Validation Report for **adoptr** package

Kevin Kunzmann & Maximilian Pilz

2019-04-03

Contents

Chapter 1

Introduction

This work is licensed under the CC-BY-SA 4.0 license

1.1 Concept

R package validation for regulatory environments is a tedious endevour. Note that there is no such thing as a 'validated R package': validation is by definition a process conducted by the *user*. This valdation report may thus only be seen as a means to facilitate validation of **adoptr** as much as possible. No warranty whatsoever as to the correctness of **adoptr** not the completeness of the validation report are given by the authors!

We assume that the reader is familiar with the notation an theoretical background of **adoptr**. Otherwise, the resources linked at https://github.com/kkmann/adoptr might be a good starting point *ToDo: publications*. The report explores a variety of essential scenarios and both formally tests results (wherever possible) using **testthat**. Detailed results are also printed in the report itself. Any failure of the integrated formal tests will cause the build status of the validation report to switch from 'passing' to 'failed'. An overview of the respective test scenarios is given in Validation Scenarios.

The online version of this report can be found at https://kkmann.github.io/adoptr-validation-report/ and is automatically rebuild and redeployed on a daily basis using Travis-CI. It uses the respective most current CRAN version of **adoptr**. The source code repository of this report can be found at https://github.com/kkmann/adoptr-validation-report.

1.2 Validation Scenarios

1.2.1 Scenario I: Large effect, point prior

This is the default scenario.

- **Data distribution:** Two-armed trial with normally distributed test statistic
- **Prior:** $\delta \sim \delta_{0.4}$
- **Null hypothesis:** $\mathcal{H}_0 : \delta \leq 0$

1.2.1.1 Variant I.1: Minimizing Expected Sample Size under the Alternative

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.4\big]$
- **Constraints:**

    1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.4\big] \geq 0.8$
    2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
    3. Three variants: two-stage, group-sequential, one-stage.
- **Formal tests:**
    1. All three **adoptr** variants (two-stage, group-sequential, one-stage) comply with constraints. Internally validated by testing vs. simulated values of the power curve at respective points.
    2. ESS of optimal two-stage design is lower than ESS of optimal group-sequential one and that is in turn lower than the one of the optimal one-stage design.
    3. ESS of optimal group-sequential design is lower than ESS of externally computed group-sequential design using the rpact package.
    4. Are the ESS values obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?
    5. Is $n()$ of the optimal two-stage design monotonously decreasing on continuation area?

1.2.1.2 Variant I.2: Minimizing Expected Sample Size under the Null Hypothesis

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1) \,|\, \delta = 0.0\big]$
- **Constraints:**
    1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.4\big] \geq 0.8$
    2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
- **Formal tests:**
    1. Validate constraint compliance by testing vs. simulated values of the power curve at respective points.
    2. $n()$ of optimal design is monotonously increasing on continuation area.
    3. ESS of optimal two-stage design is lower than ESS of externally computed group-sequential design using the rpact package.
    4. Are the ESS values obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?

1.2.1.3 Variant I.3: Condtional Power Constraint

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1) \,|\, \delta = 0.4\big]$
- **Constraints:**
    1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.4\big] \geq 0.8$
    2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
    3. $CP := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.4, X_1 = x_1\big] \geq 0.7$ for all $x_1 \in (c_1^f, c_1^e)$
- **Formal tests:**
    1. Check $Power$ and $TOER$ constraints with simulation. Check CP constraint on three different values of x_1 in (c_1^f, c_1^e)
    2. Are the CP values at the three test-pivots obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?
    3. Is ESS of optimal two-stage design with CP constraint higher than ESS of optimal two-stage design without this constraint?

1.2.2 Scenario II: Large effect, Gaussian prior

Similar in scope to Scenario I, but with a continuous Gaussian prior on δ.

- **Data distribution:** Two-armed trial with normally distributed test statistic
- **Prior:** $\delta \sim \mathcal{N}(0.4, .3)$
- **Null hypothesis:** $\mathcal{H}_0 : \delta \leq 0$

1.2.2.1 Variant II.1: Minimizing Expected Sample Size

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta > 0.0\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
  3. Three variants: two-stage, group-sequential, one-stage.
- **Formal tests:**
  1. All designs comply with type one error rate constraints (tested via simulation).
  2. ESS of optimal two-stage design is lower than ESS of optimal group-sequential one and that is in turn lower than the one of the optimal one-stage design.

1.2.2.2 Variant II.2: Minimizing Expected Sample Size under the Null hypothesis

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1) \,|\, \delta \leq 0\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta > 0.0\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
- **Formal tests:**
  1. Does the design comply with $TOER$ constraint (via simulation)?
  2. Check CP constraint on three different values of x_1 in (c_1^f, c_1^e)
  3. Is ESS lower than expected sample size under the null hypothesis for the optimal two stage design from Variant II-1?

1.2.2.3 Variant II.3: Condtional Power Constraint

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta > 0.0\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
  3. $CP := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta > 0.0, X_1 = x_1\big] \geq 0.7$ for all $x_1 \in (c_1^f, c_1^e)$
- **Formal tests:**
  1. Check $TOER$ constraint with simulation. Check CP constraint on three different values of x_1 in (c_1^f, c_1^e)
  2. Is ESS of optimal two-stage design with CP constraint higher than ESS of optimal two-stage design without the constraint?

1.2.3 Scenario III: Large effect, uniform prior

- **Data distribution:** Two-armed trial with normally distributed test statistic
- **Prior:** sequence of uniform distributions $\delta \sim \text{Unif}(0.4 - \Delta_i, 0.4 + \Delta_i)$ around 0.4 with $\Delta_i = (3 - i)/10$ for $i = 0 \ldots 3$. I.e., for $\Delta_3 = 0$ reduces to a point prior on $\delta = 0.4$.
- **Null hypothesis:** $\mathcal{H}_0 : \delta \leq 0$

1.2.3.1 Variant III.1: Convergence under Prior Concentration

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta > 0.0\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$

- **Formal tests:**
  1. Simulated type one error rate is compared to $TOER$ constraint for each design.
  2. Number of iterations are checked agaist default maximum to ensure proper convergence.
  3. ESS decreases with prior variance.

Additionally, the designs are compared graphically. Inspect the plot to see convergence pattern.

1.2.4 Scenario IV: Smaller effect size, larger trials

1.2.4.1 Variant IV.1: Minimizing Expected Sample Size under the Alternative

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.2\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.2\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.025$
  3. Three variants: two-stage, group-sequential, one-stage.
- **Formal tests:**
  1. All three adoptr variants (two-stage, group-sequential, one-stage) comply with costraints. Internally validated by testing vs. simulated values of the power curve at respective points.
  2. ESS of optimal two-stage design is lower than ESS of optimal group-sequential one and that is in tunr lower than the one of the optimal one-stage design.
  3. ESS of optimal group-sequential design is lower than ESS of externally computed group-sequential design using the rpact package.
  4. Are the ESS values obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?
  5. Is $n()$ of the optimal two-stage design monotonously decreasing on continuation area?

1.2.4.2 Variant IV.2: Increasing Power

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.2\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.2\big] \geq 0.9$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.025$
  3. Three variants: two-stage, group-sequential, one-stage.
- **Formal tests:**
  1. Does the design respect all constraints (via simulation)?
  2. ESS of optimal two-stage design is lower than ESS of optimal group-sequential one and that is in tunr lower than the one of the optimal one-stage design.
  3. ESS of optimal group-sequential design is lower than ESS of externally computed group-sequential design using the rpact package.
  4. Are the ESS values obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?
  5. Is $n()$ of the optimal two-stage design monotonously decreasing on continuation area?

1.2.4.3 Variant IV.3: Increasing Maximal Type One Error Rate

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.2\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.2\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.05$
  3. Three variants: two-stage, group-sequential, one-stage.
- **Formal tests:**

1. Does the design respect all constraints (via simulation)?
2. ESS of optimal two-stage design is lower than ESS of optimal group-sequential one and that is in tunr lower than the one of the optimal one-stage design.
3. ESS of optimal group-sequential design is lower than ESS of externally computed group-sequential design using the rpact package.
4. Are the ESS values obtained from simulation the same as the ones obtained by using numerical integration via `adoptr::evaluate`?
5. Is $n()$ of the optimal two-stage design monotonously decreasing on continuation area?

1.2.5 Scenario V: Single-arm design, medium effect size

- **Data distribution:** One-armed trial with normally distributed test statistic
- **Prior:** $\delta \sim \delta_{0.3}$
- **Null hypothesis:** $\mathcal{H}_0 : \delta \leq 0$

1.2.5.1 Variant V.1: Sensitivity to Integration Order

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.3\big]$
- **Constraints:**
  1. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.3\big] \geq 0.8$
  2. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.025$
  3. Three variants: integration order 5, 8, 11 two-stage designs.
- **Formal tests:**
  1. Do all designs respect all constraints (via simulation)?
  2. Do all designs converge within the respective iteration limit?
  3. Does constraint compliance get better with increased order?
  4. Does the simulated ESS get better with increased order?

1.2.5.2 Variant V.2: Utility Maximization

- **Objective:** $\lambda\, Power - ESS := \lambda\, \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.3\big] - \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.3\big]$. for $\lambda = 100$ and 200
- **Constraints:**
  1. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.025$
- **Formal tests:**
  1. Do both desings respect the type one error rate constraint (via simulation)?
  2. Is the power of the design with larger λ larger?

1.2.5.3 Variant V.3: n_1 penalty

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.3\big] + \lambda\, n_1$ for $\lambda = 0.05$ and 0.2.
- **Constraints:**
  1. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.0\big] \leq 0.025$
  2. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2\,|\,\delta = 0.3\big] \geq 0.8$
- **Formal tests:**
  1. Is n_1 for the optimal design smaller than the order-5 design in V.1?

1.2.5.4 Variant V.4: n_2 penalty

- **Objective:** $ESS := \boldsymbol{E}\big[n(X_1)\,|\,\delta = 0.3\big] + \lambda$ `AverageN2` for $\lambda = 0.01$ and 0.1.

- **Constraints:**
  1. $TOER := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.0\big] \leq 0.025$
  2. $Power := \boldsymbol{Pr}\big[c_2(X_1) < X_2 \,|\, \delta = 0.3\big] \geq 0.8$
- **Formal tests:**
  1. Is the `AverageN2` for the optimal design smaller than for the order-5 design in V.1?

1.3 Technical Setup

All scenarios are run in a single, shared R session. Required packages are loaded here *ToDo: make sure that is the case!*, the random see is defined and set centrally, and the default number of iteration is increased to make sure that all scenarios converge properly. Additionally R scripts with convenience functions are sourced here as well *ToDo: descibe or eliminate these functions!*.

```r
library(adoptr)
library(tidyverse)

# load custom functions in folder subfolder '/R'
for (nm in list.files("R", pattern = "\\.[RrSsQq]$"))
   source(file.path("R", nm))

# define seed value
seed <- 42

# define custom tolerance and iteration limit for nloptr
opts = list(
    algorithm = "NLOPT_LN_COBYLA",
    xtol_rel  = 1e-5,
    maxeval   = 100000
)
```

Chapter 2

Scenario I: large effect, point prior

2.1 Details

In this scenario an alternative effect size of $\delta = 0.4$ with point prior distribution is investigated. The null hypothesis is $\delta \leq 0$. Currently, `adoptr` only supports normal distributed data what is widely spread in the development of adaptive designs. We protect the one-sided type one error rate at $\alpha = 0.025$ and require the power of the design to be at least $1 - \beta = 0.8$.

2.1.1 Data distribution

Two-armed trial with normally distributed test statistic

```
datadist <- Normal(two_armed = TRUE)
```

2.1.2 Null hypothesis

The null hypothesis is $\mathcal{H}_0 : \delta \leq 0$

```
H_0 <- PointMassPrior(.0, 1)
```

2.1.3 Prior assumptions

A point mass prior with probability mass on $\delta = 0.4$ is assumed.

```
prior <- PointMassPrior(.4, 1)
```

2.2 Variant I-1: Minimizing Expected Sample Size under Point Prior

2.2.1 Objective

Expected sample size under the respective prior is minimized, i.e., $\boldsymbol{E}\big[n(\mathcal{D})\big]$.

```
ess <- expected(ConditionalSampleSize(datadist, prior))
```

2.2.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis.

```
toer_cnstr <- expected(ConditionalPower(datadist, H_0)) <= .025
```

Power must be larger than 0.8.

```
pow_cnstr <- expected(ConditionalPower(datadist, prior)) >= .8
```

2.2.3 Initial Design

`adoptr` requires the definition of an initial design for optimization. We start with a group-sequential design from the package `rpact` that fulfills these constraints and is used later for comparison. The order of integration is set to

```
order <- 7L
```

For usage as two-stage design with variable sample size, it has to be converted to a `TwoStageDesign`.

```
init_design_gs <- rpact_design(0.4, 0.025, 0.8, TRUE, order)

init_design    <- TwoStageDesign(init_design_gs)
```

2.2.4 Optimization

The optimal design is computed in three variants: two-stage, group-sequential and one-stage. The input only differs with regard to the initial design.

```
opt_design <- function(initial_design) {
    minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = initial_design,

        opts = opts

    )
}

opt1_ts <- opt_design(init_design)
opt1_gs <- opt_design(init_design_gs)
opt1_os <- opt_design(OneStageDesign(200, 2.0))
```

2.2.5 Test Cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 3402  985   24
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

The n_2 function of the optimal two-stage design is expected to be monotonously decreasing.

```
testthat::expect_equal(
    sign(diff(opt1_ts$design@n2_pivots)),
    rep(-1, (order - 1))
)
```

Type one error rate constraint is tested for the three designs. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                  function(x) sim_pr_reject(x$design, .0, datadist))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer <= .025*(1.01)))

df_toer
```

```
##       toer           se
## 1 0.024951 0.0001559759
## 2 0.024978 0.0001560581
## 3 0.025116 0.0001564775
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                  function(x) sim_pr_reject(x$design, .4, datadist))
df_pow <- data.frame(
    pow  = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_pow$pow >= .8 * (1 - 0.01)))

df_pow
```

```
##        pow           se
## 1 0.798641 0.0004010159
## 2 0.799669 0.0004002482
```

```
## 3 0.799317 0.0004005115
```

The expected sample sizes should be ordered in a specific way.

```
testthat::expect_gte(
    evaluate(ess, opt1_os$design),
    evaluate(ess, opt1_gs$design)
)

testthat::expect_gte(
    evaluate(ess, init_design_gs),
    evaluate(ess, opt1_gs$design)
)

testthat::expect_gte(
    evaluate(ess, opt1_gs$design),
    evaluate(ess, opt1_ts$design)
)
```

The expected sample size of the optimal designs is simulated and compared to the outomce of `adoptr::evaluate()`. The tolerance is set to 0.5 what is due to rounding one patient per group in the worst case.

```
ess_0 <- expected(ConditionalSampleSize(datadist, H_0))

testthat::expect_equal(
    sim_n(opt1_os$design, .0, datadist),
    evaluate(ess_0, opt1_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_gs$design, .0, datadist),
    evaluate(ess_0, opt1_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_ts$design, .0, datadist),
    evaluate(ess_0, opt1_ts$design),
    tolerance = .5
)
```

Additionally, the sample sizes under the point prior are compared.

```
testthat::expect_equal(
    sim_n(opt1_os$design, .4, datadist),
    evaluate(ess, opt1_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_gs$design, .4, datadist),
    evaluate(ess, opt1_gs$design),
    tolerance = .5
)
```

```
testthat::expect_equal(
    sim_n(opt1_ts$design, .4, datadist),
    evaluate(ess, opt1_ts$design),
    tolerance = .5
)
```

2.3 Variant I-2: Minimizing Expected Sample Size under Null Hypothesis

2.3.1 Objective

Expected sample size under the null hypothesis prior is minimized, i.e.,

```
ess_0 <- expected(ConditionalSampleSize(datadist, H_0))
```

2.3.2 Constrains

The constraints remain the same as before.

2.3.3 Initial Design

For runtime issues the previous initial design has to be updated. It turns out that a constant c_2-starting value is much more efficient in this case. Furthermore, a more strict upper-boundary design than the default one needs to be defined because stopping for efficacy would otherwise only happen for very large values of x_1 due to optimization under the null hypothesis.

```
init_design_2 <- init_design
init_design_2@c2_pivots <- rep(2, order)


ub_design <- TwoStageDesign(
    opt1_os$design@n1,
    opt1_os$design@c1f,
    3,
    rep(300, order),
    rep(3.0, order)
)
```

2.3.4 Optimization

The optimal two-stage design is computed.

```
opt2_ts <- minimize(

        ess_0,

        subject_to(

            toer_cnstr,
```

```
            pow_cnstr

        ),

        initial_design = init_design_2,

        upper_boundary_design = ub_design,

        opts = opts

)
```

```
## Warning in minimize(ess_0, subject_to(toer_cnstr, pow_cnstr),
## initial_design = init_design_2, : initial design is infeasible!
```

2.3.5 Test Cases

Check if the optimization algorithm converged.

```
print(opt2_ts$nloptr_return$iterations)
```

```
## [1] 20640
```

```
testthat::expect_true(opt2_ts$nloptr_return$iterations < opts$maxeval)
```

The n_2 function of the optimal two-stage design is expected to be monotnously increasing.

```
testthat::expect_equal(
    sign(diff(opt2_ts$design@n2_pivots)),
    rep(1, (order - 1))
)
```

Type one error rate constraint is tested for the optimal design. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sim_pr_reject(opt2_ts$design, .0, datadist)
df_toer2 <- data.frame(
    toer = as.numeric(tmp[1]),
    se   = as.numeric(tmp[2])
)
rm(tmp)

testthat::expect_true(all(df_toer2$toer <= .025*(1.01)))

df_toer2
```

```
##       toer           se
## 1 0.024971 0.0001560368
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sim_pr_reject(opt2_ts$design, .4, datadist)
df_pow2 <- data.frame(
    pow  = as.numeric(tmp[1]),
    se   = as.numeric(tmp[2])
)
```

```
rm(tmp)

testthat::expect_true(all(df_pow2$pow >= .8 * (1 - 0.01)))

df_pow2
```

```
##       pow           se
## 1 0.80175 0.0003986817
```

The expected sample size under the null should be lower than the ess under the null of the initial design derived from `rpact`.

```
testthat::expect_gte(
    evaluate(ess_0, init_design),
    evaluate(ess_0, opt2_ts$design)
)
```

The expected sample size of the optimal designs is simulated and compared to the outomce of `adoptr::evaluate()`. The tolerance is set to 0.5 what is due to rounding one patient per group in the worst case.

```
testthat::expect_equal(
    sim_n(opt2_ts$design, .0, datadist),
    evaluate(ess_0, opt2_ts$design),
    tolerance = .5
)
```

Additionally, the sample sizes under the point prior are compared.

```
testthat::expect_equal(
    sim_n(opt2_ts$design, .4, datadist),
    evaluate(ess, opt2_ts$design),
    tolerance = .5
)
```

2.4 Variant I-3: Conditional Power Constraint

2.4.1 Objective

Expected sample size under the point prior is minimized and has already been defined.

2.4.2 Constrains

The constraints remain the same as before, additionally to a constraint on conditional power.

```
cp <- ConditionalPower(datadist, prior)

cp_cnstr <- cp >= .7
```

2.4.3 Initial Design

The previous initial design can still be applied.

2.4.4 Optimization

The optimal two-stage design is computed.

```
opt3_ts <- minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr,
            cp_cnstr

        ),

        initial_design = init_design,

        opts = opts

)
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr, cp_cnstr),
## initial_design = init_design, : initial design is infeasible!
```

2.4.5 Test Cases

Check if the optimization algorithm converged.

```
print(opt3_ts$nloptr_return$iterations)
```

```
## [1] 3316
```

```
testthat::expect_true(opt3_ts$nloptr_return$iterations < opts$maxeval)
```

Type one error rate constraint is tested for the optimal design. Due to numerical issues we allow a realtive error of 1%.

```
tmp      <- sim_pr_reject(opt3_ts$design, .0, datadist)
df_toer3 <- data.frame(
    toer = as.numeric(tmp[1]),
    se   = as.numeric(tmp[2])
)
rm(tmp)

testthat::expect_true(all(df_toer3$toer <= .025*(1.01)))

df_toer3
```

```
##      toer           se
## 1 0.02496 0.0001560033
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sim_pr_reject(opt3_ts$design, .4, datadist)
df_pow3 <- data.frame(
```

```
    pow   = as.numeric(tmp[1]),
    se    = as.numeric(tmp[2])
)
rm(tmp)

testthat::expect_true(all(df_pow3$pow >= .8 * (1 - 0.01)))

df_pow3
```

```
##        pow           se
## 1 0.798916 0.0004008109
```

The expected sample size under the prior should be higher than in the case without the constraint that was analyzed in I.1.

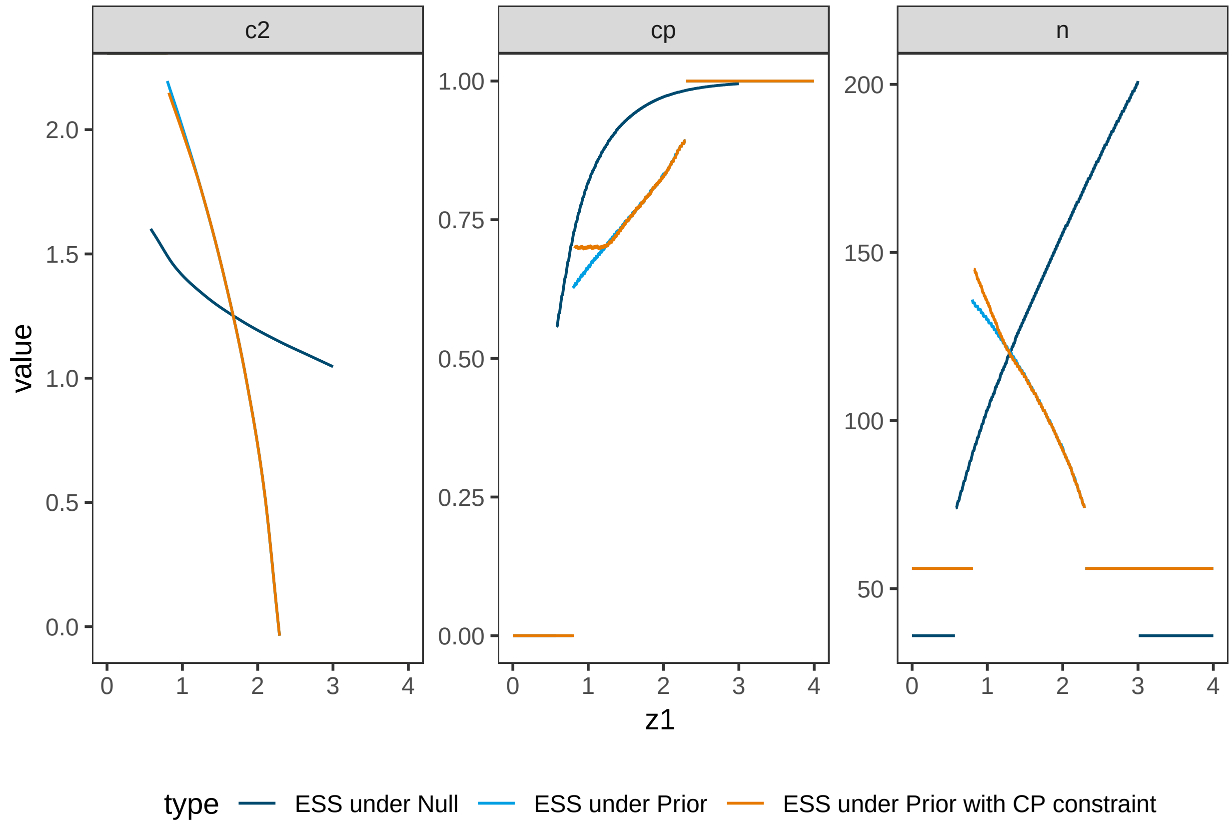
```
testthat::expect_gte(
    evaluate(ess, opt3_ts$design),
    evaluate(ess, opt1_ts$design)
)
```

The conditional power constraint needs to be tested. Select three points for this and check the constraint.

```
x <- adoptr:::scaled_integration_pivots(opt3_ts$design)[c(1, 3, 5)]

cp_val <- sapply(x, function(z) evaluate(cp, opt3_ts$design, z))

testthat::expect_true(all(cp_val >= 0.7))
```

2.5 Plot Two-Stage Designs

The optimal two-stage designs stemming from the different variants are plotted together.


c2
cp
n
value
z1
2.0
1.5
1.0
0.5
0.0
1.00
0.75
0.50
0.25
0.00
200
150
100
50
0
1
2
3
4
type
ESS under Null
ESS under Prior
ESS under Prior with CP constraint

Chapter 3

Scenario II: Large effect, Gaussian prior

3.1 Details

In this scenario a Gaussian prior on the effect size $\delta \sim \mathcal{N}(0.4, 0.2^2)$ is investigated. The null hypothesis is $\delta \leq 0$. Currently, `adoptr` only supports normal distributed data what is widely spread in the development of adaptive designs. We protect the one-sided type one error rate at $\alpha = 0.025$ and require the expected power of the design to be at least $1 - \beta = 0.8$.

3.1.1 Data distribution

Two-armed trial with normally distributed test statistic

```
datadist <- Normal(two_armed = TRUE)
```

3.1.2 Null hypothesis

The null hypothesis is $\mathcal{H}_0 : \delta \leq 0$

```
H_0 <- PointMassPrior(.0, 1)
```

3.1.3 Prior assumptions

A Gaussian prior with mean $\delta = 0.4$ and standard deviation $\tau = .2$ is defined.

```
prior <- ContinuousPrior(function(delta) dnorm(delta, mean = .4, sd = .2),
                         support = c(-5, 5),
                         tighten_support = TRUE)
```

3.2 Variant II-1: Minimizing Expected Sample Size under Point Prior

3.2.1 Objective

Expected sample size under the prior is minimized, i.e., $\boldsymbol{E}\big[n(\mathcal{D})\big]$.

```
ess <- expected(ConditionalSampleSize(datadist, prior))
```

3.2.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis.

```
toer_cnstr <- expected(ConditionalPower(datadist, H_0)) <= .025
```

Expected Power (rejection probability for positive effect sizes) must be larger than 0.8.

```
pow_cnstr <- expected(
    ConditionalPower(datadist, condition(prior, c(0,3)))
    ) >= .8
```

3.2.3 Initial Design

`adoptr` requires the definition of an initial design for optimization. We start with a group-sequential design from the package `rpact` that fulfills the type-one error rate constraint and the power constraint for a point effect size at $\delta = 0.4$. The order of integration is set to 5. For usage as two-stage design with variable sample size, it has to be converted to a `TwoStageDesign`.

```
order <- 5L

init_design_gs <- rpact_design(0.4, 0.025, 0.8, TRUE, order)

init_design    <- TwoStageDesign(init_design_gs)
```

3.2.4 Optimization

The optimal design is computed in three variants: two-stage, group-sequential, and one-stage. The input only differs with regard to the initial design.

```
opt_design <- function(initial_design) {
    minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = initial_design,
```

```
        opts = opts

)
}

opt1_gs <- opt_design(init_design_gs)
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = initial_design, : initial design is infeasible!
```

```
opt1_os <- opt_design(OneStageDesign(300, 2.0))
opt1_ts <- opt_design(TwoStageDesign(opt1_gs$design))
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = initial_design, : initial design is infeasible!
```

3.2.5 Test Cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 1368  449   31
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

Type one error rate constraint is tested for the three designs. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                  function(x) sim_pr_reject(x$design, .0, datadist))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer <= .025*(1.01)))

df_toer
```

```
##       toer           se
## 1 0.024986 0.0001560824
## 2 0.024904 0.0001558327
## 3 0.025116 0.0001564775
```

The expected sample sizes should be ordered in a specific way.

```
testthat::expect_gte(
    evaluate(ess, opt1_os$design),
    evaluate(ess, opt1_gs$design)
)
```

```
testthat::expect_gte(
    evaluate(ess, opt1_gs$design),
    evaluate(ess, opt1_ts$design)
)
```

3.3 Variant II-2: Minimizing Expected Sample Size under Null Hypothesis

3.3.1 Objective

Expected sample size conditioned on negative effect sizes is minimized, i.e.,

```
ess_0 <- expected(ConditionalSampleSize(datadist, condition(prior, c(-3, 0))))
```

3.3.2 Constrains

The constraints remain the same as before.

3.3.3 Initial Design

The previous initial design can still be applied.

3.3.4 Optimization

The optimal group-sequential design and based on this the optimal two-stage design are computed.

```
opt2 <- function(initial_design) {
    minimize(

        ess_0,

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = initial_design,

        opts = opts

)
}

opt2_gs <- opt2(init_design_gs)
```

```
## Warning in minimize(ess_0, subject_to(toer_cnstr, pow_cnstr),
## initial_design = initial_design, : initial design is infeasible!
```

```
opt2_ts <- opt2(TwoStageDesign(opt2_gs$design))
```

```
## Warning in minimize(ess_0, subject_to(toer_cnstr, pow_cnstr),
## initial_design = initial_design, : initial design is infeasible!
```

3.3.5 Test Cases

Check if the optimization algorithm converged.

```
print(opt2_ts$nloptr_return$iterations)
```

```
## [1] 1348
```

```
testthat::expect_true(opt2_ts$nloptr_return$iterations < opts$maxeval)
```

Type one error rate constraint is tested for the optimal design. Due to numerical issues we allow a realtive error of 1%.

```
tmp      <- sim_pr_reject(opt2_ts$design, .0, datadist)
df_toer2 <- data.frame(
    toer = as.numeric(tmp[1]),
    se   = as.numeric(tmp[2])
)
rm(tmp)

testthat::expect_true(all(df_toer2$toer <= .025*(1.01)))

df_toer2
```

```
##       toer           se
## 1 0.024804 0.0001555274
```

The expected sample size under the null hypothesis should be lower than of the design from variant II.1 where expected sample size under the full prior was minimized.

```
testthat::expect_lte(
    evaluate(ess_0, opt2_ts$design),
    evaluate(ess_0, opt1_ts$design)
)
```

3.4 Variant II-3: Conditional Power Constraint

3.4.1 Objective

Expected sample size under the prior is minimized and has already been defined.

3.4.2 Constrains

The constraints remain the same as before, additionally to a constraint on conditional power.

```
cp <- ConditionalPower(datadist, condition(prior, c(0, 3)))

cp_cnstr <- cp >= .7
```

3.4.3 Initial Design

The previous initial design can still be applied.

3.4.4 Optimization

The optimal two-stage design is computed.

```
opt3_ts <- minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr,
            cp_cnstr

        ),

        initial_design = init_design,

        opts = opts

)
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr, cp_cnstr),
## initial_design = init_design, : initial design is infeasible!
```

3.4.5 Test Cases

Check if the optimization algorithm converged.

```
print(opt3_ts$nloptr_return$iterations)
```

```
## [1] 2503
```

```
testthat::expect_true(opt3_ts$nloptr_return$iterations < opts$maxeval)
```

Type one error rate constraint is tested for the optimal design. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sim_pr_reject(opt3_ts$design, .0, datadist)
df_toer3 <- data.frame(
    toer = as.numeric(tmp[1]),
    se   = as.numeric(tmp[2])
)
rm(tmp)

testthat::expect_true(all(df_toer3$toer <= .025*(1.01)))

df_toer3
```

```
##       toer           se
## 1 0.024991 0.0001560976
```

The expected sample size under the prior should be higher than in the case without the constraint that was analyzed in II.1.

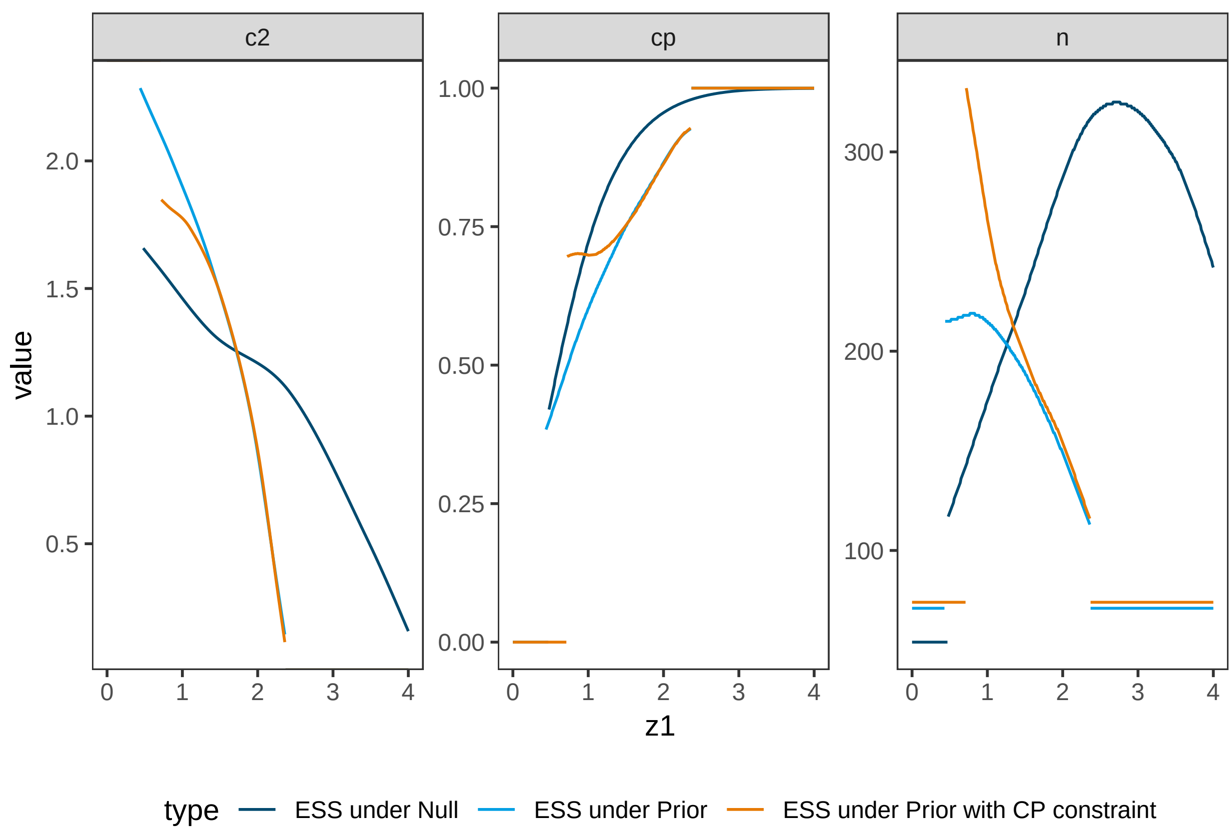
```
testthat::expect_gte(
    evaluate(ess, opt3_ts$design),
    evaluate(ess, opt1_ts$design)
)
```

The conditional power constraint needs to be tested. Select three points for this and check the constraint.

```
x <- adoptr:::scaled_integration_pivots(opt3_ts$design)[c(1, 3, 5)]

cp_val <- sapply(x, function(z) evaluate(cp, opt3_ts$design, z))

testthat::expect_true(all(cp_val >= 0.7))
```

3.5 Plot Two-Stage Designs

The optimal two-stage designs stemming from the different variants are plotted together.



Chapter 4

Scenario III: large effect, uniform prior

4.1 Details

This scenario is a variant of Scenario I. The purpose is to asses whether placing uniform priors with decreasing width of support centered at the $\delta = 0.4$ leads to a sequence of optimal designs which converges towards the solution in Case I-1.

4.1.1 Data distribution

Two-armed trial with normally distributed test statistic

```
datadist <- Normal(two_armed = TRUE)
```

4.1.2 Null hypothesis

The null hypothesis is $\mathcal{H}_0 : \delta \leq 0$

```
H_0 <- PointMassPrior(.0, 1)
```

4.1.3 Prior assumptions

In this scenario we consider a sequence of uniform distributions $\delta \sim \text{Unif}(0.4 - \Delta_i, 0.4 + \Delta_i)$ around 0.4 with $\Delta_i = (3 - i)/10$ for $i = 0 \ldots 3$. I.e., for $\Delta_3 = 0$ reduces to `PointMassPrior` on $\delta = 0.4$.

```
prior <- function(delta) {
    if (delta == 0)
        return(PointMassPrior(.4, 1.0))
    a <- .4 - delta; b <- .4 + delta
    ContinuousPrior(function(x) dunif(x, a, b), support = c(a, b))
}
```

4.2 Variant III.1: Convergence under prior concentration

Make sure that the optimal solution converges as the prior is more and more concentrated at a point mass.

4.2.1 Objective

Expected sample size under the respective prior is minimized, i.e., $\boldsymbol{E}\big[n(\mathcal{D})\big]$.

```
objective <- function(delta) {
    expected(ConditionalSampleSize(datadist, prior(delta)))
}
```

4.2.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis.

```
toer_cstr <- expected(ConditionalPower(datadist, H_0)) <= .025
```

Expected power $\boldsymbol{Pr}\big[c_2(\mathcal{D}, X_1) < X_2 \,|\, \delta \geq 0.0\big]$ must be larger than 0.8.

```
ep_cnstr <- function(delta) {
    prior     <- prior(delta)
    cnd_prior <- condition(prior, c(0, bounds(prior)[2]))
    return( expected(ConditionalPower(datadist, cnd_prior)) >= 0.8 )
}
```

4.2.3 Optimization problem

The optimization problem depending on Δ_i is defined below. The default optimization paramters, 5 pivot points, and a fixed initial design is used. The initial design is chosen such that the error constraints are fulfilled. Early stopping for futility is applied if the effect shows in the opponent direction to the alternative, i.e. $c_1^f = 0$. c_2 is chosen close to and c_1^e a little larger than the $1 - \alpha$-quantile of the standard normal distribution. The sample sizes are selected to fulfill the error constraints.

```
init <- TwoStageDesign(
    n1    = 150,
    c1f   = 0,
    c1e   = 2.3,
    n2    = 125.0,
    c2    = 2.0,
    order = 5
)

optimal_design <- function(delta) {

    minimize(

        objective(delta),

        subject_to(

            toer_cstr,
            ep_cnstr(delta)
```

```
        ),

        initial_design = init

    )
}
```

Compute the sequence of optimal designs

```
deltas  <- 3:0/10
results <- lapply(deltas, optimal_design)
```

4.2.4 Test cases

Check that iteration limit was not exceeded in any case.

```
iters <- sapply(results, function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 1746 1857 2438 2684
```

```
testthat::expect_true(all(iters <= 10000))
```

Check type one error rate control

```
sim_toer <- function(design) {
    simdata <- simulate(
        design,
        nsim  = 10^6,
        dist  = datadist,
        theta =   .0,
        seed  = 42
    )
    return(list(
        toer = mean(simdata$reject),
        se   = sd(simdata$reject) / sqrt(nrow(simdata))
    ))
}

tmp     <- sapply(results, function(x) sim_toer(x$design))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer < .025))

df_toer
```

```
##       toer           se
## 1 0.024979 0.0001560611
## 2 0.024957 0.0001559941
## 3 0.024972 0.0001560398
```

```
## 4 0.024979 0.0001560611
```

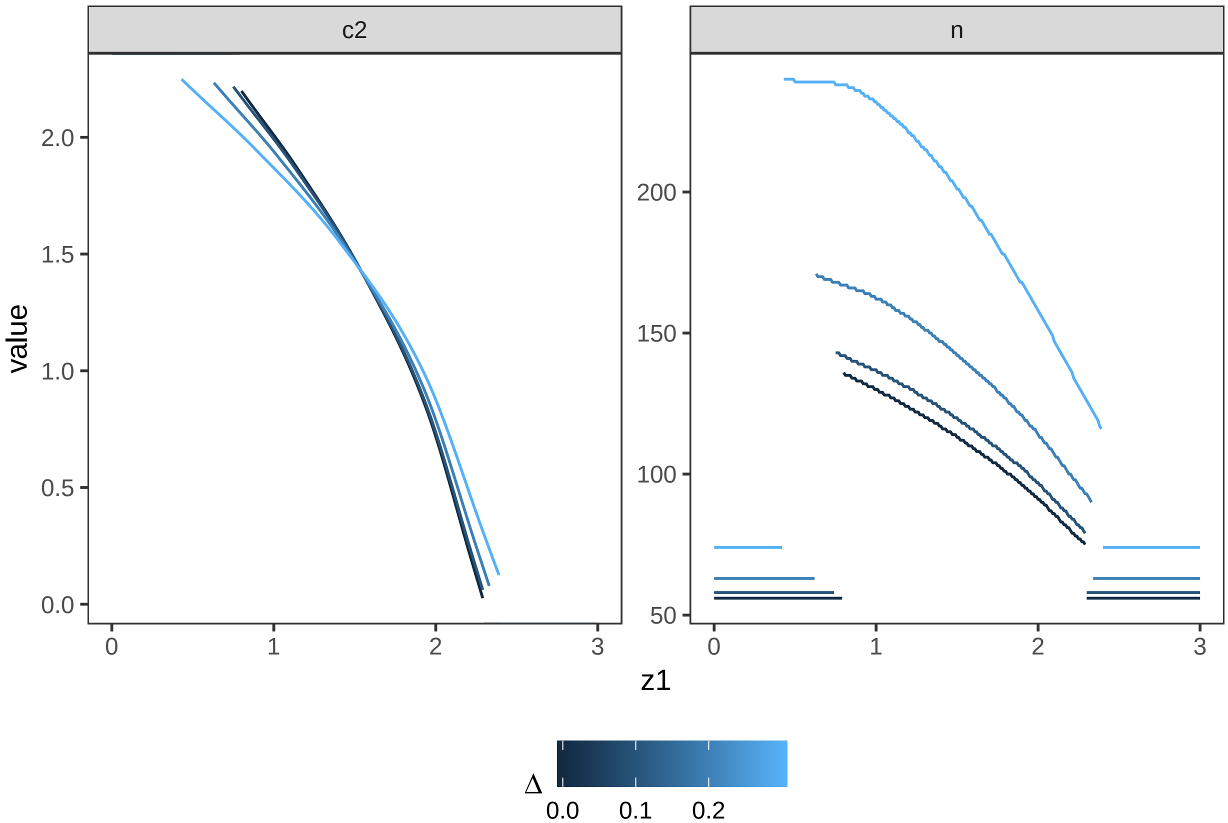
Check that expected sample size decreases with decreasing prior variance.

```r
testthat::expect_gte(
  evaluate(objective(deltas[1]), results[[1]]$design),
  evaluate(objective(deltas[2]), results[[2]]$design)
)

testthat::expect_gte(
  evaluate(objective(deltas[2]), results[[2]]$design),
  evaluate(objective(deltas[3]), results[[3]]$design)
)

testthat::expect_gte(
  evaluate(objective(deltas[3]), results[[3]]$design),
  evaluate(objective(deltas[4]), results[[4]]$design)
)
```

4.2.5 Plot designs

Plot and assess for convergence

```r
z1 <- seq(0, 3, by = .01)

tibble(
    delta  = deltas,
    design = lapply(results, function(x) x$design)
) %>%
    group_by(delta) %>%
    do(
        z1 = z1,
        n  = adoptr::n(.$design[[1]], z1),
        c2 = c2(.$design[[1]], z1)
    ) %>%
    unnest() %>%
    mutate(
        section = ifelse(
            is.finite(c2),
            "continuation",
            ifelse(c2 == -Inf, "efficacy", "futility")
        )
    ) %>%
    gather(variable, value, n, c2) %>%
    ggplot(aes(z1, value, color = delta)) +
        geom_line(aes(group = interaction(section, delta))) +
        facet_wrap(~variable, scales = "free_y") +
        theme_bw() +
        scale_color_continuous(bquote(Delta)) +
        theme(
            panel.grid = element_blank(),
            legend.position = "bottom"
        )
```


c2
n
value
2.0
1.5
1.0
0.5
0.0
200
150
100
50
0
1
2
3
z1
Δ
0.0
0.1
0.2

Chapter 5

Scenario IV: smaller effect, point prior

5.1 Details

In this scenario an alternative effect size of $\delta = 0.2$ with point prior distribution is investigated. This smaller effect size should lead to larger sample sizes than in scenario I. The null hypothesis is $\delta \leq 0$. Currently, `adoptr` only supports normal distributed data what is widely spread in the development of adaptive designs. We protect the one-sided type one error rate at $\alpha = 0.025$ and require the power of the design to be at least $1 - \beta = 0.8$ in the first case and vary these values in the following cases.

5.1.1 Data distribution

Two-armed trial with normally distributed test statistic

```
datadist <- Normal(two_armed = TRUE)
```

5.1.2 Null hypothesis

The null hypothesis is $\mathcal{H}_0 : \delta \leq 0$

```
H_0 <- PointMassPrior(.0, 1)
```

5.1.3 Prior assumptions

A point mass prior with probability mass on $\delta = 0.2$ is assumed.

```
prior <- PointMassPrior(.2, 1)
```

5.2 Variant IV-1: Minimizing Expected Sample Size under Point Prior

5.2.1 Objective

Expected sample size under the respective prior is minimized, i.e., $\boldsymbol{E}\big[n(\mathcal{D})\big]$.

```
ess <- expected(ConditionalSampleSize(datadist, prior))
```

5.2.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis.

```
toer_cnstr <- expected(ConditionalPower(datadist, H_0)) <= .025
```

Power must be larger than 0.8.

```
pow_cnstr <- expected(ConditionalPower(datadist, prior)) >= .8
```

5.2.3 Initial Design

`adoptr` requires the definition of an initial design for optimization. We start with a group-sequential design from the package `rpact` that fulfills these constraints and is used later for comparison. The order of integration is set to 5.

```
order <- 5L

init_design_gs <- rpact_design(0.2, 0.025, 0.8, TRUE, order)
```

5.2.4 Optimization

The optimal design is computed in three variants: two-stage, group-sequential and one-stage. The input only differs with regard to the initial design. The optimal group-sequential design is used as initial design to compute the optimal two-stage design.

```
opt_design <- function(initial_design) {
    minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = initial_design,

        opts = opts

)
}

opt1_gs <- opt_design(init_design_gs)
opt1_ts <- opt_design(TwoStageDesign(opt1_gs$design))
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = initial_design, : initial design is infeasible!
```

```
opt1_os <- opt_design(OneStageDesign(500, 2.0))
```

5.2.5 Test Cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 1960  912   20
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

The n_2 function of the optimal two-stage design is expected to be monotonously decreasing.

```
testthat::expect_equal(
    sign(diff(opt1_ts$design@n2_pivots)),
    rep(-1, (order - 1))
)
```

Type one error rate constraint is tested for the three designs. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                  function(x) sim_pr_reject(x$design, .0, datadist))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer <= .025*(1.01)))

df_toer
```

```
##       toer           se
## 1 0.024975 0.0001560489
## 2 0.024978 0.0001560581
## 3 0.025116 0.0001564775
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt1_ts, opt1_gs, opt1_os),
                  function(x) sim_pr_reject(x$design, .2, datadist))
df_pow <- data.frame(
    pow  = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_pow$pow >= .8 * (1 - 0.01)))

df_pow
```

```
##        pow           se
## 1 0.799799 0.0004001509
## 2 0.799670 0.0004002475
## 3 0.799317 0.0004005115
```

The expected sample sizes should be ordered in a specific way.

```
testthat::expect_gte(
    evaluate(ess, opt1_os$design),
    evaluate(ess, opt1_gs$design)
)

testthat::expect_gte(
    evaluate(ess, init_design_gs),
    evaluate(ess, opt1_gs$design)
)

testthat::expect_gte(
    evaluate(ess, opt1_gs$design),
    evaluate(ess, opt1_ts$design)
)
```

The expected sample size of the optimal designs is simulated and compared to the outomce of `adoptr::evaluate()`. The tolerance is set to 0.5 what is due to rounding one patient per group in the worst case.

```
ess_0 <- expected(ConditionalSampleSize(datadist, H_0))

testthat::expect_equal(
    sim_n(opt1_os$design, .0, datadist),
    evaluate(ess_0, opt1_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_gs$design, .0, datadist),
    evaluate(ess_0, opt1_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_ts$design, .0, datadist),
    evaluate(ess_0, opt1_ts$design),
    tolerance = .5
)
```

Additionally, the sample sizes under the point prior are compared.

```
testthat::expect_equal(
    sim_n(opt1_os$design, .2, datadist),
    evaluate(ess, opt1_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_gs$design, .2, datadist),
```

```
    evaluate(ess, opt1_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt1_ts$design, .2, datadist),
    evaluate(ess, opt1_ts$design),
    tolerance = .5
)
```

5.3 Variant IV-2: Increase Power

5.3.1 Objective

The objective remains the same as before.

5.3.2 Constrains

The power is increased to 90%.

```
pow_cnstr_2 <- expected(ConditionalPower(datadist, prior)) >= .9
```

5.3.3 Initial Design

The initial design is updated to a group-sequential design that fulfills the new power constraint.

```
order <- 5L

init_design_2_gs <- rpact_design(0.2, 0.025, 0.9, TRUE, order)

init_design_2    <- TwoStageDesign(init_design_2_gs)
```

5.3.4 Optimization

The optimal two-stage design is computed.

```
opt_design <- function(initial_design) {
    minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr_2

        ),

        initial_design = initial_design,
```

```
        opts = opts

)
}

opt2_ts <- opt_design(init_design_2)
opt2_gs <- opt_design(init_design_2_gs)
opt2_os <- opt_design(OneStageDesign(500, 2.0))
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr_2),
## initial_design = initial_design, : initial design is infeasible!
```

5.3.5 Test Cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(list(opt2_ts, opt2_gs, opt2_os),
                function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 3008 1168   30
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

The n_2 function of the optimal two-stage design is expected to be monotonously decreasing.

```
testthat::expect_equal(
    sign(diff(opt2_ts$design@n2_pivots)),
    rep(-1, (order - 1))
)
```

Type one error rate constraint is tested for the three designs. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt2_ts, opt2_gs, opt2_os),
                  function(x) sim_pr_reject(x$design, .0, datadist))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer <= .025*(1.01)))

df_toer
```

```
##       toer           se
## 1 0.024980 0.0001560642
## 2 0.024946 0.0001559606
## 3 0.025116 0.0001564775
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt2_ts, opt2_gs, opt2_os),
                  function(x) sim_pr_reject(x$design, .2, datadist))
```

```
df_pow <- data.frame(
    pow  = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_pow$pow >= .9 * (1 - 0.01)))

df_pow
```

```
##        pow           se
## 1 0.900126 0.0002998321
## 2 0.899828 0.0003002293
## 3 0.899523 0.0003006351
```

The expected sample sizes should be ordered in a specific way.

```
testthat::expect_gte(
    evaluate(ess, opt2_os$design),
    evaluate(ess, opt2_gs$design)
)

testthat::expect_gte(
    evaluate(ess, init_design_2_gs),
    evaluate(ess, opt2_gs$design)
)

testthat::expect_gte(
    evaluate(ess, opt2_gs$design),
    evaluate(ess, opt2_ts$design)
)
```

The expected sample size of the optimal designs is simulated and compared to the outomce of `adoptr::evaluate()`. The tolerance is set to 0.5 what is due to rounding one patient per group in the worst case.

```
ess_0 <- expected(ConditionalSampleSize(datadist, H_0))

testthat::expect_equal(
    sim_n(opt2_os$design, .0, datadist),
    evaluate(ess_0, opt2_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt2_gs$design, .0, datadist),
    evaluate(ess_0, opt2_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt2_ts$design, .0, datadist),
    evaluate(ess_0, opt2_ts$design),
    tolerance = .5
)
```

Additionally, the sample sizes under the point prior are compared.

```
testthat::expect_equal(
    sim_n(opt2_os$design, .2, datadist),
    evaluate(ess, opt2_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt2_gs$design, .2, datadist),
    evaluate(ess, opt2_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt2_ts$design, .2, datadist),
    evaluate(ess, opt2_ts$design),
    tolerance = .5
)
```

5.4 Variant IV-3: Increase Type One Error rate

5.4.1 Objective

Expected sample size under the point prior is minimized and has already been defined.

5.4.2 Constrains

The maximal type one error rate is increased to 5%.

```
toer_cnstr_2 <- expected(ConditionalPower(datadist, H_0)) <= .05
```

5.4.3 Initial Design

The initial design is updated to a group-sequential design that fulfills the new type one error rate constraint.

```
order <- 5L

init_design_3_gs <- rpact_design(0.2, 0.05, 0.9, TRUE, order)

init_design_3    <- TwoStageDesign(init_design_3_gs)
```

5.4.4 Optimization

The optimal two-stage design is computed.

```
opt_design <- function(initial_design) {
    minimize(

        ess,
```

```
        subject_to(

            toer_cnstr_2,
            pow_cnstr_2

        ),

        initial_design = initial_design,

        opts = opts

)
}

opt3_ts <- opt_design(init_design_3)
opt3_gs <- opt_design(init_design_3_gs)
opt3_os <- opt_design(OneStageDesign(500, 2.0))
```

```
## Warning in minimize(ess, subject_to(toer_cnstr_2, pow_cnstr_2),
## initial_design = initial_design, : initial design is infeasible!
```

5.4.5 Test Cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(list(opt3_ts, opt3_gs, opt3_os),
                function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 3294  880   27
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

The n_2 function of the optimal two-stage design is expected to be monotonously decreasing.

```
testthat::expect_equal(
    sign(diff(opt3_ts$design@n2_pivots)),
    rep(-1, (order - 1))
)
```

Type one error rate constraint is tested for the three designs. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt3_ts, opt3_gs, opt3_os),
                  function(x) sim_pr_reject(x$design, .0, datadist))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer <= .05*(1.01)))

df_toer
```

```
##       toer           se
## 1 0.050175 0.0002183060
## 2 0.049981 0.0002179058
## 3 0.050150 0.0002182545
```

The power constraint can also be tested via simulation. Due to numerical issues we allow a realtive error of 1%.

```
tmp     <- sapply(list(opt3_ts, opt3_gs, opt3_os),
                  function(x) sim_pr_reject(x$design, .2, datadist))
df_pow <- data.frame(
    pow  = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_pow$pow >= .9 * (1 - 0.01)))

df_pow
```

```
##        pow           se
## 1 0.900057 0.0002999241
## 2 0.900318 0.0002995757
## 3 0.899606 0.0003005248
```

The expected sample sizes should be ordered in a specific way.

```
testthat::expect_gte(
    evaluate(ess, opt3_os$design),
    evaluate(ess, opt3_gs$design)
)

testthat::expect_gte(
    evaluate(ess, init_design_3_gs),
    evaluate(ess, opt3_gs$design)
)

testthat::expect_gte(
    evaluate(ess, opt3_gs$design),
    evaluate(ess, opt3_ts$design)
)
```

The expected sample size of the optimal designs is simulated and compared to the outomce of `adoptr::evaluate()`. The tolerance is set to 0.5 what is due to rounding one patient per group in the worst case.

```
ess_0 <- expected(ConditionalSampleSize(datadist, H_0))

testthat::expect_equal(
    sim_n(opt3_os$design, .0, datadist),
    evaluate(ess_0, opt3_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt3_gs$design, .0, datadist),
    evaluate(ess_0, opt3_gs$design),
```

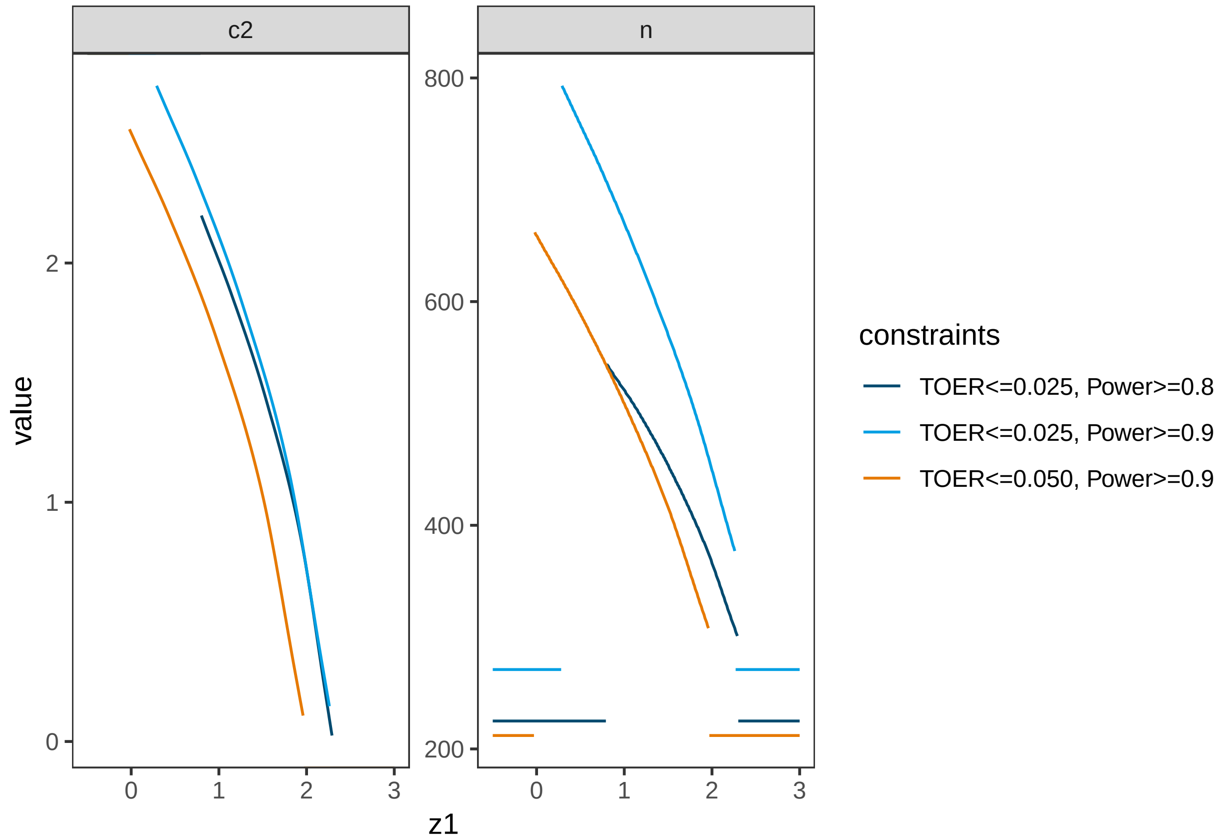
```r
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt3_ts$design, .0, datadist),
    evaluate(ess_0, opt3_ts$design),
    tolerance = .5
)
```

Additionally, the sample sizes under the point prior are compared.

```r
testthat::expect_equal(
    sim_n(opt3_os$design, .2, datadist),
    evaluate(ess, opt3_os$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt3_gs$design, .2, datadist),
    evaluate(ess, opt3_gs$design),
    tolerance = .5
)

testthat::expect_equal(
    sim_n(opt3_ts$design, .2, datadist),
    evaluate(ess, opt3_ts$design),
    tolerance = .5
)
```

5.5 Plot Two-Stage Designs

The optimal two-stage designs stemming from the three different variants are plotted together.


c2
n
value
z1
0
1
2
3
200
400
600
800
constraints
TOER<=0.025, Power>=0.8
TOER<=0.025, Power>=0.9
TOER<=0.050, Power>=0.9

Chapter 6

Scenario V: Single-arm design, medium effect size

6.1 Details

In this scenario an alternative effect size of $\delta = 0.3$ with point prior distribution is investigated. This smaller effect size should lead to larger sample sizes than in scenario I. The null hypothesis is $\delta \leq 0$. Currently, `adoptr` only supports normal distributed data what is widely spread in the development of adaptive designs.

6.1.1 Data distribution

One-armed trial with normally distributed test statistic

```
datadist <- Normal(two_armed = FALSE)
```

6.1.2 Null hypothesis

The null hypothesis is $\mathcal{H}_0 : \delta \leq 0$

```
H_0 <- PointMassPrior(.0, 1)
```

6.1.3 Prior assumptions

A point mass prior with probability mass on $\delta = 0.3$ is assumed.

```
prior <- PointMassPrior(.3, 1)
```

6.2 Variant V-1, sensitivity to integration order

6.2.1 Objective

Expected sample size under the respective prior is minimized, i.e., $\boldsymbol{E}\big[n(\mathcal{D})\big]$.

```
ess <- expected(ConditionalSampleSize(datadist, prior))
```

6.2.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis.

```
toer_cnstr <- expected(ConditionalPower(datadist, H_0)) <= .025
```

Power must be larger than 0.8.

```
pow_cnstr <- expected(ConditionalPower(datadist, prior)) >= .8
```

6.2.3 Initial Design

A fixed design for these parameters would require 176 subjects per group. We use the half of this as initial values for the sample sizes. The initial stop for futility is at $c_1^f = 0$, i.e., if the effect shows in the opponent direction to the alternative. The starting values for the efficacy stop and for c_2 is the $1 - \alpha$- quantile of the normal distribution.

```
init_design <- function(order) {
    TwoStageDesign(
        n1 = ceiling(pwr::pwr.t.test(d = .3, sig.level = .025, power = .8, alternative = "greater")$n)
        c1f = 0,
        c1e = qnorm( 1 - 0.025),
        n2 = ceiling(pwr::pwr.t.test(d = .3, sig.level = .025, power = .8, alternative = "greater")$n)
        c2 = qnorm(1 - 0.025),
        order = order
)
}
```

6.2.4 Optimization

The optimal design is computed for three different integration orders: 5, 8, and 11.

```
opt_design <- function(order) {
    minimize(

        ess,

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = init_design(order),

        opts = opts

)
}
```

```
opt1 <- lapply(c(5, 8, 11), function(x) opt_design(x))
```

```
## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = init_design(order), : initial design is infeasible!

## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = init_design(order), : initial design is infeasible!

## Warning in minimize(ess, subject_to(toer_cnstr, pow_cnstr), initial_design
## = init_design(order), : initial design is infeasible!
```

6.2.5 Test cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(opt1, function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 2044 4660 8826
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

Check type one error rate control. Due to numerical issues we allow a realtive error of 1%.

```
sim_toer <- function(design) {
    simdata <- simulate(
        design,
        nsim  = 10^6,
        dist  = datadist,
        theta =   .0,
        seed  = 42
    )
    return(list(
        toer = mean(simdata$reject),
        se   = sd(simdata$reject) / sqrt(nrow(simdata))
    ))
}

tmp     <- sapply(opt1, function(x) sim_toer(x$design))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer < .025 * 1.01))

df_toer
```

```
##       toer           se
## 1 0.024978 0.0001560581
## 2 0.024955 0.0001559881
## 3 0.024951 0.0001559759
```

Check the power constraint. For numerical reasons we allow a realtive error of 1%.

```
sim_pow <- function(design) {
    simdata <- simulate(
        design,
        nsim  = 10^6,
        dist  = datadist,
        theta =   .3,
        seed  = 42
    )
    return(list(
        pow = mean(simdata$reject),
        se  = sd(simdata$reject) / sqrt(nrow(simdata))
    ))
}

tmp     <- sapply(opt1, function(x) sim_pow(x$design))
df_pow <- data.frame(
    power = as.numeric(tmp[1, ]),
    se    = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_pow$pow > 0.8 * (1 - .01)))

df_pow
```

```
##      power           se
## 1 0.799817 0.0004001374
## 2 0.799668 0.0004002490
## 3 0.799646 0.0004002655
```

Check expected sample size under the prior.

```
sim_ess <- function(design) {
    simdata <- simulate(
        design,
        nsim  = 10^6,
        dist  = datadist,
        theta =   .3,
        seed  = 42
    )
    return(list(
        n  = mean(simdata$n1 + simdata$n2),
        se = sd(simdata$n1 + simdata$n2) / sqrt(nrow(simdata))
    ))
}

tmp     <- sapply(opt1, function(x) sim_ess(x$design))
df_ess <- data.frame(
    n  = as.numeric(tmp[1, ]),
    se = as.numeric(tmp[2, ])
)
rm(tmp)

df_ess
```

```
##          n         se
## 1 70.98114 0.02437353
## 2 70.99017 0.02438000
## 3 70.99167 0.02437997
```

6.3 Variant V-2, utility maximization

6.3.1 Objective

In this case, a utility function consisting of expected sample size and power is minimized.

```
pow <- expected(ConditionalPower(datadist, prior))

obj <- function(lambda) {
    expected(ConditionalSampleSize(datadist, prior)) +
        (-lambda) * pow
}
```

6.3.2 Constrains

The type one error rate is controlled at 0.025 on the boundary of the null hypothesis. Hence, the previous inequality can still be used.

6.3.3 Initial Design

The previous initial design with order 5 is applied.

6.3.4 Optimization

The optimal design is computed for two values of λ: 100 and 200.

```
opt2_design <- function(lambda) {

    minimize(

        obj(lambda),

        subject_to(

            toer_cnstr

        ),

        initial_design = init_design(5),

        opts = opts

)
}
```

```
opt2 <- lapply(c(100, 200), function(x) opt2_design(x))
```

```
## Warning in minimize(obj(lambda), subject_to(toer_cnstr), initial_design =
## init_design(5), : initial design is infeasible!

## Warning in minimize(obj(lambda), subject_to(toer_cnstr), initial_design =
## init_design(5), : initial design is infeasible!
```

6.3.5 Test cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(opt2, function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 3378 3024
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

Check type one error rate control for both designs via simulation. Due to numerical issues we allow a realtive error of 1%.

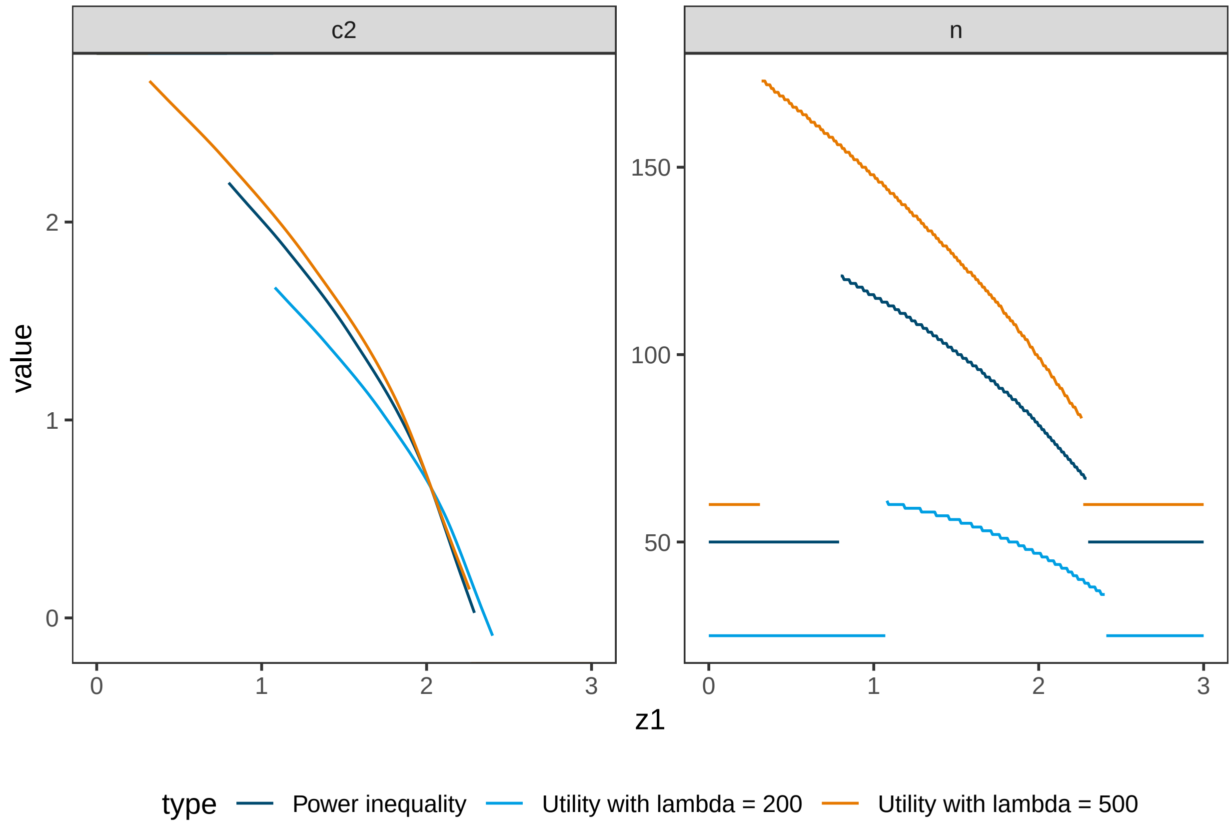
```
tmp     <- sapply(opt2, function(x) sim_toer(x$design))
df_toer <- data.frame(
    toer = as.numeric(tmp[1, ]),
    se   = as.numeric(tmp[2, ])
)
rm(tmp)

testthat::expect_true(all(df_toer$toer < .025 * 1.01))

df_toer
```

```
##       toer           se
## 1 0.025022 0.0001561919
## 2 0.024971 0.0001560368
```

Check if the power of the design with higher λ is larger.

```
testthat::expect_gte(
    evaluate(pow, opt2[[2]]$design),
    evaluate(pow, opt2[[1]]$design)
)
```

Finally the three designs computed so far are plotted together to allow comparison.



6.4 Variant V-3, n1-penalty

In this case, the influence of the regularization term `N1()` is investigated.

6.4.1 Objective

In this case, a mixed criterion consisting of expected sample size and n_1 is minimized.

```
N1 <- N1()

obj3 <- function(lambda) {
    ess + lambda * N1
}
```

6.4.2 Constrains

The inequalities from variant V.1 can still be used.

6.4.3 Initial Design

The previous initial design with order 5 is applied.

6.4.4 Optimization

The optimal design is computed for two values of λ: 0.05 and 0.2.

```
opt3_design <- function(lambda) {

    minimize(

        obj3(lambda),

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = init_design(5),

        opts = opts

)
}

opt3 <- lapply(c(.05, .2), function(x) opt3_design(x))
```

```
## Warning in minimize(obj3(lambda), subject_to(toer_cnstr, pow_cnstr),
## initial_design = init_design(5), : initial design is infeasible!

## Warning in minimize(obj3(lambda), subject_to(toer_cnstr, pow_cnstr),
## initial_design = init_design(5), : initial design is infeasible!
```

6.4.5 Test cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(opt3, function(x) x$nloptr_return$iterations)

print(iters)
```

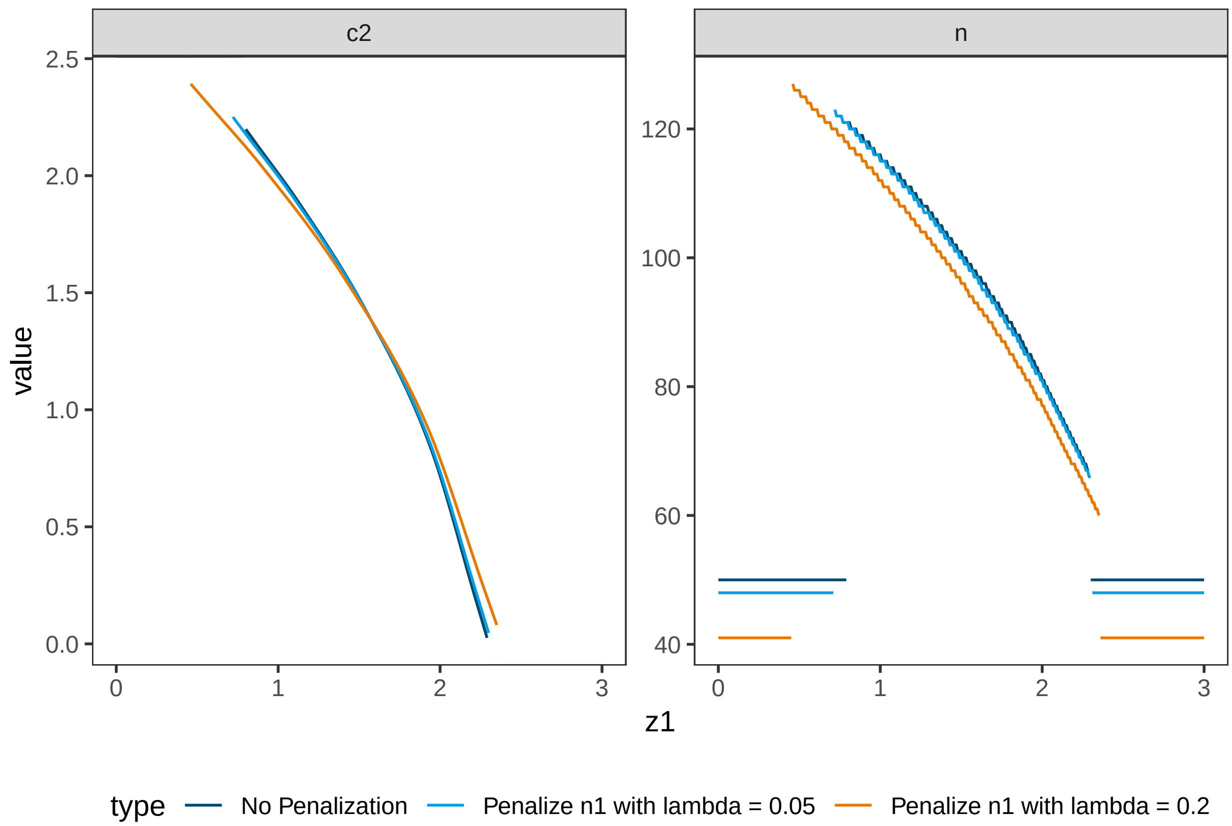
```
## [1] 2222 2222
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

Check if the n1 regularizer of the design with higher λ is lower.

```
testthat::expect_lte(
    evaluate(N1, opt3[[2]]$design),
    evaluate(N1, opt3[[1]]$design)
)


testthat::expect_lte(
    evaluate(N1, opt3[[1]]$design),
    evaluate(N1, opt1[[1]]$design)
)
```

Finally the three designs computed so far are plotted together to allow comparison.



6.5 Variant V-4, n2-penalty

In this case the average over n_2 is penalized by the predefined score `AverageN2`.

6.5.1 Objective

In this case, a mixed criterion consisting of expected sample size and average of n_2 is minimized.

```
avn2 <- AverageN2()

obj4 <- function(lambda) {
    ess + lambda * avn2
}
```

6.5.2 Constrains

The inequalities from variant V.1 can still be used.

6.5.3 Initial Design

The previous initial design with order 5 is applied.

6.5.4 Optimization

The optimal design is computed for two values of λ: 0.01 and 0.1.

```
opt4_design <- function(lambda) {

    minimize(

        obj4(lambda),

        subject_to(

            toer_cnstr,
            pow_cnstr

        ),

        initial_design = init_design(5),
        upper_boundary_design = get_upper_boundary_design(init_design(5), c2_buffer=3),

        opts = opts

)
}

opt4 <- lapply(c(.01, .1), function(x) opt4_design(x))
```

```
## Warning in minimize(obj4(lambda), subject_to(toer_cnstr, pow_cnstr),
## initial_design = init_design(5), : initial design is infeasible!

## Warning in minimize(obj4(lambda), subject_to(toer_cnstr, pow_cnstr),
## initial_design = init_design(5), : initial design is infeasible!
```

6.5.5 Test cases

Check if the optimization algorithm converged in all cases.

```
iters <- sapply(opt4, function(x) x$nloptr_return$iterations)

print(iters)
```

```
## [1] 1952 1530
```

```
testthat::expect_true(all(iters < opts$maxeval))
```

Check if the average n_2 regularizer of the design with higher λ is lower.

```
testthat::expect_lte(
    evaluate(avn2, opt4[[2]]$design),
    evaluate(avn2, opt4[[1]]$design)
)
```

```
testthat::expect_lte(
    evaluate(avn2, opt4[[1]]$design),
    evaluate(avn2, opt1[[1]]$design)
)
```

Finally the three designs computed so far are plotted together to allow comparison.