Mergulhando no R & K³ Parte 2

2.3. Equações e inequações

Equações são definidas por sua igualdade (=) relacionada a uma ou mais incógnitas. Já as inequações, são sentenças matemáticas expressas utilizando uma desigualdade, que relaciona uma ou mais variáveis. Os operadores das inequações são:

Diferente de
Menor que
Maior que
Menor ou igual a
Maior ou igual a

Tomemos o exemplo a seguir. Dois processadores com diferentes arquiteturas sendo comparados, entre si, pelo seu desempenho computacional global.

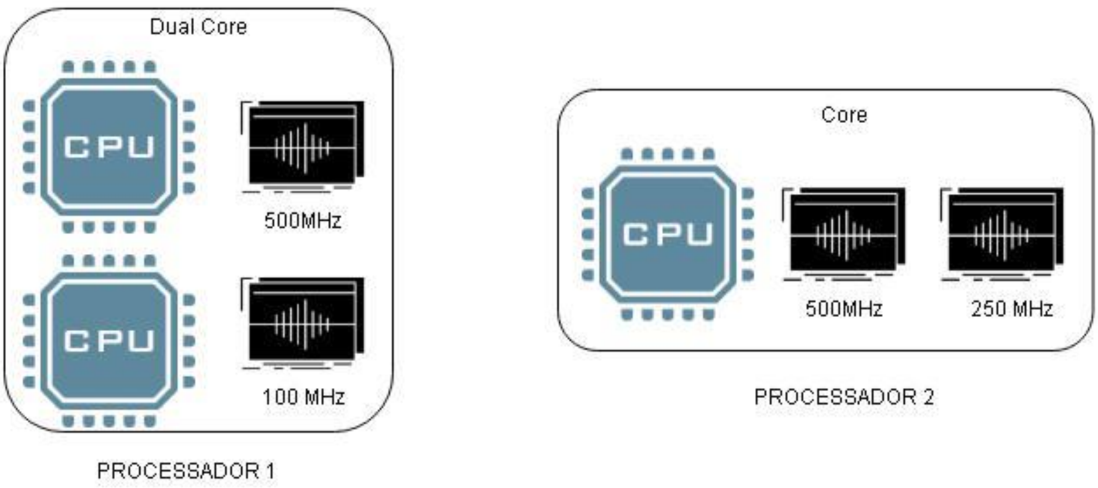


Figura 1 – Exemplos de arquiteturas computacionais conhecidas (fonte: autoral)

Dados os processadores 1 e 2 da figura 1, o primeiro se caracteriza por ter dois núcleos, cada qual trabalhando com frequências distintas, medidas em determinado processo, com 100 MHz, CPU 1 e, 500 Mhz para a CPU2. O processador 2, tem uma única CPU, que escalona processos entre 500MHz e 250MHz.

A velocidade das CPUs é dada pela frequência de seu processamento, medida em MHz, Mega Hertz.

Podemos montar um pequeno sistema de equações abstraindo da figura 1.

Atribuímos a CPU a varável x, dada como frequência total do processador.

Então:

Processador 1 = x + x + 500 +100

Processador 2 = x + 500 + 250

Queremos uma equação de comparação, ou seja, igualdade.

Mergulhando no R & K³ Parte 2

Se, **Processador 1 = Processador 2**, é verdadeiro?

Testando o modelo da equação.

$$x + x + 500 + 100 = x + 500 + 250$$

2x +600 = x + 750 ➔ 2x – x = 750 – 600 → x = 150

Então para que os processadores 1 e 2 tenham o mesmo desempenho, o processador 1 pode aumentar em até 300 MHz o seu desempenho e o Processador 2 pode aumentar somente 150 MHz. Isto, quer dizer, que o processador 1 pode trabalhar mais e o processador 2 já está próximo do seu limite.

O desempenho total de cada processador é de 900 MHz, o processador 1 pode aumentar até **33,3 %** e o processador 2 já está trabalhando em **83,3%.**

Você pode explicar esses resultados????

Teste parte da equação no R

```
> x <- 150
> 2*x+600==x+750
[1] TRUE
```

Agora vamos pensar um caso de que não sabemos a frequência do processador 1, veja a figura 2.

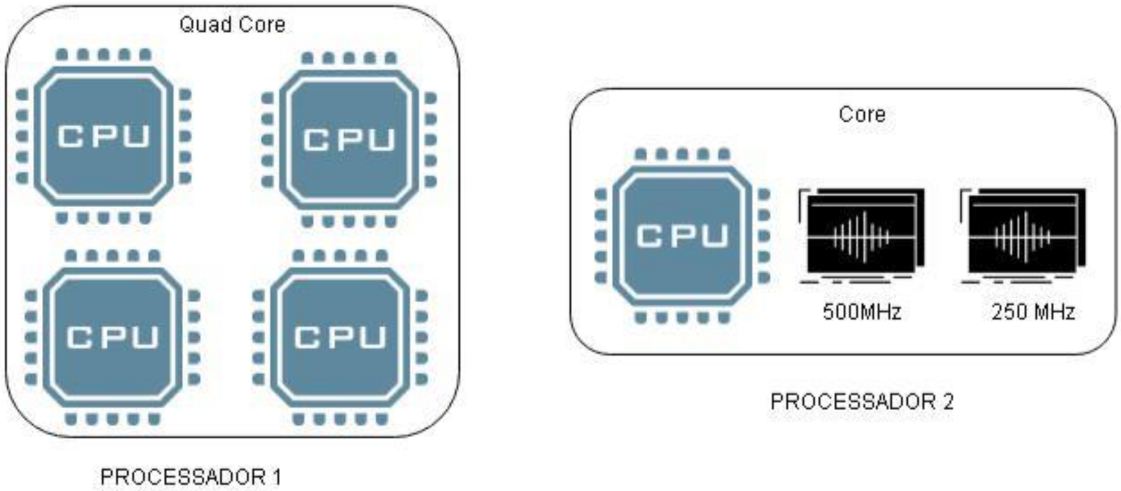


Figura 2 – Exemplos de arquiteturas computacionais, uma frequência indeterminada (fonte:autoral)

Temos além da incógnita da CPU, nenhuma referência em frequência do processador 1, mesmo que parcial, para comparar ou igualar com o processador 2.

Então vamos montar o sistema de inequações.

Processador1 = x +x +x +x

Processador2 = x + 500 + 250

Vamos testar possibilidades

Processador 1 pode ter frequência maior que Processador 2 e vice-versa.

Mergulhando no R & K^3 Parte 2

Caso A:

processador1<processador2

x +x +x +x < x + 500 + 250

4*x – x <750

3x <750

x< 750/3

x<250 MHz

Caso B:

processador1>processador2

x +x +x +x > x + 500 + 250

4*x – x >750

3x >750

x> 750/3

x>250 MHz

Então o sistema tem duas respostas,

Se o processador 1 tem desempenho menor, x <250MHz

Se o processador 1 tem maior desempenho que o 2, x >250MHz

Vamos a um exemplo de solução de equações no R:

Dados 3 processadores com a seguinte arquitetura, da figura 3.

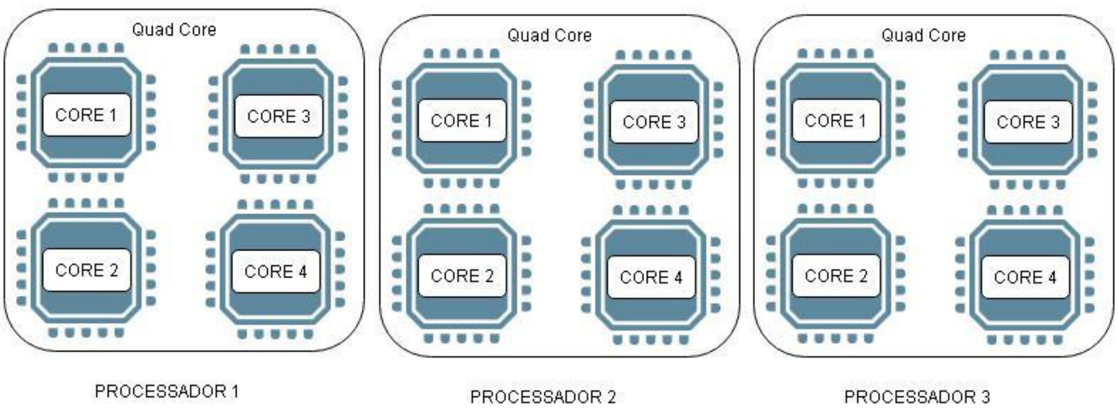


Figura 3 – Exemplos de arquiteturas computacionais, frequências indeterminadas(fonte:autoral)

Mergulhando no R & K³ Parte 2

Foi mensurado o desempenho de três cores para 3 processos em threads, cada threads rodando em um core. Sabe-se que o desempenho máximo dos 3 cores (core de processador) com 3 threads pode chegar a 750 MHz. O padrão de funcionamento médio de 3 threads é em torno de 500Mhz. E o desempenho mínimo, quando os processos estão finalizando suas threads opera com 250 MHz.

Processador 1 foi testado com o máximo de frequência, o processador 2 foi utilizado com uma frequência média na execução das threads e o processador 3 teve um desempenho mínimo no uso da frequência de seus cores.

O seguinte sistema foi **medido** e apontado a seguir:

Processador 1 → -1*core1 + 2*core2 – 1*core3 = 750
Processador 2 → 2*core1 - 2*core2 + 4*core3 = 500
Processador 3 → -1*core1 + 0.5*core2 + 1*core3 = 250

Sistema linear:

$$\begin{pmatrix} -1 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0{,}5 & 1 \end{pmatrix} * \begin{pmatrix} c1 \\ c2 \\ c3 \end{pmatrix} = \begin{pmatrix} 750 \\ 500 \\ 250 \end{pmatrix}$$

Fórmula geral → A*x =B

A é a matriz determinante com os coeficientes, b é a matriz dos termos independentes. E c1,c2 e c3 são as incógnitas do sistema. Onde A é a Matriz_cores. X são as incógnitas (c1,c2 e c3), e b são os resultados (750,500,250)

Em linguagem R, vamos montar a matriz A da fórmula geral, denominando-a de Matriz_cores, b é a matriz dos termos independentes. Usaremos o comando **solve** para resolver o sistema. Documentação: https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/solve

```
> matriz_cores <- matrix(c(-1,2,-1,2,-2,0.5,-1,4,1), nrow = 3, ncol = 3)
> matriz_cores
     [,1] [,2] [,3]
[1,]   -1  2.0   -1
[2,]    2 -2.0    4
[3,]   -1  0.5    1

> b <- matrix(c(750,500,250), ncol = 1)
> b
     [,1]
[1,]  750
[2,]  500
[3,]  250

> solve(matriz_cores,b)
         [,1]
[1,] 392.8571
[2,] 714.2857
[3,] 285.7143

> resultamat1 <- round(solve(matriz_cores,b),1)
> resultamat1
      [,1]
[1,] 392.9
[2,] 714.3
```


SÃO PAULO TECH SCHOOL

```
[3,] 285.7
```

Então, atribuir

```
> core1 <- resultamat1[1]
> core2 <- resultamat1[2]
> core3 <- resultamat1[3]
```

Vamos substituir os valores encontrados no sistema, lembrando que core1 = c1, core2=c2, core3=c3.

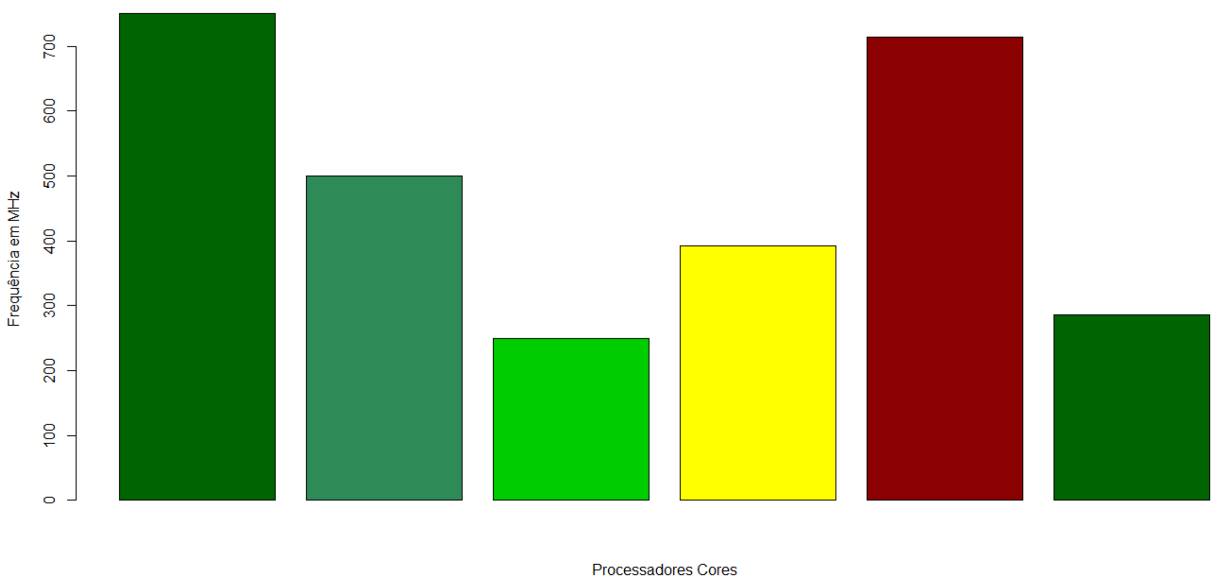
```
> proc1 <- -1*core1 + 2*core2 -1*core3
> proc1
[1] 750

> proc2 <- 2*core1 - 2*core2 + 4*core3
> proc2
[1] 500

> proc3 <- -1*core1 + 0.5*core2 + 1*core3
> proc3
[1] 249.95

> graficos <- c(proc1, proc2, proc3, resultamat1)
> graficos
[1] 750.00 500.00 249.95 392.90 714.30 285.70

> barplot(graficos, xlab = "Processadores Cores",
+ ylab = "Frequência em MHz",
+ col = c("darkgreen","seagreen","green3","yellow","red4"))
```



Exercícios de modelagem aplicada a PI

Seja o conjunto de dados obtidos pelo OPHM, assimile algumas modelagens desses dados.

Mergulhando no R & K3 Parte 2

OpenHardwareMonitorLog.2022.10.25

Filter

	X.Time	X..CPU.Core..1..	X..CPU.Core..2..	X..CPU.Total..	X..CPU.Core..1...1	X..CPU.Core..2...1	X..CPU.Package..
1	10/25/2022 16:39:58	3.90625	3.90625	3.906250	37	38	41
2	10/25/2022 16:39:59	1.56250	4.68750	3.125000	37	39	40
3	10/25/2022 16:40:00	9.37500	8.59375	8.984375	38	40	40
4	10/25/2022 16:40:01	2.34375	1.56250	1.953125	38	39	40
5	10/25/2022 16:40:02	6.25000	1.56250	3.906250	37	38	40
6	10/25/2022 16:40:03	2.34375	3.90625	3.125000	36	38	40
7	10/25/2022 16:40:04	8.59375	6.25000	7.421875	37	39	40
8	10/25/2022 16:40:05	1.56250	3.12500	2.343750	36	37	40
9	10/25/2022 16:40:06	1.56250	3.12500	2.343750	36	37	40
10	10/25/2022 16:40:07	3.12500	3.12500	3.125000	37	39	39
11	10/25/2022 16:40:08	3.90625	3.90625	3.906250	37	38	40
12	10/25/2022 16:40:09	1.56250	7.03125	4.296875	37	37	40
13	10/25/2022 16:40:10	7.03125	6.25000	6.640625	37	39	40

Showing 1 to 13 of 3,530 entries, 19 total columns

Ajustando as colunas

```
> dados <- data.frame(OpenHardwareMonitorLog.2022.10.25)
> colnames(dados)
 [1] "X.Time"                "X..CPU.Core..1.."      "X..CPU.Core..2.."
 [4] "X..CPU.Total.."        "X..CPU.Core..1...1"    "X..CPU.Core..2...1"
 [7] "X..CPU.Package.."      "X..CPU.Core..1...2"    "X..CPU.Core..2...2"
[10] "X..CPU.Package...1"    "X..CPU.Cores.."        "X..CPU.Graphics.."
[13] "X..CPU.DRAM.."         "X..Bus.Speed.."        "X..Memory.."
[16] "X..Used.Memory.."      "X..Available.Memory.." "X..Temperature.."
[19] "X..Used.Space..."
```

```
> colnames(dados) <- c("hora","cpuCorePerc1","cpuCorePerc2","cpuTotalPerc","cpuCoreTemp1","cpuCoreTemp2", "cpuPackageTemp","cpuCoreFreq1","cpuCoreFreq2","cpuPackage1","cpuCoresPerc","cpuGraphs","cpuDRAM","busSpeed","memory","usedMemory","availMemory","ttemperature","usdeSpace")
```

```
> View(dados)
```

dados

Filter

	hora	cpuCorePerc1	cpuCorePerc2	cpuTotalPerc	cpuCoreTemp1	cpuCoreTemp2	cpuPackageTemp
1	10/25/2022 16:39:58	3.90625	3.90625	3.906250	37	38	41
2	10/25/2022 16:39:59	1.56250	4.68750	3.125000	37	39	40
3	10/25/2022 16:40:00	9.37500	8.59375	8.984375	38	40	40
4	10/25/2022 16:40:01	2.34375	1.56250	1.953125	38	39	40
5	10/25/2022 16:40:02	6.25000	1.56250	3.906250	37	38	40
6	10/25/2022 16:40:03	2.34375	3.90625	3.125000	36	38	40

Showing 1 to 6 of 3,530 entries, 19 total columns

```
>id <- seq_along(dados[,1])
> dados_trusted <- data.frame(id, dados)
> View(dados_trusted)
```

Mergulhando no R & K³ Parte 2

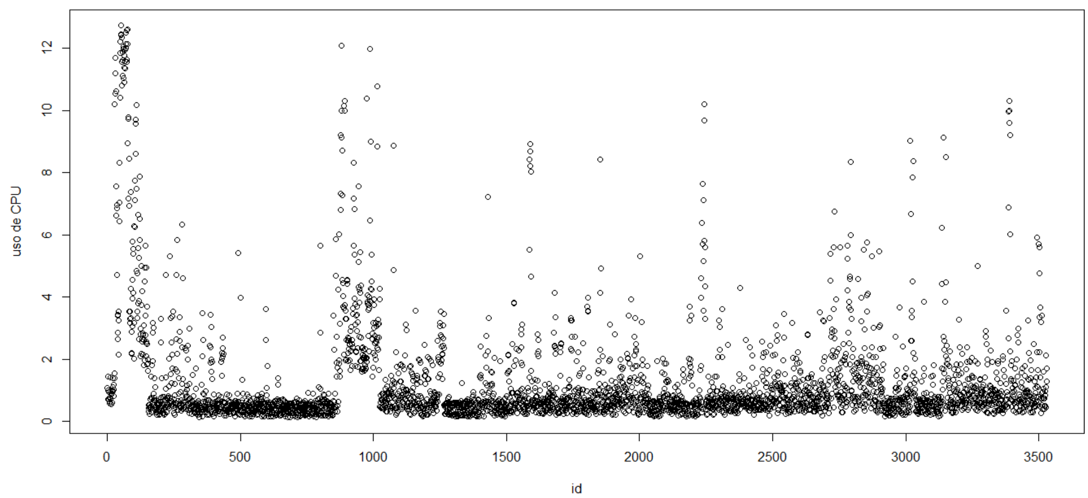
dados × | dados_trusted ×

Filter

	id	hora	cpuCorePerc1	cpuCorePerc2	cpuTotalPerc	cpuCoreTemp1	cpuCoreTemp2	cpuPackageTemp	cp
1	1	10/25/2022 16:39:58	3.90625	3.90625	3.906250	37	38	41	
2	2	10/25/2022 16:39:59	1.56250	4.68750	3.125000	37	39	40	
3	3	10/25/2022 16:40:00	9.37500	8.59375	8.984375	38	40	40	
4	4	10/25/2022 16:40:01	2.34375	1.56250	1.953125	38	39	40	
5	5	10/25/2022 16:40:02	6.25000	1.56250	3.906250	37	38	40	
6	6	10/25/2022 16:40:03	2.34375	3.90625	3.125000	36	38	40	

Showing 1 to 6 of 3,530 entries, 20 total columns

```
> plot(dados_trusted$id,dados_trusted$cpuCoresPerc, xlab = "id", ylab = "uso de CPU")
```



Qual modelo matemático explicaria este conjunto de dados? Recorde os modelos apresentados.

Vamos reduzir os dados a um modelo linear.

```
> modelo1 <- lm(dados_trusted$cpuCoresPerc~dados_trusted$id)
```

A função `lm` é a ***linear model***, modelo linear. Um modelo linear é uma função que aproxima os dados de uma função de primeiro grau.

Vamos testar o modelo (já começamos a "pensar" como uma **ML (Machine Learning)).**

```
> summary(modelo1)
```

```
Call:
lm(formula = dados_trusted$cpuCoresPerc ~ dados_trusted$id)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.5478 -0.9192 -0.5183  0.1674 10.9464 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)       1.785e+00  5.941e-02   30.04   <2e-16 ***
dados_trusted$id -2.719e-04  2.914e-05   -9.33   <2e-16 ***
```

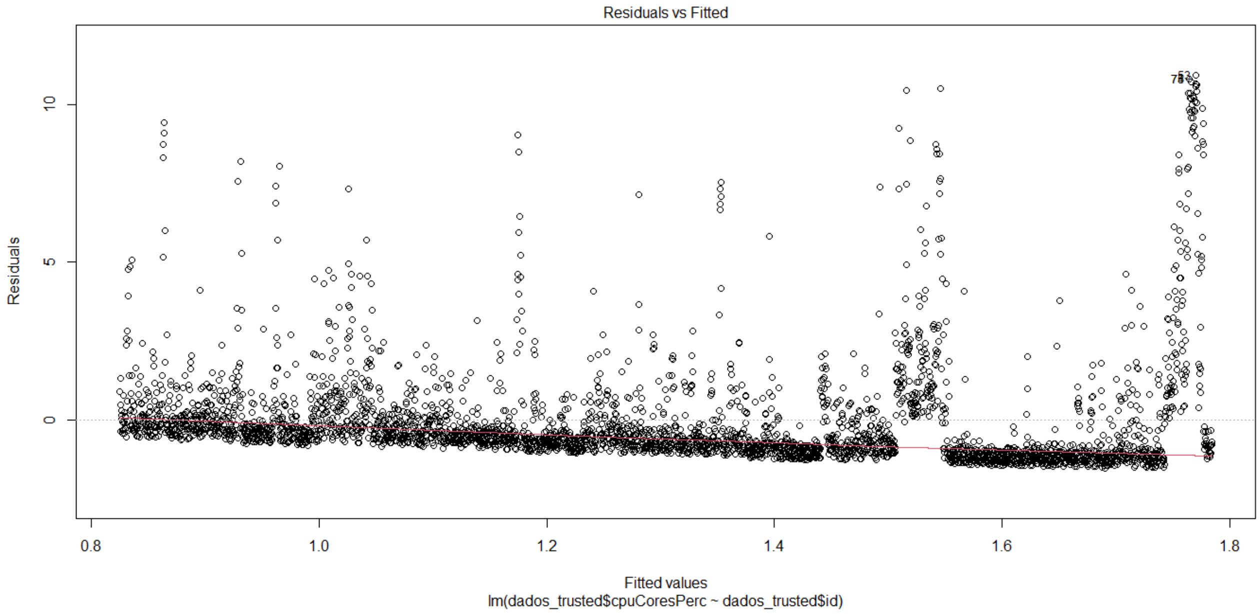
Mergulhando no R & K3 Parte 2

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.765 on 3528 degrees of freedom
Multiple R-squared:  0.02408,	Adjusted R-squared:  0.02381
F-statistic: 87.06 on 1 and 3528 DF,  p-value: < 2.2e-16
```

```
> plot(modelo1)
Hit <Return> to see next plot:
Hit <Return> to see next plot:
Hit <Return> to see next plot:
Hit <Return> to see next plot:
```

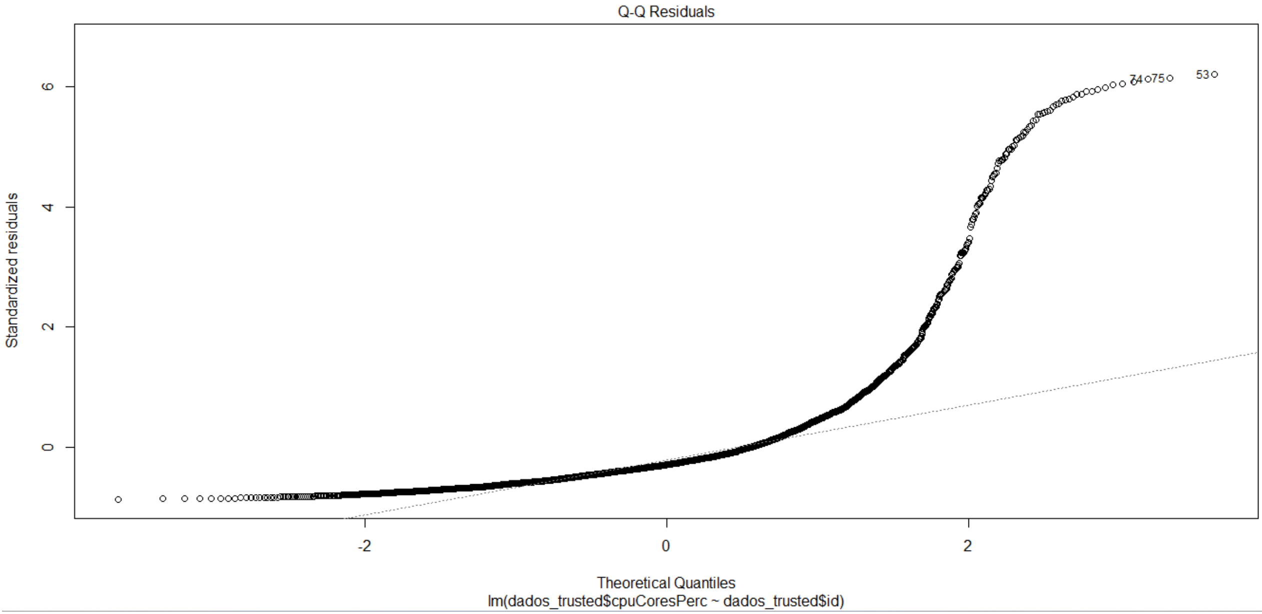
Aperte <Enter> para ver o próximo gráfico:

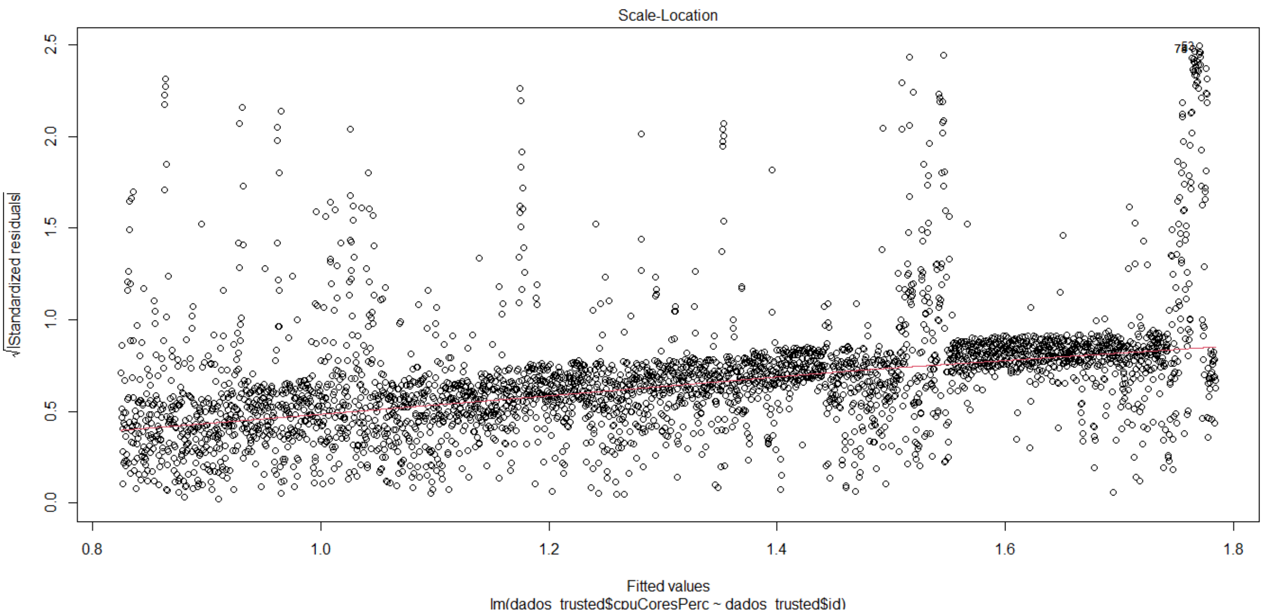


Ao conduzir uma análise **residual**, um " **gráfico de resíduos versus** ajustes " é o **gráfico** criado com mais frequência. É um **gráfico** de dispersão de **resíduos** no eixo y e valores **ajustados** (respostas estimadas) no eixo x. O **gráfico** é usado para detectar não linearidade, variâncias de erro desiguais e outliers.

O modelo QQ existe para testar a normalidade que se aproxima de uma reta. O gráfico quantil-quantil ou qq-plot, proposto por Wilk & Gnanadesikan (1968), é um dispositivo gráfico exploratório utilizado para verificar a validade de uma pressuposta distribuição para um conjunto de dados. Em geral, a ideia básica é a de calcular o valor teoricamente esperado para cada ponto de dados com base na distribuição em questão. Se os dados de fato seguirem a distribuição assumida os pontos deste gráfico formarão aproximadamente uma linha reta.

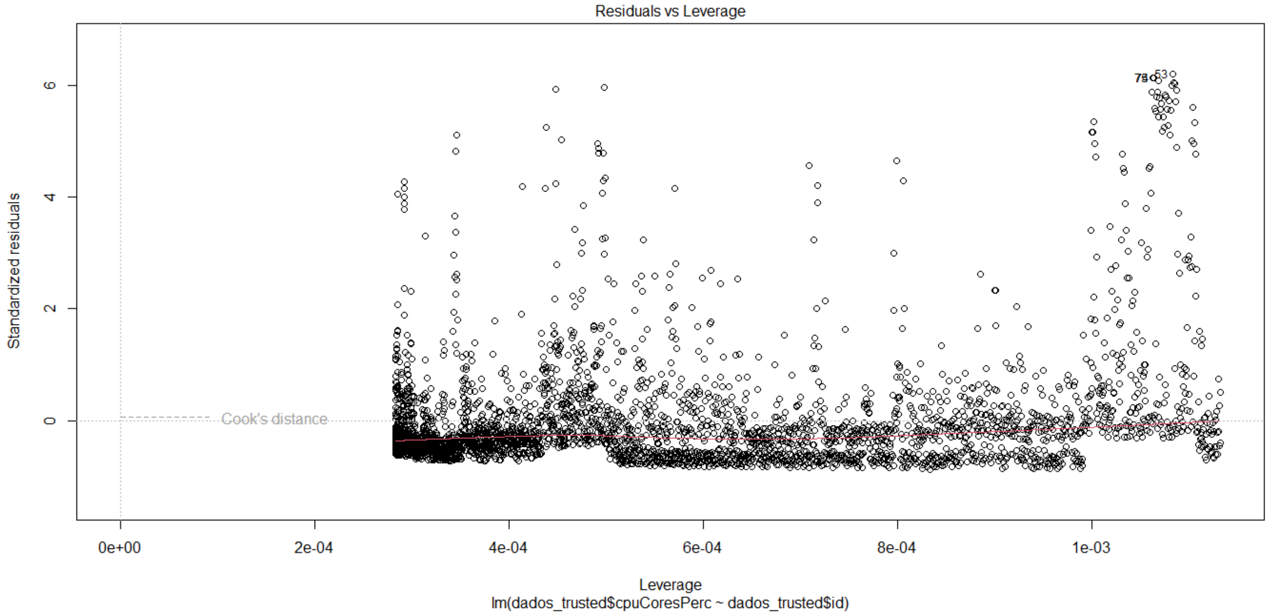
Mergulhando no R & K³ Parte 2





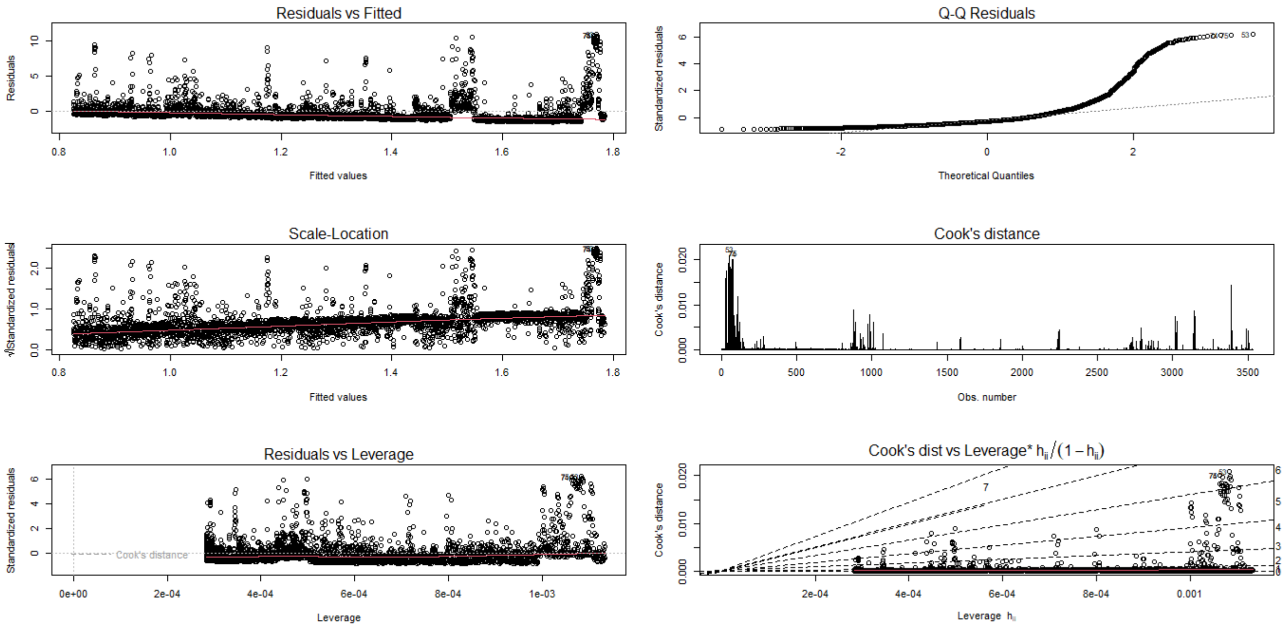
O gráfico Escala-Localização mostra se os resíduos são distribuídos igualmente ao longo dos intervalos de variáveis de entrada (preditor). A suposição de variância igual (homoscedasticidade) também pode ser verificada com este gráfico. Se virmos uma linha horizontal com pontos espalhados aleatoriamente, significa que o modelo é bom.

Mergulhando no R & K³ Parte 2

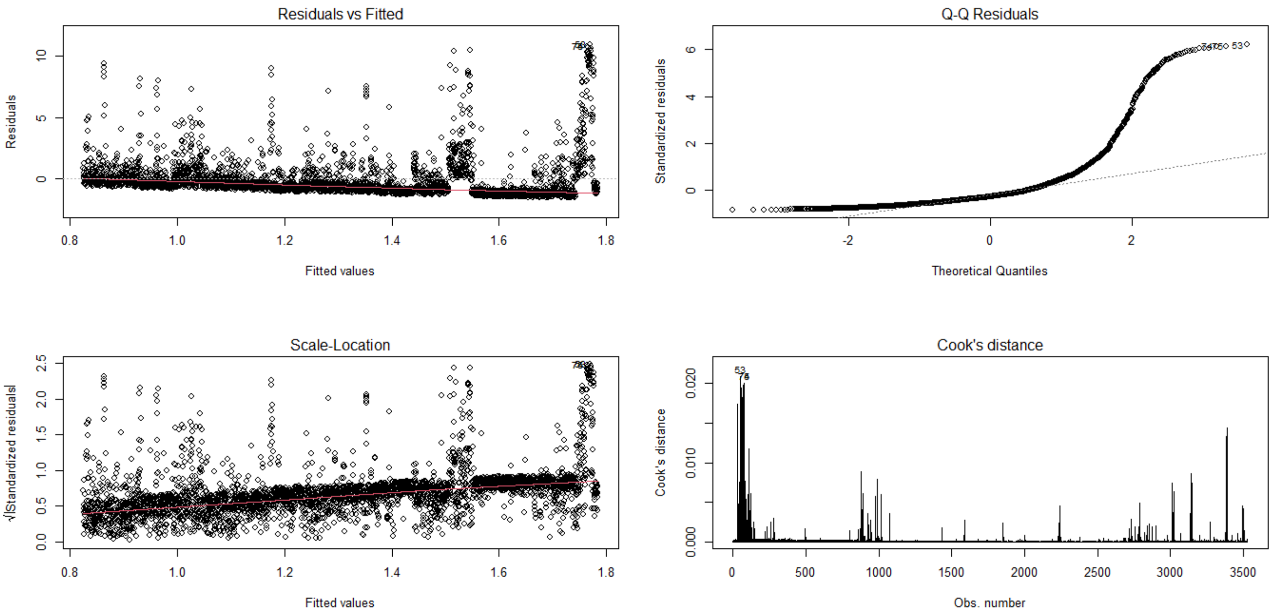


Os gráficos de Residuais vs. Alavancagem ajudam a identificar pontos de dados influentes em seu modelo. Os valores discrepantes podem ser influentes, embora não sejam necessários e alguns pontos dentro de uma faixa normal em seu modelo podem ser muito influentes. Os pontos que estamos procurando (ou não) são valores nos cantos superior direito ou inferior direito, que estão fora da linha de distância de Cook tracejada em vermelho. Esses são pontos que teriam influência no modelo e removê-los provavelmente alteraria visivelmente os resultados da regressão.

```
> model1b <- lm(cpuCoresPerc~id, dados_trusted)
> par(mfrow=c(3,2))
> plot(model1b, which = 1:6)
```




```
> par(mfrow=c(2,2))
> plot(model1b, which = 1:4)
```


Residuals vs Fitted
Residuals
Fitted values
Q-Q Residuals
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Cook's distance
Cook's distance
Obs. number

2.4. Matemática financeira básica

Vale lembrar aqui uma premissa muito importante.

O VALOR DO DINHEIRO NO TEMPO MUDA

Para esta premissa, quando comparamos duas quantias, precisamos equipará-las a uma data (date) base.

Alguns conceitos:

Juros → é o rendimento obtido ou pago quando alguma pessoa aplica ou toma emprestado determinado valor em dinheiro, com custos financeiros embutidos.

Taxa de juros → É um coeficiente aplicado como valor de juros por um período determinado ou atrelado a um contrato. Essa taxa tem por objetivo recuperar o risco envolvido no empréstimo ou financiamento, sobre o poder a degradação do poder de compra ou a efetiva inadimplência.

Como diferencio um e outro.

Taxa de juros é um coeficiente com uma unidade entre 0 a 100%, e o juros é o valor calculado em quantia de dinheiro. Taxa é o percentual e juros é o valor em dinheiro.

A taxa de juros pode ser unitária ou percentual.

Taxa unitária → reflete o valor dos juros para cada unidade do capital

Se, Tu = juros/Capital = R$ 10.00/R$ 100.00 = 0,1 (não tem unidade de medida)

Taxa percentual → reflete o valor o valor dos juros para cada cento

Se, Tp = (juros/Capital) *100 = (R$10.00/R$100.00) *100 = 10 %

Faça em R

```
juros <- 10.00

capital <- 100.00

tu <- juros/capital

tu

[1] 0.1

tp <- (juros/capital)*100

tp

[1] 10
```

Juros simples

Exemplo de taxa a 10%

Mês 1 - Inicial		Mês 2 - Incide Juros		Mês 3 - Incide Juros		Mês 4 - Incide Juros	
R$	100,00	R$	100,00	R$	100,00	R$	100,00
		R$	10,00	R$	10,00	R$	10,00
				R$	10,00	R$	10,00
						R$	10,00
R$	**100,00**	**R$**	**110,00**	**R$**	**120,00**	**R$**	**130,00**

Em R:

```
> capital <- 100
> juros <- 10
> meses <- 3
> jurosSimples <- capital + juros*meses
> jurosSimples
[1] 130
```

Fórmula Financeira	
Juros	**Montante final**
J = C * i *t J = juros C =capital i = taxa de juros t = tempo de aplicação(mês, bimestre, trimestre, semestre, ano....)	M=C + J M= montante final C = capital J = juros

```
> taxaJuros <- 0.1
> juros <- capital*taxaJuros*meses
> juros
[1] 30
```

Exemplo: Qual o valor do montante pago, por um financiamento de um computador de R$1500,00, em 6 parcelas, aplicado no regime de juros simples a uma taxa mensal de 1,3 % durante 6 meses?

Quanto o fica o valor da parcela e qual o montante pago ao final da 6ª parcela paga.

Capital	1500.00
Taxa de juros i	i = 1.3% = 1.3/100 = 0.013 ao mês (a.m.)
Tempo t	t = 6 meses
Juros	J = 1500.00 *0.013 *6 = 117.00 em reais
Montante M	M = C + j = 1500,00 + 117.00 = 1617,00 em reais

```
> valorFinanciado <- 1500
> txJuros <- 0.013
> tempo <- 6
> juros <- valorFinanciado*txJuros*tempo
> juros
[1] 117
> montante <- valorFinanciado + juros
> montante
```

```
[1] 1617
> parcelas <- montante/6
> parcelas
[1] 269.5
```

Juros compostos

Exemplo de taxa a 10%

Mês 1 - Inicial		Mês 2 - Incide Juros		Mês 3 - Incide Juros sobre juros		Mês 4 - Incide juros sobre juros	
R$	100,00	R$	100,00	R$	100,00	R$	100,00
		R$	10,00	R$	10,00	R$	10,00
				R$	10,00	R$	21,00
				R$	1,00	R$	2,10
R$	**100,00**	**R$**	**110,00**	**R$**	**121,00**	**R$**	**133,10**

Em R:

Pense como você faria um programa em R que se calcula os juros compostos.

Pense que a capitalização composta ou capitalização a juros compostos é a aplicação de um determinado capital, a uma taxa que gera, em um período, em n dias ou meses ou anos ou etc., um juro em cima do montante obtido no período imediatamente anterior.

M = C * (1+i%)*(1+i%)*(1+i%) ... (1+i%) , ou seja, n vezes.

M = C*(1 +i%)n**

M = montante final
C = capital
i = taxa de juros
n = 3 meses de incidência juros e juros sobre juros

Colocando os valores do exemplo anterior na fórmula:

```
> capital <- 100
> i <- 0.1
> n <- 3
> montante <- capital*((1+i)**n)
> montante
[1] 133.1
```

Mergulhando no R & K^3 Parte 2

Você assimilou que existe uma regra entre o capital e a forma como esse capital circula, na forma de mercadoria ou na forma de dinheiro mesmo, ou até mercado de capitais, ativos, imóveis alugados, outros.

Se você compra um bem, alguém vai acreditar que você poderá pagar esse bem, com um risco que pode variar de acordo com seu perfil, risco baixo à alto.

O mercado sabe quem você é, hoje suas informações estão disponíveis em todas as instituições financeiras e de crédito.

O mercado financeiro tem como função intermediar as relações entre quem poupa ou que toma capital, considerando, os riscos, prazos etc.

Como funciona:

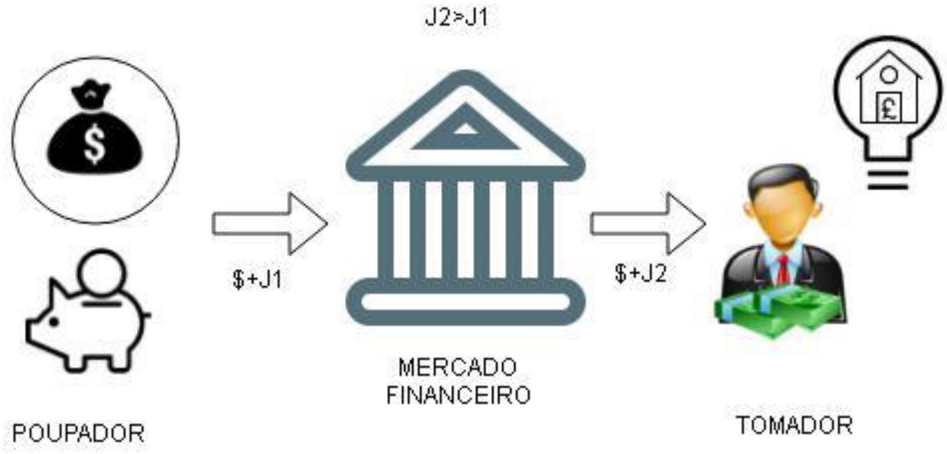


Figura 4 - Spread – Mercado Financeiro – Fonte autoral

Spread refere-se à diferença entre o preço de compra e venda de uma ação, título ou transação monetária. Analogamente, quando o banco empresta dinheiro a alguém, cobra uma taxa pelo empréstimo — uma taxa que será certamente superior à taxa de captação. A diferença entre as duas taxas é o chamado spread bancário.