Designing Correlation Structures for Clinical Trial Simulations: Ensuring Positive Definite Covariance Matrices

Technical Guide

November 12, 2025

Abstract

This document provides comprehensive guidance on designing correlation structures for multivariate normal simulations in clinical trials. We cover theoretical foundations, practical constraints, methods for ensuring positive definiteness, and specific recommendations for N-of-1 trial designs with multiple timepoints and response components.

Contents

1 The Problem

1.1 Context

In clinical trial simulations (particularly N-of-1 designs), we need to generate correlated multivariate normal data:

$$\mathbf{Y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{1}$$

where $\mathbf{Y}$ includes:

- Baseline variables: biomarker, baseline symptom level
- Time-varying components at T timepoints:
  - $\text{tv}_1, \ldots, \text{tv}_T$ (time-variant factor)
  - $\text{pb}_1, \ldots, \text{pb}_T$ (pharmacologic/expectancy factor)
  - $\text{br}_1, \ldots, \text{br}_T$ (biologic response factor)

For $T = 20$ timepoints, this creates a $(2+3T) = 62$ dimensional multivariate normal distribution.

1.2 The Challenge

Not all correlation structures produce valid (positive definite) covariance matrices. An invalid structure will cause:

- `mvrnorm()` to fail
- Negative eigenvalues in $\boldsymbol{\Sigma}$
- Need for correction via `make.positive.definite()`, which distorts the intended correlation structure

2 Theoretical Foundation

2.1 Positive Definiteness

Definition 1 (Positive Definite Matrix)**.** A symmetric matrix $\mathbf{A}$ is positive definite (PD) if for all non-zero vectors $\mathbf{x}$:

$$\mathbf{x}^T \mathbf{A} \mathbf{x} > 0 \tag{2}$$

Equivalently, all eigenvalues of $\mathbf{A}$ are strictly positive.

Definition 2 (Valid Correlation Matrix)**.** A correlation matrix $\mathbf{R}$ is valid if:

1. $\mathbf{R}$ is symmetric: $R_{ij} = R_{ji}$
2. Diagonal elements are 1: $R_{ii} = 1$
3. Off-diagonal elements satisfy: $-1 \leq R_{ij} \leq 1$
4. $\mathbf{R}$ is positive definite

Note: Conditions 1-3 are necessary but **NOT sufficient** for positive definiteness.

2.2 Why Arbitrary Correlations Fail

Consider a simple 3-variable case:

$$\mathbf{R} = \begin{pmatrix} 1 & 0.9 & 0.9 \\ 0.9 & 1 & 0.1 \\ 0.9 & 0.1 & 1 \end{pmatrix} \tag{3}$$

This matrix is **not positive definite** because:

- X_1 and X_2 are highly correlated (0.9)
- X_1 and X_3 are highly correlated (0.9)
- But X_2 and X_3 are weakly correlated (0.1)
- This violates transitivity: if $X_1 \approx X_2$ and $X_1 \approx X_3$, then we must have $X_2 \approx X_3$

Eigenvalues: $\{2.42, 0.88, -0.30\} \leftarrow$ negative!

2.3 Sufficient Conditions for Positive Definiteness

Theorem 1 (Sufficient Conditions)**.** A correlation matrix $\mathbf{R}$ is positive definite if:

1. It can be written as $\mathbf{R} = \mathbf{L}\mathbf{L}^T$ for some matrix $\mathbf{L}$ (Cholesky decomposition exists)
2. It arises from an actual data-generating process (e.g., factor model)
3. All principal minors are positive (Sylvester's criterion)

Practical implication: Build correlation structures from generative models, not arbitrary values.

3 Current Correlation Structure

3.1 Your Implementation

Your simulation uses 5 correlation parameters:

Parameter	Meaning	Current Value
c_{tv}	Autocorr: time_variant across time	0.2–0.5
c_{pb}	Autocorr: pharm_biomarker across time	0.2–0.5
c_{br}	Autocorr: bio_response across time	0.2–0.5
c_{cf1t}	Cross-corr: factors at *same* time	0.05–0.1
c_{cfct}	Cross-corr: factors at *different* times	0.05–0.4
c_{bm}	Biomarker $\leftrightarrow$ bio_response	0.2–0.6

Table 1: Correlation parameters in your simulation

3.2 Structure Visualization

For 3 factors at 3 timepoints, the correlation matrix has this block structure:

$$\mathbf{R} = \begin{pmatrix} \mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\text{tv}} & \mathbf{C}_{12} & \mathbf{C}_{13} \\ \mathbf{0} & \mathbf{C}_{21} & \mathbf{R}_{\text{pb}} & \mathbf{C}_{23} \\ \mathbf{0} & \mathbf{C}_{31} & \mathbf{C}_{32} & \mathbf{R}_{\text{br}} \end{pmatrix} \tag{4}$$

where:

- $\mathbf{I}_2$: Biomarker and baseline (uncorrelated by construction)
- $\mathbf{R}_{\text{tv}}$: Time-variant autocorrelations (all c_{tv})
- $\mathbf{C}_{ij}$: Cross-factor correlations (c_{cf1t} on diagonal, c_{cfct} off-diagonal)

4 Why Positive Definiteness Fails

4.1 Common Failure Modes

4.1.1 High Autocorrelations with Low Cross-Correlations

Problem:

- Setting $c_{\text{tv}} = c_{\text{pb}} = c_{\text{br}} = 0.8$ (high within-factor correlation)
- But $c_{\text{cf1t}} = 0.1$, $c_{\text{cfct}} = 0.05$ (low cross-factor correlation)

Why it fails: With 20 timepoints, each factor creates a near-singular 20×20 block. The three blocks are weakly coupled, creating near-linear dependencies.

4.1.2 Cross-Time Exceeds Same-Time Correlations

Problem:

- $c_{\text{cfct}} = 0.4$ (different times)
- $c_{\text{cf1t}} = 0.1$ (same time)

Why it fails: Violates temporal coherence – factors at time t should be more correlated with each other than with factors at time $t + k$.

4.1.3 Dimension Curse

As T increases, constraints become tighter:

- $T = 5$ timepoints: 17-dim matrix, relatively forgiving
- $T = 10$ timepoints: 32-dim matrix, moderate constraints
- $T = 20$ timepoints: 62-dim matrix, **very strict constraints**

4.2 Hendrickson's Values (That Work)

Hendrickson et al. used:

- $c_{\text{tv}} = 0.8$
- $c_{\text{pb}} = 0.8$
- $c_{\text{br}} = 0.8$
- $c_{\text{cf1t}} = 0.2$
- $c_{\text{cfct}} = 0.1$

Why this works:

1. High autocorrelations (0.8) are balanced
2. Same-time cross-correlations (0.2) are moderate
3. Different-time cross-correlations (0.1) < same-time (0.2) ✓
4. Ratio: $c_{\text{cf1t}}/c_{\text{autocorr}} = 0.2/0.8 = 0.25$

Your values (that fail):

- $c_{\text{tv}} = 0.2$ (too low)
- $c_{\text{cf1t}} = 0.1$
- $c_{\text{cfct}} = 0.4$ (too high, exceeds same-time!)

5 Guidelines for Choosing Correlations

5.1 Empirical Rules

Guideline 1 (Autocorrelation Magnitude)**.** For stability with $T \geq 10$ timepoints:

- **Conservative:** $0.3 \leq c_{\text{autocorr}} \leq 0.7$
- **Moderate:** $0.5 \leq c_{\text{autocorr}} \leq 0.8$
- **Aggressive:** $0.7 \leq c_{\text{autocorr}} \leq 0.9$ (risky for large T)

Very low autocorrelations (< 0.3) make the matrix ill-conditioned.

Guideline 2 (Cross-Correlation Hierarchy)**.** Maintain the temporal coherence constraint:

$$c_{\text{cfct}} < c_{\text{cf1t}} < c_{\text{autocorr}} \tag{5}$$

Recommended ratios:

$$c_{\text{cf1t}} \approx 0.2 \text{ to } 0.4 \times c_{\text{autocorr}} \tag{6}$$

$$c_{\text{cfct}} \approx 0.5 \text{ to } 0.7 \times c_{\text{cf1t}} \tag{7}$$

Guideline 3 (Biomarker Correlation)**.**

- $c_{\text{bm}} = 0$ (no interaction): Always works
- $c_{\text{bm}} \in [0.2, 0.4]$: Safe for most structures
- $c_{\text{bm}} \in [0.5, 0.7]$: Requires careful tuning of other parameters
- $c_{\text{bm}} > 0.7$: Often causes issues with high-dimensional matrices

5.2 Recommended Parameter Sets

5.2.1 Conservative (Always Works)

```r
# For T = 20 timepoints
base_correlations <- list(
  base_autocorr = 0.5,      # Moderate within-factor
  base_cross_same = 0.15,   # 0.3 * autocorr
  base_cross_diff = 0.08    # 0.5 * cross_same
)
# No dynamic adjustment based on carryover
```

Properties:

- ✓ Guaranteed PD for $T \leq 30$
- ✓ Interpretable as "moderate temporal persistence"
- ✓ Stable across parameter variations

5.2.2 Hendrickson-Style (Validated)

```r
# Matching published simulation
model_params <- list(
  c.tv = 0.8,
  c.pb = 0.8,
  c.br = 0.8,
  c.cf1t = 0.2,
  c.cfct = 0.1,
  c.bm = 0.3  # or 0, 0.3, 0.6 for parameter sweep
)
```

Properties:

- ✓ Empirically validated (Hendrickson et al., 2020)
- ✓ Represents strong individual-level stability
- ✓ Works for $T = 8$ (hybrid design)

5.2.3 Intermediate (Balanced)

```r
# Good balance of stability and flexibility
base_correlations <- list(
  base_autocorr = 0.6,      # Moderate-high within-factor
  base_cross_same = 0.18,   # 0.3 * autocorr
  base_cross_diff = 0.09    # 0.5 * cross_same
)
```

Properties:

- ✓ Works for $T \leq 20$
- ✓ Allows moderate parameter variation
- ✓ Represents realistic temporal dependencies

6 The Carryover Adjustment Issue

6.1 Current Implementation

Your code dynamically adjusts correlations based on carryover:

```
calculate_carryover_adjusted_correlations <- function(
    base_correlations, carryover_halflife) {

  carryover_strength <- carryover_halflife / (1 + carryover_halflife)

  # INCREASES autocorrelations
  autocorr_boost <- 0.3 * carryover_strength
  adjusted_br <- min(0.95, base_autocorr + autocorr_boost * 1.2)

  # INCREASES cross-time correlations
  cross_time_boost <- 0.4 * carryover_strength
  adjusted_cross_diff <- min(0.8, base_cross_diff + cross_time_boost)

  # DECREASES same-time correlations
  cross_same_reduction <- 0.1 * carryover_strength
  adjusted_cross_same <- max(0.05, base_cross_same - cross_same_reduction)
}
```

6.2 Problems with This Approach

6.2.1 Violation of Hierarchy

With carryover_halflife = 1.5:

$$\text{carryover\_strength} = 1.5/2.5 = 0.6 \tag{8}$$

$$\text{cross\_time\_boost} = 0.4 \times 0.6 = 0.24 \tag{9}$$

$$\text{adjusted\_cross\_diff} = 0.05 + 0.24 = 0.29 \tag{10}$$

But:

$$\text{adjusted\_cross\_same} = 0.1 - 0.06 = 0.04 \tag{11}$$

Result: $c_{\text{cfct}} = 0.29 > c_{\text{cf1t}} = 0.04$ VIOLATES HIERARCHY!
This creates non-PD matrices.

6.2.2 Starting from Low Base Values

Starting with:

- base_autocorr = 0.2 (too low)
- Boosting to $0.2 + 0.18 = 0.38$ (still relatively low)

The matrix remains ill-conditioned.

6.3 Recommendation

Remove dynamic carryover adjustment for two reasons:

1. **Theoretical:** Carryover should affect means, not correlations (see previous document)
2. **Practical:** Dynamic adjustment easily violates PD constraints

7 Practical Implementation Strategy

7.1 Step 1: Choose Base Values (Following Guidelines)

```r
# Start with Hendrickson-validated values
model_params <- list(
  c.tv = 0.7,       # Moderate-high (slightly lower than Hend)
  c.pb = 0.7,
  c.br = 0.7,
  c.cf1t = 0.20,    # 0.29 * autocorr
  c.cfct = 0.10,    # 0.50 * cross_same
  c.bm = 0.3        # Moderate biomarker interaction
)
```

7.2 Step 2: Remove Dynamic Adjustment

```r
# OLD (problematic):
adjusted_correlations <- calculate_carryover_adjusted_correlations(
  base_correlations, params$carryover_t1half
)

# NEW (stable):
# Use fixed correlations for all carryover levels
model_params <- list(
  c.tv = 0.7,
  c.pb = 0.7,
  c.br = 0.7,
  c.cf1t = 0.20,
  c.cfct = 0.10,
  c.bm = params$biomarker_correlation  # Vary this in parameter sweep
)
```

7.3 Step 3: Validate Before Simulation

Create a validation function:

```r
validate_correlation_structure <- function(model_params,
                                           resp_param,
                                           baseline_param,
                                           trial_design) {
  # Build sigma matrix
  sigma_result <- build_sigma_matrix(
    model_params, resp_param, baseline_param,
```

```
    trial_design,
    factor_types = c("time_variant", "pharm_biomarker",
                     "bio_response"),
    factor_abbreviations = c("tv", "pb", "br"),
    verbose = TRUE
  )

  if (is.null(sigma_result)) {
    cat("FAILED: Non-positive definite\n")
    return(FALSE)
  }

  # Check condition number
  sigma <- sigma_result$sigma
  eigenvalues <- eigen(sigma, only.values = TRUE)$values
  condition_number <- max(eigenvalues) / min(eigenvalues)

  cat("Eigenvalue range: [", min(eigenvalues), ", ",
      max(eigenvalues), "]\n")
  cat("Condition number: ", condition_number, "\n")

  # Well-conditioned if condition number < 100
  if (condition_number > 100) {
    warning("Matrix is ill-conditioned (condition number = ",
            condition_number, ")")
  }

  return(TRUE)
}
```

7.4 Step 4: Parameter Sweep Strategy

When varying parameters, validate each combination:

```
# Define parameter grid
param_grid <- expand_grid(
  n_participants = c(35, 70),
  biomarker_correlation = c(0, 0.2, 0.4, 0.6),
  carryover_t1half = c(0, 0.5, 1.0, 2.0)
)

# Validate each combination BEFORE running simulations
valid_combinations <- tibble()
for (i in 1:nrow(param_grid)) {
  current_params <- param_grid[i,]

  # Build model params with FIXED correlations
  model_params$c.bm <- current_params$biomarker_correlation
  # carryover_t1half affects MEANS only, not correlations

  is_valid <- validate_correlation_structure(
    model_params, resp_param, baseline_param, trial_design
  )

```

```
  if (is_valid) {
    valid_combinations <- bind_rows(valid_combinations,
                                    current_params)
  }
}

# Run simulations ONLY on valid combinations
```

8 Troubleshooting

8.1 If Matrix is Still Non-PD

Algorithm 1 Diagnosing Non-PD Matrices

Compute eigenvalues of $\mathbf{\Sigma}$
if smallest eigenvalue < 0 but > -0.01 **then**
    ▷ Numerical precision issue
    Use `make.positive.definite()` with small tolerance
else if smallest eigenvalue < -0.01 **then**
    ▷ Structural problem
    Check:
        1. Is $c_{\text{cfct}} < c_{\text{cf1t}}$?
        2. Are autocorrelations > 0.3?
        3. Is $c_{\text{bm}} < 0.7$?
        4. Reduce dimensionality (fewer timepoints)?
end if

8.2 Quick Fixes

8.2.1 Fix 1: Reduce Cross-Time Correlation

```
# If failing with c.cfct = 0.4
model_params$c.cfct <- 0.05  # Much more conservative
```

8.2.2 Fix 2: Increase Autocorrelations

```
# If failing with c.tv = 0.2
model_params$c.tv <- 0.6    # Increase to moderate-high
model_params$c.pb <- 0.6
model_params$c.br <- 0.6
```

8.2.3 Fix 3: Use Compound Symmetry

For a guaranteed-PD structure (less realistic):

```
# All within-factor correlations equal
model_params$c.tv <- 0.5
model_params$c.pb <- 0.5
```

```r
model_params$c.br <- 0.5
# All between-factor correlations equal
model_params$c.cf1t <- 0.15
model_params$c.cfct <- 0.15  # Same as same-time
```

This creates a compound symmetry structure, which is always PD if $\rho < 1$.

9 Recommended Final Configuration

9.1 For Your Current Simulation

```r
# Remove dynamic carryover adjustment entirely
# Use fixed, validated correlation structure

# Base correlation parameters (FIXED across all carryover levels)
model_params <- list(
  N = n_participants,

  # Autocorrelations (moderate-high for stability)
  c.tv = 0.65,
  c.pb = 0.65,
  c.br = 0.65,

  # Cross-correlations (maintain hierarchy)
  c.cf1t = 0.18,      # 0.28 * autocorr
  c.cfct = 0.09,      # 0.50 * cf1t

  # Biomarker interaction (vary in parameter sweep)
  c.bm = biomarker_correlation,  # 0, 0.2, 0.4, 0.6

  # Carryover affects MEANS only
  carryover_t1half = carryover_t1half
)

# Standard deviations
resp_param <- tibble(
  cat = c("time_variant", "pharm_biomarker", "bio_response"),
  max = c(1.0, 1.0, treatment_effect),
  disp = c(2.0, 2.0, 2.0),
  rate = c(0.3, 0.3, 0.3),
  sd = c(within_subject_sd, within_subject_sd, within_subject_sd)
)

baseline_param <- tibble(
  cat = c("biomarker", "baseline"),
  m = c(5.0, 10.0),
  sd = c(2.0, between_subject_sd)
)
```

9.2 Expected Properties

This configuration:

- ✓ Guaranteed PD for $T \leq 20$ with $c_{\text{bm}} \leq 0.6$
- ✓ Maintains proper correlation hierarchy
- ✓ Interpretable: moderate individual-level stability
- ✓ Aligns with standard statistical practice (carryover in means only)
- ✓ Comparable to Hendrickson's validated approach

10 Conclusion

10.1 Key Takeaways

1. **Start with validated values** (Hendrickson's or conservative guidelines)
2. **Maintain correlation hierarchy**: $c_{\text{cfct}} < c_{\text{cf1t}} < c_{\text{autocorr}}$
3. **Avoid dynamic adjustment** based on carryover (theoretical and practical issues)
4. **Use moderate autocorrelations** (0.5 to 0.7) for stability
5. **Pre-validate** all parameter combinations before running expensive simulations
6. **Monitor condition numbers** to detect ill-conditioning early

10.2 Implementation Checklist

- ☐ Remove `calculate_carryover_adjusted_correlations()` function
- ☐ Set fixed correlation parameters following guidelines
- ☐ Implement `validate_correlation_structure()` function
- ☐ Run validation on all parameter grid combinations
- ☐ Cache validated sigma matrices
- ☐ Document final correlation structure in methods