The `cbcTools` Package

Tools for Designing and Testing Choice-Based Conjoint Surveys in R

by John Paul Helveston

Sawtooth Software Conference

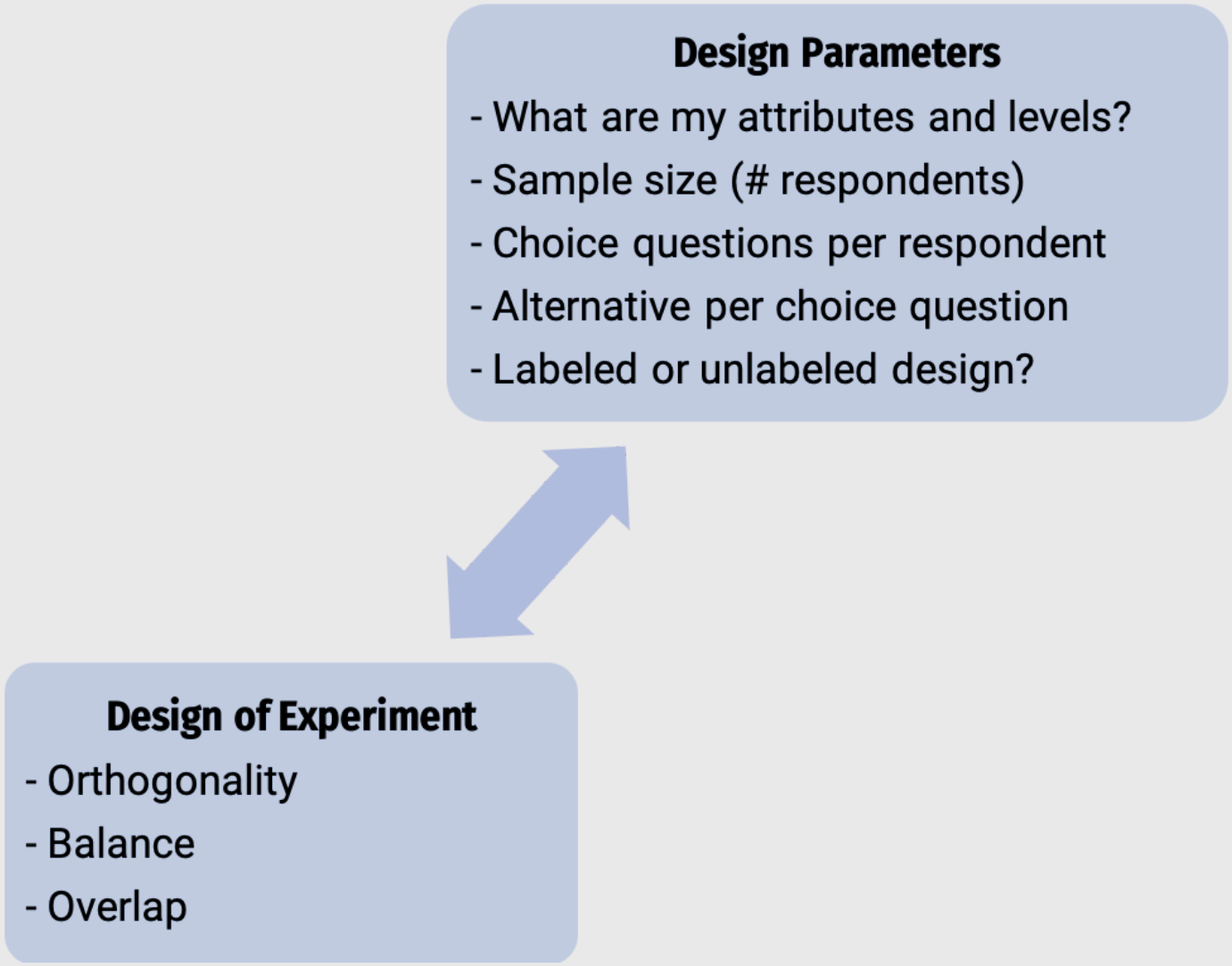
May 06, 2022

Designing a Choice-Based Conjoint Survey is Hard

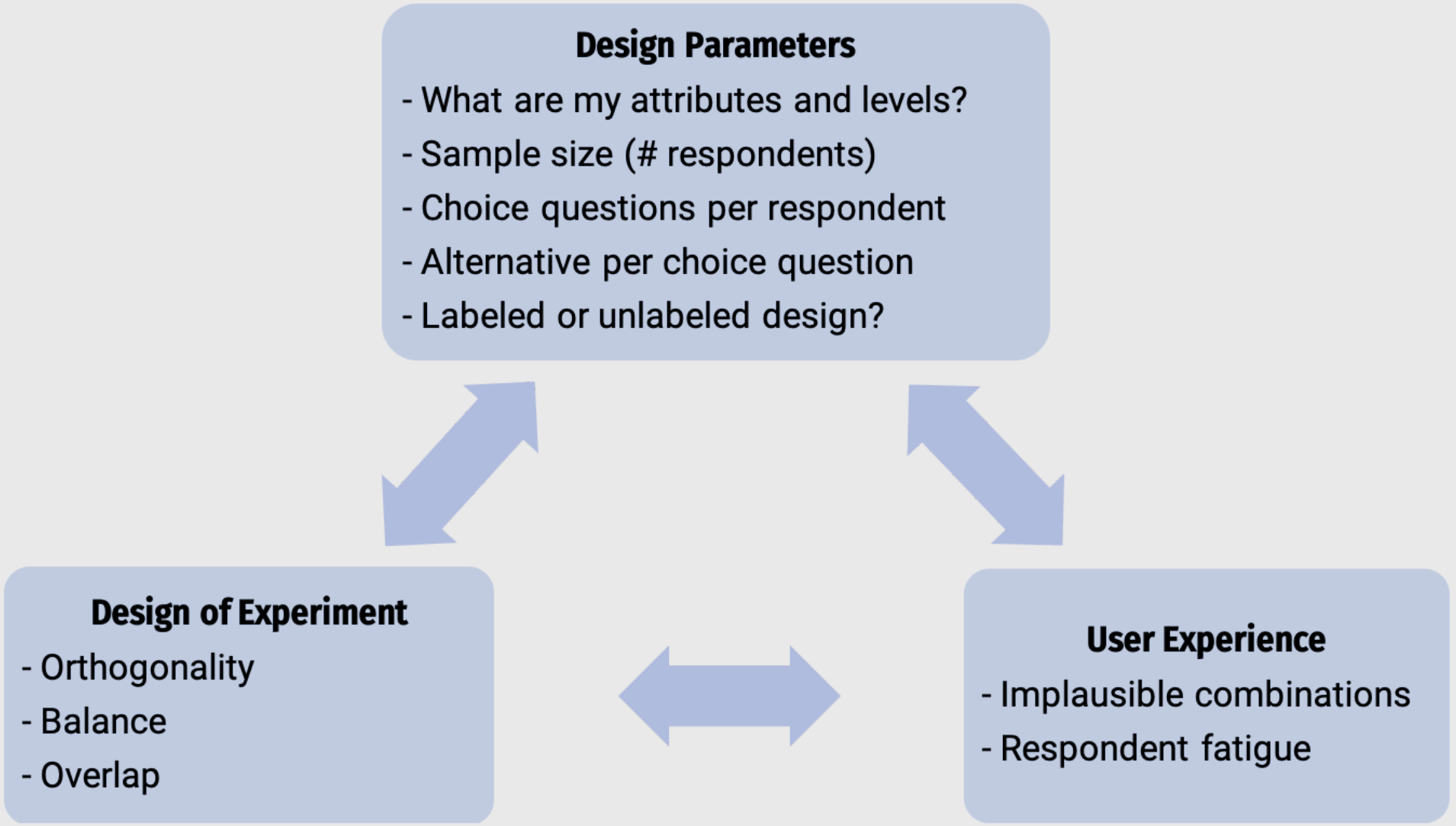
Design Parameters

- What are my attributes and levels?
- Sample size (# respondents)
- Choice questions per respondent
- Alternative per choice question
- Labeled or unlabeled design?

Designing a Choice-Based Conjoint Survey is Hard



Designing a Choice-Based Conjoint Survey is Hard



A simple conjoint experiment about *cars*

Attribute	Levels
Brand	GW, BMW, Ferrari
Price	$20k, $40k, $100k

Design: 9 choice sets, 3 alternatives each

```
Attribute counts:

brand:
  GM   BMW  Ferrari
  10    11     6

price:

 20k  40k 100k
  9    9   9
```

```
Pairwise attribute counts:

brand & price:

          20k 40k 100k
  GM        3   0    7
  BMW       4   5    2
  Ferrari   2   4    0
```

A simple conjoint experiment about *cars*

Attribute	Levels
Brand	GW, BMW, Ferrari
Price	$20k, $40k, $100k

Design: 90 choice sets, 3 alternatives each

```
Attribute counts:

brand:
  GM    BMW   Ferrari
  92    80     98

price:

  20k  40k 100k
  91   84   95
```

```
Pairwise attribute counts:

brand & price:

           20k 40k 100k
  GM       31  31  30
  BMW      25  25  30
  Ferrari  35  28  35
```

D-efficient designs

Attempts to maximize information on "Main Effects"

```
Attribute counts:

brand:
  GM    BMW   Ferrari
  90    90     90

price:

  20k  40k 100k
  90   90  90
```

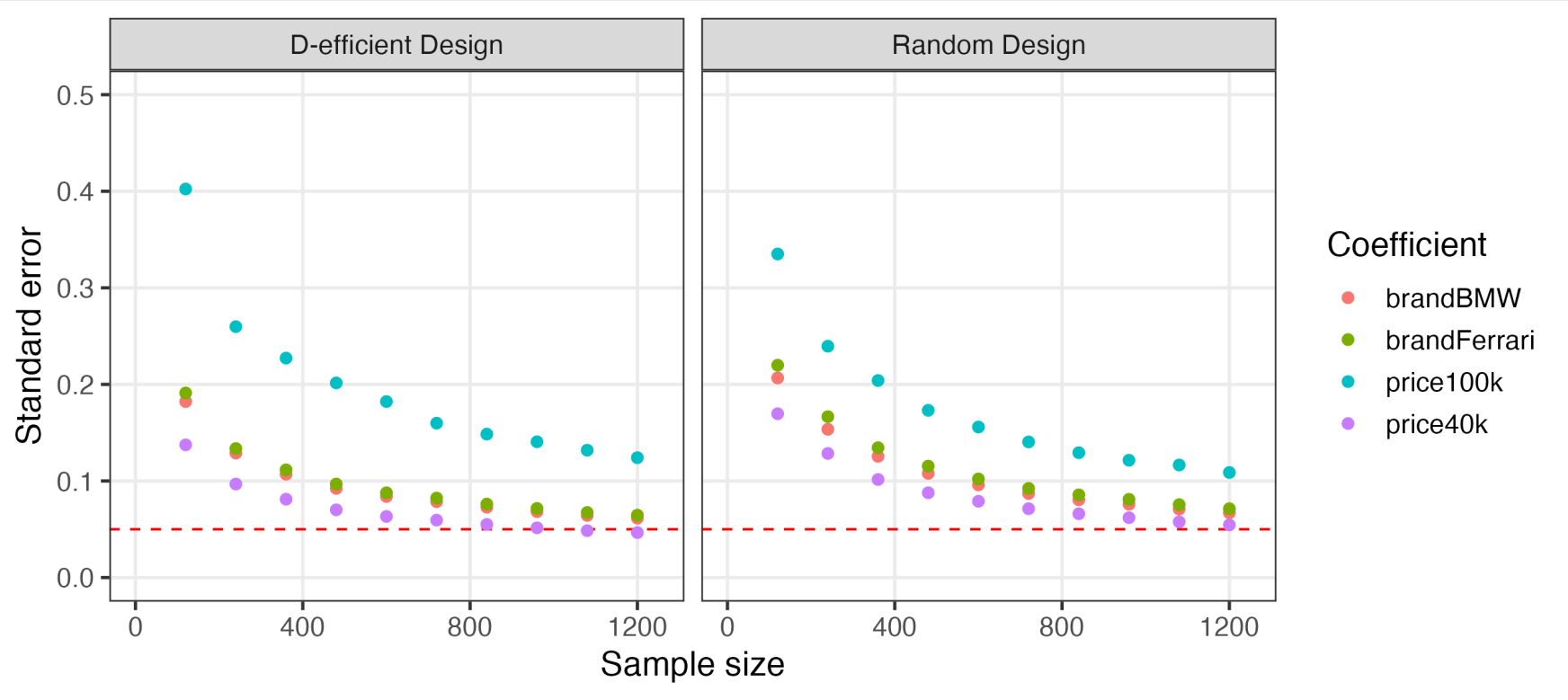
```
Pairwise attribute counts:

brand & price:

           20k 40k 100k
  GM       30  30  30
  BMW      30  30  30
  Ferrari  30  30  30
```

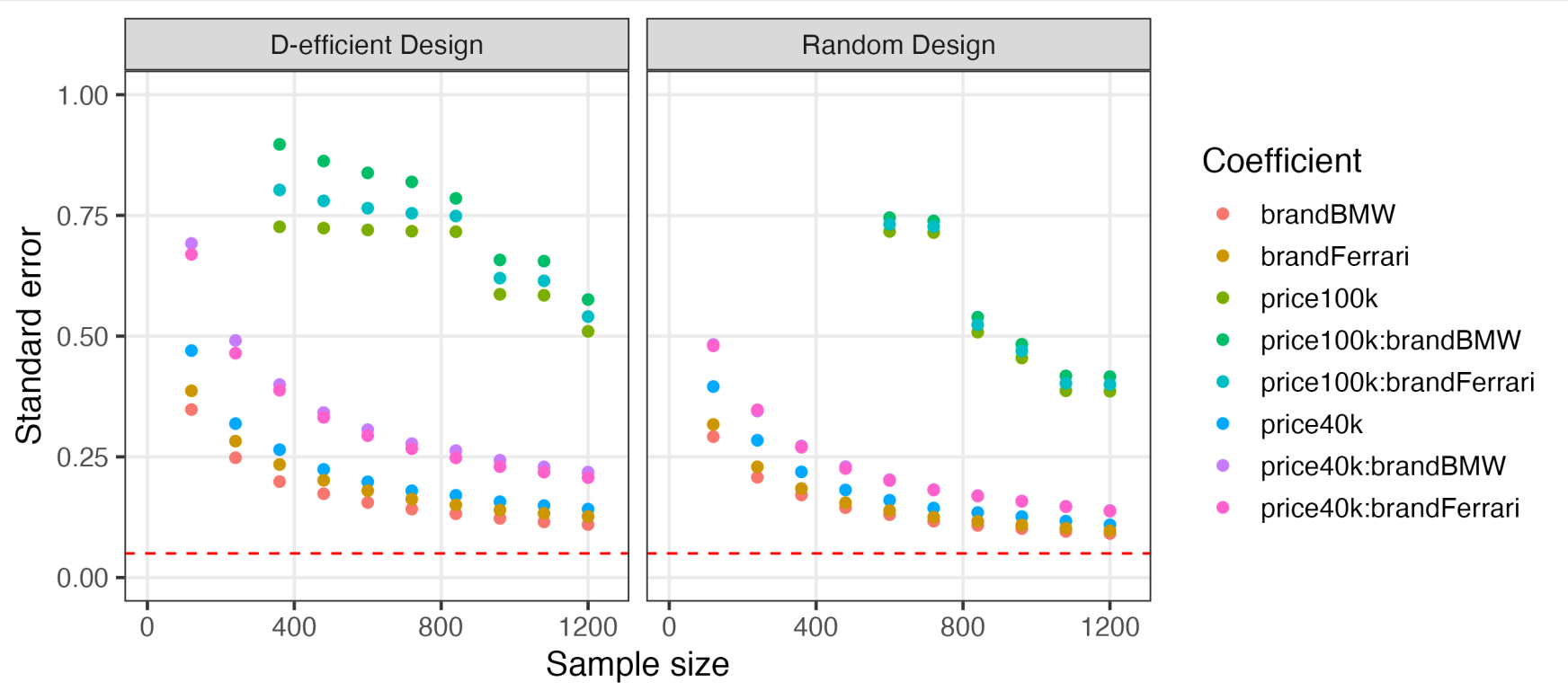
D-efficient designs

Attempts to maximize information on Main Effects



D-efficient designs

...but interaction effects are confounded in D-efficient designs



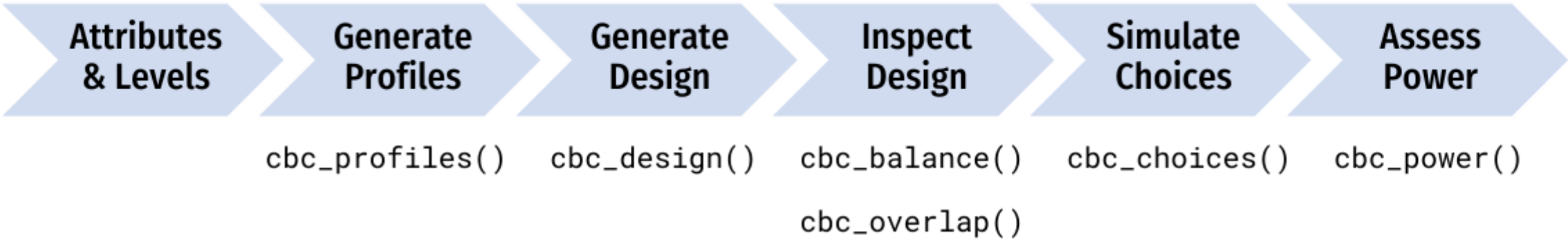
But what about other factors?

- What if I add one more choice question to each respondent?
- What if I increase the number of alternatives per choice question?
- What if I use a labeled design?
- What if there are interaction effects?

The cbcTools Package

https://github.com/jhelvy/2022-sawtooth-conf


Attributes & Levels
Generate Profiles
Generate Design
Inspect Design
Simulate Choices
Assess Power


Attributes & Levels
Generate Profiles
Generate Design
Inspect Design
Simulate Choices
Assess Power
cbc_profiles()
cbc_design()
cbc_balance()
cbc_overlap()
cbc_choices()
cbc_power()

Attribu
& Leve

```
1 library(cbcTools)
2
3 cbc_
```

cbc_balance {cbcTools}
cbc_choices {cbcTools}
cbc_design {cbcTools}
cbc_overlap {cbcTools}
cbc_power {cbcTools}
cbc_profiles {cbcTools}

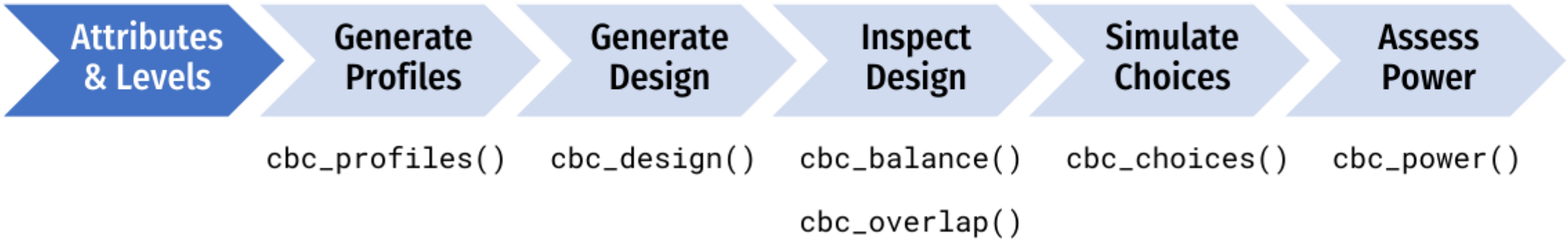
cbc_balance(design, atts = NULL)

This function prints out a summary of the counts of each level for each attribute across all choice questions as well as the two-way counts across all pairs of attributes for a given design.

Press F1 for additional help

ssess
ower

_power()


Attributes & Levels
Generate Profiles
Generate Design
Inspect Design
Simulate Choices
Assess Power
cbc_profiles()
cbc_design()
cbc_balance()
cbc_overlap()
cbc_choices()
cbc_power()

Define the attributes and levels

```
levels <- list(
  price     = c(1.00, 1.50, 2.00, 2.50, 3.00, 3.50, 4.00), # $ per pound
  type      = c("Fuji", "Gala", "Honeycrisp"),
  freshness = c("Excellent", "Average", "Poor")
)
```

```
levels
```

```
#> $price
#> [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0
#>
#> $type
#> [1] "Fuji"       "Gala"       "Honeycrisp"
#>
#> $freshness
#> [1] "Excellent" "Average"   "Poor"
```

`cbc_profiles()`

`cbc_design()`

`cbc_balance()`

`cbc_overlap()`

`cbc_choices()`

`cbc_power()`

Generate all possible profiles

```
profiles <- cbc_profiles(levels)
```

```
head(profiles)
```

```
#>   profileID price type freshness
#> 1         1   1.0 Fuji Excellent
#> 2         2   1.5 Fuji Excellent
#> 3         3   2.0 Fuji Excellent
#> 4         4   2.5 Fuji Excellent
#> 5         5   3.0 Fuji Excellent
#> 6         6   3.5 Fuji Excellent
```

```
tail(profiles)
```

```
#>    profileID price       type freshness
#> 58        58   1.5 Honeycrisp      Poor
#> 59        59   2.0 Honeycrisp      Poor
#> 60        60   2.5 Honeycrisp      Poor
#> 61        61   3.0 Honeycrisp      Poor
#> 62        62   3.5 Honeycrisp      Poor
#> 63        63   4.0 Honeycrisp      Poor
```

`cbc_profiles()` **`cbc_design()`** `cbc_balance()` `cbc_overlap()` `cbc_choices()` `cbc_power()`

Generate a survey design

```
design <- cbc_design(
  profiles = profiles,
  n_resp   = 300, # Number of respondents
  n_alts   = 3,   # Number of alternatives per question
  n_q      = 6    # Number of questions per respondent
)
```

```
head(design)
```

```
#>   respID qID altID obsID profileID price       type freshness
#> 1      1   1     1     1        31   2.0       Gala   Average
#> 2      1   1     2     1        49   4.0       Fuji      Poor
#> 3      1   1     3     1        16   1.5 Honeycrisp Excellent
#> 4      1   2     1     2        31   2.0       Gala   Average
#> 5      1   2     2     2        49   4.0       Fuji      Poor
#> 6      1   2     3     2        56   4.0       Gala      Poor
```

Include a "no choice" option

```
design <- cbc_design(
  profiles  = profiles,
  n_resp    = 300, # Number of respondents
  n_alts    = 3,   # Number of alternatives per question
  n_q       = 6,    # Number of questions per respondent
  no_choice = TRUE
)
```

```
head(design)
```

```
#>       respID qID altID obsID profileID price type_Fuji type_Gala type_Honeycrisp freshn
#> 1          1   1     1     1        34   3.5         0         1               0
#> 2          1   1     2     1        12   3.0         0         1               0
#> 3          1   1     3     1        54   3.0         0         1               0
#> 11000      1   1     4     1         0   0.0         0         0               0
#> 4          1   2     1     2         3   2.0         1         0               0
#> 5          1   2     2     2        29   1.0         0         1               0
```

Make a labeled design

```
design <- cbc_design(
  profiles = profiles,
  n_resp   = 300, # Number of respondents
  n_alts   = 3,   # Number of alternatives per question
  n_q      = 6,    # Number of questions per respondent
  label    = "type"
)
```

```
head(design)
```

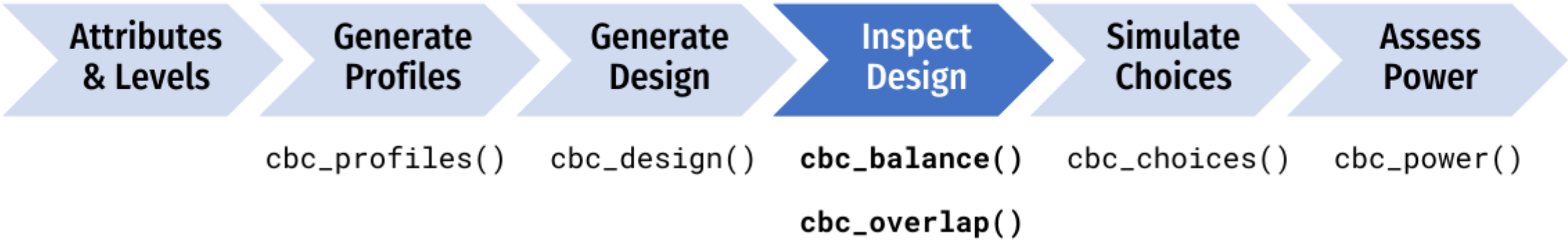
```
#>   respID qID altID obsID profileID price       type freshness
#> 1      1   1     1     1        45   2.0       Fuji      Poor
#> 2      1   1     2     1        34   3.5       Gala   Average
#> 3      1   1     3     1        62   3.5 Honeycrisp      Poor
#> 4      1   2     1     2        25   2.5       Fuji   Average
#> 5      1   2     2     2         8   1.0       Gala Excellent
#> 6      1   2     3     2        19   3.0 Honeycrisp Excellent
```

Make a D-efficient design (coming soon!)

```
design <- cbc_design(
  profiles = profiles,
  n_resp   = 300, # Number of respondents
  n_alts   = 3,   # Number of alternatives per question
  n_q      = 6,    # Number of questions per respondent
  d_eff    = TRUE
)
```

- Check out the `idefix` package

- Import a design: Sawtooth →  → 


Attributes & Levels
Generate Profiles
Generate Design
Inspect Design
Simulate Choices
Assess Power
cbc_profiles()
cbc_design()
cbc_balance()
cbc_overlap()
cbc_choices()
cbc_power()

Check design **balance**

```
cbc_balance(design)
```

```
Attribute counts:

price:

      1 1.5   2 2.5   3 3.5   4
    825 797 743 743 767 779 746

type:

    Fuji       Gala Honeycrisp
    1842       1769       1789

freshness:

    Excellent   Average      Poor
         1813      1775      1812
```

```
Pairwise attribute counts:

price & type:

      Fuji Gala Honeycrisp
  1    304  252         269
  1.5  274  251         272
  2    257  254         232
  2.5  240  254         249
  3    249  263         255
  3.5  257  250         272
  4    261  245         240
```

Check design **overlap**

```
cbc_overlap(design)
```

```
Counts of attribute overlap:
(# of questions with N unique levels)

price:

     1    2    3
    31  630 1139

type:

     1    2    3
   156 1248  396

freshness:

     1    2    3
   175 1189  436
```


Attributes & Levels
Generate Profiles
Generate Design
Inspect Design
Simulate Choices
Assess Power
cbc_profiles()
cbc_design()
cbc_balance()
cbc_overlap()
cbc_choices()
cbc_power()

Simulate random choices

```
data <- cbc_choices(
  design = design,
  obsID  = "obsID"
)
```

```
head(data)
```

```
#>   respID qID altID obsID profileID price       type freshness choice
#> 1      1   1     1     1         45   2.0       Fuji      Poor      0
#> 2      1   1     2     1         34   3.5       Gala   Average      0
#> 3      1   1     3     1         62   3.5 Honeycrisp      Poor      1
#> 4      1   2     1     2         25   2.5       Fuji   Average      0
#> 5      1   2     2     2          8   1.0       Gala Excellent      1
#> 6      1   2     3     2         19   3.0 Honeycrisp Excellent      0
```

Simulate choices according to a prior

```
data <- cbc_choices(
  design = design,
  obsID = "obsID",
  priors = list(
    price     = -0.1,
    type      = c(0.1, 0.2),
    freshness = c(0.1, -0.2)
  )
)
```

Attribute	Level	Utility
Price	Continuous	-0.1
Type	Fuji	0
	Gala	0.1
	Honeycrisp	0.2
Freshness	Average	0
	Excellent	0.1
	Poor	-0.2

Simulate choices according to a prior

```
data <- cbc_choices(
  design = design,
  obsID = "obsID",
  priors = list(
    price = -0.1,
    type = randN(
      mu    = c(0.1, 0.2),
      sigma = c(0.5, 1)
    ),
    freshness = c(0.1, -0.2)
  )
)
```

Attribute	Level	Utility
Price	Continuous	-0.1
Type	Fuji	0
	Gala	N(0.1, 0.5)
	Honeycrisp	N(0.2, 1)
Freshness	Average	0
	Excellent	0.1
	Poor	-0.2

`cbc_profiles()` `cbc_design()` `cbc_balance()` `cbc_choices()` **`cbc_power()`**

`cbc_overlap()`

Conduct a power analysis

```
power <- cbc_power(
    nbreaks = 10,
    n_q     = 3,
    data    = data,
    pars    = c("price", "type", "freshness"),
    outcome = "choice",
    obsID   = "obsID"
)
```

```
head(power)
```

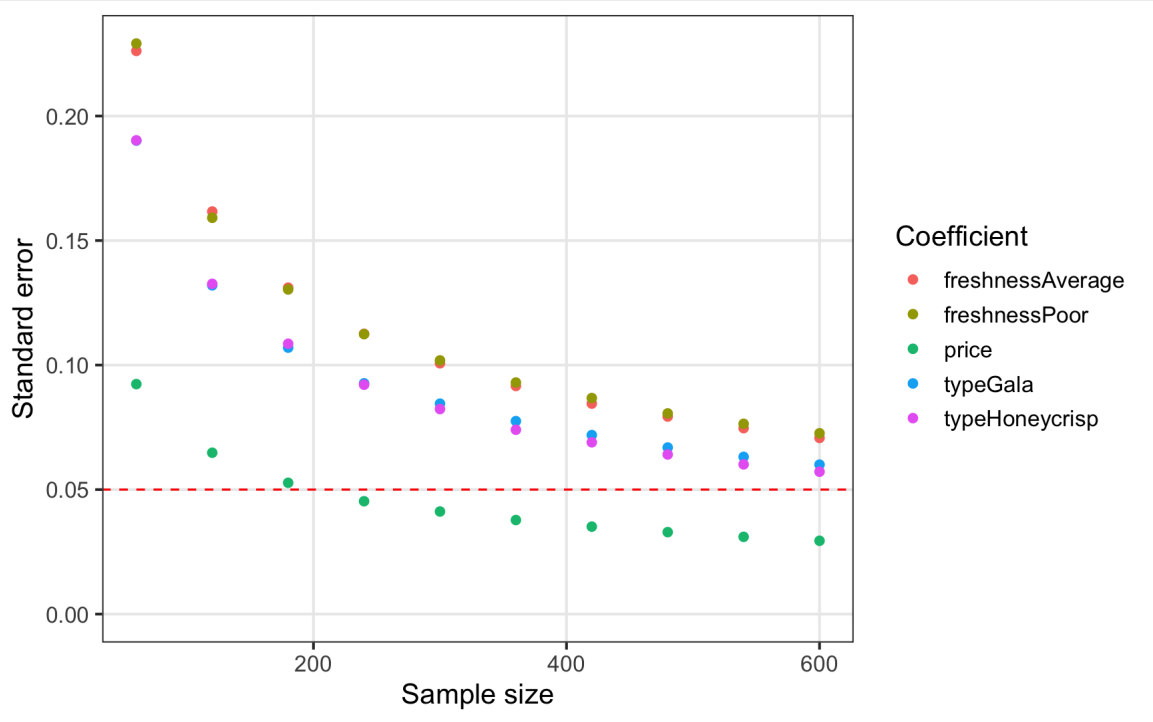
```
#>   sampleSize             coef          est
#> 1         60            price  0.01039211
#> 2         60         typeGala  0.31948570
#> 3         60   typeHoneycrisp  0.32384030
#> 4         60 freshnessAverage  0.20887605
#> 5         60    freshnessPoor  0.14539300
#> 6        120            price -0.09227504
```

```
tail(power)
```

```
#>    sampleSize             coef           e
#> 45        540    freshnessPoor -0.1177395
#> 46        600            price -0.0822952
#> 47        600         typeGala -0.0061322
#> 48        600   typeHoneycrisp  0.1968087
#> 49        600 freshnessAverage  0.1768854
#> 50        600    freshnessPoor -0.1036471
```

Conduct a power analysis

```
plot(power)
```



Attributes & Levels	Generate Profiles	Generate Design	Inspect Design	Simulate Choices	Assess Power
	`cbc_profiles()`	`cbc_design()`	`cbc_balance()` `cbc_overlap()`	`cbc_choices()`	`cbc_power()`

Thanks!

`cbcTools` documentation: https://jhelvy.github.io/cbcTools/

Slides: https://jhelvy.github.io/2022-sawtooth-conf

@johnhelveston
@jhelvy
@jhelvy
jhelvy.com
jph@gwu.edu

Extra slides