté comprendido entre los valores comentados anteriormente. Esto se puede entender fácilmente pensando que dentro de un vector que presenta unos determinados valores, existen posiciones o elementos que definen esos valores del vector. Esto se observa en el Código 7.

```
[ppm,aux] = ginput(1);
ppminf = ppm - 0.005;
ppmsup = ppm + 0.005;
miElemento = [];
for i = 1:length(X(1,:))
    if (ppminf <= X(1,i) && X(1,i) <= ppmsup)
        miElemento = i;
        break;
    end
end
```

Código 7. Selección de la posición i en el vector de posiciones X(1, :).

Ahora se introduce un algoritmo que permite elegir el valor del elemento que dé lugar al valor máximo de intensidad en una región concreta. Se crean dos variables nuevas a partir de la posición “i” elegida anteriormente: “MEinf” y “MESup”, y otras dos variables: “FsMax” y “miElementoMax”. Estas variables se igualan a cero para inicializarlas, es decir, para que el programa interprete de que existan, pero no toman ningún valor por el momento. Se genera un valor a partir de la posición “MEinf” en el vector “Z” y se guarda en la variable “FsMax”. Esto hace que tome como valor máximo de intensidad la posición “MEinf” en el vector “Z”. Ahora se vuelve a introducir otro ciclo *for* para que recorra el rango de “MEinf” hasta “MESup”. Los valores que generan las posiciones en el vector “Z” se guardan en la variable “Fs”. Se introduce una condición, cuando “Fs” sea mayor que “FsMax”, la variable “FsMax” toma el valor “Fs” y la posición del vector “Z” en el que se cumple esta condición se guarda en la variable “miElementoMax” y se termina el algoritmo. Esto se aprecia en el Código 8.

```
MEinf = miElemento - 25;
MESup = miElemento +25;
FsMax = 0;
miElementoMax = 0;
FsMax = Z(1,MEinf);
for i = MEinf:MESup
    Fs = Z(1,i);
    if (Fs > FsMax)
        FsMax = Fs;
        miElementoMax = i;
    end
end
```

Código 8. Selección de valor máximo de intensidad de la señal en una región concreta del espectro de ^1H -RMN.

Una vez obtenido la posición del valor de mayor intensidad “Fs” en una región concreta, se regenera el valor de la variable “Fs” haciendo que sea una matriz (23 filas x 1 columna). Para ello, se elige “miElementoMax” de las 23 filas de la matriz “Z” (elemento que proporciona el valor máximo de intensidad) y se divide por el valor máximo de este vector para generar un vector columna cuyo valor máximo es “1”. Después de esto, se programa para que se cree un círculo de tamaño “15” en donde proporcione el valor máximo de intensidad en la región previamente programada y el valor correspondiente de “ppm” en esa posición. Esto se hace para tener marcado el punto dónde se selecciona. Esto se observa en el Código 9.

```
Fs = Z(:,miElementoMax)/max(Z(:,miElementoMax));
u =
plot(X(1,miElementoMax),Z(1,miElementoMax), 'o', 'MarkerSize', 15);
```

Código 9. Creación del vector columna “Fs” y generación del círculo en la representación.

Para poder obtener el coeficiente de difusión es necesario conocer el vector columna “ G^2 ”. Por tanto, se procede a crear dos textos editables y dos textos estáticos para introducir el valor del gradiente inicial y la variación del gradiente de

la misma forma que se hizo para el experimento inicial y el experimento final. Para ello, se accede a la GUIDE y se arrastran los componentes mencionados: 2 “static text” y 2 “edit text”. Se entran en el inspector de propiedades y se cambian los siguientes controles:

- “String” del primer texto estático: Initial Gradient:
- “String” del segundo texto estático: Gradient Delta:
- “Tag” del primer texto editable: InitialG2
- “Tag” del segundo texto editable: VariationG2

Se elimina la propiedad “string” en ambos “edit text” para que no aparezca ningún texto. También se cambia un control en todos los componentes para que se agranden cuando se maximice la pantalla de la aplicación.

- Units: normalized

Después de realizar estos cambios, se guarda y se observa cómo se generan 4 funciones en el editor de texto:

- *InitialG2_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número del gradiente inicial.
- *InitialG2_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.
- *VariationG2_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número de la variación del gradiente.
- *VariationG2_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.

Ahora se procede a inicializar las variables que serán necesarios y a crear un algoritmo de lectura para leer el valor del gradiente y tratar con ellos. Los valores de los edit text se toman como cadenas de carácter y se guardan en las variables “InitialG2st” y “VariationG2st”, respectivamente. Como se observa en el Código 10.

```
G2 = [];  
InitialG2st = [];  
VariationG2st = [];  
InitialG2num = [];  
VariationG2num = [];  
InitialG2st = get(handles.InitialG2, 'String');  
VariationG2st = get(handles.VariationG2, 'String');
```

Código 10. Inicialización de variables G^2 .

Se convierten los valores que se introducen al formato “double” y se introduce una condición, si están vacías los “edit text”, devuelve en pantalla el siguiente mensaje de error: “error: Introduce el gradiente inicial y la variación del gradiente”. Si no están vacías, se definen las variables “VariationG2num” y el primer valor de “ G^2 ”. Como los valores están en tanto por ciento, se dividen estos valores entre

“100” y como es “ G^2 ” se eleva al cuadrado. Se vuelve a introducir un ciclo *for* que vaya desde el segundo valor de “ G^2 ” hasta la longitud del vector “ $X(:,i)$ ”. Se hace la raíz cuadrada del valor de “ G^2 ” del paso anterior (para eliminar el cuadrado) y luego se le suma el valor de variación del gradiente, y se vuelve a hacer el cuadrado. Este valor obtenido se actualiza en “ G^2 ” y se guarda como el siguiente valor en el vector fila de “ G^2 ”. Es decir, se van generando valores en saltos de “0.04” a partir de “ $(0.08)^2$ ”. Una vez generados los 23 valores de “ G^2 ” se termina el bucle. Las variables creadas se exportan al espacio de trabajo. Si no se genera correctamente la variable “ G^2 ” se muestra en pantalla el siguiente mensaje de error: “error: El gradiente inicial y la variación de gradientes deben ser números, y se exportan las demás variables y se termina la correspondiente función. A continuación, se muestra en el Código 11 las líneas de código correspondientes.

```

try
    InitialG2num = str2double(InitialG2st);
    VariationG2num = str2double(VariationG2st);
    if (isempty(InitialG2st) | isempty(VariationG2st))
        fprintf('Error: Introduce the initial gradient and the
variation gradient');
    else

        VariationG2num = VariationG2num/100;
        G2(1) = (InitialG2num/100)^2;
        for i = 2:length(X(:,i))
            G2(i) = (G2(i-1))^0.5;
            G2(i) = (G2(i) + VariationG2num)^2;
        end

        end
        assignin ('base', 'G2', G2);
        assignin ('base', 'Z', Z);
        assignin ('base', 'X', X);
        assignin ('base', 'Fs', Fs);
    catch
        fprintf('Error: The initial gradient and the variation
gradient must be numbers');
        assignin ('base', 'Z', Z);
        assignin ('base', 'X', X);
        assignin ('base', 'Fs', Fs);
    end

end
end

```

Código 11. Algoritmo de lectura de gradientes “ G^2 ” y exportación de las variables al espacio de trabajo.

Hasta aquí, todas las líneas de código se encuentran dentro de la función *Select_Folder_NMR_Callback*, que se generó cuando se creó el botón de seleccionar las carpetas de experimentos. Por tanto, se tienen guardadas las variables “ F_s ” (vector columna de intensidades) y la variable “ G^2 ” (vector fila de gradientes). Por lo tanto, para aplicar el algoritmo, es necesario crear otro botón y una serie de componentes que serán de utilidad a la hora de introducir los parámetros correspondientes para generar el espectro de coeficientes de difusión. Para ello, se entra en la GUIDE y se

crean 3 “static text”, 3 “edit text”, una casilla de verificación y un botón con los siguientes controles cambiados, que posteriormente conforme vayan apareciendo en el programa se irá explicando cada uno de ellos.

- “String2 del primer texto estático: Lambda Value:
- “String” del segundo texto estático: Diffusion’s factor:
- “String” del tercer texto estático: Iterations:
- “Tag” del primer texto editable: Lambda
- “Tag” del segundo texto editable: FD
- “Tag” del tercer texto editable: iter
- “String” de la casilla de verificación: Lin/Log
- “Tag” de la casilla de verificación: LinLog
- “String” del botón: Run ITAMeD
- “Tag” del botón: Run_ITAMeD

Los “edit text” no llevan cadenas de carácter y todos los componentes se agrandan de la misma forma:

- Units: normalized

Después de realizar las sucesivas modificaciones, se guarda y se crean ocho nuevas funciones en el editor de texto que se explican a continuación:

- *FD_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número del factor de difusión, que es un factor que se obtiene al resolver la ecuación diferencial de Stejskal-Tanner y corresponde con el experimento en cuestión. Este factor tiene en cuenta el tipo de gradiente (s), la duración del gradiente (δ), el tiempo de difusión (Δ), la correlación con el disolvente y la secuencia del experimento que se emplea.
- *FD_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación de este objeto.
- *iter_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el número de iteraciones que se emplea en el algoritmo.
- *iter_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.
- *Lambda_Callback(hObject, eventdata, handles)*: se ejecuta al teclear el valor del sparsity-promoting. Es un parámetro que mantiene el balance entre los datos experimentales y los datos teóricos.
- *Lambda_CreateFcn(hObject, eventdata, handles)*: es una propiedad que se lleva a cabo durante la creación del objeto.
- *LinLog_Callback(hObject, eventdata, handles)*: se ejecuta al hacer clic en el checkbox. Dependiendo de si se ejecuta o no este objeto, se generará un vector de coeficientes de difusión espaciado linealmente o logarítmicamente.
- *Run_ITAMeD_Callback(hObject, eventdata, handles)*: se ejecuta al presionar el botón de ejecución del algoritmo “FISTA *with backtracking*”.

Ejecutando el programa presionando la tecla F5 o cliqueando en *RUN* se observa la aplicación creada. Esto se ilustra en la Figura 13.

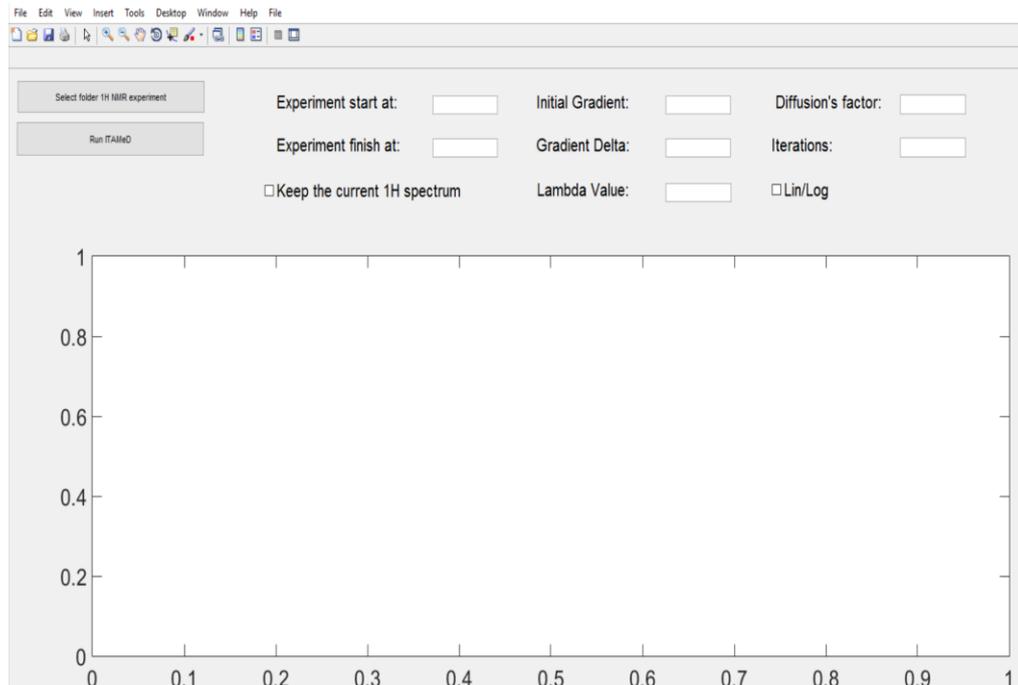


Figura 13. Creación de la aplicación final del Visor.

A partir de ahora, se definen los parámetros iniciales que serán necesarios dentro de la función *Run_ITAMeD* para la ejecución del algoritmo. Primero, se aplica a las dos variables “Fs” y “G²” la función “evalin” para ejecutar la expresión en el espacio de trabajo especificado. Seguidamente, se procede con la inicialización de los parámetros iniciales como se observa en el Código 12.

```
% --- Executes on button press in Run_ITAMeD.
function Run_ITAMeD_Callback(hObject, eventdata, handles)
% hObject      handle to Run_ITAMeD (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
Fs = evalin('base','Fs');
G2 = evalin('base','G2');
A_I1 = [];
A_I2 = [];
DI_space = [];
H = [];
iter = 10000;
inicio = 0.0000000001;
final = 20;
espacio = 1024;
diffusion_range = [inicio final espacio];
lambda = 1e-4;
lin_log = 0;
factor = [];
```

Código 12. Inicialización de los parámetros de entrada para el algoritmo.

Una vez definido todos los parámetros, se procede a implementar un algoritmo de lectura de los “edit text” y de la casilla de verificación, como se ilustra en el Código 13.

```
factorst = get(handles.FD, 'String');
if (isempty(factorst))
    factor = 1;
else
    factor = str2double(factorst);
end
iterst = get(handles.iter, 'String');
if (isempty(iterst))
    iter = 100000;
else
    iter = str2double(iterst);
end
lambdast = get(handles.Lambda, 'String');
if (isempty(lambdast))
    lambda = 1e-4;
else
    lambda = str2double(lambdast);
end

Value = get(handles.LinLog, 'Value');
if ~(Value)
    lin_log = 1;
else
    lin_log = 0;
end
[A_I1,A_I2,DI_space,H] =
ITAMED(iter,diffusion_range,Fs(1:end),G2(1:end),lambda,factor,lin_log);
```

Código 13. Algoritmos de lectura e implementación del algoritmo.

En el caso de introducir condiciones que no se cumplan, se pasa a la siguiente línea de código.

Si está vacía la variable “factorst”, el factor de difusión toma el valor de “1” y si no, toma el valor que se haya indicado en el cuadro de texto. Si está vacía la variable “iterst”, la iteración toma el valor de “100.000” y si no, toma el valor que se haya indicado.

Si está vacía la variable “lambdast”, toma el valor de “10⁻⁵” y si no, toma el valor que se indique en el edit text. Si no se cliquee la casilla de verificación, “lin_log” toma el valor de “1” (coeficientes de difusión espaciados linealmente) y si no, toma el valor “0” (coeficientes de difusión espaciados logarítmicamente). Una vez terminado esto, se implementa el algoritmo pasándole por parámetro todos los argumentos de entrada que se necesitan para la ejecución de esta. El algoritmo desarrollado puede observarse en el anexo (ver apartado Código “FISTA *with backtracking*”).

Una vez descrito el código implementado en este TFG, a continuación, se visualizarán un conjunto de espectros de ¹H-RMN con el programa TopSpin y seguidamente el espectro de coeficientes de difusión de las muestras PPG-450, PPG-500 y PPG-Mix con el programa MATLAB. La Figura 14 muestra la superposición de los tres espectros de ¹H-RMN para cada una de las muestras. El espectro en verde

(parte superior) pertenece al polímero de peso molecular 5000 Da. El espectro en rojo (parte intermedia) corresponde al polímero de peso molecular 450 Da. El espectro azul (parte inferior) pertenece a la mezcla de polímeros de pesos moleculares 450 y 5000 Da.

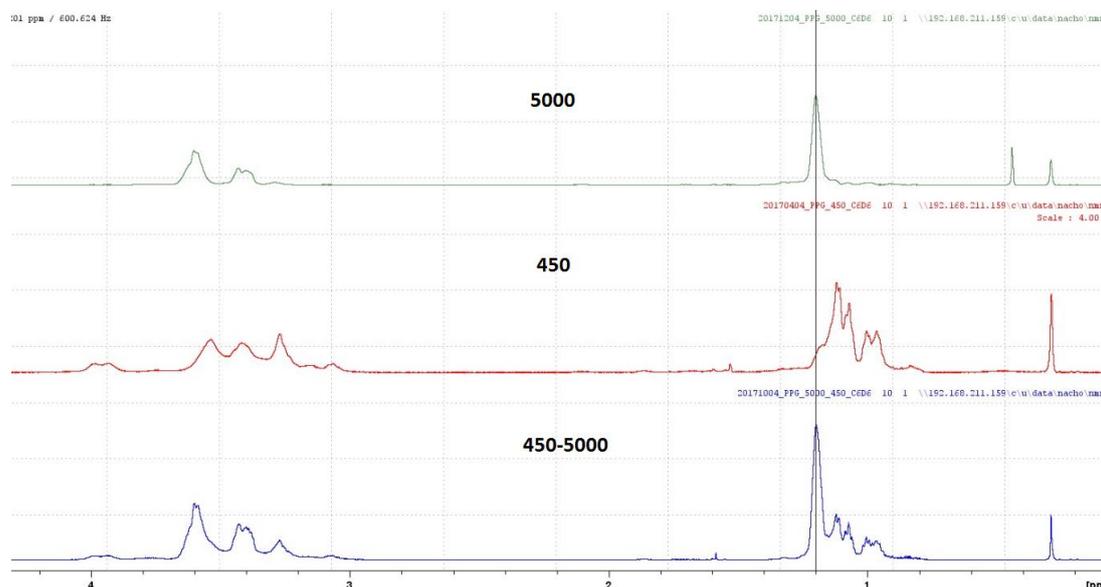


Figura 14. Superposición de espectros de las 3 muestras de polímero PPG con distintos pesos moleculares de ^1H -RMN.

Para cada muestra, se van a obtener 3 coeficientes de difusión que corresponden con las tres series de experimentos 10-32, 33-55, 56-78. Por tanto, el coeficiente de difusión promedio (D_{AV}) será la media aritmética de estas tres soluciones.

En la representación de coeficientes de difusión, la curva azul corresponde con el “FISTA *with constant stepsize*” (original)^{13,24} y la curva negra corresponde al propuesto en este TFG, “FISTA *with backtracking*”,²⁴. Con objeto de comparar ambos algoritmos se ha empleado el algoritmo desarrollado por Urbanczyk y colaboradores,¹³ previamente implementado en MATLAB por el grupo de investigación en el que se ha desarrollado este TFG.

Hay que destacar que el parámetro lambda y el número de iteraciones juegan un rol importante en el algoritmo propuesto. En nuestro caso, cada uno de estos valores se ha optimizado de forma manual para cada una de las muestras. En la primera serie de experimentos de la muestra PPG-450 se teclea que el experimento empieza en el 10 y finaliza en el 32, ya que hay que recordar que son 23 experimentos en total. Como gradiente inicial se elige el valor de 8% y que la variación del gradiente sea en incrementos de 4%. El factor de difusión empleado fue de 0,21270292. Los valores optimizados para Lambda y número de iteraciones fueron de $1 \cdot 10^{-3}$ y 50000, respectivamente.

Una vez realizado esto, se hace clic en el botón de seleccionar las carpetas de experimentos de ^1H -RMN, se pulsa en seleccionar carpeta y se procede a monitorizar un punto del espectro. En este caso, se selecciona el punto en donde $x = 3,418$ ppm (Figura 15) y se hace clic en “Run ITAMeD”. En la Figura 16 se puede observar la solución obtenida con un máximo centrado en el valor de D.

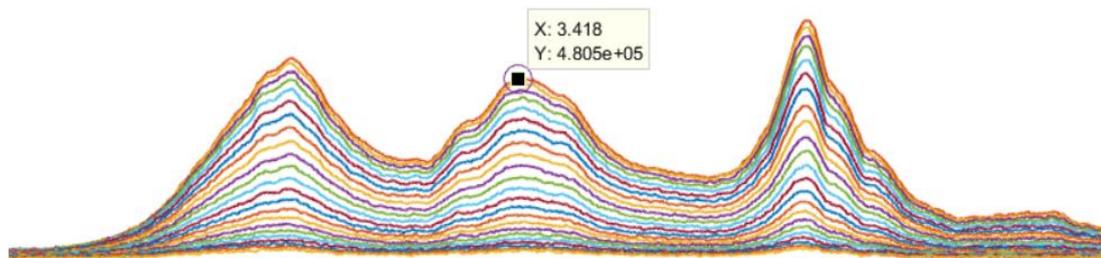


Figura 15. Monitorización de un punto para los experimentos 10-32 de la muestra PPG-450.

La curva azul se corresponde con el algoritmo “FISTA *with constant stepsize*” (original), mientras que la negra corresponde al “FISTA *with backtracking*” (propuesto por el TFG).

En ambos casos, el valor del coeficiente de difusión (D_1) es de $0,765 \cdot 10^{-9} \text{ m}^2/\text{s}$.

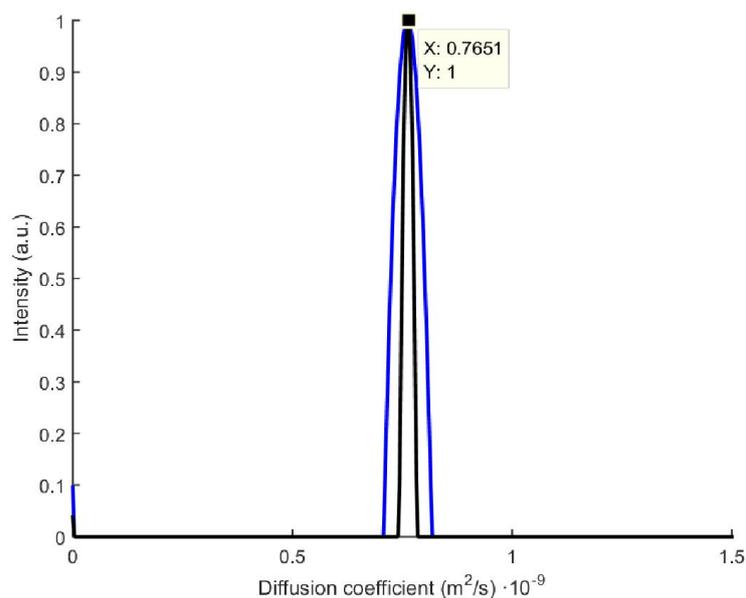


Figura 16. Representación del coeficiente de difusión para los experimentos 10-32 de la muestra PPG-450.

Seguidamente, se pretende obtener el coeficiente de difusión (D_2) de la segunda serie de experimentos para la misma muestra. Para ello, se cierra la aplicación y se vuelve a reiniciar. Se empieza en el experimento 33 y finaliza en el experimento 55 (23 experimentos en total) con un factor de difusión de 0,15917267. Los demás parámetros se mantienen constantes.

Una vez realizado esto, se hace clic en el botón Seleccionar la carpeta de experimentos de ^1H -RMN y se monitoriza el mismo punto del espectro ($x = 3,418 \text{ ppm}$) y se hace clic en “Run ITAMeD”. Se observa en la Figura 17 la solución de donde se deduce el coeficiente de difusión (D_2) de $0,753 \cdot 10^{-9} \text{ m}^2/\text{s}$.

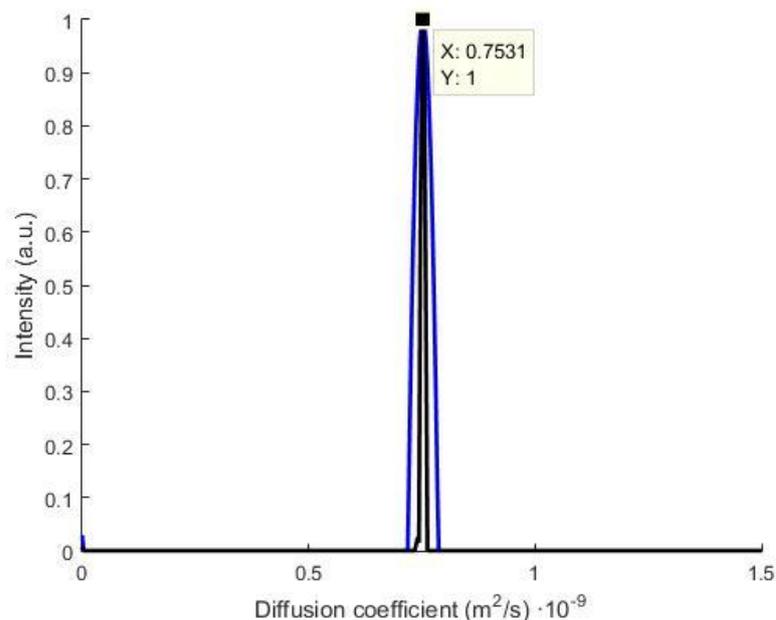


Figura 17. Representación del coeficiente de difusión para los experimentos 33-55 de la muestra PPG-450.

Para la última serie de experimentos de la muestra PPG-450, se vuelven a modificar las carpetas y parámetros anteriores y se vuelve a reiniciar la aplicación. El experimento inicial empieza en 56 y el experimento acaba en el 78 (23 experimentos totales). El valor del factor de difusión toma ahora el valor de 0,12716857. Los demás parámetros vuelven a permanecer constantes.

Una vez cambiado esto, se monitoriza el mismo punto en el espectro de ^1H -RMN ($x = 3,418$ ppm) y se vuelve a hacer “Run ITAMeD” para ejecutar el algoritmo sobre el mismo punto del espectro. La curva de esta solución puede observarse en la Figura 18.

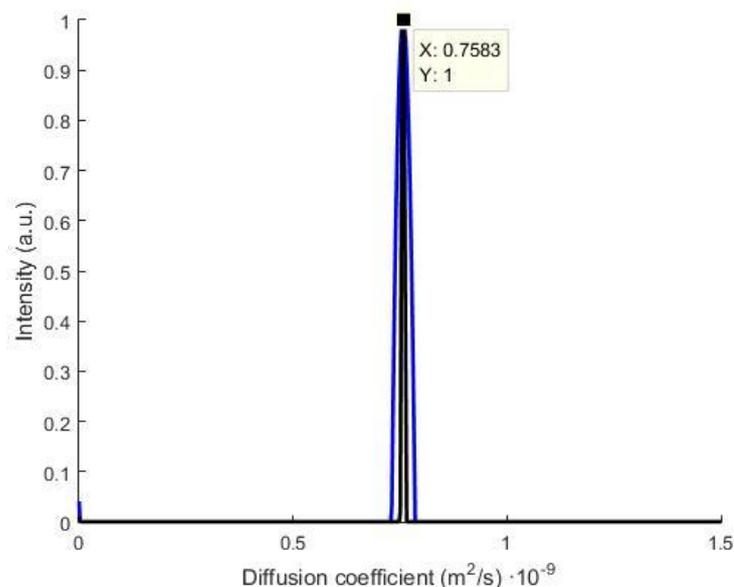


Figura 18. Representación del coeficiente de difusión para los experimentos 56-78 de la muestra PPG-450.

El valor del coeficiente de difusión (D_3) obtenido para para los experimentos del 56-78 de la muestra PPG-450 es $0,758 \cdot 10^{-9} \text{ m}^2/\text{s}$.

El valor del coeficiente de difusión promedio (D_{AV-450}) para esta muestra es por tanto $0,759 \cdot 10^{-9} \text{ m}^2/\text{s}$.

Para la muestra PPG-5000 se llevó acabo un procedimiento análogo al de la muestra anterior. Se vuelve a abrir la aplicación y se tecléa que el experimento inicial empieza en el 10 y finaliza en el 32 (23 experimentos totales). El valor del gradiente inicial es de 8% en incrementos del 4%. El factor de difusión toma en este caso un valor de 0,06481508. El valor de Λ toma el valor de $1 \cdot 10^{-2}$ y el número de iteraciones es 3000.

Después de esto, se pulsa el botón Selecciona la carpeta de experimentos de ^1H -RMN para seleccionar las carpetas indicadas y se monitoriza un punto, en este caso $x = 3,601$ ppm.

A continuación, se ilustra en la Figura 19 una imagen del punto monitorizado en el espectro de ^1H -RMN de los experimentos 10-32 de la muestra PPG-5000.

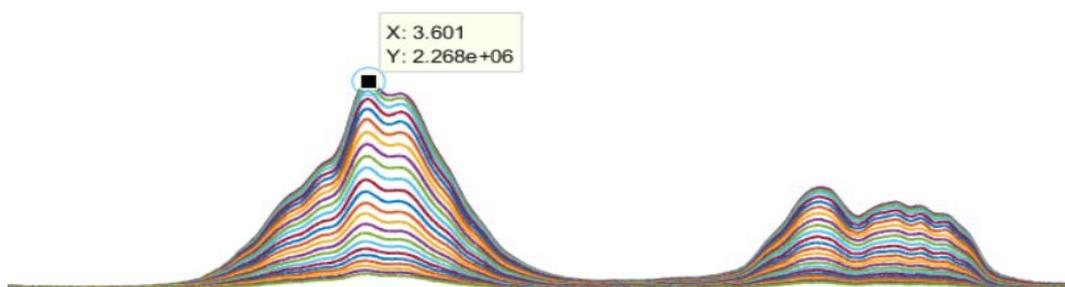


Figura 19. Monitorización de un punto del espectro de los experimentos 10-32 de la muestra de PPG-5000.

Para monitorizar el punto se pulsa el botón “Run ITAMeD” para ejecutar el algoritmo, obteniéndose la solución mostrada en la Figura 20. El valor del coeficiente de difusión (D_1) para para los experimentos 10-32 de la muestra PPG-5000 es $0,186 \cdot 10^{-9} \text{ m}^2/\text{s}$.

Seguidamente, se pretende obtener el coeficiente de difusión (D_2) y (D_3) de la segunda y tercera serie de experimentos. En cada caso, se cierra la aplicación y se vuelve a reiniciar. Para D_2 se empieza en el experimento 33 y finaliza en el experimento 55, mientras que para D_3 se comienza en el 56 y se termina en el 78.

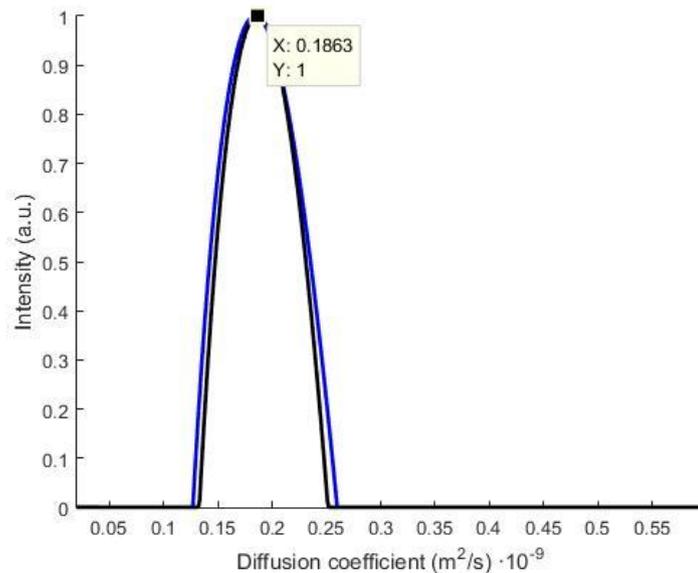


Figura 20. Representación del coeficiente de difusión de los experimentos 10-32 de la muestra PPG-5000.

Los factores de difusión empleados en ambos casos fueron $0,04801389$ y $0,03863687$, respectivamente. El Λ u número de iteraciones empleados fueron los mismos a los empleados en la serie de experimentos anterior ($1 \cdot 10^{-2}$ y 1200, respectivamente).

Las soluciones para ambas series se muestran en la Figura 21.

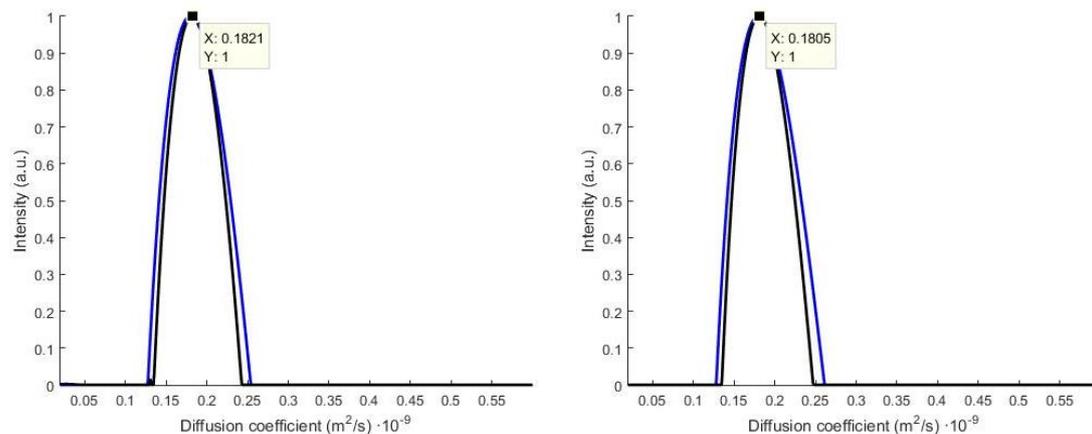


Figura 21. Representación del coeficiente de difusión de los experimentos 33-55 (izquierda) y 56-78 (derecha) de la muestra PPG-5000.

Los valores de coeficiente de difusión (D_2) y (D_3) para ambas series de experimentos fueron $0,182 \cdot 10^{-9} \text{ m}^2/\text{s}$ y $0,181 \cdot 10^{-9} \text{ m}^2/\text{s}$, respectivamente.

El valor del coeficiente de difusión promedio ($D_{\text{Av-5000}}$) para esta muestra es por tanto $0,183 \cdot 10^{-9} \text{ m}^2/\text{s}$.

En el tercer caso, se procede a determinar los coeficientes de difusión de la mezcla de polímeros PPG-Mix, constituida por los dos polímeros anteriores en una proporción 1:1. Los factores de difusión introducidos fueron $0,08448412$,

0,063138183, y 0,050432, para las tres series de experimentos. En todos ellos el valor de Λ empleado fue de $1 \cdot 10^{-2}$, el número de iteraciones de 10000. El punto monitorizado corresponde a la coordenada $x = 3,428$ ppm.

A modo de ejemplo, la Figura 22 muestra una superposición de esta mezcla de polímeros a distintos porcentajes de gradiente, indicando el punto seleccionado.

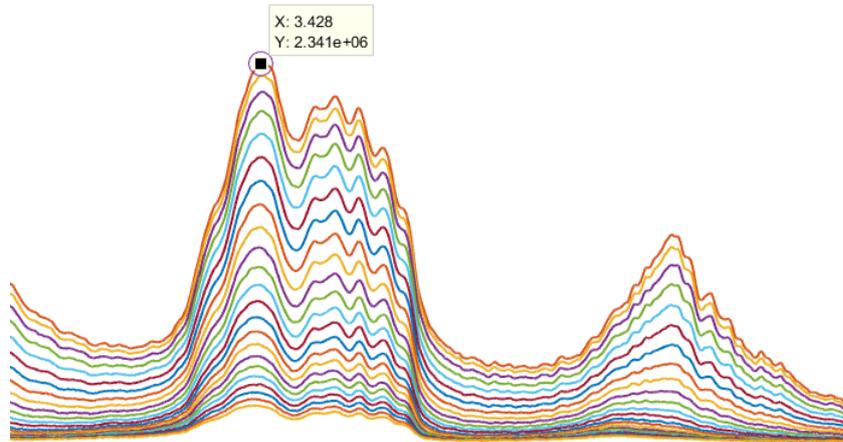


Figura 22. Monitorización de un punto del espectro de los experimentos 10-32 de la muestra de la mezcla de ambos polímeros.

De nuevo pulsando el botón “Run ITAMeD” se obtiene la solución para esta mezcla. En esta ocasión, la solución está constituida por dos máximos, uno centrado en $0,177 \cdot 10^{-9} \text{ m}^2/\text{s}$, y el otro centrado en $0,674 \cdot 10^{-9} \text{ m}^2/\text{s}$ (Figura 23).

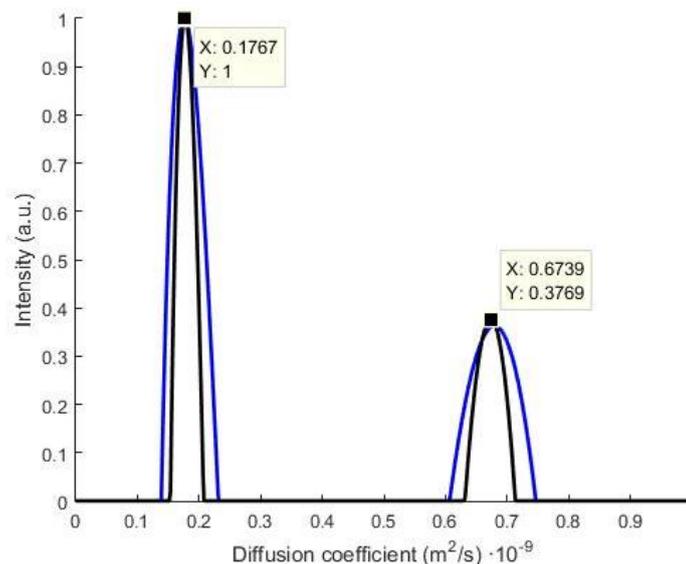


Figura 23. Representación del coeficiente de difusión de los experimentos 10-32 de la muestra de la mezcla de ambos polímeros: PPG-5000 ($D_{5000} = 0,177 \cdot 10^{-9} \text{ m}^2/\text{s}$) y PPG-450 ($D_{450} = 0,674 \cdot 10^{-9} \text{ m}^2/\text{s}$).

El primero de los máximos corresponde al polímero de mayor peso molecular (5000 Da), mientras que el segundo corresponde al de 450 Da. Al igual que en las muestras independientes, se llevó a cabo este mismo procedimiento para las dos

series de experimentos, con objeto de verificar la ausencia de convección. En ambos casos la solución consiste en dos curvas, cada una de ellas asociada a cada uno de los polímeros constituyentes de la mezcla.

Cuando se hacen los promedios de las tres series, los valores obtenidos son $0,176 \cdot 10^{-9} \text{ m}^2/\text{s}$ y $0,668 \cdot 10^{-9} \text{ m}^2/\text{s}$ para PPG-5000 y PPG-450, respectivamente.

Tabla 1. Coeficientes de difusión obtenidos y parámetros empleados.

Muestra	Experimentos	Coeficiente de Difusión (D)·10 ⁻⁹ (m ² /s)		Lambda (λ)	Iteraciones (k)	Factor de Difusión (FD)
450	10-32	0,765		10 ⁻³	50000	0,212702
	33-55	0,753		10 ⁻³	50000	0,159172
	56-78	0,758		10 ⁻³	50000	0,127168
5000	10-32	0,186		10 ⁻²	3000	0,064815
	33-55	0,182		10 ⁻²	1200	0,048013
	56-78	0,181		10 ⁻²	1200	0,038636
PPG-Mix	10-32	0,674	0,177	10 ⁻²	10000	0,084482
	33-55	0,669	0,175	10 ⁻²	10000	0,063138
	56-78	0,662	0,177	10 ⁻²	10000	0,050432

El valor del coeficiente de difusión para la muestra PPG-5000 es mucho menor que para la muestra PPG-450, debido a que presenta un tamaño mucho mayor lo que se traduce en un movimiento molecular mucho más lento.

Además, se observa que el valor del coeficiente de difusión de la muestra PPG-5000 se aproxima mucho al obtenido en la mezcla. Estos valores son, $0,183 \cdot 10^{-9} \text{ m}^2/\text{s}$ ($D_{\text{Av-5000}}$) y $0,176 \cdot 10^{-9} \text{ m}^2/\text{s}$ ($D_{\text{Av-5000-mix}}$), respectivamente. Sin embargo, el valor del coeficiente de difusión obtenido en la muestra PPG-450 se aleja más al obtenido en la mezcla. Estos valores son, $0,759 \cdot 10^{-9} \text{ m}^2/\text{s}$ ($D_{\text{Av-450}}$) y $0,668 \cdot 10^{-9} \text{ m}^2/\text{s}$ ($D_{\text{Av-450-mix}}$), respectivamente. Los errores relativos de la mezcla se muestran en la tabla 2. Los errores promedio cometidos para los parámetros empleados en el algoritmo implementado son del 11,7% y 3,6% para PPG-450 y PPG-5000 respectivamente.

Tabla 2. Errores relativos obtenidos en la mezcla Mix con respecto al valor de coeficiente de difusión promedio de cada polímero medido individualmente. a) El valor del D promedio considerado es $0,758 \text{ m}^2/\text{s}$. b) El valor de D promedio considerado es $0,183 \text{ m}^2/\text{s}$.

PPG-Mix	Experimentos	Coeficiente de Difusión (D)·10 ⁻⁹ (m ² /s)	Error (%)
450 ^a	10-32	0,674	11,2
	33-55	0,669	11,8
	56-78	0,662	12,7
5000 ^b	10-32	0,177	3,3
	33-55	0,175	4,4
	56-78	0,177	3,3

Estos errores de coeficiente de difusión están dentro de los resultados esperados, ya que errores cometidos en polímeros entorno al 10% o menores son considerados normales.^{29,30}

Las representaciones de los coeficientes de difusión de la mezcla han sido satisfactorias, los espectros proporcionan ambas bandas de coeficiente de difusión bien separadas entre sí, por lo que no se observan solapamientos ni artefactos. Además, el algoritmo “FISTA with backtracking” (propuesta de este TFG) presenta una banda de coeficiente de difusión más estrecha que el “FISTA with constant stepsize” (el algoritmo original). El número de iteraciones requerido para la convergencia es menor en “FISTA with backtracking” el “FISTA with constant stepsize”. En el caso de la resolución de la muestra de 450 Da (experimentos 10-32), el “FISTA with constant stepsize” (original) tarda 41 segundos mientras que el “FISTA with backtracking” (propuesto por este TFG) tarda 136 segundos. Por otro lado, el algoritmo “FISTA with backtracking” es más adecuado para problemas de difusión molecular de gran dimensionalidad ya que no tiene que calcular los autovalores del espacio de Hilbert.

El inconveniente del algoritmo actualmente programado es que se necesita saber de antemano el valor del factor de difusión. Este factor se calcula resolviendo la ecuación 11. Los parámetros a tener en cuenta en la solución es esta ecuación son el tiempo de difusión, el tiempo del gradiente de campo magnético y la forma del gradiente de campo magnético. Futuros TFGs tratarán de mejorar esta limitación con la implementación de un algoritmo que en paralelo calcule directamente este factor.

En la tabla A.1 del anexo, se encuentran tabulados los parámetros de adquisición empleados en los experimentos de difusión.

7. CONCLUSIONES

A continuación, se describen en cuatro apartados las conclusiones de este TFG incluyendo entre paréntesis las competencias adquiridas, y descritas específicamente en la sección de objetivos, para presentar una correlación directa entre conclusión y competencia, para poner de manifiesto el cumplimiento de todas ellas.

1. Se ha presentado un seminario acerca de la programación del visor de ¹H-RMN relacionado con este TFG (E-Q1, E-Q3 y E-Q5).
2. Se ha creado dicha aplicación (Visor ¹H-RMN) para visualizar uno y un conjunto de experimentos de ¹H-RMN monodimensionales mediante su programación en MATLAB, contenidos no impartidos en la titulación del grado de Química (E-Q1, E-Q2 y E-Q6).
3. Se han preparado y medido por Resonancia Magnética Nuclear de ¹H tres muestras de polímeros de tipo propilenglicol (PPG) preparadas en este TFG (E-Q4 y E-Q6).
4. Se ha implementado por primera vez el algoritmo “FISTA with backtracking” a decaimientos multiexponenciales obtenidos por RMN de ¹H, para obtener los correspondientes coeficientes de difusión en muestras constituidas por componentes puros, y para una muestra constituida por una mezcla de dos, lo que nos ha permitido evaluar la capacidad que presenta este algoritmo para resolver mezclas complejas en el que dos especies distintas presentan un solapamiento total en el espectro de RMN (E-Q2, E-Q3 y E-Q6).

Es necesario mencionar que, la primera parte de este TFG (creación del Visor ^1H -RMN) ha sido aceptado para su publicación en la revista internacional *Journal of Chemical Education* de la *American Chemical Society* (<http://pubs.acs.org/journal/jceda8>). El artículo se encuentra actualmente en edición por parte de la editorial, y se adjunta en el Anexo su versión aceptada.

8. BIBLIOGRAFÍA

- (1) Universidad de Burgos. Resonancia Magnética Nuclear (RMN). <http://www.ubu.es/parque-cientifico-tecnologico/servicios-cientifico-tecnicos/resonancia/resonancia-magnetica-nuclear-rmn> (acceso 27 de febrero de 2017).
- (2) Expósito-López, J. M.; Giménez-Martínez, J. J. Fundamentos Físicos de la Espectroscopia de RMN. *RMN para químicos orgánicos*, Universidad de Almería, Servicio de Publicaciones e Instituto de Estudios Almerienses: Almería, 1998; 5, 12-42.
- (3) Figura 1. Hipermedia - Spin e o momento magnético. http://www.google.es/search?q=momento+magnetico&espv=2&source=lnms&tbm=isch&sa=X&ved=0ahUKewjqrnvqcrSAhXCPRoKHWnLADgQ_AUIBigB&biw=1536&bih=759#imgrc=ujKm8WJ1_e-CWM (acceso 08 de marzo de 2017).
- (4) Feynman, R. P.; Leighton, R. B.; Sands, M. The Maxwell Equations. En *The Feynman Lectures on Physics* [Online]. Basic Books: New York, 1977; pp 220. <http://bayanbox.ir/view/5883831911734658469/The-Feynman-Lectures-on-Physics-Vol-II-Electromagnetism-and-Matter.pdf> (acceso 22 de junio de 2017).
- (5) Wikidot. Principios Básicos Resonancia Magnética. <http://sifreakthemighty.wikidot.com/mri:basics> (acceso 28 de febrero de 2017).
- (6) Espectroscopia de Resonancia Magnética Nuclear. <http://www.sinorg.uji.es/Docencia/FUNDOO/TEMA10FOO.pdf> (acceso 28 de febrero de 2017).
- (7) Ley de Ampère. https://es.wikipedia.org/wiki/Ley_de_Amp%C3%A8re (acceso 23 de junio de 2017). Wikipedia.
- (8) Corriente de desplazamiento. https://es.wikipedia.org/wiki/Corriente_de_desplazamiento (acceso 23 de junio de 2017). Wikipedia.
- (9) a) Fernández, I. Medidas de difusión. DOSY. En *Resonancia Magnética Nuclear*, Jiménez, M. A., Ed.; Grupo Especializado en Resonancia Magnética Nuclear: Madrid, 2011; 261-286. b) Edelman, D. C. Stejskal-Tanner Equation Derived in Full. *Harv. Bus. Rev.* **2010**, 88 (12), 205-214. c) Sinnaeve, D. The Stejskal-Tanner Equation Generalized for Any Gradient Shape—An Overview of Most Pulse Sequences Measuring Free Diffusion. *Harv. Bus. Rev.* **2010**, 88 (12), 39-65.
- (10) Stejskal, E. O.; Tanner, J. E. Spin Diffusion Measurements: Spin Echoes in the Presence of a Time-Dependent Field Gradient. *J. Chem. Phys.* **1965**, 42, 288-292.
- (11) Li, D.; Keresztes, I.; Hopson, R.; Williard, P. G. Characterization of Reactive Intermediates by Multinuclear Diffusion-Ordered NMR Spectroscopy (DOSY). *Acc. Chem. Res.*, 2009, **42**, 270-280.
- (12) a) Arrabal-Campos, F. M.; Oña-Burgos, P.; Fernández, I. Molecular weight prediction with no dependence on solvent viscosity. A quantitative pulse field gradient diffusion NMR approach. *Polym. Chem.* **2016**, 7, 4326-4329. b) Einstein, A. Eine neue bestimmung der moleküldimensionen. *Ann. Phys.* **1906**, 153, 135-153. c) Gierer, A.; Wirtz, K. Molekulare Theorie der Mikroreibung. *Zeitschrift für Naturforsch. A* **1953**, 8 (9), 532-538.
- (13) Bernin, D.; Kazimierzuk, K.; Kozminski, W.; Urbanczyk, M. Iterative Thresholding Algorithm for Multiexponential Decay Applied to PGSE NMR Data. *Anal. Chem.* **2013**, 85, 1828-1833.
- (14) a) Kharlamov, S. V.; Latypov, S. K. Modern Diffusion-Ordered NMR Spectroscopy in Chemistry of Supramolecular Systems: The Scope and Limitations. *Russ. Chem. Rev.*, 2010, **79**, 635-653. b) Macchioni, A.; Ciancaleoni, G.; Zuccaccia, C.; Zuccaccia, D. Determining accurate molecular sizes in solution through NMR diffusion spectroscopy. *Chem. Soc. Rev.* **2008**, 37 (3), 479-489. c) P. Flory, Principles of Polymer Chemistry, Cornell University, New York, 1953. (d) S. F. Edwards and M. Doi, The Theory of Polymer Dynamics, Oxford University Press, New. e) Pollack, L. G. Atomic test of the Stokes-Einstein law: Diffusion and solubility of Xe. **1981**, 23, 1-4.
- (15) File exchange. https://es.mathworks.com/matlabcentral/fileexchange/?s_tid=gn_mlc_fx (acceso 16 de marzo de 2017). (Licencia UAL).

(16) Ruelas, D. Definiciones de programación. <http://elblogderuelasdelaprogramacion.blogspot.com.es/2014/02/definiciones-de-programacion.html> (acceso 02 de marzo de 2017).

(17) Rodríguez-Sala, J. J.; Santamaría-Arana, L.; Rabasa-Dolado, A.; Martínez-Bonastre, O. Técnicas de representación de algoritmos. En *Introducción a la programación: Teoría y práctica* [Online]; Club Universitario: Alicante, 2003; pp 17. <https://books.google.es/books?id=nLMJsInMyBwC&printsec=frontcover&dq=que+es+la+programacion&hl=es&sa=X&ved=0ahUKEwjmkpH236HTAhXH2xoKHQnKBpMQ6AEIHjAB#v=onepage&q&f=false> (acceso 13 de abril de 2017).

(18) Arrabal-Campos, F. M.; Cortés-Villena, A.; Fernández, I. Building “My First NMRViewer”: A Project Incorporating Coding and Programming Tasks in the Undergraduate Chemistry Curricula. *J. Chem. Educ.* **2017**, *6*, aceptado en prensa.

(19) a) Buydens, L.M.C.; Huo, R.; van Duynhoven, J.; Wehrens, R. Assesment of techniques for DOSY NMR data processing. *Anal. Chim. Acta.* **2003**, *490*, 232. b) Antalek, B. Using pulsed gradient spin echo NMR for chemical mixture analysis: How to obtain optimum results. *Concepts Magn. Reson.* **2002**, *14* (4), 225–258. c) Van Gorkom, L. C. M.; Hancewicz, T. M. J. *Magn. Reson.* 1998. d) Provencher, S.W.; Vogel, R.H. Regularization techniques for inverse problems in molecular biology in: *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, eds. P. Deuflhard & E. Hairer (Birkhäuser, Boston, 1983), pp. 304-319.

(20) Provencher, S. CONTIN: A general purpose constrained regularization program for inverting noisy linear algebraic and integral equations. *Comput. Phys. Commun.* **1982**, *27*, 229-242.

(21) Mendoza-Domínguez, A.; Vanoye-García, A. Y. Revisión de Métodos de Regularización Directa y sus Aplicaciones en las Ciencias Atmosféricas. Presentado en el 41^a Congreso de Investigación y Desarrollo Tecnológico de Monterrey, Monterrey, Ciudad de México, octubre, 2010.

(22) Delsuc, M. A.; Malliavin, T. E. Maximum Entropy Processing of DOSY NMR Spectra. *Anal. Chem.* **1998**, *70*, 2146-2148.

(23) Daubechies, I.; Defrise, M.; De Mol, C. An iterative Thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math.* **2004**, *57*, 1413-1457.

(24) Beck, A.; Teboulle, M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. Imaging Sci.* **2009**, *2*, 183-202.

(25) Wthres. <https://es.mathworks.com/help/wavelet/ref/wthres.html> (acceso 05 de julio de 2017).

(26) a) Bai, X.; Goldfarb, D.; Scheinberg, K. Fast First-Order Methods for Composite Convex Optimization with Backtracking. *Found. Comput. Math.* **2014**, *14*, 389-417. B) Nesterov, Y. Accelerating the cubic regularization of Newton’s method on convex problems. *Math. Program.* **2008**, *112* (1), 159–181.

(27) Fichas Internacionales de Seguridad Química. <http://www.insht.es/InshtWeb/Contenidos/Documentacion/FichasTécnicas/FISQ/Ficheros/0a100/nspn0015.pdf> (acceso 20 de abril de 2017).

(28) RBNMR. <https://es.mathworks.com/matlabcentral/fileexchange/40332-rbnmr> (acceso 30 de marzo de 2017).

(29) Fleischer, G. The effect of polydispersity on measuring polymer self-diffusion with the n.m.r. pulsed field gradient technique. *Polymer (Guildf).* **1985**, *26* (11), 1677–1682.

(30) Gong, X.; Hansen, E. W.; Chen, Q. A simple access to the (log/normal) molecular weight distribution parameters of polymers using PGSTE NMR. *Macromol. Chem. Phys.* **2011**, *212* (10), 1007–1015.

9. ABREVIATURAS Y ACRÓNIMOS

TFG: Trabajo de Fin de Grado

RMN: Resonancia Magnética Nuclear

¹H-RMN: Resonancia Magnética Nuclear de protón

rf: Radiofrecuencia

P: Momento angular intrínseco

μ : Momento magnético nuclear

I: Número cuántico del momento de espín angular

γ : Constante giromagnética o magnetogírica

^1H : Isótopo de protón
 ^{12}C : Isótopo de carbono
 ^{13}C : Isótopo de carbono
A: Número másico
Z: Número atómico
 B_0 : Campo magnético externo
 ω_0 : Frecuencia angular
 ν_L : Frecuencia de Larmor
 \vec{M}_0 : Vector Magnetización neto
 T_1 : Tiempo de relajación longitudinal
 T_2 : Tiempo de relajación transversal
FID: Free Induction Decay
 ν : Frecuencia
 k_B : Constante de Boltzmann
 c : Factor de microficción
 r_{S1} : Radio del disolvente
 r_H : Radio hidrodinámico
 $f_S(a, b)$: Factor de forma
 μ : Viscosidad
A: Constante de proporcionalidad
 α : Medida de dimensión fractal
 β : Interacción molecular
 δ : Desplazamiento químico
TMS: Tetrametilsilano
RMI: Resonancia Magnética de Imágenes
D: Coeficiente de difusión
PGSE: Pulsed Field Gradient Spin Echo
STE: Eco de Espín estimulada
 $A(D)$: Distribución de coeficientes de difusión
 G_z : Canal de pulso de gradiente en el eje z
G1: Gradiente de desfase
G2: Gradiente de reenfoque
 Δ : Tiempo de difusión
 δ : Duración del gradiente
Gp: Gradiente de purga
MATLAB: MATrix LABoratory
SPLMOD: SPLine MODel
CONTIN: Coeficiente de difusión CONTINuo
DECRA: Direct Exponential Curve Resolution Algorithm
MCR: Multivariate Curve Resolution
GRAM: Generalized Range Annihilation Method

ILT: Inverse Laplace Transform

ITAMeD: Iterative Thresholding Algorithm for Multiexponential Decay

MaxEnt: Maximum Entropy

ISTA: Iterative Shrinkage Thresholding Algorithm

FISTA: Fast Iterative Shrinkage Thresholding Algorithm

M_w : Peso molecular promedio

PPG: Poli(propilenglicol)

THF: Tetrahidrofurano

τ : Tiempo transcurrido después del pulso de gradiente de campo magnético

G: Intensidad del gradiente de campo magnético

σ : Forma del gradiente de campo magnético

GUIDE: Graphical User Interface Development Environment

D_{AV} : Coeficiente de difusión promedio

$\vec{\nabla}$: Operador diferencial vectorial

\vec{H} : Campo magnético

\vec{J} : Flujo de corriente eléctrica

μ_0 : Permeabilidad magnética

\vec{D} : Corriente de desplazamiento

ϵ_0 : Permitividad

$\partial\phi$: Variación del campo eléctrico

ANEXO

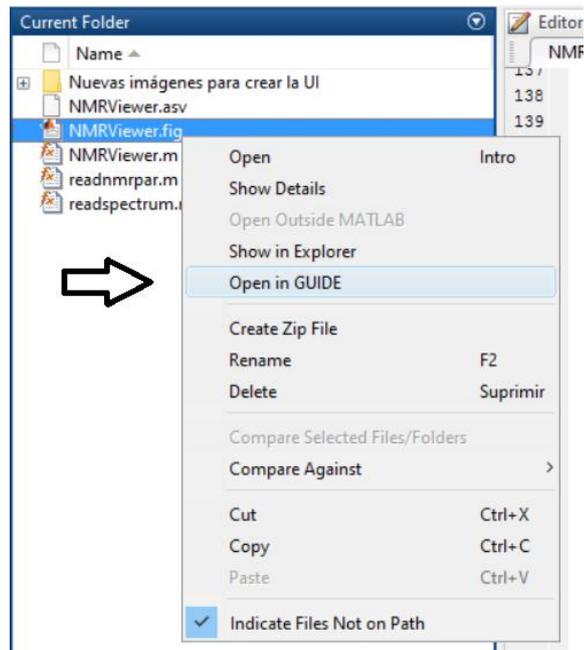


Figura A.1. Acceso a la GUI.

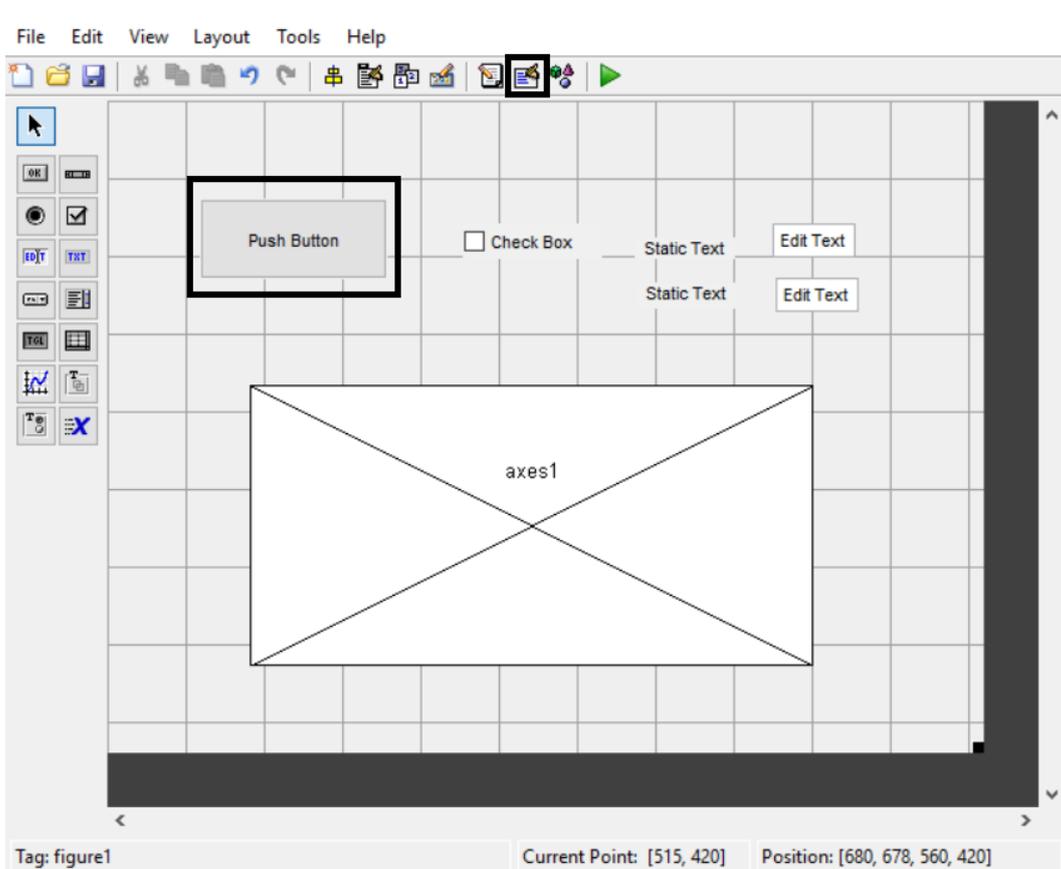


Figura A.2. Cambio de propiedades de los elementos.

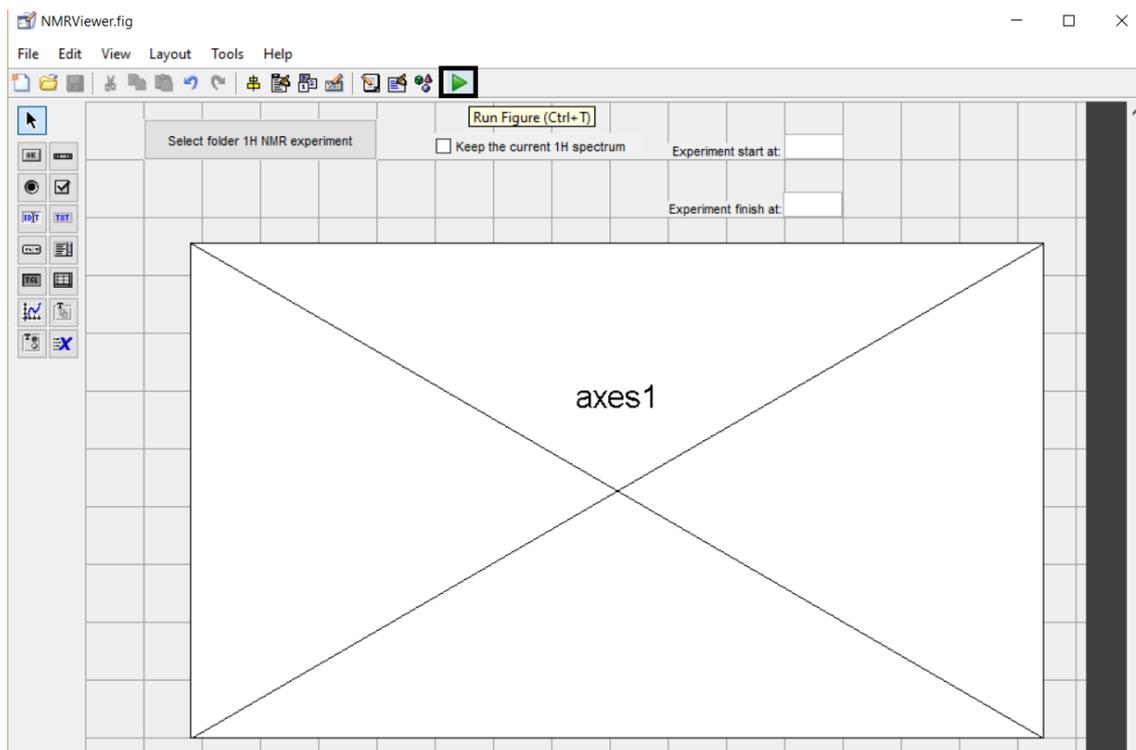


Figura A.3. Ejecución de la interfaz gráfica de usuario.

Tabla A.1. Tabulación de los factores experimentales para la obtención de los espectros de difusión.

Muestra	Experimentos	Duración del pulso de rf (μ s)	Potencia del pulso de rf (W)	Duración del gradiente (δ) (ms)	Tiempo de difusión (Δ) (ms)
450	10-32	10,05	9	2,2	75
	33-55	10,05	9	2,2	100
	56-78	10,05	9	2,2	125
5000	10-32	10,05	9	4,0	75
	33-55	10,05	9	4,0	100
	56-78	10,05	9	4,0	125
mezcla	10-32	9,7	9	3,5	75
	33-55	9,7	9	3,5	100
	56-78	9,7	9	3,5	125

Código FISTA

```
function [A_final] =
fista(A,integration_corrected,lambda,matA1,iterk,D_scale)
    tmax = 1/(2*norm(matA1'*matA1));
    t = 1/max(2*eig(matA1'*matA1));
    y = A';
    x1 = y;
    x = A';
    s = 1;
    figure;
    hold on;
cont = 0;
    h_old=plot(D_scale,x/max(x),'b','LineWidth',2);
    sr =
ShrinkageOperator(y,integration_corrected,matA1,1/t,lambda,tmax);
    not_zero = size(find(x));
    x1 = max(sr',0);
    s1 = (1+sqrt(1+4*s^2))/2;
    y = x1+(s-1)/(s1)*(x1-x);
    s = s1;
    x = (x1);
    if cont == 50
        h = plot(D_scale,x/max(x),'b','LineWidth',2);
        delete(h_old);
        h_old = h;
        drawnow;
        cont = 0;
    end
    cont = cont +1;
end
display('Fin ITAMeD');
A_final = max(x,0);
end
```

Código FISTA with backtracking

```
function [A_final] =
fistabt(A,integration_corrected,lambda,matA1,iterk,n,lo,D_scale)
    y = A';
    x1 = y;
    x = A';
    s = 1;
    ith = 1;
    F = [];
    Q = [];
    L_1 = lo;
    L = [];
    Li = [];
    sr = [];
    hold on;
    h_old=plot(D_scale,x/max(x),'k','LineWidth',2);
    xline = 1:1:length(y);
    cont = 0;
    for k = 1:iterk
        for j = 1:1000
            ith = j;
            L = n^ith*L_1;
            sr =
ShrinkageOperator(y,integration_corrected,matA1,L,lambda,tmax);
```

```

        F =
L1FunctionCost(integration_corrected,sr,matA1,lambda);
        Q =
QuadraticFunctionModel(integration_corrected,sr,y,matA1,lambda,L);
        if (F<=Q)

                L_1 = L*rand();
                break;
        end

    end
    sr =
ShrinkageOperator(y,integration_corrected,matA1,L,lambda,tmax);
    not_zero = size(find(x));
    x1 = sr';
    s1 = ((1+sqrt(1+4*s^2))/2)^(1);

    y = x1+1*((s-1)/(s1))*(x1-x).^1;
    s = s1;
    x = (x1);
    Li(k) = (L);
    if cont == 75
        h = plot(D_scale,max(x1/max(x1),0),'k','LineWidth',2);
        delete(h_old);
        h_old = h;
        drawnow;
        cont = 0;
    end
    cont = cont+1;
end
for i = 1:size(x,1)
    if x(i) < 0

        x(i) = 0;
    end
end
display('Fin ITAMeD');
A_final = x;

end

```

Código ITAMeD

```

function [A_final1,A_final2,D_scale,H] = ITAMED
(iter,diffusion_range,II,KK,lambda,factor,lin_log)
if lin_log == 1
    D_scale = linspace(diffusion_range(1),
diffusion_range(2),diffusion_range(3));
else
    D_scale = logspace(log10(diffusion_range(1)),
log10(diffusion_range(2)),diffusion_range(3));
end

testsize = size(II);
if testsize(1) ~= 1
    data.II = II';
else
    data.II = II;
end

```

```

testsize = size(KK);
if testsize(1) ~= 1
    data.KK = KK';
else
    data.KK = KK;
end
data.NmrData.Arraydim = size(data.II,2);
matA1 = matA_if_nofile(D_scale,data);
H = matA1;
A = zeros(size(D_scale));
tic
[A_final1] = fista(A,data.II',lambda,matA1,iter,D_scale*factor);
toc
tic
[A_final2] =
fistabt(A,data.II',lambda,matA1,iter,1.1,0.1,D_scale*factor);
toc
display('Fin')
end

```

Código Norma L1

```

function [value] = L1FunctionCost(Fs,x,H,lambda)
value = norm(x*H'-Fs')^2+lambda*norm(x,1);

```

Código matA_if_nofile

```

function [mata] = matA_if_nofile(D,data)
size_D = size(D);
for i = 1:size_D(2)
    u = 0;
    for j = 1: data.NmrData.Arraydim
        u = u+1;
        mata(j,i) = exp(-D(i)*data.KK(u));
    end
end
end

```

Código Función cuadrática

```

function [value] = QuadraticFunctionModel(Fs,x,y,H,lambda,L)
x_y = x-y';
value = norm(y'*H'-Fs')^2 + lambda*norm(x,1) + x_y*(2*H'*(H*y-
Fs))+L/2*norm(x_y)^2;

```

Código Operador de Shrinkage

```

function [x] = ShrinkageOperator(x,Fs,H,L,lambda,tmax)
soft_thresholding = [];
t = 1/(L);
c=x-2*1*t*H'*(H*x-Fs);
soft_thresholding = wthresh(c,'s',lambda*t)';
x = soft_thresholding;

```

Código función vline

```
function hhh=vline(x,in1,in2)

if length(x)>1 % vector input
    for I=1:length(x)
        switch nargin
            case 1
                linetype='r: ';
                label='';
            case 2
                if ~iscell(in1)
                    in1={in1};
                end
                if I>length(in1)
                    linetype=in1{end};
                else
                    linetype=in1{I};
                end
                label='';
            case 3
                if ~iscell(in1)
                    in1={in1};
                end
                if ~iscell(in2)
                    in2={in2};
                end
                if I>length(in1)
                    linetype=in1{end};
                else
                    linetype=in1{I};
                end
                if I>length(in2)
                    label=in2{end};
                else
                    label=in2{I};
                end
            end
            h(I)=vline(x(I),linetype,label);
        end
    end
else
    switch nargin
        case 1
            linetype='r: ';
            label='';
        case 2
            linetype=in1;
            label='';
        case 3
            linetype=in1;
            label=in2;
        end
    end

    g=ishold(gca);
    hold on

    y=get(gca,'ylim');
    h=plot([x x],y,linetype,'LineWidth',2);
    if length(label)
        xx=get(gca,'xlim');
        xrange=xx(2)-xx(1);
    end
end
```

```
        xunit=(x-xx(1))/xrange;
        if xunit<0.8
            text(x+0.01*xrange,y(1)+0.1*(y(2)-
y(1)),label,'color',get(h,'color'))
        else
            text(x-.05*xrange,y(1)+0.1*(y(2)-
y(1)),label,'color',get(h,'color'))
        end
    end

    if g==0
        hold off
    end
    set(h,'tag','vline','handlevisibility','off')
end % else

if nargout
    hhh=h;
end
```

Building “My First NMRviewer”: A Project

Incorporating Coding and Programming Tasks in the Undergraduate Chemistry Curricula

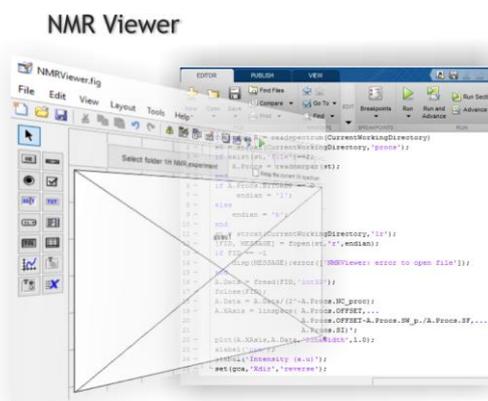
Francisco M. Arrabal-Campos†, Alejandro Cortés-Villena† and Ignacio Fernández†,*

† Department of Chemistry and Physics, Research Centre for Agricultural and Food Biotechnology (BITAL), University of Almería, Ctra. Sacramento, s/n, Almería, E-04120 (Spain).

ABSTRACT

This paper presents a programming project named NMRviewer, which allows students to visualize transformed and processed ^1H NMR data in an accessible, interactive format, while allowing instructors to incorporate programming content in the chemistry curricula. Using the MATLAB graphical user interface development environment (GUIDE), students can build an NMR viewer software program via three main features: a display window, an open file dialogue, and a stack plot option. The experience is designed so that students acquire basic notions of flow diagrams, control loops, Boolean logic rules, and further aspects of the syntax of coding in MATLAB. By building and using the NMR viewer, students can improve their skills and understanding of basic concepts of programming, a discipline often undervalued in current undergraduate chemistry curricula. Importantly, the described NMRviewer is designed for Bruker’s data and Windows operating system, although the tools provided in this project allow students to build their own viewer adapted to their needs.

GRAPHICAL ABSTRACT



KEYWORDS

Upper-Division Undergraduate, Physical Chemistry, Computer-Based Learning, Hands-On Learning/Manipulative, NMR Spectroscopy, Laboratory Computing.

INTRODUCTION

Contemporary chemists are surrounded by technologically advanced instruments and laboratory automation. Spectroscopic and spectrometric techniques such as nuclear magnetic resonance (NMR), mass spectrometry (MS), X-ray diffraction (XRD), among many other techniques, are available to chemistry students during their education, and most of these modern instruments afford raw data in the form of PC files. Frequently, one uses commercially available software to read these files and display them in many interesting forms on one's own computer or laptop. As computer technologies are playing an increasingly important role in chemical research, implementation of computer-based laboratories is growing in order to expose students to practical research problems as a substitute for bench-top experiments.ⁱ The usefulness of computer-assisted experiments and the so-called virtual laboratories for teaching chemistry concepts has been recently reported in this *Journal* demonstrating the applicability of computer programs to teach key concepts.ⁱⁱ Interface software is therefore mandatory and in many cases are used by the students as black boxes where they do not have real knowledge of the programming insights. Many universities have already included MATLAB training in chemical engineering undergraduate curricula,ⁱⁱⁱ but is less extended in chemistry degrees.^{iv} In Spain, upper-division chemistry students are involved in the use of NMR spectrometers for the structural elucidation of, in most of the cases, organic compounds. In this context, inspired by the work of Hans J. Reich, who twenty years ago, developed the WinDNMR program,^v we envisaged a teaching project oriented to improve programming skills as part of their formation as chemists. Undoubtedly, these contents may enable students an easier transition into the industry. With that end, the students will generate a MATLAB code software, named NMRviewer, aimed to visualize ¹H nuclear magnetic resonance (NMR) spectra in a friendly and interactive format. The project is innovative in the

sense that chemistry students will get closer to computational programming in a self-paced way via the development of an interactive tool. Thus, basic concepts such as flow diagrams, control-loops and Boolean logic rules and further aspects about the syntax of coding in MATLAB will be learned. We recommend the MATLAB program^{vi} as a widely-used software package with three main advantages: i) It contains easy to code routines; ii) Allow the easy implementation of complex calculations within few lines, and iii) the student could get deeper in the programming of other learning areas such as chemometrics in the case of subjects related to analytical chemistry.^{vii}

In the last years, MATLAB has acquired new capabilities oriented to coding and it has improved the user interfaces.^{viii} The teaching project we describe herein will use the MATLAB graphical user interface development environment, abbreviated as GUIDE. This tool is a software development kit for creating apps with graphical user interfaces, which are independent programs that automate a specific task or calculation. The use of MATLAB software would improve active involvement of the student in the classroom instruction and would enhance their performances. In our case, we will systematize the reading of already Fourier transformed NMR raw data and its display on the screen.

SCHEDULE

The level of computer expertise could vary among undergraduate students entering the course. Although some students taking the course have had exposure to a programming language, the majority may come without this basic knowledge. We believe our experiment is designed to be performed by uninitiated students, while still providing a challenge for the more advanced. The fact of giving a detailed handout for the former's and the needed tools to create more sophisticated viewers for the latter's support this statement. The project is performed during three sessions that are summarized in Table 1.

Table 1. Distribution of the activities in each section and the sequence of the activities

<i>No. of Session</i>	<i>Duration (h)</i>	<i>Activities</i>
Session 1a	1.5	MATLAB introduction, source code, programming techniques, etc.
Session 1b	0.5	NMR folder tree, raw data, NMR routines, critical files to be read.
Session 2	1.5	Design of the user-interface
Session 3	2.5	NMR Viewer programming

In the first session, students will have a one and half hour lecture exposure to MATLAB philosophy and functioning, and half of an hour lecture on operational NMR. The latter is basically spent on how Bruker files are structured and how the NMR data is read. It is important to mention that if the set of available NMR data does not come from a Bruker spectrometer, or the operating system of choice is not under Windows, the instructors would need in this section to modify the program given in the supporting information. Other files such as JCAMP or any other format where spectral data is given in terms of x,y coordinates, are also possible, and the way they are structured should be explained in this time frame. Regarding the former part, the source code, programming techniques, control-loops, operators, m-files, conditional structures, data input-output routines, and GUIs, are among the concepts to be given by the instructor. There are many sources in the literature that describe the inner workings of MATLAB, and we refer instructors with no experience to these sources.^{ix} The second and third sessions are reserved for user-interface design and programming the NMR viewer, respectively. It is worth mentioning that the viewer code will be provided in the student handout (see Supporting Information) which is explained in a step-by-step way.

PROCEDURE

Session 1

As mentioned earlier, this time window is organized by the instructor depending on the student's background or institutional recommendations. An overall view of MATLAB and its main features may be given.

Session 2

The students start typing GUIDE on the MATLAB command line which makes the development environment appear, which is a working area that contains the

user interface. For our NMR viewer, we basically need three controls: i) a pair of axis where the ^1H NMR spectrum is displayed; ii) a checkbox which allows the overlapping of several ^1H NMR spectra without making any of them disappear, and iii) a button that when pressed it calls an open folder dialog in order to define the corresponding NMR folder.

These controls are available in the development environment and they are very easy to handle using the mouse. All of them contain features such as size, color, name, location, strings, etc. that can be edited at any time during the development of the viewer. An important issue is the callback function that is activated every time you perform a mouse left-click over any control. In this sense, we will set this left-click event on the button control.

The design of the user-interface and handling all the control properties with a logic sequential are difficulties that will make students critically evaluate what they are programming, and will enable them to acquire knowledge of additional techniques enhancing their formation as future, well-trained chemists.

The students are instructed to generate a user interface layout as the one shown in Figure 1 which includes the three controls mentioned above, i.e. the pair of axes in the middle, the button located on the top-left of the working area, and the checkbox just close to it.

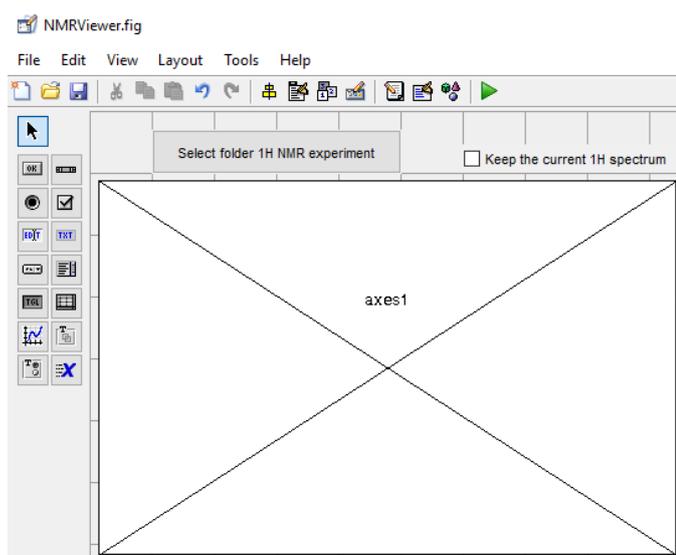


Figure 1. User interface layout (NMRviewer.fig, see Supporting Information) of our NMRviewer. It clearly shows the three main objects that need to be created: button, checkbox and a white square where the pair of axes will be dragged in.

Once the user interface has been created, the next step is programming the routines that our NMRviewer will execute. To do so, it is convenient to plan them in advance, and for that purpose the flow diagram shown in Figure 2 will be given and discussed at the end of this second session.

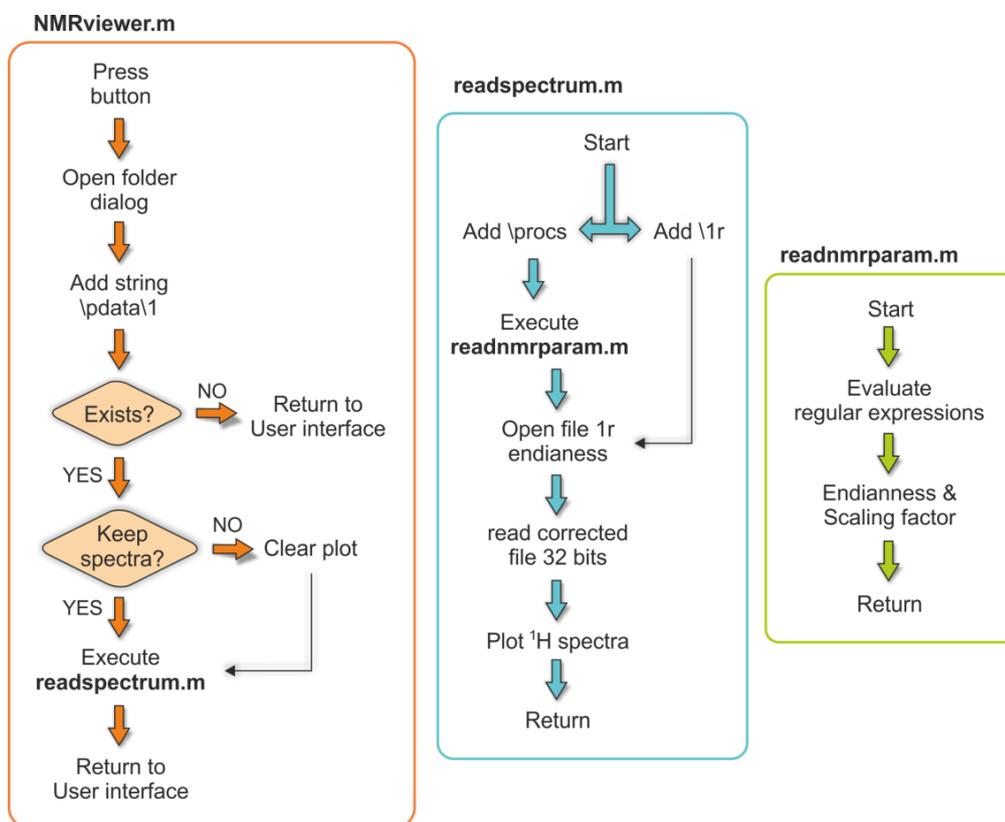


Figure 2. Flow diagram provided to the student at the end of the second session. The three boxes correspond to the files that are programmed: “NMRviewer.m” (left), “readspectrum.m” (middle) and “readnmrparam.m” (right).

Session 3

In this last stage, three files are programmed: “NMRviewer.m”, “readspectrum.m” and “readnmrparam.m”. The first one is the main file created when the GUIDE is executed and from which the other two are called (Figure 2).

In the “NMRviewer.m” file, the students will set the callback function associated to the button. The student handout provides detailed information on how to set this up. When the open folder dialog is visualized, the students will define the directory path (... \pdata\1) where the NMR data (1r) is located. It is important at this point to add some programming lines that will make the software check if the NMR file does exist. If it does not exist, the program will return to the user interface as it is

depicted in Figure 2 and the student handout. Following with the logic sketched in the flow diagram, and only if the file 1r does exist, the “readspectrum.m” file will read the processing parameters of the procs file. From this file, the program will employ the endianness and the scale factor, which are required to get the 1r file in the adequate form. Next, the spectrum will be plot along the pair of axes which were defined in the user interface (see above) after typing the corresponding code lines. The content of the third “readnmrparam.m” file is directly given in the student’s handout, which they just need to copy and paste. This file allows the processing parameters to be read by means of evaluating its regular expressions.^x Once the three files are all set, the viewer is then ready to be tested. For doing so, the students only need to type in the MATLAB command line the word NMRviewer and visualize their own NMR spectra. Illustrative already Fourier transformed raw data obtained from a mixture of solvents will be provided to the students in order to let them practice. Figure 3 shows the appearance of the final viewer.

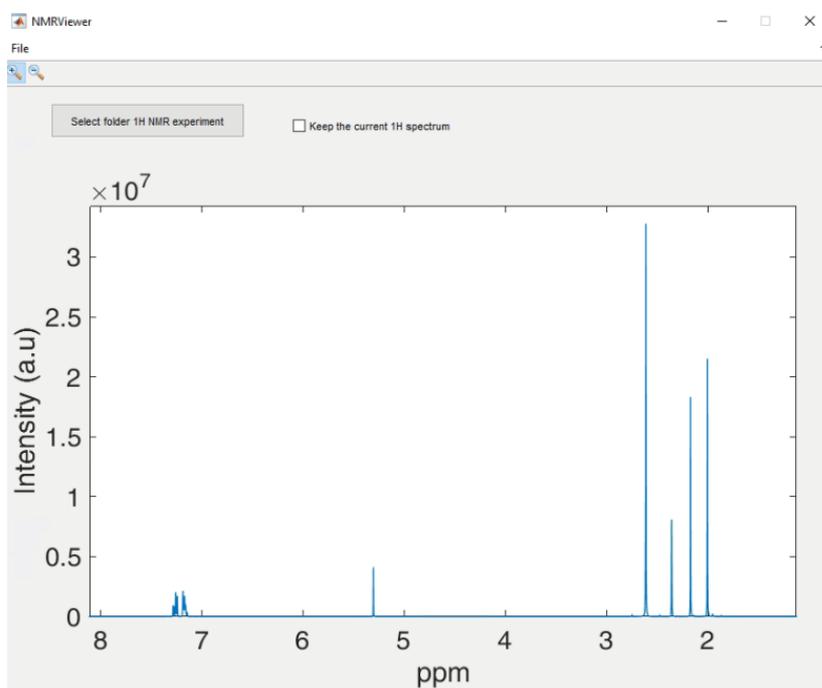


Figure 3. The NMRviewer interface. It shows the ^1H NMR spectrum obtained from a mixture of solvents (acetone, dicloromethane, toluene, dimethylsulfoxide and acetonitrile) in deuterated chloroform. The file can be downloaded from www2.ual.es/NMRMBC/solutions/.

CONCLUSION

A software named NMRviewer, ready to visualize ^1H NMR spectra has been created. The software has been designed by means of a user interface development environment where students will be familiarized with basic notions of flow diagrams, control-loops, Boolean logic rules and further aspects about the syntax of coding in MATLAB. Within this teaching project students will improve on their basic concepts of programming, a discipline undervalued in current undergraduate chemistry curricula. The most serious problem of the project presented herein is our potential failure to give all students the necessary MATLAB skills and provide a disproportional work-load with little time for reading and study. To remedy that, a rationalized schedule is provided based on three sessions in order to make the students learn from almost no knowledge. In addition, the provided tools will allow them to create more sophisticated viewers that could, for instance, execute peak-peaking or integrate certain spectral regions in an automatic way. To summarize, we provide herein a platform where programming skills could be taught, and more importantly, an educational pathway where logical analysis, MATLAB syntax, and automatization tools could be instructed in chemistry curricula.

ASSOCIATED CONTENT

Supporting Information

Student handout which includes all the necessary material to create their own NMRviewer. It includes four MATLAB files and two NMR experiments as given examples.

AUTHOR INFORMATION

Corresponding Author

*E-mail: ifernan@ual.es.

ACKNOWLEDGMENTS

This work was supported by the Junta de Andalucía (Spain) under the project number P12-FQM-2668 and by Bruker Española SA. We thank Dr. Miguel A. Sierra (Universidad Complutense, Madrid) for useful discussions and for the help provided during the revision of the manuscript.

REFERENCES

- ¹. Ibrahim, D. Engineering simulation with MATLAB: improving teaching and learning effectiveness. *Proc. Comput. Sci.* 2011, 3, 853–858.
 - ¹. González-Gómez, D.; Rodríguez, D. A.; Cañada-Cañada, F.; Jeong, J. S. A Comprehensive Application To Assist in Acid–Base Titration Self-Learning: An Approach for High School and Undergraduate Students. *J. Chem. Educ.* 2015, 92, 855-863.
 - ¹. Li, X.; Huang, Z. An inverted classroom approach to educate MATLAB in chemical process control. *Educ. Chem. Engl.* 2017, 19, 1-12, and references cited therein.
 - ¹ a) Ohrstrom, L.; Svensson, G.; Larsson, S.; Christie, M.; Niklasson, C. The Pedagogical Implications of Using MATLAB in Integrated Chemistry and Mathematics Courses. *Int. J. Engng Ed.* 2005, 21, 683-691; b) Zoerb, M. C.; Harris, C. B. A Simulation Program for Dynamic Infrared (IR) Spectra. *J. Chem. Educ.* 2013, 90, 506–507.
 - ¹ Reich, H. J. WinDNMR: Dynamic NMR Spectra for Windows. *J. Chem. Educ.* 1995, 72, 1086.
 - ¹. Matlab 9.2 Release 17a, Copyright 1984–2017 The MathWorks, Inc., <http://www.mathworks.com/> (accessed Jun 2017).
 - ¹. a) O'Haver, T. C. Teaching and Learning Chemometrics with MatLab. *Chemometr. Intell. Lab. Syst.* 1989, 6, 95-103; b) Pierce, K. M.; Schale, S. P.; Le, T. M.; Larson, J. C. An Advanced Analytical Chemistry Experiment Using Gas Chromatography–Mass Spectrometry, MATLAB, and Chemometrics To Predict Biodiesel Blend Percent Composition. *J. Chem. Educ.* 2011, 88, 806–810.
 - ¹. MATLAB's name derives from MATrix LABoratory, which was created by MathWorks in 1984, and it is especially used by universities, research centers and engineering schools. An added value that makes this package useful to our teaching project is that it has free repositories available in the web which could be downloaded and are widely used. https://es.mathworks.com/matlabcentral/fileexchange/?s_tid=gn_mlc_fx (accessed June 2017).
 - ¹ Lent, C. S. *Learning to Program with MATLAB Building GUI Tools*; Wiley: Hoboken, NJ, 2013.
 - ¹ A regular expression is a sequence of characters that defines a search pattern. It allows the identification of strings of letters or numbers that match a certain pattern. This pattern is used to set the parameters in acquisition file and processed file in the case of Bruker's files. See reference vi for more details.
-