Design declaration, diagnosis, and redesign | *Déclaration, diagnostic et redesign*

MIDA — introduction | *MIDA — introduction*

Macartan / Alyssa

2026-06-12

Learning goals
Objectifs d'apprentissage

1. Understand that if you can state the MIDA elements of a design in code, you can assess the quality of a design
2. Learn how design adjustments can be optimized
3. Get exposure to the kind of code that lets you do this in practice

1. Comprendre que si vous pouvez déclarer les éléments MIDA d'un design en code, vous pouvez évaluer la qualité d'un design
2. Apprendre à optimiser les ajustements de design
3. Découvrir le type de code qui permet de le faire en pratique

Declaration
Déclaration

Declaration tells the computer (and readers) what **M**, **I**, **D**, and **A** are.

La **déclaration** indique à l'ordinateur (et aux lecteurs) ce que sont **M**, **I**, **D** et **A**.

Diagnosis
Diagnostic

Diagnosis asks how the design will perform under imagined conditions.
We estimate **diagnosands**: power, bias, RMSE, error rates, ethical harm, amount learned, ...

Le **diagnostic** examine comment le design se comportera dans des conditions imaginées.
On estime des **indicateurs de diagnostic** : puissance statistique, biais, erreur quadratique moyenne, taux d'erreur, risques éthiques, quantité apprise, ...

Redesign adjusts data- and answer-strategy features to see how diagnosands change:

- sample size
- randomization procedure
- estimation strategy
- implementation trade-offs (e.g. compliance vs. sample size)

L'étape **redesign** (re-concevoir) adapte la stratégie de données et d'analyse afin d'observer comment les indicateurs de diagnostic évoluent. On peut modifier :

- la taille de l'échantillon
- la procédure d'assignation aléatoire (randomisation)
- la stratégie d'analyse
- les compromis liés à la mise en œuvre (p.ex. respect de la conformité vs. la taille d'échantillon)

Section 1

Declaration

Simplest design

design la plus simple

What is the simplest **diagnosable** design?

Quelle est la conception de recherche **diagnostiquable** la plus simple ?

```
simplest_design <-
  declare_model(N = 100,
                Y0 = rnorm(N),
                Y1 = Y0 + .2) +
  declare_inquiry(Q = mean(Y1- Y0)) +
  declare_assignment(X = simple_ra(N)) +
  declare_measurement(Y = X*Y1 + (1-X)*Y0) +
  declare_estimator(Y ~ X)
```

What it does

Ce que fait le code

- M: Draw 100 units with potential outcomes
- I: Define an inquiry: the **sample average effect**
- D: Assign a treatment and measure an outcome
- A: Estimate the mean with a regression intercept

- M : Tirer 100 unités avec leurs résultats potentiels
- I : Définir une interrogation (une question de recherche) : l'**effet moyen dans l'échantillon**
- D : Assigner le traitement et mesurer le résultat
- A : Estimer la moyenne avec l'ordonnée à l'origine d'une régression

Run once

Exécuter le code une fois

Run the whole design once — type its name or call `run_design()`. This produces data, **estimands**, **estimates**, and ancillary statistics.

Exécutez le code de la conception de recherche dans son intégralité une fois — tapez son nom ou appelez `run_design()`. Cela génère des données, des **estimandes**, des **estimations** et des statistiques auxiliaires.

```
simplest_design |> run_design()
```

inquiry	estimand	estimator	term	estimate	std.error	statistic
Q	0.2	estimator	X	0.1	0.22	0.45

Repeat the design many times with `simulate_design()`.

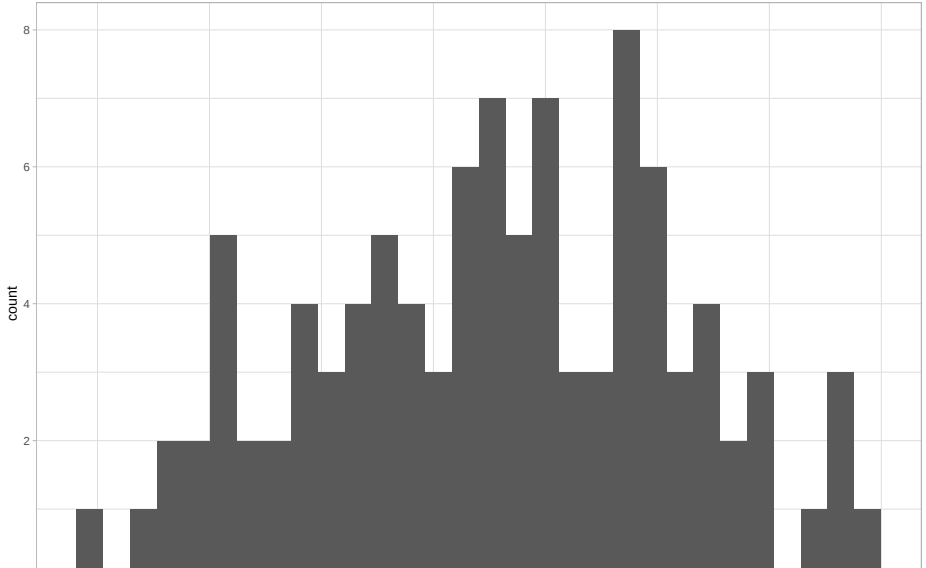
Répétez le design **plusieurs fois** avec `simulate_design()`.

```
some_runs |> head()
```

design	sim_ID	inquiry	estimand	estimator	term	estimate
simplest_design	1	Q	0.2	estimator	X	0.61
simplest_design	2	Q	0.2	estimator	X	0.18
simplest_design	3	Q	0.2	estimator	X	0.25
simplest_design	4	Q	0.2	estimator	X	0.42
simplest_design	5	Q	0.2	estimator	X	0.17
simplest_design	6	Q	0.2	estimator	X	0.14

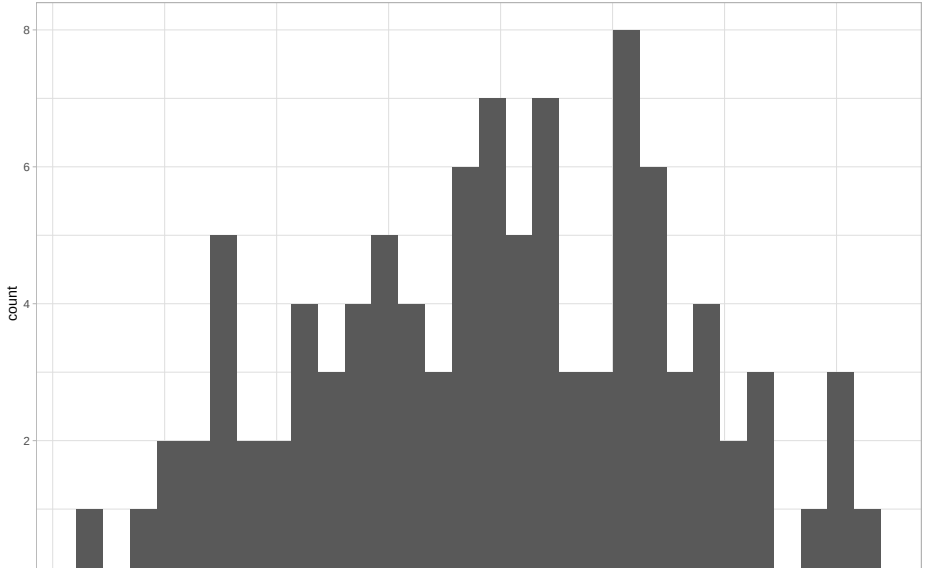
Distribution of estimates

Distribution des estimations



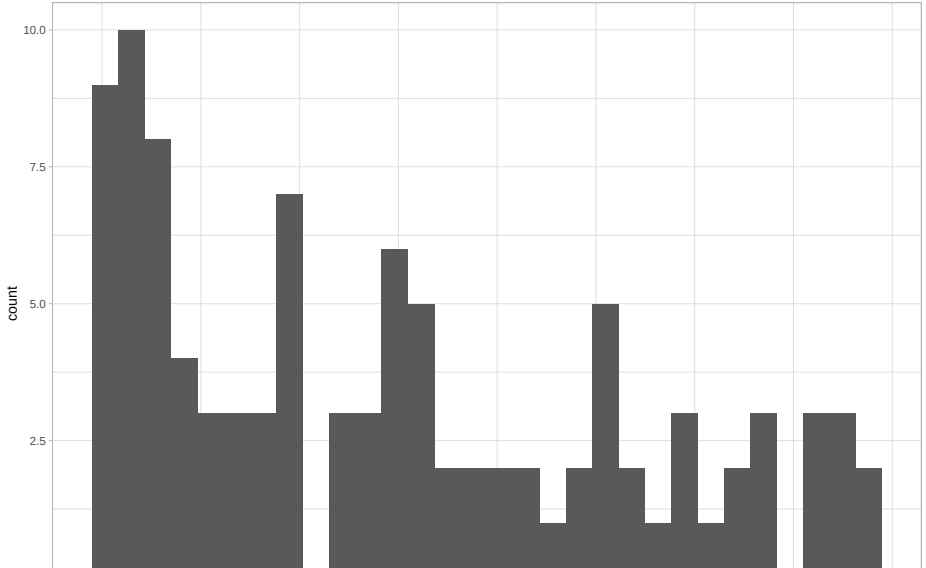
Distribution of errors

Distribution des erreurs



Distribution of p values

Distribution des p-valeurs



Section 2

Diagnosis

Bias by hand

Biais à la main

After many simulations, ask about **bias**: the average gap between estimand and estimate.

Après de nombreuses simulations, mesurez le **biais** : la différence moyenne entre l'estimande et l'estimation.

```
some_runs |>
  mutate(error = estimate - estimand) |>
  summarize(
    mean_estimate = mean(estimate),
    mean_estimand = mean(estimand),
    bias = mean(error)
  )
```

mean_estimate	mean_estimand	bias
0.19	0.2	-0.01

`diagnose_design()` computes common diagnosands in one step.

`diagnose_design()` calcule les indicateurs de diagnostic en une seule étape.

```
diagnosis <- simplest_design |> diagnose_design()
```

Design	N Sims	Mean Estimand	Mean Estimate	Bias
simplest_design	1000	0.20 (0.00)	0.20 (0.01)	0.00 (0.01)

Section 3

Redesign

What is redesign?

Qu'est-ce que le redesign ?

Make a new design by taking an existing design and change a **parameter**

```
design_2 <- design |> redesign(N = 300)
```

Créez un nouveau design en partant d'un design existant et en modifiant une **variable**

```
design_2 <- design |> redesign(N = 300)
```

Design parameters

Variables de design

- To do this your design needs *parameters*
- A **parameter** is a quantity in your environment that the design references explicitly — e.g. `N` instead of a fixed number.
- You create it in your environment and then reference it when you make your design

- Pour cela, votre design a besoin de *variables*
- Une **variable** est une quantité dans votre environnement que le design cite explicitement — p.ex. `N` au lieu d'un nombre fixé.
- Vous la créez dans votre environnement, puis vous y faites référence lorsque vous construisez votre design.

```
N <- 100

simplest_design <-
  declare_model(N = N, Y0 = rnorm(N), Y1 = 1 + rnorm(N)) +
  declare_inquiry(Q = mean(Y1- Y0)) +
  declare_assignment(X = simple_ra(N)) +
  declare_measurement(Y = X*Y1 + (1-X)*Y0) +
  declare_estimator(Y ~ X)
```

Simple redesign

Redesign simple

`redesign()` returns a modified design — here with `N = 200`.

`redesign()` renvoie un design modifié — ici avec `N = 200`.

```
design_200 <- simplest_design |> redesign(N = 200)
design_200 |> draw_data() |> nrow()
```

```
[1] 200
```

A list of designs

Une liste de designs

- Pass a **vector** of parameter values to get several designs at once.
- With two parameters, the function `redesign()` builds a grid of designs — easy to diagnose and compare.

- Passez un **vecteur** de valeurs de variables pour obtenir plusieurs designs à la fois.
- Avec deux variables, la fonction `redesign()` construit un tableau de designs — facile à diagnostiquer et à comparer.

```
N <- 100
b <- .2

simplest_design <-
  declare_model(N = N, Y0 = rnorm(N), Y1 = b + rnorm(N)) +
  declare_inquiry(Q = mean(Y1- Y0)) +
  declare_assignment(X = simple_ra(N)) +
  declare_measurement(Y = X*Y1 + (1-X)*Y0) +
  declare_estimator(Y ~ X)
```

```
designs <- redesign(simplest_design, N = c(100, 200, 500), b = c(0, .2, .5))
designs |> diagnose_design() |> tidy()
```

Compare diagnoses

Comparer les diagnostics

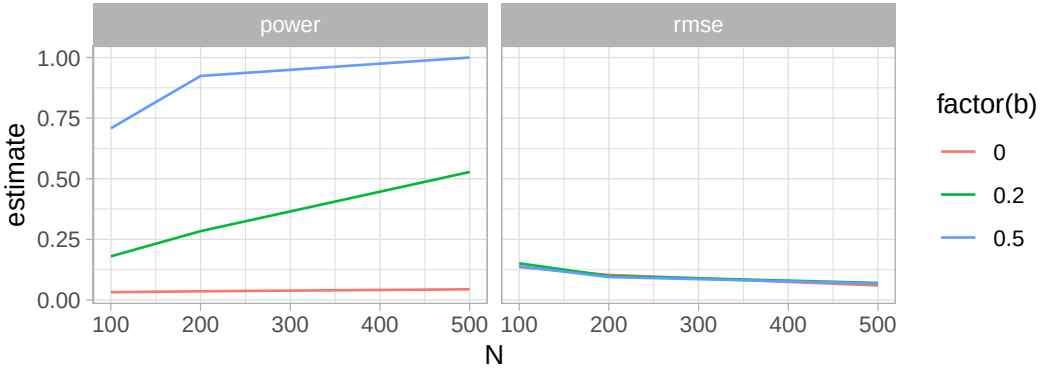
N	b	diagnosand	estimate	std.error	conf.low	conf.high
100	0.0	rmse	0.14	0.01	0.12	0.15
100	0.0	power	0.03	0.01	0.01	0.05
200	0.0	rmse	0.10	0.00	0.10	0.11
200	0.0	power	0.04	0.01	0.02	0.06
500	0.0	rmse	0.06	0.00	0.06	0.06
500	0.0	power	0.04	0.01	0.02	0.07
100	0.2	rmse	0.15	0.01	0.13	0.17
100	0.2	power	0.18	0.03	0.13	0.24
200	0.2	rmse	0.10	0.00	0.09	0.11
200	0.2	power	0.28	0.03	0.23	0.34
500	0.2	rmse	0.07	0.00	0.06	0.08

Plot after redesign

Graphique après redesign

After `tidy()`, ggplot makes comparison across redesigns straightforward.

Après `tidy()`, ggplot permet de comparer facilement les redesigns.



Power depends on N and effect size; RMSE depends on N only.

La puissance statistique dépend du N et de la taille de l'effet ; L'erreur quadratique moyenne ne dépend que de N.

Section 4

Resources

declare_* commands

*Commandes declare_*

Key steps for **building** a design:

- `declare_model()`
- `declare_inquiry()`
- `declare_sampling()`
- `declare_assignment()`
- `declare_measurement()`
- `declare_estimator()`
- … and more `declare_*` functions

Étapes clés pour **construire** un design :

- `declare_model()`
- `declare_inquiry()`
- `declare_sampling()`
- `declare_assignment()`
- `declare_measurement()`
- `declare_estimator()`
- … et d'autres fonctions `declare_*`

Using a design
Utiliser un design

Key commands for **running** and **learning** from a design:

- `draw_data(design)`, `draw_estimands(design)`, `draw_estimates(design)`
- `get_estimates(design, data)`
- `run_design(design)`, `simulate_design(design)`
- `diagnose_design(design)`
- `redesign(design, N = 200)`
- `compare_designs()`, `compare_diagnoses()`

Commandes clés pour **exécuter** un design et **en tirer des conclusions** :

- `draw_data(design)`, `draw_estimands(design)`, `draw_estimates(design)`
- `get_estimates(design, data)`
- `run_design(design)`, `simulate_design(design)`
- `diagnose_design(design)`
- `redesign(design, N = 200)`
- `compare_designs()`, `compare_diagnoses()`

Other resources

Autres ressources

- Full slide deck: macartan.github.io/dd_bootcamp
- Website: declaredesign.org
- Book: book.declaredesign.org
- Console help: `?DeclareDesign`

- Diapositives complètes : macartan.github.io/dd_bootcamp
- Site web : declaredesign.org
- Livre : book.declaredesign.org
- Aide console : `?DeclareDesign`