

SUMÁRIO

LISTA DE FIGURAS	2
RESUMO	4
ABSTRACT	5
1. INTRODUÇÃO	6
1.1 <i>A Linguagem C</i>	6
1.1.1 Histórico	6
1.1.2 A organização de programas C	7
1.2 <i>A Ressonância Magnética Nuclear</i>	7
1.2.1 Fundamentos da RMN	8
1.2.2 Origem do sinal de RMN	11
Relaxação longitudinal	11
Relaxação transversal	12
Determinação de T_2	13
1.3 <i>Aplicações da RMN de baixa resolução em sementes oleaginosas</i>	16
2. METODOLOGIA	18
2.1 <i>Equipamentos Utilizados</i>	18
2.2 <i>Processamento dos dados</i>	20
2.2.1 Aquisição dos dados	20
2.2.2 Correção de fase	21
Decomposição em Valores Singulares (DVS)	21
2.2.3 Pré-tratamento dos dados	22
2.2.4 Ajuste Exponencial	23
Método da Regressão não-linear de Gauss-Newton	23
3. RESULTADOS E DISCUSSÕES	26
3.1 <i>CPMG fitting v.1</i>	26
MAIN	26
PPC1	27
MEDIA	30
PRI_ORD	32
SEC_ORD	34
3.2 <i>CPMG fitting v.2</i>	37
3.3 <i>Avaliação do software</i>	41
4. CONCLUSÃO	48
REFERÊNCIAS BIBLIOGRÁFICAS	49
ANEXOS	51
Anexo A – <i>Script convert</i>	51
Anexo B – <i>CPMG fitting v.1</i>	51
Anexo C – <i>CPMG fitting v.2</i>	60
Anexo D – <i>Script CPMG</i>	62

LISTA DE FIGURAS

Figura 1. Representação dos níveis de energia quando expostos a um campo magnético B_0 .	9
Figura 2. Representação da precessão dos spins nucleares na presença de um campo magnético externo	9
Figura 3. Polarização da magnetização após a aplicação de um pulso.	11
Figura 4. Decaimento do FID representando o decaimento natural T_2 e o decaimento efetivo T_2^* .	13
Figura 5. Seqüência de pulsos da técnica de CPMG.	14
Figura 6. Representação vetorial da seqüência de pulsos da técnica de CPMG.	15
Figura 7. Seqüência de spin eco aplicada na determinação de óleo em sementes. A linha vermelha representa o sinal do material sólido e da água de hidratação, enquanto a linha preta representa o sinal do óleo.	16
Figura 8. Magneto do Tomógrafo/espectrômetro de 2,1 T e CAT-100 da Tecmag.	18
Figura 9. Diagrama mostrando o espectrômetro Apollo (Tecmag) operando num imã de 0,2T (frequência de 9,5 MHz para o ^1H).	19
Figura 10. Circuito LC de uma sonda de rádio-frequência utilizada em experimentos de RMN.	19
Figura 11. Decaimento da amostra pelo NTNMR.	20
Figura 12. Detalhamento das partes real, imaginária e magnitude das amostras.	21
Figura 13. Interface do <i>software</i> com o usuário.	27
Figura 14. Dados exportados em *.txt.	27
Figura 15. Detalhamento das aquisições.	29
Figura 16. Variações nas aquisições.	30
Figura 17. Últimos 4 pontos de aquisição.	30
Figura 18. Reta ajustada à curva para a determinação dos parâmetros iniciais.	32
Figura 19. Decaimento do amendoim ajustada à curva bi-exponencial.	35
Figura 20. Divisão do decaimento do amendoim, 15% dos pontos refere-se à primeira exponencial e o restante referente à segunda exponencial.	35
Figura 21. Reta ajustada aos primeiros pontos e amplitudes.	36
Figura 22. Menu de edição de <i>scripts</i> .	37

Figura 23. Interface de edição de <i>scripts</i> do NTNMR.....	38
Figura 24. Fluxograma referente ao desenvolvimento do <i>script</i>	39
Figura 25. Indicação do local onde o <i>script</i> faz a leitura dos dados de entrada no NTNMR.....	40
Figura 26. Resultado para coeficientes mono-exponenciais.	42
Figura 27. Resultados para coeficientes bi-exponenciais.	43
Figura 28. Comparação entre os valores ajustado manualmente e automaticamente, a) valores obtidos por ajuste mono-exponencial e b) valores obtidos por ajuste bi-exponencial (somente os valores da primeira componente T_21 foram apresentados). 46	

RESUMO

A ressonância magnética nuclear em baixa resolução *on-line* tem sido usada na Embrapa Instrumentação Agropecuária como um método não destrutivo para medir a qualidade de produtos agro-industriais. As metodologias envolvem as análises da quantidade e qualidade de óleo em sementes para aplicação em biodiesel. O espectrômetro utilizado é baseado no console de RMN CAT-100 da Tecmag e um magneto supercondutor Oxford (2,1T). O principal método para medir a qualidade de óleo em sementes é baseado na sequência de pulsos CPMG. Essa sequência gera um decaimento exponencial relacionado com o tempo de relaxação transversal T_2 . Para tornar a análise por CPMG do sinal de RMN automatizada, foi desenvolvido e implementado um *software* baseado em linguagem C chamado CPMG fitting e utilizando um compilador Bloodshed DEV C/C++. Foi desenvolvido um *software* CPMG fitting v.2 pra realizar o processamento em tempo real. Esse *software* foi implementado no *software* NTNMR (*software* utilizado no controle do espectrômetro de RMN) realizando o processamento de dados de CPMG para cada amostra em poucos segundos. O CPMG fitting processou os dados de CPMG corrigindo a fase pelo "Principal Phase Correction Algorithm", baseado na decomposição em valores singulares (DVS). Posteriormente os dados foram ajustados a uma curva exponencial utilizando o algoritmo de regressão não-linear de Gauss Newton.

Palavras chave: Linguagem C, CPMG, PPC, DVS, Gauss-Newton.

ABSTRACT

On-line low resolution nuclear magnetic resonance has been used at Embrapa Instrumentação Agropecuária as a non-destructive method to measure the quality of agro-industrial products. The methodologies involve the analysis of oil content and quality in seed for biodiesel application. The spectrometer used is based on NMR console CAT100 Tecmag and superconducting magnet Oxford (2.1 T). The principal method to measure the oil quality in the seeds is based on the CPMG pulse sequence. This sequence generates an exponential decay related to the transverse relaxation time, T_2 . To perform the automated CPMG analysis of the NMR signal, it was developed and implemented a software based on C named CPMG fitting language using a compiler Bloodshed DEV C/C ++. It was also developed a software CPMG fitting v.2 to perform the real time processing. This software was implemented in the NTNMR software (software used in the control of NMR spectrometer), allowing to process of CPMG data of each sample in few seconds. CPMG fitting software process the raw CPMG data correcting the phase by "Principal Phase correction algorithm", based on singular value decomposition (SVD). The data is then fitted with exponential curve using the non linear Gauss-Newton regression algorithm.

Key words: C language, CPMG, PPC, SVD, Gauss-Newton.

1. INTRODUÇÃO

1.1 A Linguagem C

C é uma linguagem de programação que foi desenvolvida por Dennis Ritchie durante o começo dos anos 70. Tinha por finalidade inicial a implementação de sistemas operacionais e outras tarefas de programação de baixo nível ^[1].

1.1.1 Histórico

O desenvolvimento de C ocorreu entre 1969 e 1973, de acordo com Ritchie, o período mais criativo foi durante 1972. Em 1973 ela se tornou poderosa o suficiente para reimplementar o sistema operacional Unix. Em 1978, Brian Kernighan e Dennis Ritchie publicaram o agora bastante conhecido "C Programming Language", também conhecido como "o livro branco", K&R.

A linguagem C tornou-se imensamente popular fora do seu ambiente de criação (Bell Labs) depois de 1980 e foi por um tempo a linguagem dominante em programação de sistemas e de aplicações de micro-computadores. Ela se consagrou como a linguagem de programação de sistemas, e é a mais importante da comunidade Open Source.

Foi chamada "C" porque apresenta muitas características derivadas de uma linguagem anterior chamada B, em homenagem ao seu antecedente, BCPL.

Não é necessariamente uma linguagem orientada a objeto, mas é possível fazer uma programação orientada a objeto se assim desejar. Não é uma linguagem funcional, mas pode-se programar funcionalmente. Como dá para perceber, a linguagem C não impõe qualquer ponto de vista. Além disso, possui todas as características de uma linguagem de programação de uso geral, como recursividade, procedimentos como tipos de dados de primeira classe e muito mais.

A linguagem C não tem a simplicidade da linguagem Java ou a sofisticação da linguagem C++ ^[2], é uma linguagem simples, sem muitas complexidades. Mas é justamente esta simplicidade que faz com que se possa controlar o que os programas realizam sem ter que se preocupar com qualquer característica escondida. A linguagem permanece transparente, mesmo quando se implementa características mais sofisticadas, algo do tipo da "garbage collection", típica da linguagem Java.

A linguagem de programação C foi padronizada na forma de um padrão ISO, ISO 9899. A primeira edição do ISO deste documento foi publicada em 1990 (ISO 9899:1990) e foi uma pequena modificação de um padrão ANSI um pouco anterior, ANSI X3.159-1989 "Programming Language C". Edições seguintes têm sido feitas, algumas das quais têm sido amplamente ignoradas ^[1].

1.1.2 A organização de programas C

Um programa C é composto por funções, isto é, porções menores de código que realizam determinadas tarefas, e por dados, ou seja, variáveis ou tabelas que são inicializadas antes do início do programa. Existe uma função especial denominada "main", onde a execução do programa se inicia. As funções são organizadas em módulos agrupados nos arquivos fonte. Em C, a organização do código em arquivos tem um significado semântico. O arquivo fonte principal, passado como argumento para o compilador, define uma unidade de compilação.

Uma unidade pode importar definições usando a diretiva *#include* ou apenas declarando algum identificador como externo.

A linguagem C permite um modelo de compilação separado, isto é, pode-se dividir o programa em várias unidades independentes que são compiladas separadamente, e depois são concatenadas com o *link* editor para construir o programa final. Normalmente, cada módulo é escrito num arquivo texto separado que contém funções ou declarações de dados. As interfaces entre os módulos são escritas em "header files" (arquivos de cabeçalho) que descrevem tipos ou funções visíveis para vários módulos do programa. Estes arquivos possuem a extensão ".h" e são de dois tipos: privados, específicos para aplicação que está sendo elaborada; e para o sistema ^[1].

Toda função possui uma lista de parâmetros, um corpo e, eventualmente, um valor de retorno. O corpo pode conter declarações de variáveis locais, ou seja, variáveis que são ativadas quando a execução alcançar o corpo da função ^[2].

1.2 A Ressonância Magnética Nuclear

A espectroscopia de Ressonância Magnética Nuclear (RMN) é uma das mais importantes técnicas analíticas, com aplicações nas mais diversas áreas da química. A primeira observação do fenômeno de RMN foi vista em 1945, independentemente por

dois grupos de pesquisadores: Purcell, Torrey e Pound na Universidade de Harvard, e Hansen e Packard na Universidade de Stanford. Já no início da década de 50, a RMN foi aplicada pela primeira vez na solução de problemas químicos. Desde então vem sendo amplamente utilizada na análise de estrutura de amostras orgânicas, biológicas, inorgânicas, polímeros, entre outros. A grande contribuição da RMN para a química foi reconhecida com os Prêmios Nobel de química em 1991 e 2003, recebido pelos pesquisadores suíços Richard R. Ernst e Kurt Wüthrich, respectivamente.

A maioria dos experimentos de RMN utiliza espectrômetros de alta resolução em alto campo, de 5 a 20 Teslas (frequência de ressonância de 200 a 800 MHz para ^1H) baseados em ímãs supercondutores.

Uma categoria de espectrômetros de RMN de baixo custo, que é menos difundida na química, mas que está sendo gradualmente inserida nos laboratórios de pesquisa, desenvolvimento e controle de qualidade, são os equipamentos de baixa resolução em baixo campo (1,4T). Neste tipo de aparelho, o deslocamento químico praticamente não é observado e as medidas são realizadas no domínio do tempo, sem necessidade de Transformada de Fourier. As aplicações são principalmente em análises quantitativas de um dos componentes de uma amostra heterogênea. Recentemente, os espectros de RMN no domínio do tempo, começaram também a ser utilizados em análises qualitativas e na quantificação de mais de um produto na mesma amostra ^[3-5].

1.2.1 Fundamentos da RMN

A RMN se baseia nas propriedades magnéticas do núcleo atômico, denominado de momento magnético nuclear μ , que se deve à rotação de núcleos carregados em torno de um de seus eixos. Esse movimento é denominado spin nuclear, expresso em termos de um número quântico de spin (I). Porém, a RMN é somente observada em núcleos que contém I total diferente de zero, portanto, em isótopos que apresentam número ímpar de prótons e/ou nêutrons.

O fenômeno da RMN é observado quando a amostra é submetida a um campo magnético externo, denominado B_0 . Na presença de B_0 , os spins são separados em $2I + 1$ níveis de energia, com uma diferença de energia $\Delta E = h\nu$, onde h é a constante de Planck e ν é a frequência de excitação dada em Hz. No caso do ^1H , I é igual a $1/2$, com dois estados de energia, α e β (Figura 1). Para atingir a condição de ressonância, irradia-se a amostra com uma frequência igual a ν .

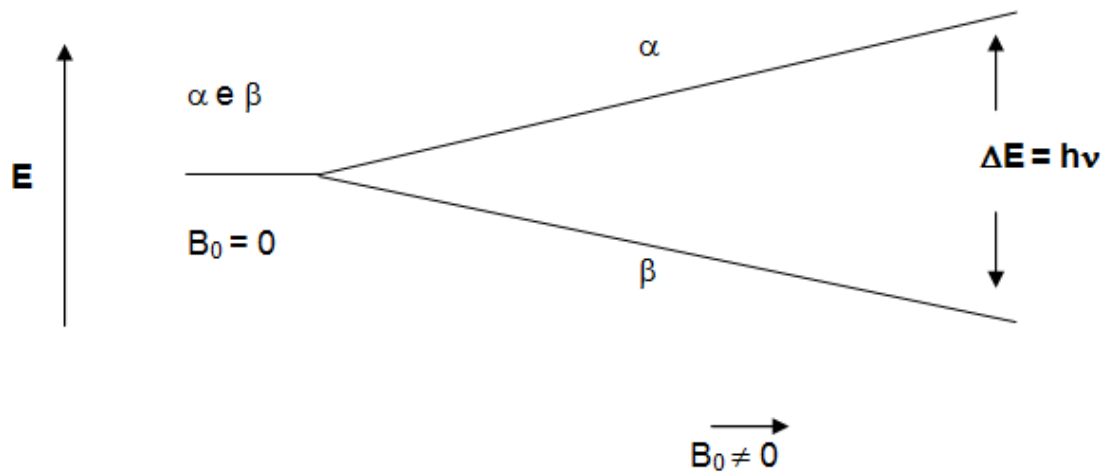


Figura 1. Representação dos níveis de energia quando expostos a um campo magnético B_0 .

No modelo clássico, a interação de B_0 com o momento magnético nuclear faz com que eles se orientem em relação ao campo e passem a precessionar com uma frequência angular denominada frequência de Larmor, ω_0 (Figura 2).

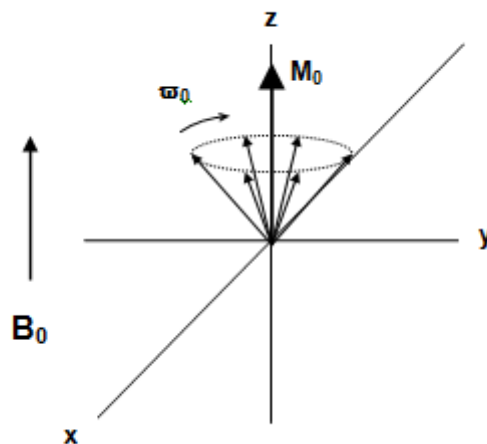


Figura 2. Representação da precessão dos spins nucleares na presença de um campo magnético externo.

A frequência de precessão é proporcional ao campo magnético B_0 e à razão magnetogírica γ , tal que:

$$\nu = \gamma \frac{B_0}{2\pi} \quad \text{ou} \quad \omega_0 = \gamma B_0 \quad (1)$$

onde γ é uma constante intrínseca de cada núcleo em evidência, que caracteriza, em particular, sua frequência na presença de B_0 .

A condição de ressonância é atingida quando se aplica uma radiação eletromagnética de frequência igual à frequência de precessão ou de Larmor, tornando possível, portanto, a ocorrência de transições de spins nucleares entre os níveis α e β . A distribuição da população de spins em cada nível de energia é dada pela equação de Boltzmann:

$$\frac{N_{\alpha}}{N_{\beta}} = e^{\frac{\Delta E}{kT}} \quad (2)$$

onde N_{α} = população no nível de menor energia

N_{β} = população no nível de maior energia

k = constante de Boltzmann

T = temperatura (K).

Esta equação demonstra que nas condições experimentais haverá sempre um pequeno excesso de população no estado de menor energia α . Essa diferença de população dá origem a uma resultante, M_0 , que é o somatório do excesso de todos os momentos magnéticos μ que precessionam em torno de B_0 (Eq. 3), formando o chamado cone de precessão num referencial de coordenadas fixas (Figura 2).

$$M_0 = \sum \mu \quad (3)$$

A aplicação de uma radiação eletromagnética na forma de pulso de rádio-frequência (r.f.) igual à de Larmor em uma bobina, resulta na formação de campo magnético B_1 perpendicularmente a B_0 . Isso faz com que ocorra alteração na distribuição de população de Boltzmann, promovendo assim, transições simultâneas entre os níveis α e β e conseqüentemente modificando o módulo da magnetização resultante, denominada a partir de então como M_z (Figura 3).

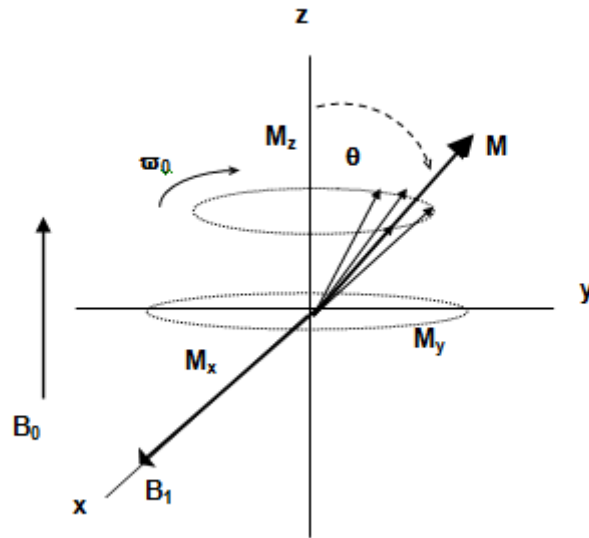


Figura 3. Polarização da magnetização após a aplicação de um pulso.

O ângulo de M varia proporcionalmente com a duração do pulso incidido sobre a amostra e com a intensidade do campo B_1 , tal que:

$$\theta = \gamma B_1 t_p \quad (4)$$

onde θ = ângulo formado pela magnetização com o eixo z

t_p = tempo de duração do pulso de r.f.

M_z e sua precessão em torno da direção z determinam o aparecimento de uma magnetização no plano xy, com componentes M_x e M_y variando periodicamente, com frequência $\nu = \omega_0/2\pi$ e com uma diferença de fase de $\pi/2$.

1.2.2 Origem do sinal de RMN

Após o pulso, ocorre novamente a precessão dos spins nucleares em torno de B_0 induzindo o sinal de RMN na bobina, gerando o sinal de RMN denominado decaimento livre induzido, FID, “Free induction decay”. Após o pulso, ocorrem dois processos de relaxação distintos: a relaxação transversal e a relaxação longitudinal.

Relaxação longitudinal

O mecanismo de relaxação, também denominado de T_1 , é responsável por M_z retomar seu estado de equilíbrio M_0 . Este processo está intimamente ligado ao

restabelecimento da distribuição de população dos níveis Zeeman de energia, envolvendo transferência do excesso de energia potencial magnética do estado excitado para a vizinhança ou rede, sob a forma de energia cinética molecular.

Essa transferência é resultante da existência de momentos magnéticos transitórios produzidos pelos movimentos rotacionais e translacionais de moléculas da vizinhança, oscilando com frequências apropriadas em torno da frequência de Larmor, tornando a rede capaz de absorver essa energia. Esse processo de transferência é conhecido como relaxação spin-rede ou relaxação longitudinal, o qual é responsável pela recuperação exponencial do módulo da magnetização ao longo do eixo z, que transcorre no tempo com uma constante de velocidade $1/T_1$, e é dada pela seguinte equação:

$$M_z(t) = M_0 \left(1 - e^{-\frac{t}{T_1}} \right) \quad (5)$$

O tempo de relaxação T_1 tem um efeito prático marcante. Se for longo, o tempo total de uma análise de RMN se estende muito, uma vez que cada seqüência de pulsos deverá ser precedida por um tempo de espera, da ordem de $5T_1$ para que a magnetização retorne 99% de sua intensidade no equilíbrio térmico, evitando a saturação do sinal.

Relaxação transversal

O tempo de relaxação transversal, denominado T_2 , independe da transferência de energia e está relacionado com a perda de coerência da magnetização no plano xy. Essa perda de coerência é atribuída a interações diretas entre os momentos magnéticos individuais, sem haver modificação na energia do sistema com um aumento na entropia, causando um processo de relaxação conhecido como spin-spin ou relaxação transversal.

A relaxação transversal provoca um decaimento exponencial da magnetização, dado pela seguinte equação:

$$M_y(t) = M_{xy} \cos(\omega_0 t) e^{-\frac{t}{T_2}} \quad (6)$$

Determinação de T_2

Existem muitas dificuldades experimentais para medir T_2 , decorridas da necessidade de anular os efeitos da não homogeneidade de campo B_0 no decaimento do sinal FID. Cada vetor magnetização individual dos spins, chamados isocromatas de spins, sente um campo ligeiramente diferente. Isso causa uma perda mais rápida de coerência de fase quando o processo de relaxação se dá logo após cessar a duração do pulso. Este processo acelera o desaparecimento da componente transversal da magnetização total. Dessa forma, a constante de tempo com a qual o FID decai é na verdade representada por um tempo de relaxação efetivo T_2^* , tal que:

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \gamma \Delta B_0 \quad (7)$$

A diminuição de T_2^* contribui para que o sinal de RMN decaia mais rapidamente, causando o alargamento das linhas do espectro no domínio da frequência. Por isso, têm-se buscado campos magnéticos cada vez mais homogêneos, onde T_2^* se aproxime do valor de T_2 .

Na Figura 4 pode ser observado que após a excitação da amostra com um pulso de 90° , o sinal de decaimento (linha vermelha) e sua constante de tempo T_2^* é bem menor que a constante de tempo natural da amostra (T_2)^[6].

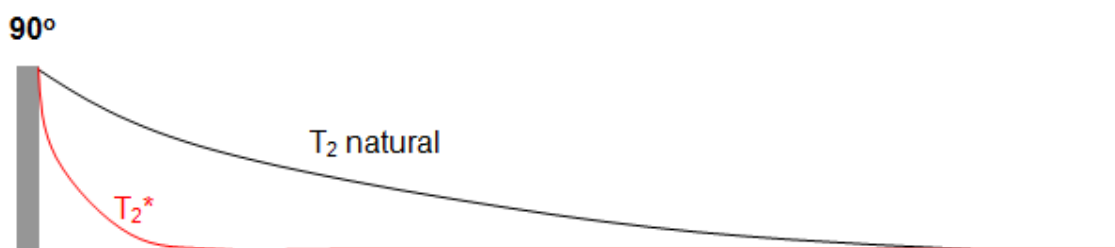


Figura 4. Decaimento do FID representando o decaimento natural T_2 e o decaimento efetivo T_2^* .

Para obter a medida do tempo de relaxação transversal independente da homogeneidade de campo, Hahn propôs o uso de seqüências de pulsos, hoje denominadas, eco de Hahn.

A primeira técnica de RMN pulsada para medida de T_2 foi proposta por Hahn em 1950. Essa seqüência é composta por dois pulsos de 90° ^[7]. Esta técnica não teve grande aplicação prática, pois logo Carr e Purcell introduziram a seqüência com o segundo pulso de 180° , gerando o dobro da intensidade do sinal de eco^[8].

Carr e Purcell (CP) também propuseram uma seqüência onde é aplicado um trem de pulsos de 180° após o primeiro pulso de 90° , ficando assim descrita: $90^\circ_{x'} - \tau - [180^\circ_{x'} - \tau - (\text{eco}) - \tau]_n$. Essa nova seqüência permitiu o uso de pequenos valores de τ , eliminado o efeito de difusão da amostra, durante o tempo 2τ de formação do eco. O método CP ainda permite economizar tempo, principalmente em experimentos onde o T_1 da amostra é muito longo, pois uma só seqüência permite obter uma série de n ecos, ao passo que o método de Hahn requer n seqüências com um longo tempo de espera ^[8].

Posteriormente, Meiboom e Gill utilizaram a técnica CP, contudo, aplicando os pulsos de 180° na direção $+y'$, isto é, com uma diferença de fase de 90° em relação ao pulso inicial, melhorando o método de Carr-Purcell, originando o método chamado CPMG ^[9] (Figuras 5 e 6).

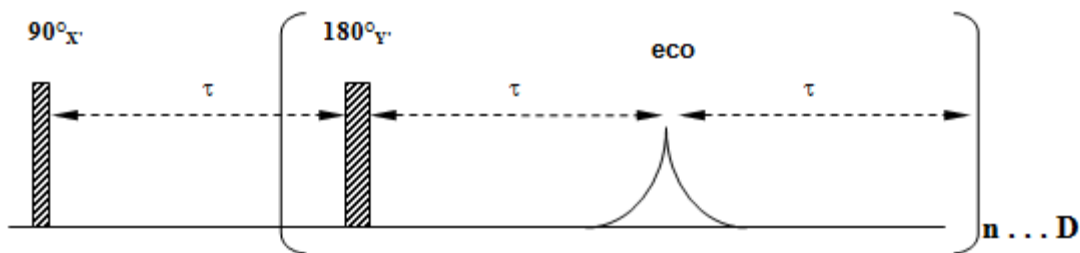


Figura 5. Seqüência de pulsos da técnica de CPMG.

A Figura 6 demonstra a técnica de CPMG, onde um pulso de 90° é aplicado em x' no instante zero (A), causando a rotação dos spins para o plano de coordenadas girantes ($x'y'$) (B). Ao longo do tempo τ , os spins se separam (C), então um pulso de 180° é aplicado em y' (D), ocorrendo a refocalização para o tempo 2τ no eixo y' (E). Após o defasamento seguinte (F), o segundo pulso de 180° , aplicado em y' , rotacionando os spins para o plano $x'y'$ (G) onde ocorre novamente a refocalização para $t = 4\tau$ (H).

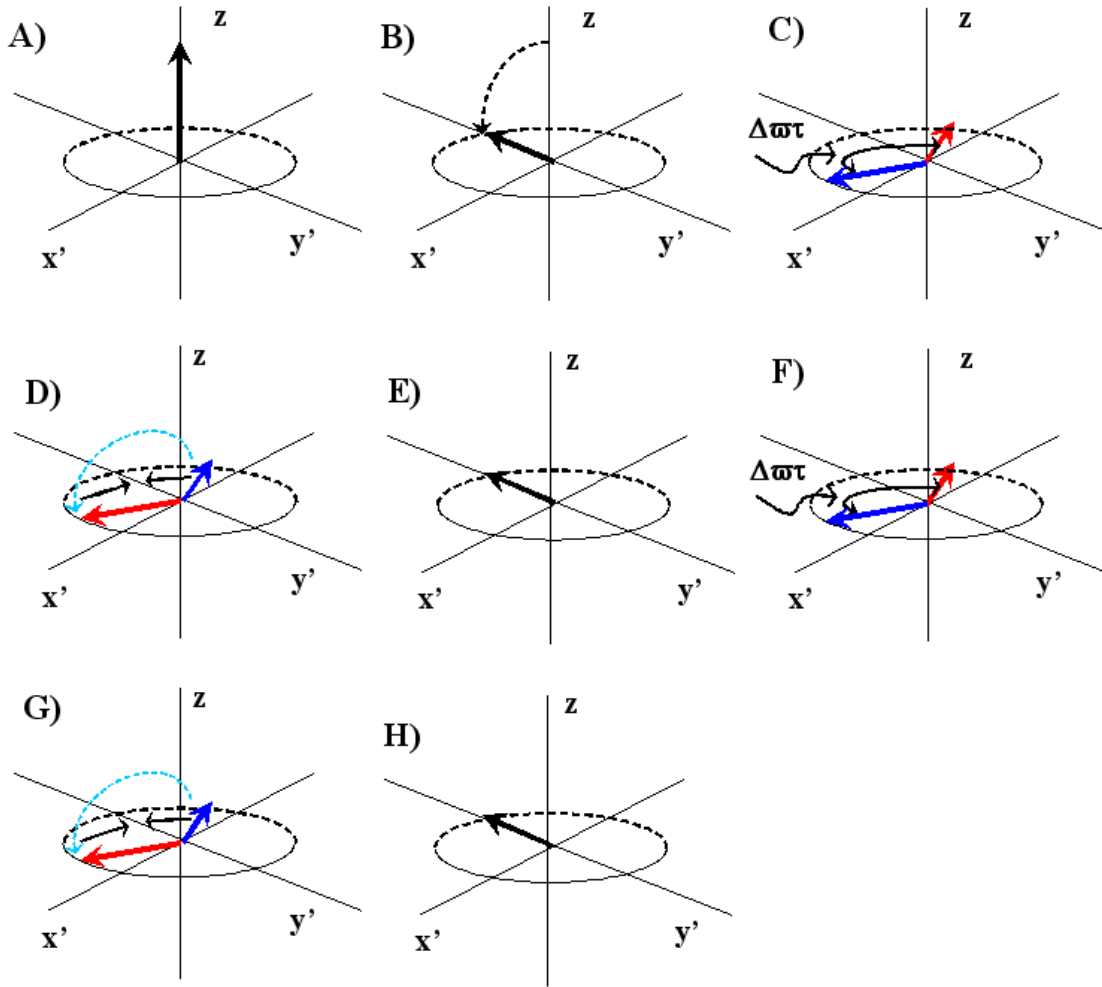


Figura 6. Representação vetorial da sequência de pulsos da técnica de CPMG.

Essa mudança de direção permite refocalizar todos os ecos em uma mesma fase, o que não ocorre com o método CP.

Como a intensidade do eco (I_{eco}) depende de τ e T_2 , pode-se medir T_2 a partir do estudo do eco no tempo $t=k\tau$, com $k = 1, 2, \dots, n$, em função de t utilizando a equação 8.

$$I_{eco}(k\tau) \propto e^{\frac{-t}{T_2}} \quad (8)$$

A partir de um ajuste da curva exponencial dada pela equação acima ou através de um gráfico semi-logaritmo da intensidade do eco em função de t , tem-se T_2 .

1.3 Aplicações da RMN de baixa resolução em sementes oleaginosas

As aplicações da RMN de baixa resolução nas análises de óleo em sementes começaram na década de 60 com o uso dos aparelhos de onda contínua de banda larga ^[10]. Essas análises eram realizadas nas sementes intactas em alguns minutos, permitindo que a técnica fosse utilizada para o melhoramento genético, possibilitando a seleção das variedades com alto conteúdo de óleo e conseqüente potencial de produção de grãos em agroindústrias de refinamento de óleo ^[11]. Os métodos analíticos convencionais, apesar de serem muito precisos, impossibilitam esse emprego devido ao custo, ao tempo e à impossibilidade de reutilização das amostras.

Com a introdução dos espectrômetros pulsados de baixa resolução no início da década de 70, o tempo de análise foi reduzido a alguns segundos.

Os principais constituintes das sementes oleaginosas que contêm hidrogênio, são os carboidratos, as proteínas, a água e o óleo. Os sinais dos materiais sólidos, proteínas e carboidratos ou de baixa mobilidade, como a água de hidratação, têm T_2 curto de alguns microssegundos, portanto, decaem rapidamente; enquanto que os sinais da água livre e do óleo, que têm T_2 longo de alguns milissegundos, decaem mais lentamente. Para se determinar o teor de óleo, elimina-se a água livre secando as sementes até 5% de umidade. Aplicando a seqüência de eco de spin, o sinal dos materiais sólidos e da água de hidratação decai por completo após o pulso de 90° , tornando-os incapazes de serem refocalizados pelo pulso de 180° . Dessa forma, obtém-se o sinal do eco em 2τ apenas para o óleo, conforme mostra a Figura 7.

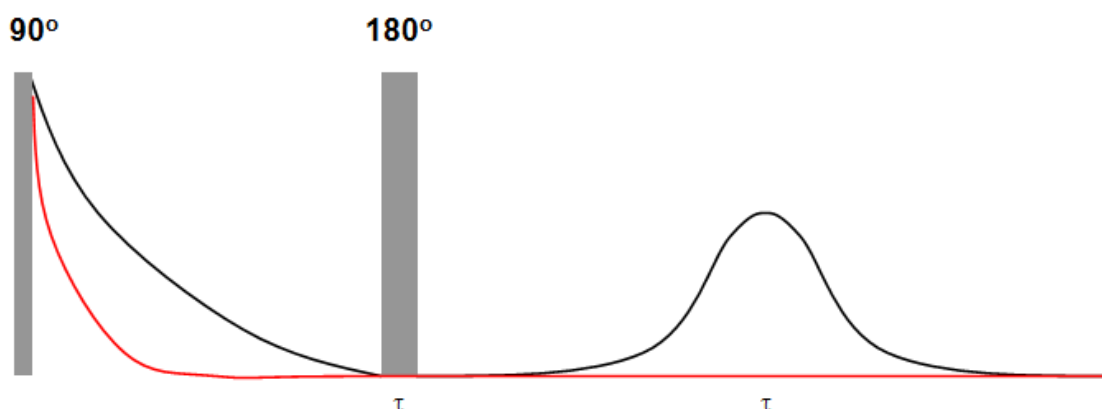


Figura 7. Seqüência de spin eco aplicada na determinação de óleo em sementes. A linha vermelha representa o sinal do material sólido e da água de hidratação, enquanto a linha preta representa o sinal do óleo.

Como a análise quantitativa por RMN em baixo campo não é absoluta, as quantificações se baseiam na correlação linear entre a intensidade do sinal de RMN e a concentração do produto a ser analisado. Logo, é necessário fazer uma curva de calibração entre a intensidade ou área do sinal de RMN e a concentração de amostras padrões. Com a curva de calibração, assim como em outros métodos espectroscópicos, obtém-se a concentração de uma substância em uma amostra desconhecida somente com a medida da intensidade do seu sinal de RMN, seja FID ou eco ^[10].

Determinações simultâneas da porcentagem da mistura e do conteúdo de óleo na semente podem ser feitas com as técnicas de eco de spin de Hahn e de CPMG pela análise do T_2 . O FID tem sido usado nessas determinações, porém exige maior homogeneidade do campo magnético, uma vez que a posição angular da semente afeta significativamente o sinal ^[12]. Hoje a RMN pulsada é um método padrão para medidas de óleo e umidade em alimentos/produtos agrícolas, normas IUPAC 2.322, IUPAC 2150, AOCS Cd16b-93, e ISO 10565 ^[3].

Além da determinação da quantidade de óleo em sementes há uma demanda muito grande por métodos rápidos e não destrutivos de medidas da qualidade dos óleos vegetais. No caso de uso alimentar, essa demanda surgiu com o fato dos ácidos graxos trans, produzidos durante o processo de hidrogenação, levar a problemas cardiovasculares. A hidrogenação é usada para reduzir a insaturação do óleo e conseqüentemente levar a uma maior estabilidade e um maior tempo de prateleira dos alimentos. Para evitar a necessidade da hidrogenação, hoje estão sendo desenvolvidos novos cultivares de oleaginosas com alto teor de ácido oléico e ácidos graxos saturados.

Com o uso de óleos vegetais para produção de biodiesel também será necessário o desenvolvimento de cultivares com teores de ácidos graxos modificados, ricos em ácido oléico. Com isso, uma metodologia RMN de baixo campo, rápida e automática (on-line) poderá ser uma poderosa ferramenta para a determinação da qualidade dos óleos vegetais nas sementes oleaginosas, podendo acelerar o processo de obtenção de novos cultivares, tanto para uso alimentar quanto energético.

Visando o desenvolvimento de tal ferramenta, o *software* CPMG fitting foi desenvolvido com o intuito de acelerar o processo de obtenção dos resultados da técnica de CPMG. Com duas versões, uma para análise imediatamente após a

aquisição de cada experimento e outra para análise posterior de um conjunto de experimentos foi possível processar os dados em questão de segundos.

Para tanto foi utilizada a linguagem C, pois esta possibilita o máximo de eficiência durante o desenvolvimento, minimizando os custos do projeto. Outra grande vantagem dessa linguagem é a sua portabilidade, possibilitando que programas criados para uma determinada plataforma possam ser migrados para outra plataforma sem grandes dificuldades. Os programas feitos em C também são mais fáceis de entender e dar manutenção.

2. METODOLOGIA

2.1 Equipamentos Utilizados

Os dados a serem analisados foram obtidos com a técnica de CPMG. Foi utilizado um transmissor/receptor CAT-100 da Tecmag e um magneto de 2,1 T (frequência de ressonância de 85 MHz para ^1H) de um Tomógrafo/espectrômetro de RMN da Varian com uma abertura ("bore") de 30 cm (Figura 8).



Figura 8. Magneto do Tomógrafo/espectrômetro de 2,1 T e CAT-100 da Tecmag.

O equipamento opera em conjunto com o Amplificador de potência 2035 AMT, pré-amplificador Miteq AU 1448, conforme diagrama eletrônico descrito na Figura 9.

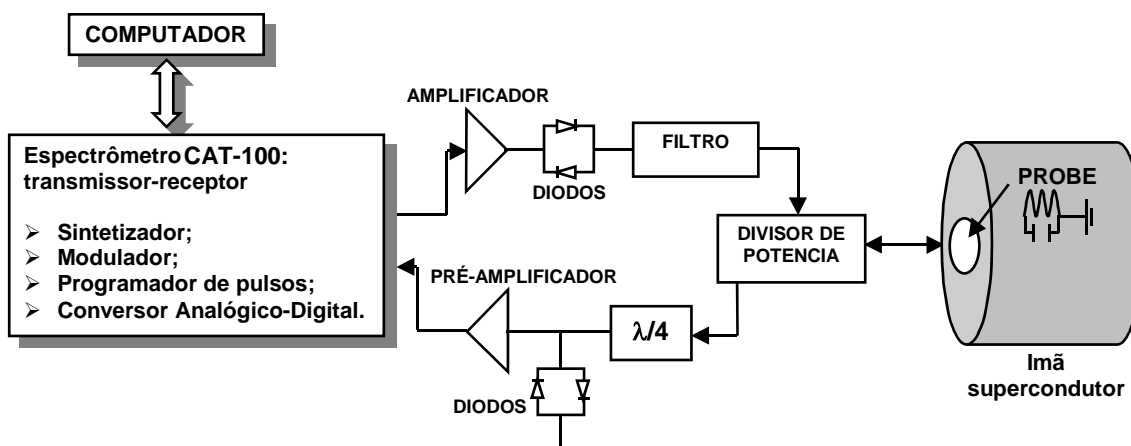


Figura 9. Diagrama mostrando o espectrômetro Cat-100 (Tecmag) operando num ímã de 2,1 T (frequência de 85 MHz para o ^1H).

A sonda consiste de um circuito do tipo LC (indutor-capacitor), composto de dois capacitores variáveis e uma bobina indutora (Figura 10). A bobina foi construída num formato solenoidal a partir de um fio de cobre. As espiras da bobina indutora foram moldadas em uma resina epóxi de forma a conferir rigidez à mesma, evitando oscilações mecânicas e deformações.

No circuito LC utilizado, um dos capacitores é ligado em série (C_s) com a bobina indutora (L), enquanto que o outro é ligado em paralelo (C_p). Nesta configuração o circuito LC opera tanto como transmissor, quanto como receptor.

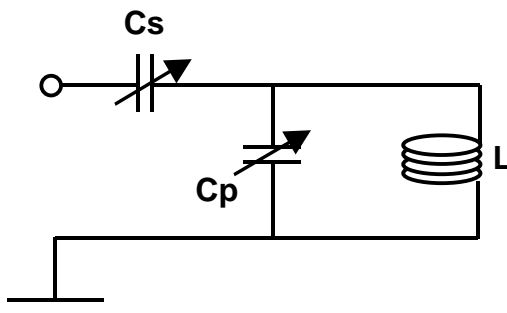


Figura 10. Circuito LC de uma sonda de rádio-freqüência utilizada em experimentos de RMN.

O circuito LC é então sintonizado, através da manipulação das capacitâncias de C_s e C_p , obtendo assim uma absorção de reflexão satisfatória. Além disso, o processo de sintonia evita que aquela radiação refletida, ou seja, não absorvida pela amostra, seja suficientemente grande para que algum componente eletrônico constituinte do espectrômetro seja danificado.

2.2 Processamento dos dados

Antes do desenvolvimento do *software*, todo o tratamento dos dados gerados era feito manualmente ou através de outro *software* desenvolvido em Matlab. Esse tratamento era feito para se obter os valores de T_2 das amostras solicitadas.

O processamento manual era demorado, visto que o tratamento de cada amostra levava em torno de 4 minutos. Levando-se em consideração o alto número de amostras que a EMBRAPA recebe para tais análises, o tempo gasto se torna um obstáculo. Além dos problemas citados, estava sujeito a erros grosseiros.

O tratamento através do *software* desenvolvido em Matlab era mais demorado do que o manual e, além disso, os dados não eram completamente tratados, exigindo ainda que uma parte fosse feita manualmente. Dependendo da amostra, o *software* gastava em torno de 15 minutos por tratamento.

2.2.1 Aquisição dos dados

Após a inserção da amostra no tomógrafo e realizadas as configurações de seqüência, pontos de aquisição e outras, necessárias de acordo com o método utilizado, inicia-se a aquisição dos dados.

A duração da aquisição é dependente do método utilizado e o número de pulsos da seqüência. Ao final observa-se um decaimento como o exemplo da Figura 11, correspondente a uma amostra de amendoim.

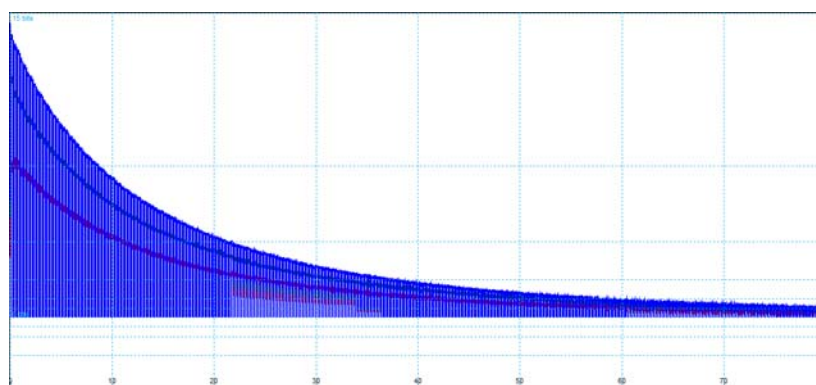


Figura 11. Decaimento da amostra pelo NTNMR.

Para o tratamento desses dados era necessário que se exportasse tal arquivo para o formato *.txt. Nesse formato era possível visualizar os valores das componentes real e imaginária de cada ponto. Na Figura 12 estão representados

pelas cores azul, verde e vermelho, os dados de magnitude, real e imaginário, respectivamente.

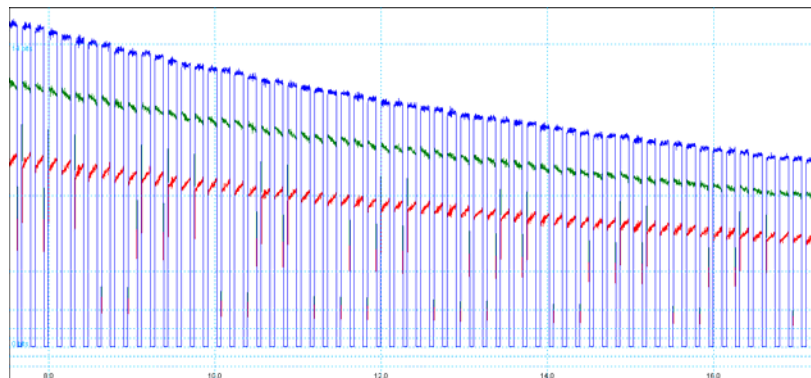


Figura 12. Detalhamento das partes real, imaginária e magnitude das amostras.

A determinação das variáveis de entrada foi feita de acordo com as necessidades de cada tipo de amostra, bem como o número de pontos de cada arquivo e tempo de decaimento, entre outros fatores. Essas variáveis foram utilizadas principalmente para a determinação de espaço de memória das matrizes no *software* em desenvolvimento.

2.2.2 Correção de fase

A correção de fase consistiu em filtrar os dados e extrair o valor da intensidade do eco com o mínimo de ruído possível, para tanto, foi utilizada a técnica “Principal Phase Correction” (PPC), baseada na Decomposição em Valores Singulares (DVS)^[13].

Na Figura 12 observa-se uma variação da aquisição nos dados reais e imaginários. Ao longo da aquisição os valores reais diminuem e os imaginários aumentam, mostrando o erro de fase encontrado na aquisição.

Decomposição em Valores Singulares (DVS)

A DVS de uma matriz A , $n \times m$, que representa a anomalia magnética, é vista como um método de filtragem bidimensional de coerência que separa informações correlacionáveis e não correlacionáveis contidas na matriz de dados magnéticos A . O filtro DVS é definido através da expansão da matriz A em autoimagens e valores singulares. Este método de filtragem se baseia no fato de que as autoimagens associadas a grandes valores singulares concentram a maior parte da informação correlacionável presente nos dados, enquanto que a parte não correlacionada, presumidamente constituída de ruídos causados por fontes magnéticas externas ou

ruídos introduzidos pelo processo de medida, estão concentrados nas autoimagens restantes ^[1].

A DVS corresponde à determinação dos autovetores e autovalores da matriz dos dados. Esses autovetores ou componentes principais formam uma base ortonormal que descreve as direções de maior variação dos dados ^[15].

No caso de espectros, podemos associar às direções de maior variação os componentes das amostras em questão, como por exemplo, óleo e água que apresentam sinais em regiões diferentes do espectro. Sendo esses os componentes que variam de uma amostra para outra e conhecendo-se as variações através das análises de referência, podemos correlacionar os autovalores e as concentrações. Se considerarmos apenas os autovetores cujos autovalores são mais significativos, o método atua como um filtro de ruído, pois apenas as direções de maior variação são consideradas e essas correspondem aos picos mais intensos do espectro, que carregam a informação desejada. A decomposição da matriz de dados em novas componentes que sejam boas representações dos dados originais é possível com a utilização da DVS, que transforma a matriz M , retangular e, portanto não inversível, em um produto de três matrizes: u , s e v^T .

$$M = u * s * v^T$$

A matriz s é diagonal, com elementos em ordem decrescente, e as matrizes u e v^T são matrizes unitárias. Das propriedades de matrizes sabemos que a inversa de uma matriz diagonal é a matriz formada pelos inversos de seus elementos, e a inversa de uma matriz unitária é a transposta dessa matriz. Resolve-se assim o problema da inversão. Tomando-se apenas os elementos mais significativos das matrizes u , s e v^T , podemos reconstruir a matriz de espectros com menos elementos, que são os que carregam a informação importante. O produto $u * s$ é a matriz de autovalores da base formada pelos autovetores, v^T ^[15]. Em seguida multiplica-se a matriz gerada pelo produto $u * s$ pela função sinal da soma de seus elementos e encontra-se a representação com fase corrigida.

2.2.3 Pré-tratamento dos dados

O pré-tratamento dos dados consistiu basicamente em reagrupar os dados de forma a permitir o ajuste exponencial dos mesmos, eliminando valores de aquisição nulos, separando os dados em grupos de aquisição com a finalidade de calcular a

média de cada eco, normalizando e eliminando o *offset* dos dados a fim de comparação entre amostras distintas.

2.2.4 Ajuste Exponencial

Com o vetor de tempo e médias normalizadas calculados, foi feito o ajuste da curva com duas opções de modelo: exponencial de primeira ordem ou exponencial de segunda ordem. Esses modelos são descritos nas equações 9 e 10.

$$f(t) = A_1 e^{\frac{-t}{t_1}} \quad (9)$$

e

$$f(t) = A_1 e^{\frac{-t}{t_1}} + A_2 e^{\frac{-t}{t_2}} \quad (10)$$

onde $f(t)$ é o vetor de médias

A é a intensidade do sinal e

t é o vetor de tempo.

A opção pelo ajuste da curva ficou a cargo do usuário, essa decisão é tomada de acordo com as características de cada amostra. Existem amostras que possuem mais de um componente, sendo melhor representadas, nesse caso, por uma exponencial de segunda ordem ou superior.

A decisão sobre qual método adotar para o ajuste das curvas de primeira ou segunda ordem foi estudada através de testes de ajustes realizados com diferentes métodos. O Método de Newton por regressão não linear se mostrou uma ótima solução para o problema.

Os valores de A_1 , A_2 , t_1 e t_2 foram armazenados em um arquivo do tipo *.txt. Com esses coeficientes foi possível a reprodução do decaimento de cada amostra usando *softwares* como Matlab ou Origin para melhor comparação dos resultados.

Método da Regressão não-linear de Gauss-Newton

Existem muitos casos na engenharia onde modelos não lineares precisam ser ajustados aos dados. Esses modelos são definidos por aqueles em que existe uma dependência não linear entre seus parâmetros. Por exemplo:

$$f(x) = a_0 - a_1 e^{-t_0 x} \quad (11)$$

A Regressão Não Linear é baseada na determinação dos valores dos parâmetros que minimizem a soma dos quadrados dos resíduos. Entretanto, para o caso não linear, a solução deve proceder de uma maneira iterativa. Assim, para se obter uma solução satisfatória, a regressão não linear é altamente dependente dos valores iniciais.

O método de Gauss-Newton é um algoritmo usado para minimizar a soma dos quadrados dos resíduos entre os dados e a equação não linear. O conceito chave sobre a técnica é que a expansão da série de Taylor é utilizada para expressar a equação não linear como uma equação de aproximação de forma linear. Assim, a teoria dos mínimos quadrados pode ser utilizada para obter novas estimativas para os parâmetros no sentido de minimizar o resíduo.

Para ilustrar como isso é feito, primeiramente, a relação entre a equação não linear e os dados pode ser genericamente expressa como:

$$y_i = f(x_i; a_0, a_1, \dots, a_m) + e_i \quad (11)$$

onde y_i é o valor medido da variável dependente

$f(x_i; a_0, a_1, \dots, a_m)$ é uma função da variável independente x_i e uma função não linear dos parâmetros a_0, a_1, \dots, a_m e

e_i é o erro randômico.

Por conveniência, este modelo pode ser expresso numa forma abreviada omitindo-se os parâmetros.

$$y_i = f(x_i) + e_i \quad (12)$$

O modelo não linear pode ser expandido numa série de Taylor em torno dos valores dos parâmetros e reduzido depois da primeira derivada. Por exemplo, para o caso de dois parâmetros:

$$f(x_i)_{j+1} = f(x_i)_j + \frac{\partial f(x_i)_j}{\partial a_0} \Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1} \Delta a_1 \quad (13)$$

onde j corresponde à iteração

j+1 corresponde à iteração j+1

$$\Delta a_0 = a_{0,j+1} - a_{0,j}$$

$$\Delta a_1 = a_{1,j+1} - a_{1,j}$$

Assim, linearizou-se o modelo original em relação aos parâmetros. A equação 13 pode ser substituída na equação 12 como se segue:

$$y_i - f(x_i)_j = f(x_i)_j + \frac{\partial f(x_i)_j}{\partial a_0} \Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1} \Delta a_1 + e_i \quad (14)$$

Ou na forma de matriz:

$$\{D\} = [Z_j]\{\Delta A\} + \{E\} \quad (15)$$

Onde $[Z_j]$ é a matriz de derivadas parciais da função calculadas para os valores iniciais, j.

$$[Z_j] = \begin{bmatrix} \frac{\partial f_1}{\partial a_0} & \frac{\partial f_1}{\partial a_1} \\ \vdots & \vdots \\ \frac{\partial f_n}{\partial a_0} & \frac{\partial f_n}{\partial a_1} \end{bmatrix} \quad (16)$$

onde n é o número de pontos

$\partial f_i / \partial a_k$ é a derivada parcial da função em relação ao k-ésimo valor de parâmetro calculado no i-ésimo ponto.

O vetor $\{D\}$ contém as diferenças entre os valores calculados e os valores da função.

$$\{D\} = \begin{bmatrix} y_1 & f(x_1) \\ \vdots & \vdots \\ y_n & f(x_n) \end{bmatrix} \quad (17)$$

E o vetor $\{\Delta A\}$ contém as variações nos valores dos parâmetros.

$$\{\Delta A\} = \begin{bmatrix} \Delta a_0 \\ \Delta a_1 \end{bmatrix} \quad (18)$$

Aplicando a teoria linear dos mínimos quadrados da equação 15 resulta na seguinte equação normal:

$$\left[[Z_j]^T [Z_j] \right] \{\Delta A\} = [Z_j]^T \{D\} \quad (19)$$

Calculando-se a matriz inversa para ambos os lados da igualdade da equação 19, temos:

$$\left[[Z_j]^T [Z_j] \right]^{-1} \left[[Z_j]^T [Z_j] \right] \{\Delta A\} = \left[[Z_j]^T [Z_j] \right]^{-1} [Z_j]^T \{D\} \quad (20)$$

Para a multiplicação de uma matriz pela sua inversa temos a matriz identidade, então:

$$\{\Delta A\} = \left[[Z_j]^T [Z_j] \right]^{-1} [Z_j]^T \{D\} \quad (21)$$

Assim, a aproximação consiste na resolução da equação 21 para $\{\Delta A\}$ e pode ser implementado computacionalmente melhorando os valores dos parâmetros como:

$$a_{0,j+1} = a_{0,j} + \Delta a_0 \quad (22)$$

e

$$a_{1,j+1} = a_{1,j} + \Delta a_1 \quad (23)$$

Este procedimento é repetido até que a solução convirja, isto é, até

$$|\epsilon_a|_k = \left| \frac{a_{k,j+1} + a_{k,j}}{a_{k,j+1}} \right| 100\% \quad (24)$$

tornar-se um critério de parada aceitável ^[16].

3. RESULTADOS E DISCUSSÕES

3.1 CPMG fitting v.1

O *software* CPMG fitting v.1 foi escrito em linguagem de programação C para processamento automático das medidas qualitativas do óleo em sementes. Foi subdividido em 4 funções além da função principal “main”.

MAIN

A função “main” englobou a conversão do arquivo com extensão *.tnt para *.txt, interface com o usuário e entrada dos dados, chamada das rotinas de execução e a finalização do *software*.

A conversão do arquivo foi feita através de um *script* chamado “convert” desenvolvido em ambiente de *script* Visual Basic, fornecido pela Tecmag. A execução foi realizada através do comando “system” do Windows. Esse *script* pode ser visto no anexo A ^[14].

A interface com o usuário pode ser vista na Figura 13.

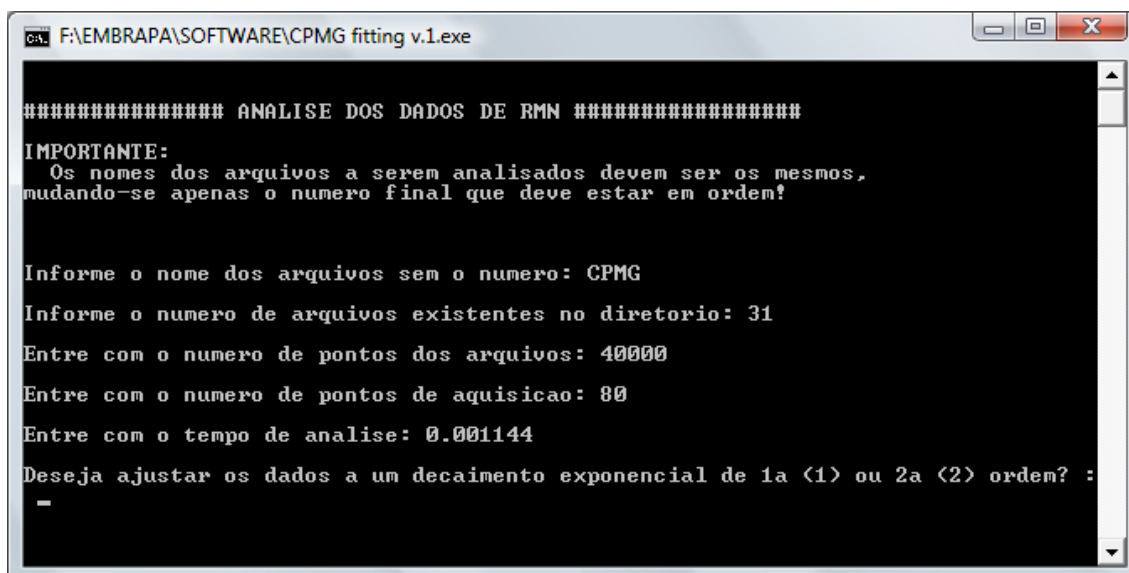


Figura 13. Interface do *software* com o usuário.

PPC1

A função “ppc1” tratou do pré-processamento dos dados. Após a exportação dos dados (Figura 14), aparecem três linhas de cabeçalho. Esse cabeçalho contém o endereço do arquivo, o número de pontos de aquisição e a linha de cabeçalho da tabela. Para o efeito de cálculos, a função “ppc1” elimina tais linhas.

Real	Imag	usec
6505.000	10857.000	0.000
15968.000	26579.000	2.000
16028.000	26590.000	4.000
16261.000	26904.000	6.000
16208.000	26898.000	8.000
16194.000	26749.000	10.000
16156.000	26935.000	12.000
16117.000	26535.000	14.000
16581.000	26848.000	16.000
16470.000	26810.000	18.000
16567.000	26559.000	20.000
16496.000	26870.000	22.000
16555.000	27019.000	24.000
16456.000	26883.000	26.000
16326.000	26689.000	28.000
16697.000	27015.000	30.000
16287.000	26356.000	32.000
16987.000	26911.000	34.000
16793.000	26475.000	36.000
16849.000	26654.000	38.000
16591.000	26598.000	40.000
16861.000	26879.000	42.000
16674.000	26686.000	44.000
16792.000	26735.000	46.000
16709.000	26304.000	48.000
16904.000	26707.000	50.000
16760.000	26365.000	52.000

Figura 14. Dados exportados em *.txt.

Na Figura 14, a primeira coluna corresponde aos valores reais. A segunda coluna corresponde aos valores imaginários, e a terceira coluna corresponde ao tempo da aquisição, dado em microssegundos.

Para o compilador utilizado, a memória de programa não é suficiente para armazenar todos os valores do arquivo em matrizes (mais de 40.000 linhas). Diante disso, utilizaram-se ponteiros para armazenar completamente os dados na matriz.

Os dados foram armazenados em uma matriz e em seguida iniciou-se a correção de fase através da DVS.

Tipicamente, utiliza-se da transformação das componentes de dados de ressonância magnética para corrigir os erros de fase na aquisição. Este procedimento representa uma simples transformação independente da detecção do ângulo de fase e pode, matematicamente, ser descrita na Eq. 25. Nesta equação as componentes da magnitude são dadas por a e b , e x é a magnitude.

$$x_{\text{magnitude}} = \sqrt{(a^2 + b^2)} \quad (25)$$

Essa transformação não diferencia valores negativos, tornando-se um problema especificamente para dados com altos níveis de ruído. Para o caso em estudo, os níveis de ruído aumentam quando a intensidade tende a zero. Este fato introduz um prejuízo nas características exponenciais dos dados de magnitude, não sendo possível um bom ajuste exponencial dos dados através da transformação da

Eq. 25, visto que para as análises, um erro muito grande no ajuste pode levar a interpretações equivocadas.

A solução para esse problema veio através da técnica PPC ^[13]. A PPC é independente dos erros na aquisição dos ângulos de fase e também tem a vantagem de filtrar qualquer ruído nas medidas individualmente.

A transformação dos dados através da PPC é representada pela DVS. Idealmente, o segundo valor singular é zero se não há variações de fase durante as medidas. Na prática, entretanto, diferenças mínimas são observadas e o segundo componente singular representa tal ruído ^[13]. O produto das primeiras matrizes singulares proporciona uma ótima representação dos dados com fase corrigida, minimizando assim a influência do ruído.

Outro aspecto muito importante desta expansão da matriz de dados em autoimagens e valores singulares é o de mostrar, sob o ponto de vista computacional, que a armazenagem dos dados contidos na matriz, que exige uma quantidade $n \times m$ de endereços de memória, pode ser diminuída consideravelmente utilizando autoimagens. Dessa forma, uma escolha apropriada do número e dos índices das autoimagens usadas na decomposição mostra a potencialidade do método no processamento de dados magnéticos.

Assim, a manipulação de matrizes e cálculos como a inversa ou a simples multiplicação tornaram-se fatores decisivos para aplicação da DVS. A solução veio através do uso de uma biblioteca desenvolvida para DEV C/C++ chamada "GMATRIX.h", para manipulação de matrizes, cuja autoria foi do professor Geovany Araújo Borges da Universidade de Brasília. Essa biblioteca simplificou o desenvolvimento do programa, visto que no decorrer do mesmo foram necessários muitos cálculos entre matrizes.

Os dados seguem as aquisições de acordo com a configuração determinada pelo usuário.

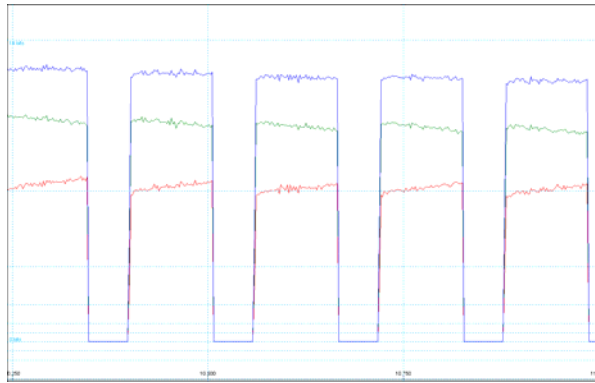


Figura 15. Detalhamento das aquisições.

Analisando o espectro da Figura 15, observa-se que existem intervalos entre as aquisições que correspondem a 0 na magnitude. Os valores de zero encontrados nos dados correspondem aos intervalos entre cada aquisição. As aquisições correspondem às amplitudes de cada eco. Observa-se também que à medida em que há o aumento nos valores imaginários de aquisição para cada eco, há a diminuição dos valores reais de aquisição dos mesmos, demonstrando haver erro de fase na aquisição dos dados.

Diante desse problema, a função “ppc1” faz a correção de fase e elimina os intervalos nulos dos dados e os armazena numa nova matriz de dados. Essa matriz é passada à função “media”.

MEDIA

Após o pré-processamento realizado na função “ppc1”, os dados foram divididos em vetores correspondentes a cada conjunto de pontos de aquisição. Cada intervalo de aquisição de dados é determinado por certo número de pontos, de acordo com a escolha do usuário, porém trata-se de uma variável que pôde ser calculada a partir de outras variáveis de entrada no *software*. Esses pontos estão sujeitos a variações devido a ruídos do equipamento. Tais variações são mostradas na Figura 16, para os pontos de aquisição dos dados da magnitude (azul), real (verde) e imaginário (vermelho).

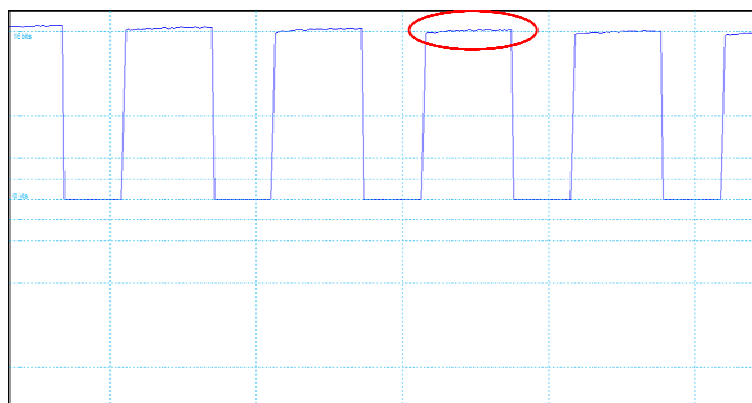


Figura 16. Variações nas aquisições.

Para minimizar esses erros a função “ppc1” calcula a média de cada aquisição para que seja utilizado apenas esse valor no ajuste da curva, diminuindo consideravelmente a quantidade de dados armazenados.

Nos últimos pontos do espectro, apesar de a magnitude ser muito baixa, não decai totalmente a zero. Esse fato pode ser mostrado na Figura 17, correspondente aos últimos pontos de aquisição.



Figura 17. Últimos 4 pontos de aquisição.

Sendo assim, foi necessário forçá-lo a zero obtendo o ajuste sem *off-set*, para efeito de comparação entre as amostras. O fato de a amplitude do sinal não decair totalmente a zero pode ser explicado pela teoria de RMN, na qual apenas 99% das isocromatas de spin retornam ao equilíbrio térmico.

Calculadas as médias, foi feita a normalização. O objetivo desse processo foi possibilitar a comparação da qualidade de óleo entre amostras distintas.

Após processados os dados, foi montado um vetor de tempo de aquisição a partir do dado “Analysis Time” inserido pelo usuário para corresponder às médias normalizadas e possibilitar o ajuste da curva.

Para as manipulações das matrizes e criação de vetores citadas para a função “media” foram utilizados os comandos da biblioteca “GMATRIX.h”.

Os dados de média calculados foram armazenados numa nova matriz de proporções da ordem de até 300 vezes menor que a matriz original contendo todos os dados. Essa matriz foi passada à função de ajuste.

PRI_ORD

A função “pri_ord” é chamada a partir da função “main”, dependendo da opção do usuário de ajuste de primeira ou segunda ordem. Nesse caso, a função “pri_ord” faz o ajuste mono-exponencial do modelo da equação 9.

Para utilizar o método de Gauss-Newton são necessários bons valores iniciais para A_1 e t_1 , pois trata-se de um método de iteração. Valores iniciais longe dos valores reais de ajuste não garantem a convergência do método.

O desenvolvimento do *software* teve por objetivo o ganho de tempo nas análises, portanto, além de ser rápido, teve de ser versátil, com capacidade para analisar amostras diferentes num mesmo processamento, sem a necessidade de ter que parar o processamento de amendoim, por exemplo, para analisar mamona, que possuem diferentes valores de A_1 e t_1 .

Nesse sentido, desenvolveu-se um método para encontrar os valores iniciais adequados a cada amostra no momento da análise, de maneira a garantir a convergência para cada amostra.

O método foi baseado nas características da equação exponencial. Para o valor inicial de A_1 foi utilizada a média entre os 10 pontos iniciais da matriz de magnitudes. Essa média entre os pontos foi calculada para não haver a dependência de um único ponto apenas.

Para o valor de t_1 ajustou-se uma reta a 7% dos dados de média iniciais. Essa quantidade de valores utilizados para o ajuste da reta é suficiente para obter a inclinação inicial da curva.

O exemplo do amendoim a seguir ilustra a metodologia usada.

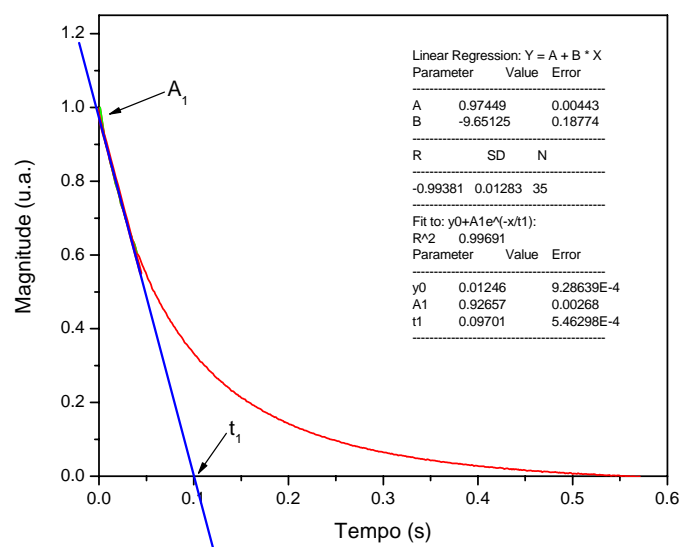


Figura 18. Retra ajustada à curva para a determinação dos parâmetros iniciais.

Na Figura 18 observa-se o decaimento de uma amostra de amendoim. Nesse caso foram separados 7% dos pontos iniciais, correspondentes a 35 pontos, e ajustou-se uma reta. O valor inicial de A_1 encontra-se indicado na figura e corresponde ao ponto em que a reta cruza o eixo de amplitudes. Esse valor é o coeficiente linear da reta, 0,97.

Na curva exponencial, a constante elevada a potencia de e equivale ao inverso do ponto da reta onde cruza o eixo de tempo. Dessa maneira, no modelo da equação 25, t_1 é encontrado diretamente no gráfico.

Para o exemplo, t_1 foi encontrado a partir da equação linear:

$$t_1 = \frac{A}{B} \quad (25)$$

$$t_1 = \frac{0,97}{-9,65} = -0,10 \quad (26)$$

O valor negativo de t_1 explica-se pelo fato de tratar-se de uma reta decrescente. Esse valor negativo foi corrigido no *software* multiplicando-se t_1 por -1. Esse fato pode ser observado na Figura 18, onde a reta corta o eixo de tempo.

Nesse exemplo, os valores iniciais encontrados para A_1 e t_1 foram 0,97 e 0,10 respectivamente. Observa-se que após o ajuste utilizando o *software* Origin, foram encontrados 0,926 e 0,097 para A_1 e t_1 , respectivamente. O coeficiente de correlação encontrado foi de 0,997, indicando um bom ajuste exponencial para os dados.

Verificamos também que os valores iniciais encontrados pelo método da reta tangente aos pontos estão próximos dos valores encontrados pelo ajuste, garantindo a convergência do método de Gauss-Newton.

O critério de finalização para ajuste mono-exponencial foi representado para um resíduo de 0,1. Para o caso de ajustes cujos erros não foram pequenos suficientes para atingir o critério de finalização, desenvolveu-se uma rotina de maneira a elevar o valor de erro limite e possibilitar a continuidade da execução do programa. Para esses casos não pudemos garantir uma boa aproximação dos dados, obtendo assim um baixo valor de coeficiente de correlação.

Calculados os valores de ajuste, foi feita a gravação dos dados no arquivo “coeficientesMONO.txt”, e a execução foi finalizada.

SEC_ORD

A função “sec_ord” é inicializada a partir da função “main” dependendo da escolha efetuada pelo usuário. Neste caso, a função realiza o ajuste bi-exponencial dos dados na forma da equação 10.

O ajuste foi realizado através do método de Gauss-Newton seguindo os mesmos procedimentos descritos para o ajuste mono-exponencial, porém houve o agravante de encontrar os valores iniciais para garantir a convergência do método, visto que agora é necessário que se encontre quatro parâmetros iniciais próximos dos valores reais para garantir a convergência de Gauss-Newton.

A função bi-exponencial é a soma de duas funções mono-exponenciais, este fato auxiliou no desenvolvimento de um método para o cálculo dos parâmetros iniciais.

Para as amostras utilizadas, verificou-se que são compostas por mais de 1 tipo de óleo, dos quais podemos citar os saturados, monoinsaturados e poliinsaturados. Os saturados apresentam um decaimento mais rápido por serem de mais alta viscosidade em relação aos demais. Sendo assim, é possível separar os tipos de óleo em dois grandes grupos, os saturados e os insaturados.

Os dados foram divididos em duas exponenciais. A divisão da curva referente aos dados de amendoim segue nas Figuras 19 e 20, assim como as retas ajustadas a cada uma delas nas Figuras 21 e 22.

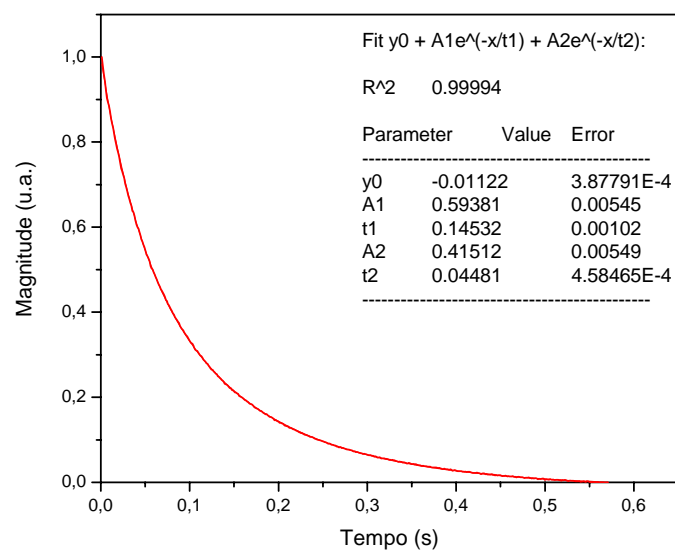


Figura 19. Decaimento do amendoim ajustada à curva bi-exponencial.

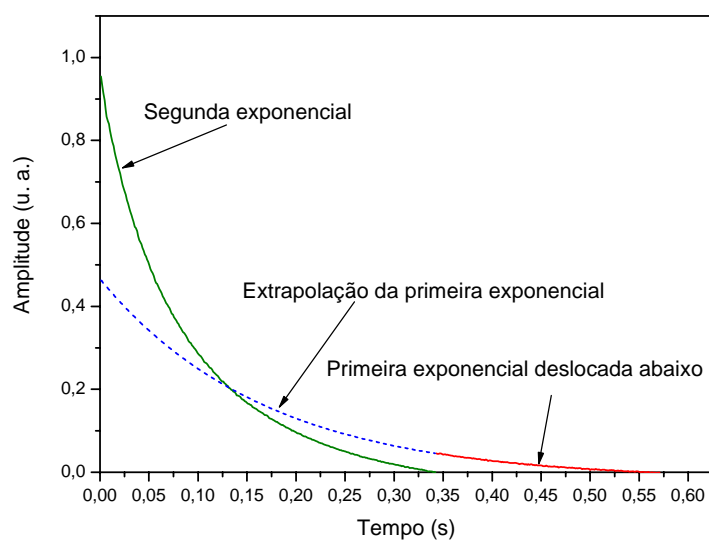


Figura 20. Divisão do decaimento do amendoim, 60% dos pontos refere-se à primeira exponencial e o restante referente à segunda exponencial.

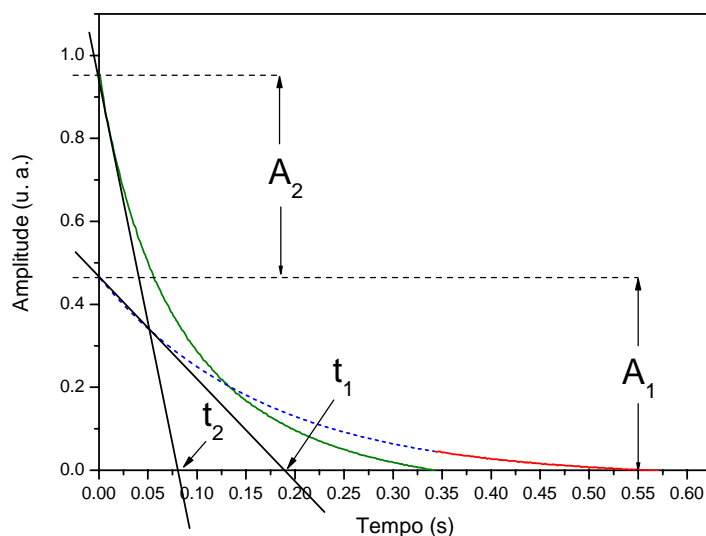


Figura 21. Retas ajustadas aos primeiros pontos e amplitudes.

Da mesma maneira que foram encontrados os valores iniciais A_1 e t_1 na função “pri_ord”, pode-se encontrar os valores de A_1 , t_1 , A_2 e t_2 iniciais através do gráfico. Assim:

$$A_1 = 0,47 \text{ e } t_1 = 0,18$$

$$A_2 = 0,95 - 0,47 = 0,48 \text{ e } t_2 = 0,06$$

Na Figura 19 observa-se que após o ajuste utilizando o *software* Origin, foram encontrados 0,59 e 0,14 para A_1 e t_1 , respectivamente, e 0,41 e 0,04 para A_2 e t_2 , respectivamente. O coeficiente de correlação encontrado foi de 0,999, indicando um bom ajuste exponencial para os dados. Verifica-se também que os valores iniciais encontrados pelo método da reta tangente aos pontos estão próximos dos valores encontrados pelo ajuste, garantindo a convergência do método de Gauss-Newton.

Quanto maior o número de exponenciais a serem ajustadas à mesma curva, mais complexo fica o algoritmo para a convergência pelo método de Gauss-Newton, pois exige-se uma maior quantidade de parâmetros iniciais, além de terem de estar mais próximos dos valores reais. Se o objetivo fosse separar a curva em três mono-exponenciais, ou seja, separar os saturados, os monoinsaturados e os poliinsaturados, então os parâmetros iniciais teriam que ser muito mais próximos dos parâmetros originais da curva, uma pequena variação de qualquer um dos seis parâmetros já seria suficiente para a não convergência do método.

3.2 CPMG fitting v.2

O CPMG fitting v.2 foi criado visando a utilização em tempo real. Para isso, foi estudada uma maneira de se adaptar o CPMG fitting v.1 ao *software* NTNMR.

O NTNMR é provido de habilidades para automatizar muitos de seus processos. É possível fazer aquisições mais sofisticadas utilizando-se de longas rotinas e processos que podem ser simplificados e executados em um simples *click*. Aplicações que incluem o processamento complexo de dados, execução automática de série de experimentos, calibração de experimentos ou ainda criação de uma especialização virtual para a interface do NTNMR.

O termo “scripting” tem sido adotado para referir a automação de processos no NTNMR por causa da sua capacidade de superar o que é tipicamente referido como “Macros”. Conceitualmente, entretanto, “scripts” no NTNMR pode ser entendido como “macros” em outros *softwares*.

O Visual Basic 6.0 e o VBScript foi adotado pela Tecmag como linguagem de *Script* suportada oficial.

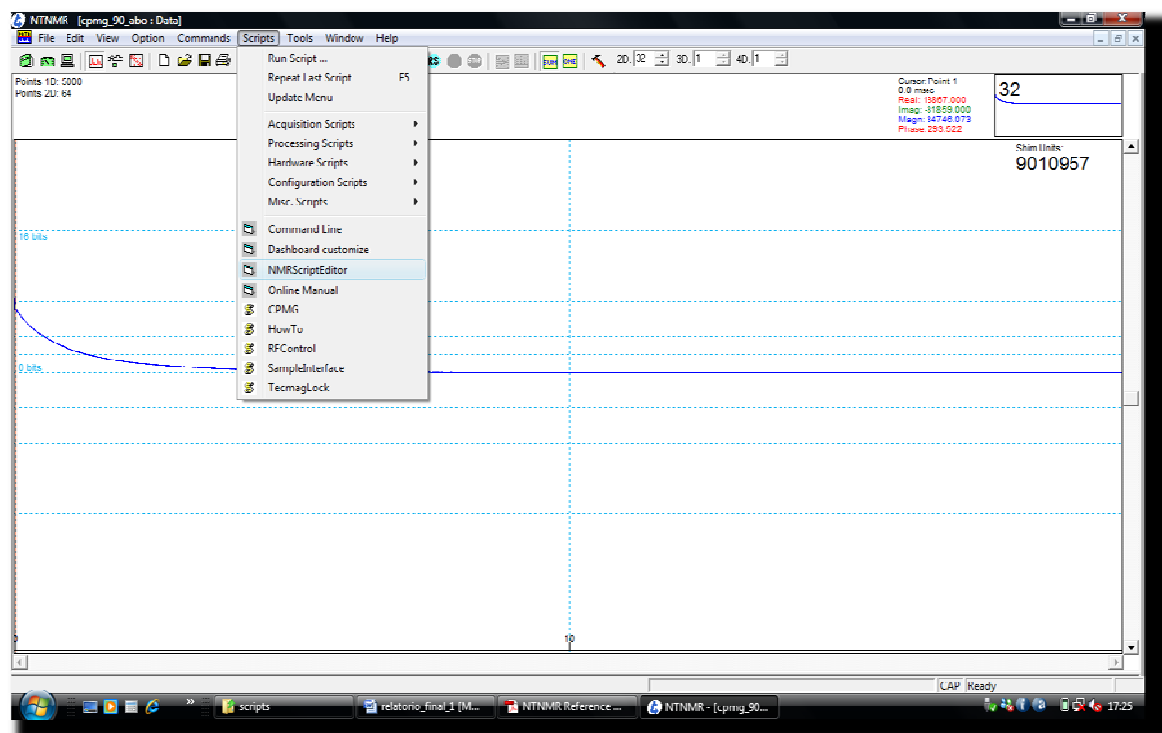


Figura 222. Menu de edição de *scripts*.

O menu de *scripts* do NTNMR como é visto na Figura 23, permite que o usuário crie, edite e execute *scripts* salvos como executáveis (*.exe) ou arquivos NMRScript

(*vbs). Todos os arquivos salvos em “NTNMR\scripts\” são mostrados no menu *scripts* do NTNMR [18].

Para que o CPMG fitting v.2 executasse a tarefa em tempo real, foi criado um *script*, visto a facilidade de utilizar essa ferramenta através do NTNMR.

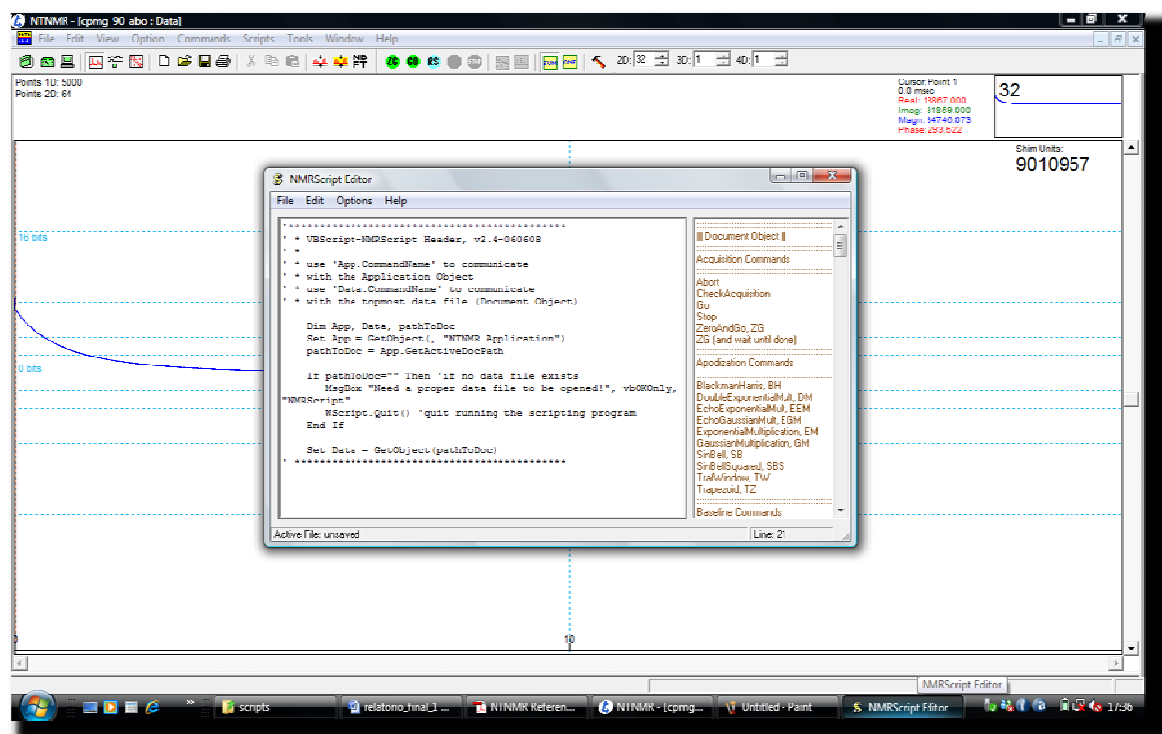


Figura 233. Interface de edição de *scripts* do NTNMR.

Quando o editor de *scripts* é acionado, um novo arquivo é automaticamente criado com uma pré-comunicação inserida como cabeçalho (Figura 24). Essa comunicação é necessária para estabelecer um *link* com o NTNMR. O editor de *scripts* contém alguns comandos de utilização rápida para execução de algumas funções específicas do NTNMR. Esse fator é vantajoso por facilitar a edição de um novo *script*, clicando apenas nos comandos a serem inseridos.

É possível fazer a aquisição de dados localizados na interface do NTNMR através do comando “Data.GetNMRParameter”. Porém, alguns dados de entrada como o tempo de análise e a ordem do decaimento não são explícitos na interface do NTNMR. Contudo, esse oferece uma ferramenta importante, a criação de variáveis locais e globais. Foram criadas duas variáveis locais, a “Analysis Time” e a “Decay Order”. Isso possibilitou que se insiram as entradas apenas uma vez durante um conjunto de experimentos, além de tornar desnecessária a interface do CPMG fitting v.2 com o usuário.

O fluxograma de automação proposta pelo *script* segue na Figura 25.

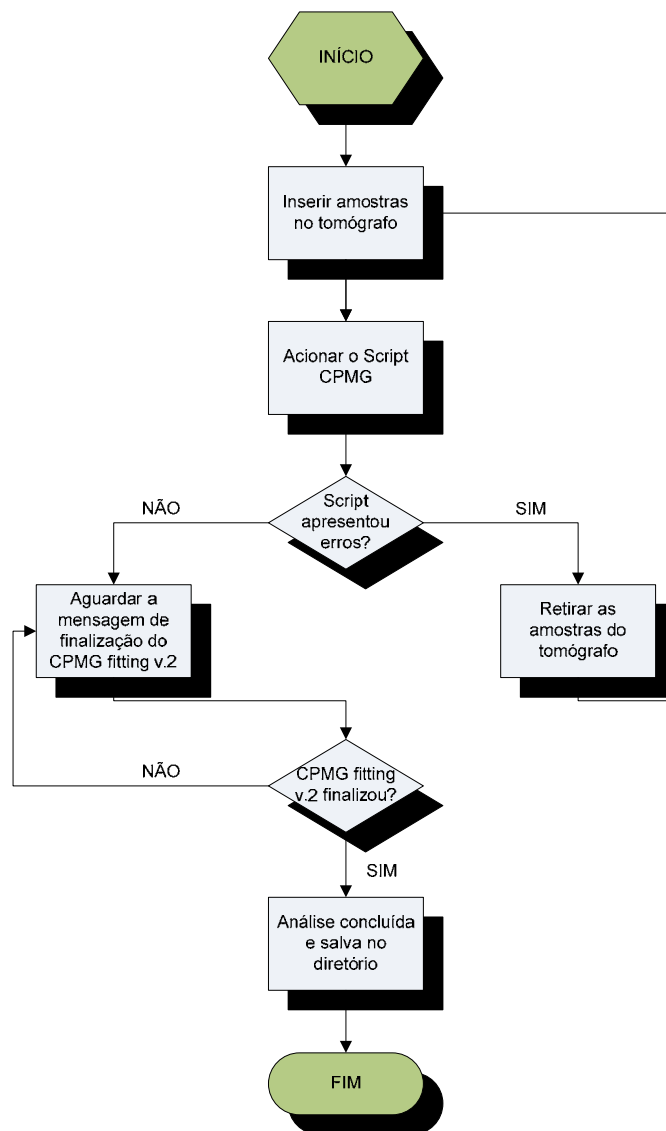


Figura 244. Fluxograma referente ao desenvolvimento do *script*.

Baseado no fluxograma, desenvolveu-se o novo *script* apresentado no anexo D e este recebeu o nome de CPMG.

Para fazer com que o *script* passasse os valores de entrada ao CPMG fitting v.2, foi necessário armazená-los através de um arquivo *.txt. Assim, o *script* gravou dados no arquivo e em seguida o CPMG fitting v.2 fez a leitura dos mesmos, seguindo a sua execução.

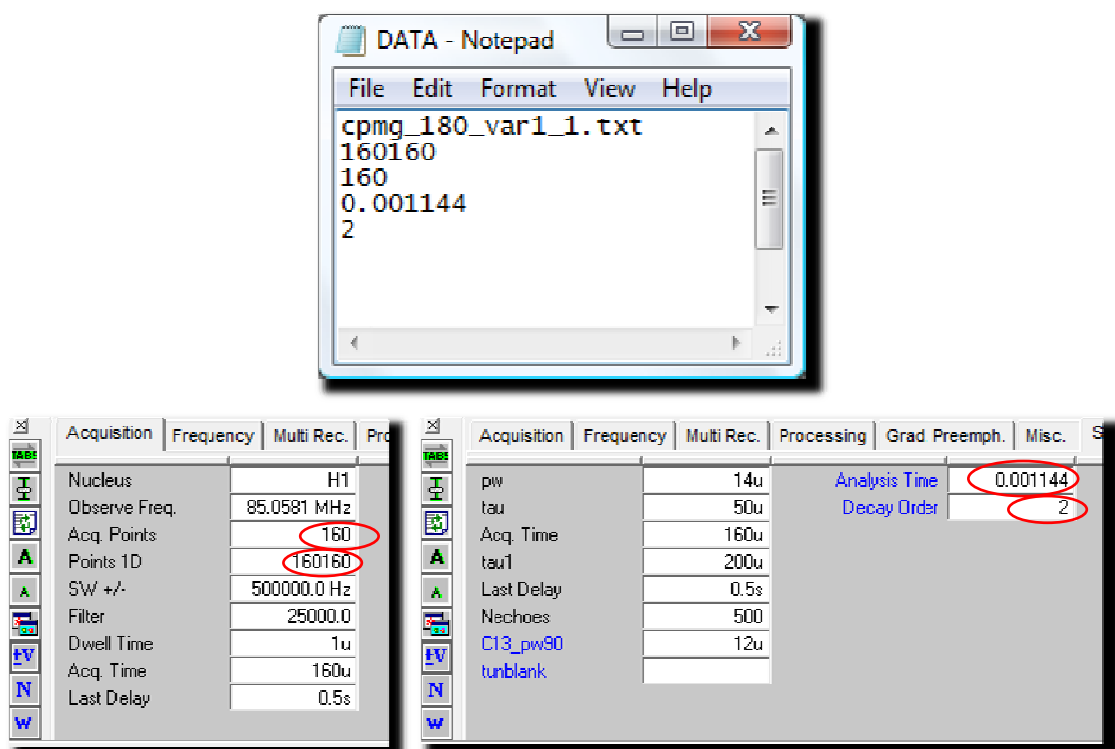


Figura 255. Indicação do local onde o *script* faz a leitura dos dados de entrada no NTNMR.

A Figura 26 apresenta o arquivo “DATA.txt” gerado pelo *script*. A primeira linha corresponde ao nome do arquivo a ser processado. A segunda linha corresponde ao número de pontos do arquivo gerado pelo *script* “convert”. A terceira linha corresponde ao número de pontos de aquisição. A quarta linha corresponde ao tempo de análise (Analysis Time). E a última linha corresponde à ordem de ajuste (Decay Order), de primeira ou segunda ordem.

Com o novo *script* concluído, foi necessário alterar o CPMG fitting v.1 de modo que não fosse mais necessário a entrada dos dados, visto que tais dados já são armazenados no arquivo “DATA.txt”.

A alteração no CPMG fitting v.1 consistiu na busca, leitura e armazenamento dos dados encontrados no arquivo “DATA.txt”. A nova versão do *software* recebeu o nome de “CPMG fitting v.2” e a alteração foi efetuada apenas na função “main”, que deixou de ter interface com o usuário, visto que esse papel ficou para o *script*, apenas exibindo mensagens de informação, alerta e erro. Assim, o usuário não precisou mais inserir qualquer dado durante a execução do *software*. A função “main” do CPMG fitting v.2 pode ser observada no anexo C.

3.3 Avaliação do software

Para a avaliação do CPMG fitting foram utilizadas 31 amostras oleaginosas variadas: abóbora, algodão, amêndoa, amendoim, andiroba, baru, castanha do Brasil, bucha, café, castanha de caju, côco, gergelim, girassol, linhaça, macadâmia, macaúba, mamona, melancia, moringa, chapéu-de-napoleão, níger, noz, pati, pequi carne, pequi semente, piaçava, pinhão-manso, pistache, rabanete, soja e tucumã.

Foram utilizadas tais amostras pelo fato de já terem sido analisadas pelo método manual e assim se ter parâmetros para validar as respostas calculadas pelo CPMG fitting.

Foram realizados 2 testes, o primeiro para a determinação dos coeficientes do decaimento de primeira ordem e em seguida para a determinação dos coeficientes para decaimento de segunda ordem. Os resultados gerados pelo programa foram salvos em arquivos do tipo *.txt e são mostrados nas Figuras 27 e 28.

coeficientesMONO - Notepad

File Edit Format View Help

ps dados estao ajustados na forma:
 $f(t) = A1 \cdot \exp(-t/t1)$

CPMG1.txt	- A1: 0.945493	t1: 0.107493	R2: 0.996262
CPMG2.txt	- A1: 0.943067	t1: 0.104491	R2: 0.996561
CPMG3.txt	- A1: 0.905411	t1: 0.095223	R2: 0.993327
CPMG4.txt	- A1: 0.929156	t1: 0.097828	R2: 0.995183
CPMG5.txt	- A1: 0.924572	t1: 0.061857	R2: 0.990122
CPMG6.txt	- A1: 0.932653	t1: 0.090165	R2: 0.994815
CPMG7.txt	- A1: 0.943316	t1: 0.098512	R2: 0.995552
CPMG8.txt	- A1: 0.961940	t1: 0.112694	R2: 0.997782
CPMG9.txt	- A1: 0.895341	t1: 0.072759	R2: 0.990006
CPMG10.txt	- A1: 0.944469	t1: 0.091853	R2: 0.995253
CPMG11.txt	- A1: 0.964945	t1: 0.111461	R2: 0.997700
CPMG12.txt	- A1: 0.938281	t1: 0.105418	R2: 0.995846
CPMG13.txt	- A1: 0.953888	t1: 0.113044	R2: 0.997351
CPMG14.txt	- A1: 0.944618	t1: 0.123453	R2: 0.997249
CPMG15.txt	- A1: 0.984740	t1: 0.084770	R2: 0.996548
CPMG16.txt	- A1: 0.966823	t1: 0.100363	R2: 0.996962
CPMG17.txt	- A1: 1.012337	t1: 0.019058	R2: 0.989397
CPMG18.txt	- A1: 0.961068	t1: 0.119568	R2: 0.998119
CPMG19.txt	- A1: 0.936587	t1: 0.086964	R2: 0.995264
CPMG20.txt	- A1: 0.933230	t1: 0.093249	R2: 0.995035
CPMG21.txt	- A1: 0.912101	t1: 0.110864	R2: 0.994567
CPMG22.txt	- A1: 0.950626	t1: 0.118181	R2: 0.997327
CPMG23.txt	- A1: 0.958055	t1: 0.108303	R2: 0.997444
CPMG24.txt	- A1: 0.949855	t1: 0.101539	R2: 0.997307
CPMG25.txt	- A1: 0.946695	t1: 0.096067	R2: 0.996298
CPMG26.txt	- A1: 0.968232	t1: 0.104457	R2: 0.997300
CPMG27.txt	- A1: 0.913360	t1: 0.102467	R2: 0.994523
CPMG28.txt	- A1: 0.932876	t1: 0.098462	R2: 0.995118
CPMG29.txt	- A1: 0.922854	t1: 0.077439	R2: 0.992378
CPMG30.txt	- A1: 0.928212	t1: 0.099091	R2: 0.995446
CPMG31.txt	- A1: 0.944916	t1: 0.101946	R2: 0.997181

Figura 266. Resultado para coeficientes mono-exponenciais.

coeficientesBI - Notepad

File Edit Format View Help

Os dados estão ajustados na forma:
 $f(t) = A1 \cdot \exp(-t/t1) + A2 \cdot \exp(-t/t2)$

CPMG1.txt	- A1: 0.257383	t1: 0.032284	A2: 0.771655	t2: 0.124750	R2: 0.999670
CPMG2.txt	- A1: 0.266338	t1: 0.035133	A2: 0.751465	t2: 0.122613	R2: 0.999654
CPMG3.txt	- A1: 0.341294	t1: 0.029011	A2: 0.676524	t2: 0.118363	R2: 0.999784
CPMG4.txt	- A1: 0.317710	t1: 0.032720	A2: 0.703396	t2: 0.119303	R2: 0.999802
CPMG5.txt	- A1: 0.605592	t1: 0.030025	A2: 0.428630	t2: 0.102355	R2: 0.999897
CPMG6.txt	- A1: 0.364578	t1: 0.033592	A2: 0.660025	t2: 0.114342	R2: 0.999865
CPMG7.txt	- A1: 0.338269	t1: 0.037209	A2: 0.687419	t2: 0.122048	R2: 0.999775
CPMG8.txt	- A1: 0.194470	t1: 0.033511	A2: 0.830302	t2: 0.125420	R2: 0.999632
CPMG9.txt	- A1: 0.487969	t1: 0.028531	A2: 0.528274	t2: 0.105557	R2: 0.999668
CPMG10.txt	- A1: 0.348153	t1: 0.033535	A2: 0.686187	t2: 0.114480	R2: 0.999850
CPMG11.txt	- A1: 0.201044	t1: 0.033542	A2: 0.828081	t2: 0.124562	R2: 0.999652
CPMG12.txt	- A1: 0.278156	t1: 0.032926	A2: 0.746972	t2: 0.124388	R2: 0.999705
CPMG13.txt	- A1: 0.202335	t1: 0.030934	A2: 0.823730	t2: 0.126213	R2: 0.999618
CPMG14.txt	- A1: 0.167723	t1: 0.026280	A2: 0.852723	t2: 0.134048	R2: 0.999268
CPMG15.txt	- A1: 0.541201	t1: 0.049119	A2: 0.502622	t2: 0.120589	R2: 0.999783
CPMG16.txt	- A1: 0.297039	t1: 0.039137	A2: 0.737446	t2: 0.119936	R2: 0.999763
CPMG17.txt	- A1: 0.861770	t1: 0.010748	A2: 0.305824	t2: 0.040003	R2: 0.999782
CPMG18.txt	- A1: 0.148955	t1: 0.028794	A2: 0.872176	t2: 0.128895	R2: 0.999486
CPMG19.txt	- A1: 0.381212	t1: 0.035648	A2: 0.638212	t2: 0.111827	R2: 0.999757
CPMG20.txt	- A1: 0.350926	t1: 0.034319	A2: 0.671926	t2: 0.116913	R2: 0.999835
CPMG21.txt	- A1: 0.268726	t1: 0.027287	A2: 0.752399	t2: 0.128953	R2: 0.999572
CPMG22.txt	- A1: 0.183757	t1: 0.028417	A2: 0.841506	t2: 0.129976	R2: 0.999489
CPMG23.txt	- A1: 0.211253	t1: 0.031199	A2: 0.818207	t2: 0.121671	R2: 0.999712
CPMG24.txt	- A1: 0.243633	t1: 0.033544	A2: 0.776161	t2: 0.117011	R2: 0.999811
CPMG25.txt	- A1: 0.305613	t1: 0.034476	A2: 0.720935	t2: 0.115788	R2: 0.999850
CPMG26.txt	- A1: 0.255195	t1: 0.037481	A2: 0.777673	t2: 0.121735	R2: 0.999725
CPMG27.txt	- A1: 0.300484	t1: 0.030772	A2: 0.712170	t2: 0.123239	R2: 0.999545
CPMG28.txt	- A1: 0.321555	t1: 0.033182	A2: 0.703507	t2: 0.120344	R2: 0.999794
CPMG29.txt	- A1: 0.472853	t1: 0.031987	A2: 0.555359	t2: 0.108966	R2: 0.999897
CPMG30.txt	- A1: 0.300269	t1: 0.032004	A2: 0.718513	t2: 0.119155	R2: 0.999752
CPMG31.txt	- A1: 0.235092	t1: 0.031244	A2: 0.784268	t2: 0.116646	R2: 0.999802

Figura 277. Resultados para coeficientes bi-exponenciais.

Os dados encontrados estão coerentes quando comparados ao método manual. Essa comparação segue nas Tabelas 1 e 2.

Tabela 1. Valores de T_2 ajustados mono-exponencialmente.

Sementes	Manual	Automático
	T_2	T_2
Abóbora	0,105	0,107
Algodão	0,101	0,104
Amêndoas	0,093	0,095
Amendoim	0,094	0,098
Andiroba	0,060	0,062
Baru	0,088	0,090
Castanha do Brasil	0,097	0,099
Bucha	0,110	0,113
Café	0,069	0,073
Castanha de Caju	0,092	0,092
Coco	0,108	0,111
Gergelim	0,103	0,105
Girassol	0,109	0,113
Linhaça	0,123	0,123
Macadâmia	0,084	0,085
Macaúba	0,100	0,100
Mamona	0,019	0,019
Melancia	0,119	0,120
Moringa	0,084	0,087
Chapéu de Napoleão	0,090	0,093
Niger	0,106	0,111
Noz	0,116	0,118
Pati	0,106	0,108
Pequi carne	0,098	0,102
Pequi semente	0,093	0,096
Piaçava	0,104	0,104
Pinhão manso	0,100	0,102
Pistache	0,095	0,098
Rabanete	0,076	0,077
Soja	0,097	0,099
Tucumã	0,099	0,102

Tabela 2. Valores de T_2 ajustados bi-exponencialmente.

Sementes	Manual	Automático	Manual	Automático
	T₂1	T₂1	T₂2	T₂2
Abóbora	0,160	0,125	0,048	0,032
Algodão	0,164	0,123	0,054	0,035
Amêndoas	0,138	0,118	0,038	0,029
Amendoim	0,144	0,119	0,044	0,033
Andiroba	0,109	0,102	0,032	0,030
Baru	0,132	0,114	0,040	0,034
Castanha do Brasil	0,169	0,122	0,055	0,037
Bucha	0,175	0,125	0,060	0,034
Café	0,107	0,106	0,029	0,029
Castanha de Caju	0,137	0,114	0,043	0,034
Coco	0,165	0,125	0,055	0,034
Gergelim	0,157	0,124	0,047	0,033
Girassol	0,162	0,126	0,050	0,031
Linhaça	0,185	0,134	0,053	0,026
Macadâmia	0,164	0,121	0,055	0,049
Macaúba	0,164	0,120	0,056	0,039
Mamona	0,040	0,040	0,011	0,011
Melancia	0,178	0,129	0,059	0,029
Moringa	0,132	0,112	0,044	0,036
Chapéu de Napoleão	0,148	0,117	0,048	0,034
Niger	0,156	0,129	0,039	0,027
Noz	0,174	0,130	0,052	0,028
Pati	0,150	0,122	0,050	0,031
Pequi carne	0,143	0,117	0,050	0,034
Pequi semente	0,138	0,116	0,045	0,034
Piaçava	0,170	0,122	0,058	0,037
Pinhão manso	0,147	0,123	0,043	0,031
Pistache	0,146	0,120	0,044	0,033
Rabanete	0,123	0,109	0,037	0,032
Soja	0,144	0,119	0,043	0,032
Tucumã	0,139	0,117	0,044	0,031

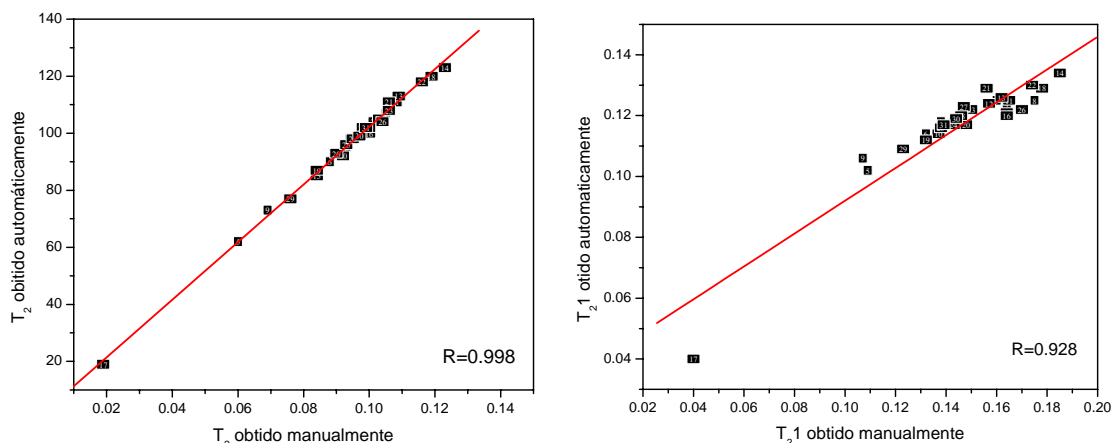


Figura 288. Comparação entre os valores ajustado manualmente e automaticamente, a) valores obtidos por ajuste mono-exponencial e b) valores obtidos por ajuste bi-exponencial (somente os valores da primeira componente T_{21} foram apresentados).

Na Figura 28 verifica-se que para o ajuste mono-exponencial, a correlação entre os dados obtidos manualmente e os calculados pelo CPMG fitting é satisfatória, com índice de correlação bem próximo de 1. Para o caso bi-exponencial houve uma correlação de 0,92 para o decaimento da primeira exponencial. Isso se deve ao fato da maior dificuldade em separar-se os dados sempre em 40% para a primeira exponencial, pois há uma grande variedade de amostras no conjunto analisado. Os valores de T_2 para as amostras é bastante diferenciado. Assim a generalização para 40% pode explicar eventuais diferenças entre os coeficientes calculados manualmente e os calculados via CPMG fitting. Outro fator que pode trazer baixos valores de correlação é o modo manual de fazer a correção de fase, utilizando uma ferramenta do *software* NTNMR que consiste em minimizar a parte imaginária dos dados apenas, e não utiliza o método PPC.

Na Tabela 3 pode-se visualizar a porcentagem total de ácidos graxos saturados, monoinsaturados e poliinsaturados ^[17].

Tabela 3. Porcentagem de ácidos graxos nas amostras.

Amostras	Ácidos graxos sat. (%)	Ácidos graxos monoinsat. (%)	Ácidos graxos poliinsat. (%)
Linhaça	8,40	24,5	67,1
Amêndoa	8,61	59,46	31,93
Soja 1	17,09	31,04	51,87
Macadâmia	11,42	70,11	1,64
Baru	12,32	52,70	34,97
Milho Pipoca	13,83	28,55	57,61
Amendoim-g	14,91	62,06	23,02
Milho BR	15,13	31,96	52,90
Gergelin Preto	15,51	41,94	42,54
Amendoim-p	16,00	45,50	38,50
Castanha Caju	16,30	61,00	22,70
Girassol	19,10	49,55	31,33
Pinhão Manso	19,32	36,54	42,94
Abóbora	22,31	18,53	59,35
Castanha Para	28,90	38,8	32,30
Média	15,94	43,48	39,38
Desvio Padrão	5,22	15,37	16,95

Através da Tabela 3 observa-se que a proporção entre os ácidos saturados e os insaturados, na média, segue a relação 15:85. Observando-se a Figura 27 tem-se que a relação entre as amplitudes A_1 e A_2 também segue, na média, a relação 15:85, demonstrando que a curva bi-exponencial é composta por duas curvas mono-exponenciais das quais uma representa o decaimento de ácidos saturados e outra representa o decaimento dos ácidos insaturados.

4. CONCLUSÃO

O *software* foi escrito em linguagem de programação C, onde as várias ações antes realizadas manualmente passaram a ser processadas automaticamente, levando aos resultados das medidas qualitativas do óleo em sementes.

O fato de tratar-se de um *software* extremamente rápido para o processamento dos dados permite que ele seja utilizado para o processamento em larga escala, sem custos com a compra do compilador DEV C/C++, visto que este é um compilador gratuito. Assim, é possível utilizá-lo em escala comercial sendo necessário apenas entrar com as configurações iniciais das amostras, deixando os cálculos sobre a responsabilidade do CPMG fitting.

No método manual, a correção de fase dependia da capacidade de análise do usuário, sendo passível de erros. A correção de fase através da PPC trouxe grandes vantagens quando comparada ao método manual, pois a mesma agora torna-se padronizada, independentemente da influência do usuário. A utilização do DVS trouxe também mais precisão à correção de fase, visto que se trata de um modelo matemático e independente de qualquer influência externa.

O CPMG fitting V.2 permitiu a análise das amostras em tempo real, gerando resultados segundos após a aquisição dos dados. Este fato é relevante, visto interesse em realizar análises em larga escala. É possível separar cada amostra de acordo com sua classe de oleosidade no momento em que a mesma é retirada do equipamento de análise.

Diante dos testes, o CPMG fitting apresentou um bom desempenho para as análises. A principal vantagem adquirida através do *software* se refere ao tempo de análise dos dados. O processamento das 31 amostras levou apenas 58s para ser concluído, ou seja, a amostra que levava 4 minutos para ser manualmente processada ou 15 minutos para ser processada através do *software* desenvolvido em Matlab passou a ser processada em apenas 1,87s. Isso equivale a uma diminuição de mais de 100 vezes no tempo de processamento manual e quase 500 vezes para o processamento via Matlab.

REFERÊNCIAS BIBLIOGRÁFICAS

1. PREUSS, E. **Linguagem de Programação II: C, Pascal e Delphi**. Universidade Regional Integrada do Alto Uruguai e das Missões. 2006.
2. Bloodshed Software. **DEV – C/C++ Download Free Software**. Disponível na internet. <http://www.bloodshed.net/>. 2008.
3. NORDON, A.; MCGILL, C. A.; LITTLEJOHN, D. **Process NMR spectrometry**. *Analyst*, 126, 260-272, 2001.
4. BELTON, P. S.; HILLS, B. P.; WEBB, G. A. **Advances in magnetic resonance in food science**. The Royal Chemical Society: Cambridge, 1999.
5. LE BOTLAN, D. L.; OUGUERRAM, L. **Spin-spin relaxation time determination of intermediate states in heterogeneous products from free induction decay NMR signals**. *Analytica Chimica Acta*, 349, 339-347, 1997.
6. GIL, V. M. S.; GERALDES, C. F. G. C. **Ressonância Magnética Nuclear: fundamentos, métodos e aplicações**. Lisboa: Fundação Calouste Gulbenkian, 1987.
7. HAHN, E. L. **Spin echoes**. *Physical Review*, 80, 580-594, 1950.
8. CARR, H. Y.; PURCELL, E. M. **Effects of diffusion on free precession in nuclear magnetic resonance experiments**. *Physical Review*, v. 94, p. 630-638, 1954.
9. MEIBOOM, S.; GILL, D. **Modified spin-echo method for measuring nuclear relaxation times**. *The Review of Scientific Instruments*, v. 29, p. 688-691, 1958.
10. COLNAGO, L. A.; MARTIN-NETO, L.; BISCEGLI, C. I.; NASCIMENTO, O. R.; BONAGAMBA, T. J.; PANEPUCCI, H.; VIEIRA, E. M.; SEIDL, P. R.; SPOSITO, G.; OPELLA, S. J. **Aplicações da ressonância magnética nuclear (RMN) e ressonância paramagnética eletrônica (EPR)**. In: CRESTANA, S.; CRUVINEL, P. E.; MASCARENHAS, S.; BISCEGLI, C. I.; MARTIN-NETO, L.; COLNAGO, L. A. *Instrumentação agropecuária: contribuições no limiar do novo século*. Brasília: EMBRAPA-SPI, 1996. Cap.1, p. 15-50.
11. COLNAGO, L. A. **Análise do teor de óleo em sementes por RMN**. São Carlos: Embrapa Instrumentação Agropecuária, 1996. 14p. (Embrapa Instrumentação Agropecuária. Circular Técnica, 3). Tipo: SP (PROCI-96.00022).
12. RUBEL, G. **Simultaneous determination of oil and water contents in different oilseeds by pulsed nuclear-magnetic-resonance**. *Journal of the American Oil Chemists Society*, 71, 1057-1062, 1994.
13. PEDERSEN, H. T.; BRO, R.; ENGELSEN, S. B. **Towards Rapid and Unique Curve Resolution of Low-Field NMR Relaxation Data: Trilinear SLICING versus Two Dimensional Curve Fitting**. *Journal of Magnetic Resonance*. 157f. 141-155, 2002.
14. SCHILDT, H. **C – Completo e Total**. e.3, Makron Books, 827p., 1997.
15. ZERI, A. C. **Aplicação de análise multivariada de dados a espectroscopia de alta resolução em sólidos por ressonância magnética nuclear**. 105f. Dissertação de mestrado em ciências: física básica – Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 1997.
16. CHAPRA, S. C.; CANALE, R. P. **Numerical Methods for Engineers**. New York; McGraw-Hill, 1998.

17. PRESTES, R. A. Avaliação das alterações bioquímicas diagnosticadas em plantas com morte súbita do citrus. São Carlos: IQSC-USP, 2008.

18. TECMAG. NTNMR Reference Manual: Manual. Texas: Tecmag USA. 389p.

ANEXOS

Anexo A – Script convert

```
'*****
' * VBScript-NMRScript Header, v2.3-030205
' *
' * use 'App.CommandName' to communicate
' * with the Application Object
' * use 'Data.CommandName' to communicate
' * with the topmost data file (Document Object)

Dim App, Data, pathToDoc
Set App = GetObject(, "NTNMR.Application")
pathToDoc = App.GetActiveDocPath

' If pathToDoc="" Then 'if no data file exists
'     MsgBox "Need a proper data file to be opened!", vbOKOnly, "NMRScript"
'     WScript.Quit() 'quit running the scripting program
' End If

' Set Data = GetObject(pathToDoc)
' *****
strMsg1 = "This script converts all .tnt files in a user specified folder" & vbCRLF
strMsg2 = "to .txt ascii text files in the same folder. Note: the user" & vbCRLF
strMsg3 = "will be required to manually specify whether to export the" & vbCRLF
strMsg4 = "active record or all records of a multi-dimensional data set."
' *****

strTitle = "Batch TNT to TXT Conversion"
msgbox strMsg1 & strMsg2 & strMsg3 & strMsg4,,strTitle

DocFolder = left(pathToDoc,instrrev(pathToDoc,"\"))
Folderpath = Inputbox("Please input the folder path: ",strTitle,DocFolder)
If Folderpath = "" then Wscript.Quit()

Set objShell = CreateObject("Shell.Application")
Set objFolder = objShell.Namespace (Folderpath)

For Each objFile in objFolder.Items
    If instr(objFile.Path,"tnt")>0 then
        App.OpenFile objFile.Path
        pathToDoc = App.GetActiveDocPath
        strNewPath=left(objFile.Path,instrrev(objFile.Path,".")-1) & ".txt"
        Set Data = GetObject(pathToDoc)
        Data.Export strNewPath, 0
        App.CloseFile(pathToDoc)
    end if
Next

msgbox "Conversion Completed",,,strTitle
```

Anexo B – CPMG fitting v.1

```
/*
Name: CPMG Fitting v.1
Copyright: EMBRAPA CNPDIA - Instrumentação Agropecuária
Author: ALLAN ALMEIDA DE LIMA
Date: 2009 19:08
Description: ESTE SOFTWARE FAZ O AJUSTE EXPONENCIAL DE DADOS ORIUNDOS DA TÉCNICA
DE RMN-CPMG.
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```

#include <gmatrix.c>
#include <math.h>

int ppcl (char arquivo[], long int lin, float dados[]);
int media (float dados[], int lin, int col, float tempo,
           PGMATRIX col_media, int n_lin, int cont5);
int pri_ord (PGMATRIX col_media, int n_lin, FILE *F, char arquivo[]);
int sec_ord(PGMATRIX col_media, int n_lin, FILE *F, char arquivo[]);

int ppcl (char arquivo[], long int lin, float dados[]){
    int cont3, cont4, cont5, i;
    float *t, *dataA, *dataB, soma_ele, sinal;
    char let;
    FILE *f, *e;

    t = malloc((lin)*sizeof(float));
    if(!t){
        printf ("falha na alocao dos dados\n");
        system("PAUSE");
        exit(1);
    }
    dataA = malloc((lin)*sizeof(float));
    if(!dataA){
        printf ("falha na alocao dos dados\n");
        system("PAUSE");
        exit(1);
    }
    dataB = malloc((lin)*sizeof(float));
    if(!dataB){
        printf ("falha na alocao dos dados\n");
        system("PAUSE");
        exit(1);
    }

    f = fopen (arquivo, "r+");
    if ((f = fopen(arquivo,"r+"))==NULL){
        puts("O arquivo nao pode ser aberto.\n");
        system ("PAUSE");
        exit(1);
    }

    e = fopen ("lixo.txt", "w+");
    if ((e = fopen("lixo.txt","w+"))==NULL){
        puts("O arquivo temporario nao pode ser aberto.\n");
        system ("PAUSE");
        exit(1);
    }

    cont3 = 0;
    let = getc (f);
    while (let != '\n' || cont3 != 3){
        let = getc (f);
        if (let == '\n') cont3++;
    }
    while (!feof(f)){
        let = getc (f);
        putc (let, e);
    }
    fclose (f);
    f = fopen (arquivo, "w");
    fseek (e, 0, 0);
    while (!feof (e)){
        let = getc (e);
        putc (let, f);
    }
    fclose (e);
    fclose (f);

    f = fopen (arquivo, "r");
    for (i=0; i<lin; i++){
        fscanf (f, "%f", &dataA[i]);
        fscanf (f, "%f", &dataB[i]);
    }
}

```

```

        fscanf (f, "%f", &t[i]);
    }
    fclose (f);

    f = fopen (arquivo, "w");
    e = fopen ("lixo.txt", "r");
    fseek (e, 0, 0);
    fprintf(f,"LINHA1\nLINHA2\nLINHA3\n");
    while (!feof (e)){
        let = getc (e);
        putc (let, f);
    }
    fclose (e);
    fclose (f);

    PGMATRIX U;
    U = PGMATRIX_ALLOC(lin,2);
    PGMATRIX S;
    S = PGMATRIX_ALLOC(lin,2);
    PGMATRIX V;
    V = PGMATRIX_ALLOC(lin,2);
    PGMATRIX mat;
    mat = PGMATRIX_ALLOC(lin,2);
    PGMATRIX mataux;
    mataux = PGMATRIX_ALLOC(lin,2);
    PGMATRIX mataux2;
    mataux2 = PGMATRIX_ALLOC(lin,1);
    PGMATRIX dado;
    dado = PGMATRIX_ALLOC(lin,1);

    cont4 = 0;
    for (i=1; i<=lin+1; i++){
        PGMATRIX_DATA(mat,i,1) = dataA[cont4];
        cont4++;
    }
    cont4 = 0;
    for (i=1; i<=lin+1; i++){
        PGMATRIX_DATA(mat,i,2) = dataB[cont4];
        cont4++;
    }

    PGMATRIX_SVD(U,S,V,mat,TRUE);
    PGMATRIX_MULTIPLY_COPY(mataux,U,S);
    PGMATRIX_COPY_COLUMN(mataux2,1,mataux,1);
    soma_ele = PGMATRIX_SUMENTRIES(mataux2);
    sinal = GMATRIXMACRO_SIGN(soma_ele);
    PGMATRIX_MULTIPLY_CONST_COPY(dado,mataux2,sinal);

    //elimina os valores de 0 do vetor de media e aloca-os em 'dados'
    cont5 = 0;
    for (i=1;i<=lin;i++){
        if (PGMATRIX_DATA(dado,i,1) != 0){
            dados[cont5] = PGMATRIX_DATA(dado,i,1);
            cont5++;
        }
    }

    PGMATRIX_FREE(U);
    PGMATRIX_FREE(S);
    PGMATRIX_FREE(V);
    PGMATRIX_FREE(mat);
    PGMATRIX_FREE(mataux);
    PGMATRIX_FREE(mataux2);
    PGMATRIX_FREE(dado);
    free(t);
    free(dataA);
    free(dataB);

    return cont5;
}

```

```

int media (float dados[], int lin, int col, float tempo, PGMATRIX col_media, int
n_lin, int cont5){

    int col_anl, i, j, k;
    float ult_media, norma;
    FILE *T;
    col_anl = col-((lin-cont5)/(lin/col));

    PGMATRIX aquisicao;
    aquisicao = PGMATRIX_ALLOC(n_lin,col_anl);

    if (lin%col != 0){
        printf("O numero de pontos do arquivo ou de aquisicao esta incorreto.\n");
        system ("PAUSE");
        exit(1);
    }

    PGMATRIX_DATA(col_media, 1, 1) = 0;
    k = 0;
    for (i=1; i<=n_lin; i++){
        for (j=1; j<=col_anl; j++){
            PGMATRIX_DATA(aquisicao, i, j) = dados[k];
            k++;
        }
        for (j=2; j<=col_anl+1; j++){
            PGMATRIX_DATA(col_media, i, 1) = PGMATRIX_DATA(col_media, i, 1) +
PGMATRIX_DATA(aquisicao, i, j-1);
        }
        PGMATRIX_DATA(col_media, i, 1) = PGMATRIX_DATA(col_media, i, 1)/col_anl;
    }

    ult_media = 0;
    for (i=n_lin-20; i<=n_lin; i++){
        ult_media = ult_media + PGMATRIX_DATA(col_media, i, 1);
    }
    ult_media = ult_media/20;

    k = 2;
    norma = PGMATRIX_DATA(col_media, 2, 1);

    for(i=2; i<=n_lin; i++){

        PGMATRIX_DATA(col_media, i, 1) = (PGMATRIX_DATA(col_media, i, 1)-
ult_media)/(norma-ult_media);
        PGMATRIX_DATA(col_media, i, 2) = k*tempo;
        k++;
    }

    PGMATRIX_FREE (aquisicao);
}

int pri_ord (PGMATRIX col_media, int n_lin, FILE *F, char arquivo[]){
    int n_linZ, i, i_cm, cont_it, n_pares;
    float A1, t1, R2, R2num, R2den1, R2den2, erro_lim, erro, erro_total[30],
SX, SX2, SY, SXY, a0, a1;
    n_linZ = n_lin-1;
    PGMATRIX Z;
    Z = PGMATRIX_ALLOC(n_linZ,2);
    PGMATRIX Y;
    Y = PGMATRIX_ALLOC(n_linZ,1);
    PGMATRIX FX;
    FX = PGMATRIX_ALLOC(n_linZ,1);
    PGMATRIX ZT;
    ZT = PGMATRIX_ALLOC(2,n_linZ);
    PGMATRIX ZTZ;
    ZTZ = PGMATRIX_ALLOC(2,2);
    PGMATRIX ZTZI;
    ZTZI = PGMATRIX_ALLOC(2,2);
    PGMATRIX D;
    D = PGMATRIX_ALLOC(n_linZ,1);
    PGMATRIX DA;

```

```

DA = PGMATRIX_ALLOC(2,1);
PGMATRIX_ZTD;
ZTD = PGMATRIX_ALLOC(2,1);
PGMATRIX_mat_erro;
mat_erro = PGMATRIX_ALLOC(n_linZ,1);

n_pares = 0.07*n_lin;
SX = SX2 = SY = SXY = 0;
for (i=2;i<=n_pares;i++){
    SX = SX + PGMATRIX_DATA(col_media,i,2);
    SX2 = SX2 + PGMATRIX_DATA(col_media,i,2)*PGMATRIX_DATA(col_media,i,2);
    SY = SY + PGMATRIX_DATA(col_media,i,1);
    SXY = SXY + PGMATRIX_DATA(col_media,i,2)*PGMATRIX_DATA(col_media,i,1);
}
a0=(SY*SX2-SX*SXY)/(n_pares*SX2-SX*SX);
a1=(n_pares*SXY-SX*SY)/(n_pares*SX2-SX*SX);

A1=PGMATRIX_DATA(col_media,10,1);
t1=-a0/a1;

i_cm=2;
for (i=1;i<=n_linZ;i++){
    PGMATRIX_DATA(Y,i,1) = PGMATRIX_DATA(col_media,i_cm,1);
    i_cm++;
}

erro_lim=0.1;

loop2:
cont_it=0;
erro_total[cont_it]=10;
while(erro_total[cont_it]>erro_lim){
    i_cm=2;
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(Z,i,1) = exp(-PGMATRIX_DATA(col_media,i_cm,2)/t1);
        PGMATRIX_DATA(Z,i,2) = (PGMATRIX_DATA(col_media,i_cm,2)*A1*exp(-
PGMATRIX_DATA(col_media,i_cm,2)/t1))/(t1*t1);
        i_cm++;
    }

    i_cm=2;
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(FX,i,1) = A1*exp(-
PGMATRIX_DATA(col_media,i_cm,2)/t1);
        i_cm++;
    }

    PGMATRIX_SUBTRACT_COPY(D,Y,FX);
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(mat_erro,i,1) =
PGMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
    }
    erro=0;
    for (i=1;i<=n_linZ;i++){
        erro = erro+PGMATRIX_DATA(mat_erro,i,1);
    }
    erro_total[cont_it+1]=erro;

    if (cont_it>0 && erro_total[cont_it+1]>erro_total[cont_it]+2){
        printf ("\nERRO. Possivelmente nao houve uma boa aproximacao
inicial!\n");
        system("PAUSE");
        exit(1);
    }

    PGMATRIX_TRANSPOSE_COPY(ZT,Z);
    PGMATRIX_MULTIPLY_COPY(ZTZ,ZT,Z);
    PGMATRIX_INVERSE_COPY(ZTZI,ZTZ);
    PGMATRIX_MULTIPLY_COPY(ZTD,ZT,D);
    PGMATRIX_MULTIPLY_COPY(DA,ZTZI,ZTD);

```

```

A1=A1+PGMATRIX_DATA(DA,1,1);
t1=t1+PGMATRIX_DATA(DA,2,1);

if (A1<0 || t1<0){
    printf ("\nAjuste com problemas, verifique os dados!");
    system ("PAUSE");
    exit(1);
}

cont_it++;
if (cont_it==10){
    erro_lim=erro_lim+0.1;
    goto loop2;
}

i_cm=2;
for (i=1;i<=n_linZ;i++){
    PMATRIX_DATA(FX,i,1) = A1*exp(-PGMATRIX_DATA(col_media,i_cm,2)/t1);
    i_cm++;
}

PGMATRIX_SUBTRACT_COPY(D,Y,FX);
for (i=1;i<=n_linZ;i++){
    PMATRIX_DATA(mat_erro,i,1) =
PGMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
}

R2num = 0;
R2den1 = 0;
R2den2 = 0;
for (i=1;i<=n_linZ;i++){
    R2num = R2num + PMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
    R2den1 = R2den1 + PMATRIX_DATA(Y,i,1)*PGMATRIX_DATA(Y,i,1);
    R2den2 = R2den2 + PMATRIX_DATA(Y,i,1);
}
R2 = 1-((n_linZ*R2num)/(n_linZ*R2den1-R2den2*R2den2));

PGMATRIX_FREE (Z);
PGMATRIX_FREE (Y);
PGMATRIX_FREE (FX);
PGMATRIX_FREE (ZT);
PGMATRIX_FREE (ZTZ);
PGMATRIX_FREE (ZTZI);
PGMATRIX_FREE (D);
PGMATRIX_FREE (DA);
PGMATRIX_FREE (ZTD);
PGMATRIX_FREE (mat_erro);

fprintf (F, "\n%s - A1: %f    t1: %f    R2: %f\n",arquivo,A1,t1,R2);
}

int sec_ord(PGMATRIX col_media, int n_lin, FILE *F, char arquivo[]){
    int n_linZ, i, i_cm, cont_it, n_pares1, n_pares2, mono;
    float A1, t1, A2, t2, R2, R2num, R2den1, R2den2, erro, erro_lim, SX, SX2,
SY, SXY, a0, a1, div, retal, reta2,erro_total[30];

    div=0.15;
    retal=0.1;
    reta2=0.015;

    n_linZ = n_lin-1;
    mono=div*n_lin;

    PMATRIX Z;
    Z = PMATRIX_ALLOC(n_linZ,4);
    PMATRIX Y;
    Y = PMATRIX_ALLOC(n_linZ,1);
    PMATRIX FX;
    FX = PMATRIX_ALLOC(n_linZ,1);
    PMATRIX mat_erro;

```



```

mat_erro = PGMATRIX_ALLOC(n_linZ,1);
PGMATRIX ZT;
ZT = PGMATRIX_ALLOC(4,n_linZ);
PGMATRIX ZTZ;
ZTZ = PGMATRIX_ALLOC(4,4);
PGMATRIX ZTZI;
ZTZI = PGMATRIX_ALLOC(4,4);
PGMATRIX D;
D = PGMATRIX_ALLOC(n_linZ,1);
PGMATRIX DA;
DA = PGMATRIX_ALLOC(4,1);
PGMATRIX ZTD;
ZTD = PGMATRIX_ALLOC(4,1);
PGMATRIX col_mono;
col_mono = PGMATRIX_ALLOC(mono,1);

for (i=2;i<=mono;i++){
    PGMATRIX_DATA(col_mono,i,1)=PGMATRIX_DATA(col_media,i,1)-
PGMATRIX_DATA(col_media,mono,1);
}

n_pares1 = reta1*mono;
SX = SX2 = SY = SXY = 0;
for (i=2;i<=n_pares1;i++){
    SX = SX + PGMATRIX_DATA(col_media,i,2);
    SX2 = SX2 + PGMATRIX_DATA(col_media,i,2)*PGMATRIX_DATA(col_media,i,2);
    SY = SY + PGMATRIX_DATA(col_mono,i,1);
    SXY = SXY + PGMATRIX_DATA(col_media,i,2)*PGMATRIX_DATA(col_mono,i,1);
}

a0=(SY*SX2-SX*SXY)/((n_pares1-1)*SX2-SX*SX);
a1=((n_pares1-1)*SXY-SX*SY)/((n_pares1-1)*SX2-SX*SX);

A1=0;
for (i=5;i<=9;i++){
    A1= A1+PGMATRIX_DATA(col_mono,i,1);
}
A1=A1/5;
t1=-a0/a1;

n_pares2 = reta2*(n_lin-mono);
SX = SX2 = SY = SXY = 0;
for (i=mono;i<=mono+n_pares2;i++){
    SX = SX + PGMATRIX_DATA(col_media,i-mono+1,2);
    SX2 = SX2 + PGMATRIX_DATA(col_media,i-
mono+1,2)*PGMATRIX_DATA(col_media,i-mono+1,2);
    SY = SY + PGMATRIX_DATA(col_media,i,1);
    SXY = SXY + PGMATRIX_DATA(col_media,i-
mono+1,2)*PGMATRIX_DATA(col_media,i,1);
}
a0=(SY*SX2-SX*SXY)/((n_pares2+1)*SX2-SX*SX);
a1=((n_pares2+1)*SXY-SX*SY)/((n_pares2+1)*SX2-SX*SX);

A2=0;
for (i=mono;i<mono+5;i++){
    A2=A2+PGMATRIX_DATA(col_media,i,1);
}
A2=A2/5;;
t2=-a0/a1;

i_cm=2;
for (i=1;i<=n_linZ;i++){
    PGMATRIX_DATA(Y,i,1) = PGMATRIX_DATA(col_media,i_cm,1);
    i_cm++;
}

erro_lim=0.00005;

loop1:
cont_it=0;
erro_total[cont_it]=10;

```

```

while(erro_total[cont_it]>erro_lim){

    i_cm=2;
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(Z,i,1) = exp(-PGMATRIX_DATA(col_media,i_cm,2)/t1);
        PGMATRIX_DATA(Z,i,2) = exp(-PGMATRIX_DATA(col_media,i_cm,2)/t2);
        PGMATRIX_DATA(Z,i,3) = (PGMATRIX_DATA(col_media,i_cm,2)*A1*exp(-
PGMATRIX_DATA(col_media,i_cm,2)/t1))/(t1*t1);
        PGMATRIX_DATA(Z,i,4) = (PGMATRIX_DATA(col_media,i_cm,2)*A2*exp(-
PGMATRIX_DATA(col_media,i_cm,2)/t2))/(t2*t2);
        i_cm++;
    }

    i_cm=2;
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(FX,i,1) = A1*exp(-PGMATRIX_DATA(col_media,i_cm,2)/t1)
+ A2*exp(-PGMATRIX_DATA(col_media,i_cm,2)/t2);
        i_cm++;
    }

    PGMATRIX_SUBTRACT_COPY(D,Y,FX);
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(mat_erro,i,1) =
PGMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
    }

    erro=0;
    for (i=1;i<=n_linZ;i++){
        erro = erro+PGMATRIX_DATA(mat_erro,i,1);
    }
    erro_total[cont_it+1]=erro;

    if (cont_it>0 && erro_total[cont_it+1]>erro_total[cont_it]+10){
        printf ("\nERRO. Possivelmente nao houve uma boa aproximacao
inicial!\n");
        system("PAUSE");
        exit(1);
    }

    PGMATRIX_TRANSPOSE_COPY(ZT,Z);
    PGMATRIX_MULTIPLY_COPY(ZTZ,ZT,Z);
    PGMATRIX_INVERSE_COPY(ZTZI,ZTZ);
    PGMATRIX_MULTIPLY_COPY(ZTD,ZT,D);
    PGMATRIX_MULTIPLY_COPY(DA,ZTZI,ZTD);

    A1=A1+PGMATRIX_DATA(DA,1,1);
    A2=A2+PGMATRIX_DATA(DA,2,1);
    t1=t1+PGMATRIX_DATA(DA,3,1);
    t2=t2+PGMATRIX_DATA(DA,4,1);

    if (A1<0 || t1<0 || A2<0 || t2<0){

        printf ("\nAjuste com problemas, verifique os dados!");
        system ("PAUSE");
        exit(1);
    }

    cont_it++;
    if (cont_it==10){
        erro_lim=erro_lim + 0.01;
        goto loop1;
    }
}

i_cm=2;
for (i=1;i<=n_linZ;i++){
    PGMATRIX_DATA(FX,i,1) = A1*exp(-PGMATRIX_DATA(col_media,i_cm,2)/t1) +
A2*exp(-PGMATRIX_DATA(col_media,i_cm,2)/t2);
    i_cm++;
}

```

```

    }

    PGMATRIX_SUBTRACT_COPY(D,Y,FX);
    for (i=1;i<=n_linZ;i++){
        PGMATRIX_DATA(mat_erro,i,1) =
PGMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
    }

    R2num = 0;
    R2den1 = 0;
    R2den2 = 0;
    for (i=1;i<=n_linZ;i++){
        R2num = R2num + PGMATRIX_DATA(D,i,1)*PGMATRIX_DATA(D,i,1);
        R2den1 = R2den1 + PGMATRIX_DATA(Y,i,1)*PGMATRIX_DATA(Y,i,1);
        R2den2 = R2den2 + PGMATRIX_DATA(Y,i,1);
    }
    R2=0;
    R2 = 1-((n_linZ*R2num)/(n_linZ*R2den1-R2den2*R2den2));

    PGMATRIX_FREE (Z);
    PGMATRIX_FREE (Y);
    PGMATRIX_FREE (FX);
    PGMATRIX_FREE (ZT);
    PGMATRIX_FREE (ZTZ);
    PGMATRIX_FREE (ZTZI);
    PGMATRIX_FREE (D);
    PGMATRIX_FREE (DA);
    PGMATRIX_FREE (ZTD);
    PGMATRIX_FREE (mat_erro);
    PGMATRIX_FREE (col_mono);

    if (R2<0.99) goto loop1;

    if (t1<t2) fprintf (F, "\n%s - A1: %f    t1: %f    A2: %f    t2: %f    R2:
%f\n",arquivo,A2,t2,A1,t1,R2);
    else fprintf (F, "\n%s - A1: %f    t1: %f    A2: %f    t2: %f    R2:
%f\n",arquivo,A1,t1,A2,t2,R2);
}

main(){
    int lin, col, n_lin, n, cont0, cont, cont2, ordem, cont5;
    float tempo;
    char c, nome[30], S[30];
    FILE *F, *G, *H;
    system("convert");
    printf ("\n\n##### ANALISE DOS DADOS DE RMN #####");
    printf ("\n\nIMPORTANTE:\n Os nomes dos arquivos a serem analisados devem
ser os mesmos,\nmudando-se apenas o numero final que deve estar em ordem!\n");
    printf("\n\nInforme o nome dos arquivos sem o numero: ");
    scanf("%s",&S);
    printf ("\nInforme o numero de arquivos existentes no diretorio: ");
    scanf ("%d",&n);
    printf ("\nEntre com o numero de pontos dos arquivos: ");
    scanf ("%d",&lin);
    printf ("\nEntre com o numero de pontos de aquisicao: ");
    scanf ("%d",&col);
    printf ("\nEntre com o tempo de analise: ");
    scanf ("%f",&tempo);
    printf ("\nDeseja ajustar os dados a um decaimento exponencial de 1a (1) ou
2a (2) ordem? : ");
    scanf ("%d",&ordem);

    float dados[lin];
    n_lin = lin/col;
    PGMATRIX col_media;
    col_media = PGMATRIX_ALLOC(n_lin,2);

    if (ordem==1){
        F = fopen ("coeficientes_MONO.txt", "w+");
        if ((F = fopen("coeficientes_mono.txt","w+"))==NULL){
            puts("O arquivo
'coeficientes_mono.txt' nao pode ser aberto.\n");

```

```

system ("PAUSE");
exit(1);
}
fprintf (F, "Os dados estao ajustados na forma:\n
f(t)=A1.exp(-t/t1)\n");
}
if(ordem==2){
F = fopen ("coeficientes_BI.txt", "w+");
if ((F = fopen("coeficientes_bi.txt","w+"))==NULL){
puts("O arquivo
'coeficientes_bi.txt' nao pode ser aberto.\n");
system ("PAUSE");
exit(1);
}
fprintf (F, "Os dados estao ajustados na forma:\n      f(t)=A1.exp(-
t/t1)+A2.exp(-t/t2)\n");
}

G = fopen ("nomes.txt","w");
if (!G) printf("Erro.\n");
else{
for (cont0=1; cont0<=n; cont0++){
fprintf(G,"%s%d.txt",S,cont0);
fprintf(G,"\n");
}
}
fclose (G);

G = fopen ("nomes.txt","r");
c = getc (G);
while (!feof (G)){
for (cont2 = 0; cont2 <= n-1; cont2++){
cont = 0;
while (c != '\n'){
nome[cont] = c;
cont++;
c = getc (G);
}
nome[cont] = '\0';
c = getc (G);

cont5 = ppcl (nome, lin, dados);

media (dados, lin, col, tempo, col_media, n_lin, cont5);
if (ordem==1) pri_ord(col_media, n_lin, F, nome);
else sec_ord(col_media, n_lin, F, nome);
printf("\nArquivo '%s' processado com sucesso.\n",nome);
}
}
PGMATRIX_FREE (col_media);
fclose (F);
fclose (G);
remove ("lixo.txt");
remove ("nomes.txt");
system ("PAUSE");
}

```

Anexo C – CPMG fitting v.2

```

/*
Name: CPMG Fitting v.2
Copyright: EMBRAPA CNPDIA - Instrumentação Agropecuária
Author: ALLAN ALMEIDA DE LIMA
Date: 2009 19:08
Description: ESTE SOFTWARE FAZ O AJUSTE EXPONENCIAL DE DADOS ORIUNDOS DA TÉCNICA
DE RMN-CPMG.
*/

main(){
int lin, col, n_lin, n, cont0, cont, cont2, ordem, cont5;

```

```

float tempo;
char c, ch, nome[30], S[50], S1[30], S2[30], S3[30], S4[30], S5[30];
FILE *F, *G, *H, *A;

system("convert");

A = fopen ("DATA.txt", "r");
ch = getc (A);
cont = 0;
//pega nome do arquivo
while (ch!='\n'){
    S[cont] = ch;
    cont++;
    ch = getc (A);
}
S[cont] = '\0';
ch = getc (A);
cont = 0;
//pega numero de pontos
while (ch!='\n'){
    S1[cont] = ch;
    cont++;
    ch = getc (A);
}
S1[cont] = '\0';
lin = atol(S1);
ch = getc (A);
cont = 0;
//pega pontos de aquisição
while (ch!='\n'){
    S2[cont] = ch;
    cont++;
    ch = getc (A);
}
S2[cont] = '\0';
col = atol(S2);
ch = getc (A);
cont = 0;
//pega o tempo de aquisição
while (ch!='\n'){
    S4[cont] = ch;
    cont++;
    ch = getc (A);
}
S4[cont] = '\0';
tempo = atof(S4);
ch = getc (A);
cont = 0;
//pega a ordem de decaimento
while (ch!='\n'){
    S5[cont] = ch;
    cont++;
    ch = getc (A);
}
S5[cont] = '\0';
ordem = atoi(S5);
fclose (A);
remove ("DATA.txt");

float dados[lin];
n_lin = lin/col;
PGMATRIX col_media;
col_media = PGMATRIX_ALLOC(n_lin,2);

if (ordem==1){
    F = fopen ("coeficientes_mono.txt", "w+");
    if ((F = fopen("coeficientes_mono.txt", "w+"))==NULL){
        puts("O arquivo
'coeficientes_mono.txt' nao pode ser aberto.\n");
        system ("PAUSE");
        exit(1);
    }
}

```

```

        fprintf (F, "Os dados estao ajustados na forma:\n
f(t)=A1.exp(-t/t1)\n");
    }
    if(ordem==2){
        F = fopen ("coeficientes_bi.txt", "w+");
        if ((F = fopen("coeficientes_bi.txt","w+"))==NULL){
            puts("O arquivo
'coeficientes_bi.txt' nao pode ser aberto.\n");
            system ("PAUSE");
            exit(1);
        }
        fprintf (F, "Os dados estao ajustados na forma:\n      f(t)=A1.exp(-
t/t1)+A2.exp(-t/t2)\n");
    }

    G = fopen ("nomes.txt","w");
    if (!G) printf("Erro.\n");
    else{
        for (cont0=1; cont0<=n; cont0++){
            fprintf(G,"%s%d.txt",S,cont0);
            fprintf(G,"\n");
        }
    }
    fclose (G);

    G = fopen ("nomes.txt","r");
    c = getc (G);
    while (!feof (G)){
        for (cont2 = 0; cont2 <= n-1; cont2++){
            cont = 0;
            while (c != '\n'){
                nome[cont] = c;
                cont++;
                c = getc (G);
            }
            nome[cont] = '\0';
            c = getc (G);

            cont5 = ppc1 (nome, lin, dados);

            media (dados, lin, col, tempo, col_media, n_lin, cont5);
            if (ordem==1) pri_ord(col_media, n_lin, F, nome);
            else sec_ord(col_media, n_lin, F, nome);
        }
    }
    PGMATRIX_FREE (col_media);
    fclose (F);
    fclose (G);
    remove ("lixo.txt");
    remove ("nomes.txt");
    system ("PAUSE");
}

```

Anexo D – Script CPMG

```

'*****
' * VBScript-NMRScript Header, v2.3-030205
' *
' * use 'App.CommandName' to communicate
' * with the Application Object
' * use 'Data.CommandName' to communicate
' * with the topmost data file (Document Object)

Dim App, Data, pathToDoc
Set App = GetObject(, "NTNMR.Application")
pathToDoc = App.GetActiveDocPath

If pathToDoc="" Then 'if no data file exists
    MsgBox "Need a proper data file to be opened!", vbOKOnly, "NMRScript"
    WScript.Quit() 'quit running the scripting program

```

```

End If

Set Data = GetObject(pathToDoc)
' *****

msgbox "You have to locate the sample and press Ok!",1,"Ntnmr"

if num1=2 then
WScript.Quit()
end if

'RETIRA O NUMERO DE EXPERIMENTOS A SEREM REALIZADOS
n_exp = Data.GetNMRParameter ("Points 2D")

'RETIRA O TEMPO DE AQUISIÇÃO
tempo_aqui = Data.GetNMRParameter ("Analysis Time")

'RETIRA A ORDEM DO AJUSTE
ordem = Data.GetNMRParameter ("Decay Order")

'RETIRA O NUMERO DE PONTOS DOS ARQUIVOS
n_pto = Data.Get1DSize

'RETIRA O NUMERO DE PONTOS DE AQUISIÇÃO DOS ARQUIVOS
pto_acq = Data.GetNMRParameter ("Acq. Points")

MyVariable = App.GetActiveDocPath
posicao = InStrRev(MyVariable, "\")
comprimento = Len(MyVariable)
tamanho = comprimento - posicao - 4
nom_arquivo = Mid(MyVariable, posicao + 1, tamanho)

i=0
do
i=i+1

Data.ZG
Do While Not Data.CheckAcquisition
Loop

'ESSE TRECHO SALVA O TNT COMO
j = cstr(i)
App.SaveAs nom_arquivo+"_"+j

'ESSE TREHO EXPORTA
MyVariable2 = App.GetActiveDocPath
posicao2 = InStrRev(MyVariable2, "\")
comprimento2 = Len(MyVariable2)
tamanho2 = comprimento2 - posicao2 - 4
nom_arquivo2 = Mid(MyVariable2, posicao2 + 1, tamanho2)
nom_arquivo3 = nom_arquivo2 + ".txt"
App.Export nom_arquivo3, 0

'ESSE TRECHO CRIA E ESCRIVE NUM ARQUIVO TXT OS DADOS PARA O PROCESSAMENTO DO
PROGRAMA
ForAppending = 8
ForReading = 1
ForWriting = 2
Dim objFSO, logFile
Set objFSO = CreateObject("Scripting.FileSystemObject")
logfilepath = "C:\DATA.txt"
Set logFile = objFSO.OpenTextFile(logfilepath, ForAppending, True)
logFile.WriteLine(nom_arquivo3)
logFile.WriteLine(n_pto)
logFile.WriteLine(pto_acq)
logFile.WriteLine(tempo_aqui)
logFile.WriteLine(ordem)
logFile.WriteLine(" ")
logFile.Close

'ESSE TRECHO PROCESSA OS DADOS
Set WSHShell = CreateObject("WScript.Shell")

```

```
WSHShell.Run "C:\CPMG_fitting_v.2.exe",0,false

If n_exp = i Then
Exit Do
End If

Loop

msgbox "Completed samples!",0,"Ntnmr"
```