XV COBREAP - CONGRESSO BRASILEIRO DE ENGENHARIA DE AVALIAÇÕES E PERICIAS - IBAPE/SP - 2009

TRABALHO DE AVALIAÇÃO

UTILIZAÇÃO DO *SOFTWARE* (GRATUITO) R NA ENGENHARIA DE AVALIAÇÕES

Resumo:*O emprego da metodologia científica e a investigação de modelos explicativos do mercado imobiliário abrangem diversas etapas de análise, razão pela qual se torna imprescindível o uso de computadores e softwares (pacotes) adequados à manipulação de dados e à interpretação dos resultados no trabalho avaliatório. Contudo, observa-se que os programas comerciais disponíveis, voltados especificamente para o segmento de avaliação de bens, apresentam um custo de aquisição relativamente elevado e aqueles aplicativos de acesso livre (gratuitos) são, em sua maioria, limitados. Diante deste cenário, o presente trabalho tem como objetivo apresentar, disseminar e incentivar o uso do Ambiente R, ou simplesmente R, um poderoso software de domínio público, utilizado e recomendado pela comunidade acadêmica de todo o mundo, nas mais diversas áreas do conhecimento. Importante destacar que, além de ser gratuito, o R apresenta código fonte aberto e permite a manipulação, avaliação e interpretação de procedimentos estatísticos aplicados à análise de dados. Para tanto, abordaremos neste trabalho os fundamentos da linguagem R, bem como faremos uma demonstração do software R através de uma aplicação com dados reais utilizando técnicas econométricas e testes estatísticos visando à obtenção de um modelo de regressão linear preditivo do mercado imobiliário que se analisa. Vale salientar que este trabalho não tem por finalidade difundir qualquer propaganda ou fazer qualquer comparação do aplicativo R com os demais softwares comerciais/livres de avaliação de bens existentes no mercado, mas sim proporcionar aos profissionais que atuam na área da engenharia de avaliações um ponto de partida para utilização desta importante ferramenta de análise e manipulação de dados em seus trabalhos avaliatórios.*

Palavras-chave: Avaliação de bens, Ambiente R, Análise de dados.

1.0 - Introdução

A avaliação de bens com base na metodologia científica é atualmente impensável sem o computador e o acesso a alguns *softwares* ou pacotes estatísticos.

Conforme Dantas (2005) bem observou, na utilização da informática para construção de um modelo explicativo do mercado, cresce muito a importância da "velocidade de processamento" do sistema/equipamento utilizado, tendo em vista a quantidade de dados e variáveis envolvidos. Além disto, a confiabilidade dos resultados obtidos a partir do uso de um programa de análise de dados é condição *sine qua non* para que o avaliador não cometa erros de especificação do modelo e possa inferir sobre o mercado.

Neste sentido, diversos programas computacionais específicos para a área de engenharia de avaliações foram lançados e estão disponíveis no mercado, tais como: REGRE, SISREN, SISREG, TS-SISREG, SAB, INFER, entre outros. Por se tratarem de *softwares* comerciais, tais aplicativos apresentam inúmeras facilidades de interação, adaptação e interpretação dos resultados. Contudo, em geral, possuem um custo de aquisição relativamente elevado e a maioria deles não permite que o usuário implemente ou modifique cálculos, gráficos e/ou funções de forma arbitrária, ou seja, são restritos (fechados) quanto ao código fonte.

Outro programa comercial que tem sido largamente utilizado para análise de dados por profissionais que militam na área de avaliações de bens é o aplicativo de planilha eletrônica de cálculo, escrito e produzido pela Microsoft, denominado de "Microsoft Excel". Embora não seja um *software* estatístico e não tenha sido desenvolvido com a finalidade de avaliar imóveis, seu uso é bastante difundido na área, talvez pela popularidade e interface bastante intuitiva que o aplicativo apresenta. Apesar da grande popularidade, duras críticas têm sido realizadas pela comunidade acadêmica nos últimos anos (vide, por exemplo, McCullough & Wilson, 2005) acerca das limitações, precisão e deficiências (erros) das análises estatísticas utilizando o Excel, razão pela qual seu emprego não tem sido incentivado e recomendado em trabalhos científicos robustos.

Diante destes fatos, surge naturalmente o seguinte questionamento: existe algum *software* que realize análise e manipulação de dados, cálculos, gere gráficos, seja eficiente, confiável, preciso e ainda seja de domínio público (gratuito)?

A resposta é SIM e este *software* é o Ambiente R, ou simplesmente R, como é usualmente conhecido pelos seus usuários.

O R encontrou uma rápida aceitação entre estatísticos, engenheiros e cientistas em todo o mundo e já vem sendo utilizado por diversas empresas, como, por exemplo: Google, Pfizer, Merck, Bank of America, InterContinental Hotels Group e Shell. É sobre este poderoso *software* e sua aplicação na engenharia de avaliações que se alicerça o presente trabalho.

2.0 - Ambiente R

O R faz parte do projeto GNU (*General Public License*) e é uma linguagem orientada a objetos, criada em 1995 por Ross Ihaka e Robert Gentleman, que, aliada a um ambiente integrado, permite a manipulação de dados, realização de cálculos e geração de gráficos. O R é semelhante a linguagem S, desenvolvida pela AT&T Bell Laboratories, cuja versão comercial é o S-Plus, mas tem a vantagem de ser de livre distribuição e apresentar código fonte aberto, o que permite sua modificação ou implementação com novos procedimentos desenvolvidos pelo usuário a qualquer momento. O R está disponível para as plataformas Linux, Apple Macintosh e Microsoft Windows.

É importante destacar que o R não é simplesmente um programa estatístico, uma vez que, devido às suas rotinas, permite também a manipulação, avaliação e interpretação de procedimentos estatísticos aplicados a dados. O R é, portanto, uma importante ferramente na análise e manipulação de dados, pois apresenta testes paramétricos e não paramétricos, modelagem linear e não linear, análise de séries temporais, análise de sobrevivência, simulação e estatística espacial, bem como permite a execução de operações matemáticas, manipulação de vetores e matrizes, confecção de gráficos com alta qualidade, entre outros recursos.

Outro aspecto bastante atrativo do R, refere-se ao fato de que pesquisadores em todo o mundo contribuem continuamente no desenvolvimento e implementação de novos recursos. Desde 1997 existe um núcleo - *o R Core Team* - de profissionais com a tarefa exclusiva de colaborar com avanços para novas versões do R. Esta é uma característica que contribui para o grande desenvolvimento do programa em um curto espaço de tempo.

Detalhes sobre o projeto R, colaboradores, documentação e diversas outras informações podem ser encontradas na página oficial do projeto em: `http://www.r-project.org`.

2.1 - Instalação

Para fazer o *download* do R é necessário acessar o *site* `www.r-project.org`, clicar em "CRAN" e escolher um servidor, de preferência no país e na cidade mais próxima do usuário. A seguir, deverá clicar em *"Download and Install R"*, e, após, no *link* que corresponde ao sistema operacional do computador onde será instalado o programa (no caso do Windows - "Windows"), e depois no *link* "base". Em seguida, clicar em "R 2.9.1 for Windows", onde a numeração 2.9.1 representa a versão do R disponível e mais recente no momento.

Concluído o download pelo usuário, basta salvar o arquivo executável em seu computador, executá-lo e seguir toda a rotina de instalação.

2.2 - Pacotes ou *packages*

Pacotes (*packages*), bibliotecas ou livrarias são os nomes mais usados para designar várias funções e comandos agrupados. Os pacotes contêm um conjunto de funções que facilitam ou possibilitam a realização das análises estatísticas. Os comandos básicos do R, por exemplo, estão em uma biblioteca chamada "base" e que já está inclusa na instalação deste programa. Existem inúmeras bibliotecas e várias delas foram desenvolvidas pelos próprios usuários, que criaram funções e comandos para suprir suas necessidades. Posteriormente, estas funções e comandos são agrupadas em um pacote (uma biblioteca) com um determinado nome e disponibilizadas a todos os usuários, para que outras pessoas que

necessitem usar as mesmas funções não precisem recriá-las. É essa colaboração mútua que faz do R um programa amplo e interdisciplinar.

Os pacotes extras podem ser obtidos a partir do próprio programa ou através do *site* `www.r-project.org`, sendo que em ambos os casos o usuário deve estar conectado a internet. No caso da instalação dos pacotes diretamente pelo programa R, o usuário deve seguir os passos abaixo:

1. Abrir o R;
2. Clicar em "Pacotes" e depois em "Instalar pacote(s)...";
3. Na janela "CRAN mirror", escolher um servidor que esteja mais próximo da cidade do usuário e clicar em "OK";
4. Selecionar um pacote de interesse e clicar em "OK". Este procedimento deve ser repetido para cada pacote a ser instalado, haja vista que só é possível instalar um por vez.

2.3 - Iniciando o R

O R armazena os dados em um arquivo chamado `.RData` que permanece gravado em disco entre sessões do R. É possível armazenar diversos arquivos `.RData` em diferentes diretórios, basta executar o R em um determinado diretório para que o arquivo seja criado. Desta forma, é aconselhável que ao iniciar um novo projeto seja criado um novo diretório e neste diretório seja iniciado o programa R.

No Windows, deve-se iniciar o R e definir o diretório de trabalho clicando em "Arquivo" $\rightarrow$ "Mudar diretório". Com o R iniciado, aparece o símbolo "$>$" em vermelho, que é o *prompt* do R, em uma janela conhecida como R *console*. Este sinal de *prompt* indica que o sistema está pronto para receber os comandos do usuário.

Para operar o R na forma usual é necessário conhecer e digitar comandos. Alguns usuários acostumados com outros programas irão notar, no início, a falta de "menus". Com o uso do programa, os usuários, ou boa parte deles, tendem a preferir o mecanismo de comandos, pois é mais flexível e apresenta mais recursos.

Entretanto, usuários iniciantes ainda podem preferir utilizar algum tipo de "menu". Visando atender a esta demanda, John Fox desenvolveu o pacote `Rcmdr`. Para utilizar este *package* basta instalá-lo e carregar com o comando `require(Rcmdr)` e o "menu" abrirá automaticamente. Atenção: o `Rcmdr` não provê acesso a toda funcionalidade do R, mas apenas a alguns procedimentos estatísticos mais usuais. Maiores informações sobre este pacote podem ser encontradas na página do `Rcmdr`, `http://socserv.socsci.mcmaster.ca/jfox/Misc/Rcmdr/`.

2.4 - Ajuda

O R tem um sistema de ajuda bastante elaborado que permite obter grande quantidade de informação sobre a linguagem e diversos outros aspectos que lhe estão associados. Na versão Windows do R, a forma mais fácil de obter ajuda é através da utilização do menu *"Help"*, disponível na janela da aplicação R. Através deste menu é possível, por exemplo, escolher a opção *"Html help"*, o que fará lançar um *browser* que acessará uma série de manuais e outros tipos de ajuda disponíveis no R.

No entanto, se o interesse reside em obter ajuda sobre uma função em particular do R, a forma mais simples será, provavelmente, utilizar a função `help()` (por exemplo `>help`(lm)). Este comando irá fazer aparecer uma pequena janela com todo um conjunto de informações úteis sobre a função escolhida, que vai da simples descrição dos seus argumentos até exemplos de utilização, bem como funções relacionadas. Alternativamente, pode-se optar pelo comando `?lm`, produzindo exatamente o mesmo efeito. Quando se desconhece o nome correto da função sobre a qual se quer ajuda, é possível usar como alternativas as funções `apropos()` e `help.search()`. De maneira geral, ambas produzem uma lista das funções do R que contêm referências ao texto que incluímos como argumento. Por exemplo, `apropos('model')` ou `help.search('model')`.

Para dúvidas mais complexas existe uma documentação gratuita disponível no *site* do R (`www.r-project.org`), assim como a *mailing list* de apoio no mesmo *site*.

Por fim, uma alternativa poderosa que reúne várias destas formas de ajuda no R é utilizar a função `RSiteSearch()`. O resultado da procura envolve a ajuda de todas as funções do R, ajuda nas *mailing lists*, bem como em outros documentos. Por exemplo, se o interesse é saber o que existe nestes locais sobre redes neurais, é possível lançar mão do seguinte comando:

```
> RSiteSearch('neural networks')
```

2.5 - Sair do programa

Para encerrar os trabalhos, deve-se sair do programa digitando no *prompt* `q()`. Imediatamente o programa exibirá a seguinte mensagem:

"Salvar imagem da área de trabalho? [Sim/Não/Cancelar]"

Se a resposta for "Não", então o R será finalizado e a sessão não será gravada. Caso a opção escolhida seja "Cancelar", a sessão do R continuará aberta para prosseguimento do trabalho. Porém, se a resposta for "Sim", o R vai guardar a informação contida na memória do computador em um ficheiro, de modo que da próxima vez que o R for executado no local onde este ficheiro foi armazenado, o trabalho poderá ser retomado exatamente de onde foi realizado o comando `q()`.

A informação guardada consiste basicamente no histórico de comandos executados na sessão, bem como dos objetos que foram criados. Neste caso, o R vai criar 2 ficheiros: um chamado `.Rhistory` contendo a lista dos comandos executados, e outro chamado `.RData` contendo os objetos criados na sessão. Os ficheiros com o estado da sessão são sempre gravados no diretório atual onde o R está funcionando. Para saber o diretório atual do R basta digitar no *prompt* `getwd()`. Em resposta a este comando o R irá apresentar na tela o diretório atual.

2.6 - Fundamentos da Linguagem R

No R, os símbolos ou variáveis são objetos e podem ter as mais variadas estruturas, tais como matrizes, vetores, *data frames*, listas, funções, expressões e muitas outras. É importante ressaltar que o nome de um objeto deve começar com uma letra qualquer, maiúscula ou minúscula, que pode ser seguida de outra letra, números ou caracteres especiais como o ponto. Cumpre registrar que as atribuições no R podem ser feitas por $<-$ ou $->$, dependendo da direção em que se atribui o objeto, ou por $=$ (sinal de igualdade). A operação de atribuição é destrutiva no sentido de que ao atribuir um novo valor a um objeto

já existente, perde-se o conteúdo que estava armazenado anteriormente. Além disso, o R faz distinção entre letras maiúsculas e minúsculas, de forma que os caracteres A e a são entendidos como sendo símbolos diferentes, referindo-se, portanto, a variáveis distintas.

Os comandos do R devem ser preferencialmente escritos em um editor de texto, como, por exemplo, no bloco de notas. O R permite abrir tais arquivos textos, denominados de *scripts*, nos quais são executados os códigos digitados marcando-os e teclando simultaneamente *"ctrl + r"*. Cada linha deve conter uma função, uma atribuição ou um comando. Pode-se digitar mais de um comando por linha se houver separação por ponto e vírgula (;). Ao atribuir um valor a um objeto o programa não irá imprimir nada na tela. Apenas quando for digitado o nome do objeto, o programa irá imprimir seu conteúdo na tela. Por exemplo, se um objeto `x` tem o valor 5, então ao se digitar `x`, aparecerá `[1]5`. O dígito 1 entre colchetes indica que o conteúdo exibido inicia-se com o primeiro elemento do objeto `x`.

2.6.1 - Operações aritméticas

O R também pode ser utilizado como uma calculadora. Ao digitar uma operação aritmética, sem atribuir o resultado a um objeto, o programa imprimirá o resultado na tela. Por exemplo:

```
> 3+8+1          #somando estes números ...
[1] 12           #obtem-se a resposta marcada com [1]

> 1+5*2          #adição e multiplicação
[1] 11           #prioridade de operações (multiplicação primeiro)

> 6/4+8
[1] 9.5          #assim como divisão

> 2*3**3         #potências são indicadas por  ** ou ^
[1] 54           #e tem prioridade sobre multiplicação e divisão
```

O R também disponibiliza funções como as que estão disponíveis em uma calculadora:

```
> exp(2)           #antilog de x
[1] 7.389056

> cos(pi)          #o valor "pi" está disponível como uma constante.
[1] -1

> sqrt(81)         #raiz quadrada
[1] 9

> factorial(4)     #4!=4*3*2*1
[1] 24

> choose(4,2)      #combinação de 4 elementos, 2 a 2. n!/(x!(n-x)!)
[1] 6
```

Na Tabela 2.1 a seguir apresentamos uma lista resumida de algumas funções no R.

Tabela 2.1: Funções no R

Nome	Função
sqrt(x)	raiz quadrada de x
abs(x)	valor absoluto (positivo) de x
sin(x) cos(x) tan(x)	funções trigonométricas de x em radianos
asin(x) acos(x) atan(x)	funções trig. inversas de x em radianos
sinh(x) cosh(x) tanh(x)	funções trig. hiperbólicas de x em radianos
asinh(x) acosh(x) atanh(x)	funções trig. hiperbólicas inversas de x em radianos
log(x)	log de base 'e' de "x"(logaritmo natural)
log10(x)	logaritmo de base 10 de x
log(x,n)	logaritmo de base n de x
exp(x)	antilog de x (e^x)

Estas funções podem ser agrupadas e combinadas em expressões mais complexas, por exemplo:

```
> log(cos(180/pi)/2)
[1] -1.002864
```

2.6.2 - Vetores

Vetores são o tipo básico e mais simples de objeto para armazenar dados no R. Este aplicativo é uma linguagem vetorial e, portanto, capaz de operar vetores e matrizes diretamente sem a necessidade de "loops", como por exemplo em códigos C e/ou Fortran.

Antes de apresentar alguns exemplos utilizando operações com vetores é preciso destacar que a função `c()` ("c" de concatenar) é usada para criar um vetor, já os colchetes [ ] são utilizados para indicar seleção de elementos e as funções `rep()`, `seq()` e o símbolo ":" são usados para facilitar a criação de vetores que tenham alguma lei de formação. Exemplos:

```
> x <- c(1,3,5,7,9)   # os 5 primeiros números ímpares
> x
[1]  1  3  5  7 9     # digitando o nome do objeto é exibido o seu conteúdo
```

Os argumentos de `c()` podem ser escalares ou vetores, vejamos:

```
> y <- c(x,11,13,15,17)   # adicionando  mais quatro números ímpares
> y
[1]  1  3  5  7  9  11  13  15  17

> x2[1]
[1] 1
> x2[2]
[1] 3

> xx <- 1:8
> xx
 [1]  1  2  3  4  5  6  7  8
```

Se o vetor é muito longo e não “cabe” em uma única linha, o R vai usar as linhas seguintes para continuar imprimindo o vetor na tela, conforme representado no exemplo a seguir:

```
> xx <- 100:1    # sequência decrescente de 100 a 1
> xx
  [1] 100  99  98  97  96  95  94  93  92  91  90  89  88  87  86  85  84  83
 [19]  82  81  80  79  78  77  76  75  74  73  72  71  70  69  68  67  66  65
 [37]  64  63  62  61  60  59  58  57  56  55  54  53  52  51  50  49  48  47
 [55]  46  45  44  43  42  41  40  39  38  37  36  35  34  33  32  31  30  29
 [73]  28  27  26  25  24  23  22  21  20  19  18  17  16  15  14  13  12  11
 [91]  10   9   8   7   6   5   4   3   2   1
```

Os números entre colchetes não fazem parte do objeto e indicam a posição do vetor naquele ponto. Pode-se observar que [1] indica que o primeiro elemento do vetor está naquela linha, [19] indica que a linha seguinte começa pelo décimo nono elemento do vetor e assim por diante. Outros exemplos:

```
> seq(1,10,1)             # o mesmo que 1:10
 [1]  1  2  3  4  5  6  7  8  9 10
> seq(1,10,2)             # de 2 em 2
[1] 1 3 5 7 9             # não necessariamente termina em 10

> rep(1,9)                #repete o primeiro argumento o número de vezes
 [1] 1 1 1 1 1 1 1 1 1    #determinado pelo segundo argumento

> rep(c(1,2),10)
 [1] 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
```

2.6.3 - Caracteres

O R pode armazenar dados alfanuméricos da mesma forma que armazena dados numéricos. Dados na forma de caracteres devem ser representados entre aspas simples (‘ ’) ou duplas (“ ”). É possível usar uma ou outra, desde que sejam os mesmos símbolos no início e no final. Exemplos:

```
> x <- "Avaliação de Bens"   # um caracter como escalar
> x
[1]"Avaliação de Bens"

> x1 <- c("Área","Frente","Orientação solar")  # um vetor de caracteres
> x1
[1] "Área" "Frente" "Orientação solar"
```

Obviamente, operações numéricas como soma, subtração, multiplicação etc., não fazem sentido.

2.6.4 - Fatores

Fatores são usados para armazenar dados categóricos. Por exemplo, suponha que precisamos armazenar a informação sobre a natureza da informação coletada de um dado

de mercado. Neste caso, é possível usar um código numérico, como 0 (zero) para “transação” e 1 (um) para “oferta”, ou pode-se utilizar um código na forma de caractere, como “T” para transação e “O” para oferta. Mas, em ambos os casos deve-se usar um fator.

Fatores são facilmente construídos a partir de vetores alfanuméricos através da função `as.factor`, vejamos:

```
> y1 <- c("O","O","T","T","O","T")
> y1
[1] "O" "O"' "T" "T" "O" "T"
> z1 <- as.factor(y1)
> z1
[1] O O T T O T
Levels:  O T
```

Note que fatores são mostrados de forma semelhante, mas não idêntica aos vetores alfanuméricos. Os valores dos fatores são impressos sem aspas e os níveis do fator são também impressos. As categorias (níveis) de um fator podem ser vistas usando a função `levels()`, como apresentamos abaixo:

```
> levels(z1)
[1]"O" "T"      # o resultado é um vetor alfanumérico
```

Além disso, os níveis de um fator podem ser facilmente modificados assinalando um novo vetor alfanumérico aos níveis do fator. Exemplo:

```
> levels(z1) <- c("OFERTA","TRANSAÇÃO")
> z1
[1] OFERTA OFERTA TRANSAÇÃO TRANSAÇÃO OFERTA TRANSAÇÃO
Levels: OFERTA TRANSAÇÃO
```

2.6.5 - Valores lógicos

O R possibilita a computação de valores *booleanos* ou variáveis lógicas. Estas variáveis assumem TRUE ou FALSE (verdadeiro ou falso) e podem ser armazenadas da mesma forma que os valores numéricos ou caracteres. Os valores lógicos são gerados quando uma certa condição é testada, conforme demonstrado a seguir:

```
> idade.imovel <- 1:10
> idadeM5 <- idade.imovel > 5
> idadeM5
 [1]  FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE

> idade.imovel == 6
 [1] FALSE FALSE FALSE  FALSE FALSE TRUE FALSE FALSE FALSE FALSE

> idadeM3 <- idade.imovel > 3
> idadeM3
 [1] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

2.6.5.1 - Operadores lógicos

É possível fazer operações do tipo “e” e “ou” em vetores usando os símbolos “&” e “ | ”, respectivamente. Exemplos:

```
> idade.imovel > 2 & idade.imovel <= 5
[1] FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
> idade.imovel < 2 | idade.imovel >= 7
 [1] TRUE FALSE FALSE FALSE FALSE FALSE TRUE  TRUE  TRUE  TRUE
```

Acrescenta-se que o R interpreta qualquer operação numérica com valores lógicos TRUE e FALSE como sendo os valores 1 e 0, respectivamente. Isto pode ser utilizado, por exemplo, para contar o número de valores em um vetor que obedece a uma determinada condição, senão vejamos:

```
> area <- c(200,150,185,30,90,120,450,230,210)
> y <- area < 200
> y
[1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE
> y*1
[1] 0 1 1 1 1 1 0 0 0
> sum(y)                # conta o número de TRUE's em y
[1] 5
```

2.6.6 - Formato das funções

As funções no R são compostas de uma lista de argumentos separados por vírgulas entre parênteses. Argumentos são classificados em *unnamed* (sem nome - não nomeados), *named* (com nome - nomeados), ou *missing* (ausentes). Podemos exemplificar usando a função `seq( )`. Esta função possui três argumentos e é utilizada para gerar uma sequência:

```
> seq(1,12,3)
[1]  1  4  7 10
```

Estes são argumentos do tipo *unnamed*. Pode-se usar argumentos com nomes na seguinte forma:

```
> seq(from=1,to=12,by=3)
[1]  1  4  7 10 13 16 19 22 25
```

Note que cada valor do argumento é precedido por “nome =”. Isto ocorre por duas razões, porque o código fica mais legível e porque algumas funções podem retornar resultados diferentes dependendo dos nomes dos argumentos. Por exemplo, a função `seq( )` pode gerar uma sequência de “certo” (alguma quantidade) número de elementos se for utilizado o argumento `length=`:

```
> seq(from=1,to=12,length=5)
[1] 1.00 3.75 6.50 9.25 12.00
```

Isto retorna uma sequência de 5 (cinco) valores que começa no valor do argumento *from* e termina no valor do argumento *to*. Neste caso, o argumento *by* com os intervalos entre os valores é calculado automaticamente (no exemplo é igual a 2.75).

Os nomes dos argumentos podem ser abreviados desde que sejam identificados unicamente dentro do conjunto dos possíveis nomes de argumentos da determinada função. No exemplo acima, seria possível escrever `seq(f=1,t=12,l=5)`. Além disso, a ordem dos argumentos com nomes é irrelevante, ou seja, o mesmo resultado poderia ser obtido digitando `seq(t=12,l=5,f=1)`.

Os argumentos não fornecidos são chamados de *missing* (ausentes). Se a função tem um valor *"default"* para um argumento ausente, este argumento é considerado como um argumento opcional, caso contrário é um argumento requerido (obrigatório), que se não for fornecido faz com que a função emita uma mensagem.

```
> seq(1,4)   # by ausente,  default igual a 1
[1] 1 2 3 4
> seq(4)     # from ausente, defaults igual a 1
[1] 1 2 3 4
> seq()      # todos ausentes, defaults igual a 1
[1] 1
```

O R em geral tem *defaults* razoáveis para a maioria dos argumentos, mas nem todas funções tem *defaults*. Exemplo:

```
> cos()
Erro em cos() : 0 arguments passed to "cos" which requires 1
```

2.6.7 - Listas

As listas são objetos utilizados para combinar diferentes "estruturas" em um mesmo objeto. Estas "estruturas" podem ser vetores, matrizes, números e/ou caracteres e até mesmo outras listas. Exemplo:

```
> dado1 <- list(idade=15,vocação="comercial",dist.polos.infl=c(100,1000,3000))
> dado1
$idade
[1] 15
$vocação
[1] "comercial"
$dist.polos.infl
[1]  100 1000 3000
```

Listas são construídas com a função `list( )`. Os componentes da lista são introduzidos usando a forma usual `nome=arg` de atribuir argumentos em uma função. Cada elemento da lista pode ser acessado individualmente por seu nome antecedido pelo símbolo $:

```
> dado1$idade          # idade
[1] 15
> dado1$dist.polos.infl[2]      # segundo elemento de $dist.polos.infl
[1] 1000
```

Pode-se ainda acessar cada elemento pelo seu número de ordem na lista utilizando colchetes duplos:

```
> dado1[[1]]
[1] 15
> dado1[[2]]
[1] "comercial"
```

2.6.8 - Matrizes

Há várias formas de criar uma matriz no R. A função `matrix( )` recebe um vetor como argumento e o transforma em uma matriz de acordo com as dimensões especificadas. Exemplo:

```
> x <- 1:16
> y <- matrix(x,ncol=4)
> y
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
```

Neste exemplo foi construída uma matriz de 4 colunas e 4 linhas usando os números de 1 a 16. Por *default* a matriz é preenchida ao longo das linhas. Para inverter este padrão, ou seja, para que a matriz seja preenchida por colunas, deve-se adicionar o argumento `byrow=T`. Exemplo:

```
> y <- matrix(x,ncol=4,byrow=T)
> y
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
```

Para extrair um único elemento da matriz utilizam-se colchetes com dois números separados por vírgula. O primeiro número indica o número da linha, enquanto o segundo indica o número da coluna. Exemplo:

```
> y[2,3]
[1] 7
```

Pode-se extrair uma linha inteira ou uma coluna inteira usando apenas um número e a vírgula. Para extrair uma coluna, deve-se digitar o número da coluna desejada depois da vírgula. Da mesma forma, para extrair uma linha, digita-se o número da linha desejada depois da vírgula. Quando se extrai uma linha ou uma coluna o resultado é um vetor, conforme se observa a seguir:

```
> y[,4]                 # extraindo a quarta coluna
[1]  4  8 12 16
> y[2,]                 # extraindo a segunda linha
[1] 5 6 7 8
```

2.6.9 - *Data Frames*

Um *data frame* é um objeto do R normalmente utilizado para guardar tabelas de dados de um problema qualquer. Na sua forma, um *data frame* é muito semelhante a uma matriz, mas as suas colunas têm nomes e podem conter dados de tipos diferentes, contrariamente a uma matriz. Um *data frame* pode ser visto como uma tabela de uma base de dados, onde cada linha corresponde a um registo (linha) da tabela e cada coluna corresponde às

propriedades (campos) a serem armazenadas para cada registo da tabela. Um *data frame* pode ser criado da seguinte forma:

```
> dados <- data.frame(area = c(120, 90, 80, 200),
+ orient.solar = c("nascente", "nascente", "poente", "poente"),
+ valor.unit = c(100.3, 90.3,145.2, 215))
> dados
   area   orient.solar   valor.unit
1  120    nascente       100.3
2  90     nascente       90.3
3  80     poente         145.2
4  200    poente         215.0
```

Os elementos de um *data frame* podem ser selecionados como em uma matriz, senão vejamos:

```
> dados[1,2]
[1] nascente
Levels: nascente poente
```

2.6.10 - Entrada de dados externos

O conjunto de dados a ser usado em determinada análise pode ser digitado diretamente no *console* do R ou lido de arquivos externos. Neste caso, diversos comandos podem ser utilizados com a finalidade de ler ou editar dados. Entre eles podemos citar: `scan( ), edit( ), read.table( )` e `data( )`.

Se os dados estiverem em uma planilha de dados, deve-se fazer com que o R leia esta planilha e a transforme em um objeto. Mas para isso, deve ser informado o que separa uma coluna da outra. As planilhas do tipo *.xls*, que é o formato padrão do Excel, não possuem nenhum sinal separando as colunas. Portanto, o R não conseguirá separá-las e irá exibir uma mensagem de erro. Um modo fácil de resolver este problema é salvar a planilha de dados com um outro formato *(.csv)*, que utiliza o sinal ";" para separar as colunas. Isto pode ser feito no próprio Excel.

Outro possível problema que pode ocorrer na entrada de dados é a configuração do sinal utilizado para separar as casas decimais. No Brasil, o padrão utilizado é a vírgula, enquanto o R utiliza o ponto, que é o padrão da língua inglesa. Temos duas soluções possíveis para a questão: informar ao R que o arquivo a ser lido utiliza a vírgula como separador decimal quando ele irá convertê-la em ponto ou, ainda, alterar a configuração do editor de planilha para que ele aceite o ponto como separador decimal. Ao adotar a segunda opção, para salvar a planilha de dados em formato *(.csv)*, deve-se clicar em "Arquivo" e logo após em "Salvar como...", em seguida selecionar a pasta onde o arquivo será salvo, bem como o formato (CSV (separado por vírgulas)(*csv)). O Excel retornará duas mensagens, clicar em "OK" na primeira e "Sim" na segunda.

Antes de iniciar a entrada dos dados para o R, deve-se alterar a pasta de trabalho padrão para a pasta de trabalho onde foi salva a planilha de dados na extensão *.csv*. Os passos para verificar se o arquivo salvo (por exemplo `teste.csv`) está na pasta de trabalho e para ler os seus dados são:

```
> dir()
[1] "teste.csv"
> dadoscasas=read.table(file="teste.csv",sep=";",header=T,dec=".")
```

Os termos técnicos utilizados anteriormente são definidos a seguir:

- **dadoscasas**: objeto pelo qual os dados lidos serão reconhecidos pelo R, cujo nome poderia ser outro;
- **=**: sinal que atribui os dados lidos ao objeto `dadoscasas`;
- **read.table**: função que lê arquivos do tipo *.csv*;
- **file**: informa o endereço e o nome do arquivo a ser lido;

Além do endereço e do nome, outros atributos do arquivo de dados devem ser especificados:

- **sep**: indica qual é o separado de colunas;
- **header**: informa se o nome das colunas estão na primeira linha (TRUE) ou não (FALSE);
- **dec**: informa qual é o separador decimal para que o R converta para ponto.

Ao final, para ver se os dados foram lidos corretamente, deve-se digitar o nome **dadoscasas** no *console*, cujos dados irão aparecer no formato *data frame*, que é o valor retornado pela função utilizada.

3.0 - Aplicação do *software* R com dados reais

3.1 - Dados da análise

Visando à obtenção de um modelo de regressão linear para previsão do valor médio de mercado (na condição de oferta) para compra e venda de uma casa residencial (veraneio) situada na praia de Porto de Galinhas, Ipojuca, Pernambuco, considerou-se uma amostra composta por 40 observações - casas à venda semelhantes ao avaliando - extraídas de pesquisa realizada em novembro/2007 junto a corretores/classificados de jornais ou proprietários.

Cumpre registrar que o conjunto de dados utilizado no presente trabalho subsidiou a elaboração do laudo de avaliação do imóvel avaliando (casa em Porto de Galinhas) para lastrear (através de hipoteca) operação de crédito de terceiros junto a uma instituição financeira pública brasileira no ano de 2007. Com isto, o referido laudo foi enquadrado na época como de uso restrito e sob a condição de não ser revelado a outrem. Por este motivo, não disponibilizamos o banco de dados com as informações referentes à identificação dos valores assumidos pelas variáveis que caracterizam o avaliando e as observações que compõem a amostra.

Destaca-se, entretanto, que as exposições, análises e conclusões realizadas neste trabalho não foram comprometidas em virtude desta restrição, haja vista que não houve perda de informação inerente a compreensão dos resultados obtidos e/ou apresentados.

3.2 - Detalhes metodológicos

As apresentações gráficas e a análise de regressão (manipulação de dados, estimação de parâmetros, mudanças de escala nas variáveis, testes de hipóteses, identificação de pontos influenciantes, intervalos de confiança, entre outras investigações) realizadas ao longo deste trabalho foram produzidas através do ambiente de programação R, tendo sido utilizada a versão 2.9.1 para a plataforma Windows. O Apêndice A contém todos os códigos fonte de programação na plataforma R empregados na análise de regressão deste trabalho.

Visando à estimação empírica dos diversos modelos de avaliação para estimação do valor médio de mercado (na condição de oferta) para compra e venda de uma casa residencial (veraneio) situada na praia de Porto de Galinhas percorremos sete fases interrelacionadas, a saber: (i) Especificação do modelo estatístico. (ii) Estimação dos parâmetros. (iii) Teste de especificação do modelo. (iv) Verificação das violações das premissas do modelo de regressão linear clássico. (v) Testes de hipóteses. (vi) Identificação de pontos influenciantes. (vii) Escolha do modelo.

Para construção dos modelos de regressão foram consideradas as seguintes variáveis:

- Variável dependente:

    1. PREÇO UNITÁRIO (PU): corresponde ao valor do imóvel, na condição de oferta, dividido pela área construída privativa da unidade, quantificada em $R\$/m^2$. Ao longo deste trabalho será referenciada pela abreviação PU.

- Variáveis independentes:

    1. ÁREA CONSTRUÍDA (AC): considerada como variável quantitativa correspondente à área privativa da unidade, medida em m^2. Ao longo deste trabalho será referenciada pela abreviação AC.;

2. DISTÂNCIA À PRAIA (DP): considerada como variável quantitativa correspondente à distância (em metros) da unidade até a beira-mar (praia). Ao longo deste trabalho será referenciada pela abreviação DP;
3. LOCALIZAÇÃO (LOC): adotada como uma variável *dummy*, assumindo valor 0 (zero) para casas isoladas e 1(um) para imóveis em condomínio. Ao longo deste trabalho será referenciada pela abreviação LOC.

A fim de estimar os parâmetros do modelo empregamos o método dos mínimos quadrados ordinários. Embora para estimação pontual dos parâmetros não seja necessária nenhuma pressuposição acerca da distribuição de probabilidade dos erros, utilizamos o teste de normalidade Jarque-Bera (JB) para verificar se os resíduos são normalmente distribuídos e, consequentemente, estabelecer intervalos de confiança e fazer testes de hipóteses. Salienta-se que em todos os testes realizados adotamos o nível de significância igual a 5%. Para detecção de erros de especificação no modelo aplicamos o teste RESET de Ramsey com a inclusão de $\hat{Y}^2$ e $\hat{Y}^3$ como regressores adicionais.

Em se tratando das violações das premissas do modelo clássico de regressão linear, verificamos a transgressão destas hipóteses no que tange a multicolinearidade nos regressores e heteroscedasticidade dos termos de erro, além dos já comentados erros de especificação do modelo e distribuição normal dos erros. A multicolinearidade quase exata foi averiguada simultaneamente pelas seguintes formas: análise das correlações entre pares de regressores e determinação dos fatores de inflação de variância (FIV). No que diz respeito à detecção da presença de heteroscedasticidade, utilizamos os testes de Goldfeld-Quandt, Breusch-Pagan e o teste de Koenker. Os testes de hipóteses utilizados para verificar a significância individual dos coeficientes parciais da regressão e a significância geral do modelo foram o teste t e o teste F, respectivamente.

As observações influentes que possuem um erro associado "muito" grande (*outliers*) ou que estão desproporcionalmente distantes do grosso dos valores de um ou mais regressores (pontos de alavanca) foram inquiridas por meio da matriz chapéu H, dos DFFITS e da distância de Cook.

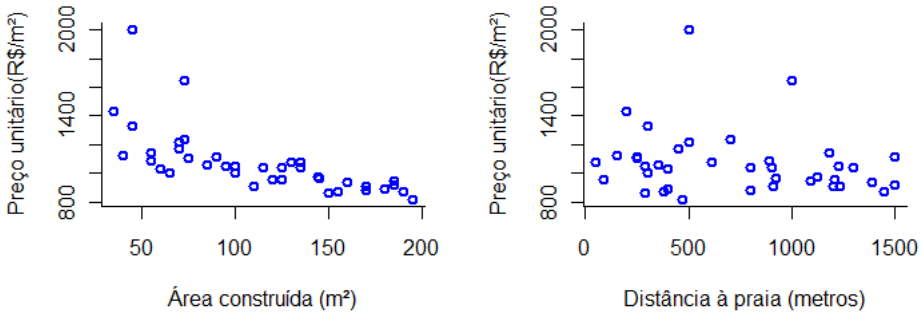
Por fim, a adequação e a escolha do modelo se basearam na observância da coerência teórica e lógica das variáveis explanatórias em relação à variável dependente, tanto no que diz respeito às expectativas a priori dos sinais dos coeficientes estimados como pela significância estatística dos regressores. Adicionalmente, os modelos concorrentes à realização da predição foram comparados (quando possível) entre si de acordo com o critério de informação de Akaike (AIC) e critério de informação bayesiano (BIC), além do coeficiente de determinação ajustado $(\bar{R}^2)$.

3.3 - Resultados e Análises

3.3.1 - Análise explanatória

O objetivo desta seção é proceder à análise exploratória dos dados através do uso de gráficos, tabelas e descrição das medidas de posição e dispersão. Primeiramente, apresentamos na Figura 3.1 abaixo os diagramas de dispersão das variáveis ÁREA CONSTRUÍDA e DISTÂNCIA À PRAIA com a variável PREÇO UNITÁRIO.

Figura 3.1: Diagramas de dispersão



Conforme podemos observar, há uma tendência de decréscimo do PREÇO UNITÁRIO do imóvel na medida em que a sua ÁREA CONSTRUÍDA aumenta. Contudo, não podemos tirar a mesma conclusão entre a relação do PREÇO UNITÁRIO e a DISTÂNCIA A PRAIA. Neste caso, o diagrama de dispersão está indicando, aparentemente, que a DISTÂNCIA À PRAIA não exerce (ou exerce pouca) influência (linear) sobre o PREÇO UNITÁRIO.

Quanto à forma da curva, verifica-se que os pontos dispostos no gráfico de PREÇO UNITÁRIO versus ÁREA CONSTRUÍDA apresentam características de linearidade, enquanto que entre as variáveis PREÇO UNITÁRIO e DISTÂNCIA À PRAIA não evidenciam quaisquer formas específicas.

Outra análise que podemos fazer diz respeito à matriz de correlação dois a dois (Tabela 3.1). Mediante exame desta tabela podemos complementar as observações mencionadas nos dois parágrafos anteriores, uma vez que é possível constatar a influência inversa e moderada da ÁREA CONSTRUÍDA sobre o PREÇO UNITÁRIO através da obtenção da medida de correlação (-0.551) entre PREÇO UNITÁRIO e ÁREA CONSTRUÍDA. Em se tratando do PREÇO UNITÁRIO e da DISTÂNCIA À PRAIA, percebemos que, de fato, esta associação linear é bastante fraca (-0.203). Cumpre registrar ainda que o coeficiente de correlação $(r = 0.4967)$ entre as variáveis ÁREA CONSTRUÍDA e DISTÂNCIA À PRAIA sinaliza que há uma pequena dependência linear entre elas, ou seja, dificilmente iremos ter violação da premissa de não multicolinearidade do modelo de regressão linear clássico se incluídas conjuntamente.

Tabela 3.1: Correlações entre as variáveis

	Preço unitário	Área construída	Distância à praia
Preço unitário	1.0000	-0.551	-0.203
Área construída	-0.551	1.0000	0.4967
Distância à praia	-0.203	0.4967	1.0000

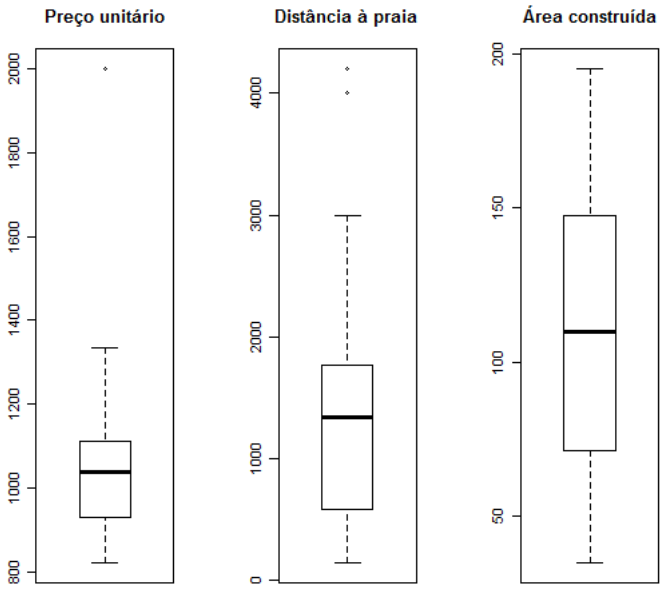
Sintetizamos na Tabela 3.2 abaixo algumas medidas de posição e dispersão acerca das variáveis observadas no conjunto de dados. Vale salientar que as unidades de medida das variáveis PREÇO UNITÁRIO, ÁREA CONSTRUÍDA e DISTÂNCIA À PRAIA são $R\$/m^2$ (reais por metro quadrado de área construída privativa da unidade), m^2 (metro quadrado) e m (metro), respectivamente.

Tabela 3.2: Medidas de posição e dispersão

	Preço unitário	Distância à praia	Área construída
Mínimo	820.50	50.00	35.00
1º Quartil	932.90	300.00	72.25
Mediana	1038.50	655.00	110.00
Média	1069.20	721.20	111.38
3º Quartil	1111.10	1136.00	146.25
Máximo	2000.00	1500.00	195.00
Desvio Padrão	221.17	445.91	47.01

Um recurso bastante útil e que auxilia na interpretação da variabilidade e distribuição dos dados das variáveis é o gráfico *box-plot*, que apresentamos a seguir (Figura 3.2). Analisando os referidos gráficos, constatamos que a variável PREÇO UNITÁRIO se distribui de maneira assimétrica e apresenta um possível ponto *outlier*, a observação número 01. No que diz respeito à variável DISTÂNCIA À PRAIA, verificamos que a assimetria na distribuição é à esquerda e que duas observações foram consideradas atípicas, as observações de número 05 e 15. Em se tratando da variável ÁREA CONSTRUÍDA, percebemos que a distribuição é aproximadamente simétrica e que não há observações discrepantes.

Figura 3.2: Gráficos Box-Plot



3.3.2 - Escolha do modelo

Nesta seção discorreremos acerca do processo de escolha do modelo de regressão linear para previsão do valor médio de mercado para compra e venda (na condição de oferta) de uma casa residencial (veraneio) situada na praia de Porto de Galinhas, Ipojuca, Pernambuco.

Com base na introspecção e em trabalhos empíricos anteriores, formulamos um modelo que consideramos capaz de captar a essência do fenômeno estudado. Então, submetemos o modelo à testes empíricos. Depois de obter os resultados começamos a dissecação, mantendo em mente os critérios de um bom modelo. Cabe aqui, entretanto, o alerta de Stigler (1987) acerca dos testes de hipóteses:

"Cuidado para não testar hipóteses demais; quanto mais torturamos os dados, maior a probabilidade de que confessem, mas a confissão obtida à força pode não ser admissível no tribunal da opinião científica."

Com o objetivo de não incorrer em formulação de modelos ilógicos e realizar testes de hipóteses inapropriados, inicialmente, expusemos as expectativas a priori acerca das possíveis relações entre as variáveis independentes e o PREÇO UNITÁRIO do imóvel na condição de oferta:

- Variável independente ÁREA CONSTRUÍDA: suspeitamos que a variável ÁREA CONSTRUÍDA tenha um efeito negativo no PREÇO UNITÁRIO médio dos imóveis, ou seja, quanto maior a área construída da unidade menor o preço unitário (médio);
- Variável independente DISTÂNCIA À PRAIA: acreditamos que a variável DISTÂNCIA À PRAIA tenha um efeito inversamente proporcional no PREÇO UNITÁRIO médio das unidades, haja vista que trabalhos realizados anteriormente em outras regiões litorâneas apresentaram esta característica (embora em alguns casos esta influência não tenha sido estatisticamente significante).

Adicionalmente e baseado em experiências anteriores, incluímos variáveis do tipo *dummy* na tentativa de captar a relação entre a localização da unidade e o PREÇO UNITÁRIO médio dos imóveis. Diante disto, propusemos a seguinte variável dicotômica:

- VARIÁVEL *DUMMY* LOCALIZAÇÃO: atribui 1 (um) se o imóvel observado está localizado dentro de um condomínio e 0 (zero) caso contrário (por exemplo casas isoladas). A expectativa é de que os imóveis situados dentro de um condomínio possuam PREÇO UNITÁRIO médio superior aqueles que estão localizados fora do condomínio.

A partir destas considerações, examinamos diversos modelos incluindo transformações na variável resposta (PREÇO UNITÁRIO (PU)) e nos regressores (ÁREA CONSTRUÍDA (AC) e DISTÂNCIA À PRAIA (DP)), bem como incluímos a variável do tipo *dummy* LOCALIZAÇÃO (LOC) e fizemos interações entre os previsores. Na Tabela 3.3 a seguir resumimos os principais modelos testados e as observações relevantes acerca dos testes realizados.

Tabela 3.3: Modelos candidatos a predição

Modelos	Forma Funcional	Considerações
1	$\text{PU} = \beta_0 + \beta_1 \text{AC} + \beta_3 \text{DP} + \epsilon$	A variável DP mostrou-se não significativa mediante aplicação do teste t. $\bar{R}^2 = 0,3097$ $AIC = 418,5884$ e $BIC = 425,9029$
2	$\text{PU} = \beta_0 + \beta_1 \text{AC} + \epsilon$	Embora a variável AC seja significativa através da aplicação do teste t, o teste Reset de Ramsey indicou que o modelo está mal especificado. $\bar{R}^2 = 0,3031$ $AIC = 417,0265$ e $BIC = 422,5125$
3	$\log(\text{PU}) = \beta_0 + \beta_1 \log(\text{AC}) + \epsilon$	Através do teste comparativo da forma funcional, constatou-se que este modelo é equivalente ao modelo 2. $\bar{R}^2 = 0,2913$ $AIC = -32,3904$ e $BIC = -26,9045$
5	$\text{PU} = \beta_0 + \beta_1 \text{AC} + \beta_2 \text{DP} + \beta_3 \text{LOC} + \beta_4 \text{DP} \times \text{LOC} + \epsilon$	O resultado do teste de Koenker levou a rejeição de homoscedasticidade e os fatores de inflação da variância indicaram presença de multicolinearidade. $\bar{R}^2 = 0,5608$ $AIC = 401,7893$ e $BIC = 412,7612$
6	$\text{PU} = \beta_0 + \beta_1 \text{AC} + \beta_2 \text{LOC} + \epsilon$	Neste modelo todas as variáveis independentes foram significativas sobre a análise do teste t. Além disso, as premissas do modelo de regressão linear clássico não foram violadas. $\bar{R}^2 = 0,3931$ $AIC = 412,6700$ e $BIC = 419,9846$
7	$\log(\text{PU}) = \beta_0 + \beta_1 \log(\text{AC}) + \beta_2 \text{LOC} + \epsilon$	As considerações dos resultados obtidos para este modelo são análogas às do modelo 6. $\bar{R}^2 = 0,3852$ $AIC = -36,9273$ e $BIC = -29,6127$

Com relação aos critérios de seleção utilizados, os modelos que forneceram os maiores valores para $\bar{R}^2$ e menores valores de AIC e BIC foram os modelos 5, entre aqueles que possuem PU como variável resposta, e o 7, entre os que possuem a variável resposta na forma logaritmada do PU. No entanto, o modelo 5 apresentou indícios de multicolinearidade e heteroscedasticidade, enquanto que no modelo 7, embora todas as variáveis explicativas sejam significativas e não tenha sido violada nenhuma das premissas do modelo de regressão linear clássico, a interpretação dos seus coeficientes de regressão não se dá de forma direta. Em virtude disto, optamos pelo modelo 6 (seis), ou seja, $\text{PU} = \beta_0 + \beta_1 \text{AC} + \beta_2 \text{LOC} + \epsilon$. A preponderância do modelo 6 se fundamenta nos seguintes aspectos:

1. O $\bar{R}^2$ forneceu o segundo maior valor obtido entre os modelos com a mesma variável resposta;
2. Os valores dos AIC e BIC foram os segundos menores entre os modelos com a mesma variável resposta;
3. Nenhuma das premissas do modelo de regressão linear clássico foi violada;
4. Princípio da parcimônia. O modelo 6 apresenta facilidade de interpretação dos coeficientes da regressão e é coerente com a teoria e os dados que se têm disponíveis.

3.3.3 - Análise do modelo

Inicialmente, observamos que o teste F aplicado ao modelo 6 forneceu p-valor igual a $2,1750 \times 10^{-5}$, indicando a significância do modelo ao nível de 5%. Além disso, os p-valores obtidos do teste t também ficaram abaixo de 5% (Tabela 3.5), implicando a não-rejeição das hipóteses de nulidade dos parâmetros individualmente. Os intervalos de confiança apresentados nesta tabela confirmam este resultado, visto que o valor zero não pertence a nenhum desses intervalos. Com isto, constata-se que tanto a variável ÁREA CONSTRUÍDA como a variável *dummy* LOCALIZAÇÃO contribuem significativamente na explicação da variabilidade do regressor. O valor do coeficiente de determinação, R^2, foi de $0,3931$, ou seja, o modelo 6 explica $39,31\%$ da variação total da variável PREÇO UNITÁRIO. O valor de R^2, embora possa parecer baixo, é estatisticamente significativo, já que o F calculado é altamente significativo, com p-valor próximo de zero.

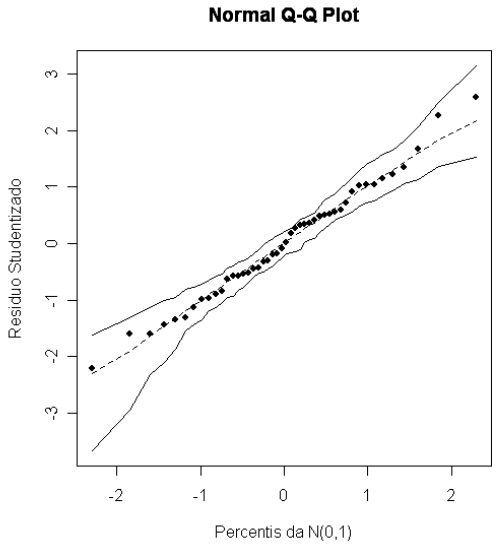
Tabela 3.5: Estimativas dos parâmetros e erros-padrão, valor t e o p-valor para o modelo 6

	Estimativa	Erro Padrão	valor t	p-valor	$IC(\beta_j, 95\%)$
β_0	1345,270	161,40	8,330	0,000	(1022,47 ; 1668,07)
β_1	$-$3,980	0.83	$-$4,790	0,000	($-$5,64 ; $-$2,30)
β_2	85,210	34,08	2.50	0,013	(17,05 ; 153,37)

Objetivando inquirir sobre erros de especificação do modelo, aplicamos o teste RESET, o qual forneceu p-valor igual à $0,0646$, indicando que ao nível de significância de 5%, os parâmetros do modelo são lineares e, portanto, o modelo está bem especificado.

Os testes de Goldfeld-Quandt, de Breusch-Pagan e de Koenker, que avaliam a homoscedasticidade dos erros, forneceram p-valores iguais à $0,9831$, $0,3277$ e $0,2306$, respectivamente. Todos esses testes, portanto, levaram à não rejeição da hipótese nula, evidenciando que os erros são homoscedásticos. Em relação à multicolinearidade, todos os fatores de inflação da variância se situaram abaixo de 5: $1,000048$ para b_2 e $1,000048$ para b_3, ou seja, não há indícios de multicolinearidade. Por fim, o gráfico envelope, Figura 3.3, sugere que os erros têm distribuição normal; esse fato é confirmado pelo teste de Jarque-Bera que forneceu p-valor de $0,7538$, indicando a não-rejeição da hipótese de normalidade dos erros ao nível de significância 5%.

Figura 3.3: Gráfico de Envelope

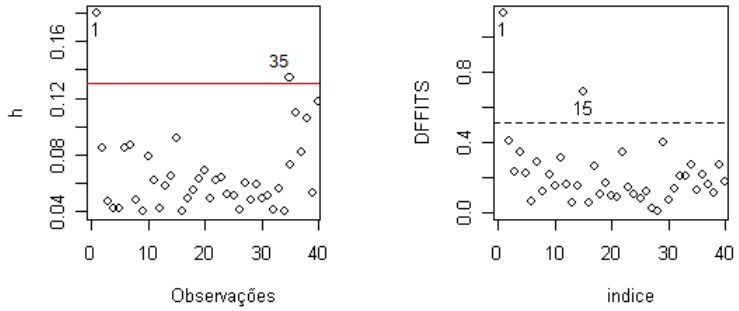


3.3.4 - Análise de Diagnóstico

O teste de Bonferroni indicou que a observação 01 é um *outlier*, com p-valor igual a $0,6136$, ou seja, trata-se de uma observação com erro associado demasiadamente grande. Além disso, conforme apresentado no gráfico de pontos de alavancagem da Figura 3.4, as observações 01 e 35 possuem padrões atípicos de seus regressores. Essas referidas observações apresentam o menor e maior preço unitário dentre os imóveis que compõem a amostra, respectivamente.

No que diz respeito ao gráfico dos DFFITS, as observações 1 e 15 são pontos de influência, ou seja, são capazes de afetar substancialmente as estimativas dos coeficientes da regressão.

Figura 3.4: Gráficos de Pontos de alavancagem e DFFITS



As variações das estimativas entre os modelos ajustados com todas as observações e sem as observações influentes são exibidas na Tabela 3.6, bem como as variações do R^2 e do $\bar{R}^2$. Através desta tabela, observamos que a maior variação nas estimativas se dá quando retiramos a observação 1 isoladamente, enquanto que a retirada da observação 15 muda substancialmente o valor de R^2.

Tabela 3.6: Variação percentual das estimativas dos coeficientes de regressão e valor do $\bar{R}^2$ de acordo com a retirada de observações influenciantes.

Modelos	Observações Retiradas	β_0	β_1	β_2	R^2
Modelo 6.1	1	$12,38$	$21,55$	$14,63$	$26,53$
Modelo 6.2	15	$5,12$	$10,24$	$14,55$	$42,30$
Modelo 6.3	1 e 15	$6,84$	$10,70$	$0,40$	$25,82$

3.3.5 - Modelo Alternativo

Tendo em vista que a observação 01 é altamente influente, uma alternativa seria classificar essa observação como uma variável *dummy*. Sendo assim, definimos a seguinte variável:

$$IN = \begin{cases} 1 & \text{se a } t\text{-ésima observação for a 01} \\ 0 & \text{caso contrário.} \end{cases}$$

Em seguida, ajustamos o modelo 6.4, dado por

$$\text{PU} = \beta_0 + \beta_1 \text{AC} + \beta_2 \text{LOC} + \beta_3 \text{IN} + \epsilon.$$

O sumário das estimativas está apresentado na Tabela 3.7, onde observamos que os p-valores do teste t,indicam que os parâmetros são significativos, fato esse confirmado pelos intervalos de confiança. Do mesmo modo, o p-valor para o teste F, $4,69 \times 10^{-06}$, está abaixo do nível de significância, o que leva à conclusão de que há estrutura de regressão para o

modelo 6.4. O valor do R^2, $0,4763$, e o do $\bar{R}^2$ $0,4389$, teve um aumento considerável em relação ao modelo 6.

Tabela 3.7: Estimativas dos parâmetros e erros-padrão, valor t e o p-valor

	Estimativa	Erro Padrão	valor t	p-valor	$IC(\beta_j, 95\%)$
β_0	1180,060	163,920	7,190	0,000	(852.22 ; 1507,90)
β_1	$-$3,120	0,840	$-$3,690	0,001	($-$4,803 ; $-$1,44)
β_2	72,717	32,344	2,240	0,028	(8,029 ; 137,393)
β_3	257,58	35,770	7,201	0,000	(186,04 ; 329,12)

Com relação as suposições do modelo, utilizando teste de hipóteses ao nível de confiança de 5%, não foi verificada nehuma violação. O teste RESET forneceu p-valor igual a $0,5649$, indicando a não-rejeição da hipótese de que o modelo 6.4 está corretamente especificado. Os testes de Goldfeld-Quandt, de Breusch-Pagan e de Koenker forneceram os respectivos p-valores $0,8248$, $0,751$ e $0,6087$, consequentemente, não rejeitamos a hipóteses de homoscedasticidade dos erros. Todos os fatores de inflação da variância se situaram abaixo de 5: $1,1704$ para b_2, $1,0232$ para b_3 e $1,1930$ para b_4, ou seja, não há indícios de multicolinearidade. Finalmente, o teste de Jarque-Bera com o p-valor de $0,641$, resultou na não-rejeição da hipótese de normalidade dos erros.

3.3.6 - Interpretação dos resultados

Com base no modelo alternativo 6.4, podemos tirar as seguintes conclusões:

- Ratificaram-se as expectativas teóricas de que a área construída influencia significativamente o preço unitário médio de compra e venda, na condição de oferta, de casas residencias no mercado analisado, sendo esta relação dada de forma inversa. Em se tratando da distância à praia, a mesma mostrou-se estatisticamente insignificante. Em relação à localização (inserida ou não em condomínio), verificamos que, de fato, os imóveis situados em condomínios possuem um preço unitário médio superior aqueles que estão isolados. Além disso, na observação 01, o preço unitário médio se dá de maneira ainda mais particular do que nas demais observações.

- O modelo de regressão estimado constata que cerca de $47,63\%$ da variabilidade do preço unitário de venda das unidades é explicada por variações na área construída das 40 casas observadas em função de sua localização e, de uma forma particular, pela observação número 01.

- O conjunto de dados sobre o qual a análise de regressão foi realizada contém apenas 40 observações e 03(três) variáveis independentes. É evidente que outras variáveis podem influenciar decisivamente a estimação do valor de venda do imóvel além daquelas que coletamos e compõem o banco de dados, por exemplo: idade, conservação, padrão construtivo, orientação solar, entre outras. Em virtude disto, alertamos que as previsões realizadas neste trabalho podem ser melhoradas mediante a inclusão de novos fatores e subsequente modelagem.

- A busca incessante da maximização do $\overline{R}^2$ não deve ser o objetivo do avaliador quando lida com análise de regressão, mas, antes, obter estimativas confiáveis dos coeficientes de regressão para a população e fazer inferências estatísticas a respeito deles. O avaliador deve estar mais preocupado com a relevância lógica ou teórica das variáveis explanatórias em relação a variável dependente e sua significância estatística, conforme foi realizado ao longo deste trabalho.

4.0 - Conclusões e recomendações

Este trabalho teve como objetivo apresentar o *software* (gratuito) R e as potencialidades de sua utilização na engenharia de avaliações. Para tanto, apresentamos um tutorial sintético do uso do programa e fizemos uma aplicação com dados reais utilizando técnicas econométricas e testes estatísticos na plataforma R visando à obtenção de um modelo de regressão linear preditivo do mercado imobiliário analisado.

Foram empregados os recursos disponíveis no R para manipulação de dados, criação e exposição de gráficos, uso de funções estatísticas pré-existentes e programação de novas funções.

Verificamos que o Ambiente R pode ser útil para profissionais atuantes na área de avaliações de bens, não apenas em razão de sua flexibilidade e abrangência, mas também pela linguagem acessível e confiabilidade dos resultados.

A utilização do R exige bem mais do que o simples “aperto de alguns botões em série” e, embora muitos interpretem esse fato como uma desvantagem, entendemos que, na verdade, aí se situa a preeminência do R. Para trabalhar com o R é preciso compreender as etapas da modelagem empírica e estar familiarizado com a natureza e estrutura dos dados, a fim de que se possa programar com eficiência e versatilidade.

Por fim, sugerimos e incentivamos a disseminação e o uso deste poderoso *software* de domínio público por engenheiros e arquitetos que buscam empregar metodologia científica em seus trabalhos avaliatórios, capaz de contribuir para uma maior eficiência do desenvolvimento das atividades relacionadas às rotinas de avaliações

Compartilhamos a idéia de que o R é uma demonstração real do poder da colaboração de pesquisadores e usuários em todo o mundo. Cabe a nós interagir, criar e propor novos pacotes e funções que sejam úteis as nossas atividades de avaliação.

5.0 - Apêndice A

```
#################################################################
#  XV COBREAP - CONGRESSO BRASILEIRO DE ENGENHARIA
#  DE AVALIAÇÕES E PERICIAS - IBAPE/SP - 2009}
#
#  UTILIZAÇÃO DO SOFTWARE (GRATUITO) R
#  NA ENGENHARIA DE AVALIAÇÕES}
#
#
#
#
#
#
#  Códigos fonte de programação na plataforma R referentes
#  a análise de regressão
#################################################################

###  PACOTES UTILIZADOS  ###
library(lmtest)   #testes RESET, Goldfeld-Quandt, Breusch-Pagan e Koenker
library(car)      #fatores de inflação da variância e teste Bonferroni
library(tseries)  #teste Jarque-Bera

###  LEITURA DOS DADOS  ###
dir()
dados=read.table(file="portodegalinhas.csv", sep=";", header=T,dec=".")
dados

### VERIFICAÇÃO DE INCONSISTÊNCIAS NOS DADOS ###
is.na(dados) # Verdadeiro se existir dados ausentes
attach(dados)# O R adiciona na memória o objeto que contém os dados
summary(dados)

### GRÁFICOS DE DISPERSÃO, BOX-PLOTS E HISTOGRAMAS ###

par(mfrow=c(1,2))
plot(AC,PU,bty="l",lwd=2,col = "blue",xlab="Área construída (m2)",
     ylab="Preço unitário(R$/m2)")
plot(DP,PU,bty="l",lwd=2,col = "blue",
     xlab="Distância à praia (metros)",ylab="Preço unitário(R$/m2)")

par(mfrow=c(1,2))
plot(AC,DP,bty="l",lwd=2,xlab="Área construída(m2)",
    ylab="Distância à praia (metros)")
plot(LOC,DP,bty="l",lwd=2,xlab="Localiz.(0=casa isolada; 1=casa em condom.)",
    ylab="Distância à praia (metros)")

par(mfrow=c(1,3))
boxplot(PU,main="Preço unitário");boxplot(DP,main="Distância à praia")
boxplot(AC,main="Área construída")
```

```
dados$LOC=factor(dados$LOC)
summary(dados$LOC)

pairs(dados) # Gráficos de dispersão entre todas as variáveis
hist(PU,main="Histograma do Preço unitário", xlab="Valores",
     ylab="Frequência",col="blue",border="black", col.axis="red", prob=F)

###  MEDIDAS DESCRITIVAS  ###

cv=function(x)
{
f=sd(x)/mean(x)
return(f)
}

summary(cbind(PU,DP,AC))
apply(cbind(PU,DP,AC),2,var)
apply(cbind(PU,DP,AC),2,sd)
apply(cbind(PU,DP,AC),2,cv)

### CORRELAÇÕES ###

cor(cbind(PU,DP,AC,LOC))

###  AJUSTE DOS MODELOS CANDIDATOS  ###

# MODELO 1 #

modelo1=lm(PU~AC+DP)
summary(modelo1)

# MODELO 2 #

modelo2=lm(PU~AC)
summary(modelo2)

## Teste de RESET de especificação
resettest(PU ~ AC, power=2:3, type="fitted", data=dados)

# MODELO 3 #

logPU=log(PU);logAC=log(AC);
modelo3=lm(logPU~logAC)
summary(modelo3)

## Teste de Forma Funcional, comparando os modelos 2 e 3.
SSEl=anova(modelo2)[2,2]
SSEll=anova(modelo3)[2,2]
T=length(PU)
logy=log(PU)
```

```
yg=exp((1/T)*sum(logy))
(l= (T/2)*abs(log( (SSE1/yg^2)/SSE11 ) ))
(vc=qchisq(0.95,1))

# MODELO 5 #

modelo5=lm(PU~AC+DP*LOC)
summary(modelo5)

## Teste de RESET de especificação
resettest(PU~AC+DP*LOC, power=2:3, type="fitted",  data=dados)

## Teste de Goldfeld-Quandt
gqtest(modelo5)

## Teste de Koenker
bptest(modelo5, studentize=TRUE)

## Teste de Breusch-Pagau
bptest(modelo5, studentize=FALSE)

## Fatores de inflação da variância
vif(modelo5)

# MODELO 6 #

modelo6=lm(PU~AC+LOC)
summary(modelo6)

## Intervalos de Confiança
confint(modelo6)

## Teste de RESET de especificação
resettest(PU~AC+LOC, power=2:3, type="fitted",  data=dados)

## Teste de Goldfeld-Quandt
gqtest(modelo6)

## Teste de Koenker
bptest(modelo6, studentize=TRUE)

## Teste de Breusch-Pagau
bptest(modelo6, studentize=FALSE)

## Fatores de inflação da variância
vif(modelo6)

## Teste de Jarque-Bera
res<-residuals(modelo6)
jarque.bera.test(res)
```

```
# MODELO 7 #

modelo7=lm(logPU~logAC+LOC) #log(PU)~log(AC)+LOC
summary(modelo7)

## Teste de RESET de especificação
resettest(logPU~logAC+LOC, power=2:3, type="fitted",  data=dados)

## Teste de Goldfeld-Quandt
gqtest(modelo7)

## Teste de Koenker
bptest(modelo7, studentize=TRUE)

## Teste de Breusch-Pagau
bptest(modelo7, studentize=FALSE)

## Fatores de inflação da variância
vif(modelo7)

## Teste de Jarque-Bera
res<-residuals(modelo7)
jarque.bera.test(res)

###  CRITÉRIOS DE INFORMAÇÃO  ###

## AIC
AIC(modelo1)
AIC(modelo2)
AIC(modelo3)
AIC(modelo5)
AIC(modelo6)
AIC(modelo7)

## BIC
AIC(modelo1,k = log(nrow(dados)))
AIC(modelo2,k = log(nrow(dados)))
AIC(modelo3,k = log(nrow(dados)))
AIC(modelo5,k = log(nrow(dados)))
AIC(modelo6,k = log(nrow(dados)))
AIC(modelo7,k = log(nrow(dados)))

###  GRÁFICO ENVELOPE  ###

fit.model<-modelo6
X <- model.matrix(fit.model)
n <- nrow(X)
p <- ncol(X)
H <- X%*%solve(t(X)%*%X)%*%t(X)
h <- diag(H)
```

```
si <- lm.influence(fit.model)$sigma
r <- resid(fit.model)
tsi <- r/(si*sqrt(1-h))
#
ident <- diag(n)
epsilon <- matrix(0,n,100)
e <- matrix(0,n,100)
e1 <- numeric(n)
e2 <- numeric(n)
#
for(i in 1:100){
     epsilon[,i] <- rnorm(n,0,1)
     e[,i] <- (ident - H)%*%epsilon[,i]
     u <- diag(ident - H)
     e[,i] <- e[,i]/sqrt(u)
     e[,i] <- sort(e[,i]) }
#
for(i in 1:n){
     eo <- sort(e[i,])
     e1[i] <- (eo[2]+eo[3])/2
     e2[i] <- (eo[97]+eo[98])/2 }
#
med <- apply(e,1,mean)
faixa <- range(tsi,e1,e2)
#
par(pty="s")
qqnorm(tsi,xlab="Percentis da N(0,1)",
ylab="Residuo Studentizado", ylim=faixa, pch=16)
par(new=T)
qqnorm(e1,axes=F,xlab="",ylab="",type="l",ylim=faixa,lty=1)
par(new=T)
qqnorm(e2,axes=F,xlab="",ylab="", type="l",ylim=faixa,lty=1)
par(new=T)
qqnorm(med,axes=F,xlab="",ylab="",type="l",ylim=faixa,lty=2)

### OBSERVAÇÕES ATÍPICAS ###

# PONTOS DE ALAVANCA #
X=model.matrix(modelo6)
H=X%*%solve(t(X)%*%X)%*%t(X)
h=diag(H)
m=ncol(X)/T
plot(h,xlab="Observações")
abline(h=2*m,col=2)
identify(h,n=2)

# PONTO INFLUENTES #

## DFFITS
k=ncol(X)
```

```
T=length(PU)
lms<-summary(modelo6)
s<-lms$sigma
r<-resid(lms)
ti<-r/(s*sqrt(1-h))
dffits<-abs(ti)*sqrt(h/(1-h))
plot(dffits,xlab="indice",ylab="DFFITS",main="Pontos Influentes")
abline(2*sqrt(k/T),0,lty=2)
identify(dffits,n=2)

## Distância de Cook
rt<-r/(s*sqrt(1-h))
ct<-(rt^2*h)/(p*(1-h))
plot(ct,xlab="indice",ylab="DCook", main="Pontos Influentes")
abline(4/(T-k),0,lty=2)
identify(ct,n=2)

## Teste de Bonferroni
outlier.test(modelo6)

### MODELOS 6 SEM AS OBSERVAÇÕES INFLUENTES ###

# MODELO 6.1 (sem a obs. 1) #

modelo6.1=lm(PU~AC+LOC ,subset=-c(1))
summary(modelo6.1)

# MODELO 6.2 (sem a obs. 15) #

modelo6.2=lm(PU~AC+LOC ,subset=-c(15))
summary(modelo6.2)

# MODELO 6.3 (sem as obs. 1 e 15)#

modelo6.3=lm(PU~AC+LOC ,subset=-c(1,15))
summary(modelo6.3)

###  MODELO ALTERNATIVO  ###

## Dummy para o bservação 1
DADO1=c(1,rep(0,21))

# Modelo 6.4  #

modelo6.4=lm(PU~AC+LOC+DADO1)
summary(modelo6.4)

## Intervalos de Confiança
confint(modelo6.4)
```

```
## Teste de RESET de especificação
resettest(PU~AC+LOC+DADO1, power=2:3, type="fitted",  data=dados)

##Teste de Goldfeld-Quandt
gqtest(modelo6.4)

##Teste de Koenker
bptest(modelo6.4, studentize=TRUE)

##Teste de Breusch-Pagau
bptest(modelo6.4, studentize=FALSE)

## Fatores de inflação da variância
vif(modelo6.4)

## Teste de Jarque-Bera
res<-residuals(modelo6.4)
jarque.bera.test(res)
```

6.0 - Bibliografia consultada

- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT) (2001). *"NBR 14653 - Avaliação de bens - Parte 1: procedimentos gerais"*. Rio de Janeiro.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT) (2004). *"NBR 14653 - Avaliação de bens - Parte 2: imóveis urbanos"*. Rio de Janeiro.
- COOK, R.D. (1977). Detection of Influential Observations in Linear Regression. *Technometrics*, vol. 19, pp. 15-18.
- CORDEIRO, G. M. (2008). Corrected Maximum Likelihood Estimators in Linear Heteroscedastic Regression Models. *Brazilian Review of Econometrics*, v. 28, p. 53-67,.
- CRIBARI NETO, F.; FERRARI S.L.P.; OLIVEIRA, W.A.S.C. (2005). Numerical evaluation of tests based on diferent heteroskedasticity-consistent covariance matrix estimators. *Journal of Statistical Computation and Simulation*, 75, 611-628.
- CRIBARI NETO, F; SOUZA, T. C. ; VASCONCELLOS, K.L. (2007). Inference Under Heteroskedasticity and Leveraged Data. *Communications in Statistics. Theory and Methods*, v. 36, p. 1877-1888.
- CRIBARI NETO, F. & ZARKOS, S.G. (1999). R: yet another econometric programming environment. *Journal of Applied Econometrics*, 14, 319-329.
- DALGAARD, P. (2002). *Introductory Statistics with R*. New York: Springer-Verlag.
- DANTAS, R. A (2005). *Engenharia de Avaliações: uma introdução à metodologia científica*, Pini, São Paulo.
- FERREIRA, D. F (2007). *Estatística computacional utilizando o R*. Notas de aula.
- GUJARATI, D.N. (2006). *Basic Econometrics*, 4^{a} ed. Nova York: McGraw-Hill.
- IHAKA, R. e GENTLEMAN, R. (1996). R: a language for data analysis and graphics. *Journal of Computational Graphics and Statistics*, 5:299-314.
- INSTITUTO BRASILEIRO DE AVALIAÇÕES E PERÍCIAS DE ENGENHARIA (IBAPE/SP) (2005). *"Norma para avaliação de imóveis urbanos"*. São Paulo.
- JARQUE, C. M. e BERA, A. K. (1980). Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics Letters*, v. 6, p. 255−259.
- McCULLOUGH, B. D. & Wilson, B. (2005). On the accuracy of statistical procedures in Microsoft Excel 2003. *Computational Statistics and Data Analysis*,49(4), 1244-1252.
- MURRELL, P. (2005). *R Graphics*. New York: Chapman & Hall/CRC.
- MONTGOMERY, D. C.; Runger, G. C. *Applied statistics and probability for engineers*. New York: John Wiley & Sons, 1994. 1004p.
- 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, URL http://www.R-project.org.

- SPANOS, A. (1999). *Probability, Theory and Statistical Inference: Econometric Modeling with Observational Data*. Reino Unido: Cambridge University Press, p. 21.
- STIGLER, S. M. (1987). “Testing Hypothesis or Fitting Models? Another Look at Mass Extinctions”. In: Nitecki, Matthew H.; Hoffman Antoni. *Neutral Models in Biology*. Oxford: Oxford University Press, p.148.
- ZENI, A. M (1990). “Curso de Métodos Matemáticos e Estatísticos na Engenharia de Avaliações”. *Anais do VI COBREAP*, Belo Horizonte.
- ZENI, A. M (1996). *“Curso Básico de Engenharia de Avaliações - Metodologia Científica”*. ABDE.