

# Electronic Supplementary Material (ESM)

## Sequential randomization tests using e-values for clinical trial monitoring across endpoint types

Fernando G Zampieri, Arthur M Albuquerque, Aaditya Ramdas

### ESM Section 1. Additional details on e-RTb

This section collects the supplementary material for the binary-endpoint construction: wager-policy comparisons, sensitivity to the tuning parameters, and validity checks under restricted and unequal randomization.

#### 1.1 Wager policies: efficiency, null behavior, and effect magnification

Four wager policies are compared on the same simulated binary trial streams: the adaptive half-scale policy, and design wagers that understate, match, or overstate the true absolute risk reduction (ARR). Scenarios use fixed-sample designs powered at 80% or 90% for true ARR of 5 or 10 percentage points; an oracle wager that knows the true ARR is included as the efficiency ceiling. Assignments are drawn by simple randomization at probability 0.5 (no blocks), so the constant denominator is exact; this applies to 1.1 and 1.2. Wager policy changes how quickly evidence accumulates; null validity does not depend on it.

**Table S1.** Empirical null crossing rates for e-RTb wager policies. These are e-process crossing rates for  $W \geq 20$ , not combined OBF/e-RT familywise rejection rates.

| Design ARR | Target power | N     | Policy              | Wager ARR | Type I error | SE    |
|------------|--------------|-------|---------------------|-----------|--------------|-------|
| 5.0 pp     | 80%          | 2,942 | Adaptive half-scale | —         | 3.8%         | 0.27% |
| 5.0 pp     | 80%          | 2,942 | Design under        | 2.5 pp    | 0.7%         | 0.11% |
| 5.0 pp     | 80%          | 2,942 | Design matched      | 5.0 pp    | 4.0%         | 0.28% |
| 5.0 pp     | 80%          | 2,942 | Design over         | 10.0 pp   | 4.6%         | 0.30% |
| 5.0 pp     | 90%          | 3,938 | Adaptive half-scale | —         | 3.5%         | 0.26% |
| 5.0 pp     | 90%          | 3,938 | Design under        | 2.5 pp    | 0.9%         | 0.13% |
| 5.0 pp     | 90%          | 3,938 | Design matched      | 5.0 pp    | 3.9%         | 0.27% |
| 5.0 pp     | 90%          | 3,938 | Design over         | 10.0 pp   | 4.6%         | 0.30% |
| 10.0 pp    | 80%          | 712   | Adaptive half-scale | —         | 2.1%         | 0.20% |
| 10.0 pp    | 80%          | 712   | Design under        | 5.0 pp    | 0.5%         | 0.10% |
| 10.0 pp    | 80%          | 712   | Design matched      | 10.0 pp   | 3.9%         | 0.27% |
| 10.0 pp    | 80%          | 712   | Design over         | 15.0 pp   | 4.5%         | 0.29% |

| Design ARR | Target power | N   | Policy              | Wager ARR | Type I error | SE    |
|------------|--------------|-----|---------------------|-----------|--------------|-------|
| 10.0 pp    | 90%          | 954 | Adaptive half-scale | –         | 2.2%         | 0.21% |
| 10.0 pp    | 90%          | 954 | Design under        | 5.0 pp    | 0.8%         | 0.12% |
| 10.0 pp    | 90%          | 954 | Design matched      | 10.0 pp   | 3.7%         | 0.27% |
| 10.0 pp    | 90%          | 954 | Design over         | 15.0 pp   | 4.3%         | 0.29% |

**Table S2.** Power, first crossing, and Type M behavior for e-RTb wager policies under alternative scenarios. Type M is the ratio of the absolute effect estimate at first crossing to the absolute final effect estimate among crossed trials.

| True ARR | Target power | N     | Policy                | Wager ARR | Power | Median cross-ing | % N | ARR at cross-ing | Final ARR | Median Type M |
|----------|--------------|-------|-----------------------|-----------|-------|------------------|-----|------------------|-----------|---------------|
| 5.0 pp   | 80%          | 2,942 | Adaptive – half-scale | –         | 48.3% | 1,459            | 50% | 7.9 pp           | 5.0 pp    | 1.59          |
| 5.0 pp   | 80%          | 2,942 | Design under          | 2.5 pp    | 55.5% | 2,111            | 72% | 6.6 pp           | 5.0 pp    | 1.33          |
| 5.0 pp   | 80%          | 2,942 | Design matched        | 5.0 pp    | 75.1% | 1,373            | 47% | 6.6 pp           | 5.0 pp    | 1.33          |
| 5.0 pp   | 80%          | 2,942 | Design over           | 10.0 pp   | 55.7% | 739              | 25% | 8.9 pp           | 5.0 pp    | 1.78          |
| 5.0 pp   | 80%          | 2,942 | Oracle                | 5.0 pp    | 75.1% | 1,373            | 47% | 6.6 pp           | 5.0 pp    | 1.33          |
| 5.0 pp   | 90%          | 3,938 | Adaptive – half-scale | –         | 61.8% | 1,896            | 48% | 7.1 pp           | 4.9 pp    | 1.42          |
| 5.0 pp   | 90%          | 3,938 | Design under          | 2.5 pp    | 78.5% | 2,426            | 62% | 5.9 pp           | 4.9 pp    | 1.19          |
| 5.0 pp   | 90%          | 3,938 | Design matched        | 5.0 pp    | 86.1% | 1,525            | 39% | 6.2 pp           | 4.9 pp    | 1.25          |
| 5.0 pp   | 90%          | 3,938 | Design over           | 10.0 pp   | 60.5% | 838              | 21% | 8.4 pp           | 4.9 pp    | 1.69          |
| 5.0 pp   | 90%          | 3,938 | Oracle                | 5.0 pp    | 86.1% | 1,525            | 39% | 6.2 pp           | 4.9 pp    | 1.25          |
| 10.0 pp  | 80%          | 712   | Adaptive – half-scale | –         | 49.6% | 400              | 56% | 14.7 pp          | 9.9 pp    | 1.47          |
| 10.0 pp  | 80%          | 712   | Design under          | 5.0 pp    | 53.1% | 520              | 73% | 13.3 pp          | 9.9 pp    | 1.33          |
| 10.0 pp  | 80%          | 712   | Design matched        | 10.0 pp   | 74.9% | 336              | 47% | 13.3 pp          | 9.9 pp    | 1.33          |

| True ARR | Target power | N   | Policy                | Wager ARR | Power | Median cross-ing | % N | ARR at cross-ing | Final ARR | Median Type M |
|----------|--------------|-----|-----------------------|-----------|-------|------------------|-----|------------------|-----------|---------------|
| 10.0 pp  | 80%          | 712 | Design over           | 15.0 pp   | 67.5% | 249              | 35% | 15.0 pp          | 9.9 pp    | 1.50          |
| 10.0 pp  | 80%          | 712 | Oracle                | 10.0 pp   | 74.9% | 336              | 47% | 13.3 pp          | 9.9 pp    | 1.33          |
| 10.0 pp  | 90%          | 954 | Adaptive – half-scale | –         | 65.9% | 491              | 51% | 13.4 pp          | 10.1 pp   | 1.34          |
| 10.0 pp  | 90%          | 954 | Design under          | 5.0 pp    | 77.6% | 596              | 62% | 11.9 pp          | 10.1 pp   | 1.19          |
| 10.0 pp  | 90%          | 954 | Design matched        | 10.0 pp   | 86.1% | 372              | 39% | 12.4 pp          | 10.1 pp   | 1.24          |
| 10.0 pp  | 90%          | 954 | Design over           | 15.0 pp   | 76.2% | 274              | 29% | 14.3 pp          | 10.1 pp   | 1.43          |
| 10.0 pp  | 90%          | 954 | Oracle                | 10.0 pp   | 86.1% | 372              | 39% | 12.4 pp          | 10.1 pp   | 1.24          |

Code: e-RTb kernel (SuppCode\_eRTb.R, Section 1.5); seed 20260502 (Tables S1–S2).

**The two-sided mixture in detail.** The mirrored process  $W^{(-)}$  is built by reversing the anticipated direction: for the design wager, swap the anticipated event risks, which at 1:1 replaces  $\lambda^*$  by  $1 - \lambda^*$ ; for the adaptive wager, flip the sign of the learned score. When the effect lies in the anticipated direction the mirrored component decays toward zero, so  $W^{\text{mix}} \approx \frac{1}{2}W^{(+)}$  and reaching the threshold 20 requires the directional component to reach about 40: one extra doubling, the betting analogue of  $\alpha/2$  per side. An unequal pre-specified split ( $w, 1 - w$ ) shifts efficiency toward the favored direction at the cost of slower detection in the other.

## 1.2 Sensitivity to burn-in, ramp, and score multiplier

The table varies burn-in, ramp, and the score multiplier under the 80%-power scenarios. Power and crossing time respond to tuning; null validity does not. Larger multipliers cross earlier when they cross, but power falls — the volatility cost of aggressive betting.

**Table S3.** Adaptive e-RTb tuning sensitivity under 80% fixed-sample design scenarios.

| True ARR | N     | Setting            | Burn-in | Ramp | Score multiplier | Power | Median crossing |
|----------|-------|--------------------|---------|------|------------------|-------|-----------------|
| 5.0 pp   | 2,942 | Default half-scale | 50      | 100  | 0.50             | 47.3% | 1,391           |
| 5.0 pp   | 2,942 | Short burn-in/ramp | 30      | 50   | 0.50             | 44.5% | 1,488           |
| 5.0 pp   | 2,942 | Proportional 5/10% | 147     | 294  | 0.50             | 52.6% | 1,546           |

| True ARR | N     | Setting             | Burn-in | Ramp | Score multiplier | Power | Median crossing |
|----------|-------|---------------------|---------|------|------------------|-------|-----------------|
| 5.0 pp   | 2,942 | Quarter scale       | 50      | 100  | 0.25             | 40.5% | 1,833           |
| 5.0 pp   | 2,942 | Three-quarter scale | 50      | 100  | 0.75             | 38.6% | 1,191           |
| 5.0 pp   | 2,942 | Full scale          | 50      | 100  | 1.00             | 26.4% | 689             |
| 10.0 pp  | 712   | Default half-scale  | 50      | 100  | 0.50             | 52.4% | 387             |
| 10.0 pp  | 712   | Short burn-in/ramp  | 30      | 50   | 0.50             | 48.1% | 364             |
| 10.0 pp  | 712   | Proportional 5/10%  | 36      | 71   | 0.50             | 52.7% | 388             |
| 10.0 pp  | 712   | Quarter scale       | 50      | 100  | 0.25             | 34.1% | 474             |
| 10.0 pp  | 712   | Three-quarter scale | 50      | 100  | 0.75             | 47.0% | 332             |
| 10.0 pp  | 712   | Full scale          | 50      | 100  | 1.00             | 39.1% | 283             |

Code: e-RTb kernel (SuppCode\_eRTb.R, Section 1.5); seed 20260502.

### 1.3 Permuted-block randomization: the conditional $p_i$ versus a naive 0.5

Worked slots in a 1:1 block of four: the first patient gets  $p_i = 2/4 = 0.5$ ; after one treatment assignment the next gets  $1/3$ ; after treatment, control, treatment, the final slot is control with certainty. The monitor adapts by knowing the block size and the patient's position in it.

Three bettors were run under permuted blocks of 4 with no treatment effect ( $N = 2,690$  per trial, null event rate 35%, 5,000 simulated trials). Row 1 is the method as defined in the main text: the adaptive e-RTb wager with the block-aware conditional  $p_i$  and the deterministic-slot skip rule. Row 2 is the same adaptive wager priced as if  $p_i$  were 0.5 at every update: the exposure of running the method unmodified under blocks. Row 3 is an adversarial demonstration, not a realistic monitor: a bettor that never uses an outcome and tracks only block composition, priced at 0.5.

One block shows why row 3 always wins. The wealth update divides by the assumed probability of the observed assignment, and this bettor's updates always divide by 0.5. Suppose the block's assignments turn out treatment, control, treatment, control, and the bettor bets before each reveal using only the block count so far. Patient 1: the block is empty, the true chance of treatment is  $2/4 = 0.5$ , and there is nothing to exploit. Patient 2: one treatment slot is used, so the true chance of treatment is  $1/3$ ; betting that chance against a 0.5 denominator gains on average. Patient 3: the same, with a larger imbalance. Patient 4: after treatment, control, treatment, the last slot must be control. The bettor stakes everything on control, control happens with certainty, and the update multiplies wealth by  $(1 - 0)/(1 - 0.5) = 2$ . Wealth doubles with zero risk, using no outcome. Every block of four ends in at least one forced slot, so wealth at least doubles every four patients: 1, 2, 4, 8, 16, 32. With the additional gains from the in-between slots, the median crossing lands at

patient 14, and every null run crosses, compounding to about  $10^{285}$  by the end of the stream. The fix puts the true conditional probability in the denominator: the forced slot then multiplies wealth by  $1/1 = 1$ , and there is nothing to win. Equivalently, the forced slot is skipped.

**Table S4.** Null crossing behavior under permuted blocks of 4 for three bettors.

| Bettor                                                                  | Null crossing rate<br>( $W \geq 20$ ) | Median first<br>crossing | Median peak W | 90th pct peak W |
|-------------------------------------------------------------------------|---------------------------------------|--------------------------|---------------|-----------------|
| Adaptive e-RTb<br>wagers, correct<br>conditional $p_i$ +<br>skip rule   | 2.2% (MC SE<br>0.21pp)                | 600                      | 1.58          | 5.42            |
| Adaptive e-RTb<br>wagers, naive<br>$p=0.5$ (practical<br>exposure)      | 2.6% (MC SE<br>0.23pp)                | 596                      | 1.63          | 6.05            |
| History-only<br>bettor, naive $p=0.5$<br>(adversarial<br>demonstration) | 100.0% (MC SE<br>0.00pp)              | 14                       | 9.62e+285     | 9.82e+285       |

Code: self-contained wealth loop as described above, mirroring the SuppCode\_eRTb.R wager defaults (Section 1.5); seed 20260545.

#### 1.4 Unequal allocation: a 2:1 check

Under unequal allocation the adaptive e-RTb policy generalizes to center its bet at the allocation probability  $p$  rather than 0.5, betting  $\lambda_i = p \pm 0.5 c_i \hat{\delta}_{i-1}$ , so that placing no bet again means  $\lambda_i = p$ . This is the theoretical generalization; the kernel shipped with this paper (Section 1.5) keeps the 0.5 centering, as stated below.

$N = 2,690, 5,000$  simulated trials per cell. Assignments are drawn by simple randomization at probability  $2/3$  (unequal, but constant, with no blocks). Wealth updates divide by the true allocation probabilities ( $p = 2/3$ ,  $1 - p = 1/3$ ). Any predictable wager is valid at any allocation. The adaptive wager as shipped centers its bet at 0.5 rather than at  $p$ , so its power under 2:1 is not representative; the generalized design wager row is the informative one.

**Table S5.** e-RTb null validity and design power at 2:1 allocation.

| Scenario                   | Allocation      | Method                                              | Crossing rate ( $W \geq 20$ ) |
|----------------------------|-----------------|-----------------------------------------------------|-------------------------------|
| Null: 35% vs 35%           | 2:1 ( $p=2/3$ ) | e-RTb design<br>(generalized lambda*)               | 3.9% (0.27pp)                 |
| Null: 35% vs 35%           | 2:1 ( $p=2/3$ ) | e-RTb adaptive (as<br>shipped; 0.5-centered<br>bet) | 4.3% (0.29pp)                 |
| Alternative: 35% vs<br>30% | 2:1 ( $p=2/3$ ) | e-RTb design<br>(generalized lambda*)               | 69.8% (0.65pp)                |

| Scenario                | Allocation      | Method                                        | Crossing rate ( $W \geq 20$ ) |
|-------------------------|-----------------|-----------------------------------------------|-------------------------------|
| Alternative: 35% vs 30% | 2:1 ( $p=2/3$ ) | e-RTb adaptive (as shipped; 0.5-centered bet) | 4.2% (0.28pp)                 |

Validity holds in every row; the design wager carries the power under unequal allocation.

Code: e-RTb kernel (SuppCode\_eRTb.R, Section 1.5) with allocation parameter  $p = 2/3$ ; seed 20260549.

## 1.5 Code for e-RTb

The listing below is the complete e-RTb kernel (SuppCode\_eRTb.R), the implementation behind every simulation in this section. Each subsection above states the seed that drove it; the three kernel files, with every simulation script and frozen output, are at <https://github.com/fzampier/ert-methods-paper> (archived at <https://doi.org/10.5281/zenodo.22906388>).

Two implementation notes (audit, 2026-08-13). First, `compute_eRT()` initializes  $W_1 = 1$  and places its first bet on patient 2; patient 1's outcome-assignment pair enters the history only. For adaptive wagers this is inside the burn-in and has no effect; for design wagers it forgoes one early bet. A replay on the frozen comparator streams shifts design power by about 0.1 percentage point, below every reported rounding. All reported design-wager results are for this as-shipped rule. Second, the kernels take a constant allocation probability: they implement simple randomization at any fixed  $p$ . The conditional- $p_i$  rule for restricted designs (main-text deployment requirement) is a few-line modification of the wealth loop, demonstrated in the blocked-randomization simulation of Section 1.3; stratified and covariate-adaptive variants follow the same recipe but are not shipped as kernels.

```
# =====
# e-RT Simulations: Type I Error and Power (Unified)
# =====

library(tidyverse)

# -----
# Core e-RT function
# -----

binary_design_lambdas <- function(p_trt, p_ctrl, p_random = 0.5) {
  event_denom <- p_random * p_trt + (1 - p_random) * p_ctrl
  nonevent_denom <- p_random * (1 - p_trt) + (1 - p_random) * (1 - p_ctrl)

  if (event_denom <= 0 || nonevent_denom <= 0) {
    stop("Design rates imply a degenerate event or non-event denominator")
  }

  list(
    event = p_random * p_trt / event_denom,
    nonevent = p_random * (1 - p_trt) / nonevent_denom
  )
}
```

```

)
}

compute_eRT <- function(treatment, outcome, p = 0.5, burn_in = 50, ramp = 100,
                        wager = c("adaptive", "design", "fixed", "oracle"),
                        p_trt_design = NULL, p_ctrl_design = NULL,
                        p_trt_oracle = NULL, p_ctrl_oracle = NULL,
                        fixed_event_lambda = NULL,
                        fixed_nonevent_lambda = NULL,
                        ramp_fixed = TRUE,
                        kelly_fraction = 0.5) {
  n <- length(treatment)
  wager <- match.arg(wager)

  if (is.factor(treatment)) treatment <- as.numeric(treatment) - 1
  if (is.factor(outcome)) outcome <- as.numeric(outcome) - 1

  if (wager == "design" &&
      (is.null(p_trt_design) || is.null(p_ctrl_design))) {
    stop("Design wager requires p_trt_design and p_ctrl_design")
  }

  if (wager == "oracle" &&
      (is.null(p_trt_oracle) || is.null(p_ctrl_oracle))) {
    stop("Oracle wager requires p_trt_oracle and p_ctrl_oracle")
  }

  if (wager == "fixed" &&
      (is.null(fixed_event_lambda) || is.null(fixed_nonevent_lambda))) {
    stop("Fixed wager requires fixed_event_lambda and fixed_nonevent_lambda")
  }

  if (wager == "design") {
    design_lambdas <- binary_design_lambdas(p_trt_design, p_ctrl_design, p)
  } else if (wager == "oracle") {
    design_lambdas <- binary_design_lambdas(p_trt_oracle, p_ctrl_oracle, p)
  } else if (wager == "fixed") {
    design_lambdas <- list(event = fixed_event_lambda, nonevent = fixed_nonevent_lambda)
  } else {
    design_lambdas <- NULL
  }

  wealth <- numeric(n)
  wealth[1] <- 1

  n_trt <- as.integer(treatment[1] == 1)
  n_ctrl <- as.integer(treatment[1] == 0)
  events_trt <- if (treatment[1] == 1) outcome[1] else 0

```

```

events_ctrl <- if (treatment[1] == 0) outcome[1] else 0

for (i in 2:n) {
  c_i <- max(0, min(1, (i - burn_in) / ramp))

  if (wager == "adaptive") {
    rate_trt <- if (n_trt > 0) events_trt / n_trt else 0.5
    rate_ctrl <- if (n_ctrl > 0) events_ctrl / n_ctrl else 0.5

    delta_hat <- rate_trt - rate_ctrl

    if (outcome[i] == 1) {
      lambda <- 0.5 + kelly_fraction * c_i * delta_hat
    } else {
      lambda <- 0.5 - kelly_fraction * c_i * delta_hat
    }
  } else {
    lambda_target <- if (outcome[i] == 1) {
      design_lambdas$event
    } else {
      design_lambdas$nonevent
    }

    if (ramp_fixed) {
      lambda <- p + c_i * (lambda_target - p)
    } else {
      lambda <- lambda_target
    }
  }

  lambda <- max(0.001, min(0.999, lambda))

  if (treatment[i] == 1) {
    multiplier <- lambda / p
  } else {
    multiplier <- (1 - lambda) / (1 - p)
  }

  wealth[i] <- wealth[i-1] * multiplier

  if (treatment[i] == 1) {
    n_trt <- n_trt + 1
    events_trt <- events_trt + outcome[i]
  } else {
    n_ctrl <- n_ctrl + 1
    events_ctrl <- events_ctrl + outcome[i]
  }
}

```

```

    return(wealth)
}

# -----
# Simulate a single trial
# -----

simulate_trial <- function(n, p_trt = 0.5, rate_trt, rate_ctrl) {
  treatment <- rbinom(n, 1, p_trt)
  outcome <- numeric(n)
  outcome[treatment == 1] <- rbinom(sum(treatment == 1), 1, rate_trt)
  outcome[treatment == 0] <- rbinom(sum(treatment == 0), 1, rate_ctrl)
  return(data.frame(treatment = treatment, outcome = outcome))
}

# -----
# Unified simulation function
# -----

simulate_eRT <- function(n_sims = 5000,
                        p_ctrl,
                        p_trt = NULL,
                        hypothesized_ARR = NULL,
                        target_power = 0.80,
                        alpha = 0.05,
                        burn_in = 50,
                        ramp = 100,
                        wager = c("adaptive", "design", "fixed", "oracle"),
                        p_trt_design = NULL,
                        p_ctrl_design = NULL,
                        fixed_event_lambda = NULL,
                        fixed_nonevent_lambda = NULL,
                        ramp_fixed = TRUE,
                        kelly_fraction = 0.5) {
  wager <- match.arg(wager)

  # Determine hypothesized ARR for sample size calculation
  if (is.null(hypothesized_ARR) && is.null(p_trt)) {
    stop("Must specify either p_trt or hypothesized_ARR")
  }

  if (is.null(hypothesized_ARR)) {
    hypothesized_ARR <- p_ctrl - p_trt
  }

  # Calculate sample size based on hypothesized effect
  ss <- power.prop.test(p1 = p_ctrl, p2 = p_ctrl - hypothesized_ARR,

```

```

                                power = target_power, sig.level = alpha)
n_per_arm <- ceiling(ss$n)
n_patients <- 2 * n_per_arm

# Determine true p_trt (null if not specified)
if (is.null(p_trt)) {
  p_trt <- p_ctrl # Null is true
  true_ARR <- 0
  sim_type <- "Type I Error"
} else {
  true_ARR <- p_ctrl - p_trt
  sim_type <- "Power"
}

cat(sprintf("%s Simulation\n", sim_type))
cat(sprintf("  Design: p_ctrl = %.2f, hypothesized ARR = %.2f, target power = %.0f%%\n",
            p_ctrl, hypothesized_ARR, target_power * 100))
cat(sprintf("  Sample size: n = %d per arm = %d total\n", n_per_arm, n_patients))
cat(sprintf("  Truth: p_ctrl = %.2f, p_trt = %.2f, true ARR = %.2f\n",
            p_ctrl, p_trt, true_ARR))
cat(sprintf("  Wager policy: %s\n", wager))
cat(sprintf("  n_sims = %d\n", n_sims))

rejections <- 0
first_crossing <- numeric(n_sims)
final_evals <- numeric(n_sims)

pb <- txtProgressBar(min = 0, max = n_sims, style = 3)

for (sim in 1:n_sims) {
  trial <- simulate_trial(n_patients, rate_trt = p_trt, rate_ctrl = p_ctrl)
  wealth <- compute_eRT(
    trial$treatment,
    trial$outcome,
    burn_in = burn_in,
    ramp = ramp,
    wager = wager,
    p_trt_design = p_trt_design,
    p_ctrl_design = p_ctrl_design,
    p_trt_oracle = p_trt,
    p_ctrl_oracle = p_ctrl,
    fixed_event_lambda = fixed_event_lambda,
    fixed_nonevent_lambda = fixed_nonevent_lambda,
    ramp_fixed = ramp_fixed,
    kelly_fraction = kelly_fraction
  )

  final_evals[sim] <- wealth[n_patients]

```

```

    crossing <- which(wealth >= 1/alpha)
    if (length(crossing) > 0) {
      rejections <- rejections + 1
      first_crossing[sim] <- crossing[1]
    } else {
      first_crossing[sim] <- NA
    }

    setTxtProgressBar(pb, sim)
  }
close(pb)

rejection_rate <- rejections / n_sims
se <- sqrt(rejection_rate * (1 - rejection_rate) / n_sims)
median_stop <- median(first_crossing, na.rm = TRUE)

cat(sprintf("\nResults:\n"))
cat(sprintf("  Rejection rate: %.3f (SE: %.3f)\n", rejection_rate, se))
cat(sprintf("  95% CI: [%.3f, %.3f]\n", rejection_rate - 1.96*se, rejection_rate + 1.96*se))
if (sim_type == "Power") {
  cat(sprintf("  Target power: %.0f%%\n", target_power * 100))
  cat(sprintf("  Median crossing: %.0f patients\n", median_stop))
} else {
  cat(sprintf("  Nominal alpha: %.3f\n", alpha))
}

return(list(
  sim_type = sim_type,
  rejection_rate = rejection_rate,
  se = se,
  n_per_arm = n_per_arm,
  n_patients = n_patients,
  p_ctrl = p_ctrl,
  p_trt = p_trt,
  hypothesized_ARR = hypothesized_ARR,
  true_ARR = true_ARR,
  wager = wager,
  target_power = target_power,
  alpha = alpha,
  median_crossing = median_stop,
  first_crossing = first_crossing,
  final_values = final_values
))
}

# -----
# Plot trajectories

```

```

# -----

plot_trajectories <- function(n_trials = 30,
                              p_ctrl,
                              p_trt = NULL,
                              hypothesized_ARR = NULL,
                              target_power = 0.80,
                              alpha = 0.05,
                              burn_in = 50,
                              ramp = 100,
                              wager = c("adaptive", "design", "fixed", "oracle"),
                              p_trt_design = NULL,
                              p_ctrl_design = NULL,
                              fixed_event_lambda = NULL,
                              fixed_nonevent_lambda = NULL,
                              ramp_fixed = TRUE,
                              title = NULL) {
  wager <- match.arg(wager)

  if (is.null(hypothesized_ARR) && is.null(p_trt)) {
    stop("Must specify either p_trt or hypothesized_ARR")
  }

  if (is.null(hypothesized_ARR)) {
    hypothesized_ARR <- p_ctrl - p_trt
  }

  if (is.null(p_trt)) {
    p_trt <- p_ctrl # Null
  }

  # Calculate sample size
  ss <- power.prop.test(p1 = p_ctrl, p2 = p_ctrl - hypothesized_ARR,
                       power = target_power, sig.level = alpha)
  n_patients <- 2 * ceiling(ss$n)

  true_ARR <- p_ctrl - p_trt

  if (is.null(title)) {
    if (true_ARR == 0) {
      title <- sprintf("e-RTb Under Null (n = %d, designed for %.0f%% ARR)",
                       n_patients, hypothesized_ARR * 100)
    } else {
      title <- sprintf("e-RTb Under Alternative (n = %d, %.0f%% power)",
                       n_patients, target_power * 100)
    }
  }
}

```

```

trajectories <- list()

for (i in 1:n_trials) {
  trial <- simulate_trial(n_patients, rate_trt = p_trt, rate_ctrl = p_ctrl)
  wealth <- compute_eRT(
    trial$treatment,
    trial$outcome,
    burn_in = burn_in,
    ramp = ramp,
    wager = wager,
    p_trt_design = p_trt_design,
    p_ctrl_design = p_ctrl_design,
    p_trt_oracle = p_trt,
    p_ctrl_oracle = p_ctrl,
    fixed_event_lambda = fixed_event_lambda,
    fixed_nonevent_lambda = fixed_nonevent_lambda,
    ramp_fixed = ramp_fixed
  )
  trajectories[[i]] <- data.frame(
    patient = 1:n_patients,
    wealth = wealth,
    trial = i
  )
}

df <- bind_rows(trajectories)

p <- ggplot(df, aes(x = patient, y = wealth, group = trial)) +
  geom_line(alpha = 0.4, color = "steelblue") +
  geom_hline(yintercept = 1/alpha, linetype = "dashed", color = "red") +
  geom_hline(yintercept = 1, linetype = "dotted", color = "gray50") +
  scale_y_log10() +
  labs(
    title = title,
    subtitle = sprintf("Control = %.0f%%, Treatment = %.0f%%, True ARR = %.0f%%",
                       p_ctrl * 100, p_trt * 100, true_ARR * 100),
    x = "Patient",
    y = "e-value (log scale)"
  ) +
  annotate("text", x = n_patients * 0.95, y = 1/alpha * 1.5,
           label = sprintf("1/alpha = %.0f", 1/alpha), color = "red", hjust = 1) +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold"),
    panel.grid.minor = element_blank()
  )

return(p)

```

```

}

# -----
# Run all simulations
# -----

run_all <- function(n_sims = 5000,
                    p_ctrl = 0.40,
                    ARRs = c(0.05, 0.10),
                    target_powers = c(0.80, 0.90),
                    alpha = 0.05,
                    seed = 42) {

  set.seed(seed)
  results <- list()

  # Create grid of scenarios
  scenarios <- expand.grid(
    hypothesized_ARR = ARRs,
    target_power = target_powers,
    stringsAsFactors = FALSE
  )

  # For each scenario, run Type I error and Power
  all_results <- data.frame()

  for (i in 1:nrow(scenarios)) {
    hyp_ARR <- scenarios$hypothesized_ARR[i]
    tgt_pow <- scenarios$target_power[i]

    cat(sprintf("\n=====\n"))
    cat(sprintf("Scenario: ARR = %.0f%%, Target Power = %.0f%%\n",
                hyp_ARR * 100, tgt_pow * 100))
    cat(sprintf("=====\n\n"))

    # Type I error (null true, same sample size)
    cat("--- Type I Error ---\n")
    t1 <- simulate_eRT(
      n_sims = n_sims,
      p_ctrl = p_ctrl,
      p_trt = NULL, # Null is true
      hypothesized_ARR = hyp_ARR,
      target_power = tgt_pow,
      alpha = alpha
    )

    # Power (alternative true)
    cat("\n--- Power ---\n")

```

```

pow <- simulate_eRT(
  n_sims = n_sims,
  p_ctrl = p_ctrl,
  p_trt = p_ctrl - hyp_ARR, # True effect = hypothesized
  target_power = tgt_pow,
  alpha = alpha
)

all_results <- rbind(all_results, data.frame(
  hypothesized_ARR = hyp_ARR,
  target_power = tgt_pow,
  n_patients = t1$n_patients,
  type1_error = t1$rejection_rate,
  type1_se = t1$se,
  eRT_power = pow$rejection_rate,
  power_se = pow$se,
  median_crossing = pow$median_crossing
))
}

results$summary <- all_results

# Print summary
cat("\n=====\n")
cat("SUMMARY\n")
cat("=====\n\n")

print(all_results %>%
  mutate(
    hypothesized_ARR = sprintf("%.0f%%", hypothesized_ARR * 100),
    target_power = sprintf("%.0f%%", target_power * 100),
    type1_error = sprintf("%.3f", type1_error),
    eRT_power = sprintf("%.1f%%", eRT_power * 100)
  ) %>%
  select(hypothesized_ARR, target_power, n_patients,
    type1_error, eRT_power, median_crossing))

return(results)
}

# -----
# Run
# -----

if (interactive()) {

  results <- run_all(
    n_sims = 5000,

```

```

    p_ctrl = 0.40,
    ARRs = c(0.05, 0.10),
    target_powers = c(0.80, 0.90),
    alpha = 0.05
)

# Trajectory plots - Alternative
p1 <- plot_trajectories(p_ctrl = 0.40, p_trt = 0.30, target_power = 0.80)
ggsave("traj_alt_10pct_80pow.pdf", p1, width = 8, height = 5)

p2 <- plot_trajectories(p_ctrl = 0.40, p_trt = 0.30, target_power = 0.90)
ggsave("traj_alt_10pct_90pow.pdf", p2, width = 8, height = 5)

# Trajectory plots - Null
p3 <- plot_trajectories(p_ctrl = 0.40, p_trt = NULL, hypothesized_ARR = 0.10, target_power = 0.80)
ggsave("traj_null_10pct_80pow.pdf", p3, width = 8, height = 5)

p4 <- plot_trajectories(p_ctrl = 0.40, p_trt = NULL, hypothesized_ARR = 0.10, target_power = 0.90)
ggsave("traj_null_10pct_90pow.pdf", p4, width = 8, height = 5)
}

```

## ESM Section 2. Additional details on e-RTe

This section states and proves the validity lemma for event-only betting, gives the ordering corollary and a worked example, and measures what happens when the observation-process condition fails.

### 2.1 The event-coin lemma

e-RTe bets on the arm of each confirmed event. Its validity needs the event-arm sequence to be a fair coin at the design probability. The lemma states when that holds.

**Lemma.** Let assignments  $T_1, \dots, T_N$  be drawn independently with constant probability  $p$  (simple randomization), independently of all outcome and observation data. Assume: (a) this randomization design; (b) the null extends to the observation process — for every patient, the entire observed record (whether an event occurs, when it is confirmed, and all other outcome and observation data) is unaffected by assignment; and (c) ties between simultaneous confirmations are broken without reference to assignments. Index confirmed events  $j = 1, 2, \dots$  in confirmation order, write  $\pi(j)$  for the patient who provides event  $j$ , and set  $X_j = T_{\pi(j)}$ . Let  $\mathcal{G}_{j-1}$  contain the identities, times, and arms of the first  $j - 1$  confirmed events, the identity and time of the  $j$ -th confirmation, all outcome and observation data available at that moment, and the design. Then  $\Pr(X_j = 1 \mid \mathcal{G}_{j-1}) = p$  for every  $j$ .

**Proof.** Under (b) and (c), which patients have events, and the order and times of their confirmations, are functions of outcome and observation data alone; the assignment vector plays no part in them. The selection-and-ordering map  $\pi$  is therefore independent of  $(T_1, \dots, T_N)$ . Condition on the full assignment-free data record, which determines  $\pi$ : the relabeled sequence  $T_{\pi(1)}, T_{\pi(2)}, \dots$  picks distinct patients, so it is a sequence of independent Bernoulli( $p$ ) draws.  $\mathcal{G}_{j-1}$  is generated by a subset of that record together with the first  $j - 1$  relabeled assignments, so the tower property gives that, given  $\mathcal{G}_{j-1}$ , the next relabeled assignment  $T_{\pi(j)}$  is still Bernoulli( $p$ ).  $\square$

**Corollary (choice of ordering).** The proof uses only one property of the ordering: it is determined without reference to assignments. Any such ordering (confirmation order, calendar order of the underlying events, or any prespecified assignment-free rule) gives the same conclusion.

**Worked example (90-day mortality).** Patients enroll over calendar time. Deaths within 90 days are reported with a reporting lag; survivors are confirmed only when their 90-day window closes. Deaths are therefore confirmed early and roughly in death-date order, long before most survivor confirmations. Under the null, and with reporting lags unaffected by assignment (condition (b)), the sequence of death-arm labels is an independent Bernoulli( $p$ ) coin — whatever the enrollment pattern, lag distribution, or event rate. e-RTe bets on that coin without waiting for any survivor. If treatment changed the reporting of deaths (treated deaths confirmed later, say), condition (b) fails; subsection 2.2 measures what happens then.

## 2.2 Treatment-dependent censoring of events

Both arms share identical true event rates: the outcome null holds throughout. A fraction  $c$  of treatment-arm events is never ascertained, so the observed event coin has probability  $q = (1 - c)/(2 - c)$  instead of 0.5. Condition (b) fails by construction. Each stream contains 430 observed events; 5,000 streams per cell; assignments by 1:1 simple randomization. Two bettors run: the adaptive event wager and a design wager aimed at a 35% versus 30% benefit. Crossings at  $c > 0$  detect the observation-process difference; there is no treatment effect to detect.

**Table S6.** e-RTe null crossing rates under treatment-dependent censoring of events.

| Treatment-event<br>censoring $c$ | Observed event coin $q$ | Bettor                         | Crossing rate ( $W \geq 20$ ) |
|----------------------------------|-------------------------|--------------------------------|-------------------------------|
| 0.0%                             | 0.500                   | adaptive                       | 2.4% (0.22pp)                 |
| 0.0%                             | 0.500                   | design (35% vs 30%<br>benefit) | 0.9% (0.13pp)                 |
| 10.0%                            | 0.474                   | adaptive                       | 7.8% (0.38pp)                 |
| 10.0%                            | 0.474                   | design (35% vs 30%<br>benefit) | 6.5% (0.35pp)                 |
| 25.0%                            | 0.429                   | adaptive                       | 58.7% (0.70pp)                |
| 25.0%                            | 0.429                   | design (35% vs 30%<br>benefit) | 59.2% (0.70pp)                |
| 50.0%                            | 0.333                   | adaptive                       | 100.0% (0.03pp)               |
| 50.0%                            | 0.333                   | design (35% vs 30%<br>benefit) | 100.0% (0.03pp)               |

At  $c = 0$  the coin is fair and crossing rates sit at or below nominal. As  $c$  grows, the observed coin tilts and the crossing rate rises smoothly to certainty. Two readings follow. First, any analysis of the recorded events (log-rank, Kaplan-Meier, or a group-sequential boundary) inherits the same distortion; the condition concerns the data stream, not this method. Second, a crossing under selective reporting is itself a finding: one arm's events going unrecorded is a trial-conduct problem, and e-RTe surfaces it early. A crossing is evidence that the arms differ in the recorded stream; whether the difference lies in the outcomes or in their reporting is what the post-crossing review determines.

Code: e-RTe kernel (SuppCode\_eRTe.R, Section 2.4); seed 20260548.

## 2.3 Power under design-anticipated benefits

Detection power is measured on the same trials as the e-RTb tables (Tables S1-S2): control event rate 40%, design ARR of 5 and 10 percentage points, and the two-proportion (chi-square) sample sizes for 80% and 90% target power. e-RTe sees only the events of those trials: the number of events per stream is Binomial(N, pooled rate), and the event-arm coin is 0.467 (5 pp) or 0.429 (10 pp), priced at the null coin 0.5. No censoring: lemma condition (b) holds.

**Table S7.** e-RTe detection power at traditional sample sizes.

| Design<br>ARR | Target<br>power | N (chi-<br>square) | Median<br>events | Bettor            | Power<br>(W $\geq$ 20) | Median<br>crossing<br>(event) | % events |
|---------------|-----------------|--------------------|------------------|-------------------|------------------------|-------------------------------|----------|
| 5.0 pp        | 80%             | 2,942              | 1103             | Adaptive          | 31.7%<br>(0.66pp)      | 550                           | 49.9%    |
| 5.0 pp        | 80%             | 2,942              | 1103             | Design<br>matched | 51.0%<br>(0.71pp)      | 694                           | 63.0%    |
| 5.0 pp        | 90%             | 3,938              | 1476             | Adaptive          | 42.3%<br>(0.70pp)      | 711                           | 48.2%    |
| 5.0 pp        | 90%             | 3,938              | 1476             | Design<br>matched | 67.0%<br>(0.66pp)      | 792                           | 53.7%    |
| 10.0 pp       | 80%             | 712                | 249              | Adaptive          | 33.0%<br>(0.66pp)      | 158                           | 63.5%    |
| 10.0 pp       | 80%             | 712                | 249              | Design<br>matched | 43.4%<br>(0.70pp)      | 182                           | 73.1%    |
| 10.0 pp       | 90%             | 954                | 334              | Adaptive          | 46.7%<br>(0.71pp)      | 191                           | 57.2%    |
| 10.0 pp       | 90%             | 954                | 334              | Design<br>matched | 62.9%<br>(0.68pp)      | 211                           | 63.2%    |

The matched design wager crosses in 43–67% of streams and the adaptive wager in 32–47%. e-RTb with access to every outcome reaches 75–86% on the same trials (Table S2): the gap is the information discarded with the non-event stream.

Code: e-RTe kernel (SuppCode\_eRTe.R, Section 2.4); seed 20260550.

## 2.4 Code for e-RTe

The listing below is the complete e-RTe kernel (SuppCode\_eRTe.R), the implementation behind every simulation in this section. Each subsection above states the seed that drove it; the three kernel files, with every simulation script and frozen output, are at <https://github.com/fzampier/ert-methods-paper> (archived at <https://doi.org/10.5281/zenodo.22906388>).

```
# e-RTe: Event-Only Sequential Monitoring
# Supplementary R Code for e-RT V7
# Date: 2026-03-08
#
```

```

# A simplified e-process that monitors only events (e.g., deaths, disease
# progression, myocardial infarction), without requiring non-event data or
# denominator tracking. Based on the insight that under 1:1 randomization,
#  $P(\text{event from treatment} \mid \text{event occurred}) = 0.5$  under null.
#
# Key advantage: No non-event ascertainment needed; just log events.
# Active manuscript comparisons use explicit sample sizes, including same-N
# comparisons against e-RTb.

library(tidyverse)

# === CORE E-PROCESS ===

#' Compute the e-RTe wealth process from a stream of events.
#'
#' @param arms Character or numeric vector: 1 = treatment, 0 = control.
#'   Each element represents one event, in the order events occurred.
#' @param burn_in Number of events before betting begins (default 30).
#' @param ramp Number of events over which betting ramps from 0 to full (default 50).
#' @param threshold Rejection threshold, typically  $1/\alpha$  (default 20).
#' @param kelly_fraction Fractional adaptive Kelly intensity; 1 preserves the
#'   default full-intensity event-only wager.
#' @return A list with:
#'   - wealth: numeric vector of wealth after each event
#'   - crossed: logical, did wealth ever reach threshold?
#'   - crossed_at: event number at first crossing (NA if never)
#'   - final_p: final  $P(\text{event from trt} \mid \text{event})$ 
#'   - final_rr: final event rate ratio  $p/(1-p)$ 
event_coin <- function(p_ctrl, p_trt, p_random = 0.5) {
  denom <- p_random * p_trt + (1 - p_random) * p_ctrl
  if (denom <= 0) stop("Event rates imply a degenerate event-coin denominator")
  p_random * p_trt / denom
}

compute_eRTe <- function(arms, burn_in = 30, ramp = 50, threshold = 20,
  p_random = 0.5,
  wager = c("adaptive", "design", "fixed", "oracle"),
  p_trt_design = NULL, p_ctrl_design = NULL,
  p_trt_oracle = NULL, p_ctrl_oracle = NULL,
  fixed_lambda = NULL,
  ramp_fixed = TRUE,
  kelly_fraction = 1) {
  n <- length(arms)
  wager <- match.arg(wager)

  if (is.factor(arms)) arms <- as.numeric(arms) - 1
  arms <- as.numeric(arms)

```

```

if (wager == "design" &&
    (is.null(p_trt_design) || is.null(p_ctrl_design))) {
  stop("Design wager requires p_trt_design and p_ctrl_design")
}

if (wager == "oracle" &&
    (is.null(p_trt_oracle) || is.null(p_ctrl_oracle))) {
  stop("Oracle wager requires p_trt_oracle and p_ctrl_oracle")
}

if (wager == "fixed" && is.null(fixed_lambda)) {
  stop("Fixed wager requires fixed_lambda on the event-coin scale")
}

fixed_target <- switch(
  wager,
  design = event_coin(p_ctrl_design, p_trt_design, p_random),
  oracle = event_coin(p_ctrl_oracle, p_trt_oracle, p_random),
  fixed = fixed_lambda,
  adaptive = NA_real_
)

wealth <- numeric(n)
wealth[1] <- 1

d_trt <- 0
d_ctrl <- 0
crossed <- FALSE
crossed_at <- NA

for (i in 1:n) {
  # Step 1: Compute p_obs from PREVIOUS events (before updating counts)
  total <- d_trt + d_ctrl
  p_obs <- if (total > 0) d_trt / total else 0.5

  # Step 2: Compute lambda (wager)
  c_i <- min(1, max(0, (i - burn_in) / ramp))
  if (wager == "adaptive") {
    if (i > burn_in && total > 0) {
      lambda <- p_random + kelly_fraction * c_i * (p_obs - p_random)
      lambda <- max(0.001, min(0.999, lambda))
    } else {
      lambda <- p_random # Neutral bet during burn-in
    }
  } else {
    if (ramp_fixed) {
      lambda <- p_random + c_i * (fixed_target - p_random)
    } else {

```

```

    lambda <- fixed_target
  }
  lambda <- max(0.001, min(0.999, lambda))
}

# Step 3: Update wealth
if (arms[i] == 1) {
  # Treatment event
  mult <- lambda / p_random
} else {
  # Control event
  mult <- (1 - lambda) / (1 - p_random)
}

if (i == 1) {
  wealth[i] <- mult
} else {
  wealth[i] <- wealth[i - 1] * mult
}

# Step 4: Update counts AFTER betting
if (arms[i] == 1) {
  d_trt <- d_trt + 1
} else {
  d_ctrl <- d_ctrl + 1
}

# Check threshold (record first crossing only)
if (!crossed && wealth[i] >= threshold) {
  crossed <- TRUE
  crossed_at <- i
}
}

# Final estimates
total <- d_trt + d_ctrl
final_p <- if (total > 0) d_trt / total else 0.5
final_rr <- if (final_p > 0.001 && final_p < 0.999) {
  final_p / (1 - final_p)
} else if (final_p <= 0.001) {
  0
} else {
  Inf
}

list(
  wealth = wealth,
  crossed = crossed,

```

```

    crossed_at = crossed_at,
    final_p = final_p,
    final_rr = final_rr,
    d_trt = d_trt,
    d_ctrl = d_ctrl
  )
}

# === SIMULATION HELPERS ===

#' Generate a stream of events for simulation.
#'
#' @param n_events Number of events to simulate.
#' @param p_trt_given_event P(event from treatment | event occurred).
#' Under null: 0.5. Under alternative with treatment benefit: < 0.5.
#' @return Numeric vector of arm labels (1 = treatment, 0 = control).
simulate_events <- function(n_events, p_trt_given_event) {
  rbinom(n_events, 1, p_trt_given_event)
}

#' Calculate expected number of events from event rates and sample size.
#'
#' @param n_patients Total number of patients (both arms).
#' @param p_ctrl Control arm event rate.
#' @param p_trt Treatment arm event rate.
#' @return Expected number of events.
expected_events <- function(n_patients, p_ctrl, p_trt) {
  ceiling(n_patients / 2 * (p_ctrl + p_trt))
}

# === FULL SIMULATION ===

#' Run e-RTe simulations under null and alternative hypotheses.
#'
#' @param n_sims Number of simulations (default 1000).
#' @param p_ctrl Control arm event rate.
#' @param p_trt Treatment arm event rate (under alternative).
#' For null simulation, both arms use p_ctrl.
#' @param n_patients Total number of patients. If NULL, uses a legacy
#' convenience default equal to the frequentist sample size with 2.5x
#' inflation. Active manuscript scripts pass explicit sample sizes.
#' @param target_power Target power for sample size calculation (default 0.80).
#' @param alpha Significance level (default 0.05).
#' @param burn_in Burn-in period in events (default 30).
#' @param ramp Ramp period in events (default 50).
#' @return A list with null and alternative results.
simulate_eRTe <- function(n_sims = 1000, p_ctrl, p_trt,
                          n_patients = NULL, target_power = 0.80,

```

```

        alpha = 0.05, burn_in = 30, ramp = 50,
        wager = c("adaptive", "design", "fixed", "oracle"),
        p_trt_design = NULL, p_ctrl_design = NULL,
        fixed_lambda = NULL,
        ramp_fixed = TRUE,
        kelly_fraction = 1) {
wager <- match.arg(wager)

threshold <- 1 / alpha

# Sample size calculation
if (is.null(n_patients)) {
  ss <- power.prop.test(p1 = p_ctrl, p2 = p_trt,
                        power = target_power, sig.level = alpha)
  n_freq <- 2 * ceiling(ss$n)
  n_patients <- ceiling(n_freq * 2.5) # 2.5x inflation for e-RTe
} else {
  ss <- power.prop.test(p1 = p_ctrl, p2 = p_trt,
                        power = target_power, sig.level = alpha)
  n_freq <- 2 * ceiling(ss$n)
}

# Expected events under null and alternative
n_events_null <- expected_events(n_patients, p_ctrl, p_ctrl)
n_events_alt <- expected_events(n_patients, p_ctrl, p_trt)

# Event coin probabilities
p_coin_null <- 0.5
p_coin_alt <- event_coin(p_ctrl, p_trt)

cat(sprintf("e-RTe Simulation\n"))
cat(sprintf("  Control event rate: %.1f%%\n", p_ctrl * 100))
cat(sprintf("  Treatment event rate: %.1f%%\n", p_trt * 100))
cat(sprintf("  ARR: %.1f%%\n", (p_ctrl - p_trt) * 100))
cat(sprintf("  Event coin (null): %.3f\n", p_coin_null))
cat(sprintf("  Event coin (alt): %.3f (tilt = %.1f points)\n",
            p_coin_alt, abs(p_coin_alt - 0.5) * 100))
cat(sprintf("  Wager policy: %s\n", wager))
cat(sprintf("  Frequentist N: %d\n", n_freq))
cat(sprintf("  e-RTe N: %d (%.1fx inflation)\n", n_patients, n_patients / n_freq))
cat(sprintf("  Expected events (null): %d\n", n_events_null))
cat(sprintf("  Expected events (alt): %d\n", n_events_alt))
cat(sprintf("  Threshold: %.0f\n\n", threshold))

# --- Null simulations ---
cat("Running null simulations...\n")
null_rejections <- 0
null_first_crossing <- numeric(n_sims)

```

```

null_final_evalues <- numeric(n_sims)

pb <- txtProgressBar(min = 0, max = n_sims, style = 3)
for (sim in 1:n_sims) {
  events <- simulate_events(n_events_null, p_coin_null)
  res <- compute_eRTe(
    events,
    burn_in = burn_in,
    ramp = ramp,
    threshold = threshold,
    wager = wager,
    p_trt_design = p_trt_design,
    p_ctrl_design = p_ctrl_design,
    p_trt_oracle = p_ctrl,
    p_ctrl_oracle = p_ctrl,
    fixed_lambda = fixed_lambda,
    ramp_fixed = ramp_fixed,
    kelly_fraction = kelly_fraction
  )
  null_final_evalues[sim] <- tail(res$wealth, 1)

  if (res$crossed) {
    null_rejections <- null_rejections + 1
    null_first_crossing[sim] <- res$crossed_at
  } else {
    null_first_crossing[sim] <- NA
  }
  setTxtProgressBar(pb, sim)
}
close(pb)

type1 <- null_rejections / n_sims

# --- Alternative simulations ---
cat("Running alternative simulations...\n")
alt_rejections <- 0
alt_first_crossing <- numeric(n_sims)
alt_final_evalues <- numeric(n_sims)

pb <- txtProgressBar(min = 0, max = n_sims, style = 3)
for (sim in 1:n_sims) {
  events <- simulate_events(n_events_alt, p_coin_alt)
  res <- compute_eRTe(
    events,
    burn_in = burn_in,
    ramp = ramp,
    threshold = threshold,
    wager = wager,

```

```

    p_trt_design = p_trt_design,
    p_ctrl_design = p_ctrl_design,
    p_trt_oracle = p_trt,
    p_ctrl_oracle = p_ctrl,
    fixed_lambda = fixed_lambda,
    ramp_fixed = ramp_fixed,
    kelly_fraction = kelly_fraction
  )
  alt_final_values[sim] <- tail(res$wealth, 1)

  if (res$crossed) {
    alt_rejections <- alt_rejections + 1
    alt_first_crossing[sim] <- res$crossed_at
  } else {
    alt_first_crossing[sim] <- NA
  }
  setTxtProgressBar(pb, sim)
}
close(pb)

power <- alt_rejections / n_sims

# --- Summary ---
cat(sprintf("\n=====\n"))
cat(sprintf("  RESULTS\n"))
cat(sprintf("=====\n"))
cat(sprintf("Type I Error: %.2f%% (SE: %.2f%%)\n",
            type1 * 100,
            sqrt(type1 * (1 - type1) / n_sims) * 100))
cat(sprintf("Power:          %.1f%% (SE: %.1f%%)\n",
            power * 100,
            sqrt(power * (1 - power) / n_sims) * 100))
if (sum(!is.na(alt_first_crossing)) > 0) {
  med_cross <- median(alt_first_crossing, na.rm = TRUE)
  cat(sprintf("Median crossing: %.0f events (%.0f%% of total)\n",
              med_cross, med_cross / n_events_alt * 100))
}

list(
  # Parameters
  p_ctrl = p_ctrl,
  p_trt = p_trt,
  n_patients = n_patients,
  n_freq = n_freq,
  n_events_null = n_events_null,
  n_events_alt = n_events_alt,
  p_coin_null = p_coin_null,
  p_coin_alt = p_coin_alt,

```

```

    wager = wager,
    threshold = threshold,
    # Null results
    type1 = type1,
    type1_se = sqrt(type1 * (1 - type1) / n_sims),
    null_final_evaluations = null_final_evaluations,
    null_first_crossing = null_first_crossing,
    # Alternative results
    power = power,
    power_se = sqrt(power * (1 - power) / n_sims),
    alt_final_evaluations = alt_final_evaluations,
    alt_first_crossing = alt_first_crossing,
    median_crossing = median(alt_first_crossing, na.rm = TRUE)
  )
}

# === TRAJECTORY VISUALIZATION ===

#' Plot e-RTe wealth trajectories under null and/or alternative.
#'
#' @param n_trials Number of trajectories to plot (default 30).
#' @param n_events Number of events per trajectory.
#' @param p_trt_given_event P(event from treatment | event occurred).
#' @param burn_in Burn-in period (default 30).
#' @param ramp Ramp period (default 50).
#' @param alpha Significance level (default 0.05).
#' @param title Plot title.
#' @return A ggplot2 object.
plot_trajectories_eRTe <- function(n_trials = 30, n_events,
                                   p_trt_given_event,
                                   burn_in = 30, ramp = 50,
                                   alpha = 0.05,
                                   wager = c("adaptive", "fixed"),
                                   fixed_lambda = NULL,
                                   ramp_fixed = TRUE,
                                   title = NULL) {

  wager <- match.arg(wager)

  threshold <- 1 / alpha

  if (is.null(title)) {
    title <- if (p_trt_given_event == 0.5) {
      sprintf("e-RTe Under Null (p = 0.50, %d events)", n_events)
    } else {
      sprintf("e-RTe Under Alternative (p = %.3f, %d events)", p_trt_given_event, n_events)
    }
  }
}

```

```

trajectories <- lapply(1:n_trials, function(i) {
  events <- simulate_events(n_events, p_trt_given_event)
  res <- compute_eRTe(
    events,
    burn_in = burn_in,
    ramp = ramp,
    threshold = threshold,
    wager = wager,
    fixed_lambda = fixed_lambda,
    ramp_fixed = ramp_fixed
  )
  data.frame(event = 1:n_events, wealth = res$wealth, trial = i)
})

df <- bind_rows(trajectories)

ggplot(df, aes(x = event, y = wealth, group = trial)) +
  geom_line(alpha = 0.4, color = "steelblue") +
  geom_hline(yintercept = threshold, linetype = "dashed", color = "red") +
  geom_hline(yintercept = 1, linetype = "dotted", color = "gray50") +
  scale_y_log10() +
  labs(title = title,
       subtitle = sprintf("P(trt event | event) = %.3f, threshold = %.0f",
                          p_trt_given_event, threshold),
       x = "Event Number", y = "e-value (log scale)") +
  annotate("text", x = n_events * 0.95, y = threshold * 1.5,
          label = sprintf("1/alpha = %.0f", threshold), color = "red", hjust = 1) +
  theme_minimal() +
  theme(plot.title = element_text(face = "bold"), panel.grid.minor = element_blank())
}

# === SIGNAL CONCENTRATION ANALYSIS ===

#' Compute the event-coin tilt for different baseline event rates.
#'
#' Demonstrates the signal concentration property: when baseline event rate
#' is low, the event coin tilt (deviation from 0.5) exceeds the ARR,
#' concentrating the signal in the event stream.
#'
#' @param baseline_rates Vector of baseline (control) event rates.
#' @param arr Absolute risk reduction (default 0.05).
#' @return Data frame with baseline rate, treatment rate, event coin, and tilt.
signal_concentration_table <- function(baseline_rates = c(0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40),
                                       arr = 0.05) {
  data.frame(
    baseline = baseline_rates,
    trt_rate = baseline_rates - arr,
    event_coin = sapply(baseline_rates, function(p) event_coin(p, p - arr)),

```

```

    tilt_from_half = sapply(baseline_rates, function(p) abs(event_coin(p, p - arr) - 0.5)),
    arr = arr
  )
}

# === EXAMPLE USAGE ===
# Uncomment to run:

# # Signal concentration demonstration
# cat("\n=== Signal Concentration ===\n")
# sc <- signal_concentration_table(arr = 0.05)
# print(sc, digits = 3)
#
# # Quick simulation: 15% baseline event rate, 5pp ARR
# results <- simulate_eRTE(n_sims = 1000, p_ctrl = 0.15, p_trt = 0.10)
#
# # Plot trajectories
# p_alt <- event_coin(0.15, 0.10)
# n_events <- expected_events(results$n_patients, 0.15, 0.10)
#
# p_null <- plot_trajectories_eRTE(n_events = n_events, p_trt_given_event = 0.5)
# p_alt_plot <- plot_trajectories_eRTE(n_events = n_events, p_trt_given_event = p_alt)
#
# print(p_null)
# print(p_alt_plot)

```

## ESM Section 3. Additional details on e-RTc

This section gives the roles of the e-RTc tuning parameters, the null behavior of the construction across outcome distributions, and sensitivity to the wager cap.

### 3.1 Parameter roles

| Parameter          | e-RTc default             | Role                                                | Effect on validity | Effect on efficiency                                           |
|--------------------|---------------------------|-----------------------------------------------------|--------------------|----------------------------------------------------------------|
| $\epsilon$ (clamp) | 0.001 (all e-RT variants) | keeps $\lambda_i$ inside $(0, 1)$                   | none               | prevents degenerate wealth multipliers; rarely engaged         |
| $n_0$ (burn-in)    | 20 (conservative: 50)     | no bet for the first $n_0$ patients                 | none               | avoids betting on early noise; delays the first possible gains |
| $n_r$ (ramp)       | 50 (conservative: 100)    | bet grows linearly to full size over $n_r$ patients | none               | slower ramp bets less on immature estimates                    |

| Parameter        | e-RTc default                                         | Role                                   | Effect on validity | Effect on efficiency                                                          |
|------------------|-------------------------------------------------------|----------------------------------------|--------------------|-------------------------------------------------------------------------------|
| $c_{\max}$ (cap) | 0.6                                                   | bounds the maximum bet                 | none               | larger caps cross earlier when right and lose power from volatility (see 3.4) |
| direction form   | sign of mean difference (conservative: Cohen's- $d$ ) | equation (16) or (17) of the main text | none               | Cohen's- $d$ preserves magnitude; suits zero-inflated or discrete outcomes    |

Every entry in the validity column is “none”: Type I error control comes from the randomization denominator and the predictability of  $\lambda_i$ , not from tuning. Tuning trades speed against power: larger bets cross earlier when they are right and lose more when they are wrong.

Code: defaults set in the e-RTc kernel (SuppCode\_eRTc.R, Section 3.5); the `adaptive_direction` argument selects equation (16) or (17).

**Degenerate-case safeguards, as implemented.** If the running MAD  $s_{i-1}$  is zero or not finite (after a long stretch of identical outcomes early in a zero-inflated stream, say), the implementation replaces the scale by a fallback value of 1 and betting continues, with  $r_i$  computed against that unit scale. If instead the direction cannot be computed (an arm still empty under the sign form, or a zero pooled variance or an arm with fewer than two prior patients under the Cohen's- $d$  form), the direction is set to  $D_{i-1} = 0$ , so  $\lambda_i = p$ : the bet for that patient is neutral and the wealth multiplier equals one exactly. In every case the patient's outcome and assignment still enter the history used by later wagers; each  $\lambda_i$  remains a fixed, predictable function of past data, so the validity argument is unaffected. Separately, because  $c_{\max} = 0.6$  exceeds 0.5, the linear wager rule can land outside  $(0, 1)$  for very extreme standardized residuals; in those cases the global clamp sets  $\lambda_i$  to the nearest of  $\epsilon$  or  $1 - \epsilon$ . The clamp is rarely engaged in the simulations and trial illustrations.

### 3.2 Null behavior across data-generating processes

The null behavior of the e-RTc construction was checked by Monte Carlo across four continuous data-generating processes (DGPs): standard normal, Student's  $t$  with three degrees of freedom, lognormal, and a zero-inflated integer distribution (about 20% zeros, otherwise truncated Poisson with mean near 15). The four DGPs span light tails, heavy tails, skewed positive support, and a discrete distribution where the sign of the running median split carries little information; the zero-inflated integer case mirrors days-free-of-event count endpoints common in critical-care research. Each cell used  $n = 500$  patients per simulated trial and 5,000 simulated trials, with assignments by 1:1 simple randomization. Two parameter settings were evaluated: the defaults (burn-in 20, ramp 50, sign direction) and the conservative variant (burn-in 50, ramp 100, Cohen's- $d$  direction) suited to zero-inflated or discrete outcomes.

**Table S8.** Null behavior of e-RTc across four DGPs and two parameter settings. Each cell summarises 5,000 simulated null trials of  $n = 500$  patients (treatment assignment independent of outcome). Empirical Type I error is the fraction of trials ever crossing  $W \geq 20$  (nominal 5%), with Monte Carlo standard errors. Peak  $W$  is the maximum of the wealth trajectory within a trial,  $\max_{i \leq n} W_i$ ; the table reports its median and 95th percentile across trials.

| DGP                   | Setting                                        | N   | n_sims | Empirical Type I | Median peak W | Q95 peak W |
|-----------------------|------------------------------------------------|-----|--------|------------------|---------------|------------|
| Normal                | Default<br>(sign; burn 20, ramp 50)            | 500 | 5000   | 4.5% (SE 0.29%)  | 1.76          | 17.84      |
| Normal                | Conservative<br>(Cohen's d; burn 50, ramp 100) | 500 | 5000   | 2.4% (SE 0.22%)  | 1.58          | 10.73      |
| Student's t (df=3)    | Default<br>(sign; burn 20, ramp 50)            | 500 | 5000   | 4.1% (SE 0.28%)  | 1.75          | 16.56      |
| Student's t (df=3)    | Conservative<br>(Cohen's d; burn 50, ramp 100) | 500 | 5000   | 2.5% (SE 0.22%)  | 1.57          | 10.93      |
| Lognormal             | Default<br>(sign; burn 20, ramp 50)            | 500 | 5000   | 4.1% (SE 0.28%)  | 1.75          | 16.12      |
| Lognormal             | Conservative<br>(Cohen's d; burn 50, ramp 100) | 500 | 5000   | 2.4% (SE 0.22%)  | 1.60          | 10.17      |
| Zero-inflated integer | Default<br>(sign; burn 20, ramp 50)            | 500 | 5000   | 4.3% (SE 0.29%)  | 1.74          | 17.12      |
| Zero-inflated integer | Conservative<br>(Cohen's d; burn 50, ramp 100) | 500 | 5000   | 2.6% (SE 0.22%)  | 1.59          | 10.92      |

The empirical Type I error rate is 4.1–4.5% at the default settings and 2.4–2.6% at the conservative settings, at or below the nominal 5% in every cell. Because the validity argument does not involve the outcome distribution, agreement across DGPs checks the implementation against the theory; the differences between DGPs are of the same order as the Monte Carlo standard errors. Typical null paths peak below 2, and even the 95th percentile of peaks stays under 20 at both settings. The longer-burn-in Cohen's-*d* variant is the more conservative of the two; it is the setting used for the ventilator-free-days illustration in the main text.

Code: e-RTc kernel (SuppCode\_eRTc.R, Section 3.5); seed 20260518.

### 3.3 Power under normal-shift alternatives

Each trial uses the traditional two-sample t-test sample size for its shift and target power (two-sided alpha 0.05):  $N = 788$  and  $1,054$  for a 0.2 SD shift,  $N = 352$  and  $470$  for a 0.3 SD shift. Control outcomes are  $N(0,1)$  and treatment outcomes  $N(\delta,1)$ , with 1:1 simple randomization and 5,000 trials per cell.

**Table S9.** e-RTc detection power at traditional sample sizes.

| Shift (SD) | Target power | N (t-test) | Bettor                                                  | Power (W $\geq$ 20) | Median crossing (patient) | % N   |
|------------|--------------|------------|---------------------------------------------------------|---------------------|---------------------------|-------|
| 0.2        | 80%          | 788        | Adaptive default<br>(sign; burn 20, ramp 50)            | 9.7%<br>(0.42pp)    | 65                        | 8.2%  |
| 0.2        | 80%          | 788        | Adaptive conservative<br>(Cohen's d; burn 50, ramp 100) | 50.1%<br>(0.71pp)   | 400                       | 50.8% |
| 0.2        | 80%          | 788        | Design matched<br>(normal shift)                        | 73.8%<br>(0.62pp)   | 408                       | 51.8% |
| 0.2        | 90%          | 1,054      | Adaptive default<br>(sign; burn 20, ramp 50)            | 9.9%<br>(0.42pp)    | 65                        | 6.2%  |
| 0.2        | 90%          | 1,054      | Adaptive conservative<br>(Cohen's d; burn 50, ramp 100) | 62.9%<br>(0.68pp)   | 486                       | 46.2% |
| 0.2        | 90%          | 1,054      | Design matched<br>(normal shift)                        | 84.9%<br>(0.51pp)   | 437                       | 41.5% |
| 0.3        | 80%          | 352        | Adaptive default<br>(sign; burn 20, ramp 50)            | 18.6%<br>(0.55pp)   | 67                        | 19.0% |
| 0.3        | 80%          | 352        | Adaptive conservative<br>(Cohen's d; burn 50, ramp 100) | 48.7%<br>(0.71pp)   | 217                       | 61.6% |
| 0.3        | 80%          | 352        | Design matched<br>(normal shift)                        | 71.3%<br>(0.64pp)   | 202                       | 57.4% |
| 0.3        | 90%          | 470        | Adaptive default<br>(sign; burn 20, ramp 50)            | 17.9%<br>(0.54pp)   | 66                        | 14.0% |

| Shift (SD) | Target power | N (t-test) | Bettor                                               | Power (W $\geq$ 20) | Median crossing (patient) | % N   |
|------------|--------------|------------|------------------------------------------------------|---------------------|---------------------------|-------|
| 0.3        | 90%          | 470        | Adaptive conservative (Cohen's d; burn 50, ramp 100) | 63.4% (0.68pp)      | 251                       | 53.4% |
| 0.3        | 90%          | 470        | Design matched (normal shift)                        | 85.0% (0.50pp)      | 221                       | 47.0% |

The matched design wager reaches 71–85%, close to the fixed-sample targets. The conservative adaptive setting reaches 49–63%. The default sign setting crosses early or not at all (10–19%, median crossing near patient 65): its aggressive bets drain wealth after the early peaks, the behavior visible in the null trajectories. For continuous endpoints the conservative setting is the practical adaptive choice; it is the setting used for the ventilator-free-days illustration.

Code: e-RTc kernel (SuppCode\_eRTc.R, Section 3.5); seed 20260551.

### 3.4 Sensitivity to the cap $c_{\max}$

The cap  $c_{\max}$  (default 0.6) bounds the maximum bet. The rows below rerun the default policy at  $c_{\max} \in \{0.4, 0.8\}$  under a fresh random stream; the original grid of 3.2 reproduces exactly on its untouched stream.

**Table S10.** e-RTc null behavior at alternative values of the cap  $c_{\max}$ , default policy otherwise.

| DGP                | Setting                                                 | N   | n_sims | Empirical Type I | Median peak W | Q95 peak W |
|--------------------|---------------------------------------------------------|-----|--------|------------------|---------------|------------|
| Normal             | Default policy, $c_{\max}=0.4$ (sign; burn 20, ramp 50) | 500 | 5000   | 4.4% (SE 0.29%)  | 1.81          | 16.99      |
| Normal             | Default policy, $c_{\max}=0.8$ (sign; burn 20, ramp 50) | 500 | 5000   | 4.2% (SE 0.28%)  | 1.82          | 15.95      |
| Student's t (df=3) | Default policy, $c_{\max}=0.4$ (sign; burn 20, ramp 50) | 500 | 5000   | 4.8% (SE 0.30%)  | 1.76          | 18.94      |

| DGP                          | Setting                                                        | N   | n_sims | Empirical<br>Type I | Median peak<br>W | Q95 peak W |
|------------------------------|----------------------------------------------------------------|-----|--------|---------------------|------------------|------------|
| Student's t<br>(df=3)        | Default<br>policy,<br>c_max=0.8<br>(sign; burn<br>20, ramp 50) | 500 | 5000   | 3.9% (SE<br>0.28%)  | 1.73             | 15.75      |
| Lognormal                    | Default<br>policy,<br>c_max=0.4<br>(sign; burn<br>20, ramp 50) | 500 | 5000   | 4.5% (SE<br>0.29%)  | 1.80             | 17.94      |
| Lognormal                    | Default<br>policy,<br>c_max=0.8<br>(sign; burn<br>20, ramp 50) | 500 | 5000   | 3.9% (SE<br>0.27%)  | 1.72             | 15.10      |
| Zero-<br>inflated<br>integer | Default<br>policy,<br>c_max=0.4<br>(sign; burn<br>20, ramp 50) | 500 | 5000   | 4.2% (SE<br>0.28%)  | 1.79             | 16.02      |
| Zero-<br>inflated<br>integer | Default<br>policy,<br>c_max=0.8<br>(sign; burn<br>20, ramp 50) | 500 | 5000   | 3.7% (SE<br>0.27%)  | 1.71             | 15.04      |

Null crossing rates stay between 3.7% and 4.8% across both cap values and all four DGPs: the cap affects efficiency only; validity does not depend on it.

Code: e-RTc kernel (SuppCode\_eRTc.R, Section 3.5); c\_max rows appended under fresh seed 20260547 (original grid reproduces exactly).

### 3.5 Code for e-RTc

The listing below is the complete e-RTc kernel (SuppCode\_eRTc.R), the implementation behind every simulation in this section. Each subsection above states the seed that drove it; the three kernel files, with every simulation script and frozen output, are at <https://github.com/fzampier/ert-methods-paper> (archived at <https://doi.org/10.5281/zenodo.22906388>).

```
# e-RTc: Continuous-Outcome Sequential Monitoring
#
# Adaptive mode preserves the standalone V7 script's effect-size-agnostic
# sign-direction wager. Design/oracle mode uses a prespecified normal-shift
# working model to choose the wager, while validity still follows from
```

```

# randomized treatment assignment.

continuous_design_lambda <- function(y, mu_ctrl, mu_trt, sd,
                                     p_random = 0.5) {
  if (!is.finite(sd) || sd <= 0) stop("sd must be positive")
  if (p_random <= 0 || p_random >= 1) stop("p_random must be in (0, 1)")

  log_trt <- log(p_random) + dnorm(y, mean = mu_trt, sd = sd, log = TRUE)
  log_ctrl <- log1p(-p_random) + dnorm(y, mean = mu_ctrl, sd = sd, log = TRUE)

  1 / (1 + exp(log_ctrl - log_trt))
}

cohens_d_estimate <- function(treatment, outcome, upto = length(treatment)) {
  if (is.na(upto) || upto < 1) return(NA_real_)
  upto <- min(length(treatment), as.integer(upto))

  idx <- seq_len(upto)
  trt <- treatment[idx] == 1
  ctrl <- treatment[idx] == 0
  if (sum(trt) < 2 || sum(ctrl) < 2) return(NA_real_)

  y_trt <- outcome[idx][trt]
  y_ctrl <- outcome[idx][ctrl]
  sd_trt <- sd(y_trt)
  sd_ctrl <- sd(y_ctrl)
  if (!is.finite(sd_trt) || !is.finite(sd_ctrl)) return(NA_real_)

  s_pooled <- sqrt((sd_trt^2 + sd_ctrl^2) / 2)
  if (!is.finite(s_pooled) || s_pooled <= 0) return(NA_real_)

  (mean(y_trt) - mean(y_ctrl)) / s_pooled
}

simulate_trial_continuous <- function(n, p_trt = 0.5, mu_ctrl = 0,
                                     mu_trt = 0, sd = 1) {
  treatment <- rbinom(n, 1, p_trt)
  outcome <- numeric(n)
  outcome[treatment == 0] <- rnorm(sum(treatment == 0), mean = mu_ctrl, sd = sd)
  outcome[treatment == 1] <- rnorm(sum(treatment == 1), mean = mu_trt, sd = sd)
  data.frame(treatment = treatment, outcome = outcome)
}

design_n_continuous <- function(design_d, target_power = 0.80,
                               alpha = 0.05, sd = 1) {
  if (!is.finite(design_d) || design_d == 0) {
    stop("design_d must be nonzero")
  }
}

```

```

ss <- power.t.test(
  delta = abs(design_d) * sd,
  sd = sd,
  power = target_power,
  sig.level = alpha,
  type = "two.sample",
  alternative = "two.sided"
)
2 * ceiling(ss$n)
}

compute_eRTc <- function(treatment, outcome, p = 0.5, burn_in = 20,
  ramp = 50, c_max = 0.6,
  wager = c("adaptive", "design", "oracle"),
  adaptive_direction = c("sign", "cohens_d"),
  mu_ctrl_design = NULL, mu_trt_design = NULL,
  sd_design = NULL,
  mu_ctrl_oracle = NULL, mu_trt_oracle = NULL,
  sd_oracle = NULL,
  ramp_fixed = TRUE) {
  n <- length(treatment)
  if (length(outcome) != n) stop("treatment and outcome must have the same length")
  if (is.factor(treatment)) treatment <- as.numeric(treatment) - 1
  treatment <- as.numeric(treatment)
  wager <- match.arg(wager)
  adaptive_direction <- match.arg(adaptive_direction)

  if (wager == "design" &&
    (is.null(mu_ctrl_design) || is.null(mu_trt_design) || is.null(sd_design))) {
    stop("Design wager requires mu_ctrl_design, mu_trt_design, and sd_design")
  }
  if (wager == "oracle" &&
    (is.null(mu_ctrl_oracle) || is.null(mu_trt_oracle) || is.null(sd_oracle))) {
    stop("Oracle wager requires mu_ctrl_oracle, mu_trt_oracle, and sd_oracle")
  }

  wealth <- numeric(n)
  w <- 1

  for (i in seq_len(n)) {
    c_i <- min(1, max(0, (i - burn_in) / ramp))

    if (wager == "adaptive") {
      if ((i - 1) < burn_in) {
        lambda <- p
      } else {
        out_prev <- outcome[seq_len(i - 1)]

```

```

trt_prev <- treatment[seq_len(i - 1)]

med_prev <- median(out_prev, na.rm = TRUE)
mad_prev <- mad(out_prev, center = med_prev, constant = 1, na.rm = TRUE)
if (!is.finite(mad_prev) || mad_prev <= 0) mad_prev <- 1

s_i <- (outcome[[i]] - med_prev) / mad_prev
g_i <- s_i / (1 + abs(s_i))

mean_trt <- mean(out_prev[trt_prev == 1])
mean_ctrl <- mean(out_prev[trt_prev == 0])
if (is.nan(mean_trt) || is.nan(mean_ctrl)) {
  direction <- 0
} else if (adaptive_direction == "cohens_d") {
  d_hat <- cohens_d_estimate(trt_prev, out_prev)
  if (!is.finite(d_hat)) d_hat <- 0
  direction <- max(-1, min(1, d_hat))
} else {
  direction <- sign(mean_trt - mean_ctrl)
}

lambda <- p + c_i * c_max * g_i * direction
}
} else {
  if (wager == "design") {
    target <- continuous_design_lambda(
      outcome[[i]],
      mu_ctrl = mu_ctrl_design,
      mu_trt = mu_trt_design,
      sd = sd_design,
      p_random = p
    )
  } else {
    target <- continuous_design_lambda(
      outcome[[i]],
      mu_ctrl = mu_ctrl_oracle,
      mu_trt = mu_trt_oracle,
      sd = sd_oracle,
      p_random = p
    )
  }
  lambda <- if (ramp_fixed) p + c_i * (target - p) else target
}

lambda <- max(0.001, min(0.999, lambda))
multiplier <- if (treatment[[i]] == 1) lambda / p else (1 - lambda) / (1 - p)
w <- w * multiplier
wealth[[i]] <- w

```

```

}

wealth
}

```

## ESM Section 4. Additional details on practical deployment

This section documents the worked-alert example of Figure 3 (how the displayed trial was selected, the reproduction checks, and the code that replays it) and quantifies two deployment realities: inspecting the wealth path in batches and a constant outcome-ascertainment lag.

### 4.1 The worked-alert (Figure 3)

Figure 3 shows one simulated trial, not an average. The trial is selected deterministically from the frozen per-trial output of the union-doctrine simulation (5,000 alternative-scenario trials, seed 20260542). Candidates are trials whose alert fired before the second scheduled look, whose companion one-sided  $z$  at the alert lay between 2.35 and 2.65 (around the frozen median of about 2.5) and below the Repair A bar of 3.0, and whose scheduled design stopped at the second look. Among the 1,019 candidates, the displayed trial (simulation index 447) is the one whose lead (patients between the alert and the scheduled stop) equals the subset median of 483. The monitor is e-RTb with the design wager (anticipating the 35% versus 30% benefit). The replay regenerates that trial's assignment and outcome stream under the frozen seed and recomputes the wealth and  $z$  paths; the script stops with an error unless the replayed alert patient,  $z$  at alert, and look-2 stop reproduce the stored values exactly. Result: alert at patient 1,310;  $z$  at alert 2.56; lead 483 patients, 18% of maximum enrollment ( $N = 2,690$ ).

Code: the replay script printed in Section 4.3 (deterministic; frozen stream seed 20260542). Kernel: Supp-Code\_eRTb.R (Section 1.5).

### 4.2 Batched inspection and ascertainment delay

Wagers must be locked in patient order, but the wealth path may be computed and inspected in batches, and outcome confirmation may lag enrollment. Neither touches validity: inspecting a subset of times can only lower the supremum, so Ville's bound still holds and sub-sampled inspection is conservative; a constant lag changes the calendar position of each alert, not the path. The two tables quantify both on the frozen fair-comparator streams (seeds 20260542/43; wealth inspected at multiples of the batch size, final look always included).

**Table S11.** Batched inspection of the wealth path (5,000 trials per scenario; Monte Carlo standard errors in parentheses, percentage points).

| Scenario                   | Method         | Inspect every | Crossing rate | Median crossing |
|----------------------------|----------------|---------------|---------------|-----------------|
| Alternative: 35%<br>vs 30% | e-RTb adaptive | 1             | 45.9% (0.70)  | 1,343           |
| Alternative: 35%<br>vs 30% | e-RTb adaptive | 50            | 41.7% (0.70)  | 1,525           |

| Scenario                | Method                   | Inspect every | Crossing rate | Median crossing |
|-------------------------|--------------------------|---------------|---------------|-----------------|
| Alternative: 35% vs 30% | e-RTb adaptive           | 100           | 40.0% (0.69)  | 1,600           |
| Alternative: 35% vs 30% | e-RTb adaptive           | 250           | 37.2% (0.68)  | 1,750           |
| Alternative: 35% vs 30% | e-RTb design directional | 1             | 74.9% (0.61)  | 1,282           |
| Alternative: 35% vs 30% | e-RTb design directional | 50            | 72.4% (0.63)  | 1,350           |
| Alternative: 35% vs 30% | e-RTb design directional | 100           | 71.1% (0.64)  | 1,400           |
| Alternative: 35% vs 30% | e-RTb design directional | 250           | 69.0% (0.65)  | 1,500           |
| Alternative: 35% vs 30% | e-RTb mixture two-sided  | 1             | 64.4% (0.68)  | 1,506           |
| Alternative: 35% vs 30% | e-RTb mixture two-sided  | 50            | 62.1% (0.69)  | 1,600           |
| Alternative: 35% vs 30% | e-RTb mixture two-sided  | 100           | 60.9% (0.69)  | 1,600           |
| Alternative: 35% vs 30% | e-RTb mixture two-sided  | 250           | 59.0% (0.70)  | 1,750           |
| Null: 35% vs 35%        | e-RTb adaptive           | 1             | 3.1% (0.25)   | 421             |
| Null: 35% vs 35%        | e-RTb adaptive           | 50            | 2.0% (0.20)   | 625             |
| Null: 35% vs 35%        | e-RTb adaptive           | 100           | 1.5% (0.17)   | 700             |
| Null: 35% vs 35%        | e-RTb adaptive           | 250           | 1.2% (0.15)   | 750             |
| Null: 35% vs 35%        | e-RTb design directional | 1             | 4.1% (0.28)   | 1,358           |
| Null: 35% vs 35%        | e-RTb design directional | 50            | 3.2% (0.25)   | 1,400           |
| Null: 35% vs 35%        | e-RTb design directional | 100           | 3.0% (0.24)   | 1,500           |
| Null: 35% vs 35%        | e-RTb design directional | 250           | 2.5% (0.22)   | 1,500           |
| Null: 35% vs 35%        | e-RTb mixture two-sided  | 1             | 3.3% (0.25)   | 1,655           |
| Null: 35% vs 35%        | e-RTb mixture two-sided  | 50            | 2.7% (0.23)   | 1,750           |
| Null: 35% vs 35%        | e-RTb mixture two-sided  | 100           | 2.3% (0.21)   | 1,750           |
| Null: 35% vs 35%        | e-RTb mixture two-sided  | 250           | 1.8% (0.19)   | 1,750           |

Inspecting the design wager only every 100 patients costs 3.8 crossing points (71.1% versus 74.9% continuous) and moves the median alert from 1,282 to 1,400, still about 400 patients before the second scheduled look. Null rates fall under batching (2.0–3.1% at every-50), as the supremum argument implies.

**Table S12.** Constant ascertainment lag of  $L$  patients (post-processing of the frozen alternative-scenario first

crossings; no new simulation).

| Method         | Lag $L$ (patients) | Crossing rate (unchanged) | Alert before enrollment ends | Median calendar alert |
|----------------|--------------------|---------------------------|------------------------------|-----------------------|
| e-RTb adaptive | 0                  | 45.9%                     | 45.9%                        | 1,343                 |
| e-RTb design   | 0                  | 74.9%                     | 74.9%                        | 1,282                 |
| directional    |                    |                           |                              |                       |
| e-RTb mixture  | 0                  | 64.4%                     | 64.4%                        | 1,506                 |
| two-sided      |                    |                           |                              |                       |
| e-RTb adaptive | 100                | 45.9%                     | 44.1%                        | 1,443                 |
| e-RTb design   | 100                | 74.9%                     | 73.3%                        | 1,382                 |
| directional    |                    |                           |                              |                       |
| e-RTb mixture  | 100                | 64.4%                     | 62.6%                        | 1,606                 |
| two-sided      |                    |                           |                              |                       |
| e-RTb adaptive | 300                | 45.9%                     | 40.9%                        | 1,643                 |
| e-RTb design   | 300                | 74.9%                     | 69.6%                        | 1,582                 |
| directional    |                    |                           |                              |                       |
| e-RTb mixture  | 300                | 64.4%                     | 58.5%                        | 1,806                 |
| two-sided      |                    |                           |                              |                       |

Under a constant lag the update order is unchanged, so the wealth path, and with it the crossing rate, is identical; every alert lands  $L$  enrollments later on the calendar (capped at enrollment end). Even at  $L = 300$ , a design-wager alert precedes the end of enrollment in 69.6% of all simulated trials (the “before enrollment end” column is an all-trials rate; among the 74.9% of trials that alerted, this is 93.0%).

### 4.3 Code for the worked-alert replay

The listing below is the complete replay script behind Figure 3 and the worked-alert numbers. It reads only frozen outputs, asserts exact reproduction, and writes new files. It is printed for audit and runs inside the authors’ full analysis tree (its source() lines and input paths refer to that tree); the self-contained kernels are in the repository named above.

```
# Worked-alert replay for the practical-deployment section.
# DETERMINISTIC: selects one representative trial from the frozen
# per-trial output of the union-doctrine simulation
# (tables/union_doctrine_first_crossings.csv) and replays
# its stream under the frozen seed to recover the full wealth and z paths.
# No new randomness; the replayed trial must reproduce its CSV row exactly
# (asserted below) or the script stops.
#
# Selection (Alternative: 35% vs 30%): alert fired before look 2 (fc20 < 1,793),
# companion z at alert below the Repair-A bar (z < 3.0) and near the frozen
# median (2.35–2.65), GS stop at look 2 (gs_time == 1,793); among those, the
# trial whose lead (1,793 - fc20) is closest to the subset median; smallest
# sim index breaks ties.
#
```

```

# Outputs: figures/fig3_worked_alert.{png,pdf}
#          tables/worked_alert_replay.md    (ALERT_N / ALERT_Z / LEAD_N / LEAD_PCT)

source("scripts/paths.R")
suppressPackageStartupMessages(source(local_ert_file("ertb.R")))
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(ggplot2))
suppressPackageStartupMessages(library(patchwork))

root <- paper_root()
table_dir <- paper_file("tables", root = root, must_work = FALSE)
fig_dir <- paper_file("figures", root = root, must_work = FALSE)

# frozen design constants (mirror make_ertb_fair_comparators.R)
seed_alt <- 20260542
n_patients <- 2690
p_ctrl <- 0.35
p_trt_alt <- 0.30
look3 <- round(n_patients * (1:3) / 3)      # 897 1793 2690
bounds_2s <- c(3.710, 2.511, 1.993)        # LD-0BF K3 two-sided a=0.05 (pinned)
z_bar <- 3.0                                # Repair A confirmation bar
threshold <- 20

# trial machinery (identical to make_ertb_fair_comparators.R)
simulate_trial <- function(n, p_trt, p_ctrl, p_random = 0.5) {
  treatment <- rbinom(n, 1, p_random)
  outcome <- numeric(n)
  outcome[treatment == 1] <- rbinom(sum(treatment == 1), 1, p_trt)
  outcome[treatment == 0] <- rbinom(sum(treatment == 0), 1, p_ctrl)
  list(treatment = treatment, outcome = outcome)
}

z_at_look <- function(treatment, outcome, n_look) {
  idx <- seq_len(n_look)
  trt <- treatment[idx] == 1
  ctrl <- treatment[idx] == 0
  n_t <- sum(trt)
  n_c <- sum(ctrl)
  if (n_t == 0 || n_c == 0) return(NA_real_)
  p_t <- mean(outcome[idx][trt])
  p_c <- mean(outcome[idx][ctrl])
  p_pool <- mean(outcome[idx])
  se <- sqrt(p_pool * (1 - p_pool) * (1 / n_t + 1 / n_c))
  if (!is.finite(se) || se <= 0) return(NA_real_)
  (p_c - p_t) / se
}

# -----

```

```

# 1. Select the representative trial from the frozen per-trial CSV
# -----

fc <- read.csv(file.path(table_dir, "union_doctrine_first_crossings.csv"))

cand <- fc %>%
  filter(scenario == "Alternative: 35% vs 30%",
         gs_time == look3[2],
         !is.na(fc20), fc20 < look3[2],
         !is.na(z_at_alert), z_at_alert < z_bar,
         z_at_alert >= 2.35, z_at_alert <= 2.65) %>%
  mutate(lead = look3[2] - fc20)

stopifnot(nrow(cand) > 0)
med_lead <- median(cand$lead)
pick <- cand %>%
  arrange(abs(lead - med_lead), sim) %>%
  slice(1)
cat(sprintf("Picked sim %d: alert at %d, z at alert %.3f, lead %d (subset median lead %.0f, n candid
            pick$sim, pick$fc20, pick$z_at_alert, pick$lead, med_lead, nrow(cand)))

# -----
# 2. Replay the stream to the picked trial (kernels consume no RNG)
# -----

set.seed(seed_alt)
for (s in seq_len(pick$sim)) {
  trial <- simulate_trial(n_patients, p_trt = p_trt_alt, p_ctrl = p_ctrl)
}

w_up <- compute_eRT(trial$treatment, trial$outcome,
                   wager = "design",
                   p_trt_design = p_trt_alt, p_ctrl_design = p_ctrl,
                   ramp_fixed = FALSE)
z_seq <- vapply(seq_len(look3[2]),
               function(n) z_at_look(trial$treatment, trial$outcome, n),
               numeric(1))

# reproduction checks: the replayed trial must match its frozen CSV row
alert_n <- which(w_up >= threshold)[1]
stopifnot(identical(as.integer(alert_n), as.integer(pick$fc20)))
stopifnot(abs(z_seq[alert_n] - pick$z_at_alert) < 1e-9)
stopifnot(abs(z_seq[look3[1]]) < bounds_2s[1]) # no GS stop at look 1
stopifnot(abs(z_seq[look3[2]]) >= bounds_2s[2]) # GS stop at look 2
cat("Reproduction checks passed: replayed trial matches the frozen CSV row.\n")

alert_z <- z_seq[alert_n]
lead_n <- look3[2] - alert_n

```

```

lead_pct <- 100 * lead_n / n_patients

# -----
# 3. Values for the manuscript box
# -----

md <- c(
  "# Worked-alert replay (practical-deployment section)",
  "",
  sprintf("Frozen stream: seed %d, sim %d (Alternative: 35%% vs 30%%, N = %d).",
    seed_alt, pick$sim, n_patients),
  "",
  "| Placeholder | Value |",
  "| --- | --- |",
  sprintf("| ALERT_N | %d |", alert_n),
  sprintf("| ALERT_Z | %.2f |", alert_z),
  sprintf("| LEAD_N | %d |", lead_n),
  sprintf("| LEAD_PCT | %.0f%% |", lead_pct),
  "",
  sprintf("Candidates matching the selection window: %d; subset median lead %.0f.",
    nrow(cand), med_lead),
  "Reproduction checks (CSV row, look-1 no-cross, look-2 cross) all passed."
)
writeLines(md, file.path(table_dir, "worked_alert_replay.md"))

# -----
# 4. Figure: wealth path (A) and companion z path (B), aligned on patients
# -----

x_max <- look3[2] + 60
df_w <- data.frame(n = seq_len(look3[2]), w = w_up[seq_len(look3[2])])
df_z <- data.frame(n = seq_len(look3[2]), z = z_seq) %>% filter(is.finite(z), n >= 30)

look_lines <- data.frame(n = look3[1:2], lab = c("look 1", "look 2 - trial stops"))

pA <- ggplot(df_w, aes(n, w)) +
  geom_vline(xintercept = look3[1:2], linetype = "dotted", colour = "#9a9a9a", linewidth = 0.4) +
  geom_hline(yintercept = threshold, linetype = "dashed", colour = "#b23a48", linewidth = 0.45) +
  geom_line(colour = "#0072B2", linewidth = 0.55) +
  geom_point(data = data.frame(n = alert_n, w = w_up[alert_n]),
    colour = "#0072B2", size = 2.2) +
  annotate("text", x = alert_n, y = w_up[alert_n], label = sprintf("alert (patient %d)", alert_n),
    hjust = 1.05, vjust = -0.8, size = 2.9, colour = "#1a1a1a") +
  annotate("text", x = 20, y = threshold, label = "W = 20", hjust = 0, vjust = -0.6,
    size = 2.7, colour = "#b23a48") +
  scale_y_log10() +
  scale_x_continuous(breaks = c(0, 500, look3[1], 1500, look3[2])) +
  coord_cartesian(xlim = c(0, x_max)) +

```

```

labs(x = NULL, y = "Wealth W (log scale)") +
theme_minimal(base_size = 10) +
theme(panel.grid.minor = element_blank(),
      panel.grid.major = element_line(colour = "#ecec", linewidth = 0.3),
      axis.text.x = element_blank(),
      plot.background = element_rect(fill = "white", colour = NA))

df_bound <- data.frame(n = look3[1:2], z = bounds_2s[1:2])

pB <- ggplot(df_z, aes(n, z)) +
  geom_vline(xintercept = look3[1:2], linetype = "dotted", colour = "#9a9a9a", linewidth = 0.4) +
  geom_hline(yintercept = z_bar, linetype = "dashed", colour = "#b23a48", linewidth = 0.45) +
  geom_line(colour = "#4d4d4d", linewidth = 0.5) +
  geom_point(data = df_bound, aes(n, z), shape = 21, fill = "white",
            colour = "#E69F00", size = 2.6, stroke = 0.9) +
  geom_point(data = data.frame(n = alert_n, z = alert_z),
            colour = "#0072B2", size = 2.2) +
  annotate("text", x = alert_n, y = alert_z,
          label = sprintf("z = %.2f at alert", alert_z),
          hjust = 1.08, vjust = 2.6, size = 2.9, colour = "#1a1a1a") +
  annotate("text", x = 20, y = z_bar, label = "Repair A bar, z = 3.0", hjust = 0,
          vjust = -0.6, size = 2.7, colour = "#b23a48") +
  annotate("text", x = look3[1], y = bounds_2s[1], label = "scheduled boundary",
          hjust = -0.08, vjust = 0.4, size = 2.7, colour = "#8a5b00") +
  scale_x_continuous(breaks = c(0, 500, look3[1], 1500, look3[2])) +
  coord_cartesian(xlim = c(0, x_max)) +
  labs(x = "Patients enrolled", y = "One-sided z") +
  theme_minimal(base_size = 10) +
  theme(panel.grid.minor = element_blank(),
        panel.grid.major = element_line(colour = "#ecec", linewidth = 0.3),
        plot.background = element_rect(fill = "white", colour = NA))

p <- pA / pB + plot_annotation(tag_levels = "A")

ggsave(file.path(fig_dir, "fig3_worked_alert.png"), p, width = 7.2, height = 5.6,
       dpi = 360, bg = "white")
ggsave(file.path(fig_dir, "fig3_worked_alert.pdf"), p, width = 7.2, height = 5.6,
       device = cairo_pdf, bg = "white")
cat("Wrote fig3_worked_alert.png/.pdf and tables/worked_alert_replay.md\n")

```

## ESM Section 5. Additional details on the comparison with calibrated designs

This section carries the full grids behind the comparison with calibrated designs: the calibrated boundary values, the complete fourteen-method operating characteristics, the cumulative-crossing table at the scheduled looks, the union path decomposition, the complete design-versus-actual grid, the conditional shared-crossing comparisons carried from the submission, and a fixed-horizon decomposition of the design-wager premium.

## 5.1 Calibrated boundary values

Boundaries were computed with `rpact` (error-spending calibration) and verified against an independent multivariate-normal recursion; the maximum deviation across all boundary values is below 0.005. One-sided designs at  $\alpha = 0.025$  use the same critical values as two-sided designs at  $\alpha = 0.05$ , as expected. The last two rows are the recalibrated schedules used by Repair B.

**Table S13.** Critical values (z-scale) for the calibrated error-spending designs, listed first to last look.

| Design                                                              | Critical values                   |
|---------------------------------------------------------------------|-----------------------------------|
| LD-OBF $K = 3$ two-sided $\alpha = 0.05$                            | 3.710, 2.511, 1.993               |
| LD-OBF $K = 5$ two-sided $\alpha = 0.05$                            | 4.877, 3.357, 2.680, 2.290, 2.031 |
| LD-OBF $K = 3$ one-sided $\alpha = 0.05$                            | 3.200, 2.141, 1.695               |
| LD-OBF $K = 5$ one-sided $\alpha = 0.05$                            | 4.229, 2.888, 2.298, 1.962, 1.740 |
| LD-OBF $K = 3$ one-sided $\alpha = 0.025$                           | 3.710, 2.511, 1.993               |
| LD-OBF $K = 5$ one-sided $\alpha = 0.025$                           | 4.877, 3.357, 2.680, 2.290, 2.031 |
| KD $\rho = 1$ $K = 5$ one-sided $\alpha = 0.05$                     | 2.326, 2.219, 2.120, 2.033, 1.956 |
| KD $\rho = 3$ $K = 5$ one-sided $\alpha = 0.05$                     | 3.353, 2.753, 2.350, 2.019, 1.722 |
| LD-OBF $K = 3$ two-sided $\alpha = 0.04$ (Repair B, $W \geq 100$ )  | 3.863, 2.623, 2.083               |
| LD-OBF $K = 3$ two-sided $\alpha = 0.045$ (Repair B, $W \geq 200$ ) | 3.783, 2.564, 2.036               |

## 5.2 Full operating characteristics, all fourteen methods

**Table S14.** Operating characteristics in the binary simulation, all methods on the same simulated streams (5,000 trials per scenario; Monte Carlo standard errors in parentheses, percentage points). Alternative: 35% versus 30%; null: 35% versus 35%; harm: 35% versus 40%.

| Method                              | Alternative crossing | Alt. median crossing | Alt. $\mathbb{E}[\min(\tau, N)]$ | Null crossing | Harm-side crossing | Harm median crossing |
|-------------------------------------|----------------------|----------------------|----------------------------------|---------------|--------------------|----------------------|
| OBF two-sided                       | 79.5% (0.57)         | 1,793                | 2,259                            | 5.9% (0.33)   | 77.7% (0.59)       | 1,793                |
| Pocock two-sided                    | 71.6% (0.64)         | 1,793                | 2,012                            | 5.1% (0.31)   | 69.4% (0.65)       | 1,793                |
| Haybittle-Peto                      | 78.9% (0.58)         | 2,690                | 2,505                            | 5.4% (0.32)   | 77.2% (0.59)       | 2,690                |
| LD-OBF K3 two-sided $\alpha = 0.05$ | 78.3% (0.58)         | 1,793                | 2,316                            | 5.2% (0.31)   | 76.6% (0.60)       | 2,690                |
| LD-OBF K5 two-sided $\alpha = 0.05$ | 77.7% (0.59)         | 2,152                | 2,188                            | 5.2% (0.31)   | 76.0% (0.60)       | 2,152                |

| Method                                     | Alternative crossing | Alt. median crossing | Alt. $\mathbb{E}[\min(\tau, N)]$ | Null crossing | Harm-side crossing | Harm median crossing |
|--------------------------------------------|----------------------|----------------------|----------------------------------|---------------|--------------------|----------------------|
| LD-OBF K3 one-sided $\alpha = 0.05$        | 86.5% (0.48)         | 1,793                | 2,156                            | 5.3% (0.32)   | 0.0% (0.00)        | –                    |
| LD-OBF K5 one-sided $\alpha = 0.05$        | 86.0% (0.49)         | 1,614                | 2,003                            | 5.2% (0.31)   | 0.0% (0.00)        | –                    |
| LD-OBF K3 one-sided $\alpha = 0.025$       | 78.3% (0.58)         | 1,793                | 2,316                            | 2.7% (0.23)   | 0.0% (0.00)        | –                    |
| LD-OBF K5 one-sided $\alpha = 0.025$       | 77.7% (0.59)         | 2,152                | 2,188                            | 2.8% (0.23)   | 0.0% (0.00)        | –                    |
| KD $\rho = 1$ K5 one-sided $\alpha = 0.05$ | 82.2% (0.54)         | 1,614                | 1,760                            | 5.2% (0.31)   | 0.0% (0.02)        | 538                  |
| KD $\rho = 3$ K5 one-sided $\alpha = 0.05$ | 86.4% (0.49)         | 2,152                | 1,997                            | 5.2% (0.31)   | 0.0% (0.00)        | –                    |
| e-RTb adaptive                             | 45.9% (0.70)         | 1,343                | 2,081                            | 3.1% (0.25)   | 44.4% (0.70)       | 1,359                |
| e-RTb design directional                   | 74.9% (0.61)         | 1,282                | 1,693                            | 4.1% (0.28)   | 0.0% (0.00)        | –                    |
| e-RTb mixture two-sided                    | 64.4% (0.68)         | 1,506                | 1,952                            | 3.3% (0.25)   | 61.4% (0.69)       | 1,508                |

### 5.3 Cumulative crossing at the scheduled looks

**Table S15.** Cumulative probability of first crossing by the stated patient count, all 5,000 simulated trials per scenario as denominator. Patients 897/1,793/2,690 are the  $K = 3$  schedule at one-third, two-thirds, and full information.

| Scenario                | Method           | By 897 | By 1,793 | By 2,690 |
|-------------------------|------------------|--------|----------|----------|
| Alternative: 35% vs 30% | OBF two-sided    | 3.4%   | 44.7%    | 79.5%    |
| Alternative: 35% vs 30% | Pocock two-sided | 23.8%  | 51.8%    | 71.6%    |
| Alternative: 35% vs 30% | Haybittle-Peto   | 4.1%   | 16.5%    | 78.9%    |

| Scenario                | Method                                           | By 897 | By 1,793 | By 2,690 |
|-------------------------|--------------------------------------------------|--------|----------|----------|
| Alternative: 35% vs 30% | LD-OBF K3<br>two-sided<br>$\alpha = 0.05$        | 1.5%   | 40.1%    | 78.3%    |
| Alternative: 35% vs 30% | LD-OBF K5<br>two-sided<br>$\alpha = 0.05$        | 0.0%   | 29.7%    | 77.7%    |
| Alternative: 35% vs 30% | LD-OBF K3<br>one-sided<br>$\alpha = 0.05$        | 5.2%   | 54.3%    | 86.5%    |
| Alternative: 35% vs 30% | LD-OBF K5<br>one-sided<br>$\alpha = 0.05$        | 0.2%   | 44.5%    | 86.0%    |
| Alternative: 35% vs 30% | LD-OBF K3<br>one-sided<br>$\alpha = 0.025$       | 1.5%   | 40.1%    | 78.3%    |
| Alternative: 35% vs 30% | LD-OBF K5<br>one-sided<br>$\alpha = 0.025$       | 0.0%   | 29.7%    | 77.7%    |
| Alternative: 35% vs 30% | KD $\rho = 1$ K5<br>one-sided<br>$\alpha = 0.05$ | 13.0%  | 54.7%    | 82.2%    |
| Alternative: 35% vs 30% | KD $\rho = 3$ K5<br>one-sided<br>$\alpha = 0.05$ | 1.6%   | 43.0%    | 86.4%    |
| Alternative: 35% vs 30% | e-RTb adaptive                                   | 14.8%  | 30.8%    | 45.9%    |
| Alternative: 35% vs 30% | e-RTb design<br>directional                      | 20.5%  | 55.5%    | 74.9%    |
| Alternative: 35% vs 30% | e-RTb mixture<br>two-sided                       | 9.7%   | 42.0%    | 64.4%    |
| Harm: 35% vs 40%        | OBF two-sided                                    | 3.0%   | 41.4%    | 77.7%    |
| Harm: 35% vs 40%        | Pocock two-sided                                 | 22.0%  | 48.7%    | 69.4%    |
| Harm: 35% vs 40%        | Haybittle-Peto                                   | 3.9%   | 15.1%    | 77.2%    |
| Harm: 35% vs 40%        | LD-OBF K3<br>two-sided<br>$\alpha = 0.05$        | 1.4%   | 37.1%    | 76.6%    |
| Harm: 35% vs 40%        | LD-OBF K5<br>two-sided<br>$\alpha = 0.05$        | 0.0%   | 27.4%    | 76.0%    |
| Harm: 35% vs 40%        | LD-OBF K3<br>one-sided<br>$\alpha = 0.05$        | 0.0%   | 0.0%     | 0.0%     |

| Scenario         | Method                                           | By 897 | By 1,793 | By 2,690 |
|------------------|--------------------------------------------------|--------|----------|----------|
| Harm: 35% vs 40% | LD-OBF K5<br>one-sided<br>$\alpha = 0.05$        | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | LD-OBF K3<br>one-sided<br>$\alpha = 0.025$       | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | LD-OBF K5<br>one-sided<br>$\alpha = 0.025$       | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | KD $\rho = 1$ K5<br>one-sided<br>$\alpha = 0.05$ | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | KD $\rho = 3$ K5<br>one-sided<br>$\alpha = 0.05$ | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | e-RTb adaptive                                   | 14.8%  | 29.8%    | 44.4%    |
| Harm: 35% vs 40% | e-RTb design<br>directional                      | 0.0%   | 0.0%     | 0.0%     |
| Harm: 35% vs 40% | e-RTb mixture<br>two-sided                       | 10.3%  | 39.5%    | 61.4%    |
| Null: 35% vs 35% | OBF two-sided                                    | 0.1%   | 1.7%     | 5.9%     |
| Null: 35% vs 35% | Pocock two-sided                                 | 2.0%   | 3.7%     | 5.1%     |
| Null: 35% vs 35% | Haybittle-Peto                                   | 0.1%   | 0.2%     | 5.4%     |
| Null: 35% vs 35% | LD-OBF K3<br>two-sided<br>$\alpha = 0.05$        | 0.0%   | 1.3%     | 5.2%     |
| Null: 35% vs 35% | LD-OBF K5<br>two-sided<br>$\alpha = 0.05$        | 0.0%   | 0.7%     | 5.2%     |
| Null: 35% vs 35% | LD-OBF K3<br>one-sided<br>$\alpha = 0.05$        | 0.1%   | 1.8%     | 5.3%     |
| Null: 35% vs 35% | LD-OBF K5<br>one-sided<br>$\alpha = 0.05$        | 0.0%   | 1.1%     | 5.2%     |
| Null: 35% vs 35% | LD-OBF K3<br>one-sided<br>$\alpha = 0.025$       | 0.0%   | 0.7%     | 2.7%     |
| Null: 35% vs 35% | LD-OBF K5<br>one-sided<br>$\alpha = 0.025$       | 0.0%   | 0.4%     | 2.8%     |
| Null: 35% vs 35% | KD $\rho = 1$ K5<br>one-sided<br>$\alpha = 0.05$ | 1.0%   | 3.0%     | 5.2%     |

| Scenario         | Method                                           | By 897 | By 1,793 | By 2,690 |
|------------------|--------------------------------------------------|--------|----------|----------|
| Null: 35% vs 35% | KD $\rho = 3$ K5<br>one-sided<br>$\alpha = 0.05$ | 0.1%   | 1.0%     | 5.2%     |
| Null: 35% vs 35% | e-RTb adaptive                                   | 2.3%   | 2.8%     | 3.1%     |
| Null: 35% vs 35% | e-RTb design<br>directional                      | 1.2%   | 2.9%     | 4.1%     |
| Null: 35% vs 35% | e-RTb mixture<br>two-sided                       | 0.5%   | 2.0%     | 3.3%     |

#### 5.4 Union path decomposition and alert lead time

Which component caused each stop under the role-3 rules, and how much warning the role-1 alert gives. “Alert fired, unconfirmed” counts trials in which the alert fired but the confirmation bar was not met at the alert instant (Repair A only; under Repair B the trigger itself is the alert path).

**Table S16.** Path attribution for the union rules (5,000 trials per scenario).

| Scenario                | Rule                                                  | Stop rate | Via alert path | Via scheduled GS | Alert fired | Alert fired, un-confirmed | Median lead when alert won (patients) |
|-------------------------|-------------------------------------------------------|-----------|----------------|------------------|-------------|---------------------------|---------------------------------------|
| Alternative: 35% vs 30% | Repair A<br>$z \geq 3.0$                              | 78.4%     | 7.1%           | 71.4%            | 74.9%       | 67.8%                     | 1,317                                 |
| Alternative: 35% vs 30% | Repair A<br>$z \geq 2.5$                              | 80.0%     | 46.2%          | 33.8%            | 74.9%       | 28.6%                     | 945                                   |
| Alternative: 35% vs 30% | Repair B<br>$W \geq 100$<br>(+GS<br>$\alpha = 0.04$ ) | 75.8%     | 41.2%          | 34.6%            | 50.4%       | —                         | —                                     |
| Alternative: 35% vs 30% | Repair B<br>$W \geq 200$<br>(+GS $\alpha = 0.045$ )   | 77.0%     | 23.7%          | 53.3%            | 38.8%       | —                         | —                                     |
| Null: 35% vs 35%        | Repair A<br>$z \geq 3.0$                              | 5.6%      | 0.5%           | 5.1%             | 4.1%        | 3.6%                      | 1,178                                 |
| Null: 35% vs 35%        | Repair A<br>$z \geq 2.5$                              | 6.7%      | 2.3%           | 4.4%             | 4.1%        | 1.8%                      | 878                                   |
| Null: 35% vs 35%        | Repair B<br>$W \geq 100$<br>(+GS<br>$\alpha = 0.04$ ) | 4.3%      | 0.4%           | 3.9%             | 0.5%        | —                         | —                                     |

| Scenario         | Rule                                                | Stop rate | Via alert path | Via scheduled GS | Alert fired | Alert fired, un-confirmed | Median lead when alert won (patients) |
|------------------|-----------------------------------------------------|-----------|----------------|------------------|-------------|---------------------------|---------------------------------------|
| Null: 35% vs 35% | Repair B<br>$W \geq 200$<br>(+GS $\alpha = 0.045$ ) | 4.7%      | 0.1%           | 4.6%             | 0.3%        | –                         | –                                     |
| Harm: 35% vs 40% | Repair A<br>$z \geq 3.0$                            | 76.6%     | 0.0%           | 76.6%            | 0.0%        | 0.0%                      | –                                     |
| Harm: 35% vs 40% | Repair A<br>$z \geq 2.5$                            | 76.6%     | 0.0%           | 76.6%            | 0.0%        | 0.0%                      | –                                     |
| Harm: 35% vs 40% | Repair B<br>$W \geq 100$<br>(+GS $\alpha = 0.04$ )  | 73.7%     | 0.0%           | 73.7%            | 0.0%        | –                         | –                                     |
| Harm: 35% vs 40% | Repair B<br>$W \geq 200$<br>(+GS $\alpha = 0.045$ ) | 75.2%     | 0.0%           | 75.2%            | 0.0%        | –                         | –                                     |

**Table S17.** Role-1 surveillance value: among trials where the scheduled design (LD-OBF  $K = 3$  two-sided) stopped, how many had a preceding e-RTb design alert, and by how many patients.

| Scenario                | GS-stopped trials | Preceded by e-RT alert | Median lead (patients) | Q1–Q3 lead | Median $z$ at alert |
|-------------------------|-------------------|------------------------|------------------------|------------|---------------------|
| Alternative: 35% vs 30% | 78.3%             | 91.8%                  | 776                    | 445–1,105  | 2.54                |
| Null: 35% vs 35%        | 5.2%              | 39.1%                  | 582                    | 292–1,138  | 2.49                |
| Harm: 35% vs 40%        | 76.6%             | 0.0%                   | –                      | –          | NA                  |

## 5.5 Complete design-versus-actual grid

The design columns fix the conventional fixed-sample size (two-sided, at the stated target power) and the design wager; the actual column fixes the data-generating rates. The last sixteen rows reverse the actual effect: treatment is worse than control while the wager stays calibrated to the anticipated benefit. “Actual frequentist power” is the power of a single fixed-sample two-sided two-proportion  $z$ -test at  $\alpha = 0.05$  under the actual rates, with no interim analyses (`power.prop.test`).

**Table S18.** Design-versus-actual sensitivity, full grid (5,000 trials per cell).

| Design     | Actual effect | Design power | e-RTb adaptive | e-RTb design | Actual frequentist power |
|------------|---------------|--------------|----------------|--------------|--------------------------|
| 40% vs 35% | 40% vs 35%    | 80%          | 48.4%          | 75.8%        | 80.0%                    |
| 40% vs 35% | 40% vs 30%    | 80%          | 99.4%          | 100.0%       | 100.0%                   |
| 40% vs 35% | 40% vs 35%    | 90%          | 63.6%          | 86.8%        | 90.0%                    |
| 40% vs 35% | 40% vs 30%    | 90%          | 100.0%         | 100.0%       | 100.0%                   |
| 40% vs 30% | 40% vs 35%    | 80%          | 10.6%          | 26.7%        | 28.0%                    |
| 40% vs 30% | 40% vs 30%    | 80%          | 52.6%          | 76.8%        | 80.0%                    |
| 40% vs 30% | 40% vs 35%    | 90%          | 15.7%          | 33.6%        | 35.7%                    |
| 40% vs 30% | 40% vs 30%    | 90%          | 65.7%          | 85.6%        | 90.1%                    |
| 25% vs 20% | 25% vs 20%    | 80%          | 39.5%          | 76.1%        | 80.0%                    |
| 25% vs 20% | 25% vs 15%    | 80%          | 97.5%          | 100.0%       | 100.0%                   |
| 25% vs 20% | 25% vs 20%    | 90%          | 52.1%          | 87.0%        | 90.0%                    |
| 25% vs 20% | 25% vs 15%    | 90%          | 99.6%          | 100.0%       | 100.0%                   |
| 25% vs 15% | 25% vs 20%    | 80%          | 7.3%           | 25.5%        | 26.7%                    |
| 25% vs 15% | 25% vs 15%    | 80%          | 36.3%          | 76.6%        | 80.0%                    |
| 25% vs 15% | 25% vs 20%    | 90%          | 10.1%          | 32.3%        | 34.1%                    |
| 25% vs 15% | 25% vs 15%    | 90%          | 49.6%          | 86.1%        | 90.1%                    |
| 40% vs 35% | 40% vs 45%    | 80%          | 48.3%          | 0.0%         | 78.4%                    |
|            | (reversed)    |              |                |              |                          |
| 40% vs 35% | 40% vs 50%    | 80%          | 99.2%          | 0.0%         | 100.0%                   |
|            | (reversed)    |              |                |              |                          |
| 40% vs 35% | 40% vs 45%    | 90%          | 63.1%          | 0.0%         | 88.8%                    |
|            | (reversed)    |              |                |              |                          |
| 40% vs 35% | 40% vs 50%    | 90%          | 100.0%         | 0.0%         | 100.0%                   |
|            | (reversed)    |              |                |              |                          |
| 40% vs 30% | 40% vs 45%    | 80%          | 11.9%          | 0.2%         | 27.0%                    |
|            | (reversed)    |              |                |              |                          |
| 40% vs 30% | 40% vs 50%    | 80%          | 48.5%          | 0.0%         | 76.6%                    |
|            | (reversed)    |              |                |              |                          |
| 40% vs 30% | 40% vs 45%    | 90%          | 15.6%          | 0.3%         | 34.5%                    |
|            | (reversed)    |              |                |              |                          |
| 40% vs 30% | 40% vs 50%    | 90%          | 63.9%          | 0.0%         | 87.5%                    |
|            | (reversed)    |              |                |              |                          |
| 25% vs 20% | 25% vs 30%    | 80%          | 38.3%          | 0.1%         | 74.5%                    |
|            | (reversed)    |              |                |              |                          |
| 25% vs 20% | 25% vs 35%    | 80%          | 96.8%          | 0.0%         | 99.9%                    |
|            | (reversed)    |              |                |              |                          |
| 25% vs 20% | 25% vs 30%    | 90%          | 51.0%          | 0.0%         | 85.8%                    |
|            | (reversed)    |              |                |              |                          |
| 25% vs 20% | 25% vs 35%    | 90%          | 99.3%          | 0.0%         | 100.0%                   |
|            | (reversed)    |              |                |              |                          |
| 25% vs 15% | 25% vs 30%    | 80%          | 6.7%           | 0.3%         | 23.9%                    |
|            | (reversed)    |              |                |              |                          |
| 25% vs 15% | 25% vs 35%    | 80%          | 34.0%          | 0.0%         | 68.5%                    |
|            | (reversed)    |              |                |              |                          |

| Design     | Actual effect            | Design power | e-RTb adaptive | e-RTb design | Actual frequentist power |
|------------|--------------------------|--------------|----------------|--------------|--------------------------|
| 25% vs 15% | 25% vs 30%<br>(reversed) | 90%          | 10.4%          | 0.5%         | 30.4%                    |
| 25% vs 15% | 25% vs 35%<br>(reversed) | 90%          | 48.7%          | 0.1%         | 80.8%                    |

## 5.6 Conditional shared-crossing comparisons (carried from the submission)

The submission compared crossing times conditional on both e-RTb and a scheduled monitor crossing in the same trial. These columns are retained here, labeled as conditional: their denominator is the shared crossings, not all trials, so they answer “who was earlier when both saw it,” not “who saw more” — the unconditional view is Tables S14–S15. Type M is the magnification ratio  $|\widehat{ARR}_{\text{crossing}}|/|\widehat{ARR}_{\text{final}}|$  among alternative trials that crossed.

**Table S19.** Operating characteristics of the submission’s comparison (textbook boundaries; conditional columns as labeled).

| Method         | Alternative crossing | Median crossing | Null crossing | Median Type M | Earlier than OBF when both crossed | Earlier than Pocock when both crossed | Earlier than HP when both crossed |
|----------------|----------------------|-----------------|---------------|---------------|------------------------------------|---------------------------------------|-----------------------------------|
| OBF            | 79.5%                | 1,793           | 5.9%          | 1.00          | –                                  | –                                     | –                                 |
| two-sided      |                      |                 |               |               |                                    |                                       |                                   |
| Pocock         | 71.6%                | 1,793           | 5.1%          | 1.07          | –                                  | –                                     | –                                 |
| two-sided      |                      |                 |               |               |                                    |                                       |                                   |
| Haybittle-Peto | 78.9%                | 2,690           | 5.4%          | 1.00          | –                                  | –                                     | –                                 |
| e-RTb          | 45.9%                | 1,343           | 3.1%          | 1.26          | 80.6%                              | 62.3%                                 | 98.0%                             |
| adaptive       |                      |                 |               |               |                                    |                                       |                                   |
| e-RTb          | 74.9%                | 1,282           | 4.1%          | 1.19          | 99.5%                              | 90.5%                                 | 99.9%                             |
| design         |                      |                 |               |               |                                    |                                       |                                   |
| directional    |                      |                 |               |               |                                    |                                       |                                   |

As a matched-power robustness check, the submission re-ran the Pocock comparison at  $N = 3,084$ , the sample size at which Pocock reaches 80% power for the same effect. Pocock’s alternative crossing rose to 78.3% (null 4.9%), confirming the boundary’s calibration; design e-RTb at that larger  $N$  crossed in 80.0% of trials and was earlier than Pocock in 92.8% of shared crossings (adaptive: 51.7% and 62.8%), so the qualitative conclusion is unchanged.

## 5.7 The design-wager premium decomposed

Figure 5 shows design e-RTb about 11 points below the calibrated one-sided designs. To find where that difference arises, the alternative and null streams of the main comparison were regenerated from their seeds

(they reproduce the design-wager rows of Table S14: 74.9% alternative crossing, 4.1% null crossing) and two fixed-horizon tests were computed on every trial: the one-sided z-test at  $N$ , and a test on the final design-wager wealth  $W_N$ , calibrated to exact level 0.05 on one million independent null draws (critical value  $W_N \geq 1.93$ ). With no ramp,  $W_N$  is a closed-form function of the four cell counts, which was verified against the kernel on every regenerated trial (the kernel initializes wealth at the first patient and bets from the second, so 2,689 patients carry bets).

**Table S20.** Decomposition of the design e-RTb premium (5,000 trials per scenario, the streams of Table S14; Monte Carlo standard errors in parentheses, percentage points).

| Test                                                                | Alternative rejection | Null rejection |
|---------------------------------------------------------------------|-----------------------|----------------|
| Fixed- $N$ one-sided z-test,<br>$\alpha = 0.05$                     | 87.2% (0.5)           | 5.3%           |
| Fixed- $N$ test on final wealth,<br>exact level ( $W_N \geq 1.93$ ) | 87.6% (0.5)           | 5.3%           |
| Anytime crossing, $\sup_i W_i \geq 20$<br>(design e-RTb)            | 74.9% (0.6)           | 4.1%           |
| Final wealth $W_N \geq 20$ , no<br>supremum (reference)             | 61.8% (0.7)           | 0.8%           |

Paired on the same alternative streams, the z-test minus the final-wealth test is -0.4 percentage points (standard error 0.1), and the final-wealth test minus the anytime crossing is 12.7 points (0.5). At a fixed horizon the randomization-model likelihood ratio has the same power as the z-test; it is the Neyman-Pearson statistic of its model at the design alternative. The difference comes from the boundary: a test calibrated for a fixed  $N$  rejects at  $W_N \geq 1.93$ , while e-RT uses the constant boundary 20. Monitoring the supremum recovers about half of what a terminal comparison against 20 would lose (74.9% against 61.8%). This agrees with the result of Koning and van Meer (cited in the main text) that a fixed-horizon test can be made anytime-valid at no cost in power.

Code: deterministic replay of the main-comparison streams (seeds 20260542 and 20260543) through SuppCode\_eRTb.R (Section 1.5); calibration seed 20260554.

## ESM Section 6. Additional details on the completed-trial illustrations

This section carries the order-sensitivity trajectory figures and the full permutation summaries behind the ARMA and FACTT illustrations: 10,000 complete random patient-order permutations per endpoint, at the same wager tuning as the main-text figures. “Observed masked order” is the deidentified BioLINCC record order. Peak  $W$  is the maximum wealth reached along the path. Median effect at crossing and final effect are on the endpoint’s own scale; Type M is the magnification ratio  $|\text{effect at crossing}|/|\text{final effect}|$  among permutations that crossed. Crossing indices are patient indices for e-RTb/e-RTc and event indices for e-RTe.

**Figure S1.** ARMA adaptive e-RT order sensitivity. Low tidal volume ventilation is treated as intervention. Blue line and shaded region show the median and central 95% band across 10,000 complete random patient-order permutations; the black line shows the observed masked BioLINCC order. (A) Mortality e-RTb. (B) Mortality e-RTe. (C) BWA-28 failure e-RTb. (D) BWA-28 e-RTe. Indices are masked dataset or retrospective event indices.

**Table S21.** ARMA adaptive permutation sensitivity (10,000 random patient-order permutations; low tidal volume treated as intervention).

### ARMA adaptive e-RT order sensitivity

Blue line/band: median and central 95% band across 10,000 random patient-order permutations.  
Black line: observed masked BioLINCC order.

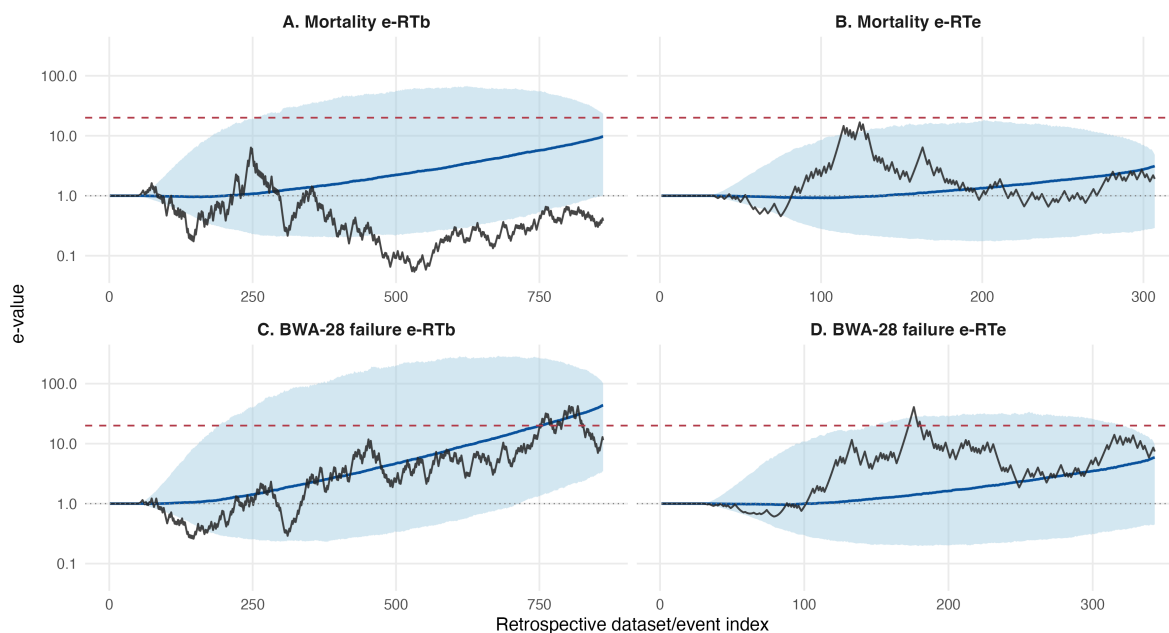


Figure 1: ESM Figure S1

| Curve                | Observed masked order              | Permutation crossings | Median crossing index | Central 95% crossing index | Median effect at crossing | Final effect | Median Type M |
|----------------------|------------------------------------|-----------------------|-----------------------|----------------------------|---------------------------|--------------|---------------|
| Mortality e-RTb      | No crossing; peak $W = 6.37$       | 55.6%                 | 567                   | 163–850                    | +12.0 pp                  | +9.3 pp      | 1.29          |
| Mortality e-RTe      | No crossing; peak $W = 16.77$      | 13.9%                 | 160                   | 76–265                     | +14.0 pp                  | +9.3 pp      | 1.51          |
| BWA-28 failure e-RTb | Crossed at masked index 751        | 94.2%                 | 552                   | 160–839                    | +12.6 pp                  | +11.2 pp     | 1.13          |
| BWA-28 failure e-RTe | Crossed at failure-event index 173 | 29.2%                 | 205                   | 80–323                     | +14.4 pp                  | +11.2 pp     | 1.28          |

**Figure S2.** FACTT adaptive e-RT order sensitivity. Blue line and shaded region show the median and central 95% band across 10,000 complete random patient-order permutations; the black line shows the observed masked BioLINCC order. (A) Conservative versus liberal fluid strategy for ventilator-free days through day 28, monitored with e-RTc. (B, C) Fluid comparison mortality with e-RTb and e-RTe. (D) Pulmonary-artery catheter versus central venous catheter for ventilator-free days. (E, F) Catheter comparison mortality with e-RTb and e-RTe.

**Table S22.** FACTT adaptive permutation sensitivity (10,000 random patient-order permutations).

| Curve                          | Observed<br>masked<br>order           | Permutation<br>crossings | Median<br>crossing<br>index | Central<br>95%<br>crossing<br>index | Median<br>effect at<br>crossing | Final<br>effect              | Median<br>Type M |
|--------------------------------|---------------------------------------|--------------------------|-----------------------------|-------------------------------------|---------------------------------|------------------------------|------------------|
| Catheter<br>VFD-28<br>e-RTc    | No<br>crossing;<br>peak $W =$<br>2.42 | 2.7%                     | 242                         | 120–574                             | $d = -0.34$<br>(–3.4<br>VFD)    | $d = -0.06$<br>(–0.6<br>VFD) | 5.81             |
| Catheter<br>mortality<br>e-RTb | No<br>crossing;<br>peak $W =$<br>1.04 | 0.3%                     | 265                         | 160–429                             | –12.8 pp                        | –1.2 pp                      | 13.64            |
| Catheter<br>mortality<br>e-RTe | No<br>crossing;<br>peak $W =$<br>1.17 | 0.2%                     | 100                         | 74–164                              | –10.3 pp                        | –1.2 pp                      | 8.54             |

The rare-crossing rows carry the magnification caution in numeric form: where permutation crossings are rare (0.2–2.7%), the median effect at crossing overstates the final effect by factors of 4–14; where crossings are common (94–99%), the overstatement is mild (Type M 1.1–1.3).

### FACTT adaptive e-RT order sensitivity

Blue line/band: median and central 95% band across 10,000 random patient-order permutations.  
Black line: observed masked BioLINCC order.

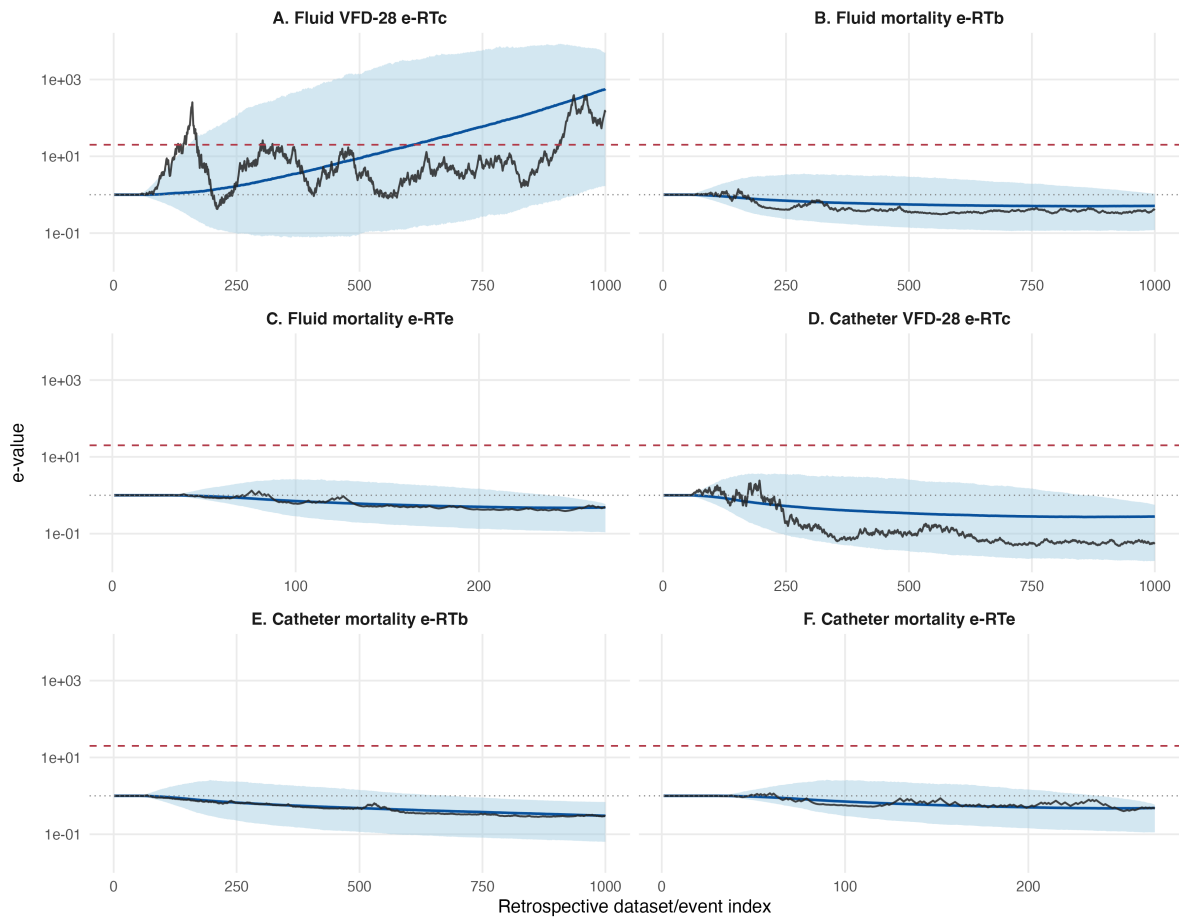


Figure 2: ESM Figure S2