Trabajo Fin de Máster

Los valores perdidos en el muestreo de poblaciones finitas. Técnicas de imputación

Gabriel Román Radío

Máster en Técnicas Estadísticas
Curso 2016 - 2017

Propuesta de Trabajo Fin de Máster

Título en galego: Os valores esquecidos na mostraxe de poboacións finitas. Técnicas de imputación
Título en español: Los valores perdidos en el muestreo de poblaciones finitas. Técnicas de imputación
English title: The missing values in finite population sampling. Imputation techniques
Modalidad: modalidad A.
Autor/a: Gabriel Román Radío, Universidad de Vigo.
Director/a: Antonio Vaamonde Liste, Universidad de Vigo.
Breve resumen del trabajo: Los valores perdidos suponen un problema, actualmente no bien resuelto, en las encuestas por muestreo. Cuando es necesaria la respuesta completa- porque lo requiere la estimación a realizar o porque lo exigen las técnicas estadísticas a utilizar en el análisis- se aplican diversas técnicas de imputación, que permiten asignar valores a los casos y variables con valores omitidos. Sin embargo las distintas técnicas no son neutrales en relación con los resultados obtenidos (por ejemplo en las encuestas electorales), y una elección inadecuada puede introducir sesgos inadmisibles. Este trabajo pretende revisar de forma crítica las distintas técnicas de imputación existentes, valorando sus ventajas e inconvenientes, e implementar su aplicación con el programa estadístico R. Eventualmente se podrá añadir a este trabajo un estudio de simulación que permita comprobar la eficacia de los diferentes métodos en escenarios diversos.

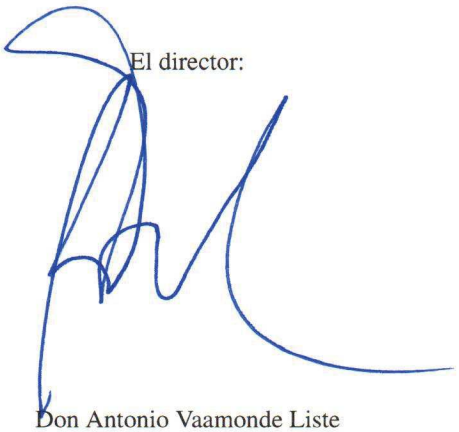
Don Antonio Vaamonde Liste, Catedrático de Universidad de Estadística e Investigación Operativa de la Universidad de Vigo informa que el Trabajo Fin de Máster titulado:

" Los valores perdidos en el muestreo de poblaciones finitas. Técnicas de imputación "

fue realizado bajo su dirección por don Gabriel Román Radío para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, da su conformidad para su presentación y defensa ante un tribunal.

En Vigo, a 6 de julio de 2017.

El director:



Don Antonio Vaamonde Liste

El autor:

Don Gabriel Román Radío

Agradecimientos

El autor de este trabajo expresa su agradecimiento a:

- Antonio Vaamonde Liste - Catedrático de Universidad de Estadística e Investigación Operativa de la Universidad de Vigo - por su apoyo y confianza durante todo el desarrollo del mismo.
- Mi madre, María Teresa Radío Rosal - Licenciada en filología francesa y máster en francés comercial - por su apoyo constante, su entrega y superación.
- Gracias

Índice general

Resumen

Resumen en español

Los valores perdidos suponen un obstáculo - actualmente no bien resuelto - cuando es necesaria la respuesta completa - porque lo requiere la estimación a realizar o porque lo exigen las técnicas estadísticas a utilizar en el análisis. Las técnicas de imputación permiten asignar valores a los casos y variables con valores omitidos. No obstante, las técnicas de imputación no son neutrales en relación con los resultados obtenidos y la elección inadecuada de la misma podría ocasionar alteraciones en la distribución de los datos. Este trabajo expone las diversas técnicas de imputación existentes, la evaluación de sus ventajas e inconvenientes y su aplicación con distintos paquetes del lenguaje de programación estadístico R.

Resumo en galego

Os valores esquecidos supoñen un obstáculo - actualmente no ben resolto - cando é necesaria a resposta compreta - porque o require a estimación a facer ou porque o esixen as técnicas estatísticas a empregar no análise. As técnicas de imputación permiten asignar valores aos casos e variables con valores omitidos. Non obstante, as técnicas de imputación no son neutrais en relación cos resultados obtidos e a elección inadecuada da mesma podería ocasionar alteracións na distribución dos datos. Este traballo expón as diversas técnicas de imputación existentes, a evaluación das súas vantaxes e inconvenientes e a súa aplicación con distintos paquetes da linguaxe de programación estatística R.

English abstract

Missing values represent an obstacle - currently not well solved - when the full response is necessary - because estimation requires it to perform or it is necessary for the statistical techniques used in the analysis. Imputation techniques allow to assign values to variables with missing values and cases. However, imputation techniques are not neutral in relation to the results obtained and the inappropriate choice could lead to alteration in the distribution of data. This work exposes the various imputation techniques existing, the assessment of their advantages and disadvantages and its application with different packages for the R statistical programming language.

Prefacio

En el desarrollo teórico de la mayoría de técnicas y modelos estadísticos no es habitual considerar algunas cuestiones que se manifiestan en la aplicación práctica. Un dilema, al que con seguridad se ha enfrentado cualquier analista, es el de los datos faltantes, también denominados perdidos o incompletos. Este problema conduce a que, en la mayoría de los estudios muestrales y/o censales, el analista se encuentre, muy probablemente, con espacios en blanco en el área de respuesta, designados " no respuesta ". La evaluación de ausencia de la no respuesta en un registro varía según varios factores como: el propio estudio, el grado de dificultad para rellenar el cuestionario o la medición de una unidad. Dependerá del analista valorar el registro como pérdida parcial o total si el número de no respuestas es considerable. Cuando no es posible ignorar la no respuesta, la técnica más empleada es completar los espacios en blanco o faltantes con valores plausibles: la imputación.

Son muchas las técnicas de imputación que han surgido, sobre todo desde la década de los sesenta del siglo pasado, que se sirven de enfoques univariantes y multivariantes a través de métodos de verosimilitud, regresión y descomposición de matrices en valores singulares. A pesar de este progreso, actualmente no se ha hallado una metodología capaz de reproducir los datos de modo totalmente satisfactorio, debido, generalmente, a la compleja problemática que presentan las alteraciones de la distribución de los datos.

El objetivo de este trabajo es exponer las principales estrategias vigentes para el análisis estadístico de matrices de datos incompletos y elaborar una guía o herramienta de consulta para el analista de datos en el supuesto de datos incompletos. El estudio se alimenta de numerosas publicaciones firmadas por autores especialistas en el tratamiento de valores perdidos, entre ellos Rubin, Little, Cochran, Figueredo y Schafer.

- Primer capítulo: Análisis de dos vertientes: la primera estudia los errores en las encuestas, las causas y efectos de la no respuesta; la segunda plantea los tratamientos de la no respuesta en las distintas fases del muestreo.
- Segundo capítulo: Análisis de la imputación de la no respuesta a través de sus efectos en la estimación, en los métodos de clasificación de unidades, en los modelos de generación de datos faltantes y en los procedimientos tradicionales.
- Tercer capítulo: Análisis de las técnicas de imputación existentes, evaluando sus ventajas e inconvenientes desde el punto de vista teórico.
- Cuarto capítulo: Análisis de la aplicación de técnicas de imputación a través del manejo de diferentes paquetes del programa estadístico R.
- Quinto capítulo: Conclusiones sobre las diferentes técnicas de imputación.

Capítulo 1

La no respuesta

1.1. Introducción

La finalidad del muestreo de poblaciones finitas es obtener información sobre las características desconocidas de la población. Para ello se utiliza la información recogida de una parte de la población - la muestra - y se emplean técnicas de estimación para obtener la información buscada. Al recoger los datos de una muestra puede surgir la " no respuesta ", que impide la utilización total o parcial de las unidades seleccionadas; es decir elementos de la muestra que no responden a una o varias de las cuestiones planteadas. Dos tratamientos son posibles, incluso complementarios, según el momento de ejecución de la encuesta: el primero es evitar o reducir la no respuesta, en la fase de planificación o durante los trabajos de campo de la encuesta; el segundo es sustituir el valor perdido por otro válido. Esta es la fase donde se incluyen las técnicas de imputación que constituyen el contenido de este trabajo.

1.2. Errores en las encuestas

La calidad de la información estadística se logra extremando el cuidado en la realización de las diferentes etapas que componen una muestra o un censo. Se deben evitar o reducir no sólo los errores de estimación derivados de las técnicas de muestreo, sino también los errores ajenos al muestreo, que aparecen también cuando se realiza un censo, o estudio de toda la población.

1.2.1. Errores de no observación

Se distinguen tres causas:

- La no cobertura.
- Las respuestas inconsistentes.
- La no respuesta.

La no cobertura es la ausencia de elementos que deberían estar presentes en el marco con el que se selecciona la muestra (error negativo). La sobrecobertura es la inclusión de elementos que no deberían estar presentes (error positivo). Las respuestas inconsistentes se deben a la respuesta incorrecta o muy poco real. La no respuesta consiste en la no obtención de observaciones; tanto de tipo completo (entrevistas no realizadas) como de tipo parcial (respuestas omitidas). En Martos (1995) se definen como observaciones totales o parciales.

Primer tipo: las causas por las que no se obtienen observaciones en el total de una unidad de la muestra pueden ser producidas por:

- Ausencia en el momento de la visita.
- Rechazo a la entrevista.
- Incapacidad o imposibilidad.

- Pérdida del cuestionario realizado.

Segundo tipo: las causas por las que no se obtienen observaciones parciales de la unidad de la muestra pueden ser producidas por:

- Rechazo a la pregunta por motivos personales, de intimidad, fiscales, etc...
- Mala exposición de la pregunta o que no se entiende.
- Falta de conocimientos para contestar.

Estas causas, totales y parciales, llevan a la falta de información en el muestreo. Se pueden desglosar entre las producidas por las opciones no sabe/no contesta y las que aparecen en blanco o son ilegibles. Para autores como Cruz (1990), Martín (1981) y Braña et al (1998), la contestación no sabe puede o no ser una información faltante, y en algunas circunstancias puede ser una respuesta tan válida como cualquier otra. El no contesta o ilegible se considera una falta de información que origina problemas básicos cuando (Cruz, 1990; Durrant, 2005; Martos, 1995):

1. Se producen sesgos porque la información faltante no es una muestra aleatoria de la población y tiene magnitud desconocida.
2. La información faltante es necesaria para el perfil del entrevistado.
3. En el análisis de etapas múltiples sobre tres o más variables, no es posible clasificar al individuo en las tablas de entrada múltiple.
4. Una disminución en el tamaño de la muestra produce un aumento en la varianza del estimador y reduce la precisión.

1.3. Efectos de la no respuesta

Cruz (1990), Díez (1996), Martín (1981) y Martos (1995) indican que al realizar un estudio de la no respuesta se debe dividir la población en dos estratos: los que responden y los que no responden. El efecto de la no respuesta sobre la estimación se puede analizar en las siguientes vertientes:

- En la estimación puntual de la media y del total.
- En la estimación de la varianza y razón poblacionales.

Para ello emplearemos la siguiente notación, se denotará por Y_1 al valor de la característica Y en la unidad i, responda o no y por X_1 al valor de la característica Y en la unidad i que responde.

1.3.1. En la estimación de la media poblacional

Definición 1.3.1.1: La media poblacional a estimar: $\overline{Y}_N = 1/N \cdot \sum_{i=1}^{N} \cdot Y_1$

Proposición 1.3.1.2: La media poblacional puede descomponerse mediante una suma de dos términos, correspondiente a las unidades que no responden y a las que responden.
Demostración Martos (1995):

$$\overline{Y}_N = 1/N \cdot \sum_{i=1}^{N} Y_1 - 1/N \cdot \left[\sum_{i=1}^{N_1} X_1 + \sum_{i=N_1+1}^{N} Y_1 \right] = 1/N \cdot \left[N_1 \cdot \overline{Y}_{N_1} + N_2 \cdot \overline{Y}_{N_2} \right] = t_{N_1} \cdot \overline{Y}_{N_1} + t_{N_2} \cdot \overline{Y}_{N_2}$$

Corolario 1.3.1.3: La media poblacional se puede descomponer en la suma de varios términos, que dependan de las unidades que responden y de las unidades que no responden, suponiendo que existan varios tipos de no respuesta.

$$\overline{Y}_N = t_{N_1} \cdot \overline{Y}_{N_1} + \sum_{i=2}^{k} t_{N_{21}} \cdot \overline{Y}_{N_{21}}$$

siendo K los diferentes tipos de no respuesta, $\overline{Y}_{N_{21}}$ la media y $t_{N_{21}}$ la tasa de no respuesta para cada uno de los diferentes grupos.

Proposición 1.3.1.4: Siendo $\overline{Y}_n$ un estimador muestral de $\overline{Y}_N$, tal que $E\left(\overline{Y}_n\right) = \overline{Y}_{N_1}$, media poblacional de los que responden, la cantidad de sesgo y sesgo relativo (SR) al estimar $\overline{Y}_N$ por $\overline{Y}_n$ vienen determinadas por:

$$sesgo\left(\overline{Y}_n\right) = t_{N_2} \cdot \left[\overline{Y}_{N_1} - \overline{Y}_{N_2}\right] \tag{1.1}$$

$$SR\left(\overline{Y}_n\right) = t_{N_2}\left(1-W\right)/\left(1-t_{N_2}\cdot\left(1-W\right)\right), \text{siendo } W = \overline{Y}_{N_2}/\overline{Y}_{N_1}$$

Demostración Martos (1995):

Dado que el sesgo viene definido por: $sesgo = E\left(\overline{Y}_n\right) - \overline{Y}_{N_1} - \overline{Y}_N$

De acuerdo con la proposición 1.3.1.2, se puede obtener que:

$$sesgo = \overline{Y}_{N_1} - \overline{Y}_N = \overline{Y}_{N_1} - \left[t_{N_1}\cdot\overline{Y}_{N_1} + t_{N_2}\cdot\overline{Y}_{N_2}\right] = t_{N_2}\cdot\left[\overline{Y}_{N_1} - \overline{Y}_{N_2}\right]$$

Por otro lado, el sesgo relativo es el cociente:

$$SR\left(\overline{Y}_n\right) = t_{N_2}\cdot\left[\overline{Y}_{N_1} - \overline{Y}_{N_2}\right]/\overline{Y}_N$$

y a través de la proposición 1.3.1.2, obtenemos que:

$$\begin{aligned} SR\left(\overline{Y}_n\right) =& t_{N_2}\cdot\left[\overline{Y}_{N_1} - \overline{Y}_{N_2}\right]/\left(t_{N_1}\cdot\overline{Y}_{N_1} + t_{N_2}\cdot\overline{Y}_{N_2}\right) = \\ & t_{N_2}\cdot\left(1-\overline{Y}_{N_2}/\overline{Y}_{N_1}\right)/\left(t_{N_1} + t_{N_2}\cdot\overline{Y}_{N_2}/\overline{Y}_{N_1}\right) = \\ & t_{N_2}\cdot\left(1-W\right)/\left(1-t_{N_2}+t_{N_2}\cdot W\right) = \\ & t_{N_2}\cdot\left(1-W\right)/\left(1-t_{N_2}\cdot\left(1-W\right)\right) \end{aligned}$$

1.3.2. En la estimación de la varianza poblacional

Definición 1.3.2.1: Varianza ajustada o cuasivarianza:

- Poblacional: $S_N^2 = 1/\left(N-1\right)\cdot\sum_{i=1}^{N}\left(Y_1 - \overline{Y}_N\right)^2$
- Poblacional de los que responden: $S_{N_1}^2 = 1/\left(N_1-1\right)\cdot\sum_{i=1}^{N_1}\left(X_1 - \overline{Y}_{N_1}\right)^2$
- Poblacional de los que no responden: $S_{N_2}^2 = 1/\left(N_2-1\right)\cdot\sum_{i=N_1+1}^{N}\left(Y_1 - \overline{Y}_{N_2}\right)^2$

Proposición 1.3.2.2: La varianza ajustada poblacional se puede descomponer en suma de cuatro términos, que dependen de las unidades que responden y de las que no responden de la siguiente manera:

$$\begin{aligned} S_N^2 =& \left(N_1-1\right)/\left(N-1\right)\cdot S_{N_1}^2 + N_1/\left(N-1\right)\cdot\left(\overline{Y}_{N_1} - \overline{Y}_N\right)^2 + \\ & \left(N_2-1\right)/\left(N-1\right)\cdot S_{N_2}^2 + N_2/\left(N-1\right)\cdot\left(\overline{Y}_{N_2} - \overline{Y}_N\right)^2 \end{aligned}$$

Demostración Martos (1995):

De la definición 1.3.2.1, se extrae que:

$$\begin{aligned} \left(N-1\right)\cdot S_N^2 =& \sum_{i=1}^{N}\left(Y_1 - \overline{Y}_N\right)^2 = \sum_{i=1}^{N_1}\left(X_1 - \overline{Y}_N\right)^2 + \sum_{i=N_1+1}^{N}\left(Y_1 - \overline{Y}_N\right)^2 = \\ & \sum_{i=1}^{N_1}\left[\left(X_1 - \overline{Y}_{N_1}\right) + \left(\overline{Y}_{N_1} - \overline{Y}_N\right)\right]^2 + \sum_{i=N_1+1}^{N}\left[\left(Y_1 - \overline{Y}_{N_2}\right)\right]^2 = \end{aligned}$$

al desarrollar, los términos con doble producto, se observa que son iguales a cero, dado que para el primer sumando se tiene:

$$\sum \left(X_1 - \overline{Y}_{N_1}\right) \cdot \left(\overline{Y}_{N_1} - \overline{Y}_N\right) = \left(\overline{Y}_{N_1} - \overline{Y}_N\right) \cdot \sum \left(X_1 - \overline{Y}_{N_1}\right) = \left(\overline{Y}_{N_1} - \overline{Y}_N\right) \cdot 0$$

siendo para el otro sumando de manera análoga, por lo que se obtendrá:

$$\sum_{i=1}^{N_1} \left(X_1 - \overline{Y}_{N_1}\right)^2 + \sum_{i=1}^{N_1} \left(\overline{Y}_{N_1} - \overline{Y}_N\right)^2 + \sum_{i=N_1+1}^{N} \left(Y_1 - \overline{Y}_{N_2}\right)^2 + \sum_{i=N_1+1}^{N} \left(\overline{Y}_{N_2} - \overline{Y}_N\right)^2 =$$

y de acuerdo con la definición 1.3.2.1, se obtiene:

$$(N_1 - 1) \cdot S_{N_1}^2 + N_1 \cdot \left(\overline{Y}_{N_1} - \overline{Y}_N\right)^2 + (N_2 - 1) \cdot S_{N_2}^2 + N_2 \cdot \left(\overline{Y}_{N_2} - \overline{Y}_N\right)^2$$

despejando S_N^2 de la igualdad, se obtiene la expresión propuesta.

Corolario 1.3.2.4: Una descomposición de la varianza poblacional es $VAR_N = t_{N_1} \cdot VAR_{N_1} + t_{N_2} \cdot VAR_{N_2} + t_{N_1} \cdot t_{N_2} \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2$, siendo VAR_N, VAR_{N_1} y VAR_{N_2} las varianzas poblacionales de los que responden y no responden, respectivamente.

Demostración Martos (1995):

A través de la proposición 1.3.1.2 se deduce:

$$\begin{aligned} \overline{Y}_{N_1} - \overline{Y}_N &= t_{N_2} \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right) \\ \overline{Y}_{N_2} - \overline{Y}_N &= -1 \cdot t_{N_1} \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right) \end{aligned}$$

Elevando al cuadrado y multiplicando cada expresión por N_1 y N_2 respectivamente y sumando miembro a miembro, se obtiene:

$$\begin{aligned} N_1 \cdot \left(\overline{Y}_{N_1} - \overline{Y}_N\right)^2 + N_2 \cdot \left(\overline{Y}_{N_2} - \overline{Y}_N\right)^2 = \\ N_1 \cdot t_{N_2}^2 \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2 + N_2 \cdot t_{N_1}^2 \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2 = \\ \left(N_1 \cdot t_{N_2}^2 + N_2 \cdot t_{N_1}^2\right) \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2 = \\ N_1 \cdot N_2 \cdot (N_1 + N_2)/N^2 \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2 = \\ N_1 \cdot N_2 \cdot 1/N \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2 \end{aligned} \tag{1.2}$$

Utilizando la expresión obtenida en la proposición 1.3.2.2 y sustituyendo en la misma, la expresión (1.2), se obtiene:

$$(N-1) \cdot S_N^2 = (N_1 - 1) \cdot S_{N_1}^2 + (N_2 - 1) \cdot S_{N_2}^2 + N_1 \cdot N_2 \cdot 1/N \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2$$

A través de la relación existente entre la varianza ajustada y la varianza, obtenemos:

$$N \cdot VAR_N = N_1 \cdot VAR_{N_1} + N_2 \cdot VAR_{N_2} + N_1 \cdot N_2 \cdot 1/N \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)^2$$

Dividiendo por N, se obtiene la expresión propuesta.

Corolario 1.3.2.5: Siendo S_n^2 la varianza ajustada muestral, ésta puede descomponerse en suma de cuatro términos, que dependen de las unidades que responden y no responden en la muestra:

$$S_n^2 = (n_1-1)/(n-1)\cdot S_{n_1}^2 + n_1/(n-1)\cdot\left(\overline{Y}_{n_1}-\overline{Y}_n\right)^2 + (n_2-1)/(n-1)\cdot S_{n_2}^2 + n_2/(n-1)\cdot\left(\overline{Y}_{n_2}-\overline{Y}_n\right)^2$$

Por lo tanto, se obtiene:

$$VAR_n = t_{n_1}\cdot VAR_{n_1} + t_{n_2}\cdot VAR_{n_2} + t_{n_1}\cdot t_{n_2}\cdot\left(\overline{Y}_{n_1}-\overline{Y}_{n_2}\right)^2$$

siendo, VAR_n, VAR_{n_1}, y VAR_{n_2} la varianza muestral, de los que responden y de los que no responden.

Proposición 1.3.2.6: Siendo S_n^2 un estimador muestral de S_N^2, tal que $E\left(S_n^2\right) = S_{N_1}^2$, varianza ajustada poblacional de los que responden, la cantidad de sesgo al estimar S_N^2 por S_n^2, viene determinado por:

$$(N_2-1)/(N-1)\cdot\left[S_{N_1}^2 - S_{N_2}^2\right] - N_1/(N-1)\cdot\left(\overline{Y}_{N_1}-\overline{Y}_N\right)^2 - N_2/(N-1)\cdot\left(\overline{Y}_{N_2}-\overline{Y}_N\right)^2 + 1/(N-1)\cdot S_{N_1}^2$$

o bien por:

$$(N_2-1)/(N-1)\cdot\left[S_{N_1}^2 - S_{N_2}^2\right] - N_1\cdot N_2\cdot 1/N\cdot 1/(N-1)\cdot\left(\overline{Y}_{N_1}-\overline{Y}_{N_2}\right)^2 + 1/(N-1)\cdot S_{N_1}^2$$

Demostración Martos (1995):

Teniendo en cuenta que: $Sesgo = E\left(S_n^2\right) - S_N^2 = S_{N_1}^2 - S_N^2$ y utilizando la expresión de S_N^2 obtenida en la proposición 1.3.2.1, obtenemos que:

$$\begin{aligned} Sesgo = \quad & S_{N_1}^2 - [(N_1-1)/(N-1)\cdot S_{N_1}^2 + N_1/(N-1)\cdot\left(\overline{Y}_{N_1}-\overline{Y}_N\right)^2 + \\ & (N_2-1)/(N-1)\cdot S_{N_2}^2 + N_2/(N-1)\cdot\left(\overline{Y}_{N_2}-\overline{Y}_N\right)^2] = \\ & [1-(N_1-1)/(N-1)]\cdot S_{N_1}^2 - (N_2-1)/(N-1)\cdot S_{N_2}^2 - \\ & N_1/(N-1)\cdot\left(\overline{Y}_{N_1}-\overline{Y}_N\right)^2 - N_2/(N-1)\cdot\left(\overline{Y}_{N_2}-\overline{Y}_N\right)^2 \end{aligned} \tag{1.3}$$

El primer término, queda reducido a $N_2/(N-1)\cdot S_{N_1}^2$. Sumando y restando a la expresión anterior el término $1/(N-1)\cdot S_{N_1}^2$, obtenemos la primera expresión propuesta.

Si sustituimos los dos últimos términos de la expresión anterior por la expresión (1.2), se consigue la segunda expresión propuesta.

1.3.3. En la estimación del total poblacional

Definición 1.3.3.1: Total poblacional a estimar: $Y_N = \sum_{i=1}^{N} Y_1$

Proposición 1.3.3.2: El total poblacional puede descomponerse en una suma de dos términos, que dependen de las unidades que responden y no responden.

Demostración Martos (1995):

$$Y_N = \sum_{i=1}^{N} Y_1 - \sum_{i=1}^{N_1} X_1 + \sum_{i=N_1+1} Y_1 = N_1\cdot\overline{Y}_{N_1} + N_2\cdot\overline{Y}_{N_2}$$

Proposición 1.3.3.3: Siendo Y_n un estimador muestral de Y_N, tal que $E(Y_n) = Y_{N_1}$, total poblacional de los que responden, la cantidad de sesgo y sesgo relativo (SR) al estimar Y_N por Y_n, vienen determinadas por:

$$sesgo\,(Y_n) = -1 \cdot N_2 \cdot \overline{Y}_{N_2}$$
$$SR\,(Y_n) = -1 \cdot t_{N_2} \cdot W / \left(1 - t_{N_2} \cdot (1 - W)\right), \text{ siendo } W = \overline{Y}_{N_2} / \overline{Y}_{N_1}$$

Demostración Martos (1995):
Dado que el sesgo viene definido por:

$$sesgo = E\,(Y_n) - Y_N = Y_{N_1} - Y_N = N_1 \cdot \overline{Y}_{N_1} - \overline{Y}_N$$

Y de acuerdo con la proposición 1.3.3.3, se obtiene que el:

$$sesgo = N_1 \cdot \overline{Y}_{N_1} - \left[N_1 \cdot \overline{Y}_{N_1} + N_2 \cdot \overline{Y}_{N_2}\right] = -1 \cdot N_2 \cdot \overline{Y}_{N_2}$$

Para el sesgo relativo, se obtiene que:

$$SR\,(Y_n) = -1 \cdot N_2 \cdot \overline{Y}_{N_2} / Y_N = -1 \cdot N \cdot t_{N_2} \cdot \overline{Y}_{N_2} / Y_N = -1 \cdot t_{N_2} \cdot \overline{Y}_{N_2} / \overline{Y}_N$$

y siguiendo con las mismas transformaciones que las indicadas en la última parte de la proposición 1.3.1.4, se obtiene la expresión propuesta.

1.3.4. En la estimación de razones

Hay situaciones en las que se cree que la tasa de Y con respecto a otra característica Z es menos variable que la de Y. En este sentido algunos autores como Cruz (1990), Martos (1995) o Durrant (2005) consideran que en este caso sería mejor estimar R_N, la tasa de Y a Z en la población, a través de la muestra y después multiplicar por el total conocido de Z, para estimar el total de Y.

Definición 1.3.4.1: Siendo Y y Z dos características de una población, definimos razón de Y a Z en la población como el cociente:

$$R_N = Y_N / Z_N, \text{ siendo } Y_N, Z_N \text{ los totales poblacionales de Y y Z}$$

Proposición 1.3.4.2: Siendo R_n un estimador muestral de R_N, tal que $E\,(R_n) = R_{N_1}$, razón poblacional de los que responden, la cantidad de sesgo al estimar R_N por R_n, viene determinada por :

$$sesgo\,(R_n) = C_{N_2} \cdot \left[R_{N_1} - R_{N_2}\right], \text{ donde } C_{N_2} = Z_{N_2} / Z_N$$

Demostración Martos (1995):

$$\begin{aligned}
\text{Dado que el sesgo es } &= E\,(R_n) - R_N = R_{N_1} - R_N = \\
&\left(Y_{N_1} / Z_{N_1}\right) - \left(Y_N / Z_N\right) = \\
&\left(Y_{N_1} / Z_{N_1}\right) - \left(Y_{N_1} + Y_{N_2}\right) / Z_N = \\
&\left[Y_{N_1}\left(Z_N - Z_{N_1}\right) - \left(Z_{N_1} \cdot Y_{N_2}\right)\right] / \left(Z_N \cdot Z_{N_1}\right) = \\
&\left(Y_{N_1} \cdot Z_{N_2} - Z_{N_1} \cdot Y_{N_2}\right) / \left(Z_N \cdot Z_{N_1}\right) = \\
&\left(Y_{N_1} \cdot Z_{N_2}\right) / \left(Z_N \cdot Z_{N_1}\right) - \left(Z_{N_1} \cdot Y_{N_2}\right) / \left(Z_N \cdot Z_{N_1}\right) = \\
&\left(Z_{N_2} / Z_N\right) \cdot \left[\left(Y_{N_1} / Z_{N_1}\right) - \left(Y_{N_2} / Z_{N_2}\right)\right]
\end{aligned}$$

1.4. Tratamiento de la no respuesta

¿Cómo evitar la no respuesta? y si no hay respuesta, ¿Cómo tratarla? En esta sección se realiza una revisión de las técnicas más usuales para tratar la no respuesta en las diferentes etapas del muestreo:

- Planificación.
- Diseño.
- Realización.
- Procesamiento o Validación.

1.4.1. Fase de planificación

Determina los fines del estudio teniendo en cuenta las condiciones, recursos y limitaciones; entre ellas el marco o listado de unidades que constituyen la población, utilizada para seleccionar la muestra. Cruz (1990) o Deming (1960) aconsejan eliminar las unidades que son inaccesibles es decir que puedan causar dificultades no justificadas para los fines de la encuesta; eliminando de antemano unidades que casi seguro serán eliminadas después de ser seleccionadas.

1.4.2. Fase de diseño

En esta fase se determinan, entre otros elementos, el tamaño de muestra, el procedimiento de selección, y el cuestionario. Se puede conseguir que la no respuesta tenga un efecto más reducido atendiendo a algunos factores de efecto indirecto o directo en la tasa de respuesta: longitud y complejidad del cuestionario, redacción adecuada de las preguntas, tratamiento adecuado de cuestiones de carácter controvertido para encuestadores y encuestados, selección adecuada y control del equipo de campo (incluyendo a los entrevistadores).

1.4.3. Fase de campo o de realización

Es la última que puede evitar o disminuir la no respuesta, empleando diferentes técnicas como: mejoramiento de los procedimientos de entrevista, garantía del anonimato del entrevistado, motivación para la cooperación del entrevistado, o concretar con el entrevistado la visita.

Repetición de intentos para conseguir la información

Se basa en los intentos deliberados (Call Backs) de obtener información (respuesta) de la no respuesta como visitas y envíos repetidos del cuestionario. En 1953 Deming formuló un modelo que incorporaba informaciones como las características demográficas y socioeconómicas, cooperación del entrevistado, incentivos al entrevistador y coste de las visitas. Ello permitió distribuir a los entrevistados en grupos a través de la estimación del número medio de entrevistas durante un período determinado, y decidir, según la tasa de respuesta, realizar una o varias visitas adicionales. Se basó en dos elementos:

1. El número posible de nuevas respuestas, que debe ser suficientemente alto para que se puedan justificar las causas por las que la encuesta no ha sido respondida.
2. El coste de tales visitas y el tiempo disponible.

En 1949, Politz y Simmons proponen otro método, basado en el cálculo de la posibilidad de encontrar al entrevistado, suponiendo que todas las visitas se hacen en K períodos similares. Dicho método consiste en preguntar al entrevistado si habría estado disponible para la entrevista en cada uno de los K períodos. Si r es el número de períodos afirmativos $(0 \leq r \leq K)$ se toma la razón $r+1/K+1$ como la estimación de la frecuencia de las horas que está disponible el entrevistado para la entrevista. Por otro lado, la ponderación W es la inversa de dicha frecuencia, de manera que W es mayor cuanto menor es r. Los resultados de la primera visita se ponen en $k+1$ grupos según el valor de $r: 0,1,2,...,t,....K$. Para el grupo t, n_t es el número de entrevistas e $\overline{Y}_t$ es la media del grupo. De esta manera el estimador de Politz-Simmons para la media poblacional quedaría representado a través de la siguiente ecuación:

$$\overline{Y}_{ps} = \frac{\sum_{t=0}^{K} n_t \cdot Y_t \cdot W_t}{\sum_{t=0}^{K} n_t W_t}, \text{ con } W_t = \frac{K+1}{r+1} \tag{1.4}$$

De la ecuación (1.4) se obtiene una media sesgada, pero con menor sesgo que la media estándar y con varianza aumentada (Cochran,1977).

En 1959 Kish y Hess sugieren el procedimiento llamado reemplazo. Consiste en agregar a las nuevas direcciones de la encuesta otras direcciones de no respuesta, tomadas de una encuesta anterior y muestreo semejante, y aplicarlas a la encuesta en que se trabaja. El único requisito es que las nos respuestas tienen que ser semejantes a las respuestas que actúan de reemplazo.

Sea N el total de direcciones para una encuesta y t_R la tasa de respuesta que se espera obtener después de K entrevistas. El total de respuestas previstas será: $N \cdot t_R$. Si suponemos que tenemos un número de direcciones de no respuesta n de un estudio anterior y que la tasa de respuesta para estas direcciones y en esta encuesta es de $t_R^{'}$, después de K visitas.

El total de respuestas previstas será: $n \cdot t_R^{'}$.

Al reemplazar $n \cdot t_R^{'}$ en las $N \cdot t_R$, el total de respuestas que necesitamos es de:

$$N \cdot t_R - n \cdot t_R^{'}$$

Y el total de direcciones:

$$1/t_R \cdot \left(N \cdot t_R - n \cdot t_R^{'}\right) = N - n \cdot t_R^{'}/t_R$$

El área de incertidumbre debida a la no respuesta se reduce en alrededor de: $(1 - t_R) \cdot \left(1 - t_R^{'}\right)$.

Submuestreo de la no respuesta

El submuestreo de las no respuestas, se le conoce por el método de Hansen y Hurwitz (1946). Dicho método considera a la población en dos estratos: los que responden y los que no responden. El procedimiento del método de Hansen y Hurwitz muestra que al seleccionar una muestra, los que responden representan una muestra aleatoria del primer estrato y los que no responden, del segundo. Por lo que el procedimiento de dicho método, toma una submuestra de un tamaño conveniente y recoge información mediante entrevistas personales a esas unidades, posteriormente tiene en cuenta las dos muestras para obtener un estimador de la media poblacional.

Encuestación delegada

Se trata de encuestar a otro miembro de la familia, al no ser posible hacerlo sobre la persona indicada. No es aconsejable cuando la información es confidencial o personal.

Muestreo por cuotas

Se establecen cuotas en base a las características conocidas de la población en estudio. Luego se determina un diseño muestral de la encuesta. La etapa final, la realizan los entrevistadores porque ellos son los que eligen las personas a encuestar, siguiendo el plan de cuotas establecido.

Técnicas de respuesta aleatorizada

En palabras de autores como Cruz (1990) y Martos (1995) es una técnica desarrollada para obtener información de confianza cuando los entrevistados no dicen la verdad o no contestan por el carácter reservado de la pregunta. El encuestado debe seleccionar una de dos preguntas: una transcendente (opción 1) y una intranscendente (opción 2). Debe contestar con un si o no sin que el encuestador sepa cuál ha sido la pregunta seleccionada. La explicación matemática considerando que ambas cuestiones realizadas por el encuestador pueden ser complementarias es la siguiente:

Se denota por π a la verdadera proporción de la población encuestada que contesta a la pregunta transcendente (opción 1) y por p a la probabilidad conocida de que el mecanismo aleatorio elija la pregunta transcendente. Por otro lado, la proporción de la población, se denota por θ, describiendo a los que contestan afirmativamente, dicha proporción quedará expresada de la siguiente manera:

$$\theta = p \cdot \pi + (1-p) \cdot (1-\pi)$$

Se observa que la verdadera proporción de la población que contesta afirmativamente a la pregunta intranscendente es representada por $1-\pi$, al suponer que las cuestiones son complementarias. A la hora de estimar la proporción de la población θ por la proporción de la población que contesta afirmativamente en la encuesta, denotada por θ^*, se obtiene el siguiente modelo:

$$\theta^* = p \cdot \pi^* + (1-p) \cdot (1-\pi^*)$$

Operando se obtiene : $\pi^* = (\theta^* - 1 + p) / (2p+1)$, donde π^* es un estimador insesgado de π (Warner, 1965).

1.4.4. Fase de procesamiento o validación

En esta fase, ya no cabe la posibilidad de evitar la no respuesta, antes de proceder a la estimación de los parámetros poblacionales. Los procedimientos empleados son: depuración e imputación (capítulo 2).

La depuración permite separar los cuestionarios en aceptables y los que no tienen respuestas. Los procedimientos más usuales son Martos (1995):

- Sistemas generales de depuración basados en la Metodología de Fellegi & Oldt (1980).
- Método de estimación máximo-verosímil: Algoritmo E.M.
- Método general de depuración CIDAC.

Capítulo 2

Imputación para la no respuesta

2.1. Introducción

La imputación consiste en obtener datos completos y consistentes para elaborar un análisis y aplicar inferencia estadística. Martos (1995) define la imputación como la etapa final del proceso de depuración de datos que consiste en reemplazar los valores faltantes por otros aceptables. En décadas anteriores, Cruz (1990), era habitual ignorar valores faltantes en alguna variable por los métodos de eliminación por lista (listwise deletion) o por pares (pairwise deletion). Las no respuestas se ignoraban en el análisis, provocando pérdida de información. Actualmente se reconoce la importancia de utilizar métodos de imputación que conserven características propias de la variable a ser imputada, lo que permite disponer de datos completos para el análisis, y aumentar la precisión de las estimaciones. Los métodos más utilizados y aceptados incluyen información auxiliar del marco.

Los valores imputados se clasifican en tres categorías:

1. Valores construídos con modelos estadísticos de predicción.
2. Valores observados como elementos de respuesta que son similares a aquellos que no responden.
3. Valores construídos mediante la opinión de expertos.

La primera y segunda categoría utilizan modelos para producir valores sustitutos. La primera se basa en la relación asumida entre variables. La segunda utiliza métodos para clasificar qué elementos similares ofrecerán respuesta y cuáles serán los receptores. Los métodos de la tercera categoría implican el conocimiento de los expertos acerca del elemento imputado.

Existen dos enfoques: la imputación completa y la combinada con calibración o enfoque combinado. El enfoque de imputación completa indica el uso de imputación tanto para compensar la no respuesta en el ítem como también la no respuesta a la unidad, por lo que la matriz de datos resultantes tendrá tantas filas como individuos participen en la encuesta y tantas columnas como variables, será de dimensión $(n \times J)$. El enfoque combinado propone imputar para obtener respuesta completa solamente para las unidades que han contestado al menos a una variable de la encuesta (no respuesta en el ítem). La matriz rectangular de dicho enfoque, al igual que el enfoque de imputación completa, tendrá tantas columnas como variables, pero tendrá tantas filas como individuos respondan $(m \times J, m < n)$.

Las variables de las encuestas generalmente están afectadas tanto por la no respuesta a la unidad como la no respuesta al ítem, por lo que los valores y_{ik} de la i-ésima variable de estudio necesaria para construir estimadores de totales estarán disponibles sólo para los $K \in r_i \subset r \subset s$ donde r es el conjunto de individuos que responden a la muestra s, y r_i es el conjunto de individuos que responden a la i-ésima variable del estudio. La idea detrás del enfoque combinado es imputar aquellos valores pertenecientes al conjunto $r - r_i$, de manera que la matriz rectangular tenga tantas filas como individuos responden y tantas columnas como variables. De esta forma, la matriz resultante contendrá los elementos $\{y_{i \cdot k} : K \in r\}$:

$$y_{i \cdot k} = \begin{cases} y_{ik} \; K \in r_i \\ \\ \widehat{y}_{ik} \; K \in r - r_i \end{cases} \tag{2.1}$$

2.2. Efectos de la imputación en la no respuesta

En esta sección, se estudiará el efecto de la imputación en el sesgo de no respuesta, cuando se estima la media, el total y la razón poblacional. En lo que sigue Y_1 representará las unidades de la muestra en las que hay unidades con no respuesta. Después de realizar una imputación se considera que:

$$Y_1 = \begin{cases} X_1 \text{ para las unidades que responden} \\ X_1^* \text{ para las unidades que no responden, pero se les ha imputado un valor} \end{cases}$$

2.2.1. En la estimación de la media poblacional

Consideramos $\overline{Y}_n^*$ un estimador de $\overline{Y}_N$, tal que:

$$E\left(\overline{Y}_n^*\right) = t_{N_1} \cdot \overline{Y}_{N_1} + t_{N_2} \cdot Y_{N_2}^*$$

siendo $\overline{Y}_{N_2}^*$ una estimación de $\overline{Y}_{N_2}$ como consecuencia de una imputación. Teniendo en cuenta la proposición 1.3.1.2, se tiene que:

$$sesgo\left(\overline{Y}_n^*\right) = E\left(\overline{Y}_n^*\right) - \overline{Y}_N = t_{N_2} \cdot \left(\overline{Y}_{N_2}^* - \overline{Y}_{N_2}\right) \tag{2.2}$$

Vamos a comparar éste sesgo con el obtenido antes de la imputación, considerando que:

$$E\left(\overline{Y}_n\right) = \overline{Y}_{N_1}$$

Proposición 2.3.1.1: La imputación reduce en términos absolutos o a lo sumo deja invariante el sesgo de no respuesta en la estimación de la media poblacional.

Demostración Martos (1995):

Admitiendo la hipótesis de que $\overline{Y}_{N_2}^*$ está más cerca de la media poblacional de los individuos que no responden, que los que responden, podemos deducir que:

$$|\overline{Y}_{N_2}^* - \overline{Y}_{N_2}| \leq |\overline{Y}_{N_1} - \overline{Y}_{N_2}| \tag{2.3}$$

y según la proposición 1.3.1.4, como $\overline{Y}_n$ es el estimador antes de la imputación:

$$sesgo\left(\overline{Y}_n\right) = t_{N_2} \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right) \tag{2.4}$$

supuesto que $E\left(\overline{Y}_n\right) = \overline{Y}_{N_1}$ y comparando la expresión (2.2) y la (2.4) y teniendo en cuenta la expresión (2.3), se puede deducir que:

$$|sesgo\left(\overline{Y}_n^*\right)| \leq |sesgo\left(\overline{Y}_n\right)| \tag{2.5}$$

Si no admitimos la hipótesis, es decir, suponemos que $\overline{Y}_{N_2}^*$ está más cerca de la media poblacional de los individuos que responden, entonces tomaremos al imputar, como valor de $\overline{Y}_{N_2}^*$, el valor $\overline{Y}_{N_1}$ y, en consecuencia la expresión (2.3) será una igualdad y los sesgos iguales.

Corolario 2.3.1.2: Una cota superior del valor absoluto del sesgo después de la imputación en la estimación de la media poblacional, es:

$$t_{N_2} \cdot \left(\overline{Y}_{N_1} - \overline{Y}_{N_2}\right)$$

Demostración Martos (1995):

Se deduce de la proposición anterior, expresión (2.4) y (2.5).

2.2.2. En la estimación del total poblacional

Consideramos Y^* un estimador de Y_N, tal que: $E\left(Y_n^*\right) = N_1 \cdot \overline{Y}_{N_1} + N_2 \cdot \overline{Y}_{N_2}$, siendo $\overline{Y}_{N_2}^*$ una estimación de $\overline{Y}_{N_2}$ como consecuencia de una imputación.

Teniendo en cuenta la proposición 1.3.3.2, se obtiene:

$$sesgo\left(Y_n^*\right) = E\left(Y_n^*\right) - Y_N = N_2 \cdot \left(\overline{Y}_{N_2}^* - \overline{Y}_{N_2}\right) \tag{2.6}$$

Vamos a comparar éste sesgo con el obtenido antes de la imputación, considerando que $E\left(Y_n\right) = Y_{N_1}$.

Proposición 2.3.2.1: La imputación reduce en términos absolutos el sesgo de no respuesta en la estimación del total poblacional.

Demostración Martos (1995):

Admitiendo la hipótesis de que $\overline{Y}_{N_2}^*$ está más cerca de la media poblacional de los individuos que no responden, obtenemos:

$$|\overline{Y}_{N_2}^* - \overline{Y}_{N_2}| \leq |\overline{Y}_{N_2}| \tag{2.7}$$

según la proposición 1.3.3.3, se tiene:

$$sesgo\left(Y_n\right) = -1 \cdot N_2 \cdot \overline{Y}_{N_2} \tag{2.8}$$

suponiendo que $E\left(Y_n\right) = Y_{N_1}$, comparando la expresión (2.6) y (2.8) y teniendo en cuenta la expresión (2.7), se deduce:

$$|sesgo\left(\overline{Y}_n^*\right)| \leq |sesgo\left(Y_n\right)| \tag{2.9}$$

Si no admitimos la hipótesis, es decir, suponemos que $Y_{N_2}^*$ está más cerca de la media poblacional de los individuos que responden, entonces tomaremos al imputar, como valor de $Y_{N_2}^*$, el valor Y_{N_1} y en consecuencia, la expresión (2.7) quedará de la siguiente forma:

$$|\overline{Y}_{N_1} - \overline{Y}_{N_2}| \leq |\overline{Y}_{N_2}| \tag{2.10}$$

y comparando la expresión (2.6) y (2.8), teniendo en cuenta la expresión (2.10), seguirá siendo válida la expresión (2.9).

Corolario 2.3.2.2: Una cota superior del valor absoluto de sesgo después de la imputación en la estimación del total poblacional es:

$$N_2 \cdot \overline{Y}_{N_2}$$

Demostración Martos (1995):

Es una consecuencia de la proposición referente a la expresiones (2.8) y (2.9).

Proposición 2.3.2.3: El sesgo relativo de no respuesta, después de la imputación, es el mismo para el estimador de la media que del total poblacional e igual a la expresión:

$$t_{N_2} \cdot (W^* - W) / (1 - t_{N_2} \cdot (1 - W))$$
$$\text{donde, } W^* = \overline{Y}^*_{N_2} / \overline{Y}_{N_1} \text{ y } W = \overline{Y}_{N_2} / \overline{Y}_{N_1}$$

Demostración Martos (1995):
Se ha visto que los sesgos, después de la imputación, para la estimación de la media y del total [1] son:

$$sesgo\left(\overline{Y}^*_n\right) = t_{N_2} \cdot \left(\overline{Y}^*_{N_2} - \overline{Y}_{N_2}\right)$$
$$sesgo\left(\overline{Y}^*_n\right) = N_2 \cdot \left(\overline{Y}^*_{N_2} - \overline{Y}_{N_2}\right)$$

Utilizando la expresión para $\overline{Y}_N$ e $\overline{Y}_N$ obtenidas en las proposiciones 1.3.1.2 y 1.3.3.2, respectivamente, se obtiene:

$$\begin{aligned} &SR\left(\overline{Y}^*_n\right) = sesgo\left(\overline{Y}^*_n\right) / \overline{Y}_N = t_{N_2} \cdot \left(\overline{Y}^*_{N_2} - \overline{Y}_{N_2}\right) / \left(t_{N_1} \cdot \overline{Y}_{N_1} + t_{N_2} \cdot \overline{Y}_{N_2}\right) \\ &SR\left(Y^*_n\right) / Y_N = N_2 \cdot \left(\overline{Y}^*_{N_2} - \overline{Y}_{N_2}\right) / \left(N_1 \cdot \overline{Y}_{N_1} + N_2 \cdot \overline{Y}_{N_2}\right) = \\ &\quad \text{(multiplicando el numerador y el denominador por N)} \\ &= t_{N_2} \cdot \left(\overline{Y}^*_{N_2} - \overline{Y}_{N_2}\right) / \left(t_{N_1} \cdot \overline{Y}_{N_1} + t_{N_2} \cdot \overline{Y}_{N_2}\right) \\ &= SR\left(\overline{Y}^*_n\right) \end{aligned} \tag{2.11}$$

Operando en la expresión (2.11), es decir, dividiendo por $\overline{Y}_{N_1}$, se obtiene la expresión propuesta.
Proposición 2.3.2.4: La pérdida relativa o disminución, en el sesgo después de la imputación, con respecto al de no respuesta es:

$$\text{Para el estimador } \overline{Y}^*_n : 1 - \left| \frac{W^* - W}{1 - W} \right| \tag{2.12}$$

$$\text{Para el estimador } Y^*_n : 1 - \left| \frac{W^* - W}{W} \right| \tag{2.13}$$

donde, $W^* = \overline{Y}^*_{N_2} / \overline{Y}_{N_1}$ y $W = \overline{Y}_{N_2} / \overline{Y}_{N_1}$.
Demostración Martos (1995):
A través de las proposiciones 1.3.1.4 y 1.3.3.2 y lo expuesto en la estimación de la media poblacional y el total poblacional, podemos deducir que:
Para el estimador $\overline{Y}^*_n$:

$$\frac{\left|sesgo\left(\overline{Y}_n\right)\right| - \left|sesgo\left(\overline{Y}^*_n\right)\right|}{\left|sesgo\left(\overline{Y}_n\right)\right|} = \frac{|t_{N_2} \cdot (1 - W)| - |t_{N_2} \cdot (W^* - W)|}{|t_{N_2} \cdot (1 - W)|} = \frac{|1 - W| - |W^* - W|}{|1 - W|} \tag{2.14}$$

Para el estimador Y^*_n:

$$\frac{|sesgo\left(Y_n\right)| - |sesgo\left(Y^*_n\right)|}{|sesgo\left(Y_n\right)|} = \frac{|-1 \cdot t_{N_2} W| - |t_{N_2} \cdot (W^* - W)|}{|-1 \cdot t_{N_2} W|} = \frac{|W| - |W^* - W|}{|W|} \tag{2.15}$$

[1] Proposiciones 2.2.1.1 y 2.3.2.1.

2.2.3. En la estimación de la razón poblacional

Consideramos R_n^* un estimador de R_N tal que: $E\left(R_n^*\right) = \left(Y_{N_1} + Y_{N_2}^*\right)/Z_N$, siendo $Y_{N_2}^*$ una estimación de Y_{N_2} como consecuencia de una imputación y Z_N el total de la variable Z conocido.

Teniendo en cuenta la definición 1.3.4.1, se obtiene:

$$sesgo\left(R_n^*\right) = E\left(R_n^*\right) - R_N = C_{N_2} \cdot \left(R_{N_2}^* - R_{N_2}\right) \tag{2.16}$$

Una vez expuesto lo anterior, en esta sección se va a comparar éste sesgo (2.16) con el obtenido antes de la imputación, considerando que $E\left(R_n\right) = R_{N_1}$.

Proposición 2.3.3.1: La imputación reduce, en términos absolutos o a lo sumo, deja invariante el sesgo de no respuesta en la estimación de la razón poblacional.

Demostración Martos (1995):

Admitiendo la hipótesis de que $R_{N_2}^*$ está más cerca de la razón poblacional de los individuos que no responden de los que responden, se deduce que:

$$\left|R_{N_2}^* - R_{N_2}\right| \leq \left|R_{N_1} - R_{N_2}\right| \tag{2.17}$$

según la proposición 1.3.4.2, se tiene:

$$sesgo\left(R_n\right) = C_{N_2} \cdot \left(R_{N_1} - R_{N_2}\right), \text{ siendo } E\left(R_n\right) = R_{N_1} \tag{2.18}$$

Comparando las expresiones (2.16) y (2.18) y teniendo en cuenta la expresión (2.17), se puede deducir que:

$$\left|sesgo\left(R_n^*\right)\right| \leq \left|sesgo\left(R_n\right)\right| \tag{2.19}$$

Si no admitimos la hipótesis, es decir, suponemos que $R_{N_2}^*$ está más cerca de la razón poblacional de los individuos que responden, entonces al imputar el valor de $R_{N_2}^*$ y el valor de R_{N_1}, la expresión (2.17) será una igualdad y los sesgos iguales.

Corolario 2.3.3.2: Una cota superior del valor absoluto del sesgo después de la imputación en la estimación de la razón poblacional, es:

$$C_{N_2} \cdot \left(R_{N_1} - R_{N_2}\right) \tag{2.20}$$

Demostración Martos (1995):

Se deduce de la proposición 2.3.3.1 y las expresiones (2.18) y (2.19).

2.2.4. En la varianza muestral

Si se considera que se ha realizado una imputación, el estimador de la media poblacional vendrá dado por la siguiente expresión:

$$\overline{Y}_n = n_1/n \cdot \overline{Y}_{n_1} + n_2/n \cdot \overline{Y}_{n_2}^*, \text{ siendo } \overline{Y}_{n_2}^* = \sum X_1^*/n_2 \tag{2.21}$$

Proposición 2.3.4.1: La varianza ajustada poblacional, después de una imputación, muestra la siguiente expresión:

$$S_n^2 = 1/\left(n-1\right) \cdot \left[\left(n-1\right) \cdot S_{n_1}^2 + n_1 \cdot \overline{Y}_{n_1}^2 + n_2 \cdot A - n \cdot \overline{Y}_n^2\right]$$

$$\text{con } A = \sum_{i=1}^{n_2} X_1^{*2}/n_2$$

Demostración Martos (1995):

La varianza ajustada muestral de los individuos que no responden, $S^2_{n_2}$, una vez realizada la imputación, muestra la siguiente expresión:

$$\begin{aligned} S^2_{n_2} =& 1/(n_2-1)\cdot\sum_{i=1}^{n_2}\left(X^*_1-\overline{Y}^*_{n_2}\right)^2 = \\ & 1/(n_2-1)\cdot\left(\sum X^{*2}_1-2\cdot\sum X^*_1\cdot\overline{Y}^*_{n_2}+\sum\overline{Y}^{*2}_{n_2}\right) = \\ & 1/(n_2-1)\cdot\left(n_2\cdot A-n_2\cdot\overline{Y}^{*2}_{n_2}\right) \end{aligned} \tag{2.22}$$

teniendo en cuenta que $A=\sum_{i=1}^{n_2} X^{*2}_1/n_2$ y $\overline{Y}^*_{n_2}=\sum X^*_1/n_2$.

De la expresión (2.21) se deduce que:

$$\overline{Y}^*_{n_2}=n/n_2\cdot\overline{Y}_n-n_1/n_2\cdot\overline{Y}_{n_1} \tag{2.23}$$

Sustituyendo la expresión (2.22) en la expresión obtenida en el corolario 1.3.2.5, obtenemos:

$$(n-1)\cdot S^2_n=(n_1-1)\cdot S^2_{n_1}+n_1\cdot\left(\overline{Y}_{n_1}-\overline{Y}_n\right)^2+n_2\cdot A-n_2\cdot\overline{Y}^{*2}_{n_2}+n_2\cdot\left(\overline{Y}^*_{n_2}-\overline{Y}_n\right)^2$$

sustituyendo la expresión (2.23) en ésta expresión y simplificando se obtiene la expresión propuesta.

Corolario 2.3.4.2: Una expresión simplificada de la varianza ajustada muestral, después de una imputación es:

$$S^2_n=n_1/n\cdot\left(S^2_{n_1}+\overline{Y}^2_{n_1}\right)+n_2/n\cdot A-\overline{Y}^2_n$$

Demostración Martos (1995):

Tomando como aproximación de $n-1$, n y de n_1-1, n_1 en la expresión obtenida anteriormente, se consigue la expresión propuesta.

2.3. Métodos de clasificación de unidades antes de la imputación

Conviene definir las reglas de construcción de los valores imputados y_k. Las técnicas expuestas en este capítulo, utilizan la clasificación en grupos, y luego imputan dentro de cada grupo las unidades con no respuesta. Existen diferentes procedimientos que se detallan a continuación:

2.3.1. Postestratificación

La definición extraída de Martos (1995) define el proceso de postestratificación de la siguiente forma:

Dada una muestra aleatoria simple de extensión n, se procede a clasificar las unidades que la componen formando L estratos en la muestra. Los tamaños de los estratos, N_1, a nivel poblacional, se obtienen de manera bastante exacta a partir de las estadísticas oficiales.

En lugar de la media muestral $\overline{Y}_n$, se usa la estimación: $\overline{Y}_n=\sum_{i=1}^{L} W_1\cdot\overline{Y}_1$, donde Y_1 es la media del estrato i y $W_1=N_1/N$.

En palabras de Cochran (1977), este método es casi tan preciso como el muestreo estratificado siempre que:

- La muestra sea razonablemente grande, algunos autores como Cochran (1977), Cruz (1990) entre otros, indican que una muestra razonablemente grande es aquella que tiene al menos 20 unidades o más por estrato.

- Los efectos de los errores en las ponderaciones W_1 tienen que poder ignorarse.

La eficiencia de la postestratificación fue evaluada por Thomsen et al (1983) los cuáles obtuvieron una expresión del sesgo de no respuesta, con la siguiente forma:

$$sesgo = 1/\bar{t} \cdot \sum^{L} \overline{Y}_{N_{i1}} \cdot W_i \cdot (t_{n_{i1}} - \bar{t}) + \sum^{L} \left(\overline{Y}_{N_{i1}} - \overline{Y}_{N_{i2}}\right) \cdot W_i \cdot (1 - t_{n_{i1}})$$

donde $t_{n_{i1}}$ es la tasa de respuesta en el estrato i y $\bar{t} = \sum^{L} W_i \cdot t_{n_{i1}}$, además $\overline{Y}_{N_{i1}}$ y $\overline{Y}_{N_{i2}}$ son las medias poblacionales de las unidades que responden y no responden respectivamente. La primera componente del sesgo puede ser estimada por la muestra, pero no la segunda al ser desconocida la media muestral de las unidades que no responden.

2.3.2. Áreas de balance y clases de ponderaciones

Un área de balance se define como una área geográfica que puede ser a su vez un estrato individual, un grupo de estratos, una provincia, una unidad muestral primaria o un conglomerado (Plateck, 1986). Una clase de ponderación es una agrupación de unidades muestrales con características comunes (tipos de vivienda, grupos especiales de renta,...) sin tener en cuenta la localización geográfica o el área en el marco muestral. La definición de una u otra es preferible en la etapa de planificación antes que en la etapa de procesamiento, dado que eligiendo la primera etapa se pueden realizar ajustes agrupando áreas si es necesario, bien porque la tasa de no respuesta sea baja o bien porque la muestra sea demasiado pequeña.

El esquema para cualquiera de estas dos técnicas de clasificación, es el siguiente:

$$Y = \sum_{b} Y_b \text{ estima el total de una característica.}$$

Para un área de balance b, se tiene que:

$$Y_b = \sum_{i}^{n_{ib}} W_i \cdot X_1 \cdot 1/\pi_i + \sum_{i+n_{1b}}^{n_{1b+u}} W_j \cdot X_j^* \cdot 1/\pi_j$$

donde:

- π_1 y π_j son las probabilidades de inclusión.
- X_1 y X_j^* son unidades que contestan y las que no son imputadas.
- n_{1b} unidades que contestan.
- u unidades que han sido imputadas, dentro de las que no contestan.

2.3.3. Árboles de clasificación y regresión

Los árboles de clasificación y regresión generan subgrupos de población que contienen elementos homogéneos entre ellos y heterogéneos entre distintos subgrupos con respecto a la variable a discriminar, que será la que se desea imputar. Dada una variable de respuesta categórica o continua cuyo valor es faltante y una serie de variables explicativas, el modelo de imputación basado en árboles emplea los registros con valor conocido en la variable respuesta y a partir de los registros se construye el árbol en función de las variables explicativas. Los nodos terminales son tratados como clases de imputación. Cada registro con valor faltante en la variable respuesta llega a un determinado nodo terminal en función de los valores en las variables explicativas.

2.4. Modelos de generación de datos faltantes

El patrón de la pérdida de los datos faltantes es un punto a considerar en la imputación de la no respuesta; de ello depende el método seleccionado de imputación. La mayoría de los estadísticos de encuestas advierten de las

diferencias que existen entre la falta de respuesta total y la no respuesta parcial. En Medina et al (2007), se establece que la no respuesta total se corrige eliminando las observaciones y ajustando los factores de expansión para que las unidades de la muestra se estimen sin sesgo. Para corregir la omisión parcial, lo común es aplicar el procedimiento Hot Deck o el método de promedios. Los distintos procedimientos de imputación establecen supuestos acerca del patrón de comportamiento de los datos omitidos y Rubin (1977) establece la siguiente clasificación:

- MCAR (Missing Completely At Random).
- MAR (Missing At Random).
- NMAR (Not Missing At Random).

2.4.1. MCAR (Missing Completely At Random)

Los datos faltantes siguen un patrón aleatorio; la probabilidad de que una respuesta a una variable sea dato faltante es independiente tanto del valor de esta variable como del valor de otras variables del conjunto de datos. La ausencia de información no procede de ninguna variable presente en la matriz de datos, ya que la base de datos se interpreta como una matriz en donde los renglones son las unidades de observación y las columnas representan a las variables de interés (Rubin, 1977). Hay que señalar, que en la práctica, la hipótesis de este supuesto no siempre es satisfactoria porque la falta de respuesta se asocia a características personales.

2.4.2. MAR (Missing At Random)

La probabilidad de que una respuesta sea dato faltante es independiente de los valores de la misma variable pero es dependiente de los valores de otras variables del conjunto de datos. En suma, la ausencia de datos está asociada a variables presentes en la matriz de datos. Por ejemplo, este supuesto, es usual en las encuestas de hogares que estudian el nivel de vida, donde la falta de respuesta en la variable ingreso en las familias con mayor renta está correlacionada con sus ingresos y por lo tanto no se cumple el supuesto. En cambio, cuando la omisión ocurre en la categoría ocupacional, la falta de información no necesariamente está correlacionada con el nivel de ingresos de las personas y por lo tanto sí se cumple el supuesto.

2.4.3. NMAR (Not Missing At Random)

La probabilidad de que una respuesta a una variable sea dato faltante es dependiente de los valores de la variable. Por ejemplo, en el estudio de las variables peso y edad; los sujetos con mayores valores de peso tienen un porcentaje de datos faltantes más elevado en esta variable para aquellos individuos con la misma edad. Este tipo de dato faltante también se denomina no ignorable.

2.5. Procedimientos tradicionales de imputación

Antes de abordar el tema de los métodos de imputación, en el cual se profundizará en el siguiente capítulo, notemos que existen distintos procedimientos para sustituir la falta de observaciones en un conjunto de datos, entre los que encontramos:

2.5.1. Análisis con datos completos (Litwise o case deletion LD)

El Litwise es el método más utilizado. Se trabaja por defecto sólo con información completa, a pesar de que no es la práctica más apropiada, ya que genera sesgos en los coeficientes de asociación y correlación Martos (1995), Cruz (1990), Medina et al (2007). Se asume que la submuestra de datos excluídos tiene las mismas características que los datos completos y que la falta de respuesta se generó de manera aleatoria. Las unidades que fueron elegidas con un procedimiento aleatorio no pueden ser ignoradas ni en el tratamiento de los datos ni en el cálculo de las estimaciones y sus errores. Si la eliminación de registros no se acompaña con el ajuste de los factores de expansión se podrían invalidar las conclusiones por la pérdida de información; sobre todo cuando el número de variables es elevado.

Ventajas

- Facilidad de implementación.
- Posibilidad de comparar los estadísticos univariantes.

Desventajas

- Pérdida de información cuando el número de variables es elevado, dado que puede generar sesgos en las estimaciones de los parámetros.

2.5.2. Análisis con los datos disponibles (pairwise deletion)

El procedimiento AC (Available case), utiliza distintos tamaños de muestra; por ello se le conoce como pairwise deletion o pairwise inclusion. Hace uso de toda la información disponible sin efectuar ningún tipo de correción en los factores de expansión. Las observaciones que no tienen datos se eliminan.

Ventajas

- Cuando se le compara con el litwise, tiene la ventaja de que hace uso de toda la información disponible.

Desventajas

- La mezcla de tamaños de muestra debilita su aplicación.

2.5.3. Reponderación

Este método se aplica cuando todos los registros de la unidad tienen todos los campos omitidos. Se intenta mejorar la precisión de las estimaciones y reducir el sesgo de los individuos que no responden. Requiere de información auxiliar tanto de los individuos que responden como de los que no responden.

Capítulo 3

Métodos de imputación

3.1. Introducción

Los métodos de imputación son una posible solución a la falta de respuesta parcial. Se denomina imputación al proceso que utiliza la información procedente de la muestra para asignar un valor a aquellas variables con registros con valor ausente, ya sea porque se carece de información o porque se detecta que algunos de los valores obtenidos no se corresponden con el comportamiento esperado. La razón principal por la cual los investigadores realizan la imputación es para obtener un conjunto de datos completo y consistente al cual se le aplica inferencia estadística. En la fase de imputación se debe escoger cuidadosamente las variables objetivo, auxiliares, los criterios de imputación y el método preciso de imputación.

Algunos criterios de imputación que se pueden considerar en la fase de imputación son: el mantenimiento de la distribución, el mantenimiento de las correlaciones entre las variables y la consistencia. Los distintos métodos de imputación de datos se diferencian en dos tipos generales: los que utilizan los valores existentes en algún registro de la variable a imputar o relacionados con ellos y los que utilizan un conjunto de campos - variables de control- de la encuesta; en todos los casos hay un registro donante. Otros criterios de agrupación permiten clasificar a los métodos en simples y múltiples, y también en determinísticos y aleatorios.

Imputación simple y múltiple

- La imputación simple consiste en asignar un valor por cada valor faltante basándose en el valor de la propia variable o de otras variables, generando una base de datos completa.
- La imputación múltiple consiste en asignar a cada valor faltante varios valores m, generando m conjuntos de datos completos. En cada conjunto de datos completo se estiman los parámetros de interés y posteriormente se combinan los resultados obtenidos.

Determinísticos y aleatorios

- Los métodos de imputación determinísticos son aquellos que producen las mismas respuestas cuando se repite la imputación en varias unidades bajo las mismas condiciones.
- Los métodos de imputación estocásticos o aleatorios son aquellos que producen resultados diferentes cuando se repite el método de imputación bajo las mismas condiciones para una unidad.

3.2. Evolución histórica

Los métodos de imputación de datos durante la década de los setenta de siglo pasado trataban de identificar y sustituir los registros sin información. En este contexto, los procedimientos Hot Deck, en sus diferentes variantes, se aplicaban principalmente para suplir información en censos y encuestas y los métodos de imputación de ajuste a través de promedios y Cold Deck eran frecuentemente utilizados (Figueredo et al 2000).

En el año 1976, Rubin propuso un marco conceptual para el análisis de datos faltantes, basado principalmente en métodos de inferencia estadística. La aparición posterior del algoritmo Expectation Maximization (EM) permitió

generar estimadores robustos a partir de la aplicación del método de máxima verosimilitud (MV) (Dempster et al 1977), en donde las observaciones faltantes se consideran variables aleatorias y los datos imputados se generan sin necesidad de ajustar modelos.

Posteriormente, Rubin en 1987 introdujo el concepto de imputación múltiple (IM), sustentado en la premisa de que cada dato debe ser reemplazado a partir de $m > 1$ simulaciones. La aplicación de esta técnica se facilitó con los avances computacionales y el desarrollo de los métodos bayesianos de simulación que aparecieron a finales de los años ochenta del siglo pasado (Schafer, 1997). A partir de la introducción de estos métodos, se propició el uso intensivo de diferentes algoritmos, y las distintas aplicaciones de IM y MV se empezaron a incluir en paquetes estadísticos comerciales o de acceso gratuito.

Durante la década de los noventa del siglo pasado, se registraron avances notables. El método de reponderación, históricamente utilizado por los estadísticos de encuestas, se modificó con el próposito de utilizarlo en la imputación de datos en modelos de regresión (Ibrahim, 1990). También se desarrollaron técnicas en el campo de la bioestadística, con el propósito de resolver situaciones en que los datos faltantes dependían de las características de la población; el objetivo de estas técnicas era disponer de algoritmos apropiados para analizar datos clínicos; en este tipo de investigaciones es común que el número de pacientes se reduzca, ya que los estudios suelen concluir con menos personas de las que iniciaron (Little, 1995).

En resumen, la literatura considera que el desarrollo del método IM ha reducido o zanjado el debate respecto a la mejor forma de imputar datos omitidos. Sin embargo, los investigadores de encuestas han demostrado que las soluciones existentes no tienen en cuenta el diseño de la muestra, ni las probabilidades de selección de las unidades de análisis, cuando los datos provienen de muestras complejas, ya que este procedimiento introduce sesgos en las estimaciones.

3.3. Objetivos teóricos y prácticos de la imputación

El objetivo de la imputación con o sin datos omitidos es el análisis estadístico a través de la inferencia. La imputación no trata sólo de obtener estimadores insesgados (o con sesgo pequeño) y de varianza mínima, o de ajustar modelos para sustituir de cualquier manera la información faltante. La imputación debe considerarse parte del proceso de investigación con el propósito de llegar a conclusiones sustentadas en evidencia empírica sólida (Medina et al 2007). En este sentido autores como Cruz, Rubin o Little, entre otros, afirman que las bondades de los procedimientos de imputación no deben valorase por el sólo hecho de que permiten completar información para ajustar modelos y probar hipótesis.

En referencia a los objetivos prácticos de la imputación, los primeros criterios fueron establecidos por las publiciaciones de Neyman et al (1933, 1937), y guardaban relación con el error cuadrático medio (ECM) y no sólo con el sesgo del estimador. Si la variable analizada (S) contienen datos faltantes, esta situación debe tenerse en cuenta en el proceso de construcción del estimador (la mayoría de los métodos de imputación no tienen en cuenta este hecho), por lo que generan estimadores sesgados. Si la imputación que se hace es adecuada, el estimador $\widehat{S}$ será cercano al verdadero valor del párametro S en las sucesivas muestras repetidas. De esta forma, se logra minimizar el sesgo, la varianza y la desviación estándar de $\widehat{S}$.

Cuando la falta de datos ocurre por razones ajenas al investigador, es necesario establecer supuestos acerca de las causas que generaron las omisiones, contrastando la veracidad de las hipótesis con el comportamiento observado en los datos. A pesar de que los criterios estadísticos son fundamentales para la elección del método de imputación, es necesario tener claridad sobre el uso que se hará de la información. Si, por ejemplo, nuestro objetivo es conocer los determinantes del nivel de vida de las familias españolas, es probable que distintos métodos sean equivalentes desde la visión estadística, pero no obstante hay que tener presente que pequeñas diferencias en el ingreso de las familias españolas pueden generar cambios significativos en el nivel de vida de las familias españolas.

3.4. Ventajas y desventajas de la imputación

Los métodos de imputación nos permiten utilizar la inferencia estadística para obtener estimadores insesgados (o con pequeño sesgo) y de varianza mínima a través del ajuste de modelos que sustituyen la información faltante. Pero la imputación tiene ventajas y desventajas. Las ventajas de la imputación son que sus procedimientos permiten obtener un conjunto de datos completo posibilitando la redución del sesgo debido a la no respuesta. En este sentido, la imputación opera sobre los datos de forma que los resultados que se obtienen de los diferentes métodos suelen llevar a un análisis consistente.

Las desventajas de la imputación son que la futura estimación obtenida del análisis no distinga entre las imputaciones y los datos reales. Los valores imputados pueden conducir a buenas estimaciones pero no son datos reales que aseguren una mejora en el sesgo respecto del sistema de datos incompletos. Si el método de imputación no es el adecuado, posiblemente aumente el sesgo y se sobreestime la varianza, al generar datos imputados inconsistentes que llevarían a la interpretación errónea de los resultados.

3.5. Diferentes tipos de técnicas de imputación

Varios estudios, como Goicochea (2002) y Platek (1986), indican que las técnicas de imputación se clasifican de la siguiente manera:

1. **Técnicas fundamentadas en información externa**: la información está basada en variables relacionadas con una encuesta perteneciente a otras bases de datos o reglas previas. Entre este tipo de técnicas podemos encontrar:
   - **Métodos deductivos**: los datos faltantes se deducen con cierto grado de certidumbre de otros registros completos del mismo caso, siguiendo algunas reglas específicas.
   - **Tablas Look-up**: se hace uso de una tabla con información relacionada, como fuente de datos externa para imputar los datos faltantes.
2. **Técnicas determinísticas**: al repetir la imputación en varias unidades bajo las mismas condiciones se producirán las mismas respuestas. Entre este tipo de técnicas podemos encontrar:
   - **Imputación de la media**: se llena el vacío del dato faltante de cada variable con la media de los registros no faltantes en caso de variables cuantitativas, con la moda en caso de variables cualitativas. La desventaja es que puede ocasionar la modificación de las distribuciones de la variable reduciendo su varianza y no conservar la relación entre variables. La ventaja es la facilidad de aplicación.
   - **Imputación de media de clases**: las respuestas de cada variable son agrupadas en clases disjuntas con diferentes medias, y a cada registro faltante se le imputará con la media respectiva de su grupo. Las desventajas son las mismas que en el caso anterior, pero en menor proporción por estar agrupadas. La ventaja sigue siendo la facilidad de aplicación.
   - **Imputación por regresión**: se ajusta un modelo lineal que describa a Y, variable a imputar, para un conjunto X de variables auxiliares que se deben disponer. Resuelve el problema de la distorsión de la distribución de la variable a imputar, pero puede crear inconsistencias dentro de la base de datos, pues podría obtener valores imposibles.
   - **Emparejamiento media**: el valor de Y estimado es comparado con casos completos, y el caso más cercano correspondiente provee el valor imputado de Y.
   - **Imputación por el vecino más cercano**: se identifica la distancia entre la variable a imputar Y, y cada una de las unidades restantes o variables auxiliares mediante alguna medida de distancia, entonces se determina la unidad más cercana a Y, usando el valor de esta unidad cercana para imputar el dato faltante.
   - **Algoritmo EM (Expectation Maximization)**: basado en la función de máxima verosimilitud, permite obtener estimaciones máximo verosímiles (MV) de los parámetros cuando hay datos incompletos con unas estructuras determinadas. Resuelve de forma iterativa el cálculo del estimador máximo verosímil mediante dos pasos en cada iteración (Little et al 1987). Este algoritmo resuelve un amplio rango de problemas, incluso los no usuales que surgen de la pérdida o información faltante, como la estimación de los componentes de la varianza.
   - **Redes Neuronales**: son sistemas de información procesados, que reconocen patrones de los datos sin algún valor perdido para aplicarlo a la información a imputar. Estas redes son más usadas para variables cualitativas que cuantitativas, siendo más adecuadas cuando la distribución no es lineal. No es aconsejable en caso de registros atípicos que distorsionan la red. Son costosos y requieren de capacitación del analista así como de software adecuado.
   - **Modelos de series de tiempo**: el problema se reduce a una situación en la cual hay una serie de tiempo, donde algunas observaciones están perdidas; se hace un uso óptimo de las interrelaciones entre observaciones en cada serie de tiempo, mediante el uso de un modelo adecuado.

3. **Técnicas aleatorias o estocásticas**: se repite el método de imputación bajo las mismas condiciones para una unidad y producen resultados diferentes. Dentro de este tipo de técnicas destacan:

   - **Imputación aleatoria de un caso seleccionado**: para cada caso con una celda faltante, se selecciona un donante aleatoriamente para ser asignado al dato faltante.
   - **Imputación de un caso seleccionado entre clases**:se realiza de igual forma que para el caso anterior, pero se lleva a cabo dentro de clases previamente creadas.
   - **Imputación secuencial Hot Deck**: cada caso es procesado secuencialmente. Si el primer registro tiene un dato faltante, éste es reemplazado por un valor inicial para imputar, pudiendo ser obtenido de información externa. Si el valor no está perdido, éste será el valor inicial y es usado para imputar el subsiguiente dato faltante. Una de las desventajas es que si el primer registro está perdido, se necesita un valor inicial, generalmente obtenido de manera aleatoria. Además si se imputan muchos registros se tiende a emplear el mismo registro donante, que conlleva la pérdida de precisión en las estimaciones.
   - **Imputación jerárquica Hot Deck**: similar al método secuencial anterior, se organiza dentro de clases que usan variables auxiliares en forma de una estructura jerárquica. Si el donante no es encontrado en un nivel de clasificación, las clases pueden ser colapsadas en grupos más amplios hasta que el donante sea encontrado.
   - **Imputación por regresión aleatoria**: se realiza primero un procedimiento de regresión; luego un término residual es adicionado para imputar los valores de Y. Este término de error puede ser obtenido de diferentes maneras, una de ellas es a través de los residuos del modelo de regresión, generado con registros completos, eligiendo uno de éstos residuos aleatoriamente.
   - **Imputación por regresión logística**: similar a la técnica anterior, pero para imputar variables binarias.

3.6. Métodos heurísticos

En Gómez et al (2006) se presentan los métodos heurísticos como una de las soluciones más habituales utilizadas en la práctica ante una matriz de datos con valores perdidos. Estos métodos pueden parecer soluciones razonables, cuando la información faltante es pequeña, pero generalmente no son procedimientos aceptables.

3.6.1. Análisis de casos completos

Dada una muestra X de K variables y n casos, supongamos que $n_{per} < n$ de estos casos presentan al menos un valor perdido para alguna de las k variables. Mediante este método se descartan los n_{per} casos y sólo se aplica la técnica o análisis sobre aquellos con valores observados para todas las variables. De este modo se pasa a trabajar con una muestra de datos completa de tamaño $n - n_{per}$. Las consecuencias de este método dependerán en gran medida de la cantidad de información que se pierda al descartar los casos con información faltante, del mecanismo según el cual faltan los datos y de la relación entre los casos completos e incompletos. La pérdida de información relevante se traducirá en sesgo y falta de precisión de las estimaciones si los casos faltantes no son una muestra aleatoria de la muestra completa, es decir, si no se verifica las hipótesis MCAR. Este método destaca por su simplicidad y por el hecho de que todos los estadísticos se calculan utilizando el mismo tamaño muestral, facilitando su comparación.

3.6.2. Análisis de casos disponibles

Si para el iésimo caso de una muestra se observan p de las K variables, al aplicar análisis de casos completos estamos perdiendo la información que sobre las $K - p$ variables restantes contiene dicho caso. Una alternativa natural es utilizar para cada cálculo toda la información disponible en la muestra. Por ejemplo, a la hora de calcular la media o varianza de las variables X_i y X_j se utilizarán los n_i y n_j datos disponibles sobre cada una de ellas respectivamente. O bien, para calcular la covarianza entre las dos variables X_i y X_j se considerán los casos h para los que los pares $\left(X_{hi}, X_{hj}\right)$ son observados, esto implicará trabajar con distintos tamaños muestrales e incluso tener que combinarlos en el cálculo de un mismo estadístico.

3.7. Métodos de imputación simple

Los métodos de imputación simple pretenden solucionar el problema de los valores perdidos sustituyendo los mismos por los valores estimados a partir de la información suministrada por la muestra. De esta manera, se consige una matriz completa sobre la que realizar los análisis. Es precisamente la forma de estimar o predecir los valores perdidos la que diferencia unos métodos de otros.

3.7.1. Imputación por media

La imputación por media es un método que fue presentado por primera vez por Wilks en el año 1932, es posiblemente uno de los procedimientos de imputación más antiguo y más sencillo. Consiste en que los valores faltantes de una variable se sustituyen mediante la media de las unidades observadas en esa variable. La imputación por media tiene varias vertientes:

Imputación por el método de medias no condicionadas

La sustitución de datos utilizando promedios es una vieja práctica entre investigadores de diversas disciplinas, a pesar de que por sus limitaciones teóricas no se considere un procedimiento apropiado. EL método de imputación de medias no condicionadas consiste en estimar la media de los valores observados, es decir, si y_{ij} es el valor de la variable Y_j para la unidad i, el método de imputación por medias incondicional trata de estimar los valores faltantes y_{ij} por $\overline{y}_j^{(i)}$, la media de los valores observados de Y_j. En su aplicación se asume que los datos faltantes siguen un patrón MCAR y ha sido ampliamente documentado que su aplicación afecta a la distribución de probabilidad de la variable imputada, ya que atenúa la correlación con el resto de las variables y subestima la varianza.

Imputación por media condicional

Imputa medias condicionadas a valores observados. Uno de los métodos comunes de imputación por media es aquel que agrupa los valores observados y no observados en estratos e imputa los valores faltantes por las medias de los valores observados en los mismos estratos.

Imputación por medias condicionadas para datos agrupados

Es el método introducido por Acock y Demos en 1994, consiste en formar categorías a partir de covariables correlacionadas con la variable de interés, y en imputar los datos omitidos con observaciones provenientes de la submuestra con características comunes. Se asume que los datos faltantes siguen un patrón MCAR y que existirán tantos promedios como categorías se formen, lo cual contribuye a reducir los sesgos en cada estrato o categoría pero en ningún caso los elimina. En la medida que la falta de información por categoría sea baja, los sesgos disminuyen pero no desaparecen.

3.7.2. Imputación con variables ficticias

Cohen en 1983, conjuntamente con otros autores, popularizaron esta metodología. Consiste en crear una variable indicador para identificar las observaciones con datos faltantes. Supongamos que la variable predictora y es la condición de ocupación (1 si no conoce la condición de ocupación y 0 si la persona manifiesta estar ocupada), entonces a las personas con datos faltantes se les asigna la media de la variable condición de ocupación. Si se estima el modelo de regresión $y = \alpha + \beta Z$, el valor de β sería el mismo que se obtendría si se utilizará el procedimiento litwise, y β daría cuenta de la diferencia entre la variable indicador Z y la media de y. Al igual que el procedimiento LD, la imputación por variables ficticias genera inconsistencias en la capacidad explicativa de los estimadores.

Los autores afirma que esta forma de proceder altera la interpretación de los coeficientes de regresión. Dado que en el modelo original $E(Y) = \beta_0 + \beta_1 X$, β_0 y β_1 representan la ordenada en el origen y la pendiente de la recta de regresión respectivamente, en tanto que en el modelo extendido $E(Y) = \beta_0 + \beta_1 + \beta_2 Z$, β_0 y β_1 equivalen a la ordenada en el origen y la pendiente de los individuos que respondieron, en tanto que $\beta_0 + \beta_2$ es la media de la variable Y, entre las observaciones que no tuvieron respuesta.

3.7.3. Imputación deductiva

Es un método de imputación que se aplica en situaciones en que las respuestas que faltan se pueden deducir del resto de la información proveniente del conunto de datos, es decir los valores se asignan mediante relaciones lógicas entre las variables. Por ejemplo, una imputación deductiva sería si falta el sexo del encuestado y la persona tiene nombre masculino, se puede deducir que el sexo es masculino.

3.7.4. Imputación por sustitución

Con esta técnica se pretende sustituir la no respuesta por un valor conocido de las unidades o relacionado con ellas. Comprende tres subtécnicas (Martos, 1995):

Sustitución por la media

Sustituye las unidades con no respuesta por la media de las unidades que responden.

Sustitución por duplicación

Sustiuye las unidades con no respuesta por un subconjunto, aleatoriamente seleccionado, de unidades muestrales con respuesta. El procedimiento se puede expresar de la siguiente manera: si n_1 y n_2 son el número de unidades que responden y no responden respectivamente en la muestra, los diferentes subconjuntos que podemos formar para sustituir las unidades que no responden por unidades que responden es:

Para el caso en que $n_1 \geq n_2$: Cn_1,n_2 donde C indica las combinaciones.

Para el caso en que $n_1 < n_2$, cada unidad de los que responden deberá ser usada el menor número posible de veces, de la siguiente manera:

Se determina k, tal que:

$n_2 = K \cdot n_1 + m$, donde K es un número entero y $m < n_1$.

con lo cual:

$n_1 - m$ unidades que responden se utilizarán K veces y m unidades que responden se utilizarán $K+1$ veces, es decir, $n_2 = K \cdot (n_1 - m) + (K+1) \cdot m$.

Sustitución utilizando variables auxiliares

Se sustituye las unidades con no respuesta por datos históricos o por datos que provienen de fuentes externas: administrativas, de otras encuestas, del censo, etc.

3.7.5. Imputación Cold Deck

Este procedimiento asigna los valores faltantes a partir de una encuesta anterior o de otras informaciones. El método Cold Deck asigna a los campos a imputar de todos los registros candidatos los valores del registro donante correspondiente al mismo estrato. La desventaja principal de este método es que la calidad de las estimaciones de los resultados dependerá de la calidad de la información externa disponible. Este método propició que a partir de esta idea se originará un nuevo procedimiento llamado Hot Deck, que describimos en la siguiente sección, donde a diferencia del método Cold Deck se asigna un valor existente de la muestra al dato faltante.

3.7.6. Imputación Hot Deck

El método de imputación Hot Deck, es un proceso de duplicación. Duplica un valor ya existente en la muestra para reemplazarlo, cuando hay un dato faltante. Originalmente se desarrolló para tratar datos faltantes de encuestas poblacionales. El procedimiento Hot Deck clasifica los encuestados en factores basados en características demográficas y además no necesita que las variables sean categóricas, lo que lleva a que preserve generalmente las distribuciones univariantes de los datos. Por otro lado, no atenúa la variabilidad de los datos rellenados como otros métodos de imputación. Sin embargo, este método no es muy apropiado para estimar medidas de asociación dado que puede producir estimaciones sesgadas de las correlaciones y los coeficientes de regresión. Dentro de este procedimiento existen diferentes variantes en función del tipo de variable a imputar, que presentamos a continuación:

Imputación aleatoria Hot Deck

Este procedimiento usa el muestreo aleatorio simple, asignando aleatoriamente un valor recogido en la muestra de la variable a imputar. Considera la distribución de los individuos que responden pero no considera si es factible la imputación ni la correlación con otras variables.

La imputación Hot Deck es un método estocástico que por lo general tiene un proceso de clasificación asociado. En dicho proceso todas las unidades de la muestra están clasificadas en grupos disjuntos de forma que las unidades sean lo más homógeneas posibles dentro de los grupos. A cada dato faltante, se le asigna un valor del mismo grupo, así la suposición que se está utilizando es que dentro de cada grupo de clasificación la no respuesta sigue la misma distribución que los individuos que responden. Además en este procedimiento las variables de clasificación deben estar correladas con los datos faltantes y con los valores de los individuos que responden. Si estos supuestos no se cumplen el método de imputación Hot Deck puede llevar a resultados equivocados. Teniendo en cuenta el procedimiento de este método podemos encontrar otras variantes como:

Imputación aleatoria Hot Deck por grupos

Este procedimiento es un método estocástico e imputa con valores recogidos de la muestra perteneciente al grupo.

Imputación Hot Deck secuencial

Este método se usa cuando la muestra tiene algún tipo de orden dentro de cada grupo de clasificación. Cada valor faltante se reemplaza por el registro sin valor faltante, perteneciente al mismo grupo e inmediatamente anterior a él; si el primer registro tiene un dato faltante, este es reemplazado por un valor inicial que puede obternerse mediante información externa. Las desventajas de este método se describen a continuación:

- Si es necesario imputar muchos registros se tiende a emplear el mismo valor, llevando implícita una pérdida de precisión en las estimaciones.
- Es díficil estudiar la precisión de las estimaciones.

Imputación Hot Deck: Vecino más cercano

Es un método de imputación determinístico basado en un procedimiento no paramétrico con la suposición de que los individuos cercanos en un mismo espacio tienen características similares. Para aplicar este método, se requiere definir una medida de distancia. Por ejemplo, si consideramos $x_i = (x_{i1}, ..., x_{ik})^T$ los valores de las k covariables para la unidad i en la cual el valor y_i es faltante, clasificando estas variables por grupos, una métrica adecuada sería:

$$d(i,j) = \begin{cases} 0 \text{ si i, j están en el mismo grupo} \\ \\ 1 \text{ si i,j están en diferentes grupos} \end{cases}$$

Otras posibles distancias podrían hayarse utilizando las siguientes métricas:

- Máxima desviación: $d(i,j) = \max_k \left| x_{ik} - x_{jk} \right|$
- Distancia de Mahalanobis: $d(i,j) = (x_i - x_j)^T S_{xx}^{-1} (x_i - x_j)$, donde S_{xx}^{-1} es una estimación de la matriz de covarianzas de x_i.
- Distancia euclídea: $d(i,j) = \sqrt{\sum_{K=1}^{K} \left(x_{ik} - x_{jk} \right)^2}$

3.7.7. Last Observation Carried Forward

Es una técnica que requiere de datos longitudinales. La técnica Last Observation Carrie Forward imputa medidas repetidas con la observación que le precede. Esta estrategia se aplica cuando hay casos que tienen datos

perdidos permanentemente o de forma intermitente. Por otro lado, esta técnica también asume que los valores no cambian significativamente después de la última medida observada o durante el período intermitente donde faltan valores.

3.7.8. Imputación por regresión

También denominada Método Buck. La imputación por regresión fue propuesta por primera vez por Buck en el año 1960. En dicho método, se emplean modelos de regresión para imputar información en la variable respuesta Y a partir de covariables $(X_1,...,X_k)$ correlacionadas con la variable respuesta Y. La imputación por regresión consiste en eliminar las observaciones con datos incompletos y ajustar la ecuación de la regresión para predecir los valores faltantes. Por ejemplo: sea n el tamaño muestral, consideramos a la variable Y cómo aquella que representa los primeros r valores observados y $n-r$ faltantes. Suponemos que las K variables, $X=(X_1,....,X_K)$, no presenta valores perdidos. Para el caso i tenemos que si el valor y_i no se observa, este valor faltante es imputado mediante el siguiente modelo de regresión:

$$g\{E(Y)\}=X\beta, \qquad Y \backsim F$$

donde g es la función link y F es la función de distribución.

La imputación por regresión permite obtener diferentes modelos, depediendo de cómo sea la variable Y, su distribución y la función g. A continuación mostramos algunos posibles modelos:

Cuando Y es una variable continua

Cuando Y es una variable continua, si consideramos que g es la identidad e Y sigue una distribución normal, entonces se obtiene el siguiente modelo de regresión lineal:

$$E(Y)=X\beta, \qquad Y \backsim \text{Normal}$$

Cuando la variable Y es continua existen dos maneras de proceder en la imputación en regresión:

1. **Imputación mediante regresión determinística**: En el caso de la imputación mediante regresión determinística estamos ante un modelo de regresión lineal, donde el valor faltante es imputado usando la siguiente ecuación de regresión:

$$\widehat{y}_i=\widetilde{\beta}_{0\cdot 12....K}+\sum_{j=1}^{K}\widetilde{\beta}_{j\cdot 12...K}x_{ij} \tag{3.1}$$

   donde $\widetilde{\beta}_{0\cdot 12...K}$ y $\widetilde{\beta}_{j\cdot 12...K}$ representan los coeficientes de la ecuación de regresión de la variable respuesta Y sobre $X=(X_1,...,X_K)$ basada en las r observaciones completas.

2. **Imputación mediante regresión estocástica**: En el caso de la imputación mediante regresión estocástica tenemos un modelo de regresión cómo el definido en la expresión (3.1), pero en este procedimiento se le incorpora un residuo aleatorio a la predeción. Por lo tanto, se imputará el valor faltante mediante el siguiente modelo de regresión:

$$\widehat{y}_i=\widetilde{\beta}_{0\cdot 12...K}+\sum_{j=1}^{K}\beta_{j\cdot 12...K}x_{ij}+z_i$$

   donde $z_i \backsim N(0,\widetilde{\sigma}_{12...K})$, siendo $\widetilde{\sigma}_{12...K}$ la varianza residual de la regresión de Y sobre X basada en las observaciones completas.

Cuando Y es una variable binaria

En este caso, si consideramos que g es la función logit e Y sigue una distribución de Bernoulli entonces se tiene el siguiente modelo de regresión logística:

$$\text{logit}\{E(Y)\} = X\beta, \qquad Y \backsim \text{Bernoulli}(p)$$

donde $\text{logit}(p) = ln\frac{p}{1-p}$.

El modelo logístico propuesto, establece la siguiente relación entre la probabilidad de que ocurra el suceso, dado que el individuo presenta los valores $X_1 = x_1, X_2 = x_2, ..., X_K = x_K$:

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \sum_{j=1}^{K} \beta_j x_j$$

donde denotamos con $p = P(Y = 1|_{x_1,...,x_K})$.

Para cada registro i con valor y_i pérdido se estima la probabilidad p_i de la siguiente manera:

$$p_i = \frac{1}{1+\exp\left(-\widetilde{\beta}_{0\cdot 12...K} + \sum_{j=1}^{K} \widetilde{\beta}_{j\cdot 12...K} x_j\right)} = \frac{1}{1+\exp\left(-X\widetilde{\beta}\right)}$$

dando la probabilidad de que el valor sea uno frente a cero. Se genera una Bernoulli (p_i) y se asigna un valor a y_i.

Cuando Y es una variable de recuento

En este caso, si consideramos que g es la función logaritmo e Y sigue una distribución de Poisson, el modelo de regresión de Poisson que se obtiene es el siguiente:

$$\ln\{E(Y)\} = X\beta, \qquad Y \backsim Poisson(\lambda)$$

De forma equivalentemente el modelo de Poisson anterior se puede expresar de la siguiente manera:

$$\ln \lambda(x_1, ..., x_k) = \beta_0 + \sum_{j=1}^{K} \beta_j x_j$$

para cada registro i con valor y_i pérdido se calcula un valor:

$$\lambda_*(x_1, ..., x_K) = e^{\overline{\beta}_{0\cdot 12...K} + \sum_{j=1}^{K} \overline{\beta}_{j\cdot 12...K} x_j}$$

Posteriormente se genera un número aleatorio de Poisson de parámetro λ_* asignándolo a y_i.

Cuando Y es una variable categórica (con más de dos categorias)

En este caso, si suponemos que Y es una variable que toma los valores $j = 0, 1, 2, ...l$. Se ajusta un modelo de regresión politómica de Y sobre X.

Para $j = 0, 1, 2, ..., l$ sea $\pi_j = P(Y = j|x)$, el modelo logit generalizado es el siguiente:

$$\ln\left(\frac{\pi_j}{\pi_0}\right) = \beta_0 + \sum_{i=1}^{K} \beta_i x_i, \qquad j = 1, ..., l$$

para cada registro i con valor y_i pérdido se calcula las probabilidades del tipo:

$$p_j = \frac{e^{g_j(x)}}{\sum_{i=0}^{K-1} e^{g_i(x)}}$$

donde $g_j(x) = ln\left(\frac{\pi_j}{\pi_0}\right)$.

Posteriormente se genera un número aleatorio con distribución multinomial, $M(1,p)$ con $p = (p_1,...,p_l)$ asignando a y_i la categoría correspondiente.

Cuando Y es una variable mixta

En este caso, supongamos que Y es una variable mixta que puede tener o bien un valor cero, o bien un valor con distribución continua. El proceso de imputación en este caso se realiza en dos pasos:

1. Se imputa si vale cero o no según el modelo logístico anterior que se haya obtenido.
2. Si resulta que hay que imputar un valor se hace según el modelo de regresión para variables continuas.

Los métodos de imputación por regresión no se sugieren aplicar cuando el análisis secundario de los datos involucra técnicas de análisis de covarianza o de correlación, ya que estos métodos sobreestiman la asociación entre variables, y los modelos de regresión múltiple como veremos en la sección siguiente pueden sobreedimensionar el valor del coeficiente de determinación R^2. Sin embargo, si el método de imputación se aplica por estrato, es necesario garantizar suficientes grados de libertad. En este caso, a pesar de que el sesgo del estimador disminuye el modelo asignará el mismo valor a un grupo de observaciones, lo cual afecta al estimador de la varianza, al coeficiente de correlación de las covariables y la variable imputada, y en modelos de regresión multivariada el R^2 es sesgado.

3.7.9. Imputación mediante el método de regresión secuencial multivariante

Este es un procedimiento estocástico que considera elementos aleatorios. La estrategia básica es la de crear imputaciones por medio de una secuencia de regresiones. El tipo de regresión dependerá de la variable que será imputada y el proceso de esta técnica se basa en recoger la correlación de todas las variables.

En Otero (2011), se recoge la forma de resolver el método que replicamos a continuación:

Sea X una matriz de datos construída con todas las variables completas (no tienen ningún valor faltante). X se compone de variables explicativas como sexo, edad, ... y otras que pueden ser continuas, binarias o categóricas. Por otro lado, sean $Y_1,...,Y_k$ las variables que tienen valores faltantes. Por tanto se tienen en global las variables: $X_1,X_2,X_3,....,Y_1,Y_2,...,Y_K$, donde X_i corresponden a variables que no tienen ningún valor pérdido e Y_j con $j = 1,...,K$ son las variables con algún dato faltante, ordenadas de menor a mayor falta de respuesta. En la iteración inicial se imputa, mediante un modelo de regresión según las siguientes distribucciones condicionadas:

$$\begin{aligned} &Y_1|X \\ &Y_2|X,Y_1 \\ &. \\ &. \\ &. \\ &Y_3|X,Y_1,...,Y_{K-1} \end{aligned}$$

Se empieza haciendo la regresión de la variable con menos falta de respuesta, Y_1, sobre las variables explicativas X. Una vez obtenida una predicción de Y_1 se incorpora esta variable a la matrix X de las variables completas y se obtiene la matriz $[X,Y_1]$ y se realiza la regresión de Y_2 sobre esta última matriz y así sucesivamente. Una vez que se ha realizado esta iteración de regresiones según el modelo correspondiente en función del tipo de variable, se tiene una primera imputación de todos los valores faltantes. En las iteraciones siguientes lo que se realiza es repetir esta iteración inicial pero incluyendo como variables explicativas todas las variables, ya que ahora no hay valores faltantes en ninguna de ellas. Iteración 2:

$$
\begin{aligned}
&Y_1|X, Y_2, ..., Y_K \\
&Y_2|X, Y_1, Y_3, ..., Y_K \\
&. \\
&. \\
&. \\
&Y_K|X, Y_1, ..., Y_{K-1}
\end{aligned}
$$

Este paso da lugar a actualizaciones de las imputaciones hechas en el paso inicial, que incorporan la información de las variables que se imputan después. El proceso se detiene cuando se alcanza el número de iteraciones especificado por el usuario.

3.7.10. Generalidades de los métodos de imputación simple desarrollados

Los métodos de imputación presentados, en el caso de la imputación simple, muestran la distinción entre aquellos que producen valores imputados determinísticos y aleatorios, y aquellos que producen valores artificiales que se basan en valores observados en otros elementos. Podemos distinguir que en el caso de los procedimientos por regresión y del vecino más cercano, existe una fuerte relación entre los valores de la variable a imputar y el vector auxiliar de imputación. Sin embargo, los procedimientos Hot Deck e imputación por la media casi no utilizan información auxiliar. Estos métodos deberían considerarse únicamente como último recurso en ausencia de información auxiliar para imputar, ya que se corre el riesgo de que los valores imputados no sean valores sustitutos cercanos de los valores reales. Si no es posible la aplicación de otros procedimientos, estos métodos en menor medida lograrán el objetivo de obtener una matriz de datos rectangular.

3.7.11. ¿Cuándo es adecuada la imputación simple?

En Medina et al (2007), se reconoce que los métodos de imputación múltiple son más adecuados que los simples. Sin embargo, la literatura reconoce que hay situaciones en que los métodos de imputación simple consiguen resultados satisfactorios. A pesar de estos resultados, no es posible definir a ciencia cierta cuándo es factible favorecer la aplicación de un método de imputación simple, por lo que se recomienda actuar con prudencia y asumir en cada caso las mejores decisiones. Por ejemplo, cuando se trabaja con datos de una encuesta con diseño complejo, en donde el porcentaje de datos faltantes es bajo y está concentrado en una submuestra con características especiales, es probable que un método de imputación simple reproduzca mejor las características de la subpoblación de interés que un algoritmo que considera a toda la muestra.

3.8. Métodos de imputación basados en verosimilitudes

En esta sección nos centramos en métodos de imputación que se basan en funciones de verosimilitud, y que son por lo tanto métodos bajo los que subyace un modelo probabilístico. La idea de utilizar métodos de máxima verosimilitud para tratar datos faltantes viene desde hace más de 50 años. Las primeras soluciones de máxima verosimilitud fueron de alcance limitado y tuvieron relativamente pocas aplicaciones prácticas. Muchos de los avances en los métodos de imputación basados en verosimilitudes se produjeron en los años 70 del siglo pasado, cuando se comenzaron afianzar las técnicas modernas de manejo de datos faltantes. La estimación por máxima verosimilitud extrae continuadamente diferentes combinaciones de valores de parámetros poblacionales hasta que se identifica el conjunto de valores que produce el valor más alto de log-verosimilitud (es decir el mejor ajuste para los datos). Conceptualmente, el proceso de estimación es el mismo con o sin datos faltantes, pero los registros de los datos incompletos requieren una ligera alteración para el cálculo de la log-verosimilitud de las observaciones. Los datos faltantes también necesitan un ajuste de los cálculos de los errores estándar. Finalmente, el análisis de datos faltantes suele requerir de algoritmos de optimización iteractivos incluso para problemas de estimación muy simple. A continuación, mostraremos el marco formal de Rubin (1976), que da idoneidad a estos métodos y que se mantiene en la actualidad.

3.8.1. Un marco formal para la inferencia basada en muestras incompletas

En Gómez et al (2006), se demuestra el marco formal para la inferencia basada en muestras incompletas de la siguiente manera. Por ejemplo, consideremos un fenómeno multivariante real cuyo comportamiento viene descrito por un vector aleatorio K dimensional $X = (X_1, ..., X_K) \in \mathbb{R}^K$ con distribución de probabilidad $P[X;\theta]$, siendo θ el vector de parámetros desconocidos. Cuando se dispone de una muestra completa de X, una amplia clase de métodos de inferencia se justifican en la interpretación de $P[X;\theta]$ como una función de verosimilitud que resume la evidencia que sobre θ hay en los datos, pero en presencia de valores perdidos sólo disponemos de X_{obs} cuya distribución se obtiene como:

$$P[X_{\text{obs}};\theta] = \int P[X;\theta]\ dX_{\text{per}} \tag{3.2}$$

Si pretendemos hacer inferencia sobre θ a partir de la parte observada, es necesario comprobar que la expresión (3.2) es una verosimilitud adecuada. En el año 1976 Rubin identifica las condiciones para que así sea, estableciendo que basta con que se verifique la hipótesis MAR, como se muestra a continuación. Según se ha formalizado el problema de las muestras incompletas, es necesario especificar un modelo para $X, P[X;\theta]$, y un modelo para la no respuesta, $P\left[R|X_{\text{obs}}, X_{\text{per}}, \xi\right]$. Mediante el producto $P\left[R|X_{\text{obs}}, X_{\text{per}}, \xi\right] \cdot P[X;\theta]$ obtenemos la distribución conjunta $P[X, R;\theta,\xi]$. La verosimilitud basada en la parte observada puede expresarse como:

$$P[X_{\text{obs}}, R;\theta,\xi] = \int P[X, R;\theta,\xi]\ dX_{\text{per}} = \int P\left[R|X_{\text{obs}}, X_{\text{per}}, \xi\right] \cdot P[X;\theta]\ dX_{\text{per}} \tag{3.3}$$

Bajo el supuesto MAR, la expresión (3.3) quedaría de la siguiente manera:

$$P[X_{\text{obs}}, R;\theta,\xi] = P[R|X_{\text{obs}},\xi] \int P[X;\theta]\ dX_{\text{per}} = P[R|X_{\text{obs}},\xi] \cdot P[X_{obs};\theta] \tag{3.4}$$

De modo que la verosimilitud obtenida en la expresión (3.3) bajo el supuesto MAR queda factorizada en dos partes, una relativa al vector θ y otra relativa al vector ξ. Si además θ y ξ son distinguibles (en la práctica este supuesto implica que ξ proporciona poca información sobre θ, y viceversa), entonces las inferencias sobre θ basadas en verosimilitudes no se verán afectadas por $P[R|X_{\text{obs}},\xi]$, esto es, el mecanismo de no respuesta puede ser ignorado y la función de verosimilitud L de θ será $L(\theta|X_{\text{obs}})\ \alpha\ P[X_{\text{obs}};\theta]$. Este resultado pone de relieve que bajo el supuesto de ignorabilidad podemos realizar inferencias sobre el vector de parámetros θ de la distribución de X a partir de la verosimilitud $L(\theta|X_{\text{obs}})$. Por otro lado, desde una perspectiva bayesiana, todas las inferencias se basan en la distribución de probabilidad a posteriori de los parámetros desconocidos, que pueden escribirse utilizando el Teorema de Bayes de la siguiente manera:

$$P[\theta,\xi|R, X_{\text{obs}}] = \frac{P[R, X_{\text{obs}}|\theta,\xi] \cdot \varphi(\theta,\xi)}{\int\int P[R, X_{\text{obsv}}|\theta,\xi] \cdot \varphi(\theta,\xi)\ d\theta d\xi} \tag{3.5}$$

donde φ denota la distribución a priori de (θ,ξ). Bajo el supuesto MAR, podemos sustituir la expresión (3.4) en la expresión (3.5), obteniendo que $P[\theta,\xi|R, X_{\text{obs}}]$ es proporcional a $P[R|X_{\text{obsv}},\xi] \cdot P[X_{\text{obsv}}|\theta] \cdot \varphi(\theta,\xi)$. Si además θ y ξ son distinguibles, entonces la distribución marginal a posteriori de θ queda como:

$$P[\theta|X_{\text{obs}}, R] = \int P[\theta,\xi|R, X_{\text{obsv}}]\ d\xi\ \alpha \cdot P[X_{\text{obs}}|\theta] \cdot \varphi_\theta(\theta) \int P[R|X_{\text{obsv}},\xi] \cdot \varphi_\xi(\xi)\ d\xi \cdot \alpha \cdot L(\theta|X_{\text{obs}}) \cdot \varphi_\theta(\theta)$$

Por lo tanto, bajo la hipótesis de ignorabilidad, toda la información sobre θ se recoge en la distribución a posteriori que ignora el mecanismo de no respuesta observado, $P[\theta|X_{\text{obs}}] \cdot \alpha \cdot L(\theta|X_{\text{obs}}) \cdot \varphi_\theta(\theta)$. La necesidad de especificar una distribución a priori para los parámetros puede resultar algo subjetivo o artificial; sin embargo la literatura destaca que conforme aumenta el tamaño muestral la verosimilitud domina a la distribución a priori, y las respuestas bayesianas y verosímiles tienden a converger.

3.8.2. El algoritmo EM (Expectation-Maximization)

El algoritmo EM es particularmente importante para el análisis de datos faltantes. Se trata de un algoritmo iterativo general basado en la idea de factorizar la función de verosimilitud, que permite obtener estimaciones

máximo verosímiles cuando hay datos no completos con unas estructuras determinadas, puesto que este algoritmo se basa en la idea de imputar los valores faltantes e iterar. Las primeras aplicaciones del algoritmo se enfocaron principalmente a la estimación de un vector de medias y una matriz de covarianzas con datos faltantes, pero se ha extendido a una variedad de complejos problemas de estimación de datos completos. Las primeras referencias de este algoritmo son de Mckendrick (1926) quien lo aplicaba en el ámbito de la medicina. Posteriormente Hartley en 1958 amplío y desarrolló la teoría del algoritmo EM y la aplicó al caso de datos procedentes de recuentos. El algoritmo EM fue diseñado por Dempster, Laird y Rubin en 1977 para la obtención de estimadores máximo-verosímiles (EMV) en problemas con muestras incompletas, donde dicho procedimiento consiste en un paso E (Expectation) y un paso M (Maximization). Siguiendo con la terminología del apartado anterior, explicaremos el algoritmo EM de la siguiente manera:

Sea $X = (X_{\text{obs}}, X_{per})$ una muestra incompleta de $X \backsim P[X;\theta]$ a partir de la cual se desea obtener el EMV de θ. Podemos factorizar $P[X;\theta]$ como:

$$P[X;\theta] = P[X_{\text{obs}};\theta] \cdot P\left[X_{\text{per}}|X_{\text{obs}},\theta\right]$$

de donde es sencillo deducir que :

$$\log L(\theta|X_{\text{obs}}) = \log L(\theta|X) - \log P\left(X_{\text{per}}|X_{\text{obs}},\theta\right) \tag{3.6}$$

siendo $\log L(\theta|X_{\text{obs}})$ la log verosimilitud para los datos observados y $\log L(\theta|X_{\text{obs}})$ con respecto a θ dada X_{obsv}. El algoritmo EM relaciona el EMV de θ a partir de $\log L(\theta|X_{\text{obs}})$ con el EMV de θ a partir de $\log L(\theta|X)$. Tomando esperanzas respecto a $P\left[X_{\text{per}}|X_{\text{obs}},\theta\right]$ a ambos lados de la expresión (3.6) y dado un estimador $\theta^{(t)}$ de θ, tenemos que:

$$\log L(\theta|X_{\text{obs}}) = Q\left(\theta;\theta^{(t)}\right) - H\left(\theta;\theta^{(t)}\right)$$

donde $Q\left(\theta;\theta^{(t)}\right) = \int \log L(\theta|X) \cdot P\left[X_{\text{per}}|X_{\text{obs}},\theta^{(t)}\right]\, dX_{\text{per}}$ y $H\left(\theta;\theta^{(t)}\right) = \int \log P\left[X_{\text{per}}|X_{\text{obs}},\theta\right] \cdot P\left[X_{\text{per}}|X_{\text{obs}},\theta^{(t)}\right]\, dX_{\text{per}}$.

El paso E (Expectation) del algoritmo EM calcula $Q\left(\theta;\theta^{(t)}\right)$, reemplazando los valores perdidos, o una función de ellos, por su esperanza condicionada dados X_{obs} y $\theta^{(t)}$. El paso M (Maximization) simplemente determina el EMV $\theta^{(t+1)}$ que maximiza $Q\left(\theta;\theta^{(t)}\right)$ como si no hubiera datos perdidos. Los pasos E y M se repiten alternativamente generando una sucesión de estimadores $\{\theta^{(t)}\}$. La diferencia en el valor de la log verosimilitud $\log L(\theta|X_{\text{obs}})$ en dos iteraciones sucesivas viene dada por:

$$\log L\left(\theta^{(t+1)}|X_{\text{obs}}\right) - \log L\left(\theta^{(t)}|X_{\text{obs}}\right) = Q\left(\theta^{(t+1)};\theta^{(t)}\right) - Q\left(\theta^{(t)};\theta^{(t)}\right) + H\left(\theta^{(t)};\theta^{(t)}\right) - H\left(\theta^{(t+1)};\theta^{(t)}\right)$$

Como el estimador $\theta^{(t+1)}$ se recoge de manera que $Q\left(\theta^{(t+1)};\theta^{(t)}\right) \geq Q\left(\theta^{(t)};\theta^{(t)}\right)$ y $H\left(\theta^{(t)};\theta^{(t)}\right) \geq H\left(\theta^{(t+1)};\theta^{(t)}\right)$, lo cual se deduce de la desigualdad de Jensen y la concavidad de la función logarítmica, tenemos que $\log L(\theta|X_{\text{obsv}})$ se va incrementando en cada iteración con lo que se converge hacia el EMV de θ.

Es importante resaltar que las estimaciones iniciales de θ pueden ser realizadas mediante diferentes procedimientos alternativos como:

- Análisis de datos completos.
- Análisis de datos disponibles.
- Imputación de valores faltantes.
- Cálculo de las medias y varianza con los valores observados fijando las covarianzas a cero.

El análisis de datos completos proporciona estimaciones consistentes si el patrón de datos es MCAR y hay un número suficiente de registros con datos completos. En el caso del análisis de datos disponibles, este procedimiento tiene la ventaja de usar toda la información disponible, pero puede llevar a estimaciones de la matriz de varianzas-covarianzas no definida positivamente dando problemas en la primera iteración. Por último, la imputación de los valores faltantes y el cálculo de las medias y varianzas con los valores observados fijando las covarianzas a cero pueden conducir a estimaciones inconsistentes de la matriz de varianzas-covarianzas.

3.9. Métodos de imputación múltiple

Los métodos de imputación múltiple consisten en realizar varias imputaciones de las observaciones faltantes para luego analizar los conjuntos de datos completados y combinar los resultados obtenidos, con el fin de obtener una estimación final. Cómo ya se ha mencionado, la imputación múltiple reemplaza cada valor perdido por un conjunto de valores simulados con el fin de incorporar la incertidumbre debida a la presencia de datos faltantes. La primera referencia destacable de los métodos de imputación múltiple es la de Rubin (1987), que a principios de la década de los ochenta del siglo pasado empezó a desarrollar este tipo de imputación. Se pueden encontrar trabajos relevantes a este procedimiento cómo : Rubin (1996), Schafer (1997), Little et al (2002) o Zhang (2003) entre otros.

La métodología de los métodos de imputación múltiple ha permanecido durante algunos años en un segundo plano debido a su dificultad de aplicación, propiciada principalmente por la inexistencia de herramientas computacionales adecuadas a este tipo de procedimiento. Es en las últimas décadas cuando los avances tecnológicos han permitido la implementación de algoritmos y procedimientos de cálculo computacionales necesarios para dar solución a problemas analíticamente intratables en los primeros años. Concretamente en la década de los noventa del siglo pasado, se comienzan a popularizar los algoritmos MCMC (Markov Chain Monte Carlo) que permiten una modelización estadística más compleja y realista y que detallaremos en la siguiente sección.

A diferencia de los métodos de imputación presentados en las secciones anteriores, que imputan un valor único a cada dato desconocido, la imputación múltiple tiene un procedimiento que se basa en imputar más de un valor para cada valor ausente. En este sentido la imputación múltiple consiste en generar $m > 1$ valores aleatorios para cada valor perdido por no respuesta, de manera que se disponga de m conjuntos de datos completos, que permitan realizar los análisis estadísticos usuales de los m cojuntos de datos generados para las m estimaciones, con la finalidad de producir una estimación con buenas propiedades estadísticas y con la posibilidad de estimar la varianza de las estimaciones.

El método de imputación múltiple consta de tres etapas:

1. **Imputación**: en esta etapa cada valor perdido se reemplaza por un conjunto de m valores simulados a partir de la distribución predictiva de X_{per} dado un modelo de probabilidad para X y una distribución a priori para θ. Dicha distribución $P\left[X_{\text{per}}|X_{\text{obs}}\right]$ puede obtenerse como:

$$P\left[X_{\text{per}}|X_{\text{obs}}\right] = \int P\left[X_{\text{per}}|X_{\text{obs}},\theta\right] \cdot P\left[\theta|X_{\text{obs}}\right]\, d\theta \tag{3.7}$$

   En la expresión (3.7) se refleja tanto la incertidumbre sobre X_{per} dado el vector de parámetros θ, como la propia incertidumbre asociada a θ. A menudo $P\left[\theta, X_{\text{obs}}\right]$ y los cálculos donde interviene resultan intratables analíticamente, especialmente en contextos multidimensionales. Es aquí donde intervienen de forma natural los algoritmos MCMC dentro de esta metodología.

2. **Análisis**: en esta etapa se aplica el análisis de datos deseado (regresión, análisis discriminante,...) sobre cada una de las matrices imputadas y se almacenan los resultados.

3. **Combinación**: finalmente las estimaciones resultantes se combinan para obtener una única estimación global. Sea Q un parámetro poblacional unidimensional de interés (por ejemplo una media, una varianza, un coeficiente β de una regresión,...) , sea $\widehat{Q}$ su estimación puntual si no hubiera datos perdidos, denotando con U la varianza estimada de $\widehat{Q}$. Tras analizar los datos imputados tenemos m estimaciones $\{\widehat{Q}_1,...,\widehat{Q}_m\}$ con varianzas estimadas asociadas $\{U_1,...,U_2\}$. Se derivan las siguientes reglas (Rubin, 1987) para combinar las m estimaciones: la estimación puntual para Q basada en imputación múltiple, $\overline{Q}_m$, y su varianza asociada, T_m, vendrá dadas por:

$$\overline{Q}_m = \frac{1}{m}\sum_{i=1}^{m}\widehat{Q}_i \text{ y } T_m = \overline{U}_m + \left(1+\frac{1}{m}\right)\cdot B_m \tag{3.8}$$

   donde $\overline{U}_m = \sum_{i=1}^{m} U_i/m$ representa la variabilidad intra-imputaciones y $B_m = \sum_{i=1}^{m}\left(\widehat{Q}_i - \overline{Q}\right)/\left(\widehat{Q}_i - \overline{Q}\right)'/(m-1)$ la variabilidad entre-imputaciones. Sin datos perdidos, las estimaciones $\widehat{Q}_i, i = 1,...,m$, serían idénticas, con lo que $B = 0$ y $T = \overline{U}$.

Finalmente, el objetivo de la imputación múltiple es hacer un uso eficiente de los datos que se han recogido, obtener estimadores no sesgados y reflejar adecuadamente la incertidumbre que la no respuesta parcial introduce en la estimación de los parámetros. El número óptimo de base de datos *m* depende del porcentaje de información faltante y según Schafer (1997) se ubica entre tres y diez grupos. Cada una de las *m* estimaciones anteriores se pueden crear con una gran variedad de métodos, desde los más simples, como la imputación por media, hasta los más complejos, como los modelos de Monte Carlo con cadenas de Markov (MCMC- Markov Chain Monte Carlo). Incialmente este procedimiento se desarrolló con técnicas de imputación simple para generar los valores a imputar. Sin embargo los métodos más utilizados en la actualidad son: aproximación bayesiana y Monte Carlo con cadenas de Markov.

3.9.1. Imputación múltiple Markov Chain Monte Carlo (MCMC)

La imputación múltiple Markov Chain Monte Carlo (MCMC) es uno de los procedimientos que se consideran más adecuados para generar imputaciones. El MCMC es una colección de procesos de simulación generados por métodos de selección aleatoria mediante cadenas de Markov. El MCMC utiliza simulación paramétrica generando muestras aleatorias a partir de métodos bayesianos y, en el método de imputación múltiple, se aplica para generar las *m* selecciones independientes de valores faltantes, las cuales se utilizan en la etapa de inferencia.

Asumiendo que los datos provienen de una distribución normal multivariable, la agregación de los datos es aplicada desde la inferencia bayesiana a datos faltantes a través de la repetición de los siguientes pasos:

1. **Imputación**: con la estimación del vector de la media y la matriz de covarianzas, el primer paso consiste en simular los valores faltantes para cada una de las observaciones de forma independiente.

2. **Distribución posterior**: concluida la simulación del primer paso, se obtiene el vector de medias de la población y de la matriz de covarianzas de la muestra completa.

Finalmente se realizan varias iteraciones. El objetivo es que estas iteraciones converjan a la distribución estacionaria y entonces se obtenga una estimación aproximada de los valores faltantes.

3.9.2. Consideraciones acerca de los procedimientos de imputación múltiple

En el trabajo de Schafer (1999), se muestran algunos de los problemas que le surgen a los usuarios en el procedimiento de imputación múltiple, los cuales enumeramos a continuación:

1. Uno de los primeros problemas que surgen en la imputación múltiple es que a la hora de eliminar registros de la base de datos se suele asumir que las observaciones completas tienen la misma distribución de probabilidad que las omitidas, es decir, que los datos faltantes representan una submuestra aleatoria de la muestra total. No obstante, y dado que no es posible corroborar si la información faltante tiene un comportamiento particular, se mantedrá la duda de cómo se comportan los grupos de datos sin información respecto a la distribución de las variables de interés.

2. Otro interrogante que surge en la aplicación de la imputación múltiple, es por qué se deben generar varias imputaciones. En el caso de que la falta de respuesta no sea importante, es probable que una imputación sea suficiente para obtener estimadores insesgados. En Medina et al (2007), se dice que esto se logra si se comprueba que la imputación no ha modificado las características estadísticas de la variable de análisis. La objeción que frecuentemente establecen los analistas a esta forma de proceder, es que a partir de un algoritmo simple no es posible estimar el error de los estimadores que utilizan valores imputados, lo cual si se puede determinar cuando se aplica el procedimiento de imputación múltiple.

3. En situaciones en que la no respuesta es baja se sugiere aplicar la imputación múltiple, especialmente cuando en el análisis se utilizan técnicas multivariadas. De hecho en Rubin (1987), se afirma que el método de imputación múltiple es capaz de generar resultados robustos con un número pequeño de iteraciones. Dado que la eficiencia relativa de *m* imputaciones es aproximadamente $(1+\lambda/m)^{-1}$, en donde λ es la tasa de registros sin información, Rubin indica que aún en situaciones en donde se tenga el 50 % de datos faltantes, el estimador generado a partir de $m = 5$ imputaciones tiene una desviación estándar que es sólo del 5 % mayor que otra simulación basada en $m = \infty$ iteraciones, debido a que $\sqrt{1+0{,}5/5} = 1{,}049$. También señala, que para tasas de respuesta inusualmente altas sólo se requiere generar entre 5 y 10 imputaciones. A pesar de que Rubin afirma que la imputación múltiple es eficiente ante tasas de no respuesta importantes, se sugiere

asumir esta afirmación con cautela. Por ejemplo: supongamos que como resultado de un trabajo de campo en una encuesta de hogares se comprueba que más del 50 % de los datos de la variable ingreso no tienen información. En este contexto, y antes de aplicar un método de imputación múltiple es necesario cuestionarse la validez de este estudio, dado que en este caso el proceso implicaría imputar valores a más de la mitad de la población en estudio, es decir, la mitad de los datos que no fueron recopilados en el terreno, serán sustituidos por valores provenientes de un modelo estadístico.

3.10. ¿Cómo seleccionar la técnica adecuada de imputación?

En Mesa et al (2006), se establece que seleccionar un método de imputación adecuado es una decisión de gran importancia, ya que para un conjunto de datos determinado algunas técnicas de imputación podrían ofrecer mejores aproximaciones a los valores reales que otras. Para la selección de la técnica de imputación adecuada, no hay unas reglas específicas que establezcan lo que se debe hacer, dependerá entonces del tipo de conjunto de datos, tamaños del archivo, tipo de no respuesta, patrón de pérdida de respuesta, de los objetivos de la investigación, características específicas de la población, características generales de la organización del estudio, software disponible, importancia de los valores agregados o de los valores puntuales, distribuciones de frecuencias de cada variable, marginal o conjunta, etc. Hay que tener en cuenta que muchas veces la técnica de imputación seleccionada puede ser adecuada para algunas variables pero para otras no, y por lo tanto será decisión del investigador seleccionar la técnica que menos afecte las estimaciones de las variables. En Fellegi et al (1976), se plantea que la técnica de imputación seleccionada debe superar las reglas de validación, cambiando lo menos posible los registros y manteniendo la frecuencia de la estructura de la información faltante.

En Goicochea (2002), se resumen los criterios a tomar en consideración para seleccionar una técnica adecuada de imputación, que enumeramos a continuación:

1. **Tipo de variable a imputar**: si es continua hay que tener en cuenta el intervalo para la cual se define y si es cualitativa, tanto nominal como ordinal las categorías de las variables.

2. **Parámetros que se desean estimar**: si deseamos conocer sólo valores agregados como la media y el total, se pueden aplicar métodos sencillos como imputación con la media o moda, sin embargo puede haber una subestimación de la varianza. En caso de que se requiera mantener la distribución de frecuencias de la variable y las asociaciones entre las distintas variables, se deben emplear métodos más elaborados aplicando imputación de todas las variables faltantes del registro.

3. **Tasas de no respuesta y exactitud necesaria**: cuando el porcentaje de no respuesta es alto en una base de datos, se considera que no hay confiabilidad en los resultados que se obtengan con el análisis.

4. **Información auxiliar disponible**: es bueno hacer uso de la información auxiliar disponible, ya que con ella podemos deducir información de los valores ausentes de una variable o hallar grupos homogéneos respecto a una variable auxiliar que se encuentre altamente correlacionada con la variable a imputar, y de esta manera encontrar un donante adecuado que sea similar al registro receptor.

3.10.1. Pasos para llevar a cabo un proceso de imputación

Según Goicochea (2002) los pasos que se llevan a cabo para realizar una imputación son cinco:

- **Paso I**: una vez que se dispone de un archivo con datos faltantes, se recopila y valida toda la información auxiliar disponible que pueda ser de ayuda para la imputación.

- **Paso II**: se estudia el patrón de pérdida de respuesta. Posteriormente se observa si hay un gran número de registros que simultáneamente tienen no respuesta en un conjunto de variables.

- **Paso III**: se seleccionan varios métodos de imputación posibles y se contrastan los resultados.

- **Paso IV**: se calculan las varianzas para los distintos métodos de imputación seleccionados con el objetivo de obtener estimaciones con el mínimo sesgo y la mejor precisión.

- **Paso V**: se concluye a partir de los resultados obtenidos.

Capítulo 4

Aplicación de los métodos de imputación

El objetivo del capítulo 4 es plantear la utilización de las técnicas de imputación presentadas en el capítulo 3 a través de diferentes paquetes especializados del lenguaje de programación R. La utilización de diferentes funciones de imputación en R nos permitirá observar los diferentes métodos de imputación y sus variantes. El lenguaje de programación R es un entorno especialmente diseñado para el tratamiento de datos, cálculo y desarrollo gráfico, que permite trabajar con facilidad con vectores y matrices y ofrece diversas herramientas para el análisis de datos. El lenguaje de programación R forma parte del proyecto GNU[1] y puede verse como una implementación alternativa del lenguaje S, desarrollado en AT&T Bell Laboratories. Dicho lenguaje se presenta como software libre, lo que implica la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. Se trata de un lenguaje creado específicamente para la visualización y exploración de datos así como para su uso en modelización y programación estadística. En la web *http: //www.r-project.org/index.html* se encuentra disponible toda la información acerca de R. Además, R es un entorno en el que se han ido implementando diversas técnicas estadísticas, muchas de ellas disponibles en paquetes (packages) accesibles a través de la web *http: //cran.au.r-project.org/*.

R proporciona un entorno de trabajo especialmente preparado para el análisis estadístico de datos, de cual se pueden destacar las siguientes características:

- R proporciona un lenguaje de programación propio, basado en el lenguaje S, que a su vez tiene muchos elementos del lenguaje C. Sin embargo, la semántica es muy distinta a la de este último; esto es porque R permite ejecuciones de comandos en línea (compilación y ejecución unidos en un mismo paso), lo cual hace que su semántica esté más próxima a la de un lenguaje de programación funcional.
- Objetos y funciones específicas para el tratamiento de datos.
- R es software libre por lo tanto permite la descarga de librerías con implementaciones concretas de funciones gráficas, métodos estadísticos, algoritmos...
- En su momento el lenguaje S evolucionó por un lado hacia R, software libre con licencia GNU, y por otro hacia S-plus, software comercial y con un entorno gráfico mucho más amigable. Esto hace que pasar de R a S-plus o al revés sea relativamente sencillo (bajo los dos subyace el mismo lenguaje de programación).

4.1. Paquetes en R para imputación

Dentro del CRAN de R en la dirección *http: //cran.au.r-project.org/*, se pueden encontrar diferentes paquetes para aplicar técnicas de imputación, algunos de los cuales se presentan a continuación con una breve descripción junto con su utilización posterior para ejemplificar los distintos métodos.

- **MissingDataGUI**: proporciona resumenes numéricos y gráficos para los datos faltantes de variables categóricas y cuantitativas. Aplica una variedad de métodos de imputación como imputaciones univariantes con valores fijos o aleatorios, imputaciones multivariantes como el vecino más cercano, imputaciones múltiples e imputaciones condicionadas a una variable categórica.

[1] El proyecto GNU se inició en el año 1984 con el propósito de desarrollar un sistema operativo compatible con Unix que fuera software libre. Para más información sobre el proyecto GNU se puede consultar la página web *http://www.gnu.org/*.

- **Amelia II**: imputa de forma múltiple datos faltantes de sección transversal única (como un estudio), de una serie de tiempo (como variables correlacionadas en un país), o de un conjunto de datos de corte transversal de series de tiempo (tales como variables correlacionadas durante años para varios países). El paquete Amelia II implementa un algoritmo basado en bootstrap, por lo que general es considerablemente más rápido que otros enfoques y puede manejar muchas más variables. A diferencia del paquete Amelia I y otro software de imputación estadísticamente riguroso, virtualmente nunca se bloquea. También incluye diagnósticos útiles del ajuste de modelos de imputación múltiple.

- **VIM**: introduce nuevas herramientas para la visualización de valores faltantes y/o imputados, que pueden ser utilizados para explorar los datos y la estructura de los valores faltantes y/o imputados. Dependiendo de la estructura de los valores faltantes, los métodos correspondientes pueden ayudar a identificar el mecanismo, generando valores perdidos y permitiendo explorar los datos incluyendo valores faltantes. Además, la calidad de imputación puede ser visualmente explorada utilizando varios métodos gráficos univariantes, bivariantes y multivariantes. Por último, el paquete VIM tiene una interfaz gráfica que permite un fácil manejo de los métodos gráficos implementados.

- **mice**: realiza imputación múltiple utilizando Folly Conditionally Specification (FCS) implementada a través del algoritmo MICE (Multiple Imputation by Chained Equations); este algoritmo permite que cada variable tenga su propio modelo de imputación. El paquete mice tiene incorporados modelos de imputación para datos continuos (pmm), datos binarios (regresión logística), datos categóricos no ordenados (regresión logística polinómica) y datos categóricos ordenados (odds proporcional). Con el paquete mice también se puede utilizar imputación pasiva para mantener consistencia entre las variables. Por otro lado también dispone de varios gráficos de diagnóstico para examinar la calidad de las imputaciones.

- **BaBooN**: incluye dos variantes del Bayesian Bootstrap Predictive Mean Matching para multiplicar e imputar datos faltantes. La primera variante es una imputación variable por variable que combina la regresión secuencial y el método de comparación predictiva media (PMM) que se ha extendido para datos categóricos no ordenados. El Bayesian Bootstrap permite generar aproximadamente imputaciones múltiples. La segunda variante del paquete BaBOON también se basa en el PMM, pero en este caso el foco está en imputar varias variables al mismo tiempo. Se recomienda utilizar el paquete BaBoon cuando el patrón de datos perdidos es una fusión de datos o bien cuando el patrón de datos perdidos es idéntico para varias variables. Ambas variantes se pueden ejecutar como imputación única, en caso de que el objeto de estudio sea puramente de carácter descriptivo.

- **ForImp**: permite la imputación de valores perdidos en un conjunto de datos de variables ordinales a través de un algoritmo de imputación directa.

- **HotDeckImputation**: proporciona métodos de imputación Hot Deck para resolver los datos que faltan. Las funciones del paquete HotDeckImputation emplean la metodología de las encuestas utilizada principalmente en el contexto de grandes estadísticas nacionales, encontrando su aplicación en la minería de datos debido a su simplicidad computacional. Un aspecto clave del paquete HotDeckImputation es la implementación del comúnmente recomendado límite de donantes. Por otro lado, incluye la aplicación de diferentes métodos como Hot Deck del vecino más cercano o Hot Deck secuencial.

- **imputeTS**: es una colección de algoritmos y herramientas para la imputación de series temporales univariadas. El paquete imputeTS ofrece varias implementaciones de algoritmos de imputación y proporciona funciones de trazado e impresión de estadísticas de datos faltantes.

- **jomo**: permite el modelado de diferentes niveles de imputación múltiple, con la posibilidad de manejar datos binarios y categóricos a través de variables normales. Por otro lado, tiene la opción de utilizar matrices de covarianzas especifícas de clúster.

- **longitudinalData**: proporciona algunas herramientas para hacer frente a la agrupación de datos longitudinales, principalmente:

  1. plotTrajmeans
  2. imputation
  3. qualityCriterion

- **miceadds**: contiene funciones auxiliares para imputación múltiple, con características como la imputación de un valor plausible y la imputación utilizando cuadrados (PLS) para predictores de alta dimensionalidad y anidados de la imputación múltiple.

- **missForest**: se utiliza para imputar valores perdidos particularmente en el caso de datos de tipo mixto. El paquete missForest, puede ser utilizado para imputar datos continuos y/o categóricos incluyendo interacciones complejas y relaciones no lineales. Proporciona una estimación de error de imputación fuera de bolsa (OOB). Además este paquete se puede ejecutar en paralelo, lo que permite ahorrar tiempo de computación.
- **simputation**: imputación basada en modelos CART o en árboles aleatorios.
- **TestDataImputation**: imputa respuestas de items faltantes para variables dicotómicas. Permite la imputación faltante a través de métodos adecuados para los datos de prueba y evaluación como: el litwise (LM), la supresión, tratamiento como incorrecto (IN), imputación de la persona media (PM), imputación media del artículo (IM), imputación bidirrecional (TW), imputación de regresión logística (LM) y imputación EM.

4.2. Paquete imputeTS

Es una colección de algoritmos y herramientas para la imputación de series temporales univariadas. Ofrece varias implementaciones de algoritmos de imputación y proporciona funciones de trazado e impresión de estadísticas para datos faltantes. Este paquete es de fácil uso ya que todas las funciones de imputación están implementadas del mismo modo.

4.2.1. Datos de aplicación del paquete imputeTS

tsAirgap (Time series of monthly airline passengers (with NAs))

Se corresponde con la serie cronólogica de pasajeros mensuales de las líneas aéreas (con NAs), con el número total mensual de los pasajeros de las líneas aéreas internacionales, desde 1949 a 1960. En el paquete imputeTS también se incluye la serie temporal tsAirgapComplete que proporciona los valores verdaderos para los valores faltantes. El conjunto de datos tsAirgap tiene un formato que incluye 144 filas con 13 NAs. Se origina a través de un modelo Box & Jenkins y es un ejemplo comúnmente usado en la literatura de las series de tiempo, dado que sus características son una fuerte tendencia y una fuerte componente estacional, lo que la convierte en un gran ejemplo para la imputación en series de tiempo.

Cuadro 4.1: Representación del conjunto de datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112	118	132	129	NA	135	148	148	NA	119	104	118
1950	115	126	141	135	125	149	170	170	NA	133	NA	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	NA	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	NA	NA	NA	374	413	405	355	306	271	306
1957	315	301	356	348	355	NA	465	467	404	347	NA	336
1958	340	318	NA	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	NA
1960	417	391	419	461	NA	535	622	606	508	461	390	432

Fuente: Elaboración propia. Los valores coloreados en rojo representa los valores perdidos.

4.2.2. Aplicación de funciones del paquete imputeTS

na.Kalman (Missing Values by Kalman Smoothing and State Space Models)

Utiliza filtrado de Kalman en modelos de series temporales estructurales (o en la representación espacial de un modelo arima) para imputación. La función acepta dos entradas:

- *" auto.arima "*: para la representación del modelo arima en espacio estado.
- *" Structs "*: para un modelo estructural ajustado por máxima verosimilitud.

Además para los parámetros adicionales de *auto.arima* y *Structs*, es también posible utilizar un modelo estado espacio creado por el usuario, que podría obtenerse a través de otro paquete en R para la modelización estructural de series temporales. La representación de un modelo arima estado espacio también es posible, pero es importante destacar que los modelos estado espacio creados por el usuario deben estar especificados bajo el requisito Kalman-Like, esto significa que el estado suministrado por el usuario tiene que estar en forma de una lista con al menos los componentes T, Z, h, V o P. En definitiva, el filtro de Kalman utilizado en la función **na.Kalman()** es el que opera en un modelo estructural básico obtenido por Structs o por la representación espacial de un modelo ARMA.

Ejemplos:

Ejemplo 1: Realización de la imputación con filtro de Kalman y representación de un modelo arima estado espacio para los datos de tsAirgap aplicando:

```
> na.kalman(tsAirgap)
```

Cuadro 4.2: Representación de un modelo arima estado espacio obtenido por estimación para los datos de tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	123.5734	135.0000	148.0000	148.0000	132.7833	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	157.1427	133.0000	130.8971	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	260.8078	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	312.6060	311.6160	311.0830	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	414.6160	465.0000	467.0000	404.0000	347.0000	298.6147	336.0000
1958	340.0000	318.0000	382.6912	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	390.9089
1960	417.0000	391.0000	419.0000	461.0000	452.5683	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.Kalman()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 2: Realización de la imputación con Kalmarun y representación de un modelo espacial arima aplicando:

```
> na.kalman(tsAirgap, smooth = FALSE)
```

Cuadro 4.3: Representación de un modelo arima estado espacio obtenido por extrapolación para los datos de tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	123.9264	135.0000	148.0000	148.0000	141.5691	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	139.5378	133.0000	141.7595	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	232.6461	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	304.4782	302.5073	298.1029	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	385.8343	465.0000	467.0000	404.0000	347.0000	327.9040	336.0000
1958	340.0000	318.0000	392.3223	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	400.7839
1960	417.0000	391.0000	419.0000	461.0000	442.0008	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.Kalman()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 3: Realización de la imputación con el modelo KalmanSmooth y Structs conjuntamente para los datos tsAirgap aplicando:

```
> na.kalman(tsAirgap, model ="StructTS", smooth = TRUE)
```

Cuadro 4.4: Representación de un modelo KalmanSmooth y Structs para los datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	123.5734	135.0000	148.0000	148.0000	132.7833	119.0000	104.0000	118.0000
19506	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	157.1427	133.0000	130.8971	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	260.8078	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	312.6060	311.6160	311.0830	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	414.6160	465.0000	467.0000	404.0000	347.0000	298.6147	336.0000
1958	340.0000	318.0000	382.6912	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	390.9089
1960	417.0000	391.0000	419.0000	461.0000	452.5683	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.Kalman()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 4: Realización de la imputación con el modelo KalmanSmooth y Structs con parámetros adicionales para los datos de tsAirgap aplicando:

```
> na.kalman(tsAirgap, model ="StructTS", smooth = TRUE, type ="trend")
```

Cuadro 4.5: Representación de un modelo KalmanSmooth y Structs para los datos tsAirgap con parámetros adicionales.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.00	118.00	132.00	129.00	132.00	135.00	148.00	148.00	133.50	119.00	104.00	118.00
1950	115.00	126.00	141.00	135.00	125.00	149.00	170.00	170.00	151.50	133.00	136.50	140.00
1951	145.00	150.00	178.00	163.00	172.00	178.00	199.00	199.00	184.00	162.00	146.00	166.00
1952	171.00	180.00	193.00	181.00	183.00	218.00	230.00	242.00	209.00	191.00	172.00	194.00
1953	196.00	196.00	236.00	235.00	229.00	243.00	264.00	272.00	237.00	211.00	180.00	201.00
1954	204.00	188.00	235.00	227.00	234.00	268.00	302.00	293.00	259.00	229.00	203.00	229.00
1955	242.00	233.00	267.00	269.00	270.00	315.00	364.00	347.00	312.00	274.00	237.00	278.00
1956	284.00	277.00	301.25	325.50	349.75	374.00	413.00	405.00	355.00	306.00	271.00	306.00
1957	315.00	301.00	356.00	348.00	355.00	410.00	465.00	467.00	404.00	347.00	341.50	336.00
1958	340.00	318.00	333.00	348.00	363.00	435.00	491.00	505.00	404.00	359.00	310.00	337.00
1959	360.00	342.00	406.00	396.00	420.00	472.00	548.00	559.00	463.00	407.00	362.00	389.50
1960	417.00	391.00	419.00	461.00	498.00	535.00	622.00	606.00	508.00	461.00	390.00	432.00

Fuente: Elaboración propia con la función **na.Kalman()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 5: Realización de la imputación con KalmanSmooth y el modelo creado por el usuario para los datos tsAirgap aplicando:

```
> usermodel <- arima(tsAirgap,order = c(1,0,1))$model
> na.kalman(tsAirgap,model = usermodel)
```

Cuadro 4.6: Representación de un modelo KalmanSmooth y un modelo creado por el usuario para los datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	128.9291	135.0000	148.0000	148.0000	136.5832	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	150.9669	133.0000	132.7634	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	270.8450	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	292.8668	316.5817	341.4769	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	409.7449	465.0000	467.0000	404.0000	347.0000	332.3090	336.0000
1958	340.0000	318.0000	329.7566	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	388.2858
1960	417.0000	391.0000	419.0000	461.0000	487.7672	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.Kalman()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores perdidos.

na.mean (Missing Value Imputation by mean value)

Los valores que faltan se reemplazan por valores medios totales. La función permite varias entradas a través del argumento *option*:

- *"mean "*: toma la media para la imputación.
- *"median "*: toma la mediana para la imputación.
- *"mode "*: toma la moda para la imputación.

Por lo tanto la función **na.mean()** calcula la media, la mediana o moda sobre todos los valores no NA y reemplaza todos los NA con ese valor. En lo referente a la opción *"mode "* hay que destacar que reemplaza NA con el valor más frecuente en la serie temporal, en caso de que dos o más valores ocurran con igual frecuencia, la función imputa con el valor inferior. Es por esto que la opción *"mode "* no es la mejor opción para los valores decimales.

Ejemplos:

Para ejemplificar la utilización de la función **na.mean()**, creamos un vector x con valores perdidos aplicando:

```
> x <- ts(c(2,3,4,5,6,NA,7,8))
```

de dicha aplicación obtenemos el siguiente vector x=(2 3 4 5 6 NA 7 8).
Ejemplo 1: Realiza la imputación del vector x con el promedio general, aplicando:

```
> na.mean(x)
```

la aplicación nos devuelve el siguiente vector x=(2 3 4 5 6 5 7 8) imputado.
Ejemplo 2: Realiza la imputación del vector con la mediana general, aplicando:

```
> na.mean(x, option ="median")
```

la aplicación nos devuelve el siguiente vector x= (2 3 4 5 6 5 7 8) imputado.

na.seadec (Seasonally Decomposed Missing Value Imputation)

Elimina el componente estacional de la serie temporal, y realiza la imputación en la serie desestacionalizada para después agregar el componente estacional otra vez. Una vez hecha la descomposición la función **na.seadec()** permite las siguientes entradas a través del argumento *algorithm*:

- *"interpolation "*: imputación por interpolación.
- *"locf "*: imputación por última observación llevada adelante.
- *"mean "*: imputación por valor medio.
- *"random "*: imputación por muestra aleatoria.
- *"Kalman "*: imputación por filtro de Kalman y Modelos en espacio de estado.

Ejemplos:

Ejemplo 1: Realizamos la imputación estacional utilizando el algoritmo *"interpolation "*, aplicando:

```
> na.seadec(tsAirgap, algorithm = "interpolation")
```

Cuadro 4.7: Representación de la serie de tiempo imputada por imputación estacional utilizando interpolación para el conjunto de datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	121.3941	135.0000	148.0000	148.0000	131.6204	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	149.7682	133.0000	113.0095	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	274.9995	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	321.8776	321.4178	329.4329	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	424.1798	465.0000	467.0000	404.0000	347.0000	305.0315	336.0000
1958	340.0000	318.0000	357.2362	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	398.3917
1960	417.0000	391.0000	419.0000	461.0000	477.2415	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.seadec()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 2: Realizamos la imputación estaciaonal utilizando el algoritmo *"mean"*, aplicando:

```
> na.seadec(tsAirgap, algorithm = "mean")
```

Cuadro 4.8: Representación de la serie de tiempo imputada por imputación estacional utilizando imputación por valor medio para el conjunto de datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	278.5342	135.0000	148.0000	148.0000	289.2064	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	289.6443	133.0000	237.2661	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	310.9817	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	282.0424	276.8787	280.1899	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	323.7648	465.0000	467.0000	404.0000	347.0000	215.2535	336.0000
1958	340.0000	318.0000	281.3777	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	242.2860
1960	417.0000	391.0000	419.0000	461.0000	281.0843	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.seadec()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

na.seasplit (Seasonally Splitted Missing Value Imputation)

Divide la serie de tiempo en temporadas, para luego realizar la imputación separadamente para cada una de las series de datos de la serie de tiempo, lo que implica que cada serie de tiempo dividida tiene una situación específica. La función **na.seasplit()** permite después de haber realizado las divisiones, las siguientes entradas a través del argumento *algorithm*:

- *"interpolation "*: imputación por interpolación.
- *"mean "*: imputación por valor medio.
- *"random "*: imputación por muestra aleatoria.
- *"kalman "*: imputación por filtro de Kalman Modelos en espacio estado.

Ejemplos:

Ejemplo 1: Realizamos la imputación dividida estacional utilizando el argumento *"interpolation "* para el conjunto de datos tsAirgap, aplicando:

```
> na.seasplit(tsAirgap, algorithm = "interpolation")
```

Cuadro 4.9: Representación de la serie de tiempo imputada por división estacional utilizando interpolación para el conjunto de datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0	118.0	132.0	129.0	125.0	135.0	148.0	148.0	184.0	119.0	104.0	118.0
1950	115.0	126.0	141.0	135.0	125.0	149.0	170.0	170.0	184.0	133.0	125.0	140.0
1951	145.0	150.0	178.0	163.0	172.0	178.0	199.0	199.0	184.0	162.0	146.0	166.0
1952	171.0	180.0	193.0	181.0	183.0	218.0	230.0	242.0	209.0	191.0	172.0	194.0
1953	196.0	196.0	236.0	235.0	229.0	243.0	264.0	272.0	237.0	211.0	180.0	201.0
1954	204.0	188.0	235.0	227.0	234.0	279.0	302.0	293.0	259.0	229.0	203.0	229.0
1955	242.0	233.0	267.0	269.0	270.0	315.0	364.0	347.0	312.0	274.0	237.0	278.0
1956	284.0	277.0	311.5	308.5	312.5	374.0	413.0	405.0	355.0	306.0	271.0	306.0
1957	315.0	301.0	356.0	348.0	355.0	404.5	465.0	467.0	404.0	347.0	290.5	336.0
1958	340.0	318.0	381.0	348.0	363.0	435.0	491.0	505.0	404.0	359.0	310.0	337.0
1959	360.0	342.0	406.0	396.0	420.0	472.0	548.0	559.0	463.0	407.0	362.0	384.5
1960	417.0	391.0	419.0	461.0	420.0	535.0	622.0	606.0	508.0	461.0	390.0	432.0

Fuente: Elaboración propia con la función **na.seasplit()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

Ejemplo 2: Realizamos la imputación dividida estacional utilizando la imputación por valor medio para el conjunto de datos tsAirgap, aplicando:

```
> na.seasplit(tsAirgap, algorithm = "mean")
```

Cuadro 4.10: Representación de la serie de tiempo imputada por división estacional utilizando imputación por valor medio para el conjunto de datos tsAirgap.

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1949	112.0000	118.0000	132.0000	129.0000	261.2222	135.0000	148.0000	148.0000	333.5000	119.0000	104.0000	118.0000
1950	115.0000	126.0000	141.0000	135.0000	125.0000	149.0000	170.0000	170.0000	333.5000	133.0000	237.5000	140.0000
1951	145.0000	150.0000	178.0000	163.0000	172.0000	178.0000	199.0000	199.0000	184.0000	162.0000	146.0000	166.0000
1952	171.0000	180.0000	193.0000	181.0000	183.0000	218.0000	230.0000	242.0000	209.0000	191.0000	172.0000	194.0000
1953	196.0000	196.0000	236.0000	235.0000	229.0000	243.0000	264.0000	272.0000	237.0000	211.0000	180.0000	201.0000
1954	204.0000	188.0000	235.0000	227.0000	234.0000	305.4000	302.0000	293.0000	259.0000	229.0000	203.0000	229.0000
1955	242.0000	233.0000	267.0000	269.0000	270.0000	315.0000	364.0000	347.0000	312.0000	274.0000	237.0000	278.0000
1956	284.0000	277.0000	256.3000	262.9091	261.2222	374.0000	413.0000	405.0000	355.0000	306.0000	271.0000	306.0000
1957	315.0000	301.0000	356.0000	348.0000	355.0000	305.4000	465.0000	467.0000	404.0000	347.0000	237.5000	336.0000
1958	340.0000	318.0000	256.3000	348.0000	363.0000	435.0000	491.0000	505.0000	404.0000	359.0000	310.0000	337.0000
1959	360.0000	342.0000	406.0000	396.0000	420.0000	472.0000	548.0000	559.0000	463.0000	407.0000	362.0000	248.8182
1960	417.0000	391.0000	419.0000	461.0000	261.2222	535.0000	622.0000	606.0000	508.0000	461.0000	390.0000	432.0000

Fuente: Elaboración propia con la función **na.seasplit()** del paquete imputeTS de R. Los valores coloreados en azul representan los valores imputados.

4.3. Paquete HotDeckImputation

Proporciona métodos de imputación Hot Deck para resolver los datos faltantes. La metodología empleada por este paquete encuentra su funcionalidad principal en la minería de datos, debido a su simplicidad computacional. Un aspecto clave de este paquete es la implementación del comúnmente recomendado límite de donantes. Sus funciones cubren aspectos claves de la imputación Hot Deck no encontrados en otros paquetes, como son la Imputación Hot Deck del vecino más cercano y CPS imputación Hot Deck secuencial.

4.3.1. Aplicación de funciones del paquete HotDeckImputation

impute.CPS_SEQ_HD (CPS Secuential Hot Deck Imputation)

Imputa los valores perdidos en cualquier variable creando clases de imputación por el método Hot Deck secuencial, a través del siguiente uso:

```
impute.CPS_SEQ_HD(DATA = NULL, covariates = NULL, initialvalues = 0,navalues = NA, modifyinplace = TRUE)
```

Argumentos:

- *DATA*: Data contiene valores perdidos, debería ser una matriz numérica.
- *covariates*: Vector que contiene las variables (columnas que deben utilizarse para crear la imputación de clases). Si es NULL *(covariates)* | *length(covariates)==0* ésta es la función predeterminada de **impute.SEQ_HD()**.
- *initialvalues*: Los valores iniciales para el proceso de puesta en marcha de la imputación. Deben ser *"integer "* y *length(initialvalues)== dim(DATA) [2]*. El valor predeterminado de 0 no es normalmente un buen valor.
- *navalues*: Código NA para cada variable que debe ser imputada. Debe ser *"integer "* y *length(initialvalues)==1 / length(initialvalues) == dim (DATA) [2]* . Por defecto es el valor NA de R.
- *modifyinplace*: ¿Debería DATA ser modificada en su lugar?

Ejemplos:

Generamos una matriz de enteros aleatorios y 2 covariables binarias, aplicando el siguiente comando:

```
> Y<-cbind(matrix(sample(0:1,replace=TRUE,size=n*2),nrow=n), matrix(sample(0:9,replace=TRUE,size=n*m),
nrow=n))
```

de dicha aplicación obtenemos la siguiente matriz $Y = \begin{bmatrix} 1&0&9&9&0\\ 0&0&1&7&7\\ 1&0&3&0&9\\ 0&1&3&8&7\\ 1&1&4&3&8\\ 1&0&0&2&5\\ 1&0&9&7&4\\ 1&1&6&3&2\\ 1&0&4&6&5\\ 0&1&9&4&5 \end{bmatrix}$

A continuación, generamos valores perdidos MCAR en todas las columnas excepto en las dos primeras y la penúltima aplicando:

```
> Y[,-c(1,2)][sample(1:length(Y[,-c(1,2)]), size=floor(pmiss*length(Y[,-c(1,2)])))]<-NA
```

de la aplicación anterior obtenemos la siguiente matriz $Y = \begin{bmatrix} 1&0&9&9&0\\ 0&0&NA&7&7\\ 1&0&NA&0&9\\ 0&1&3&8&NA\\ 1&1&4&3&8\\ 1&0&0&2&5\\ 1&0&9&7&4\\ 1&1&6&3&2\\ 1&0&4&6&5\\ 0&1&9&4&5 \end{bmatrix}$ con valores NA.

Ejemplo 1: Realizamos la imputación secuencial de la matriz Y con valores NA dentro de las clases creadas mediante la clasificación cruzada de las variables 1 y 2, aplicando:

```
> impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0, navalues=NA, modifyinplace = FALSE)
```

de la aplicación anterior se obtiene la siguiente matriz imputada $Y = \begin{bmatrix} 1&0&9&9&0\\ 0&0&0&7&7\\ 1&0&9&0&9\\ 0&1&3&8&0\\ 1&1&4&3&8\\ 1&0&0&2&5\\ 1&0&9&7&4\\ 1&1&6&3&2\\ 1&0&4&6&5\\ 0&1&9&4&5 \end{bmatrix}$ del mismo tamaño que la matriz Y de entrada con valores NA.

Ejemplo 2: Resaltando el argumento *modifyinplace* y usando el comando **cbind()** de R para mostrar los resultados de la función y los datos iniciales, aplicando:

```
> cbind(impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0,navalues=NA, modifyinplace = FALSE),Y)
```

la aplicación anterior nos devuelve la siguiente matriz imputada $Y = \begin{bmatrix} 1&0&9&9&0&1&0&9&9&0\\ 0&0&0&7&7&0&0&NA&7&7\\ 1&0&9&0&9&1&0&NA&0&9\\ 0&1&3&8&0&0&1&3&8&NA\\ 1&1&4&3&8&1&1&4&3&8\\ 1&0&0&2&5&1&0&0&2&5\\ 1&0&9&7&4&1&0&9&7&4\\ 1&1&6&3&2&1&1&6&3&2\\ 1&0&4&6&5&1&0&4&6&5\\ 0&1&9&4&5&0&1&9&4&5 \end{bmatrix}$ del mismo tamaño que la matriz Y de entrada, donde podemos observar que las columnas 8 y 10 de la matriz Y imputada siguen representando valores faltantes.

El argumento *modifyinplace= FALSE*, permite en caso de conjuntos de datos grandes que la operación computacional sea significativamente más rápida.

Ejemplo 3: Mismo procedimiento que en el ejemplo 2, excepto que en este caso el argumento *modifyinplace* está establecido en TRUE, a través de la siguiente aplicación:

```
> cbind(impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0,navalues=NA, modifyinplace = TRUE),Y)
```

la aplicación anterior nos devuelve la siguiente matriz imputada $Y = \begin{bmatrix} 1&0&9&9&0&1&0&9&9&0\\ 0&0&0&7&7&0&0&0&7&7\\ 1&0&9&0&9&1&0&9&0&9\\ 0&1&3&8&0&0&1&3&8&0\\ 1&1&4&3&8&1&1&4&3&8\\ 1&0&0&2&5&1&0&0&2&5\\ 1&0&9&7&4&1&0&9&7&4\\ 1&1&6&3&2&1&1&6&3&2\\ 1&0&4&6&5&1&0&4&6&5\\ 0&1&9&4&5&0&1&9&4&5 \end{bmatrix}$ del mismo tamaño que la matriz Y de entrada, y donde se observa que las columnas 8 y 10 representadas por la matriz Y de entrada son idénticas que las columnas 3 y 5 de la actual matriz Y imputada.

En conclusión, el método de imputación de la función **impute.CPS_SEQ_HD()** es muy rápido, dado que sólo se necesita una pasada de los datos. A través del uso del argumento *covariates*, se pueden generar datos MCAR. Por otro lado, en la aplicación del argumento *covariates*, las covariables deben estar completas (que no falten datos). Si no NA se utilizará para la construcción de clases. Este método de imputación puede no ser apropiado para algún tipo de datos, ya que la presencia de valores faltantes en las covariables no está comprobada.

impute.SEQ_HD (Sequential Hot Deck Imputation)

Resuelve los datos faltantes mediante la imputación Hot Deck secuencial, es decir, imputa los valores faltantes en cualquier variable replicando los valores observados más recientemente en esa variable. La función tiene el siguiente uso:

```
impute.SEQ_HD(DATA = NULL, initialvalues = 0, navalues = NA,modifyinplace = TRUE)
```

Argumentos:

- *DATA*: datos que contienen valores faltantes, debe ser una matriz de enteros.
- *initialvalues*: los valores iniciales para el proceso de puesta en marcha de la imputación. Debe ser "entero " y *length(initialvalues)== 1 | length(initialvalues)== dim(DATA) [2]*. El valor predeterminado 0, no es normalmente un buen valor.
- *navalues*: código NA para cada variable que debe ser imputada. Debe ser "entero " y *length(initialvalues == 1) | length(initialvalues)== dim(DATA) [2]*. Por defecto utiliza el valor NA de R.
- *modifyinplace*: ¿Debería *DATA* ser modificado en su lugar? Si *modifyinplace== FALSE* , *DATA* o más bien la variable suministrada, se edita directamente. Esto es significativamente más rápido si el conjunto de datos es grande.

Ejemplos:

Generamos una matriz de enteros aleatorios, aplicando:

```
> Y<-matrix(sample(0:9,replace=TRUE,size=n*m),nrow=n)
```

de la aplicación anterior, se obtiene la siguiente matriz $Y = \begin{bmatrix} 7 & 3 & 9 & 9 & 0 \\ 1 & 3 & 1 & 7 & 7 \\ 7 & 2 & 3 & 0 & 9 \\ 3 & 7 & 3 & 8 & 7 \\ 8 & 8 & 4 & 3 & 8 \\ 6 & 4 & 0 & 2 & 5 \\ 9 & 3 & 9 & 7 & 4 \\ 6 & 5 & 6 & 3 & 2 \\ 5 & 1 & 4 & 6 & 5 \\ 2 & 5 & 9 & 4 & 5 \end{bmatrix}$. A continuación generamos 4 valores perdidos con un patrón MCAR, aplicando:

```
> Y[-1,][sample(1:length(Y[-1,]),size=floor(pmiss*length(Y[-1,])))]<-NA
```

de la aplicación anterior obtenemos la siguiente matriz $Y = \begin{bmatrix} 7 & 3 & 9 & 9 & 0 \\ 1 & 3 & 1 & 7 & 7 \\ NA & 2 & 3 & 0 & 9 \\ 3 & 7 & 3 & 8 & 7 \\ NA & 8 & 4 & 3 & 8 \\ 6 & 4 & NA & 2 & 5 \\ 9 & 3 & 9 & 7 & 4 \\ 6 & 5 & 6 & 3 & 2 \\ 5 & 1 & 4 & 6 & 5 \\ 2 & 5 & 9 & NA & 5 \end{bmatrix}$ con valores NA.

Ejemplo 1 : Realizamos la imputación Hot Deck secuencial de la matriz Y con valores NA, aplicando:

```
> impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA, modifyinplace = FALSE)
```

de la aplicación anterior obtenemos la siguiente matriz $Y =$ $\begin{bmatrix} 7&3&9&9&0\\ 1&3&1&7&7\\ 1&2&3&0&9\\ 3&7&3&8&7\\ 3&8&4&3&8\\ 6&4&4&2&5\\ 9&3&9&7&4\\ 6&5&6&3&2\\ 5&1&4&6&5\\ 2&5&9&6&5 \end{bmatrix}$ imputada.

Ejemplo 2: Realizamos la imputación de la matriz Y con valores NA resaltando el argumento *modifyinplace* y el comando **cbind** de R, aplicando:

```
> cbind(impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA, modifyinplace = FALSE),Y)
```

de la aplicación anterior, obtenemos la siguiente matriz $Y =$ $\begin{bmatrix} 7&3&9&9&0&7&3&9&9&0\\ 1&3&1&7&7&1&3&1&7&7\\ 1&2&3&0&9&NA&2&3&0&9\\ 3&7&3&8&7&3&7&3&8&7\\ 3&8&4&3&8&NA&8&4&3&8\\ 6&4&4&2&5&6&4&NA&2&5\\ 9&3&9&7&4&9&3&9&7&4\\ 6&5&6&3&2&6&5&6&3&2\\ 5&1&4&6&5&5&1&4&6&5\\ 2&5&9&6&5&2&5&9&NA&5 \end{bmatrix}$ imputada donde las columnas 6, 8 y 9 todavía tienen valores NA.

Ejemplo 3: Mismo procedimiento que el ejemplo 2, pero este caso el argumento *modifyinplace* está establecido en TRUE:

```
> cbind(impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA, modifyinplace = TRUE),Y)
```

de la aplicación anterior obtenemos la siguiente matriz $Y =$ $\begin{bmatrix} 7&3&9&9&0&7&3&9&9&0\\ 1&3&1&7&7&1&3&1&7&7\\ 1&2&3&0&9&1&2&3&0&9\\ 3&7&3&8&7&3&7&3&8&7\\ 3&8&4&3&8&3&8&4&3&8\\ 6&4&4&2&5&6&4&4&2&5\\ 9&3&9&7&4&9&3&9&7&4\\ 6&5&6&3&2&6&5&6&3&2\\ 5&1&4&6&5&5&1&4&6&5\\ 2&5&9&6&5&2&5&9&6&5 \end{bmatrix}$ imputada donde las columnas 1 y 5 representando la matriz Y inicial son idénticas que las columnas 6 y 10 respectivamente, dado que todas las variables de la matriz Y imputada se han modificado directamente.

En conclusión, el método de imputación que aplica la función **impute.SEQ_HD()** es el más rápido, dado que sólo se necesita una pasada de los datos. Sin embargo este método no utiliza información sobre la covarianza, lo que provoca que este tipo de imputación sólo conduzca a buenos resultados cuando los datos siguen un patrón MCAR.

4.4. Paquete longitudinalData

Proporciona algunas herramientas para hacer frente a la agrupación de datos longitudinales principalmente a través de las siguientes características:

1. plotTrajmeans
2. imputation
3. qualityCriterion

4.4.1. Aplicación de la función imputation del paquete longitudinalData

Es una función que ofrece diferentes métodos para imputar los valores perdidos en datos longitudinales o una matriz, a través del siguiente uso:

```
imputation(traj,method="copyMean",lowerBound="globalMin",upperBound="globalMax")
```

Argumentos:

- *traj*: *[LongData]* o *[matrix]* trayectorias a imputar.

- *method*: *[character]*: nombre del método de imputación.
- *lowerBound*: *[character]* o *[numeric]* fija el valor más pequeño que un valor imputado puede tomar. Si se da un solo valor, este es duplicado para toda la columna. El valor especial "min "significa que el límite inferior será el valor más pequeño de la columna. El valor especial "globalMIN "significa que el límite inferior será el valor total más pequeño (de cada variable si hay varias trayectorias en las variables). El valor especial "NA "puede ser usado para imputar sin usar un límite inferior.
- *upperBound*: *[character]* o *[numeric]* fija el mayor valor que un valor imputado puede tomar. Si se da un sólo valor imputado, este es duplicado para toda la columna. El valor inicial "max "significa que el límite superior será el valor más grande de la columna. El valor especial "globalMax "significa que el límite superior será el mayor valor global (de cada variable si hay varias variables trayectorias). El valor especial "NA "se puede utilizar para imputar sin utilizar un límite superior.

Detalles:

La función **imputation()**, para cada método de imputación tiene que tratar con el valor faltante monótono (al inicio y al final de las trayectorias) y intermitente (en el medio). A continuación se ofrece una breve descripción de los métodos de imputación implementados en la función **imputation()**:

- *"linearInterpol.locf " (linear interpolation, locf)*: interpolación lineal, locf:
  - *Intermitant (Intermitente)*: los valores faltantes inmediatos que rodean los desaparecidos son embalsados por una línea.
  - *Monotnone (Monótono)*: imputado por "locf " o "nocb ".
- *"linearInterpol,global " (interpolación lineal, pendiente global)*:
  - *Intermitant (Intermitente)*: los valores inmediatos que rodean los desaparecidos son embalsados por una línea.
  - *Monotone (Monótono)*: se considera la línea que une el primer y último valor no faltante.
- *"linearInterpol.local "(interpolación lineal, pendiente global)*:
  - *Intermitant (Intermitente)*: los valores inmediatos que rodean los desaparecidos se embalsan por una línea.
  - *Monotone at start (Monótono al inicio)*: se considera la línea que une el primer y el segundo valor no faltante. El valor faltante al inicio se elige en esta línea.
  - *Monotone at end (Monótono al final)*: se considera la línea que une el último y el penúltimo valor no faltante. El valor faltante al final se elige en esta línea.
- *"linearInterpol.bisector "(interpolación lineal, bisectriz)*:
  - *Intermitant (Intermitente)*: los valores inmediatos que rodean los desaparecidos son embalsados por una línea.
  - *Monotone (Monótono)*: *linearInterpol.global* no es sensible a la variación local. El método *linearInterpol.local* puede ser demasiado sensible a un valor atípico por lo que el método *linearInterpol.bisector* ofrece una alternativa a través de la bisectriz para considerar la solución global y local. El punto se elige en las bisectrices.
- *"copyMean.locf (copia media, locf) "*: este método imputa en dos etapas. En primer lugar, utiliza *"linearInterpol.locf "* y a continuación suma a cada valor imputado una variación que haga que el valor imputado siga la forma trayectoria media.
- *"copyMean.global (copia media, pendiente global) "*: este método imputa en dos etapas. Primero usa *"linearInterpol.global "* y a continuación suma a cada valor imputado una variación que haga que el valor de la imputación siga la forma de la trayectoria media.

- *"copyMean.local (copia media, pendiente local) "*: este método imputa en dos etapas. Primero usa *"linearInterpol.local "* y a continuación suma a cada valor imputado una variación que haga que el valor de la imputación siga la forma de la trayectoria media.
- *"copyMean.bisector (copia media, bisectriz) "*: este método imputa en dos etapas. Primero usa *"linearInterpol.bisector "* y a continuación suma a cada valor imputado una variación que haga que el valor de la imputación siga la trayectoria media.
- *locf (Last Ocurrence Carried Forward)*:
  - *Intermitant and monotone at end (Intermitente y monótono al final)*: el valor no faltante anterior se desplaza hacia delante.
  - *Monotone at start (Monótono al inicio)*: el primer valor no faltante es duplicado hacia atrás (nocb).
- *nocb (Next Ocurrence Carried Backward)*:
  - *Intermitant and monotone at start (Intermitente y monótono al inicio)*: el siguiente valor no faltante se desplaza hacia atrás.
  - *Monotone at end (Monótono al final)*: el último valor no faltante es duplicado hacia delante (locf).
- *trajMean*: los valores perdidos son imputados a través de la trayectoria de la media.
- *trajMedian*: los valores perdidos son imputados a través de la trayectoria de la mediana.
- *crossMean*: el valor faltante en el tiempo t es imputado por la media de todo el valor presente en el instante t.
- *crossMedian*: el valor faltante en el tiempo t es imputado por la mediana de todo el valor presente en el instante t.
- *crossHotDeck*: cada valor faltante en el tiempo t es imputado por un valor no perdido presente en el tiempo t (elegido al azar).

Ejemplos:

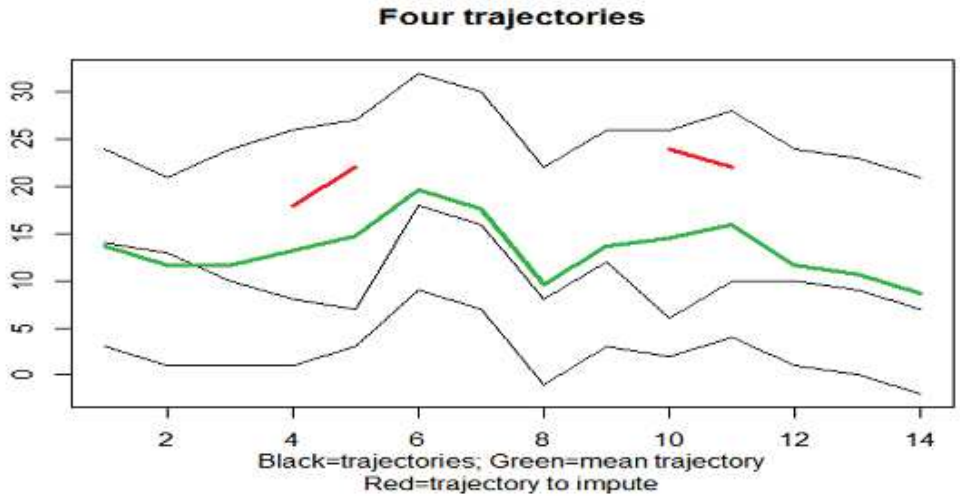
Partimos de la siguiente matriz matMissing con valores perdidos generada con el siguiente comando:

```
> par(ask=TRUE)
> timeV <- 1:14
> matMissing <- matrix( c(NA ,NA ,NA ,18 ,22 ,NA ,NA ,NA ,NA , 24 , 22 , NA , NA , NA,24 ,21 ,24 ,26 ,27 ,
32 ,30 ,22 ,26 , 26 , 28 , 24 , 23 , 21,14 ,13 , 10 , 8 , 7 ,18 ,16 , 8 ,12 , 6 , 10 , 10 , 9 , 7, 3 ,
1 , 1 , 1 , 3, 9 , 7 , -1 , 3 , 2 , 4 , 1 , 0 , -2),4,byrow=TRUE )
```

de la aplicación anterior obtenemos la siguiente matriz $matMissing = \begin{bmatrix} NA & NA & NA & 18 & 22 & NA & NA & NA & NA & 24 & 22 & NA & NA & NA \\ 24 & 21 & 24 & 26 & 27 & 32 & 30 & 22 & 26 & 26 & 28 & 24 & 23 & 21 \\ 14 & 13 & 10 & 8 & 7 & 18 & 16 & 8 & 12 & 6 & 10 & 10 & 9 & 7 \\ 3 & 1 & 1 & 1 & 3 & 9 & 7 & -1 & 3 & 2 & 4 & 1 & 0 & -2 \end{bmatrix}$ con valores NA.

Ejemplo 1: Ilustramos las trayectorias de los valores de la matriz matMissing:

Figura 4.1: Representación de las trayectorias de la matriz matMissing.



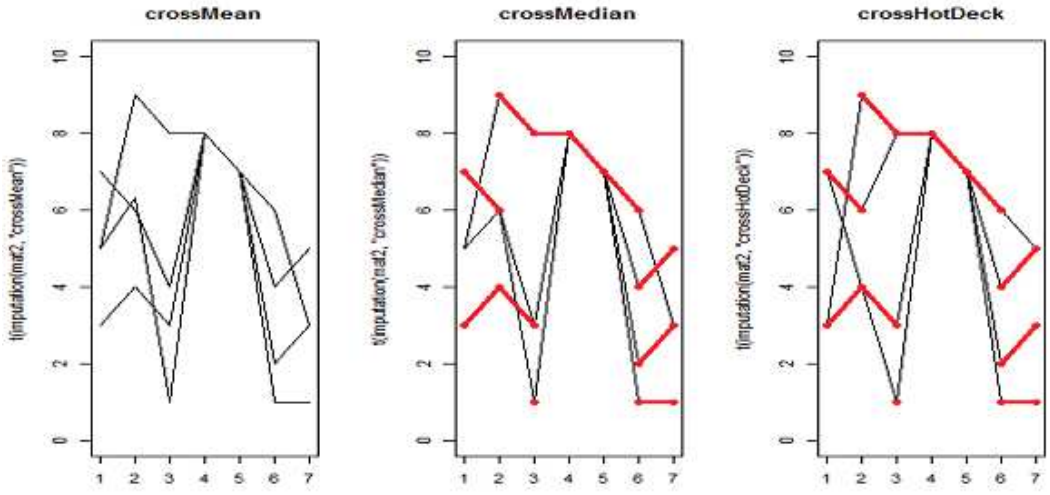
Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. Las líneas en negro representan las trayectorias de los valores por filas de la matriz matMissing, la línea verde representa la trayectoria de la media y la línea roja la trayectoria de la imputación.

Ejemplo 2: Representación de métodos que utilizan información transversal (métodos cruzados):
Generamos la siguiente matriz mat2 con valores NA aplicando:

```
> mat2 <- matrix(c(NA, 9, 8, 8, 7, 6,NA,7, 6,NA,NA,NA, 4,5, 3, 4, 3,NA,NA, 2,3, NA,NA, 1,NA,NA, 1,1),4,7,
    byrow=TRUE)
```

de la aplicación anterior obtenemos la siguiente matriz $mat2 = \begin{bmatrix} NA & 9 & 8 & 8 & 7 & 6 & NA \\ 7 & 6 & A & NA & NA & 4 & 5 \\ 3 & 4 & 3 & NA & NA & 2 & 3 \\ NA & NA & 1 & NA & NA & 1 & 1 \end{bmatrix}$ con valores NA.

Figura 4.2: Representación de métodos que utilizan información transversal para la matriz mat2.



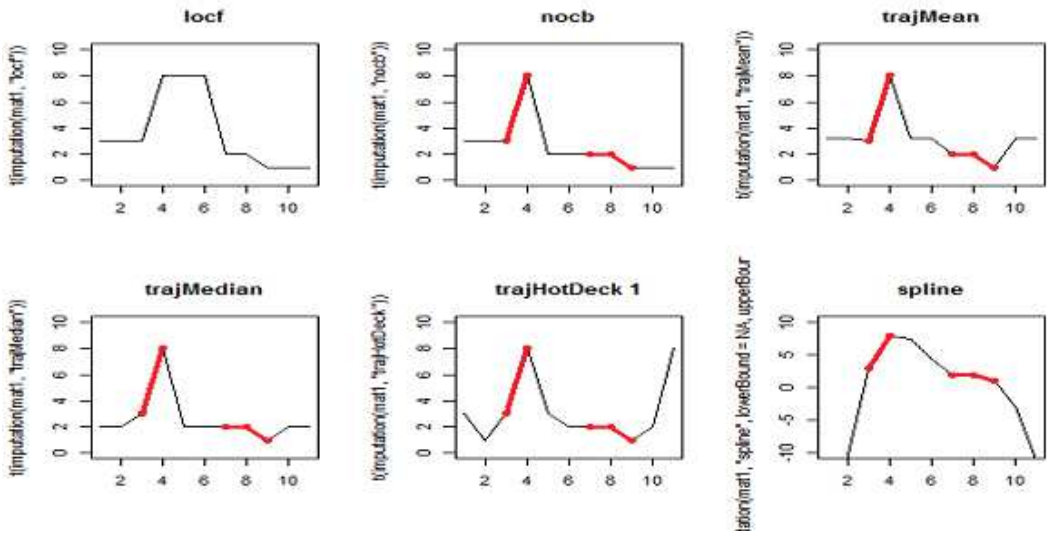
Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. El primer gráfico de izquierda a derecha representa el método crossMean, el segundo el crossMedian y el tercero el crossHotDeck. Las líneas en rojo representan la trayectoria de la imputación.

Ejemplo 3: Representación de métodos que utilizan la información de la trayectoria:
Generamos la siguiente matriz mat1 aplicando:

```
> mat1 <- matrix(c(NA,NA,3,8,NA,NA,2,2,1,NA,NA),1,11)
```

de la aplicación anterior obtenemos la siguiente matriz $mat1 = [NA\ NA\ 3\ 8\ NA\ NA\ 2\ 2\ 1\ NA\ NA]$ con valores NA.

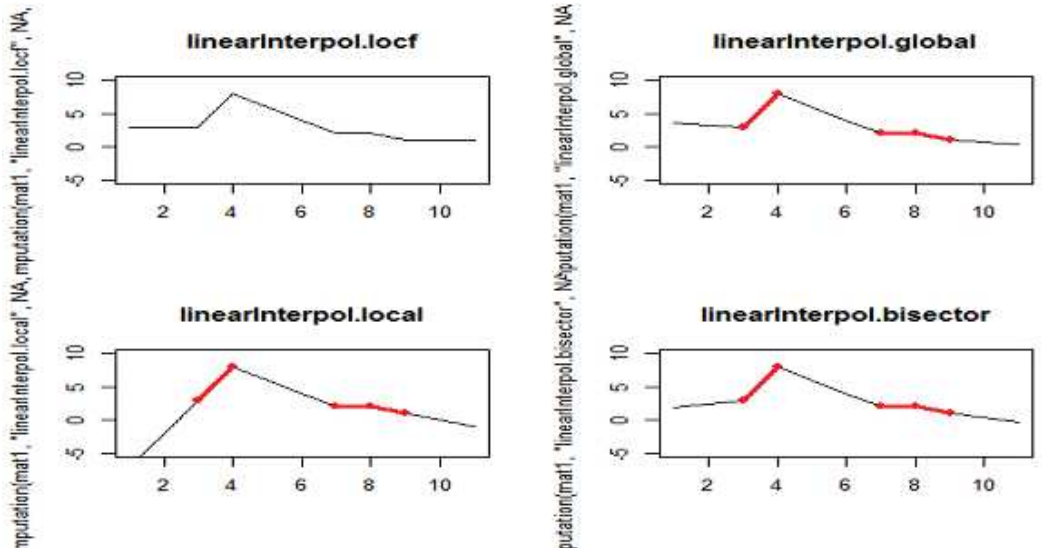
Figura 4.3: Representación de métodos que utilizan información transversal para la matriz mat1.



Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (locf, nocb, trajMean). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (trajMedian, trajHotDeck, spline). Las líneas en rojo representas la trayectoría de la imputación.

Ejemplo 4: Representación de diferentes métodos generados a través de interpolaciones lineales para la matriz mat1:

Figura 4.4: Representación de métodos generados a través de interpolaciones lineales para la matriz mat1.



Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (linearInterpol.locf y linearInterpol.global). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (linearInterpol.local y linearInterpol.bisector). Las líneas en rojo representan la trayectoría de la imputación.

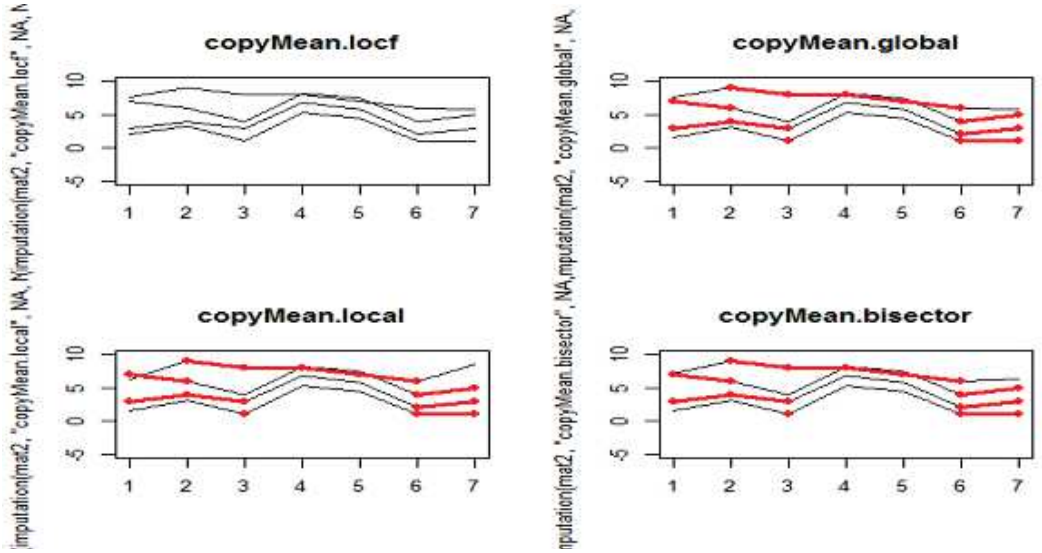
Ejemplo 5: Representación de métodos que utilizan copyMean:

Generamos una matriz mat3 con valores NA aplicando:

```
> mat3 <- matrix(c(NA, 9, 8, 8, 7, 6,NA,7, 6,NA,NA,NA, 4,5,3, 4, 3,NA,NA, 2,3,NA,NA, 1,NA,NA, 1,1),
4,7,byrow=TRUE)
```

de la aplicación anterior obtenemos la siguiente matriz $mat3 = \begin{bmatrix} NA & 9 & 8 & 8 & 7 & 6 & NA \\ 7 & 6 & NA & NA & NA & 4 & 5 \\ 3 & 4 & 3 & NA & NA & 2 & 3 \\ NA & NA & 1 & NA & NA & 1 & 1 \end{bmatrix}$ con valores NA.

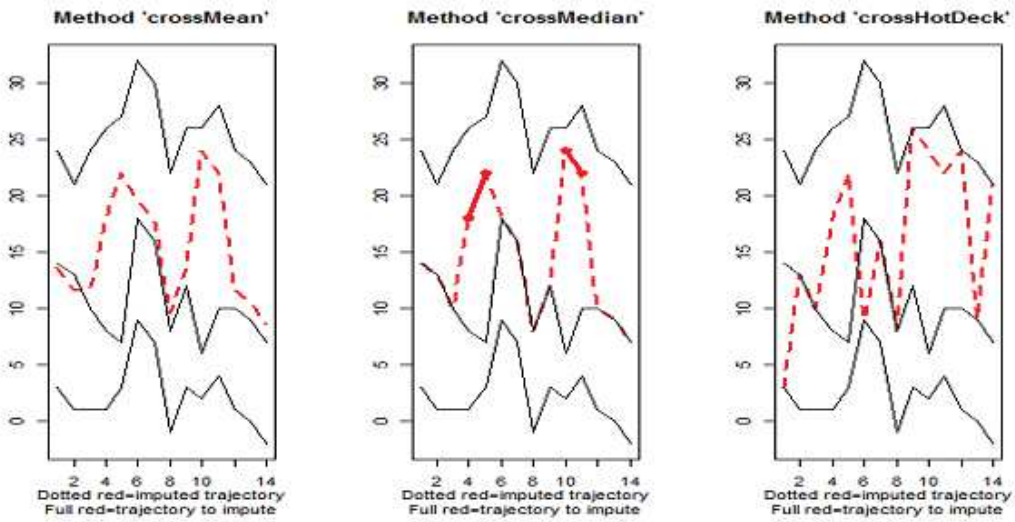
Figura 4.5: Representación de métodos que utlizan copyMean para la matriz mat3.



Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (copyMean.locf y copyMean.global). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (copyMean.local y copyMean.bisector). Las líneas en rojo representan la trayectoria de la imputación.

Ejemplo 6: Representación de métodos que utilizan crossMean:

Figura 4.6: Representación de métodos que utlizan crossMean para la matriz matMissing.

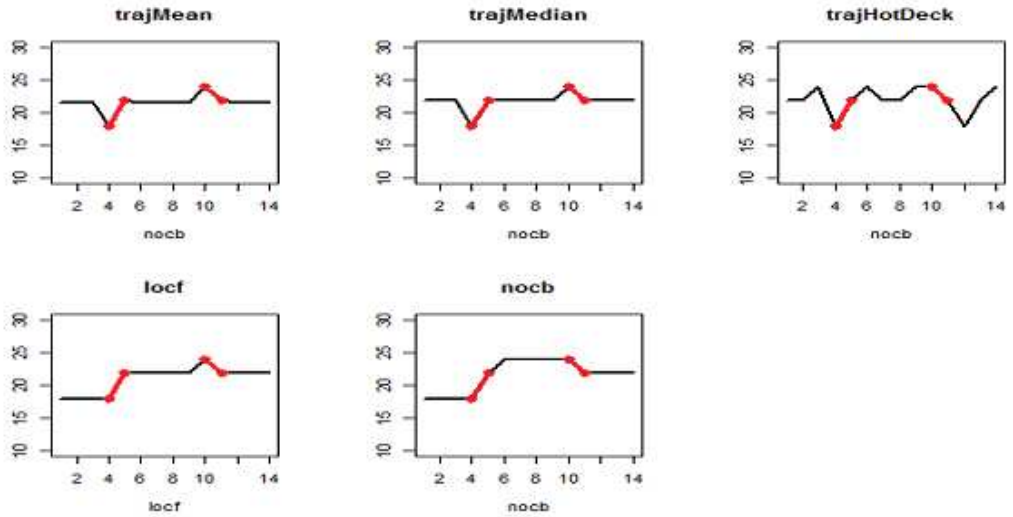


Fuente: Elaboración propia con la función **matplot()** de R para el paquete longitudinalData. El primer gráfico de izquierda a derecha representa el método crossMean, el segundo el crossMedian y el tercero el crossHotDeck. Las líneas discontinuas en rojo representan la trayectoría de la imputación.

Ejemplo 7: Representación de métodos que utilizan trayectorias:

- Caso I:

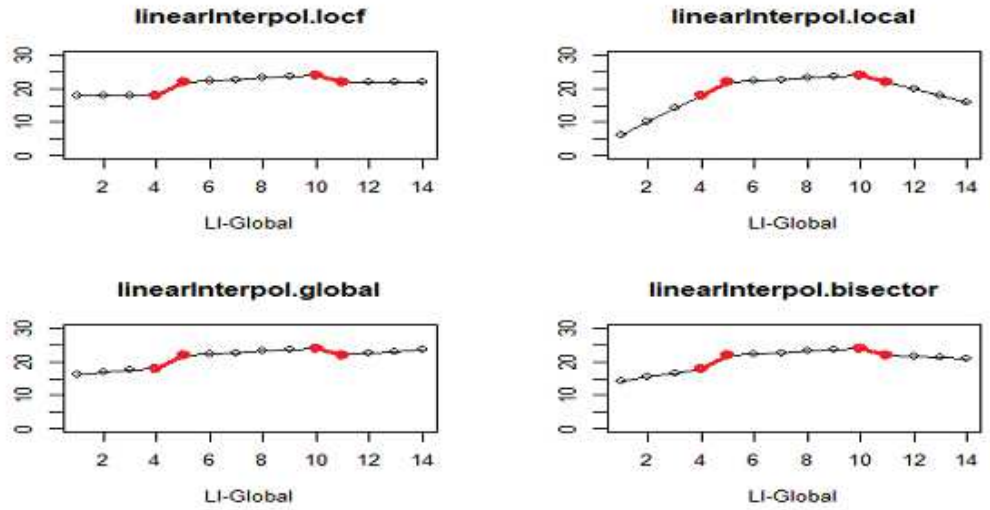
Figura 4.7: Representación de métodos que utlizan trayectorias para la matriz matMissing.



Fuente: Elaboración propia con la función **plot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (trajMean, trajMedian y trajHotDeck). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (locf y nocb). Las líneas en rojo representan la trayectoría de la imputación.

- Caso II:

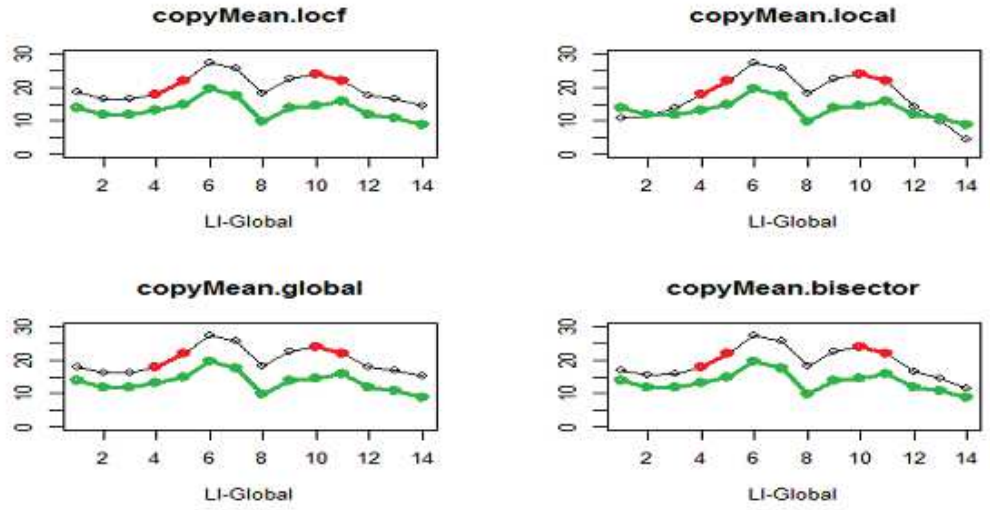
Figura 4.8: Representación de métodos que utlizan trayectorias para la matriz matMissing.



Fuente: Elaboración propia con la función **plot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (linearInterpol.locf y linearInterpol.local). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (linearInterpol.global y linearInterpol.bisector). Las líneas en rojo representan la trayectoría de la imputación.

- Caso III:

Figura 4.9: Representación de métodos que utlizan trayectorias para la matriz matMissing.



Fuente: Elaboración propia con la función **plot()** de R para el paquete longitudinalData. En la primera fila están representados respectivamente de izquierda a derecha los siguientes métodos (copyMean.locf y copyMean.local). En la segunda fila están representados respectivamente de izquierda a derecha los métodos (copyMean.global y copyMean.bisector). Las líneas en rojo representan la trayectoría de la imputación y las líneas en verde las trayectorías de la media.

4.5. Paquete missForest

El paquete missForest (Non parametric Missing Value imputation using Random Forest) permie imputar valores perdidos para el caso de datos de tipo mixto, es decir imputa datos continuos y/o categóricos a través de interacciones complejas y relaciones no lineales. Además proporciona una estimación de error de imputación fuera de bolsa (out of bag (OOB)).

4.5.1. Datos de aplicación del paquete missForest

En la aplicación de las funciones del paquete missForest, utilizaremos el conjunto de datos iris de Fisher, un conjunto de datos clásico que recoge la medida en cm de las variables longitud y anchura de sépalo y longitud y anchura de pétalo para flores de tres especies diferentes (iris setosa, versicolor y virginica). El conjunto de datos esta formado por un marco compuesto por 150 casos (filas) y 5 variables (columnas) denominadas Sepal.Length, Sepal.Width, Petal.Length, Petal.Width y Species.

Las variables o atributos que se miden de de cada flor son del siguiente tipo:

1. El tipo de flor como variable categórica.
2. El largo y el ancho del pétalo en cm. como variables numéricas.
3. El largo y el ancho del sépalo en cm. como variables numéricas.

A continuación se ofrece una pequeña visualización de las primeras seis observaciones del conjunto de datos:

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

4.5.2. Aplicación de las funciones del paquete missForest

missForest (Non parametric Missing Value Imputation using Random Forest)

Realiza la imputación de valores no paramétricos usando árboles aleatorios para imputar valores perdidos en el caso de datos de tipo mixto, a través del siguiente uso:

```
missForest(xmis, maxiter = 10, ntree = 100, variablewise = FALSE,decreasing = FALSE, verbose = FALSE,
mtry = floor(sqrt(ncol(xmis))), replace = TRUE,classwt = NULL, cutoff = NULL, strata = NULL,sampsize = NULL,
 nodesize = NULL, maxnodes = NULL,xtrue = NA, parallelize = c('no', 'variables', 'forests'))
```

Argumentos:

- *xmis*: una matriz de datos con valores faltantes. Las columnas corresponden a las variables y las filas a las observaciones.
- *maxiter*: el número máximo de iteracciones a realizar, dado el criterio de parada.
- *variablewise*: lógico. Si es "TRUE " se devuelve el error OOB para cada variable por separado. Esto puede ser útil como control de fiabilidad para las variables imputadas para un subsiguiente análisis de datos.
- *decreasing*: lógico. Si es "FALSE "entonces las variables son creadas en orden creciente de entradas perdidas.
- *verbose*: lógico. Si es "TRUE "el usuario recibe una salida adicional entre iteracciones, con el error de imputación estimado a través del tiempo de ejecución y si se suministra la matriz completa de datos se recibe el verdadero error de imputación.
- *mtry*: número de variables aleatoriamente muestreadas en cada división. Este argumento es directamente suministrado por la función **randomForest()**. Hay que tener en cuenta que el valor predeterminado es sqrt(p) para variables categóricas y continuas donde p es el número de variables del argumento *xmis*.
- *replace*: lógico. Si es "TRUE "se utiliza muestreo bootstrap (con reemplazos).
- *classwt*: lista de probabilidades a priori de las clases en las variables categóricas. Esto es equivalente al argumento *randomForest*, sin embargo en este caso el usuario tiene que establecerla para todas las variables en el conjunto de datos (para las variables continuas se establece "NULL ").
- *cutoff*: lista de puntos de corte de clase para cada variable categórica. Igual que con el argumento *classwt* para las variables continuas se establece en "1 ".
- *strata*: lista de variables (factor) utilizadas para el muestreo estratificado. Igual que con el argumento *classwt* para las variables continuas se establece en "NULL ".
- *sampsize*: lista de tamaño(s) de muestra. Esto es equivalente al argumento *randomForest*, sin embargo, en este caso el usuario tiene que establecer los tamaños para todas las variables.
- *nodesize*: tamaño mínimo de los nodos terminales. Tiene que ser un vector de longitud 2, donde el primero representa la entrada del número de las variables continuas y el segundo el número de las variables categóricas. El valor predeterminado es 1 para continuas y 5 para variables categóricas.
- *maxnodes*: número máximo de nodos terminales.
- *xtrue*: opcional a través de la matriz completa de datos. Esto se puede aplicar para probar el rendimiento. Al proporcionar la matriz de datos completa el argumento *verbose* mostrará la verdadera imputación después de cada iteración y la salida también contendrá el error de imputación.
- *parallelize*: indica si la función **missForest()** debe ejecutarse en paralelo. El valor predeterminado es "no ". Si se utiliza "variables " los datos son divididos en trazos de tamaño igual al número de núcleos registrados en el paralelo backend (backend hace referencia al conjunto de aplicaciones que gestionan el proceso y almacenamiento de los datos en la aplicación de la función **missForest()**) . Si se utiliza "forests "el número total de árboles aleatorios se divide de la misma manera.

Detalles:

La función **missForest()** después de cada iteracción evalúa la diferencia entre la matriz de datos anterior y la nueva matriz de datos imputados para las partes continuas y categóricas. El criterio de detención está definido de manera que el proceso de imputación se detenga tan pronto como ambas diferencias se hayan vuelto más grandes. En el caso de que sólo exista un tipo de variable el cálculo se detiene tan pronto como la diferencia correspondiente se incrementa por primera vez.

Ejemplos:

Ejemplo 1: Realizamos una imputación de valor faltante no paramétrico en datos de tipo mixto para los datos de iris:

Abrimos y visualizamos las primeras observaciones de los datos de iris aplicando los siguientes comandos:

```
> data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Los datos contienen cuatros variables continuas y una variable categórica. A continuación producimos artificialmente los valores perdidos con la función **prodNA()** a través del siguiente uso:

```
> iris.mis <- prodNA(iris, noNA = 0.2)
> summary(iris.mis)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.100   Min.   :0.100   setosa    :40
 1st Qu.:5.200   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:38
 Median :5.800   Median :3.000   Median :4.450   Median :1.300   virginica :44
 Mean   :5.878   Mean   :3.062   Mean   :3.905   Mean   :1.222   NA's      :28
 3rd Qu.:6.475   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.900
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
 NA's   :28      NA's   :29      NA's   :32      NA's   :33
```

Imponemos los valores faltantes proporcionados por la función **prodNA()** y ilustramos las las iteracciones a través del siguiente uso:

```
> iris.imp <- missForest(iris.mis, xtrue = iris, verbose = TRUE)
  missForest iteration 1 in progress...done!
    error(s): 0.1512033 0.03571429
    estimated error(s): 0.1541084 0.04098361
    difference(s): 0.01449533 0.1533333
    time: 0.33 seconds

  missForest iteration 2 in progress...done!
    error(s): 0.1482248 0.03571429
    estimated error(s): 0.1402145 0.03278689
    difference(s): 9.387853e-05 0
    time: 0.48 seconds

  missForest iteration 3 in progress...done!
    error(s): 0.1567693 0.03571429
    estimated error(s): 0.1384038 0.04098361
    difference(s): 6.271654e-05 0
    time: 0.38 seconds

  missForest iteration 4 in progress...done!
    error(s): 0.1586195 0.03571429
    estimated error(s): 0.1419132 0.04918033
    difference(s): 3.02275e-05 0
    time: 0.34 seconds

  missForest iteration 5 in progress...done!
    error(s): 0.1574789 0.03571429
    estimated error(s): 0.1397179 0.04098361
    difference(s): 4.508345e-05 0
    time: 0.36 seconds
```

La imputación termina después de 5 iteraciones arrojando un NMRSE. En el paquete missForest el error cuadrático medio normalizado se define como $\sqrt{\frac{mean\left(\left(X_{\text{true}}-X_{\text{imp}}\right)^2\right)}{var(X_{\text{true}})}}$, donde X_{true} es la matriz de datos completa y X_{imp} la matriz de datos imputados. Los argumentos "mean " y "var "se utilizan como notación corta para referirse a la media empírica y varianza respectivamente. Se obtiene un PFC (proporción de falsamente clasificados que se calcula sobre los valores categóricos faltantes) de 0.049. El NMRSE final estimado es de 0.159 y el PFC de 0.036.

mixError (Compute Imputation Error for Mixed type Data)

Proporciona los siguientes valores: en el caso de que la matriz de datos de entrada sólo tenga variables continuas proporciona el error cuadrático medio normalizado (NRMSE). Para el caso de matrices de datos con sólo variables categóricas proporciona el PFC y si los datos están compuestos por una matriz de tipo mixto suministra ambas medidas:

```
mixError(ximp, xmis, xtrue)
```

Argumentos:

- *ximp*: matriz de datos imputados con variables en las columnas y observaciones en las filas. Teniendo en cuenta que no debe haber valores faltantes.
- *xmis*: matriz de datos con valores faltantes.
- *xtrue*: matriz de datos completa. Teniendo en cuenta que no debe haber valores faltantes.

Ejemplos:

Ejemplo 1: Cálculo del error de imputación para los datos de data iris.
Abrimos y visualizamos los datos de data iris con los siguientes comandos:

```
> data(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

A continuación producimos los valores perdidos artificialmente usando la función **prodNA()**:

```
> iris.mis <- prodNA(iris, noNA = 0.2)
```

Imponemos los valores faltantes aplicando la función **missForest()**:

```
> iris.imp <- missForest(iris.mis)
```

Calculamos el error de imputación verdadero aplicando:

```
> err.imp <- mixError(iris.imp$ximp, iris.mis, iris)
```

obteniendo los siguientes valores un NMRSE de 0.153 y un PFC de 0.09.

4.6. Paquete ForImp

Permite realizar imputación directa mediante función para la imputación de valores faltantes en matrices de datos ordinales llamada **ForImp()** y otras funciones para generar datos ordinales o imputar valores faltantes.

4.6.1. Aplicación de las funciones del paquete ForImp

ForImp (Forward Imputation procedure)

Realiza imputación directa de datos perdidos a través del siguiente uso:

```
ForImp(mat, p=2)
```

Argumentos:

- *mat*: una matriz / marco de datos.
- *p*: el parámetro para calcular la distancia de Minkoviski utilizada en el vecino más cercano. El argumento *p* puede ser cualquier número positivo (p=2 de la distancia euclidea), si se introduce un número negativo el procedimiento utilizará la distancia máxima (o norma suprema).

Detalles:

La función **ForImp()** implementa el algoritmo Forward Imputation en una matriz ordinal de datos con valores faltantes. El algoritmo alterna análisis de componentes principales no lineales en un subconjunto de los datos sin datos faltantes e imputaciones secuenciales de los valores faltantes por el método del vecino más cercano.

Ejemplos:

Ejemplo 1: Aplicación de la función **ForImp()**:

1. Generamos una matriz de correlaciones llamada sigma a través del siguiente comando:

```
> sigma<-matrix(c(1,0.5,0.5,0.5,0.5,1,0.5,0.5,0.5,0.5,1,0.5,0.5,0.5,0.5,1),4,4)
```

de la aplicación anterior obtenemos la siguiente matriz $sigma = \begin{bmatrix} 1{,}0 & 0{,}5 & 0{,}5 & 0{,}5 \\ 0{,}5 & 1{,}0 & 0{,}5 & 0{,}5 \\ 0{,}5 & 0{,}5 & 1{,}0 & 0{,}5 \\ 0{,}5 & 0{,}5 & 0{,}5 & 1{,}0 \end{bmatrix}$.

2. Generamos una matriz de 10 filas y 4 columnas de una matriz normal multivariante denominada matc con el siguiente comando:

```
> matc<-rmvnorm(n=10, mean=rep(0,4), sigma=sigma)
```

de la aplicación anterior obtenemos la siguiente matriz $matc = \begin{bmatrix} 0{,}13407359 & -1{,}27572122 & 1{,}03688105 & 0{,}1089437 \\ 2{,}09322376 & 0{,}89319133 & 0{,}05494738 & 0{,}9888041 \\ -1{,}17172302 & -1{,}39766893 & -0{,}30513264 & -0{,}8246709 \\ -0{,}23580642 & -0{,}18402077 & -0{,}65344228 & -0{,}6386975 \\ -0{,}07083143 & 0{,}85778759 & -1{,}05257013 & 0{,}4447377 \\ 0{,}55486507 & 1{,}07115871 & 0{,}10434309 & 0{,}5810764 \\ 0{,}64617168 & 0{,}07368472 & 1{,}31139583 & 1{,}2778329 \\ 0{,}83795207 & 1{,}46488695 & 0{,}73773579 & -0{,}5598608 \\ -1{,}03720754 & -1{,}49777956 & -0{,}96659248 & -1{,}0705132 \\ -0{,}26856204 & -0{,}94246133 & -0{,}18659924 & -0{,}7612037 \end{bmatrix}$

3. Transformamos los valores númericos de la matriz matc en categorías ordinales aplicando el comando:

```
> mato<-transfmatcat(matc,4)
```

de la aplicación anterior se obtiene la siguiente matriz $mato = \begin{bmatrix} 2 & 1 & 4 & 3 \\ 4 & 4 & 3 & 4 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 4 & 1 & 3 \\ 3 & 4 & 3 & 4 \\ 3 & 3 & 4 & 4 \\ 4 & 4 & 4 & 2 \\ 1 & 1 & 1 & 1 \\ 2 & 1 & 2 & 1 \end{bmatrix}$

4. Estabelecemos el número de valores perdidos deseados aplicando el comando:

```
> nummissing<-5
```

5. Creamos los valores faltantes al azar aplicando:

```
> mat<-missingmat(mato, nummissing, pattern="r")
```

de la aplicación anterior se obtiene la siguiente matriz $mat = \begin{bmatrix} 2 & 1 & 4 & 3 \\ 4 & 4 & 3 & 4 \\ 1 & 1 & 2 & 1 \\ NA & 2 & 1 & 1 \\ 2 & 4 & 1 & 3 \\ 3 & 4 & 3 & NA \\ 3 & 3 & 4 & 4 \\ 4 & 4 & 4 & NA \\ 1 & 1 & 1 & NA \\ 2 & 1 & NA & 1 \end{bmatrix}$ con valores NA.

6. Usamos la función **FormImp()** para imputar los valores faltantes:

```
> mati<-ForImp(mat)
```

de la aplicación anterior obtenemos la siguiente matriz $mati = \begin{bmatrix} 2 & 1 & 4 & 3 \\ 4 & 4 & 3 & 4 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 4 & 1 & 3 \\ 3 & 4 & 3 & 4 \\ 3 & 3 & 4 & 4 \\ 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 2 & 1 & 4 & 1 \end{bmatrix}$ imputada.

7. Número de imputaciones correctas aplicando el comando:

```
> nummissing-sum(mati!=mato)
[1] 3
```

meanimp (Mean Imputation)

Implementa la imputación media incondicional en una matriz numérica con valores faltantes, sustituyendo a cada valor perdido la media aritmética de la variable correspondiente a través del siguiente uso:

```
meanimp(mat)
```

Argumentos:

- *mat*: una matriz numérica.

Ejemplos:

Ejemplo 1: Aplicación de la función **meanimp**():

1. Generamos una matriz con valores perdidos a través de los siguientes comandos:

```
> mat<-matrix(ceiling(runif(n*m)*4),n,m)
> matm<-mat
> matm[1,3]<-NA
> matm[9:10,1]<-NA
> matm
```

de la aplicación de los comandos anteriores obtenemos la siguiente matriz $matm = \begin{bmatrix} 2 & 1 & NA \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ NA & 2 & 4 \\ NA & 4 & 2 \end{bmatrix}$ con valores NA.

2. Aplicamos la función **meanimp()** a través del siguiente uso:

```
> medianimp(matm)
```

de la aplicación anterior obtenemos la siguiente matriz $matm = \begin{bmatrix} 2 & 1 & 2 \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ 3 & 2 & 4 \\ 3 & 4 & 2 \end{bmatrix}$ imputada.

meadianimp (Median Imputation)

Implementa la imputación mediana en una matriz de datos ordinales con valores faltantes. Sustituye a cada valor perdido la mediana de la variable correspondiente:

```
medianimp(mat)
```

Argumentos:

- *mat*: una matriz de valores ordinales, ordenada según la escala de Likert (1, 2, 3,....).La escala de Likert es una escala psicométrica comúnmente utilizada en cuestionarios y es la escala de uso más amplio en encuestas para la investigación, en las que se especifica el nivel de acuerdo o desacuerdo con una declaración (elemento, ítem o reactivo o pregunta).

Ejemplos:

Ejemplo 1: Aplicación de la función **medianimp()**:

1. Generamos una matriz con valores NA aplicando los siguientes comandos:

```
> n<-10
> m<-3
> mat<-matrix(ceiling(runif(n*m)*4),n,m)
> matm<-mat
> matm[1,3]<-NA
> matm[9:10,1]<-NA
```

de la aplicación de los comandos anteriores obtenemos la siguiente matriz $matm = \begin{bmatrix} 2 & 1 & NA \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ NA & 2 & 4 \\ NA & 4 & 2 \end{bmatrix}$ con valores NA.

2. A continuación aplicamos la función **medianimp()** a través del siguiente uso:

```
> medianimp(matm)
```

de la aplicación de la función **medianimp()** se obtiene la siguiente matriz $matm =$ $\begin{bmatrix} 2 & 1 & 2 \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ 3 & 2 & 4 \\ 3 & 4 & 2 \end{bmatrix}$ imputada.

modeimp (Mode Imputation)

Implementa la imputación a través de la moda en una matriz de datos categóricos u ordinales con valores faltantes; la función sustituye a cada valor perdido la moda de la variable correspondiente a través del siguiente uso:

```
modeimp(mat)
```

Argumentos:

- *mat*: una matriz de valores categóricos u ordinales, codificados como valores enteros (1, 2, 3,....).

Ejemplos:

Ejemplo 1: Aplicación de la función **modeimp()**:

1. Generamos una matriz con valores faltantes aplicando los siguientes comandos:

```
> n<-10
> m<-3
> mat<-matrix(ceiling(runif(n*m)*4),n,m)
> matm<-mat
> matm[1,3]<-NA
> matm[9:10,1]<-NA
```

de la aplicación de los comandos anteriores se obtiene la siguiente matriz $matm =$ $\begin{bmatrix} 2 & 1 & NA \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ NA & 2 & 4 \\ NA & 4 & 2 \end{bmatrix}$ con valores NA.

2. A continuación aplicamos la función **modeimp()** a través del siguiente uso:

```
> modeimp(mat)
```

de la aplicación de la función **modeimp()** se obtiene la siguiente matriz $matm =$ $\begin{bmatrix} 2 & 1 & 4 \\ 2 & 1 & 1 \\ 3 & 3 & 3 \\ 4 & 2 & 1 \\ 1 & 4 & 2 \\ 4 & 2 & 2 \\ 4 & 3 & 1 \\ 3 & 4 & 2 \\ 3 & 2 & 4 \\ 1 & 4 & 2 \end{bmatrix}$ imputada.

4.7. Paquete BaBoon

El paquete BaBoon incluye dos variantes del Boostrap Bayesiano para la comparación predictiva de la media con el objetivo de multiplicar y imputar datos faltantes. La primera variante que emplea es una imputación variable por variable que combina la regresión secuencial y el método de comparación predicitva media (PMM) que se extiende para datos categóricos no ordenados. La segunda variante también se basa en el PMM, pero en este caso el foco está en imputar varias variables al mismo tiempo. Ambas variantes se pueden ejecutar como " imputación única ", en caso de que el objetivo del análisis sea de carácter puramente descriptivo.

4.7.1. Aplicación de las funciones del paquete BaBoon

BBPMM (Imputation though Bayesian Bootstrap Predictive Mean Matching)

Realiza la imputación múltiple para variables de escala mixta usando una cadena de aproximación de ecuaciones (Boostrap Bayesiano) a través de la comparación predictiva de la media.

```
BBPMM(Data, M=10, nIter=10, outfile=NULL, ignore=NULL,vartype=NULL, stepmod="stepAIC", maxit.multi=3,
maxit.glm=25,maxPerc = 0.98, verbose=TRUE, setSeed, chainDiagnostics=TRUE, ...)
```

Argumentos:

- *Data*: marco o matriz de datos parcialmente incompleta.
- *M*: número de imputaciones múltiples. Si M = 1, no se lleva a cabo ningún proceso Boostrap Bayesiano. Este argumento está predeterminado en 10.
- *nIter*: número de iteraciones del algoritmo de ecuaciones encadenadas antes de que el conjunto de datos sea almacenado como un "conjunto de datos imputado ". El número de iteraciones se puede seleccionar utilizando un índice de monotonicidad de datos (mediante la función dmi). El argumento tiene el valor 10 como predeterminado.
- *outfile*: cadena de caracteres que especifica la ruta y el nombre del archivo para los conjuntos de datos imputados. Si *outfile = NULL* (predeterminado), no se almacena ningún conjunto de datos.
- *ignore*: un carácter o vector numérico que especifica posiciones que se van a excluir del modelo y proceso de imputación. Si *ignore = NULL* (valor predeterminado), todas las variables de *Data* se utilizan en la imputación.
- *vartype*: un vector de caracteres que señala la clase de variable en *Data* (sin las variables definidas por el argumento *ignore*) en"M "para escala métrica o "C "para categóricos. El valor predeterminado (NULL) se hace cargo de las clases de datos.
- *stepmod*: realiza la selección de variables para cada modelo de imputación basada en el criterio de información hacia atrás de Schwarz (Bayes). El valor predeterminado es "stepAIC ".
- *maxit.multi*: argumento importado del paquete nnet que especifica el número máximo de iteraciones para la estimación del modelo logit multinomial. El valor por defecto es 3.
- *maxit.glm*: argumento para la especificación del número máximo de iteracciones para el logit binomial (es decir glm). El valor predeterminado es 25.
- *maxPerc*: el porcentaje máximo de la categoría de una variable que se permite tener para intentar la imputación regular. Si una variable tiene distribución de Dirac, es decir, casi no tiene ninguna varianza, la imputación se lleva a cabo por el método de imputación simple Hot Deck. El valor predeterminado es 0.98.
- *verbose*: el algoritmo imprime información sobre números de imputación e iteración. El valor predeterminado es TRUE.
- *setSeed*: argumento opcional para arreglar el generador de números pseudo-aleatorios, lo que permiten obtener resultados reproducibles.
- *chainDiagnostics*: argumento que especifica si las cadenas de Monte Carlo para diagnósticos adicionales deben ser de regresión también. El valor predeteminado es TRUE.
- ....: otros argumentos de otras funciones.

Ejemplos:

Ejemplo 1: Aplicación de la función **BPPMM()**:

1. Generamos un conjunto de muestra con variables no normales aplicando los siguientes comandos:

```
> set.seed(1000)
> n <- 50
> x1 <- round(runif(n,0.5,3.5))
> x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
> x3 <- round(rnorm(n,0,3))
> y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
> y1 <- ifelse(y1<1,1,y1)
> y1 <- as.factor(ifelse(y1>4,5,y1))
> y2 <- x1+rnorm(n,0,0.5)
> y3 <- round(x3+rnorm(n,0,2))
> data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
> misrow1 <- sample(n,20)
> misrow2 <- sample(n,15)
> misrow3 <- sample(n,10)
> is.na(data1[misrow1, 4]) <- TRUE
> is.na(data1[misrow2, 5]) <- TRUE
> is.na(data1[misrow2, 6]) <- TRUE
```

de la aplicación de los comandos anteriores obtenemos el siguiente conjunto de datos data1 con valores NA:

Cuadro 4.11: Representación del conjunto de datos data1 con valores NA.

	x_1	x_2	x_3	y_1	y_2	y_3
1	1	1	2	NA	0.6639219	0
2	3	1	-5	NA	2.5831713	-1
3	1	1	1	NA	0.3330788	3
4	3	1	2	NA	NA	NA
5	2	1	4	4	1.6621906	4
6	1	1	-1	2	0.7446018	2
7	3	1	2	4	3.3974445	4
8	2	1	-2	NA	NA	NA
9	1	1	-1	NA	NA	NA
10	1	1	-5	1	0.9963446	-4
11	2	2	1	2	1.7431429	1
12	3	2	-1	NA	3.5596462	-2
13	1	2	3	NA	NA	NA
14	3	2	-2	1	2.3046369	0
15	3	2	-4	3	NA	NA
16	1	2	-2	2	1.1286590	-3
17	2	2	4	NA	2.2457760	2
18	2	2	1	3	NA	NA
19	1	2	0	1	1.4485750	1
20	2	2	-1	1	1.4086837	0
21	1	2	4	NA	0.8234375	7
22	3	2	1	4	3.6116357	2
23	1	2	0	2	1.8514327	0
24	1	2	-4	1	1.0166672	-3
25	2	2	-3	2	2.3555955	-5
26	3	2	1	NA	NA	NA
27	2	2	-3	NA	2.3986952	-2
28	3	2	1	NA	3.0505409	-2
29	1	2	-3	1	NA	NA
30	1	2	0	1	NA	NA
31	2	2	3	3	1.0894889	3
32	1	2	-5	NA	NA	NA
33	2	2	1	NA	NA	NA
34	3	2	8	5	3.0655355	10
35	2	2	-3	1	1.4605429	-4
36	3	3	2	NA	3.0929557	2
37	1	3	-1	1	1.4620976	1
38	3	3	4	5	3.9982715	4
39	2	3	4	4	1.8510383	0
40	1	3	-1	NA	0.7845201	-2
41	3	3	-1	2	2.8470401	-4
42	2	3	-2	1	2.2042233	-2
43	1	3	5	NA	NA	NA
44	2	3	-3	NA	NA	NA
45	3	3	0	3	3.0015515	-1
46	3	3	-4	2	NA	NA
47	3	3	2	4	2.6166735	0
48	2	3	1	3	1.9112920	-2
49	1	3	4	NA	0.5915667	5
50	2	3	0	2	NA	NA

Fuente: Elaboración propia con el software de programación R. Los valores coloreados en rojo representan los valores perdidos.

2. A continuación aplicamos la función **BBPMM()**:

```
> imputed.data <- BBPMM(data1, nIter=5, M=5)
```

La función **BBPMM()** nos devuelve los siguientes valores:

- *call*: la llamada de la función **BBPMM()** :

```
> imputed.data$call
BBPMM(Data = data1, M = 5, nIter = 5)
```

- *mis.num*: un vector que contiene los números de valores faltantes por columna:

```
> imputed.data$mis.num
x1 x2 x3 y1 y2 y3
 0  0  0 20 15 15
```

- *modelselection*: método de selección del modelo elegido para la llamada de la función:

```
> imputed.data$modelselection
[1] "stepAIC"
```

- *seed*: el valor de la semilla para la llamada de la función:

```
> imputed.data$seed
NULL
```

- *impdata*: el conjunto de datos imputado:

```
> imputed.data$impdata
```

Cuadro 4.12: Representación del conjunto de datos data1 con valores imputados.

	x_1	x_2	x_3	y_1	y_2	y_3
1	1	1	2	2	0.6639219	0
2	3	1	-5	1	2.5831713	-1
3	1	1	1	2	0.3330788	3
4	3	1	2	4	2.6166735	0
5	2	1	4	4	1.6621906	4
6	1	1	-1	2	0.7446018	2
7	3	1	2	4	3.3974445	4
8	2	1	-2	2	2.3046369	1
9	1	1	-1	2	0.7446018	2
10	1	1	-5	1	0.9963446	-4
11	2	2	1	2	1.7431429	1
12	3	2	-1	3	3.5596462	-2
13	1	2	3	2	0.8234375	0
14	3	2	-2	1	2.3046369	0
15	3	2	-4	3	3.5596462	0
16	1	2	-2	2	1.1286590	-3
17	2	2	4	4	2.2457760	2
18	2	2	1	3	1.9112920	-2
19	1	2	0	1	1.4485750	1
20	2	2	-1	1	1.4086837	0
21	1	2	4	2	0.8234375	7
22	3	2	1	4	3.6116357	2
23	1	2	0	2	1.8514327	0
24	1	2	-4	1	1.0166672	-3
25	2	2	-3	2	2.3555955	-5
26	3	2	1	4	3.5596462	4
27	2	2	-3	4	2.3986952	-2
28	3	2	1	1	3.0505409	-2
29	1	2	-3	1	1.8514327	0
30	1	2	0	1	1.4485750	1
31	2	2	3	3	1.0894889	3
32	1	2	-5	1	0.9963446	-4
33	2	2	1	3	1.9112920	-2
34	3	2	8	5	3.0655355	10
35	2	2	-3	1	1.4605429	-4
36	3	3	2	2	3.0929557	2
37	1	3	-1	1	1.4620976	1
38	3	3	4	5	3.9982715	4
39	2	3	4	4	1.8510383	0
40	1	3	-1	2	0.7845201	-2
41	3	3	-1	2	2.8470401	-4
42	2	3	-2	1	2.2042233	-2
43	1	3	5	1	0.5915667	2
44	2	3	-3	2	2.3555955	-5
45	3	3	0	3	3.0015515	-1
46	3	3	-4	2	3.9982715	0
47	3	3	2	4	2.6166735	0
48	2	3	1	3	1.9112920	-2
49	1	3	4	3	0.5915667	5
50	2	3	0	2	1.0894889	-3

Fuente: Elaboración propia con el argumento *impdata* de la función **BBPMM()** del paquete BaBooN de R. Los valores coloreados en azul representan los valores imputados.

- *misOverview*: el porcentaje de valores faltantes por variable incompleta:

```
> imputed.data$misOverview
[1] "number of missing values x1: 0"  "number of missing values x2: 0"
[3] "number of missing values x3: 0"  "number of missing values y1: 20"
[5] "number of missing values y2: 15" "number of missing values y3: 15"
```

- *indMatrix*: una matriz con las mismas dimensiones que data1:

```
> imputed.data$indMatrix
```

Cuadro 4.13: Representación de la matriz *indMatrix* con las mismas dimensiones que data1.

	x_1	x_2	x_3	y_1	y_2	y_3
[1,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[2,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[3,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[4,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[5,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[6,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[7,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[8,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[9,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[10,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[11,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[12,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[13,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[14,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[15,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
[16,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[17,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[18,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
[19,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[20,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[21,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[22,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[23,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[24,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[25,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[26,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[27,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[28,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[29,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
[30,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
[31,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[32,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[33,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[34,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[35,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[36,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[37,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[38,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[39,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[40,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[41,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[42,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[43,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[44,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[45,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[46,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
[47,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[48,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[49,]	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
[50,]	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE

Fuente: Elaboración propia con el argumento *indMatrix* de la función **BBPMM()** del paquete BaBooN de R.

- *M*: número de imputaciones múltiples:

```
> imputed.data$M
[1] 5
```

- *nIter*: número de iteraciones entre dos imputaciones:

```
> imputed.data$nIter
[1] 5
```

- *Chains*: lista que contiene las secuencias de muestreo de Gibbs para cada variables de imputación y para cada iteración:

```
> imputed.data$Chains
```

Cuadro 4.14: Representación de la secuencia de muestreo de Gibbs para la variable y_2 con imputación 1.

	StartSol	Iter2	Iter3	Iter4	Iter5
[1,]	3.3974445	3.3974445	3.3974445	3.3974445	2.6166735
[2,]	2.3046369	2.2042233	2.2457760	2.3555955	2.3046369
[3,]	1.1286590	0.7446018	0.7446018	0.784520	1 0.7446018
[4,]	1.1286590	0.7446018	0.8234375	0.3330788	0.8234375
[5,]	3.3974445	3.5596462	3.5596462	3.5596462	3.5596462
[6,]	1.6621906	2.3986952	2.3555955	1.9112920	1.9112920
[7,]	3.6116357	3.6116357	3.6116357	3.6116357	3.5596462
[8,]	1.4485750	1.4620976	1.8514327	0.7446018	1.8514327
[9,]	0.7845201	1.4485750	1.4485750	1.4485750	1.4485750
[10,]	1.4605429	1.4086837	1.4086837	0.9963446	0.9963446
[11,]	1.6621906	2.3986952	2.3555955	1.9112920	1.9112920
[12,]	1.4485750	0.5915667	0.3330788	0.5915667	0.5915667
[13,]	1.8510383	2.3555955	2.3986952	2.3986952	2.3555955
[14,]	3.0929557	3.5596462	3.5596462	3.0505409	3.9982715
[15,]	1.7431429	2.2042233	1.9112920	2.2042233	1.0894889

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.15: Representación de la secuencia de muestreo de Gibbs para la variable y_2 con imputación 2.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	3.0929557	2.3046369	2.8470401	2.5831713	3.3974445
[2,]	2.3555955	2.3555955	1.9112920	1.6621906	1.7431429
[3,]	0.7446018	0.7446018	0.7446018	0.7446018	0.7446018
[4,]	0.6639219	0.3330788	0.8234375	0.7845201	0.3330788
[5,]	3.9982715	3.9982715	3.9982715	3.9982715	3.5596462
[6,]	1.0894889	1.6621906	2.2042233	2.3986952	2.3555955
[7,]	3.9982715	3.9982715	3.9982715	2.6166735	3.6116357
[8,]	0.5915667	1.8514327	1.0166672	0.9963446	1.8514327
[9,]	1.4485750	1.4485750	1.4485750	1.4485750	1.4485750
[10,]	0.7845201	1.9112920	1.6621906	2.2457760	1.1286590
[11,]	1.0894889	1.6621906	2.2042233	2.3986952	2.3555955
[12,]	0.6639219	0.3330788	0.3330788	0.6639219	0.9963446
[13,]	3.0505409	2.3555955	2.2042233	2.3555955	1.8510383
[14,]	3.9982715	3.6116357	3.9982715	3.9982715	3.0015515
[15,]	2.2457760	1.7431429	2.2042233	2.2457760	2.2457760

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.16: Representación de la secuencia de muestreo de Gibbs para la variable y_2 con imputación 3.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	3.3974445	3.0929557	3.3974445	3.3974445	3.3974445
[2,]	1.7431429	2.2042233	1.0894889	1.4605429	1.8510383
[3,]	0.7446018	0.7845201	0.7446018	0.7446018	0.7446018
[4,]	0.5915667	0.8234375	1.4620976	0.9963446	1.4605429
[5,]	3.0015515	2.5831713	3.0505409	3.6116357	2.6166735
[6,]	2.3986952	1.9112920	1.9112920	1.9112920	1.9112920
[7,]	3.0505409	3.6116357	3.0505409	2.5831713	3.6116357
[8,]	0.6639219	1.0166672	0.3330788	0.9963446	1.8514327
[9,]	1.4485750	1.4485750	1.4485750	1.4485750	1.4485750
[10,]	1.4620976	0.9963446	0.9963446	1.4620976	1.4605429
[11,]	2.3986952	1.9112920	1.9112920	1.9112920	1.9112920
[12,]	0.5915667	0.8234375	1.4485750	1.4620976	1.4605429
[13,]	2.3555955	2.3986952	2.3555955	1.8510383	1.9112920
[14,]	3.0015515	2.5831713	3.0015515	2.6166735	3.0655355
[15,]	2.3555955	1.4086837	1.7431429	1.0894889	1.0894889

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.17: Representación de la secuencia de muestreo de Gibbs para la variable y_2 con imputación 4.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	3.9982715	2.5831713	3.3974445	2.8470401	2.5831713
[2,]	2.2042233	1.6621906	2.3555955	1.8510383	1.6621906
[3,]	0.7446018	0.6639219	0.6639219	0.5915667	0.7446018
[4,]	0.8234375	1.4485750	0.3330788	1.8514327	1.1286590
[5,]	2.8470401	3.6116357	2.8470401	3.6116357	3.5596462
[6,]	2.3986952	1.0894889	2.3986952	1.6621906	2.2457760
[7,]	3.6116357	3.6116357	3.3974445	3.0929557	2.3046369
[8,]	0.7845201	1.0166672	1.1286590	0.5915667	1.0166672
[9,]	1.4485750	1.1286590	0.8234375	1.0166672	1.8514327
[10,]	1.1286590	1.1286590	0.3330788	0.9963446	1.0166672
[11,]	1.9112920	2.3555955	1.8510383	2.3555955	2.3555955
[12,]	0.5915667	0.7845201	0.9963446	0.7446018	0.7845201
[13,]	1.9112920	1.9112920	1.6621906	2.3555955	1.9112920
[14,]	2.8470401	3.0015515	2.6166735	3.0929557	3.0015515
[15,]	2.3555955	2.2042233	1.4605429	1.4605429	2.2042233

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.18: Representación de la secuencia de muestreo de Gibbs para la variable y_2 con imputación 5.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	3.3974445	3.0929557	3.3974445	3.3974445	3.3974445
[2,]	1.8514327	1.9112920	2.2457760	2.2042233	1.4605429
[3,]	0.7446018	0.7446018	0.7446018	0.7446018	0.7446018
[4,]	1.0166672	1.8514327	0.8234375	0.3330788	0.3330788
[5,]	3.0655355	3.5596462	3.9982715	3.5596462	2.5831713
[6,]	2.3046369	1.9112920	1.9112920	1.9112920	1.9112920
[7,]	3.6116357	3.0505409	3.6116357	3.6116357	3.0505409
[8,]	0.8234375	1.8514327	1.1286590	0.7446018	1.1286590
[9,]	1.4485750	1.4485750	1.4485750	1.4485750	1.4485750
[10,]	1.0166672	0.9963446	1.4620976	0.9963446	0.9963446
[11,]	2.3046369	1.9112920	1.9112920	1.9112920	1.9112920
[12,]	0.8234375	0.6639219	0.3330788	1.4485750	0.3330788
[13,]	2.2457760	1.4605429	2.3986952	2.3986952	2.3986952
[14,]	3.0505409	3.0505409	3.9982715	3.0505409	2.5831713
[15,]	2.3046369	1.4605429	2.3986952	2.2042233	1.9112920

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.19: Representación de la secuencia de muestreo de Gibbs para la variable y_3 con imputación 1.

	StartSol	Iter2	Iter3	Iter4	Iter5
[1,]	4	4	4	4	0
[2,]	-2	-2	-4	-4	1
[3,]	3	2	2	2	2
[4,]	3	4	5	0	0
[5,]	-4	-1	-3	-1	0
[6,]	0	2	2	1	-2
[7,]	2	2	2	-2	4
[8,]	-4	-4	-2	-2	0
[9,]	0	1	1	1	1
[10,]	-3	-2	-4	-4	-4
[11,]	0	2	2	1	-2
[12,]	7	10	7	7	2
[13,]	-3	-5	-1	-3	-5
[14,]	-4	-1	-3	-1	0
[15,]	-2	1	-4	-2	-3

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.20: Representación de la secuencia de muestreo de Gibbs para la variable y_3 con imputación 2.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	0	0	2	3	4
[2,]	3	2	2	2	1
[3,]	2	2	2	2	2
[4,]	2	1	5	7	5
[5,]	-2	-3	-1	-5	-1
[6,]	-1	2	2	0	1
[7,]	7	3	1	0	-2
[8,]	-4	1	-2	-4	-2
[9,]	1	1	1	1	1
[10,]	-1	-1	-1	-2	-1
[11,]	-1	2	2	0	1
[12,]	2	2	4	5	7
[13,]	-2	-3	-4	-2	-4
[14,]	-4	-4	-1	-2	-1
[15,]	-3	0	-2	0	1

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.21: Representación de la secuencia de muestreo de Gibbs para la variable y_3 con imputación 3.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	4	4	4	4	4
[2,]	0	-2	0	0	-1
[3,]	2	-2	2	2	2
[4,]	5	4	4	4	7
[5,]	-1	-4	-3	-3	-3
[6,]	2	0	2	2	-2
[7,]	2	2	2	2	2
[8,]	-4	-3	-3	-2	-2
[9,]	1	1	1	1	1
[10,]	-1	-4	-3	-3	-3
[11,]	2	0	2	2	-2
[12,]	10	10	5	4	4
[13,]	-3	-3	-3	-3	-1
[14,]	-1	-4	-3	-3	-3
[15,]	0	1	-2	-2	-5

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.22: Representación de la secuencia de muestreo de Gibbs para la variable y_3 con imputación 4.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	0	0	4	0	4
[2,]	0	0	-2	2	4
[3,]	2	2	4	4	1
[4,]	2	4	7	7	3
[5,]	-3	-4	-5	-2	-3
[6,]	0	-2	-2	2	4
[7,]	2	2	-4	4	2
[8,]	-2	-4	-3	-1	-4
[9,]	1	1	0	2	1
[10,]	-3	-4	-5	-2	-3
[11,]	0	-2	2	3	4
[12,]	5	4	3	0	0
[13,]	-3	-5	-3	-2	-3
[14,]	-3	-1	-5	-2	-3
[15,]	-2	-2	2	-1	-2

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.23: Representación de la secuencia de muestreo de Gibbs para la variable y_3 con imputación 5.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	4	4	4	4	4
[2,]	2	0	2	0	-2
[3,]	2	2	2	2	2
[4,]	3	2	3	2	-2
[5,]	1	-1	-1	-1	-5
[6,]	5	1	-2	0	0
[7,]	2	2	2	2	-2
[8,]	-4	-2	-2	-5	-4
[9,]	1	0	1	1	1
[10,]	-3	-4	-3	-1	-2
[11,]	5	1	-2	0	0
[12,]	0	0	3	4	-2
[13,]	-4	-1	-4	-5	-3
[14,]	-3	-1	-3	-1	-4
[15,]	0	1	0	0	-2

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.24: Representación de la secuencia de muestreo de Gibbs para la variable y_1 con imputación 1.

	StartSol	Iter2	Iter3	Iter4	Iter5
[1,]	3	2	2	2	2
[2,]	2	2	1	1	1
[3,]	1	1	1	1	2
[4,]	4	4	4	4	4
[5,]	3	2	3	4	2
[6,]	2	2	2	2	2
[7,]	4	4	4	4	3
[8,]	3	2	2	1	2
[9,]	4	4	4	4	4
[10,]	2	2	2	1	2
[11,]	4	4	4	5	4
[12,]	1	2	1	1	4
[13,]	5	4	4	4	1
[14,]	1	1	1	1	1
[15,]	3	3	3	3	3
[16,]	2	2	4	4	2
[17,]	2	1	2	1	2
[18,]	2	1	1	2	1
[19,]	1	2	1	2	2
[20,]	1	2	1	2	3

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.25: Representación de la secuencia de muestreo de Gibbs para la variable y_1 con imputación 2.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	3	1	3	2	3
[2,]	1	1	1	1	2
[3,]	1	1	1	1	2
[4,]	4	4	4	4	4
[5,]	1	1	1	2	2
[6,]	2	2	2	2	2
[7,]	3	3	4	4	4
[8,]	2	2	2	1	1
[9,]	4	4	4	4	4
[10,]	3	1	3	2	3
[11,]	4	4	4	4	4
[12,]	2	1	2	1	2
[13,]	4	4	4	4	4
[14,]	1	1	1	1	1
[15,]	3	3	3	3	3
[16,]	4	4	4	4	4
[17,]	2	1	1	1	1
[18,]	3	3	2	2	1
[19,]	1	1	1	2	1
[20,]	3	2	1	1	1

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.26: Representación de la secuencia de muestreo de Gibbs para la variable y_1 con imputación 3.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	4	4	4	5	4
[2,]	1	1	1	2	1
[3,]	1	2	1	2	2
[4,]	4	4	4	4	4
[5,]	3	2	1	3	3
[6,]	2	2	2	2	2
[7,]	4	4	4	4	4
[8,]	3	2	2	3	4
[9,]	4	4	4	4	4
[10,]	2	1	1	3	4
[11,]	4	4	2	4	4
[12,]	1	1	2	1	1
[13,]	4	4	5	4	5
[14,]	1	1	1	1	1
[15,]	3	3	3	3	3
[16,]	4	3	2	2	1
[17,]	1	1	1	1	1
[18,]	1	2	3	3	2
[19,]	1	2	2	1	2
[20,]	1	1	1	1	1

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.27: Representación de la secuencia de muestreo de Gibbs para la variable y_1 con imputación 4.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	4	4	4	1	1
[2,]	1	1	1	1	1
[3,]	3	3	4	1	1
[4,]	4	4	4	4	4
[5,]	1	1	1	1	1
[6,]	2	2	1	1	2
[7,]	1	2	2	2	2
[8,]	4	4	1	2	2
[9,]	4	4	4	4	4
[10,]	3	4	1	2	2
[11,]	4	4	1	4	4
[12,]	1	1	1	2	1
[13,]	3	4	1	4	4
[14,]	2	2	5	1	1
[15,]	3	3	1	3	3
[16,]	1	3	1	4	4
[17,]	1	1	2	1	2
[18,]	3	3	1	1	1
[19,]	1	1	1	1	2
[20,]	2	2	2	3	3

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

Cuadro 4.28: Representación de la secuencia de muestreo de Gibbs para la variable y_1 con imputación 5.

	Iter1	Iter2	Iter3	Iter4	Iter5
[1,]	1	1	2	3	3
[2,]	2	1	2	2	1
[3,]	1	1	1	1	1
[4,]	4	4	4	4	4
[5,]	2	2	1	2	1
[6,]	2	2	2	2	2
[7,]	4	3	4	3	4
[8,]	3	3	2	2	3
[9,]	4	4	4	4	4
[10,]	3	2	2	2	1
[11,]	4	4	4	4	4
[12,]	2	2	1	2	1
[13,]	4	4	4	4	4
[14,]	1	1	1	1	1
[15,]	3	3	3	3	3
[16,]	4	4	4	4	4
[17,]	1	2	2	2	1
[18,	1	3	3	4	4
[19,]	1	1	1	2	1
[20,]	2	3	2	2	1

Fuente: Elaboración propia con el argumento *Chains* de la función **BBPMM()** del paquete BaBooN de R.

- *ignoredvariables*: TRUE / FALSE si las variables se ignoran durante la imputación:

```
> imputed.data$ignoredvariables
[1] FALSE
```

BBPMM.row ((Multiple) Imputation of variable vectors)

Realiza imputación única o múltiple de vectores de variables de escala métrica. Las imputaciones se generan utilizando la predicción media de coincidencia (PMM) como se describe en Little(1988). Esta función es útil para

patrones de ausencia por diseño, tales como fusión de datos o cuestionarios divididos. Las medias predictivas de las variables de imputación se ponderan por la inversa de la matriz de covarianza de los residuos de la regresión de estas variables completas. La idea que está detrás de esta función es que las distancias entre las medias predictivas deben ser castigadas más severamente, si la variable particual puede ser bien explicada a partir de las variables del modelo de imputación. A través de la parcialización, los residuos de la matriz de pesos se transforman en una matriz diagonal. Los pesos calculados se pueden ajustar de forma manual. La matriz de distancias es una matriz de Mahalanobis, los pesos están en el denominador y, por tanto cuanto menor sea el peso, mayor será la influencia. Como esto es algo contraintuitivo, se toma el recíproco de los pesos manuales. Por lo tanto, el mayor peso manual es la influencia del predictor de la variable correspondiente en la distancia total. Para identificar los patrones de falta por diseño es necesario transformar el conjunto de datos a través de la función **rowinpPrep()** y a diferencia de la función **BBPMM()** este algoritmo no se basa en la regresión secuencial, lo que lleva a que las variables imputadas sean independientemente condicionales de las variables completamente observadas. La función **BBPMM.row()** tiene el siguiente uso:

```
BBPMM.row(misDataPat, blockImp=length(misDataPat$blocks),M=10, outfile=NULL, manWeights=NULL, stepmod="
     stepAIC", verbose=TRUE,
tol=0.25, setSeed=NULL, ...)
```

Argumentos:

- *misDataPat*: objeto creado por la función **rowimpPrep()** que contiene información sobre todos los patrones identificados de datos perdidos.
- *blackImp*: un escalar o vector que contiene el número o números del bloque o bloques considerados para la imputación. Por defecto sólo se imputa el último bloque.
- *M*: número de imputaciones múltiples. Si *M = 1* no se lleva a cabo ningún paso Boostrap Bayesiano.
- *outfile*: cadena de caracteres que especifica la ruta y el nombre de archivo para los conjuntos de datos imputados. Si *outfile = NULL* (valor predeteminado) no se almacena ningún conjunto de datos.
- *manWeights*: argumento opcional que permite o contiene pesos manuales (no negativos) para el paso PMM. El argumento *manWeights* puede ser una lista que contiene un vector para cada patrón de datos perdidos, o sólo un vector, si sólo existe un patrón / bloque de ausencia. En cualquier caso, el número de elementos en el vector o los vectores debe coincidir con el número de bloques correspondientes. Hay que tener en cuenta que cuanto mayor sea el peso correspondiente mayor será la importancia de una buena coincidencia para la media predictiva correspondiente.
- *stepmod*: realiza la selección de variables para cada modelo de imputación basandose en el Criterio de Información hacia atrás Schwarz (Bayes). Por defecto *"stepAIC "*.
- *verbose*: el algoritmo imprime información sobre las matrices de ponderación y los números de imputación. El valor predeterminado es igual a TRUE.
- *tol*: argumento importado de la función **qr()** que especifica el nivel de tolerancia para dependencias entre las variables completas. El valor por defecto es 0.25.
- *setSeed*: argumento opcional para configurar el generador de números pseudo-aleatorios para permitir obtener resultados reproducibles.
- ...: otros argumentos pasados desde otras funciones.

Ejemplos:

Ejemplo 1: Aplicación de la función **BBPMM.row()**:

1. Generamos un conjunto de datos de muestra con variables no normales y un patrón de datos perdidos a través de los siguiente comandos:

```
> set.seed(1000)
> n <- 50
> x1 <- round(runif(n,0.5,3.5))
> x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
> x3 <- round(rnorm(n,0,3))
> y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
> y1 <- ifelse(y1<1,1,y1)
> y1 <- ifelse(y1>4,5,y1)
> y2 <- y1+rnorm(n,0,0.5)
> y3 <- round(x3+rnorm(n,0,2))
> data <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
> misrow1 <- sample(n,20)
> data[misrow1, c(4:6)] <- NA
```

de la aplicación anterior se obtiene el siguiente conjunto de datos data con valores NA:

Cuadro 4.29: Representación del conjunto de datos data con valores NA.

	x_1	x_2	x_3	y_1	y_2	y_3
1	1	1	2	NA	NA	NA
2	3	1	-5	NA	NA	NA
3	1	1	1	NA	NA	NA
4	3	1	2	NA	NA	NA
5	2	1	4	4	3.6621906	4
6	1	1	-1	2	1.7446018	2
7	3	1	2	4	4.3974445	4
8	2	1	-2	NA	NA	NA
9	1	1	-1	NA	NA	NA
10	1	1	-5	1	0.9963446	-4
11	2	2	1	2	1.7431429	1
12	3	2	-1	NA	NA	NA
13	1	2	3	NA	NA	NA
14	3	2	-2	1	0.3046369	0
15	3	2	-4	3	3.0484422	-5
16	1	2	-2	2	2.1286590	-3
17	2	2	4	NA	NA	NA
18	2	2	1	3	2.5933312	0
19	1	2	0	1	1.4485750	1
20	2	2	-1	1	0.4086837	0
21	1	2	4	NA	NA	NA
22	3	2	1	4	4.6116357	2
23	1	2	0	2	2.8514327	0
24	1	2	-4	1	1.0166672	-3
25	2	2	-3	2	2.3555955	-5
26	3	2	1	NA	NA	NA
27	2	2	-3	NA	NA	NA
28	3	2	1	NA	NA	NA
29	1	2	-3	1	1.6682071	-3
30	1	2	0	1	0.9855936	-1
31	2	2	3	3	2.0894889	3
32	1	2	-5	NA	NA	NA
33	2	2	1	NA	NA	NA
34	3	2	8	5	5.0655355	10
35	2	2	-3	1	0.4605429	-4
36	3	3	2	NA	NA	NA
37	1	3	-1	1	1.4620976	1
38	3	3	4	5	5.9982715	4
39	2	3	4	4	3.8510383	0
40	1	3	-1	NA	NA	NA
41	3	3	-1	2	1.8470401	-4
42	2	3	-2	1	1.2042233	-2
43	1	3	5	NA	NA	NA
44	2	3	-3	NA	NA	NA
45	3	3	0	3	3.0015515	-1
46	3	3	-4	2	2.1808749	-7
47	3	3	2	4	3.6166735	0
48	2	3	1	3	2.9112920	-2
49	1	3	4	NA	NA	NA
50	2	3	0	2	1.3497499	3

Fuente: Elaboración propia con el software de programación R. Los valores coloreados en rojo representan los valores perdidos

2. Aplicamos la función **rowimpPrep()** a nuestros datos como paso de preparación para la imputación:

```
> impblock <- rowimpPrep(data)
variables with missing values ordered by identical patterns:
$`block 1`
[1] "y1" "y2" "y3"
> impblock
$data
```

Cuadro 4.30: Representación del conjunto de datos data con valores NA.

	x_1	x_2	x_3	y_1	y_2	y_3
1	1	1	2	NA	NA	NA
2	3	1	-5	NA	NA	NA
3	1	1	1	NA	NA	NA
4	3	1	2	NA	NA	NA
5	2	1	4	4	3.6621906	4
6	1	1	-1	2	1.7446018	2
7	3	1	2	4	4.3974445	4
8	2	1	-2	NA	NA	NA
9	1	1	-1	NA	NA	NA
10	1	1	-5	1	0.9963446	-4
11	2	2	1	2	1.7431429	1
12	3	2	-1	NA	NA	NA
13	1	2	3	NA	NA	NA
14	3	2	-2	1	0.3046369	0
15	3	2	-4	3	3.0484422	-5
16	1	2	-2	2	2.1286590	-3
17	2	2	4	NA	NA	NA
18	2	2	1	3	2.5933312	0
19	1	2	0	1	1.4485750	1
20	2	2	-1	1	0.4086837	0
21	1	2	4	NA	NA	NA
22	3	2	1	4	4.6116357	2
23	1	2	0	2	2.8514327	0
24	1	2	-4	1	1.0166672	-3
25	2	2	-3	2	2.3555955	-5
26	3	2	1	NA	NA	NA
27	2	2	-3	NA	NA	NA
28	3	2	1	NA	NA	NA
29	1	2	-3	1	1.6682071	-3
30	1	2	0	1	0.9855936	-1
31	2	2	3	3	2.0894889	3
32	1	2	-5	NA	NA	NA
33	2	2	1	NA	NA	NA
34	3	2	8	5	5.0655355	10
35	2	2	-3	1	0.4605429	-4
36	3	3	2	NA	NA	NA
37	1	3	-1	1	1.4620976	1
38	3	3	4	5	5.9982715	4
39	2	3	4	4	3.8510383	0
40	1	3	-1	NA	NA	NA
41	3	3	-1	2	1.8470401	-4
42	2	3	-2	1	1.2042233	-2
43	1	3	5	NA	NA	NA
44	2	3	-3	NA	NA	NA
45	3	3	0	3	3.0015515	-1
46	3	3	-4	2	2.1808749	-7
47	3	3	2	4	3.6166735	0
48	2	3	1	3	2.9112920	-2
49	1	3	4	NA	NA	NA
50	2	3	0	2	1.3497499	3

Fuente: Elaboración propia con el software de programación R. Los valores coloreados en rojo representan los valores perdidos

```
$key
NULL

$blocks
$blocks$`block 1`
[1] 4 5 6

$blockNames
$blockNames$`block 1`
[1] "y1" "y2" "y3"

$compNames
[1] "x1" "x2" "x3"

$ignore
NULL

$ignored_data
NULL

$indMatrix
```

Cuadro 4.31: Representación de la matriz *indMatrix* del conjunto data.

	x_1	x_2	x_3	y_1	y_2	y_3
[1,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[2,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[3,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[4,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[5,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[6,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[7,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[8,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[9,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[10,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[11,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[12,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[13,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[14,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[15,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[16,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[17,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[18,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[19,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[20,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[21,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[22,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[23,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[24,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[25,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[26,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[27,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[28,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[29,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[30,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[31,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[32,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[33,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[34,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[35,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[36,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[37,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[38,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[39,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[40,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[41,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[42,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[43,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[44,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[45,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[46,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[47,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[48,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
[49,]	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE
[50,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Fuente: Elaboración propia con el argumento *indMatrix* de la función **BBPMM.row()** del paquete BaBooN de R.

```
attr(,"class")
[1] "impprep"
```

3. Aplicamos la función **BBPMM.row()**:

```
> imputed.data <- BBPMM.row(impblock, M=5)
```

Cuadro 4.32: Representación de la imputación 1: matriz de pesos recíprocos para el bloque 1.

	[, 1]	[, 2]	[, 3]
[1,]	3.352321	0.000000	0.0000000
[2,]	0.000000	3.311149	0.0000000
[3,]	0.000000	0.000000	0.2648336

Fuente: Elaboración propia con la función **BBPMM.row()** del paquete BaBooN de R.

Cuadro 4.33: Representación de la imputación 2: matriz de pesos recíprocos para el bloque 2.

	[, 1]	[, 2]	[, 3]
[1,]	4.535288	0.000000	0.0000000
[2,]	0.000000	8.027837	0.0000000
[3,]	0.000000	0.000000	0.1308363

Fuente: Elaboración propia con la función **BBPMM.row()** del paquete BaBooN de R.

Cuadro 4.34: Representación de la imputación 3: matriz de pesos recíprocos para el bloque 3.

	[, 1]	[, 2]	[, 3]
[1,]	2.810017	0.000000	0.0000000
[2,]	0.000000	6.297423	0.0000000
[3,]	0.000000	0.000000	0.4884128

Fuente: Elaboración propia con la función **BBPMM.row()** del paquete BaBooN de R.

Cuadro 4.35: Representación de la imputación 4: matriz de pesos recíprocos para el bloque 4.

	[, 1]	[, 2]	[, 3]
[1,]	3.816726	0.000000	0.0000000
[2,]	0.000000	5.414398	0.0000000
[3,]	0.000000	0.000000	0.1397776

Fuente: Elaboración propia con la función **BBPMM.row()** del paquete BaBooN de R.

Cuadro 4.36: Representación de la imputación 5: matriz de pesos recíprocos para el bloque 5.

	[, 1]	[, 2]	[, 3]
[1,]	2.117696	0.000000	0.0000000
[2,]	0.000000	4.242962	0.0000000
[3,]	0.000000	0.000000	0.5543487

Fuente: Elaboración propia con la función **BBPMM.row()** del paquete BaBooN de R.

MI.inference (Multiple Imputation Inference)

Aplica las reglas de combinación de Rubin a las cantidades estimadas de interés que se basan en aplicar la imputación múltiple a los conjuntos de datos imputados. La función requiere como entrada dos vectores de longitud M para la estimación y su varianza:

```
MI.inference(thetahat, varhat.thetahat, alpha=0.05)
```

Argumentos:

- *thetahat*: un vector de longitud M que contiene estimaciones de la cantidad de interés basada en múltiples conjuntos de datos imputados.
- *varhat.thetahat*: un vector de longitud M que contiene las correspondientes varianzas del argumentos *thetahat*.
- *alpha*: el nivel de confianza para el cálculo de los límites inferior y superior. Por defecto el valor es 0.05.

Ejemplos:

- Ejemplo 1: Aplicación de la función **MI.inference()**:

  1. Generamos un conjunto de datos de muestra aplicando los siguientes comandos:

```
> n <- 100
> x1 <- round(runif(n,0.5,3.5))
> x2 <- round(runif(n,0.5,4.5))
> x3 <- runif(n,1,6)
> y1 <- round(x1-0.25*x2+0.5*x3+rnorm(n,0,1))
> y1 <- ifelse(y1<2,2,y1)
> y1 <- as.factor(ifelse(y1>4,5,y1))
> y2 <- x3+rnorm(n,0,2)
> y3 <- as.factor(ifelse(x2+rnorm(n,0,2)>2,1,0))
> mis1 <- sample(100,20)
> mis2 <- sample(100,30)
> mis3 <- sample(100,25)
> data1 <- data.frame("x1"=x1,"x2"=x2,"x3"=x3,
                      "y1"=y1,"y2"=y2,"y3"=y3)
> is.na(data1$y1[mis1]) <- TRUE
> is.na(data1$y2[mis2]) <- TRUE
> is.na(data1$y3[mis3]) <- TRUE
> imputed.data <- BBPMM(data1, M=5, nIter=5)
> MI.m.meany2.hat <- sapply(imputed.data$impdata,
                            FUN=function(x) mean(x$y2))
>
> MI.v.meany2.hat <- sapply(imputed.data$impdata,
                           FUN=function(x) var(x$y2)/length(x$y2))
```

2. A continuación aplicamos la función **MI.inference**():

```
> MI.y2 <- MI.inference(MI.m.meany2.hat, MI.v.meany2.hat, alpha=0.05)
```

la función **MI.inference()** nos devuelve los siguientes valores:

- *MI.Est*: un escalar que contiene la estimación IM de la cantidad de interés (es decir, un estimador promedio en todos los conjuntos de datos M):

```
> MI.y2$MI.Est
[1] 3.665684
```

- *MI.var*: la varianza de Imputación Múltiple:

```
> MI.y2$MI.Var
[1] 0.05598576
```

- *CI.low*: el límite inferior del intervalo de confianza IM:

```
> MI.y2$CI.low
[1] 3.201886
```

- *CI.up*: el límite superior del intervalo de confianza IM:

```
> MI.y2$CI.up
[1] 4.129483
```

- *BVar*: estimación de la varianza entre clases:

```
> MI.y2$BVar
[1] 0.0008384155
```

- *Wvar*: estimación dentro de la varianza:

```
> MI.y2$WVar
[1] 0.05497966
```

Capítulo 5

Conclusiones

En este trabajo se han presentado y analizado los fundamentos teóricos de distintos procedimientos de imputación dando cuenta de sus bondades y limitaciones. Asimismo, se han aplicado diferentes posibilidades de utilización de los métodos con el lenguaje de programación estadístico R, ofreciendo escenarios diversos.

Todos los métodos de imputación presentados en este trabajo, tienen limitaciones y su correcta aplicación dependerá de la manera en que se comporten los datos faltantes, esto nos lleva afirmar que el mejor método de imputación es el que no se aplica, lo que sugiere agotar todos los recursos para minimizar la falta de respuesta total y parcial en una encuesta.

En la medida que la falta de respuesta no muestre un patrón aleatorio, la eficacia de todas las metodologías se debilita, en procedimientos estadísticamente robustos como imputación múltiple y máxima verosimilitud.

No existe el mejor método de imputación. Cada situación es diferente y la elección del procedimiento de sustitución de datos depende de la variable de estudio, del porcentaje de datos faltantes, del tipo de encuesta, del análisis estadístico que se quiere realizar, y del uso que se hará de la información imputada. Por ello no es aconsejable elegir un procedimiento de imputación y aplicarlo de forma generalizada para todas las variables en todas las encuestas.

La literatura suele recomendar la aplicación de dos procedimientos: el método de imputación por máxima verosimilitud (MV) y el de imputación múltiple (IM). No obstante, no es posible afirmar que el algoritmo de imputación múltiple genere siempre mejores resultados que los métodos simples.

En distintos campos de las ciencias sociales se demuestra que la utilización de los procedimientos de imputación Hot Deck llevan aparejados una mayor eficiencia que los métodos de imputación múltiple y de regresión paramétrica, lo que garantiza preservar al máximo la distribución de la variable imputada. Si el proceso de generación de información requiere imputar en distintas bases de datos, este algoritmo puede considerarse como una opción viable que mantiene un equilibrio adecuado entre la teoría y la práctica.

Hay que destacar que cuando imputamos se logra obtener una base de datos completa, la cual permite llevar a cabo metodologías de análisis de datos comunes. Si una imputación se lleva a cabo de manera adecuada, podría disminuirse el sesgo, en caso de existir.

El investigador debe ser consciente que el uso de un método de imputación puede llegar a afectar las distribuciones conjuntas, o incluso a las distribuciones marginales de las variables, aunque el problema es menor si la distribución de los casos ausentes es la misma que la de los casos completos (patrón de pérdida ignorable). Si la técnica no es la adecuada, es posible que aumente el sesgo, se sobreestime la varianza y se obtengan datos imputados inconsistentes produciendo una base de datos poco fiable, lo que llevará a la interpretación errónea de los resultados por parte del usuario.

La eficiencia del uso de una técnica de imputación, muchas veces se ve afectada por la heterogeneidad de los datos, que hace que el valor donante pertenezca a un registro de una característica muy distinta a la del registro a imputar, obteniendo datos inconsistentes y/o sesgos grandes. Si se lleva a cabo una imputación usando un valor aleatorio se obtendrán mejores resultados si se clasifican los datos formando grupos homogéneos y luego se selecciona el donante dentro del grupo, el cual tendría características más similares en relación al receptor. Formar clases de imputación podría generar buenas estimaciones del valor faltante, ya que se imputaría valores similares a los verdaderos y podría haber mayor consistencia con los otros valores del registro. En Little y Rubin (1987) se expone que existen dos requerimientos importantes para el logro de este objetivo: uno es que dentro de cada clase de imputación el valor del donante represente los valores de los faltantes (lo que implícitamente supone pérdida aleatoria), y la otra es que dentro de cada clase el valor donante tenga una varianza pequeña.

Existen muchas maneras de llevar a cabo la evaluación y validación de las técnicas de imputación que hayan sido aplicadas a una situación particular, siendo la más común de ellas es el uso de la simulación (lo que podría generar una línea futura de este trabajo).

En definitva obtener una base de datos real completa es prácticamente imposible en la realidad, por las múltiples causas que pueden ocurrir y llevar a la ausencia de datos. Para resolver el problema y poder hacer uso de las técnicas estadísticas tradicionales es necesario conocer la forma de evitar o minimizar la no respuesta, su tratamiento, y la mejor manera de imputar los datos faltantes, lo cual es un arte del investigador basado en el conocimiento de las técnicas existentes para obtener una base de datos consistente y adecuada. El investigador puede basarse en el tipo de información que posea, en el enfoque que más convenga y en el uso de técnicas de simulación para evaluar las diferentes opciones de imputación que puede utilizar. Siempre se debe evitar la no respuesta en la medida de lo posible, para usar imputación sólo cuando sea necesario, ya que nunca un conjunto de datos imputados será mejor que un conjunto de datos real.

Bibliografía

[1] Acock CA, Demo D (1994) Family diversity and well-being. Thousand Oaks. C. A. Sage.

[2] Andridge RR, Little R (2010) A Review of Hot Deck Imputation for survey Non-response. International Statistical Review 78:40-64.

[3] Annoni P, Barbiero A, Ferrari PA, Manzi G (2011) An imputation method for categorical variables with application to nonlinear principal component analysis. Computational Statistics & Data Analysis, vol. 55, issue 7, pp 2410-2420.

[4] Barbiero A, Pier AF, Giancarlo M (2014) ForImp: Imputation of Missing Values Through a Forward Imputation Algorithm. R package versión 1.0.3. http://www.cran.r-project.org//package=ForImp. Accedido 1 de febrero de 2017.

[5] Beale EML., Little RJA (1975) Missing Values in Multivariate Analysis. Journal of the Royal Statistical Society. Series B (Methodological), Vol. 37, No. 1, pp 129-145.

[6] Bishop G, Welch G (1995) An Introduction to the Kalman Filter. SIGGRAPH 2001. Course 8.

[7] Buehlmann P, Stckhoven DJ (2012) MissForest-nonparametric missing value imputation for mixed-typedata, Bioinformatics, 28(1) 2012, 112-118, doi:10.193/bioinformatics/btr597.

[8] Braña Tobío T, Varela Mallou J, García Carreira A, Rial Boubeta A, Vázquez Fernández XG (1996) Estimación de la respuesta de los «no sabe/no contesta» en los estudios de intención de voto. Reis 83:269-287.

[9] Carpenter J, Quartagno M (2016) jomo: Multilevel Joint Modelling Multiple Imputation. R package versión 2.3-1. http://www.cran.r-project.org/package=jomo. Accedido 1 de febrero de 2017.

[10] Cheng X, Cook D, Hofmann H (2016) MissingDataGUI: A GUI for Missing Data Exploration. R package versión 0.2-5. http://www.cran.r-project.org/package=MissingDataGUI. Accedido 1 de febrero de 2017.

[11] Cochran WG (1977) Sampling Techniques. John Wiley & Sons. New York.

[12] Cohen J, Cohen P (1983) Applied Multiple Regression Correlation Analysis the Behavioral Sciences. Hillsdale, NJ, Lawrence Erlbaum.

[13] Cruz Cantero P (1990) Del no sabe al no contesta: un lugar de encuentro para diversas respuestas. Reis 52:139-156.

[14] Dai S, Wang X, Suetina D (2016) TestDataImputation: Missing Item Responses Imputation for Test and Assessment Data. R package versión 1.0. http://www.cran.r-project.org/package=TestDataImputation. Accedido 1 de febrero de 2017.

[15] Deming W (1953) On a probability mechanism to attain an economic balance between the result and error of non-response and the bias of non-response. JASA 48: 743-772.

[16] Deming W (1960) Sample Desing in Business Research. John & Sons. New York.

[17] Dempster AP, Lair NM, Rubin DB (1977) Maximun likelihood estimation from incomplete data via the EM algorithm (with discussion). Journal of the Royal Statistical Society, Series B, 39.

[18] Dieter WJ (2015) HotDeckImputation: Hot Deck Imputation Methods for Missing Data. R package versión 1.1.0. http://www.cran.r-project.org/package=HotDeckImputation. Accedido 1 de febrero de 2017.

[19] Díez Nicolás J (1996) Predicción de escaños electorales mediante encuestas. Reis 74:269-289.

[20] Durrant Gabrielle B (2005) Imputation Methods for Handling Item Non response in the Social Sciences: A Methodological Review. ESRC, Univesity of Southampton.

[21] Fellegi I, Holt D (1976) A systematic approach to automatic edit imputation. Journal of the American Statistical Association. Vol. 71/353:17-35.

[22] Fellegi I, Oldt D (1980) Un enfoque sistemático de la edición e imputación automáticas. EE 88:33-80.

[23] Figueredo AJ, Mcknight Patrick E, Mckinght Katherine M, Sidani S (2000) Multivariate modelling of missing data withim and across assessment waves. Addiction 95 (Suplement 3), pp 361-380.

[24] Florian M, Thorsten S (2015) BaBoon: Bayesian Bootstrap Predictive Mean Matching-Multiple and Single Imputation for Discrete Data. R package version 0.2-0. http: //www.cran.r-project.org/package=BaBoon. Accedido 1 de febrero de 2017.

[25] Genolini C, Falissard B, Fang D, Tierney L (2016) longitudinalData: Longitudinal Data. R package versión 2.4.1. http://www.cran.r-project.org/package=longitudinalData. Accedido 1 de febrero de 2017.

[26] Goicoechea P (2002) Imputación basada en árboles de clasificación. EUSTAT. Victoria.

[27] Gómez García J, Palarea Albadalejo J, Martín Fernández JA (2006) Métodos de inferencia estadística con datos faltantes. Estudio de simulación sobre los efectos en las estimaciones. Estadística Española 48: 241-270.

[28] Hansen MH, Hurwitz WN (1946) The problem of non response in sample surveys. JASA 41:517-529.

[29] Harvey AC (1990) Forescanting, structural time series models and the Kalman filter. Cambridge university press.

[30] Hemel J y otros (1987) Stepwise deletion: a technique for missing data handling in multivariate analysis. Analytical Chemical Acta 193:255-268.

[31] Honaker J, King G, Blackwell M (2015) Amelia: A Program for Missing Data. R package versión 1.7.4. http://www.cran.r-project.org/package=Amelia. Accedido 1 de febrero de 2017.

[32] Hyndman RJ, Khandakar Y (2008) Automatic time series forescanting: the forescast package for R. Journal of Statistical software, 26(3).

[33] Ibrahim JG (1990) Incomplete data in generalized linear models. Journal of the American Statistical Association.

[34] Kish L, Hess I (1959) A replacement procedure for reducing the bias of non-response. AS 13/4:17-19.

[35] Koller Meindfelder F (2009) Analysis of Incomplete Survey Data-Multiple Imputation via Bayesian Bootstrap Predictive Mean Matching, doctoral thesis.

[36] Little R (1988) Missing-Data Adjustments in Large Surveys. Journal of Business and Economic Statistics,vol.36, nº 3, pp 287-296.

[37] Little R (1995) Modelling the dropout mechanism in repeated-measures studies. Journal of the American Statistical Association.

[38] Little R, Rubin D (1987) Statistical Analysis with Missing Data. Series in Probability and Mathematical Statistics. John Wiley & Sons. New York.

[39] Little R, Rubin D (2002) Statistical analysis with missing data. Wiley. New York.

[40] Martín Martínez JL (1981) Ensayo de tipificación de los "sin opinión". Reis 16:9-37.

[41] Martos Peinado J (1995) Tratamiento estadístico de la no respuesta en poblaciones finitas. Tesis, Universidad de Sevilla.

[42] Medina F, Galván M (2007) Imputación de datos: teoría y práctica. CEPAL- Serie Estudios estadísticos y prospectivos nº 54.

[43] Mesa Ávila DM, Vieche Castro LM (2006) Una introducción a la imputación de valores perdidos. Terra Nueva Etapa, vol.22, nº 31, pp 127-151. Universidad Central de Venezuela. Caracas, Venezuela.

[44] Neyman J (1937) Outline of a theory of statistical estimation based on the classical theory of probability. Philosophical Transactions of the Royal Society of London. Series A.

[45] Neyman J, Pearson ES (1933) On the problem of most efficient test of statistical hypotheses. Philosiphical Transactions of the Royal Society of London. Series A.

[46] Otero García D (2011) Imputación de datos faltantes en un sistema de Información sobre Conductas de Riesgo. Trabajo Fin de Máster. Máster en Técnicas Estadísticas. Universidad de Vigo, Universidad de Santiago de Compostela, Universidad de Coruña.

[47] Plateck R (1986) Metodología y tratamiento de la no-respuesta. EUSTAT. Vitoria.

[48] Politz AN, Simmons WR (1949) An attempt to get the "not at homes" into the sample without call-backs. JASA 44:9-31.

[49] Robitzsch A, Grund S, Henke T (2017) miceadds: Some Additional Multiple Imputation Functions, Especially for "mice". R package versión 2.3-0. http://www.cran.r-project.org/package=miceadds. Accedido 1 de febrero de 2017.

[50] Rubin DB (1977) Assignment to Treatment Group on the Bases of a covariate. Journal of Educational Statistics. vol.2, pp 1-26.

[51] Rubin DB (1976) Inference and missing data. Biometrika 63:581-592.

[52] Rubin DB (1996) Multiple Imputation after 18+ years (with discussion). Journal of the American Statistical Association 91.

[53] Rubin DB (1987) Multiple Imputation for Nonresponse in Surveys. Wiley. New York.

[54] Schafer JL (1997) Analysis of Incomplete Multivariate Data. Chapman & Hall. London, UK.

[55] Schafer JL (1999) Multiple Imputation, o prime Statistical Methods in Medical Research 8.

[56] Stckhoven Daniel S (2013) missForest: Nonparametric Missing Value Imputation using Random Forest. R package versión 1.4. http://www.cran.r-project.org/package=missForest. Accedido 1 de febrero de 2017.

[57] Steffen M (2016) imputeTs: Time Series Missing Value Imputation. R package versión 1.8. http://www.cran.r-project.org/package=imputeTs. Accedido 1 de febrero de 2017.

[58] Templ M, Alfons A, Kowarsk A, Pranter B (2016) VIM: Visualization and Imputation of Missing Values. R package versión 4.6.0. http://www.cran.r-project.org/package=VIM. Accedido 1 de febrero de 2017.

[59] Thomsen I, Siring E (1983) On the causes and effects nonresponse: Norwegian Experiences. Incomplete data in sample surveys, vol.3, pp 25-29.

[60] Todeschini R (1990) Weighted K-nearest neighbour method for the calculation of missing values. Chenometrics and Intelligent Laboratory Systems 9:201-205.

[61] Van Buuren S, Groothvis-Oudshoorn K, Robitzsch A, Vink G, Doove L, Jolani S, Schouten R, Gaffert P, Meindfelder F (2017) mice: Multivariate Imputation by Chained Equations. R package versión 2.30. http://www.cran.r-project.org/package=mice. Accedido 1 de febrero de 2017.

[62] Van der Loo M (2017) simputation: Simple Imputation. R package versión 0.2.1. http://www.cran.r-project.org/package=simputation. Accedido 1 de febrero de 2017.

[63] Villán I, Bravo MS (1990) Procedimientos de depuración de datos estadísticos. EUSTAT. Victoria.

[64] Villar I (1992) Análisis de reglas de depuración de datos. EE, vol.34, nº 129, pp 151-171.

[65] Warner SL (1965) Randomized response: A survey technique for eliminating evasive answer bias. JASA 60: 63-69.

[66] Wilks S (1932) Moments and distribution of estimates of population parameters from fragmentary simple. Annals of Mathematical Statistics B, pp 163-195.

[67] Zhang P (2003) Multiple imputation: theory and method. International Statistical Review 71/3:581-592.

Índice de figuras

Índice de cuadros

Apéndice A

Códigos en lenguaje de programación R

A.1. Paquete imputeTs

```
# Paquete imputeTS
install.packages("imputeTS")
library(imputeTS)
########################
# Funcion na.Kalman ####
########################
data("tsAirgap")
head(tsAirgap)
View(tsAirgap)
# Ejemplo 1: Realiza la imputacion con KalmanSmoother y representa un modelo
# Arima estado espacio
na.kalman(tsAirgap)
# Ejemplo 2: Realiza la imputacion con KalmanRun y representa un modelo espacial
# arima
na.kalman(tsAirgap, smooth = FALSE)
# Ejemplo 3: Realiza la imputacion con el modelo KalmanSmooth y Structs
na.kalman(tsAirgap, model ="StructTS", smooth = TRUE)
# Ejemplo 4: Realiza la imputacion con el modelo KalmanSmooth y Structs con
# parametros adicionales
na.kalman(tsAirgap, model ="StructTS", smooth = TRUE, type ="trend")
# Ejemplo 5: Realiza la imputacion con KalmanSmooth y el modelo creado por
# el usuario
usermodel <- arima(tsAirgap,order = c(1,0,1))$model
na.kalman(tsAirgap,model = usermodel)
######################
# Funcion na.mean ####
######################
# Requisito: crear series de tiempo con valores perdidos
x <- ts(c(2,3,4,5,6,NA,7,8));x
# Ejemplo 1: Realiza la imputacion con el promedio general
na.mean(x)
# Ejemplo 2: Realiza la imputacion con la mediana general
na.mean(x, option ="median")
######################
# Funcion na.seadec ###
######################
data("tsAirgap")
# Ejemplo 1 : Realiza la imputacion estacional utilizando el
# algoritmo ="interpolation"
na.seadec(tsAirgap, algorithm = "interpolation")
# Ejemplo 2: Realiza la imputacion estacional utilizando el algoritmo= "mean"
na.seadec(tsAirgap, algorithm = "mean")
#########################
# Funcion na.seasplit ####
#########################
# Ejemplo 1: Realiza la imputacion dividida estacional utilizando el
# algoritmo ="interpolation"
na.seasplit(tsAirgap, algorithm = "interpolation")
# Ejemplo 2: Realiza la imputacion dividida estacional utilizando el
# algoritmo="mean"
na.seasplit(tsAirgap, algorithm = "mean")
```

A.2. Paquete HotDeckImputation

```
# Paquete HotDeckImputation
install.packages("HotDeckImputation")
```

```
library(HotDeckImputation)
##############################
# Funcion impute.CPS_SEQ_HD ###
##############################
# Estable la semilla aleatoria en un numero arbitrario
set.seed(421)
n<-10
m<-3
pmiss<-.1
# Genera una matriz de enteros aleatorios y 2 covariables binarias
Y<-cbind(matrix(sample(0:1,replace=TRUE,size=n*2),nrow=n),
         matrix(sample(0:9,replace=TRUE,size=n*m),nrow=n))
# Genera valores perdidos MCAR en todas las columnas excepto en las dos
# primeras
Y[,-c(1,2)][sample(1:length(Y[,-c(1,2)]),
                  size=floor(pmiss*length(Y[,-c(1,2)])))]<-NA
# Realiza la imputacion secuencial de Y dentro de las clases creadas mediante
# la clasificacion cruzada de las variables 1 y 2
impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0,
                  navalues=NA, modifyinplace = FALSE)
# Un ejemplo resaltanto la opcion modifyinplace usando cbind para mostrar
# los resultados de la funcion y los datos iniciales
cbind(impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0,
                        navalues=NA, modifyinplace = FALSE),Y)
# notar que las columnas 8-10 de Y siguen representando valores
# faltantes
# Mismo procedimiento, excepto que modifyinplace esta establecido en TRUE
cbind(impute.CPS_SEQ_HD(DATA=Y,covariates=c(1,2),initialvalues=0,
                        navalues=NA, modifyinplace = TRUE),Y)
# notar que las columnas 8-10 representando Y son indenticas a las
# columnas 3-5
############################
# Funcion impute.SEQ_HD #####
############################
# Estable la semilla aleatoria en un numero arbitrario
set.seed(421)
n<-10
m<-5
pmiss<-.1
# Genera una matriz de enteros aleatorios
Y<-matrix(sample(0:9,replace=TRUE,size=n*m),nrow=n)
# Genera 4 valores perdidos, MCAR
Y[-1,][sample(1:length(Y[-1,]),size=floor(pmiss*length(Y[-1,])))]<-NA
# Realiza la imputacion secuencial de Y
impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA, modifyinplace = FALSE)
# Un ejemplo resaltando la opcion modidyinplace usando cbind para mostrar
# los resultados de la funcion y los datos iniciales
cbind(impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA,
                    modifyinplace = FALSE),Y)
# notar que las columnas 6,8 y 9 (representando Y) todavia tienen datos faltantes
# Mismo procedimiento, excepto que modifyinplace esta establecido en TRUE
cbind(impute.SEQ_HD(DATA=Y,initialvalues=0, navalues=NA,
                    modifyinplace = TRUE),Y)
# notar que las columnas 1-5 (representando Y) son identicas que las
# columnas 6-10, dado que todas las variables que impuntan a Y se han modificado
# directamente
```

A.3. Paquete longitudinalData

```
# Paquete longitudinalData
install.packages("longitudinalData")
library(longitudinalData)
#########################
# Funcion imputation ####
#########################
# Preparacion de datos
par(ask=TRUE)
timeV <- 1:14
matMissing <- matrix(
  c(NA ,NA ,NA ,18 ,22 ,NA ,NA ,NA , 24 , 22 , NA , NA , NA,
    24 ,21 ,24 ,26 ,27 ,32 ,30 ,22 ,26 , 26 , 28 , 24 , 23 , 21,
    14 ,13 , 10 , 8 , 7 ,18 ,16 , 8 ,12 , 6 , 10 , 10 , 9 , 7,
    3 ,1 , 1 , 1 , 3,9 , 7 , -1 , 3 , 2 , 4 , 1 , 0 , -2
  ),4,byrow=TRUE
)
par(mfrow=c(1,1))
matplot(t(matMissing),col=c(2,1,1,1),lty=1,type="l",lwd=c(3,1,1,1),pch=16,
xlab="Black=trajectories; Green=mean trajectory\nRed=trajectory to impute",
ylab="",main="Four trajectories")
moy <- apply(matMissing,2,mean,na.rm=TRUE)
lines(moy,col=3,lwd=3)
# Ilustracion de diferente metodos de imputacion
# Metodos que utilizan informacion transversal (metodos cruzados)
par(mfrow=c(1,3))
```

```
mat2 <- matrix(c(
  NA, 9, 8, 8, 7, 6,NA,
  7, 6,NA,NA,NA, 4,5,
  3, 4, 3,NA,NA, 2,3,
  NA,NA, 1,NA,NA, 1,1),4,7,byrow=TRUE)
mat2
# crossMean
matplot(t(imputation(mat2,"crossMean")),type="l",ylim=c(0,10),
        lty=1,col=1,main="crossMean")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# crossMedian
matplot(t(imputation(mat2,"crossMedian")),type="l",ylim=c(0,10),
        lty=1,col=1,main="crossMedian")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# crossHotDeck
matplot(t(imputation(mat2,"crossHotDeck")),type="l",ylim=c(0,10),
        lty=1,col=1,main="crossHotDeck")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# Metodos que utilizan la informacion de la trayectoria
par(mfrow=c(2,3))
mat1 <- matrix(c(NA,NA,3,8,NA,NA,2,2,1,NA,NA),1,11)
mat1
# locf
matplot(t(imputation(mat1,"locf")),type="l",ylim=c(0,10),
        main="locf")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# nocb
matplot(t(imputation(mat1,"nocb")),type="l",ylim=c(0,10),
        main="nocb")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# trajMean
matplot(t(imputation(mat1,"trajMean")),type="l",ylim=c(0,10),
        main="trajMean")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# trajMedian
matplot(t(imputation(mat1,"trajMedian")),type="l",ylim=c(0,10),
        main="trajMedian")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# trajHotDeck
matplot(t(imputation(mat1,"trajHotDeck")),type="l",ylim=c(0,10),
        main="trajHotDeck 1")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# Spline
matplot(t(imputation(mat1,"spline",lowerBound=NA,upperBound=NA)),
        type="l",ylim=c(-10,10),main="spline")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16)
# Diferentes interpolaciones lineales
par(mfrow=c(2,2))
# linearInterpol.locf
matplot(t(imputation(mat1,"linearInterpol.locf",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="linearInterpol.locf")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16,lty=1)
# linearInterpol.global
matplot(t(imputation(mat1,"linearInterpol.global",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="linearInterpol.global")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16,lty=1)
# linearInterpol.local
matplot(t(imputation(mat1,"linearInterpol.local",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="linearInterpol.local")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16,lty=1)
# linearInterpol.bisector
matplot(t(imputation(mat1,"linearInterpol.bisector",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="linearInterpol.bisector")
matlines(t(mat1),type="o",col=2,lwd=3,pch=16,lty=1)
# Metodos copyMean
mat3 <- matrix(c(
  NA, 9, 8, 8, 7, 6,NA,
  7, 6,NA,NA,NA, 4,5,
  3, 4, 3,NA,NA, 2,3,
  NA,NA, 1,NA,NA, 1,1),4,7,byrow=TRUE)
mat3
par(mfrow=c(2,2))
# copyMean.locf
matplot(t(imputation(mat2,"copyMean.locf",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="copyMean.locf")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# copyMean.global
matplot(t(imputation(mat2,"copyMean.global",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="copyMean.global")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# copyMean.local
matplot(t(imputation(mat2,"copyMean.local",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="copyMean.local")
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# copyMean.bisector
matplot(t(imputation(mat2,"copyMean.bisector",NA,NA)),type="l",
        ylim=c(-5,10),lty=1,col=1,main="copyMean.bisector")
```

```
matlines(t(mat2),type="o",col=2,lwd=3,pch=16,lty=1)
# Metodos crossMean
par(mfrow=c(1,3))
matImp <- imputation(matMissing,method="crossMean")
matImp
matplot(t(matImp),col=c(2,1,1,1),lty=c(2,1,1,1),type="l",lwd=c(2,1,1,1),
pch=16,
xlab="Dotted red=imputed trajectory\nFull red=trajectory to impute",
ylab="",main="Method 'crossMean'")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# crossMedian
matImp <- imputation(matMissing,method="crossMedian")
matImp
matplot(t(matImp),col=c(2,1,1,1),lty=c(2,1,1,1),type="l",lwd=c(2,1,1,1),
pch=16,
xlab="Dotted red=imputed trajectory\nFull red=trajectory to impute",ylab="",
        main="Method 'crossMedian'")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# crossHotDeck
matImp <- imputation(matMissing,method="crossHotDeck")
matImp
matplot(t(matImp),col=c(2,1,1,1),lty=c(2,1,1,1),type="l",lwd=c(2,1,1,1),
pch=16,
xlab="Dotted red=imputed trajectory\nFull red=trajectory to impute",ylab="",
        main="Method 'crossHotDeck'")
# Metodos usando trayectoria
par(mfrow=c(2,3))
# trajMean
matImp <- imputation(matMissing,method="trajMean")
plot(timeV,matImp[1,],type="l",lwd=2,ylim=c(10,30),main="trajMean",
ylab="",xlab="nocb")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# trajMedian
matImp <- imputation(matMissing,method="trajMedian")
plot(timeV,matImp[1,],type="l",lwd=2,ylim=c(10,30),main="trajMedian",
ylab="",xlab="nocb")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# trajHotDeck
matImp <- imputation(matMissing,method="trajHotDeck")
plot(timeV,matImp[1,],type="l",lwd=2,ylim=c(10,30),main="trajHotDeck",
ylab="",xlab="nocb")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# locf
matImp <- imputation(matMissing,method="locf")
plot(timeV,matImp[1,],type="l",lwd=2,ylim=c(10,30),main="locf",
ylab="",xlab="locf")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
#nocb
matImp <- imputation(matMissing,method="nocb")
plot(timeV,matImp[1,],type="l",lwd=2,ylim=c(10,30),main="nocb",
ylab="",xlab="nocb")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
par(mfrow=c(2,2))
# linearInterpol.locf
matImp <- imputation(matMissing,method="linearInterpol.locf")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="linearInterpol.locf",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# linearInterpol.local
matImp <- imputation(matMissing,method="linearInterpol.local")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="linearInterpol.local",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
# linearInterpol.global
matImp <- imputation(matMissing,method="linearInterpol.global")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="linearInterpol.global",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
#linearInterpol.bisector
matImp <- imputation(matMissing,method="linearInterpol.bisector")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="linearInterpol.bisector",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
par(mfrow=c(2,2))
# copyMean.locf
matImp <- imputation(matMissing,method="copyMean.locf")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="copyMean.locf",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
lines(timeV,moy,col=3,type="o",lwd=3)
# copyMean.local
matImp <- imputation(matMissing,method="copyMean.local")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="copyMean.local",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
lines(timeV,moy,col=3,type="o",lwd=3)
```

```
# copyMean.global
matImp <- imputation(matMissing,method="copyMean.global")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="copyMean.global",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
lines(timeV,moy,col=3,type="o",lwd=3)
#copyMean.bisector
matImp <- imputation(matMissing,method="copyMean.bisector")
plot(timeV,matImp[1,],type="o",ylim=c(0,30),main="copyMean.bisector",
ylab="",xlab="LI-Global")
lines(timeV,matMissing[1,],col=2,type="o",lwd=3)
lines(timeV,moy,col=3,type="o",lwd=3)
par(ask=FALSE)
```

A.4. Paquete missForest

```
# Paquete missForest
install.packages("missForest")
library(missForest)
########################
# Funcion missForest ###
########################
# Imputacion de valor faltante no parametrico en datos de tipo mixto
data(iris)
head(iris)
summary(iris)
# Los datos contienen cuatro variables continuas y una variable categorica
# Producimos artificialmente los valores perdidos usando la funcion "prodNA"
set.seed(81)
iris.mis <- prodNA(iris, noNA = 0.2)
summary(iris.mis)
# Imponemos los valores faltantes proporcionando la matriz completa para la
# ilustracion usando el argumento verbose para ver lo que sucede entre las
# iteracciones
iris.imp <- missForest(iris.mis, xtrue = iris, verbose = TRUE)
# La imputacion termina despues de 5 iteracciones que tiene un NMRSE verdadero
# final de 0.143 y un PFC de 0.036. EL NMRSE final estimado es de 0.157 y el PFC
# de 0.025
# Se puede acceder a los resultados finales directamente con el siguiente
# comando
iris.imp$OOBerror
# El verdadero error de imputacion
iris.imp$error
# la matriz de datos imputados
iris.imp$ximp
########################
# Funcion mixError #####
########################
# Error de calculo de imputacion para datos de tipo mixto
data(iris)
head(iris)
# Producimos artificialmente los valores perdidos usando la funcion "prodNA"
set.seed(81)
iris.mis <- prodNA(iris, noNA = 0.2)
# Imponemos valores faltantes usando la funcion missForest
iris.imp <- missForest(iris.mis)
# Calcula manualmente el error de imputacion verdadero
err.imp <- mixError(iris.imp$ximp, iris.mis, iris)
err.imp
```

A.5. Paquete ForImp

```
# Paquete Forlmp
install.packages("ForImp")
library(ForImp)
####################
# Funcion ForImp ###
####################
set.seed(1)
# Matriz de correlaciones
sigma<-matrix(c(1,0.5,0.5,0.5,0.5,1,0.5,0.5,0.5,0.5,1,0.5,0.5,0.5,0.5,1),4,4)
sigma
# Genera una matriz de 10*4 de una matriz normal multivariante
matc<-rmvnorm(n=10, mean=rep(0,4), sigma=sigma)
matc
# Transformamos los valores numericos en categorias ordinales
mato<-transfmatcat(matc,4)
mato
# Establecemos el numero de valores perdidos deseados
nummissing<-5
```

```
# Creamos los valores faltantes al azar
mat<-missingmat(mato, nummissing, pattern="r")
mat
# Usamos la funcion \ code {ForImp} para imputar valores faltantes,
# obteniendo la matriz mati
mati<-ForImp(mat)
mati
# Numero de imputaciones correctas
nummissing-sum(mati!=mato)
#####################
# Funcion meanimp ###
#####################
set.seed(1)
n<-10
m<-3
mat<-matrix(rnorm(n*m),n,m)
matm<-mat
matm[1,1]<-NA
matm[2,2:3]<-NA
# Matriz con valores perdidos
matm
# Matriz imputada
meanimp(mat)
# Matriz original sin valores faltantes
mat
######################
# Funcion medianimp ###
######################
set.seed(1)
n<-10
m<-3
mat<-matrix(ceiling(runif(n*m)*4),n,m)
matm<-mat
matm[1,3]<-NA
matm[9:10,1]<-NA
# Matriz con valores perdidos
matm
# Matriz imputada
medianimp(matm)
# Matriz original sin valores faltantes
mat
####################
# Funcion modeimp ###
####################
set.seed(1)
n<-10
m<-3
mat<-matrix(ceiling(runif(n*m)*4),n,m)
matm<-mat
matm[1,3]<-NA
matm[9:10,1]<-NA
# Matriz con valores perdidos
matm
# Matriz imputada
modeimp(mat)
# Matriz original sin valores perdidos
mat
```

A.6. Paquete BaBoon

```
# Paquete BaBoon
install.packages("BaBoon")
library(BaBooN)
#################################
# Aplicacion de la funcion BBPM ##
#################################
# Conjunto de datos de muestra con variables no normales
set.seed(1000)
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- as.factor(ifelse(y1>4,5,y1))
y2 <- x1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
misrow2 <- sample(n,15)
misrow3 <- sample(n,10)
is.na(data1[misrow1, 4]) <- TRUE
is.na(data1[misrow2, 5]) <- TRUE
is.na(data1[misrow2, 6]) <- TRUE
```

```
# Imputacion
imputed.data <- BBPMM(data1, nIter=5, M=5)
# Valores
imputed.data$call
imputed.data$mis.num
imputed.data$modelselection
imputed.data$seed
imputed.data$impdata
imputed.data$misOverview
imputed.data$indMatrix
imputed.data$M
imputed.data$nIter
imputed.data$Chains
imputed.data$FirstSeed
imputed.data$LastSeed
imputed.data$ignoredvariables
#######################################
# Aplicacion de la funcion BBPMM.row ###
#######################################
# Conjunto de datos de muestra con variables no normales y un patron de
# datos perdidos
set.seed(1000)
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- ifelse(y1>4,5,y1)
y2 <- y1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
data <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
data[misrow1, c(4:6)] <- NA
# Paso de preparacion
impblock <- rowimpPrep(data)
# Imputacion
imputed.data <- BBPMM.row(impblock, M=5)
# Valores
imputed.data$call
imputed.data$mis.num
imputed.data$modelselection
imputed.data$seed
imputed.data$impdata
imputed.data$indMatrix
imputed.data$M
imputed.data$weightMatrix
imputed.data$model
imputed.data$pairlst
imputed.data$FirstSeed
imputed.data$LastSeed
imputed.data$ignoredvariables
#################################
# Aplicacion de la funcion dmi ###
#################################
# sobre el conjunto de datos anterior
dmi(data)
############################################
install.packages(MASS)
library(MASS)
data(survey)
# Clasificacion via normal prelim.norm
install.packages("Hmisc")
library(Hmisc)
survey.numeric <- asNumericMatrix(survey)
install.packages("norm")
library(norm)
su.sort <- prelim.norm(survey.numeric)
new.survey <- survey[order(su.sort$ro),
                     sort(su.sort$nmis,index.return=TRUE)$ix]
# Comparacion
dmi(survey) # original
dmi(new.survey) # clasificado
#####################################################
# Aplicacion de la funcion impdiagnosticconversion ###
#####################################################
# Conjunto de datos de muestra con variables no normales
set.seed(1000)
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- as.factor(ifelse(y1>4,5,y1))
y2 <- x1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
```

```
data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
misrow2 <- sample(n,15)
misrow3 <- sample(n,10)
is.na(data1[misrow1, 4]) <- TRUE
is.na(data1[misrow2, 5]) <- TRUE
is.na(data1[misrow2, 6]) <- TRUE
# Imputacion
imputed.data <- BBPMM(data1, nIter=3, M=3)
imputed.data$call
imputed.data$mis.num
imputed.data$seed
imputed.data$impdata
imputed.data$misOverview
imputed.data$indMatrix
imputed.data$M
imputed.data$nIter
imputed.data$Chains
imputed.data$FirstSeed
imputed.data$LastSeed
imputed.data$ignoredvariables
# Test de conversion
install.packages("coda")
library(coda)
install.packages("mice")
library(mice)
require(coda)
require(mice)
require(lattice)
# Conversion to mcmc
imp.to.mcmc <- impdiagnosticconversion(imputed.data,type="mcmc")
# Conversion to mcmc.list
imp.to.mcmc.list <- impdiagnosticconversion(imputed.data,
                                            type="mcmc.list")
# Test
#mcmc
par(mfrow=c(2,2))
plot(imp.to.mcmc$means[[1]])
acfplot(imp.to.mcmc$vars[[1]])
plot(imp.to.mcmc$medians[[1]])
acfplot(imp.to.mcmc$sds[[1]])
# mcm.list
par(mfrow=c(1,1))
xyplot(imp.to.mcmc.list[[1]]) # media
qqmath(imp.to.mcmc.list[[2]]) # varianza
xyplot(imp.to.mcmc.list[[3]]) # Mediana
qqmath(imp.to.mcmc.list[[4]]) # desviaciones tipicas o estandar
#mids
mice:::plot.mids(imp.to.mids)
########################################
# Aplicacion de la funcion MI.inference ###
########################################
# Ejemplo 1
n <- 100
x1 <- round(runif(n,0.5,3.5))
x2 <- round(runif(n,0.5,4.5))
x3 <- runif(n,1,6)
y1 <- round(x1-0.25*x2+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<2,2,y1)
y1 <- as.factor(ifelse(y1>4,5,y1))
y2 <- x3+rnorm(n,0,2)
y3 <- as.factor(ifelse(x2+rnorm(n,0,2)>2,1,0))
mis1 <- sample(100,20)
mis2 <- sample(100,30)
mis3 <- sample(100,25)
data1 <- data.frame("x1"=x1,"x2"=x2,"x3"=x3,
                    "y1"=y1,"y2"=y2,"y3"=y3)
is.na(data1$y1[mis1]) <- TRUE
is.na(data1$y2[mis2]) <- TRUE
is.na(data1$y3[mis3]) <- TRUE
imputed.data <- BBPMM(data1, M=5, nIter=5)
MI.m.meany2.hat <- sapply(imputed.data$impdata,
                          FUN=function(x) mean(x$y2))

MI.v.meany2.hat <- sapply(imputed.data$impdata,
                          FUN=function(x) var(x$y2)/length(x$y2))
# MI.inference
MI.y2 <- MI.inference(MI.m.meany2.hat,
                      MI.v.meany2.hat, alpha=0.05)
MI.y2$MI.Est
MI.y2$MI.Var
MI.y2$CI.low
MI.y2$CI.up
MI.y2$BVar
MI.y2$WVar
# Ejemplo 2
# Funcion adicional simple para calcular coberturas
```

```
coverage <- function(value, bounds) {
  ifelse(min(bounds) <= value && max(bounds) >= value, 1, 0)
}
# value: valor verdadero
# bounds: vector con dos elementos (limite superior y inferior CI)
# muestra
n <- 100
# Valor verdadero para la media de y2
m.y2 <- 3.5
y2.cover <- vector(length=n)
set.seed(1000)
# Generamos 100 datos
time1 <- Sys.time()
for (i in 1:100) {
  x1 <- round(runif(n,0.5,3.5))
  x2 <- round(runif(n,0.5,4.5))
  x3 <- runif(n,1,6)
  y1 <- round(x1-0.25*x2+0.5*x3+rnorm(n,0,1))
  y1 <- ifelse(y1<2,2,y1)
  y1 <- as.factor(ifelse(y1>4,5,y1))
  y2 <- x3+rnorm(n,0,2)
  y3 <- as.factor(ifelse(x2+rnorm(n,0,2)>2,1,0))
  mis1 <- sample(n,20)
  mis2 <- sample(n,30)
  mis3 <- sample(n,25)
  data1 <- data.frame("x1"=x1,"x2"=x2,"x3"=x3,
                      "y1"=y1,"y2"=y2,"y3"=y3)
  is.na(data1$y1[mis1]) <- TRUE
  is.na(data1$y2[mis2]) <- TRUE
  is.na(data1$y3[mis3]) <- TRUE
  sim.imp <- BBPMM(data1, M=3, nIter=2,
                   stepmod="", verbose=FALSE)
  MI.m.meany2.hat <- sapply(sim.imp$impdata,
                            FUN=function(x) mean(x$y2))
  MI.v.meany2.hat <- sapply(sim.imp$impdata,
                            FUN=function(x)
                              var(x$y2)/length(x$y2))
# MI.inference

  MI.y2 <- MI.inference(MI.m.meany2.hat, MI.v.meany2.hat,
                        alpha=0.05)
  y2.cover[i] <- coverage(m.y2, c(MI.y2$CI.low,MI.y2$CI.up))
}
time2 <- Sys.time()
difftime(time2, time1, unit="secs")
# estimador de cobertura
mean(y2.cover)
########################################
# Aplicacion de la funcion rowinmpPrep ###
########################################
# Conjunto de datos de muestra con variables no normales y patron de
# datos perdidos
set.seed(1000)
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- ifelse(y1>4,5,y1)
y2 <- y1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
is.na(data1[misrow1, c(4:6)]) <- TRUE
# Paso de preparacion
impblock <- rowimpPrep(data1)
impblock$blockNames
impblock$data
impblock$key
impblock$blocks
impblock$compNames
impblock$ignore
impblock$ignored_data
impblock$indMatrix
########################################
# Aplicacion de la funcion summary.imp ###
########################################
# Conjunto de datos de muestra con variables no normales y dos patrones de
# falta de datos
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- as.factor(ifelse(y1>4,5,y1))
```

```
y2 <- x1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
misrow2 <- sample(n,15)
misrow3 <- sample(n,10)
is.na(data1[misrow1, 4]) <- TRUE
is.na(data1[misrow2, 5]) <- TRUE
is.na(data1[misrow2, 6]) <- TRUE
# Imputacion
imputed.data <- BBPMM(data1, nIter=5, M=5)
summary(imputed.data)
##############################################
# Aplicacion de la funcion summary.impprep ##
##############################################
# Conjunto de datos de muestra con variables no normales y un patron de
# datos perdidos
set.seed(1000)
n <- 50
x1 <- round(runif(n,0.5,3.5))
x2 <- as.factor(c(rep(1,10),rep(2,25),rep(3,15)))
x3 <- round(rnorm(n,0,3))
y1 <- round(x1-0.25*(x2==2)+0.5*x3+rnorm(n,0,1))
y1 <- ifelse(y1<1,1,y1)
y1 <- ifelse(y1>4,5,y1)
y2 <- y1+rnorm(n,0,0.5)
y3 <- round(x3+rnorm(n,0,2))
data1 <- as.data.frame(cbind(x1,x2,x3,y1,y2,y3))
misrow1 <- sample(n,20)
is.na(data1[misrow1, c(4:6)]) <- TRUE
# Paso de preparacion
impblock <- rowimpPrep(data1)
summary(impblock)
```