Estudo de Caso 3 - Comparação de desempenho de duas configurações de um algoritmo de otimização

Ana Júlia de Lima Martins, Antônio Carlos da Anunciação, Melchior Augusto Syrio de Melo

24 de dezembro de 2024

Resumo

Este relatório descreve uma análise estatística do desempenho de duas configurações distintas de um algoritmo de otimização baseado em evolução diferencial (DE). Neste experimento, aplicou-se um Experimento Completamente Aleatorizado com Blocos (RCBD) com o intuito de avaliar o desempenho das duas configurações em diferentes instâncias do problema. A análise teve como objetivo identificar se há diferenças estatisticamente significativas no desempenho médio das configurações, determinando qual delas apresenta o melhor desempenho médio e avaliando a magnitude das diferenças encontradas. Por fim, os resultados são apresentados, proporcionando recomendações baseadas nos achados experimentais.

1. Design do Experimento

A análise será conduzida a partir de um Experimento Completamente Aleatorizado com Blocos (RCBD), em que as instâncias do algoritmo são consideradas os blocos, e as configurações do algoritmo representam os tratamentos (níveis do fator de interesse). O objetivo é verificar se há diferenças estatisticamente significativas no desempenho médio entre as configurações do algoritmo, levando em conta a variabilidade atribuída às instâncias (blocos).

1.1. Hipótese

Para esse experimento, estamos interessados em investigar se há alguma diferença no desempenho médio do algoritmo quando equipado com diferentes configurações, para a classe de problemas de interesse. Portanto, para realizar a análise estatística, estabelecemos as seguintes hipóteses:

- **Hipótese Nula (H_0)**: Não há diferença no desempenho médio entre as configurações, isto é, não existe diferença no tamanho dos efeitos τ_i.

$$\Big\{ H_0 : \tau_i = 0, \ \forall i \in \{1, 2, ..., a\}$$

- **Hipótese Alternativa (H_a)**: Existe pelo menos uma configuração que apresenta um efeito significativamente diferente de zero, ou seja, com um desempenho superior.

$$\Big\{ H_1 : \exists \ \tau_i \neq 0$$

Se decidirmos pela rejeição da hipótese nula e as premissas do teste forem validadas, precisaremos determinar qual configuração em termos de desempenho médio. Para responder a essa pergunta, pode-se realizar uma comparação todos contra todos, utilizando o teste de Tukey devido a sua sensibilidade superior ao fazer esse tipo de comparação.

1.2. Tamanho amostral

Nesta seção, discutimos a definição do número de instâncias e de repetições necessárias para o experimento, a fim de obter um poder do teste de pelo menos $\pi^* = 0,8$, para detectar diferenças iguais ou superiores a um tamanho de efeito minimamente relevante $d^* = 0,5$, com nível de significância $\alpha = 0,05$.

1.2.1. Estimando o número de instâncias (blocos)

Sob a hipótese alternativa H1, a estatística t_0 segue uma distribuição t não central (Mathews, 2010) com parâmetro de não centralidade dado por:

$$ncp = \frac{(\mu_D - \mu_0)\sqrt{N}}{\hat{\sigma}_\Phi} = \frac{\delta\sqrt{N}}{\hat{\sigma}_\Phi} = d\sqrt{N}.$$

em que N é o número de instâncias necessárias.

Assumindo uma medida de relevância mínima $d^* = |\delta^*|/\sigma_\Phi$, o poder do teste π^* é dado pela integral da distribuição t não central com $ncp^* = d^*\sqrt{N}$ sobre os valores de t_0 para os quais a hipótese nula H_0 é rejeitada (Campelo et. al., 2019):

$$\pi^* = 1 - \beta^* = 1 - \int_{t=t_{\alpha/2}^{(N-1)}}^{t_{1-\alpha/2}^{(N-1)}} \left[t_{|\,\mathrm{ncp}^*|}^{(N-1)}\right] dt$$

Finalmente, o número de instâncias pode ser calculado como o menor inteiro N tal que o poder do teste π^* seja igual ou maior que o poder desejado. No caso da hipótese alternativa unilateral, temos que:

$$N^* = \min N \left| t_{1-\alpha}^{(N-1)} \leq t_{\beta^*\,;\,|\mathrm{ncp}^*|}^{(N-1)} \right.$$

```
d = 0.5         # Mínima diferença de importância prática (padronizada)
alpha <- 0.05   # Nível de significância
p <- 1 - alpha # Nível de confiança

n <- 2
power <- 0
while (power < 0.8)
{
  df <- n - 1          # n - 1 graus de liberdade
  ncp <- d * sqrt(n)   # Non-centrality parameter

  power <- pt(qt(p,df), df, ncp=ncp, lower.tail=FALSE)
  n <- n + 1
}
n
```

```
## [1] 28
```

```
power
```

```
## [1] 0.8118316
```

1.2.2. Estimando o número de repetições por bloco

A abordagem proposta para calcular o número de repetições para o RCBD é similar ao procedimento para um CRD (*Completely Randomized Design*).

Para o CRD, o parâmetro de não-centralidade é definido como:

$$\delta^2 = \frac{n \sum_{i=1}^{a} (\mu_i - \mu)^2}{\sigma^2}$$

A variabilidade intra-grupo, necessária para calcular o *ncp*, corresponde a variância residual estimada, i.e., a variabilidade não explicada pelo fator experimental e pelo fator bloqueado. Neste experimento, essa variabilidade foi estimada a partir de um estudo piloto com 30 repetições por bloco.

Uma vez definido o *ncp*, a estatística F segue a seguinte distribuição:

$$F \sim \begin{cases} F_{a-1,a(n-1)} & \text{sob } H_0 \\ F_{a-1,a(n-1),\delta^2} & \text{sob } H_1 \end{cases}$$

O valor crítico F^* para rejeitar H_0 com nível de significância $\alpha = 0,05$ é dado por:

$$F^* = F_{a-1,a(n-1),\alpha}$$

Esse valor pode ser calculado usando o seguinte comando em R:

```
F.crit <- qf(alpha, a-1, a*(n-1), lower.tail = FALSE) # syntax
```

Por definição, o poder do teste é a probabilidade de rejeitar a hipótese nula (H_0) quando a hipótese alternativa H_1 é verdadeira:

$$\text{Poder} = P(\text{Rejeitar } H_0 | H_1 \text{ é verdadeira}).$$

Isso é equivalente a:

$$P(F(a-1, a(n-1), \delta^2) \geq F^*)$$

Sendo assim, o poder do teste pode ser calculado usando o seguinte comando:

```
power <- pf(F.crit, a-1, a*(n-1), ncp = ncp, lower.tail = FALSE) # syntax
```

A estratégia consiste em variar o número de repetições n, considerando $a = 2$ níveis do fator de interesse, até obter um poder do teste igual ou superior ao poder desejado ($\pi \geq 0,8$)

```
# Carregar a tabela de resultados
estudo_piloto <- read.csv("estudo_piloto_2.csv")

# Criar um dicionário (ou lista) para armazenar os resultados
estatisticas_piloto <- list()

# Obter todas as dimensões únicas
dimensoes <- unique(estudo_piloto$Dimensao)

# Calcular variância e desvio padrão para cada dimensão
for (dim in dimensoes) {
  # Filtrar os dados para a dimensão atual
  dados_dimensao <- subset(estudo_piloto, Dimensao == dim)

  # Calcular variância e desvio padrão de Fbest
  variancia <- var(dados_dimensao$Fbest)
  desvio_padrao <- sd(dados_dimensao$Fbest)

  # Salvar no dicionário (como uma lista com os dois valores)
```

```r
  estatisticas_piloto[[as.character(dim)]] <- list(
    Variancia = variancia,
    DesvioPadrao = desvio_padrao
  )
}

estatisticas_piloto_df <- data.frame(
  Dimensao = as.numeric(names(estatisticas_piloto)),
  Variancia = sapply(estatisticas_piloto, function(x) x$Variancia),
  DesvioPadrao = sapply(estatisticas_piloto, function(x) x$DesvioPadrao)
)
```

```r
alpha <- 0.05 # nível de significância
d <- 0.5
a <- 2 # Niveis do fator

# Dataframe para armazenar os resultados
repeticoes <- data.frame(
  Dimensao = integer(),
  Repeticoes = integer()
)

for (dim in dimensoes) {
  # Calcula n para cada dimensão para atingir poder >= 0.95
  print(dim)
  n <- 2
  power <- 0
  sd <- estatisticas_piloto_df$DesvioPadrao[estatisticas_piloto_df$Dimensao == dim]
  print(sd)
  D <- d * sd
  print(D)

  while (power < 0.8)
  {
    df1 <- a - 1 # Graus de liberdade do numerador
    df2 <- a * (n - 1) # Graus de liberdade do denominador
    #ncp <- (n * D^2) / (2 * sd^2) # Non-centrality parameter
    estudo_piloto
    # Calcular as médias observadas para a dimensão atual
    mu_config1 <- mean(estudo_piloto$Fbest[estudo_piloto$Dimensao == dim &
                                           estudo_piloto$Configuracao == "Config1"])
    mu_config2 <- mean(estudo_piloto$Fbest[estudo_piloto$Dimensao == dim &
                                           estudo_piloto$Configuracao == "Config2"])

    # Estimação da média geral e do desvio padrão
    mu <- mean(c(mu_config1, mu_config2))  # Média geral

    # Calcula o NCP
    ncp <- (n*((mu_config1 - mu)^2 + (mu_config2 - mu)^2)) / (2 * (sd^2))
    print(ncp)

    F.crit <- qf(1 - alpha, df1, df2, lower.tail = FALSE)
    print(F.crit)
```

```
    power <- pf(F.crit, df1, df2, ncp, lower.tail = FALSE)
    print(power)
    n = n + 1
  }

  # Armazena os resultados no dataframe
  repeticoes <- rbind(repeticoes, data.frame(Dimensao = dim, Repeticoes = n))
}
```

```
## [1] 2
## [1] 40.32631
## [1] 20.16316
## [1] 0.1848675
## [1] 0.005012531
## [1] 0.954404
## [1] 7
## [1] 2231.351
## [1] 1115.675
## [1] 0.198019
## [1] 0.005012531
## [1] 0.9547021
## [1] 13
## [1] 13962.53
## [1] 6981.265
## [1] 0.7999437
## [1] 0.005012531
## [1] 0.9664495
## [1] 18
## [1] 53158.04
## [1] 26579.02
## [1] 0.903314
## [1] 0.005012531
## [1] 0.9681354
## [1] 24
## [1] 152455.2
## [1] 76227.62
## [1] 1.238018
## [1] 0.005012531
## [1] 0.9730344
## [1] 29
## [1] 309188.8
## [1] 154594.4
## [1] 1.703355
## [1] 0.005012531
## [1] 0.9786196
## [1] 35
## [1] 512012.2
## [1] 256006.1
## [1] 1.871429
## [1] 0.005012531
## [1] 0.9803388
## [1] 40
## [1] 700272.2
## [1] 350136.1
```

```
## [1] 1.904168
## [1] 0.005012531
## [1] 0.9806572
## [1] 46
## [1] 918120.5
## [1] 459060.3
## [1] 1.910426
## [1] 0.005012531
## [1] 0.9807175
## [1] 51
## [1] 1105027
## [1] 552513.5
## [1] 1.927021
## [1] 0.005012531
## [1] 0.9808764
## [1] 57
## [1] 1367740
## [1] 683870
## [1] 1.892089
## [1] 0.005012531
## [1] 0.9805403
## [1] 62
## [1] 1762409
## [1] 881204.5
## [1] 1.889793
## [1] 0.005012531
## [1] 0.980518
## [1] 68
## [1] 2014883
## [1] 1007441
## [1] 1.893131
## [1] 0.005012531
## [1] 0.9805505
## [1] 73
## [1] 2290484
## [1] 1145242
## [1] 1.921548
## [1] 0.005012531
## [1] 0.9808242
## [1] 79
## [1] 2545762
## [1] 1272881
## [1] 1.910392
## [1] 0.005012531
## [1] 0.9807172
## [1] 84
## [1] 2693852
## [1] 1346926
## [1] 1.933791
## [1] 0.005012531
## [1] 0.9809409
## [1] 90
## [1] 2999357
## [1] 1499679
```

```
## [1] 1.932845
## [1] 0.005012531
## [1] 0.9809319
## [1] 95
## [1] 3274390
## [1] 1637195
## [1] 1.944982
## [1] 0.005012531
## [1] 0.981047
## [1] 101
## [1] 3544838
## [1] 1772419
## [1] 1.946865
## [1] 0.005012531
## [1] 0.9810648
## [1] 106
## [1] 3648838
## [1] 1824419
## [1] 1.928763
## [1] 0.005012531
## [1] 0.980893
## [1] 112
## [1] 4042565
## [1] 2021283
## [1] 1.9472
## [1] 0.005012531
## [1] 0.9810679
## [1] 117
## [1] 4151628
## [1] 2075814
## [1] 1.941875
## [1] 0.005012531
## [1] 0.9810176
## [1] 123
## [1] 4454904
## [1] 2227452
## [1] 1.936602
## [1] 0.005012531
## [1] 0.9809676
## [1] 128
## [1] 4715737
## [1] 2357869
## [1] 1.936921
## [1] 0.005012531
## [1] 0.9809706
## [1] 134
## [1] 4908252
## [1] 2454126
## [1] 1.943148
## [1] 0.005012531
## [1] 0.9810296
## [1] 139
## [1] 5260229
## [1] 2630115
```

```
## [1] 1.94741
## [1] 0.005012531
## [1] 0.9810699
## [1] 145
## [1] 5456009
## [1] 2728005
## [1] 1.955528
## [1] 0.005012531
## [1] 0.9811464
## [1] 150
## [1] 5752624
## [1] 2876312
## [1] 1.956055
## [1] 0.005012531
## [1] 0.9811514
```

3. Análise Exploratória

Antes de realizar os testes de hipótese, foi realizada uma análise exploratória para obter uma visão geral dos dados. A Figura @ref(fig:dataplot) fornece um gráfico para comparar o desempenho médio das duas configuração por dimensão.

```
# Load data
data <- read.csv("estudo_piloto_2.csv")
# Aggregate data (algorithm means by instance group)
aggdata <- with(data, aggregate(x = Fbest, by = list(Configuracao, Dimensao), FUN = mean))
names(aggdata) <- c("Configuracao", "Dimensao", "Y")

library(ggplot2)

# png(filename = "../figs/algo_lineplot.png",
#     width = 1000, height = 400,
#     bg = "transparent")
p <- ggplot(aggdata, aes(x = Dimensao,
                         y = Y,
                         group = Configuracao,
                         colour = Configuracao))
p + geom_line(linetype=2) + geom_point(size=5)
```

```
# dev.off()
```

A análise gráfica indica que a Configuração X apresenta

4. Análise Estatística

Para validar nossas observações iniciais, realizamos um teste RCBD para avaliar as diferenças no desempenho das duas configurações.

```
model <- aov(Y~Configuracao+Dimensao, data=aggdata)
summary(model)
summary.lm(model)$r.squared
```

```
##              Df    Sum Sq   Mean Sq F value   Pr(>F)
## Configuracao  1 3.252e+14 3.252e+14   88.13 7.31e-13 ***
## Dimensao      1 1.981e+14 1.981e+14   53.69 1.34e-09 ***
## Residuals    53 1.956e+14 3.690e+12
```

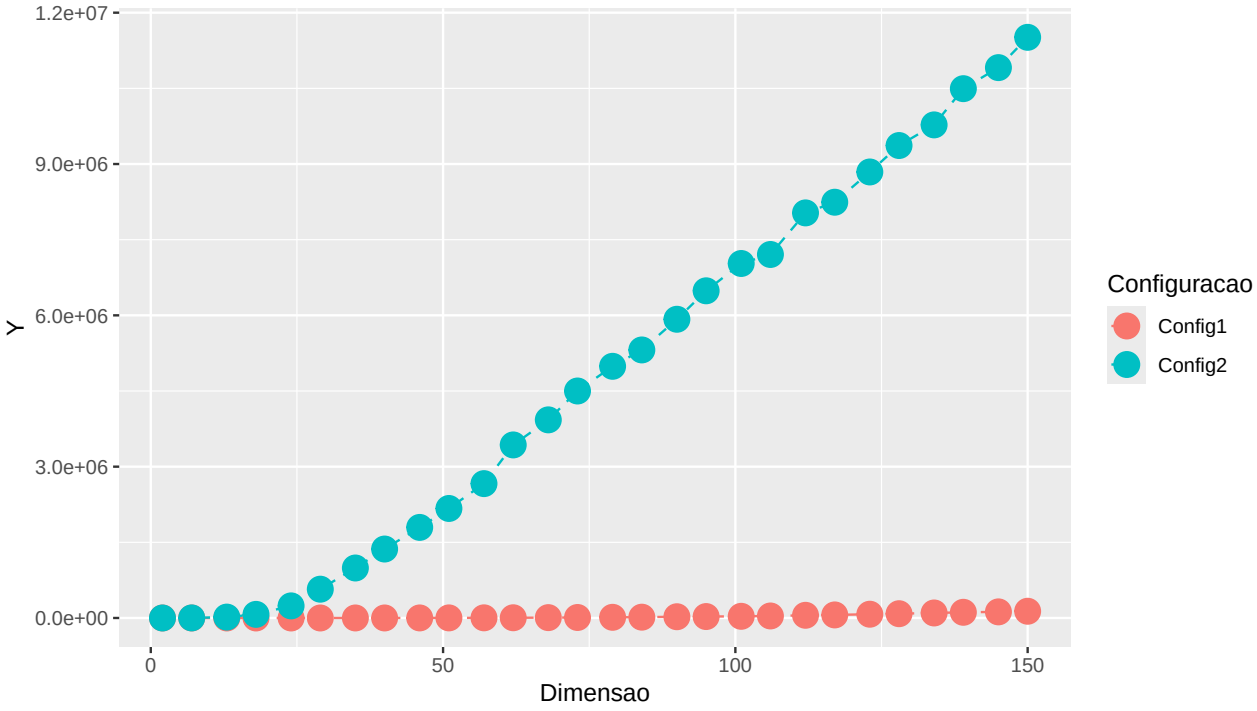
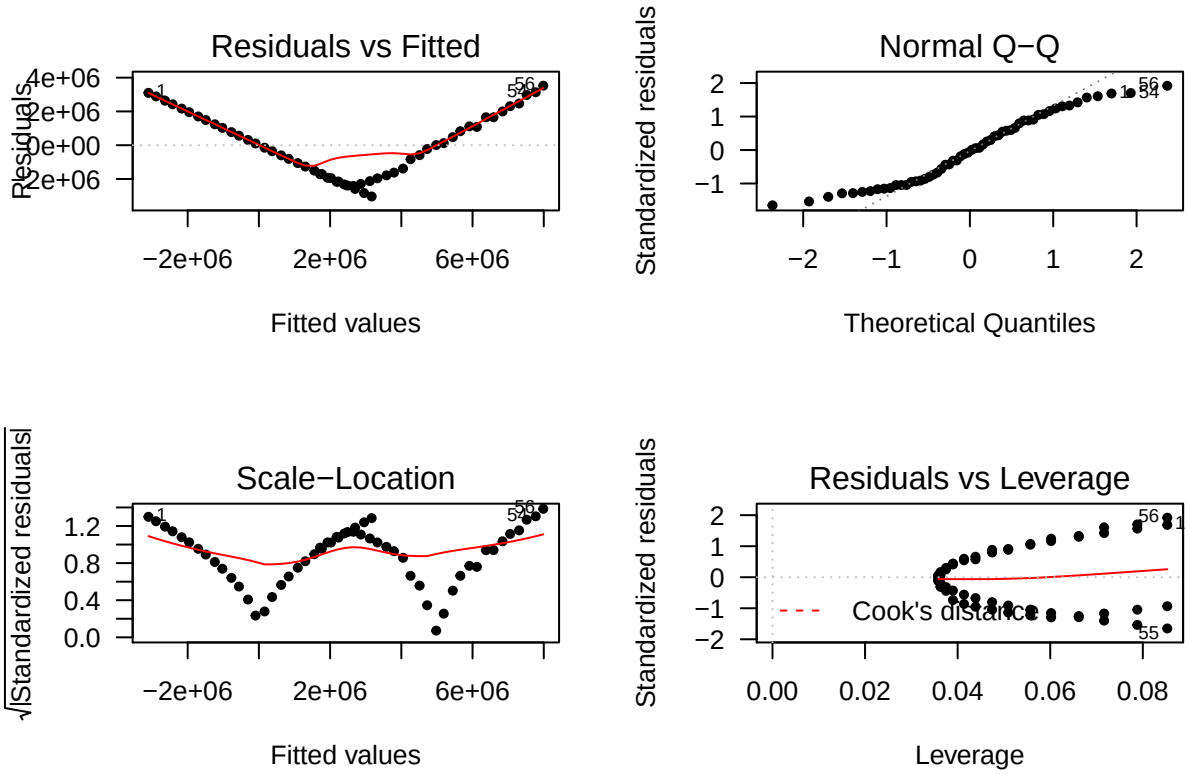


Figure 1: Desempenho médio das configurações por instância

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] 0.7279575
```

A Figura @ref(modelplot) mostra o modelo resultante do teste estatístico.

```
par(mfrow = c(2, 2))
plot(model, pch = 20, las = 1)
```



Confirmando a suspeita inicial, o teste de Tukey indica que a Ação 2 tem o maior retorno médio entre as ações analisadas, visto que as comparações entre a Ação 2 e as outras ações todas mostram diferenças significativas.

8. Referências

[1] Mathews, P.: Sample Size Calculations: Practical Methods for Engineers and Scientists, 1st edn. Matthews Malnar & Bailey Inc., Fairport Harbor (2010)

[2] Campelo, F., Takahashi, F. Sample size estimation for power and accuracy in the experimental comparison of algorithms. J Heuristics 25, 305–338 (2019). https://doi.org/10.1007/s10732-018-9396-7