

Apresentação do R com um exemplo de análise de regressão não-linear*

Eduardo Esteves

26 de Março de 2009

Instituto Superior de Engenharia, Universidade do Algarve, Campus da Penha, 8005-139 Faro e Centro de Ciências do Mar, Laboratório Associado, Campus de Gambelas, 8005-139 Faro; E-mail: eesteves@ualg.pt URL <http://w3.ualg.pt/~eesteves>

Resumo

No seguimento dum artigo anterior acerca da análise de regressão não-linear (simples) utilizando a ferramenta Solver[®] do Excel[®] (Esteves, 2008) propõe-se neste artigo a apresentação do R, uma linguagem de programação e um ambiente para computação estatística e gráfica, através da sua aplicação na “descrição” de relações estatísticas (não-lineares) entre variáveis.

Palavras-chave: R, Regressão não-linear.

Introdução

O R (R Development Core Team, 2007) é ao mesmo tempo uma linguagem de programação e um ambiente para computação estatística e gráfica. Trata-se de uma linguagem de programação especializada em computação com dados. Uma das suas principais características é o seu carácter gratuito e a sua disponibilidade para uma gama bastante variada de sistemas operativos (vd. <http://www.r-project.org>). Apesar do seu carácter gratuito, o R é uma ferramenta bastante poderosa com boas capacidades ao nível da programação e um conjunto bastante vasto (e em constante crescimento) de packages que acrescentam bastantes potencialidades à já poderosa versão base do R (Torgo, 2006). Contudo, o termo “ambiente” pretende caracterizar o R como um sistema completo e coerente ao invés dum conjunto de ferramentas muito específicas e relativamente inflexíveis (Venables et al., 2006).

O R pode ser entendido como uma implementação da linguagem S desenvolvida por Rick Becker, John Chambers e Allan Wilks nos Bell Laboratories (E.U.A.), que também constitui a base do software S-Plus[®] (Insightful Corp.). A evolução da linguagem S está descrita em quatro livros de John Chambers e colaboradores. As “distribuições” do R incluem um bom conjunto de manuais (vd. <http://cran.r-project.org/manuals.html>) e existem, actualmente, vários livros que descrevem a utilização do R para análise estatística de dados (vd. <http://www.r-project.org/doc/bib/R-books.html>), e.g. Dalgaard (2002). Foi disponibilizada recentemente (em

*Citação recomendada: Esteves, E. (2009) Apresentação do R com um exemplo de análise de regressão não-linear. Instituto Superior de Engenharia da Universidade do Algarve, Faro, 10 p. [disponível em <http://w3.ualg.pt/~eesteves>]

22/4/2008)¹ a versão 2.7.0 do R (embora neste artigo se utilize a versão 2.5.0). O ficheiro de instalação (R-2.7.0-win32.exe) para Microsoft Windows® pode ser obtido em <http://cran.at.r-project.org/bin/windows/base/R-2.7.0-win32.exe>. A instalação é simples. Também existem “versões” para Mac OS e Unix/Linux. O sítio electrónico do projecto constitui a principal referência do R e funciona como ponto de partida para explorar mais este sistema.

Boa parte das pessoas utiliza o R como um sistema para análise estatística de dados, uma vez que a maioria das “estatísticas clássicas” e muitas das metodologias mais recentes estão disponíveis, embora os promotores prefiram “pensar” o R como um ambiente no qual essas técnicas têm sido implementadas (Venables et al., 2006). Cerca de 25 pacotes (i.e. conjuntos de funções) fazem parte do sistema básico (os “recommended” packages), mas muitos outros estão disponíveis através do CRAN (via <http://CRAN.R-project.org>) para instalação.

Uma importante diferença entre a linguagem S (e o R) e outros sistemas (e.g. SPSS® ou SAS®) reside na (muito) menor quantidade de resultados apresentados para uma qualquer análise estatística, embora os resultados sejam guardados em objectos para posterior consulta ou utilização pelo R noutras funções (Venables et al., 2006).

Neste artigo, pretende-se apresentar (muito sucintamente) o R, uma linguagem de programação especializada em computação com dados, utilizando-a para analisar problemas cujo objectivo é “descrever” relações estatísticas (não-lineares) entre variáveis. Não se pretende discutir aqui os aspectos estatísticos dos resultados obtidos, apenas o *modus operandi* (ainda que de forma simplificada).

Regressão não-linear (simples)

Os vários aspectos relacionados com a regressão não-linear (simples) que servem de base a este artigo foram abordados, ainda que de forma informal e possivelmente incompleta, num artigo anterior (Esteves, 2008).

Em vários domínios do conhecimento, e.g. biologia, física, química, engenharia, etc., são usados modelos matemáticos para descrever um conjunto de dados empíricos, genericamente

$$y = f(x) + \epsilon$$

em que y é a variável dependente, x é a variável “independente” – por vezes, controlada pelo investigador – e $f(x)$ é uma função que pode incluir um ou mais parâmetros θ e ϵ são os erros aleatórios, independentes e com distribuição normal. Outra formulação, equivalente, é $\hat{y} = f(x)$ (em que \hat{y} se lê valor esperado, ou estimado, de y). Quanto melhor $f(x)$ se ajustar aos dados, mais “rigorosamente” descreverá aquela relação (Brown, 2001). Pretende-se ajustar a função $f(x)$ aos dados empíricos de forma a minimizar os erros $\epsilon_i = (y_i - \hat{y})$. De facto, o objectivo é estimar o(s) parâmetro(s) da função $f(x)$ de modo a minimizar a soma dos quadrados dos erros (ou resíduos), SQE – método dos mínimos quadrados. No caso de funções (ou modelos) não-lineares, e.g. , não é possível obter as estimativas dos parâmetros num único passo, como no caso de regressões lineares, pelo que a SQE é minimizada através dum processo iterativo (cíclico) utilizando um algoritmo apropriado que necessita dos valores iniciais dos parâmetros θ_0 (Bowen and Jerman, 1995). Tradicionalmente, transformam-se as variáveis de alguns modelos não-lineares de forma a linearizar a relação e a permitir a sua análise através da regressão linear. Contudo, esta abordagem

¹Entretanto, será disponibilizada em breve a versão 2.9.0 (R-2.9.0-win32.exe) no sítio electrónico do programa.

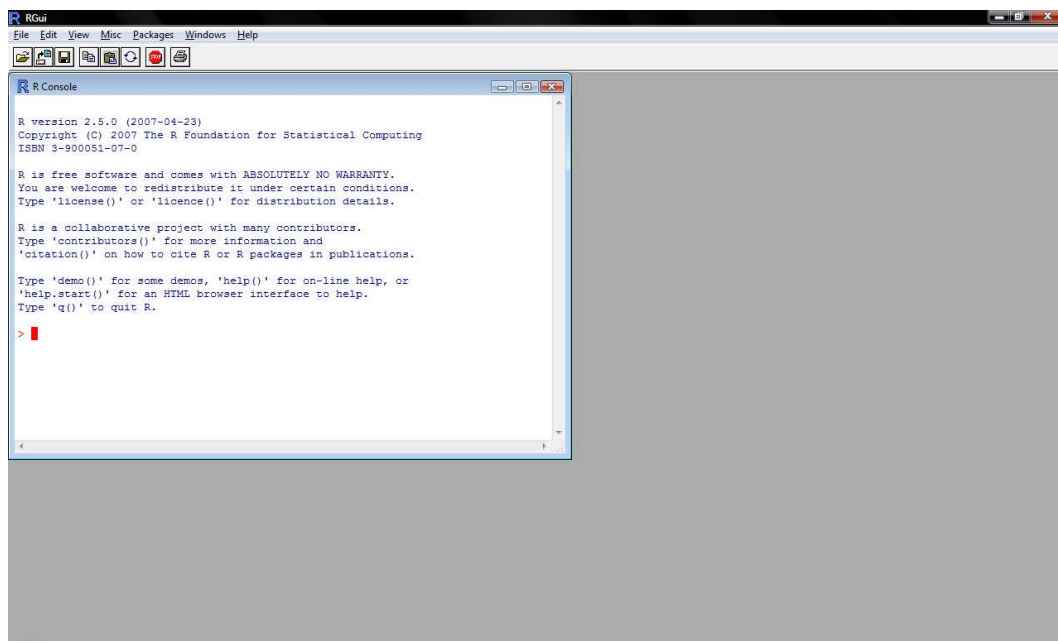


Figura 1: Aspecto da janela *R Console*.

é válida se a(s) variável(is) transformadas se verificam os pressupostos da análise de regressão linear.

Exemplo

A utilização do R com um exemplo concreto (entretanto abordado por Esteves, 2008, usando a ferramenta Solver[®] do Microsoft Excel[®]) permitirá mostrar o funcionamento e demonstrar as capacidades do software. Como se imagina, as funcionalidades não se esgotam no que aqui se apresenta.

O aspecto do R ao iniciar uma sessão de trabalho em ambiente Windows[®] ilustra-se na 1. Para além da barra de ferramentas no topo (que permite realizar as tarefas comuns: abrir/gravar ficheiros, cortar/colar texto, instalar pacotes, “gerir” as janelas, etc.) surge uma janela (R Console) na qual se introduzem os comandos, a seguir ao sinal `>`.

Admita-se que os dados que se pretendem analisar estão num ficheiro do Excel[®]. Em primeiro lugar, será necessário guardar uma versão (*.txt ou *.csv)² desse ficheiro utilizável pelo R (devem usar-se pontos, em vez de vírgulas, como separadores decimais).

Na consola do R escrever (fazendo **Enter** no fim),

```
> dados<-read.table(file="G:/MyArticles/ArtigosRegressaoNaoLinearExcel/RegressaoNaoLinear.txt",header=T,sep="\t")
```

para “carregar” a informação contida no ficheiro `RegressaoNaoLinear.txt` para o objecto `dados`. (atente-se na notação `<-`). Os parâmetros `header` e `sep` permitem considerar os rótulos dos dados e o tipo de ficheiro de texto (neste caso separado por tabulações), respectivamente. Por sinal, a quantidade de dados envolvida poderia muito bem ser introduzida directamente no R através de

² A extensão `txt` diz respeito a ficheiros de texto separado por tabulações enquanto a extensão `csv` está relacionada com ficheiros de texto separados por vírgulas.

```
> C02<-c(141,172,181,227,309,414,641, 936)
> Prop<-c(0.25,0.34,0.34,0.68,0.85,0.99,0.98,0.99)
```

em que `c(...)` permite concatenar um conjunto de dados num objecto (neste caso, `C02` e `Prop`). Podemos visualizar a informação usando a função `plot`

```
> plot(Prop~C02, las=1,xlim=c(0,1000),ylim=c(0,1),pch=16,cex=
1.5, xlab="C02 (ppm)", ylab="Proporção de respostas correctas")
```

em que `Prop~C02` descreve o “modelo” que se pretende representar (neste caso `Prop` em função de `C02`) e cujos restantes parâmetros especificam as várias características do gráfico (`las` para orientação dos rótulos do eixo dos `yy`, `xlim` e `ylim` para determinar os limites dos eixos, `pch` para definir o símbolo usado, `cex` para aumentar o tamanho relativo dos caracteres, e `xlab` e `ylab` para “legendar” os eixos) (Figura 2a). Para mais informações acerca desta (ou qualquer outra) função pode fazer-se

```
> ?plot
```

(ou então procurar com `>help.search(“termo”)` ou, ainda, usar a opção `Help>Apropos...`). Usando o botão direito do rato sobre o gráfico, é possível copiar a figura (como metaficheiro Windows® por exemplo) para inclusão num ficheiro do Microsoft Word®.

Admita-se (por razões científicas) que um modelo “dose-resposta” (logístico modificado)

$$P = \frac{\frac{1}{3} + \exp(B(T - \log(x)))}{1 + \exp(T - \log(x))}$$

em que P_i é a proporção de respostas positivas do provador i , $\log(x)$ é o logaritmo do valor do estímulo x , B é o “declive” e T é o limiar para o provador i (em $\log(x)$) é adequado para descrever os dados. Para ajustar o modelo (equação anterior) basta fazer

```
> ajuste<-nls(Prop~(1/3+exp(B*(T-log10(C02))))/(1+exp(B*(T-log10(C02)))),
data=data.frame(C02=C02,Prop=Prop),start=list(B=-10,T=2.5))
```

em que o objecto `ajuste` incluirá os resultados duma regressão não-linear (através da função `nls`)³ do modelo⁴ `Prop~(1/3+exp(B*(T-log10(C02))))/(1+exp(B*(T-log10(C02)))` considerando os dados (indicados para o parâmetro `data` sob a forma de uma `data.frame`)⁵ e as estimativas iniciais dos parâmetros do modelo descritas em `start` (através duma `list`). Os resultados do ajuste (Figura 3) podem ser consultados fazendo

```
> summary(ajuste)
```

É possível ilustrar o `ajuste`, obtendo primeiro as estimativas da variável `Prop` (através da função `predict`) e “desenhando-as”, em seguida, sobre o gráfico entretanto obtido (Figura 2a vs Figura 2b) usando a função `lines`, ou seja

³Outras funções (muito) usadas são `lm` e `aov` para regressão linear (i.e. modelos lineares) e análise de variância, respectivamente.

⁴A definição dos modelos em S está minuciosamente descrita em Chambers and Hastie (1992).

⁵Neste caso, considerou-se que o utilizador introduziu directamente no sistema os dados ao invés de “carregar” o ficheiro dos dados (neste último caso, bastaria apenas indicar `data=dados`, no comando).

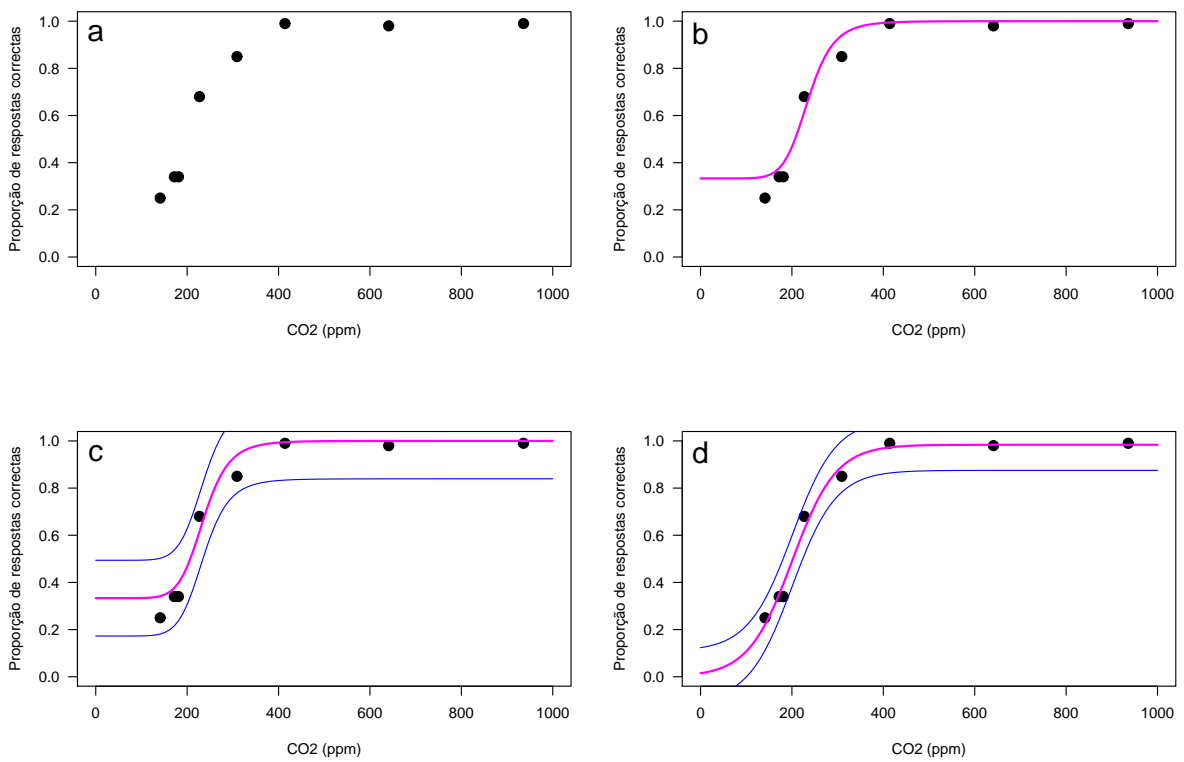


Figura 2: Diagramas de dispersão da proporção de respostas vs. concentração de CO₂: dados originais (a); modelo “dose-resposta” (logístico modificado); modelo “dose-resposta” com intervalos de 95% de confiança; e modelo logístico (ver text principal).

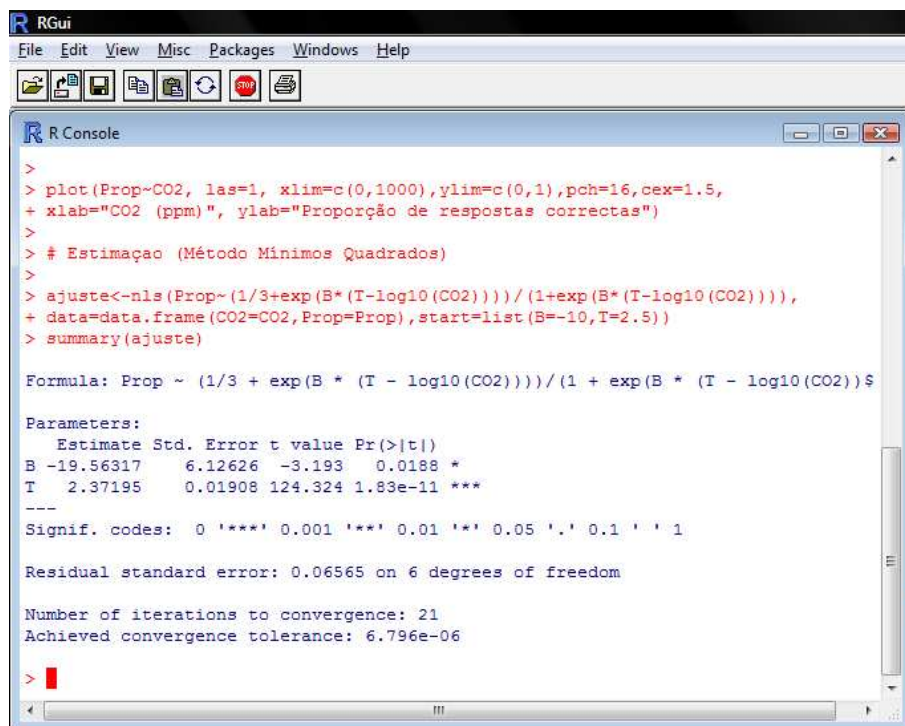


Figura 3: Aspecto da execução de comandos e respectivos resultados na janela *R Console*.

```

> PropEsp<-predict(ajuste,newdata=list(CO2=seq(0,1000,length=100)),se.fit=T)
> lines(PropEsp~seq(0,1000,length=100),lwd=2,col=6)

```

A informação contida no objecto `ajuste` permite obter as estimativas para 100 valores de CO2 entre 0 e 1000 (daí o parâmetro `seq(1,1000,length=100)` no comando acima) assim como os respectivos erros-padrão (`se.fit=T`) que serão guardadas no objecto designado `PropEsp`. A adição duma linha espessa e magenta (`lwd=2` e `col=6`) à Figura 2a é relativamente fácil (Figura 2b).

Podem, ainda, obter-se e “desenhar-se” os intervalos de confiança (`upIC` e `loIC`) que se calculam através de `qt`, recorrendo às funções `qt` (para obter os valores “teóricos” de t-Student e cujos parâmetros são $1 - \alpha/2$, neste caso 0.975, e os graus de liberdade do erro, que se obtêm de `summary(ajuste)$df[2]`), `summary(ajuste)$sigma` para obter o erro-padrão das estimativas e `lines`, respectivamente (Figura 2c).

```

> upIC<-PropEsp+qt(.975,summary(ajuste)$df[2])*summary(ajuste)$sigma
> loIC<-PropEsp-qt(.975,summary(ajuste)$df[2])*summary(ajuste)$sigma
> lines(upIC~seq(0,1000,length=100),col=4)
> lines(loIC~seq(0,1000,length=100),col=4)

```

O modo mais simples de “testar” a bondade do ajuste é representar graficamente os dados e o modelo ajustado (bem como os respectivos intervalos de confiança), de modo a verificar (visualmente) se os parâmetros obtidos numericamente descrevem, de facto, a relação entre variáveis (Figura 2c). Por outro lado, a análise gráfica dos resíduos, e.g. resíduos *vs.* valores observados de X , permite “verificar” se o modelo é adequado (os resíduos representam apenas o erro experimental se não apresentam tendências ou padrões). Sendo assim, obter os resíduos através de

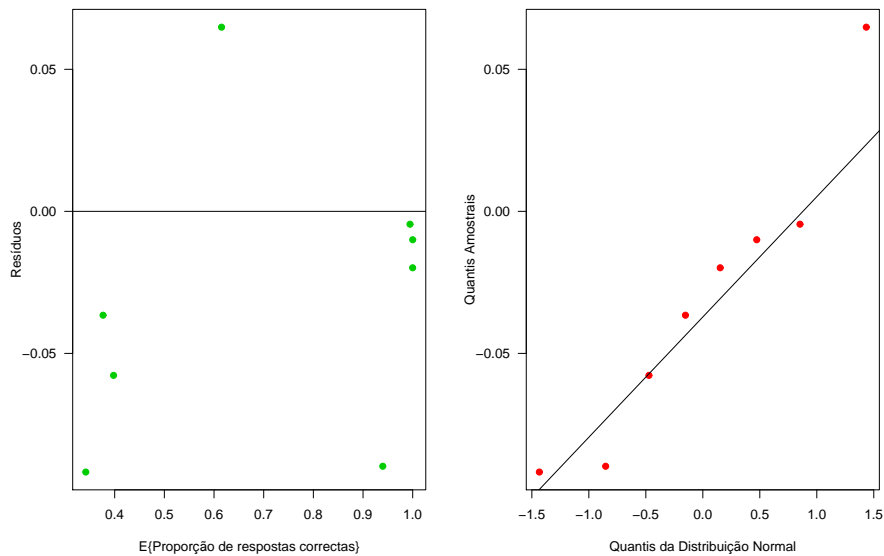


Figura 4: Análise gráfica de resíduos: resíduos *vs.* valores esperados (esq.); e *normal probability plot* dos resíduos (dir.).

```
> residuos<-summary(ajuste)$resid
```

e, em seguida, representar esse gráfico em conjunto com o normal probability plot dos resíduos (Figura 4)

```
> op<-par(mfrow=c(2,1),pty="s",las=1)
> plot(residuos~predict(ajuste),pch=16,col=3,xlab="E{Proporção de respostas
correctas}",ylab="Resíduos")
> abline(h=0)
> qqnorm(residuos,pch=16,col=2,xlab="Quantis da Distribuição Normal",
ylab="Quantis Amostrais",main="")
> qqline(residuos)
> par(op)
```

O comando `par` permite estabelecer o “esquema” da figura, neste caso, os gráficos ficam “arranjados” em 2 linhas por 1 coluna – `mfrow=c(2,1)` –, assim como definir a respectiva forma (quadrada, através de `pty="s"`). Ao gráfico dos resíduos *vs.* valores esperados de Prop (através de `plot`) acrescentámos uma linha horizontal ao nível de zero (através de `abline`) e, por fim, desenhamos o *normal probability plot* dos resíduos (com `qqnorm` e `qqline`).

É possível testar se os resíduos se distribuem normalmente através, por exemplo, de

```
> shapiro.test(residuos)
```

O resultado deste comando é

```
Shapiro-Wilk normality test
data: residuos
W = 0.936, p-value = 0.5719
```

O teste de sequências (ou “runs test” nos manuais anglófonos) permite testar, de forma simples e robusta, se os pontos (dados) diferem sistematicamente da curva ajustada (modelo) complementando a informação do diagrama dos resíduos *vs* valores observados de X . No R, será necessário carregar alguns pacotes, `tseries`, `quadprog` e `zoo` (previamente instalados, através da barra de ferramentas, a partir dos ficheiros *.zip disponíveis para o Windows® em <http://cran.at.r-project.org/bin/windows/contrib/>) fazendo e.g.

```
> library(tseries)
```

Depois basta fazer

```
> runs.test(factor(sign(residuos)))
```

para se obter o seguinte resultado

```
Runs Test
data: factor(sign(residuos))
Standard Normal = 0.5774, p-value = 0.5637
alternative hypothesis: two.sided
```

Pode comparar-se o ajuste de dois modelos distintos a um dado conjunto de dados confrontando as SQE dos modelos ajustados através do teste de F. Sendo assim, depois de ajustar um modelo logístico, $\hat{y} = C/[1 + Aexp(-Bx)]$ (Figura 2d)⁶, de formulação mais generalizada do que o modelo “dose-resposta” anterior, através de

```
> ajustes<-nls(Prop~C/(1+A*exp(-B*C02)),data=data.frame(C02=C02,Prop=Prop),
start=list(A=10,B=.0025,C=1))
> summary(ajustes)
```

os resultados do teste de F obtêm-se facilmente através de

```
> anova(ajuste,ajustes)
```

Os passos descritos anteriormente, i.e. uma sessão de trabalho (Figura 5), podem (e, na minha opinião, devem) ser “guardados”. Para isso, criar um ficheiro de *script* (através de `File>New script`), de facto um ficheiro texto, onde são “guardados” os comandos. Este ficheiro pode ser carregado noutra sessão de trabalho (através de `File>Open script...`) e as análises e gráficos podem ser repetidos n-vezes (janela R Editor). O *script* do exemplo descrito aqui pode ser obtido no sítio electrónico do autor.

Para terminar uma sessão de trabalho, primeiro “limpar” da memória os objectos criados (com a função `rm`) e, depois, encerrar o programa (geralmente não guardo uma “imagem” do ambiente de trabalho):

⁶Esta figura foi obtida através de

```
> plot(Prop~C02,las=1,xlim=c(0,1000), ylim=c(0,1),pch=16,cex=1.5, xlab="C02 (ppm)",ylab="Propor
ção de respostas correctas")
> PropEsper<-predict(ajustes,newdata= list(C02=seq(0,1000,length=100)),se.fit=T)
> lines(PropEsper~seq(0,1000,length= 100),lwd=2,col=6)
> uppIC<-PropEsper+qt(.975,summary(ajustes)$df[2])*summary(ajustes)$sigm
> lowIC<-PropEsper-qt(.975,summary(ajustes)$df[2])*summary(ajustes)$sigm
> lines(uppIC~seq(0,1000,length=100), col=4)
> lines(lowIC~seq(0,1000,length=100), col=4)
```

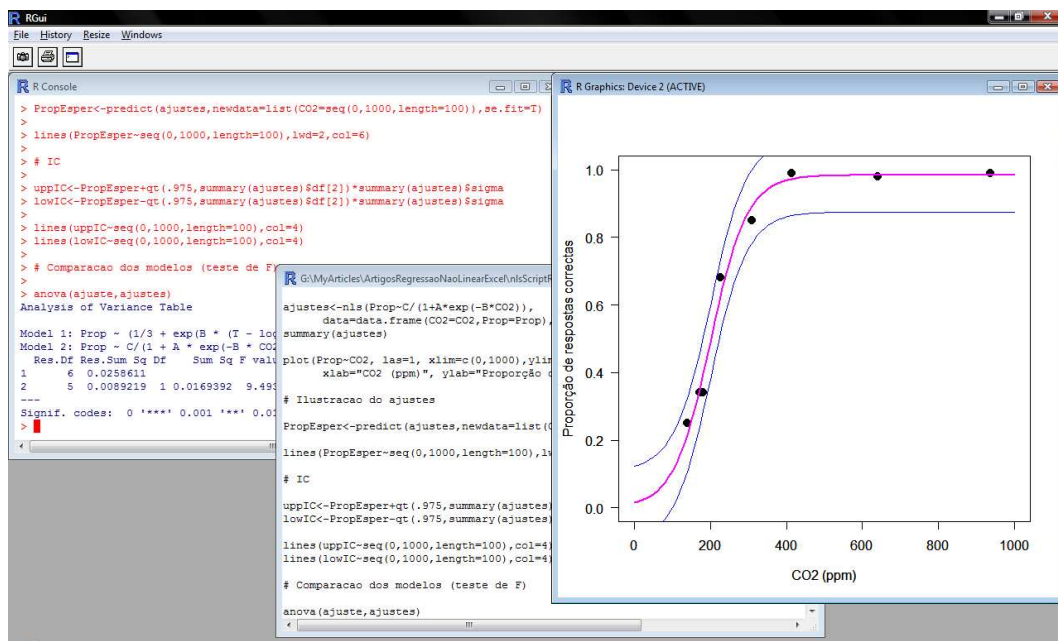


Figura 5: Aspecto duma sessão de trabalho no R.

```
> rm(list=ls()) ; q()
```

Considerações finais

As “dificuldades estatísticas” relacionadas com a utilização do Excel[®] da Microsoft (e da respectiva ferramenta Solver[®]) para análise de regressão não-linear (vd. Esteves, 2008) limitam a sua utilização aos casos “mais simples” ou para efeitos pedagógicos. Em situações “profissionais” deve usar-se software dedicado, por exemplo SPSS[®] ou R.

Embora as características do R tornem (mais) difícil a aprendizagem, o seu carácter gratuito, o facto de ser uma ferramenta bastante poderosa com boas capacidades ao nível da programação e um conjunto bastante vasto (e em constante crescimento) de packages que acrescentam bastantes potencialidades estatísticas e gráficas, o crescente interesse de utilizadores das mais variadas formações (que se reúnem anualmente desde 2004 nas conferências useR! – este ano a useR! 2009 decorrerá em França, cf. <http://www2.agrocampus-ouest.fr/math/useR-2009/>) e a disponibilidade dos promotores (R Development Core Team) e da “comunidade” de utilizadores, mais ou menos avançados, ajudam (e muito) esse processo. Os vários manuais (<http://cran.r-project.org/manuals.html>), a página wiki (<http://wiki.r-project.org/>) e a mailing list de ajuda (<http://stat.ethz.ch/mailman/listinfo/r-help>) permitem ultrapassar a (esmagadora) maioria das dificuldades surgidas durante a utilização do R.

Agradecimentos

Este artigo muito beneficiou dos comentários, correcções e sugestões dos colegas e leitores.

Referências

- Bowen, W. and Jerman, J. (1995). Nonlinear regression using spreadsheets. *Trends in Pharmacological Sciences*, 16:413–417.
- Brown, A. M. (2001). A step-by-step guide to non-linear regression analysis of experimental data using a microsoft excel spreadsheet. *Computer methods and Programs in Biomedicine*, 65:191–200.
- Chambers, J. M. and Hastie, T. J. (1992). *Statistical models in S*. Chapman & Hall, London.
- Dalgaard, P. (2002). *Introductory statistics with R*. Springer-Verlag New York Inc., New York.
- Esteves, E. (2008). Regressão não-linear utilizando a ferramenta solver[®] do microsoft excel[®]. Instituto Superior de Engenharia, Universidade do Algarve, Faro, Portugal [disponível em <http://w3.ualg.pt/~eesteves/>].
- R Development Core Team (2007). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
- Torgo, L. (2006). *Introdução à programação em R*. Faculdade de Economia da Universidade do Porto, Porto.
- Venables, W. N., Smith, D. M., and Team, R. D. C. (2006). *An Introduction to R (Notes on R: A Programming Environment for Data Analysis and Graphics)*. The R Foundation for Statistical Computing, Vienna, Austria.