



O **design formula** no DESeq2 define como os efeitos experimentais serão modelados para detectar genes diferencialmente expressos. Ele é passado ao criar o objeto com `DESeqDataSetFromMatrix`:

```
....
dds <- DESeqDataSetFromMatrix(countData = ..., colData = ..., design = ~ ...)
....
```

1. Simples: Comparação entre dois grupos

```
..
design = ~ condition
..
```

Quando usar: Você tem **duas condições** (ex: controle vs tratado).

A coluna `condition` no `colData` deve ter dois níveis (ex: `"control"`, `"treated"`).

O DESeq2 fará a comparação padrão do **segundo nível vs o primeiro** (ordem alfabética, a menos que você especifique).

2. Com bloqueio de variabilidade (batch effects)

```
..
design = ~ batch + condition
..
```

Quando usar: Você quer **controlar efeitos de batch**, sexo, indivíduo, etc.

DESeq2 estima o efeito do batch **antes** de testar o efeito de **condition**.

EXEMPLO:

```
``  
colData$batch <- factor(c("A", "A", "B", "B"))  
colData$condition <- factor(c("control", "treated", "control", "treated"))  
``
```

3. Design para comparações pareadas (pares paciente/controle)

```
``  
design = ~ subject + condition  
``
```

Quando usar: Você tem amostras **pareadas** (ex: antes/depois, mesmo indivíduo).

O **subject** identifica o par.

DESeq2 controla a variabilidade entre indivíduos.

4. Interação entre dois fatores

```
``  
design = ~ genotype + treatment + genotype:treatment  
``
```

ou

```
``  
design = ~ genotype * treatment # mesma coisa, atalho  
``
```

Quando usar: Você quer saber se o efeito do tratamento **depende do genótipo**.

DESeq2 testa:

- O efeito principal de **genotype**

- O efeito principal de `treatment`
- A interação: `genotype:treatment`

Exemplo:

```
..
colData$genotype <- factor(c("WT", "WT", "KO", "KO"))
colData$treatment <- factor(c("control", "treated", "control", "treated"))
..
```

5. Time course / séries temporais

```
..
design = ~ time + condition + time:condition
..
```

- **Quando usar:** Para estudar como a expressão muda ao longo do tempo **dependendo da condição**.
- `time` deve ser um fator (ou numérico, com cuidado).
- A interação `time:condition` permite identificar genes com resposta diferente ao longo do tempo entre os grupos.

Mudando o nível de referência

Por padrão, DESeq2 usa o **primeiro nível alfabético** como referência. Para mudar:

```
..
colData$condition <- relevel(colData$condition, ref = "control")
..
```

PARA CONCLUIR:

Hipoteticamente:

Temos 8 amostras:

- 2 grupos: `control` e `treated`

- 2 batches: A e B
- 4 indivíduos (pareados): S1, S2, S3, S4

sample	condition	batch	subject
S1_C	control	A	S1
S1_T	treated	A	S1
S2_C	control	A	S2
S2_T	treated	A	S2
S3_C	control	B	S3
S3_T	treated	B	S3
S4_C	control	B	S4
S4_T	treated	B	S4

Simular dados no R:

```
``
```

```
library(DESeq2)
```

```
# Simular metadados
```

```
samples <- data.frame(
  row.names = paste0("S", rep(1:4, each=2), "_", rep(c("C", "T"), 4)),
  condition = rep(c("control", "treated"), 4),
  batch = rep(c("A", "A", "B", "B"), each=2),
  subject = rep(paste0("S", 1:4), each=2)
)
```

```
samples$condition <- factor(samples$condition, levels = c("control", "treated"))
```

```
# Simular matriz de contagens (100 genes x 8 amostras)
```

```
set.seed(123)
counts <- matrix(rnbinom(100*8, mu=100, size=1), ncol=8)
rownames(counts) <- paste0("Gene", 1:100)
colnames(counts) <- rownames(samples)
```

```
# Introduzir efeito de tratamento em alguns genes
```

```
counts[1:10, samples$condition == "treated"] <- counts[1:10, samples$condition ==
"treated"] + 100
```

```
``
```

APLICANDO OS CONCEITOS DE DESIGN

Exemplo 1 – Design simples

```
``  
dds1 <- DESeqDataSetFromMatrix(countData = counts, colData = samples, design = ~  
condition)  
dds1 <- DESeq(dds1)  
res1 <- results(dds1)  
summary(res1)  
``
```

Aqui, não estamos controlando nenhum outro fator (batch ou indivíduo).

Pode identificar genes alterados por **condition**, mas com possível viés de batch.

Exemplo 2 – Design com batch

```
``  
dds2 <- DESeqDataSetFromMatrix(countData = counts, colData = samples, design = ~  
batch + condition)  
dds2 <- DESeq(dds2)  
res2 <- results(dds2)  
summary(res2)  
``
```

Corrige o efeito de **batch** antes de comparar **condition**.

Mais robusto quando há variação entre lotes.

Exemplo 3 – Design pareado por indivíduo

```
``  
dds3 <- DESeqDataSetFromMatrix(countData = counts, colData = samples, design = ~  
subject + condition)  
dds3 <- DESeq(dds3)  
res3 <- results(dds3)  
summary(res3)  
``
```

- Modelo ideal para **experimentos pareados**, controlando o efeito de cada indivíduo (**subject**).

- Mais sensível e específico se as amostras forem pares do mesmo paciente.

Comparar os resultados

``

```
# Número de genes significativos detectados
```

```
sum(res1$padj < 0.05, na.rm = TRUE)
```

```
sum(res2$padj < 0.05, na.rm = TRUE)
```

```
sum(res3$padj < 0.05, na.rm = TRUE)
```

``

Você verá que os resultados mudam bastante! O modelo com **subject + condition** geralmente é o mais confiável para estudos pareados.

Próximos passos: criar visualizações (volcano, PCA, heatmap) com esses resultados para mostrar graficamente a diferença entre os designs.

REFERÊNCIAS

https://www.r-bloggers.com/2024/05/a-guide-to-designs-and-contrasts-in-deseq2/#google_vignette

<https://bioconductor.org/packages/devel/bioc/manuals/DESeq2/man/DESeq2.pdf>