Supplementary Material

**Title**: Enhancing Cardiovascular Monitoring: A Non-Linear Model for Characterizing RR Interval Fluctuations in Exercise and Recovery.

**Authors**: Matías Castillo-Aguilar1, Diego Mabe-Castro1,2, David Medina4,5, Cristian Núñez-Espinosa1,3\*.

1 Centro Asistencial Docente e Investigación (CADI-UMAG), Universidad de Magallanes, Punta Arenas, Chile.

2 Departamento de Kinesiología, Universidad de Magallanes, Punta Arenas, Chile.

3 Escuela de Medicina, Universidad de Magallanes, Punta Arenas, Chile.

4 Departamento de Ingeniería en Computación, Universidad de Magallanes, Punta Arenas, Chile.

5 Centre for Biotechnology and Bioengineering, CeBiB, Universidad de Chile, Santiago, Chile.

### \*Correspondence:

Cristian Núñez-Espinosa, Escuela de Medicina, Universidad de Magallanes, Punta Arenas, Chile. Centro Asistencial de Docencia e Investigación CADI-UMAG, Chile. e-mail: [cristian.nunez@umag.cl](mailto:cristian.nunez@umag.cl). Address: Avenida Bulnes 01855, Box 113-D. Phone: +56 61 2201411.

# Methods

## RRi Data Processing

This section describes the detailed methodology used to process the RR interval (RRi) data, highlighting the decisions made and the rationale behind each step. The processing pipeline was implemented in **R** using key libraries, including data.table and signal, and was designed to ensure the integrity and standardization of RRi data for subsequent modeling.

### Data Import and Initial Filtering

RRi data were collected from .txt files stored in the data-raw/hrv\_data directory. Each file contained a series of RR intervals, measured in milliseconds. To ensure data quality, only RRi values within a physiologically plausible range (400–1200 ms) were retained. This range was selected based on established thresholds for human heart rates during moderate exercise and recovery phases. RR intervals outside this range were likely artifacts or measurement errors and were excluded to prevent skewing the analysis.

The cumulative sum of valid RRi values was calculated to determine the time elapsed (in minutes) for each recording. Data exceeding 20 minutes of cumulative time were excluded to standardize the evaluation period, focusing on the typical duration of rest-exercise-recovery protocols. This step ensured that subsequent analyses could directly compare individuals without temporal biases.

The filtered RRi data were consolidated into a single data table, with additional metadata columns, including the file identifier and the corresponding recording ID.

rri\_data <- lapply(files, \(i) {   
 rr\_interval <- as.double(x = readLines(i))  
 not\_na <- !is.na(rr\_interval)  
 rr\_interval <- rr\_interval[not\_na & between(rr\_interval, 400, 1200)]  
 time <- cumsum(rr\_interval)/60000  
 data.table(rr\_interval = rr\_interval[time <= 20],  
 time = time[time <= 20],  
 file = gsub("data-raw/", "", i, fixed = TRUE))  
}) |> rbindlist(idcol = "id")

### Exclusion of Incomplete Data

Incomplete recordings, defined as those with less than 10 minutes of cumulative time, were excluded. This threshold was chosen to maintain sufficient data length for robust analysis of the rest-exercise-recovery phases. Files corresponding to incomplete recordings were moved to a separate directory (data-raw/hrv\_data/unused) for archival purposes. This step ensured that only high-quality data were used for subsequent processing, minimizing the impact of noise and artifacts.

id\_to\_keep <- rri\_data[, max(time) > 10, id][V1 == TRUE, id]  
rri\_data <- rri\_data[id %in% id\_to\_keep]

### Signal Denoising and Filtering

To mitigate noise in the RRi signals, a 10 Hz low-pass Butterworth filter was applied. This filter effectively smooths the data while preserving the key features of the RRi dynamics1,2, such as the onset of exercise-induced changes and post-exercise recovery. A zero-phase filtering technique (filtfilt) was used to prevent phase distortion, which is critical for preserving the temporal structure of the RRi signal1,2.

denoise <- function(x, w = 0.1, n = 3) {  
 signal::filtfilt(filt = signal::butter(n, w), x)  
}  
rri\_data[, rr\_denoised := denoise(rr\_interval), id]

Edge artifacts introduced by the filtering process were mitigated by trimming the first and last five data points of each signal. This ensures that only the central, unperturbed portions of the signal are retained for analysis.

trim <- function(x, abs) {  
 x\_len <- length(x)  
 ind <- c(1:abs, (x\_len - abs):x\_len)  
 x[ind] <- NA  
 x  
}  
rri\_data[, rr\_denoised := trim(rr\_denoised, abs = 5), id]

### Visual Inspection for Manual Noise Removal

Visual inspection of denoised signals was conducted to identify recordings dominated by noise. This manual step allowed the identification of spurious data that automated filtering might miss, such as signals with irregular, non-physiological patterns. Any identified noisy recordings were excluded from further analysis and archived for transparency.

### Standardization of RRi Data

To facilitate comparison across individuals, RRi values were standardized within each recording. The mean and standard deviation of denoised RRi data were calculated for each individual. Standardized RRi values (z-scores) were then computed to ensure that inter-individual variability in baseline RRi values did not confound subsequent analyses.

rri\_data[, rri\_mean := mean(rr\_denoised, na.rm = TRUE), by = id]  
rri\_data[, rri\_sd := sd(rr\_denoised, na.rm = TRUE), by = id]  
rri\_data[, rri\_std := (rr\_denoised - rri\_mean) / rri\_sd, by = id]

## Sobol Sensitivity Analysis

This section provides a detailed explanation of the Sobol sensitivity analysis conducted to evaluate the influence of model parameters on RRi dynamics. The analysis quantifies the variance contributions of individual parameters, enabling insight into their relative importance in driving model output.

### Sampling Parameter Space

To evaluate the sensitivity of model parameters, random uniform samples were drawn for each parameter from the 95% CI estimated physiological ranges. These ranges were derived from posterior parameter estimates and reflect plausible RRi dynamics during exercise and recovery compatible with the observed data. For each Monte Carlo simulation, 1,000 parameter sets were sampled to ensure robust coverage of the parameter space.

sobol\_sample <- function(n\_samples) {  
 out <- data.frame(  
 alpha = runif(n\_samples, min = 850.57, max = 872.85),   
 beta = runif(n\_samples, min = -359.81, max = -330.97),   
 c = runif(n\_samples, min = 0.82, max = 0.86),   
 lambda = runif(n\_samples, min = -3.16, max = -2.94),   
 phi = runif(n\_samples, min = -2.71, max = -2.48),  
 tau = runif(n\_samples, min = 6.61, max = 6.81),   
 delta = runif(n\_samples, min = 3.05, max = 3.44)  
 )  
}

### Sobol Variance Decomposition

The Sobol analysis decomposes the total variance of the model output into contributions from individual parameters. For each parameter, a perturbed dataset was generated by varying the selected parameter while averaging others. The variance of the perturbed model output was compared to the total variance to compute the first-order Sobol indices:

These indices provide a measure of each parameter’s individual contribution to output variability.

sobol\_variance <- function(model\_function, samples, time\_points) {  
 total\_output <- sapply(1:nrow(samples), function(i) {  
 model\_function(as.numeric(samples[i, ]), time\_points)  
 })  
   
 total\_variance <- apply(total\_output, 1, var)  
   
 sobol\_indices <- matrix(0, nrow = length(time\_points), ncol = ncol(samples),  
 dimnames = list(NULL, names(samples)))  
   
 for (param in names(samples)) {  
 perturbed\_output <- sapply(1:nrow(samples), function(i) {  
 modified\_params <- as.numeric(samples[i, ])  
 modified\_params[names(samples) != param] <- apply(samples[names(samples) != param], 2, mean)  
 model\_function(modified\_params, time\_points)  
 })  
   
 variance\_contrib <- apply(perturbed\_output, 1, var)  
 sobol\_indices[, param] <- variance\_contrib / total\_variance  
 }  
   
 return(data.frame(time = time\_points, sobol\_indices))  
}

### Monte Carlo Simulations and Bootstrapping

To ensure the robustness of sensitivity estimates, 1,000 Monte Carlo simulations were performed, each with 1,000 parameter sets. Bootstrapping was applied to compute confidence intervals for Sobol indices, quantifying the uncertainty of parameter sensitivity across simulations.

n\_boot <- 1000  
boot <- vector(mode = "list", length = n\_boot)  
  
for (i in seq\_len(n\_boot)) {  
 boot[[i]] <- sobol\_variance(  
 model\_function = rr\_interval\_model,   
 samples = sobol\_sample(n\_samples = 1000),   
 time\_points = seq(0, 20, by = 0.1)  
 )  
}

### Results and Interpretation

The code that generated the summary table of Sobol indices is provided below:

sobol\_results[  
 j = list(estimate = mean(mean), se = mean(se)) |> lapply(round, 5),   
 by = variable  
 ][, `:=`(  
 lower = round(estimate - se \* qnorm(.975), 5),  
 upper = round(estimate + se \* qnorm(.975), 5))  
 ] |> knitr::kable()

## Parameter Estimation

### Individual-Level Parameter Estimation

This section describes the step-by-step methodology for performing individual-level parameter estimation in the hierarchical modeling of RRi dynamics during exercise and recovery. The process utilizes Bayesian posterior sampling with pre-defined priors to estimate subject-specific parameters, enabling detailed insights into personalized cardiovascular responses.

#### Loading and Preparing the Data

RRi data for individual subjects were filtered to remove missing or invalid standardized RRi values (). The input data were structured such that each subject’s measurements were indexed by their unique identifier (), ensuring traceability during individual-level analysis.

m\_data <- rri\_data[!is.na(rri\_std)]

#### Model Initialization and Posterior Sampling

The prior-only model, previously defined and fitted, served as the initial framework for sampling individual-level posteriors. Unique subject IDs were identified, and a loop was employed to sequentially fit the model for each subject’s data. For computational efficiency, the pipeline checks if a posterior sample already exists for a given subject to avoid redundant computation.

list\_models[[i]] <- update(  
 object = m\_model,  
 newdata = m\_data[id == i],  
 seed = 1234,  
 iter = 10000, warmup = 5000,  
 chains = 5, cores = 5,  
 control = list(adapt\_delta = 0.99, max\_treedepth = 50),  
 sample\_prior = "no",  
 file = file  
)

The posterior samples from all subjects were combined into a single data table, facilitating downstream analysis and parameter transformation.

#### Transforming Parameters to Original Scale

Parameters such as (baseline RRi) and (exercise-induced decline) were transformed back to their original scale using the subject-specific mean and standard deviation of RRi. This step ensures that parameter estimates are physiologically interpretable and directly comparable across individuals.

df\_models[, b\_alpha\_Intercept := b\_alpha\_Intercept \* rri\_sd + rri\_mean, id]  
df\_models[, b\_beta\_Intercept := b\_beta\_Intercept \* rri\_sd, id]  
df\_models[, sigma := sigma \* rri\_sd, id]

#### Summarizing Parameter Estimates

For each subject, the posterior means and standard deviations of parameters () were computed to create summary statistics. These metrics provide central tendencies and variability for each estimated parameter.

mu\_data <- df\_models[, lapply(.SD, mean), id, .SDcols = b\_alpha\_Intercept:sigma]  
se\_data <- df\_models[, lapply(.SD, sd), id, .SDcols = b\_alpha\_Intercept:sigma]

#### Validation Against Observed Data

Estimated parameters were used to simulate RRi dynamics for each subject. Observed RRi data were compared to the simulated predictions to calculate error metrics, including mean absolute percentage error (MAPE), root mean squared error (RMSE), and . These metrics quantify the accuracy of the individual-level model in capturing the observed RRi dynamics.

long\_data[, error := rr\_denoised - pred\_rr, id]  
long\_data[, mape := mean(abs(error/rr\_denoised), na.rm = T), id]  
long\_data[, rmse := sqrt(mean((error^2), na.rm = TRUE)), id]  
long\_data[, rsquared := 1-sum(error^2)/sum((rr\_denoised-rr\_bar)^2), id]

### Group-Level Parameter Estimation

This section describes the methodology used to perform group-level parameter estimation for modeling RR interval (RRi) dynamics across subjects. The analysis aggregates individual-level posterior estimates into a hierarchical framework, leveraging Bayesian modeling to derive population-level parameters while preserving individual variability.

#### Data and Model Setup

The dataset for this stage consists of posterior means and standard errors ( and SE) for parameters estimated at the individual level. Each parameter is modeled hierarchically using a univariate Bayesian framework, allowing for random effects across subjects.

The response variable for each parameter-specific model () is expressed as:

where represents the fixed effect (population mean), the random effect for subject , and the residual variance. This hierarchical specification captures variability both between and within subjects.

#### Model Specification

Each parameter () is modeled separately, using the posterior means and standard errors from the individual-level analysis as inputs. For each parameter-specific model:

1. The response variable () represents the posterior mean. (2) The standard error () is incorporated using the se() function in **brms**. (3) Subject-level random effects () capture between-subject variability.

m\_alpha\_model <- bf(alphaMu | se(alphaSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_beta\_model <- bf(betaMu | se(betaSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_c\_model <- bf(cMu | se(cSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_lambda\_model <- bf(lambdaMu | se(lambdaSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_phi\_model <- bf(phiMu | se(phiSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_tau\_model <- bf(tauMu | se(tauSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_delta\_model <- bf(deltaMu | se(deltaSE, sigma = TRUE) ~ 1 + (1 | id))  
m\_sigma\_model <- bf(sigmaMu | se(sigmaSE, sigma = TRUE) ~ 1 + (1 | id))

#### Prior Specification

Informative priors were specified for each parameter to reflect physiological constraints and ensure robust estimation. For instance, (baseline RRi) was given a prior centered around 850 ms with a standard deviation of 10 ms, while (exercise-induced decline) was constrained to negative values. These priors guide the model while allowing sufficient flexibility for parameter estimation.

#### Fitting and Post-Processing

The models were fit using **brms**, leveraging Hamiltonian Monte Carlo for efficient sampling. Each model was initialized with 10,000 iterations, including 5,000 warmup steps, and run with five chains to ensure convergence.

fit\_model <- function(model, priors) {  
 brms::brm(formula = model,  
 data = id\_data,  
 prior = priors,  
 family = gaussian(),  
 chains = 5, cores = 5, seed = 1234,  
 iter = 10000, warmup = 5000,  
 control = list(adapt\_delta = .99,  
 max\_treedepth = 50),  
 file = paste0("models/group\_level/",  
 deparse(substitute(model)),  
 ".RDS")  
 )  
}

Posterior distributions for all parameters were extracted and compiled into a single dataset for summary and visualization.

m\_fits\_df <- lapply(models, as\_draws\_df) |>   
 rbindlist(idcol = "Parameter")

The final dataset includes posterior estimates for population-level means (), subject-level variability (), and residual variance () for each parameter. These estimates are stored in group\_level\_posterior.csv, providing a comprehensive view of group-level dynamics and inter-individual differences.

# References

1. Salsekar, B. & Wadhwani, A. Filtering of ECG signal using butterworth filter and its feature extraction. *Int. J. Eng. Sci. Technol* **4**, 1292–1298 (2012).

2. Fedotov, A. Selection of parameters of bandpass filtering of the ECG signal for heart rhythm monitoring systems. *Biomedical Engineering* **50**, 114–118 (2016).