

P.I
(043) 62
F 830
2022

PROYECTO INTEGRADOR DE LA CARRERA DE INGENIERÍA NUCLEAR

OPTIMIZACIÓN DE ALGORITMOS DE ESTIMACIÓN DE DENSIDADES PARA EL CÁLCULO DE HACES DE NEUTRONES Y FOTONES.

Francisco Fox

Dr. Ariel Marquez

Director

Mgter. Norberto Schmidt

Co-director

Miembros del Jurado

Mgter. Roberto Maturana (Instituto Balseiro)

Ing. Daniel Hergenreder (INVAP)

14 de Junio de 2022

Departamento de Física de Reactores y Radiaciones – Centro
Atómico Bariloche

Instituto Balseiro
Universidad Nacional de Cuyo
Comisión Nacional de Energía Atómica
Argentina

(Biblioteca Leo Falicov CAB-IB)

Inventario 24603
25/08/2022

A Grannie, y Maripi.

Índice de contenidos

Índice de contenidos	v
Índice de figuras	vii
Índice de tablas	xi
Índice de símbolos	xiii
Resumen	xv
Abstract	xvii
1. Introducción	1
1.1. Motivación	1
1.2. Objetivos	1
1.3. Metodología Conceptual	2
1.4. Herramientas utilizadas	3
1.4.1. OpenMC	3
1.4.2. KDSource	4
1.4.3. McStas	4
1.5. Esquema de la tesis	4
2. KDSource	7
2.1. Algoritmos KDE	7
2.1.1. Kernel Density Estimation	7
2.1.2. KDE unidimensional	9
2.1.3. Funciones kernel	10
2.1.4. Variación de ancho de banda	11
2.2. Componentes y estructura de la herramienta	13
2.3. Flujo de trabajo	14
2.4. Nuevas implementaciones	15

3. LINAC	17
3.1. Descripción general	17
3.2. Resultados	20
3.2.1. Elección de ancho de banda	20
3.2.2. Variación de función <i>kernel</i>	23
3.2.3. Cambio de variables	26
3.2.4. Espectro	28
4. Haz de Neutrografía RA-6	31
4.1. Descripción general	31
4.2. Simulaciones utilizando KDSource	32
4.2.1. Fluxos	34
4.2.2. Tasa de dosis ambiental	37
4.2.3. Cociente dosis-flujo	41
5. Conclusiones	45
5.1. Conclusiones	45
5.2. Trabajo a futuro	46
A. Validación del código OpenMC: Benchmark de criticidad	47
B. Cálculos de dosis del haz de neutrografía	49
C. Modificaciones del código	51
D. Actividades relacionadas a la Práctica Profesional Supervisada	55
Bibliografía	57
Agradecimientos	59

Índice de figuras

1.1.	Esquema de la metodología conceptual en la implementación de algoritmos de KDE en la generación de fuentes distribucionales de radiación.	3
2.1.	Procedimiento simplificado (unidimensional) para la implementación de algoritmos de KDE a un conjunto de datos.	9
2.2.	Comparación de las gráficas de las distribuciones gausiana y Epanechnikov con ancho de banda h	11
2.3.	Comparación entre distintos anchos de banda para un mismo conjunto de datos aplicando KDE con <i>kernel</i> gaussiano.	12
2.4.	Flujo de trabajo de la herramienta KDSource, con su respectivo manejo de archivos.	15
3.1.	Esquema de la geometría utilizada para la medición del espectro de neutrones en agua liviana. En la parte superior se observa el dispositivo de tiempo de vuelo utilizado. En la parte inferior se ve con mayor detalle la fuente de neutrones, que inciden en el moderador por arriba y se extraen por el tubo reentrantte hacia la derecha para incidir en el tubo de 17 m de largo. Todas las unidades son en mm. Fuente: [1]	18
3.2.	Flujo neutrónico rápido y térmico, obtenidos a partir de la simulación con OpenMC para el experimento del LINAC.	19
3.3.	Superficies de <i>tracks</i> para la simulación del LINAC en OpenMC.	19
3.4.	Distribuciones de coordenadas espaciales usando algoritmos de KDE con <i>kernel</i> gaussiano y variando el ancho de banda.	21
3.5.	Distribuciones de coordenadas direccionales u y v usando algoritmos de KDE con <i>kernel</i> gaussiano y variando el ancho de banda.	22
3.6.	Distribuciones de variable w usando algoritmos de KDE con <i>kernel</i> gaussiano y variando el ancho de banda.	23
3.7.	Comparación de distribuciones espaciales utilizando algoritmos de KDE con distintas funciones <i>kernel</i>	24
3.8.	Comparación de distribución en w utilizando algoritmos de KDE con distintas funciones <i>kernel</i>	25

3.9. Comparación de distribución en x utilizando herramienta KDSource con distintas funciones <i>kernel</i> . La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.	25
3.10. Comparación de distribución en y utilizando herramienta KDSource con distintas funciones <i>kernel</i> . La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.	26
3.11. Distribución en el radio ρ utilizando distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.	27
3.12. Comparación de distribución en x utilizando herramienta KDSource con distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.	28
3.13. Comparación de distribución en y utilizando herramienta KDSource con distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.	28
3.14. Espectro neutrónico de la fuente generada con la herramienta KDSource, en un extremo del tubo de vuelo del LINAC.	29
3.15. Espectro neutrónico de la fuente generada con la herramienta KDSource, en un extremo del tubo de vuelo del LINAC.	29
3.16. Espectro neutrónico obtenido en el extremo del tubo de vuelo del LINAC, simulado con McStas y utilizando la fuente generada en KDSource.	30
4.1. Esquema del dispositivo del neutrografía perteneciente al conducto N1 del reactor RA-6, modelada en OpenMC en vista superior	32
4.2. Geometrías del haz de neutrografía utilizadas en las simulaciones en OpenMC.	33
4.3. Esquema de trabajo realizado para el haz de neutrografía, utilizando OpenMC y KDSource.	33
4.4. Esquema del flujo de partículas simuladas para el haz de neutrografía, utilizando OpenMC y KDSource.	34
4.5. Flujos de neutrones y fotones obtenidos en la primera simulación de OpenMC con fuente de neutrones en la superficie $S1$	35
4.6. Flujo de fotones obtenido en la primera simulación de OpenMC con fuente de fotones en la superficie $S1$	36
4.7. Flujos de neutrones y fotones obtenido en segunda simulación de OpenMC con fuentes en la superficie $S2$	37
4.8. Tasa de dosis ambiental aportada por neutrones y fotones, obtenida a partir de la simulación de OpenMC con fuente en la superficie $S2$	38

4.9. Puntos de interés para la comparación del cálculo de tasa de dosis ambiental en las afueras del dispositivo de neutrografía.	39
4.10. Comparación de tasa de dosis ambiental por neutrones obtenidas con mediciones experimentales y simulaciones.	40
4.11. Comparación de tasa de dosis ambiental por fotones obtenidas con mediciones experimentales y simulaciones.	40
4.12. Gráfica de la relación del factor de conversión de fluencia en dosis equivalente ambiental con la energía, para neutrones y fotones.	42
4.13. Espectro de neutrones y fotones en la entrada del colimador del haz, en el Conducto 1 del reactor RA-6.	42
4.14. Cociente entre la tasa de dosis y el flujo obtenido para las simulaciones del dispositivo de neutrografía utilizando OpenMC junto con KDSource.	43
A.1. Esquema de la geometría utilizada para el <i>benchmark</i> de criticidad ZPR-6/6A. Imagen obtenida de documento un documento de NEA [2].	47

Índice de tablas

3.1. Relación entre partículas simuladas de fuente, partículas en archivo de <i>tracks</i> y tiempo de simulación en OpenMC.	19
4.1. Tabla comparativa de valores de errores relativos para el cálculo de dosis por las distintas metodologías de generación de fuentes de radiación para las simulaciones, con valores calculados (C) y experimentales (E). El promedio fue ponderado con la dosis obtenida para cada punto.	41
A.1. Tabla comparativa de los resultados experimentales, obtenidos con los datos del <i>benchmark</i> de MCNP y con la simulación propia realizada en OpenMC, junto con la diferencia de los resultados.	48
B.1. Comparación entre valores de tasa de dosis ambiental por neutrones, obtenidos con las distintas simulaciones.	49
B.2. Comparación entre valores de tasa de dosis ambiental por fotones, obtenidos con las distintas simulaciones.	50

Índice de símbolos

KDE: *Kernel Density Estimation.*

BW: Ancho de banda (*Bandwidth*).

w : Versor direccional en el eje z ($\cos(\theta)$).

N_{part} : Número de partículas simuladas.

$H^*(10)$: Factor de conversión de fleuencia a dosis ambiental.

LINAC: Acelerador lineal ubicado en el Centro Atomico Bariloche.

RA-6: Reactor nuclear de investigación ubicado en el Centro Atomico Bariloche.

Resumen

En el presente trabajo se optimizaron los algoritmos de estimación de densidades implementados en la herramienta computacional, KDSource. El objetivo consiste en variaciones en las metodologías de generación de fuentes distribucionales de radiación mediante la utilización de algoritmos de KDE, utilizando particularmente una función *kernel* del tipo Epanechnikov y coordenadas espaciales cilíndricas como sistema de referencia de las partículas.

Se realizó un primer desarrollo independiente, utilizando como caso de estudio el acelerador lineal LINAC. Se obtuvieron resultados satisfactorios en comparación con la metodología ya existente, la cual consiste en la utilización de *kernel* gaussiano y sistema de referencia cartesiano. En base a esta comparación se agregaron estas modificaciones a la herramienta de cálculo, para finalmente realizar una validación del código con datos experimentales tomados de la facilidad de neutrografía del reactor de investigación RA-6. Los resultados de la validación son satisfactorios comparados a los datos experimentales.

Se concluyó entonces que la optimización de cambio de *kernel* realizada en el código fue efectiva para el cálculo de haces de radiación. Asimismo, el uso de coordenadas espaciales cilíndricas fue satisfactorio en dispositivos con ese tipo de geometrías.

Palabras clave: KDE, KDSOURCE, LINAC, RA-6

Abstract

In the present work, optimizations were made on density estimation algorithms implemented in the computational tool, KDSource. The objective consisted of variations in the methodologies for generating distributional sources of radiation through the use of KDE algorithms, particularly the use of an Epanechnikov-like kernel function and cylindrical spacial coordinates as reference system of the particles.

A first independent development was carried out, using as case study the linear accelerator, LINAC. Satisfactory results were obtained in comparison with the existing methodology, which uses a Gaussian kernel and a Cartesian reference system. Based on this comparison, these modifications were added to the tool, to finally carry out a validation of the code with experimental data taken from the RA-6 research reactor neutrography facility. The validation results are satisfactory compared to the experimental data.

It was concluded that the kernel change optimization performed in the code was effective for the calculation of radiation beams. Likewise, the use of cylindrical spatial coordinates was satisfactory in devices with this type of geometries.

Keywords: KDE, KDSOURCE, LINAC, RA-6

Capítulo 1

Introducción

1.1. Motivación

En facilidades nucleares, donde existe una lejanía entre la fuente de radiación y el haz, es usualmente necesario evaluar la intensidad de la radiación en distintos puntos en condiciones de operación. Esto es debido a que se desea obtener parámetros radiológicos como puede ser la tasa de dosis ambiental recibida en ciertos lugares. Para eso se realizan simulaciones de transporte de radiación hacia el extremo del haz. Estos cálculos resultan computacionalmente costosos si se los realiza con una única simulación Monte Carlo, generando una cantidad relativamente baja de partículas muestreadas en el punto de interés. Una forma alternativa es el uso de un método multivariado adaptativo, el cual permite aumentar la estadística de las partículas conservando su correlación en el espacio de fases. Esta metodología permite estimar la distribución de la fuente en un punto dado de la trayectoria hacia el extremo del haz. De esta manera, se busca dividir la simulación en distintas etapas, reduciendo así el tiempo de cálculo. Esto se encuentra implementado en una herramienta computacional ya desarrollada llamada KDSource, la cual utiliza algoritmos de KDE (*Kernel Density Estimation*), que permiten generar fuentes distribucionales a partir de datos discretos.

Dada esta herramienta, se busca la mayor precisión posible en cuanto a los resultados obtenidos a partir de su utilización. Para eso se requieren mejoras que permitan reflejar con mayor certeza la física del problema a la hora de utilizarla en el cálculo de haces de radiación.

1.2. Objetivos

El objetivo principal de este trabajo es la optimización de los algoritmos de estimación de densidades de partículas en una superficie dada. Para eso de buscará implementar mejoras en la herramienta computacional KDSource, las cuales permitirán

obtener con mayor precisión densidades de probabilidad multivariadas asociadas a las partículas de dicha superficie.

Más precisamente, se busca implementar una función *kernel* del tipo Epanechnikov en los algoritmos de KDE para minimizar el efecto de muestreo fuera de las condiciones de bordes físicas del problema. Además, se introducirá un cambio de variables espaciales en coordenadas cilíndricas que permita mejorar los resultados en este tipo de geometrías. Estas mejoras aumentarán las capacidades de cálculo de la herramienta. Los resultados serán comparados, en primera instancia, con los datos experimentales obtenidos en el acelerador lineal (LINAC) ubicado en el Centro Atómico Bariloche.

Con el objetivo de continuar con la validación de la herramienta, se utilizará el Conducto 1 del reactor RA6 donde se encuentra ubicada la facilidad de neutrografía. Se busca utilizar las nuevas implementaciones de la herramienta KDSource para obtener valores de flujo y tasa de dosis ambiental, comparables con resultados experimentales.

1.3. Metodología Conceptual

Dado que la lejanía entre la fuente de radiación y el punto de interés no permite realizar una única simulación computacional por la falta de partículas, se decide dividir este cálculo en tres etapas principales, las cuales se pueden observar en la Figura 1.1.

1. En primer lugar, se realiza una simulación desde la fuente, la cual tendrá como objetivo guardar las partículas en una superficie de interés. Esta superficie se la denomina de *tracks*, y tendrá información sobre todas las dimensiones de las partículas que la atraviesan. La elección de esta superficie se realiza convenientemente, en una ubicación intermedia entre la fuente y el extremo del haz de radiación.
2. A partir de la superficie de *tracks*, se aplican los algoritmos de KDE en las partículas que la componen. Estos algoritmos buscan generar una fuente distribucional continua, conservando la correlación de las distintas dimensiones asociadas al espacio de fases. Luego, se realiza un muestreo sobre estas distribuciones generadas, de manera de obtener una nueva fuente con una cantidad de partículas considerablemente mayor a la original.
3. A continuación, esta nueva fuente generada servirá como *input* para una segunda simulación en Monte Carlo que busca guardar las partículas que llegan al extremo del haz. Así se podrá visualizar con mayor estadística la distribución de las partículas en esta región.

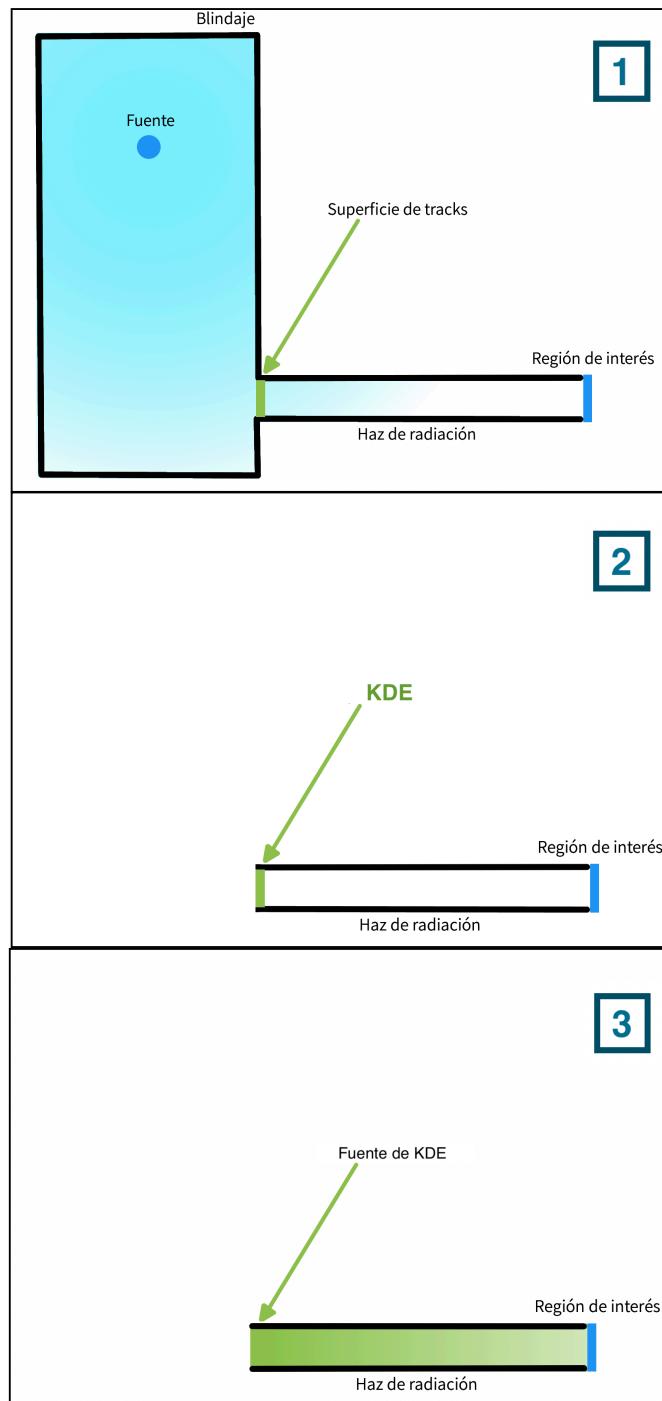


Figura 1.1: Esquema de la metodología conceptual en la implementación de algoritmos de KDE en la generación de fuentes distribucionales de radiación.

1.4. Herramientas utilizadas

1.4.1. OpenMC

OpenMC [3] es un código abierto de transporte de neutrones y fotones, por el método de Monte Carlo. Se pueden realizar simulaciones de fuente fija y de criticidad,

que permiten visualizar distintas magnitudes asociadas al transporte de radiación [4]. La biblioteca de secciones eficaces utilizada fue *ENDF/B-VII.1* [5], en formato *HDF5*.

Se encuentra escrito en el lenguaje C++, y existe a su vez una API (*Application Programming Interface*) en Python. Esta API permite definir y generar los archivos que servirán como *inputs* para realizar la simulación en el código fuente. Los archivos indispensables son: *materials.xml* (materiales), *geometry.xml* (geometría) y *settings.xml* (configuración de la simulación). Adicionalmente, pueden generarse archivos que actúen como contadores de partículas sobre cierto dominio del espacio de fases (*tallies.xml*), o como configuraciones de gráficos (*plots.xml*). Como *output* de una simulación de fuente fija se obtienen archivos de tipo *h5* que sirven para procesar y visualizar los resultados del cálculo.

Es necesario realizar una validación del código, para poder verificar la biblioteca de secciones eficaces utilizada y el correcto uso de la herramienta por parte del usuario. Esta validación se puede observar en el Apéndice A.

1.4.2. KDSource

KDSource es una herramienta que utiliza algoritmos de KDE para el modelado de densidades de probabilidad conjuntas, subyacentes al espacio de fases asociado a una lista de partículas grabadas en una superficie en una simulación de Monte Carlo [6]. También permite muestrear nuevas partículas a partir de las estimaciones de distribuciones continuas realizadas previamente. De esta manera, se genera una gran cantidad de partículas en la misma superficie conservando las correlaciones en el espacio de fases. Se realizará un análisis más detallado sobre esta herramienta en el Capítulo 2.

1.4.3. McStas

McStas es una herramienta de código abierto, para simular instrumentos y experimentos de *scattering* de neutrones [7]. El código interpreta las configuraciones de componentes y parámetros que rigen la simulación, y lo compila en un programa en lenguaje C que realiza el cálculo de Monte Carlo de manera rápida y eficiente. Se utiliza generalmente para la simulación de haces de radiación, donde la geometría y materiales no son complejos.

1.5. Esquema de la tesis

Hasta el momento se presentó la motivación y los objetivos del trabajo, la metodología empleada y las principales herramientas utilizadas en el mismo. En el Capítulo 2 se explicará con mayor detalle la herramienta computacional KDSource que conforma

la base en la cual se enfoca este proyecto. Para eso se desarrollarán los conceptos de algoritmos de KDE y su utilización en la generación de fuentes de radiación. Además, se detallarán los componentes y estructuras que conforman esta herramienta y el flujo de trabajo que se debe llevar a cabo para poder utilizarla de manera satisfactoria. Para completar este capítulo, se explicará como se llevaron a cabo las optimizaciones que se realizaron al código para obtener mejoras en la herramienta.

Luego, en el Capítulo 3 se desarrollará el primer caso de estudio que es sobre el acelerador lineal LINAC. Este caso se utilizará para presentar los primeros resultados sobre las nuevas implementaciones agregadas en la herramienta. Se prestará principal atención a las distribuciones de las variables más pertinentes en cada caso. Para finalizar, se analizará el espectro de neutrones con datos experimentales relativos, lo cual permitirá concluir una primera validación de las optimizaciones de KDSource.

En el Capítulo 4 se explicará el segundo caso de estudio, el conducto de neutrografía del reactor RA6. En este caso se analizarán flujos, dosis y coeficientes de conversión a dosis utilizando nuevamente la herramienta KDSource. Los resultados obtenidos serán comparados con valores experimentales, lo cual servirá como segunda validación de la herramienta y de sus nuevas implementaciones.

Para finalizar, en el Capítulo 5 se detallarán las conclusiones obtenidas en el trabajo y las propuestas que se plantean como líneas de trabajo futuro.

Capítulo 2

KDSouce

Objetivos de la herramienta

KDSouce es una herramienta computacional pensada para el modelado de fuentes distribucionales. Fue desarrollada recientemente en un trabajo de maestría del Instituto Balseiro relacionado con el Departamento de Física de Reactores [6]. El paquete contiene una librería en Python, la cual permite analizar y optimizar la fuente, y una librería en C para generar nuevas partículas. Además, cuenta con una aplicación de línea de comando y un conjunto de archivos auxiliares que facilitan la utilización de las fuentes. Tiene como principales beneficios tanto la reducción de varianza en simulaciones Monte Carlo, como el acople entre distintos códigos utilizados comúnmente para este tipo de simulaciones. Utiliza algoritmos de KDE para el ajuste de distribuciones de partículas, conservando la correlación entre las distintas variables en el espacio de fases.

2.1. Algoritmos KDE

2.1.1. Kernel Density Estimation

Kernel Density Estimation (KDE) es una herramienta no paramétrica utilizada en estadística para estimar funciones de densidades de probabilidad desconocidas. Para realizar esto se utiliza un estimador que dependerá del ancho de banda y de la función *kernel* utilizada [8]. Otra metodología también utilizada es la de histogramas. KDE presenta ciertas ventajas por sobre los histogramas, como la independencia del ancho del *bin* y de su posición inicial. A su vez, KDE utiliza la posición exacta de la partícula sin la necesidad de agruparlas, lo cual resulta de gran interés para el análisis que se desea realizar.

Partiendo de un *set* dado de partículas, se busca entonces estimar la función den-

sidad de probabilidad multidimensional que mejor describa la distribución de estas partículas en una superficie. Para eso, se parte de la suposición de que existe una distribución $f(x_1, x_2, \dots, x_D)$ D-dimensional, la cual se buscará estimar. Para la física de neutrones, estas dimensiones serán las siete pertenecientes al espacio de fases: $\{x, y, z, u, v, w, E\}$ siendo $\{x, y, z\}$ las componentes espaciales, $\{u, v, w\}$ las direccionales y $\{E\}$ la energía de las partículas.

Se parte de un conjunto de N partículas $\{p_1, \dots, p_i, \dots, p_N\}$, donde la partícula p_i tiene información sobre cada una de las D dimensiones del espacio de fases. Es decir, $p_i = [(p_i)_1, \dots, (p_i)_j, \dots, (p_i)_D]$, siendo j una dimensión genérica.

Se realiza en primer lugar un reescalamiento de los datos. Para eso se divide a cada variable de cada partícula por la desviación estándar del conjunto de datos para esa variable. De esta manera se define \tilde{p}_i como:

$$\tilde{p}_i = [(\tilde{p}_i)_1, \dots, (\tilde{p}_i)_j, \dots, (\tilde{p}_i)_D] / \tilde{p}_{ij} = p_{ij}/\sigma_j \quad (2.1)$$

donde σ_j representa la desviación de la dimensión j teniendo en cuenta todos los datos del *set* de partículas original.

Este reescalamiento se realiza porque hay dimensiones, como puede ser la energía, que varían en una gran cantidad de órdenes de magnitud, por lo que se busca es que todas las variables tengan desviación unitaria.

A partir de este nuevo conjunto de datos, se utiliza un estimador multidimensional de la función densidad de probabilidad de las partículas. Éste se define como:

$$\tilde{f}(x_1, x_2, \dots, x_D) = \sum_{i=1}^N \frac{1}{N} \left\{ \prod_{j=1}^D K_h(x_j, (\tilde{p}_i)_j) \right\} \quad (2.2)$$

donde K_h representa la función *kernel* parametrizada por un ancho de banda h , ambas variables a definir en el método. En el caso multidimensional el ancho de banda h es una matriz diagonal D-dimensional, donde los elementos de la diagonal principal representan los anchos de banda unidimensionales de cada variable. Lo que se busca es que esta función se asimile lo mayor posible a la distribución desconocida, es decir:

$$\tilde{f} \approx f \quad (2.3)$$

Existen entonces dos elecciones que se pueden realizar para obtener distintos estimadores de mayor o menor precisión, estas son: la función *kernel* y el ancho de banda.

Una vez encontrado el estimador, se procede a realizar el muestreo de nuevas partículas que conservarán la correlación de las variables en el espacio de fases. Esto se da ya que las distintas dimensiones quedan correlacionadas entre sí en la función \tilde{f} .

2.1.2. KDE unidimensional

De manera simplificativa, en el caso unidimensional se tiene una lista de datos $\{p_1, \dots, p_i, \dots, p_N\}$ donde cada p_i tiene un único valor asociado [9]. En este caso el estimador para la determinación de la función densidad de probabilidad queda definido como:

$$\tilde{f}(x) = \sum_{i=1}^N \frac{1}{N} K_h(x, p_i) \quad (2.4)$$

Para ver explícitamente la dependencia del ancho de banda en la determinación del estimador, este se puede escribir como:

$$\tilde{f}(x) = \frac{1}{N \cdot h} \sum_{i=1}^N K\left(\frac{x - p_i}{h}\right) \quad (2.5)$$

donde h es el ancho de banda que, en este caso, es un escalar. Por lo tanto, el procedimiento unidimensional es el que se muestra en la Figura 2.1.

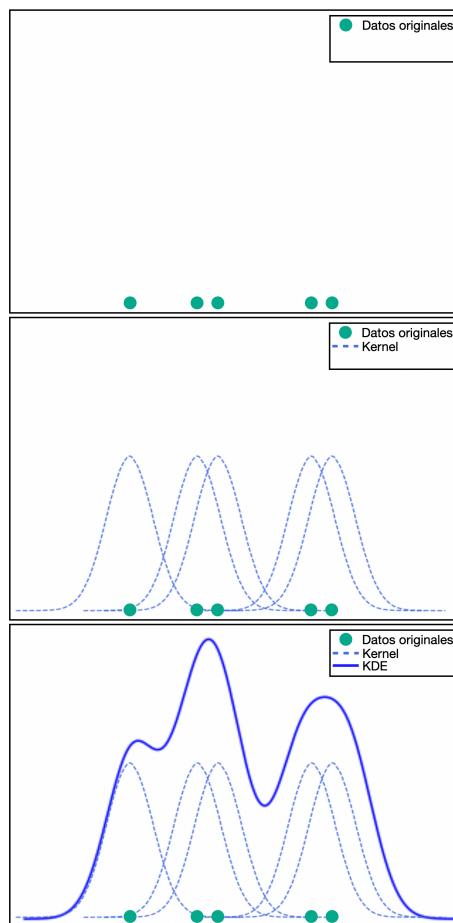


Figura 2.1: Procedimiento simplificado (unidimensional) para la implementación de algoritmos de KDE a un conjunto de datos.

En el primer gráfico se puede observar la distribución discreta de los datos de los que se parte para implementar la metodología. A cada punto se le asigna una función distribución de probabilidad dada (*kernel*), con cierto ancho de banda, la cual se observa en líneas punteadas en el segundo gráfico. Finalmente se hace la suma de las distribuciones para todas las partículas, obteniendo así en el último gráfico el estimador que pretende reproducir la distribución continua de partículas en esa dimensión. Este estimador debe normalizarse para representar dicha distribución.

2.1.3. Funciones kernel

Como se explicó previamente, la función *kernel* debe ser elegida convenientemente para poder representar correctamente los datos. Esta tiene que cumplir ciertos requerimientos:

- Debe ser definida positiva: $0 \leq K(x, p_i) \leq \infty$.
- Debe ser normalizada: $\int_{-\infty}^{\infty} K(x, p_i) dx = 1$.
- La media de la función debe ser p_i : $\int_{-\infty}^{\infty} x K(x, p_i) dx = p_i$
- La varianza de la función es función de h : $\int_{-\infty}^{\infty} x^2 K(x, p_i) dx = f(h) \neq 0$
- Usualmente se eligen funciones simétricas.

Dadas estas condiciones, se suelen usar ciertas funciones *kernel* que las satisfacen. Estas son:

- Gauss
- Epanechnikov
- Biweight
- Triangular
- Rectangular

Para el presente trabajo se tendrán en cuenta las dos primeras funciones. En el caso de la distribución Gaussiana, tiene la gráfica que se observa a la izquierda en la Figura 2.2. Su forma funcional es la siguiente:

$$K(x, p_i) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-p_i)^2}{2\sigma^2}} \quad (2.6)$$

Donde p_i representa en este caso la media, y σ la varianza. En este caso, la varianza es equivalente al ancho de banda ($\sigma = h$).

Este *kernel* es el más comúnmente utilizado en este tipo de metodologías, y es el que se encuentra implementado en la herramienta KDSource. Sin embargo, en este trabajo se estudiará la implementación de un *kernel* del tipo Epanechnikov, cuya gráfica se observa a la derecha en la Figura 2.2. Esta función tiene una forma parabólica:

$$K(x, p_i) = \frac{3}{4}(1 - (\frac{x - p_i}{h})^2) \quad (2.7)$$

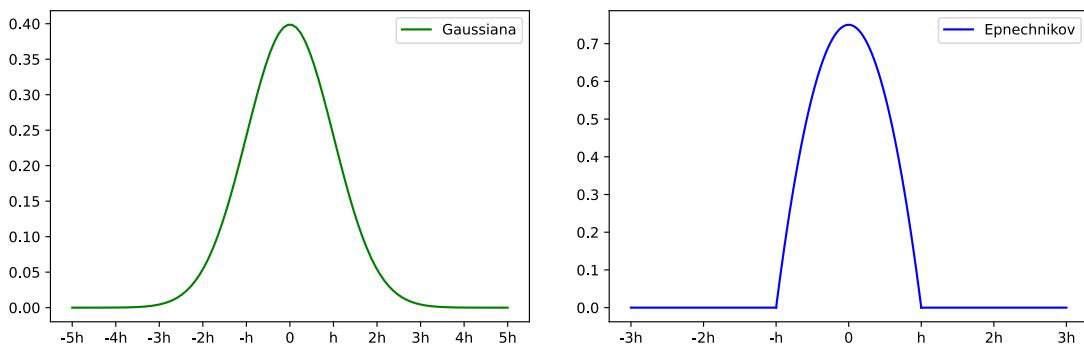


Figura 2.2: Comparación de las gráficas de las distribuciones gausiana y Epanechnikov con ancho de banda h .

La distribución de Epanechnikov presenta ciertas diferencias con la gaussiana. Lo más notable es que se anula en $\{-h, h\}$, mientras que la distribución normal solo se anula en el infinito. Esto es una desventaja para esta última ya que genera distribuciones continuas con bordes más suavizados, y esto provoca que a la hora del muestreo se generen partículas fuera de los bordes físicos del problema. En cambio, para el caso Epanechnikov los bordes se anulan más rápidamente. Esto ayuda a prevenir el suavizado del muestreo en los bordes.

2.1.4. Variación de ancho de banda

El ancho de banda, también llamado parámetro de suavizado, es un parámetro utilizado para dar forma a la función *kernel*. Se debe elegir un valor óptimo minimizando cierto error en la estimación de la función densidad de probabilidad. El impacto de la variación del ancho de banda se muestra en la Figura 2.3.

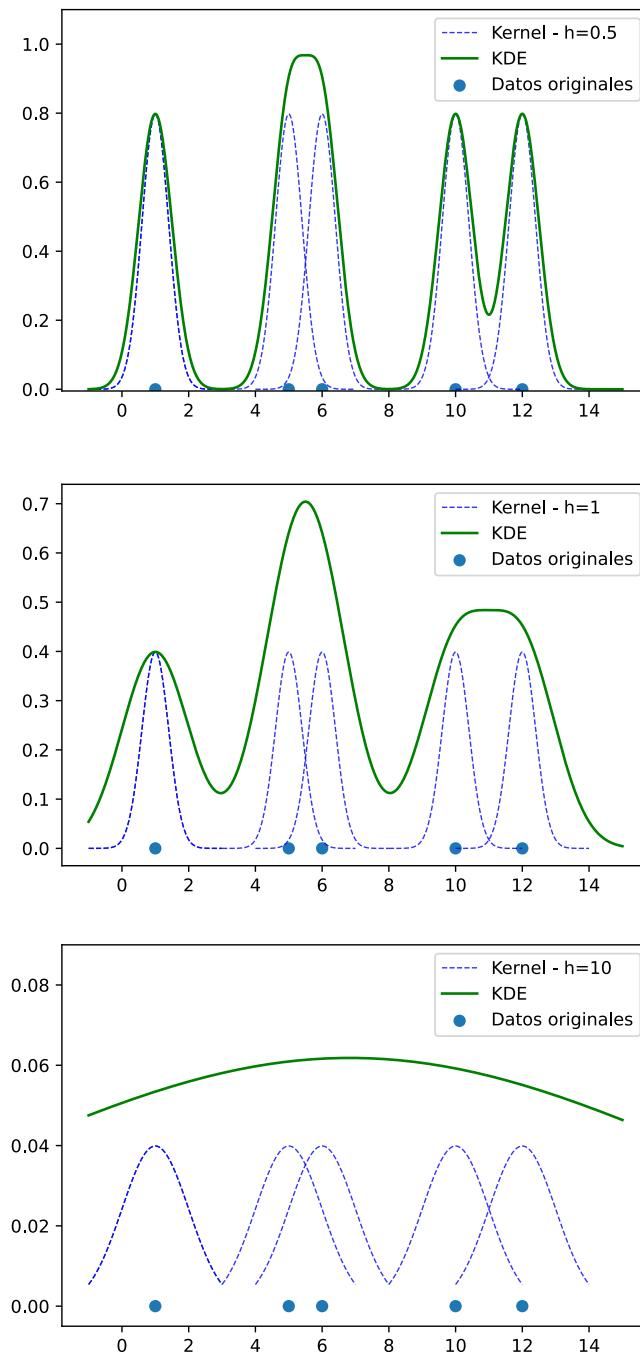


Figura 2.3: Comparación entre distintos anchos de banda para un mismo conjunto de datos aplicando KDE con *kernel* gaussiano.

En el primer gráfico se observa que la elección de un ancho de banda muy pequeño genera una distribución con muchas variaciones. Estas no reflejan la física del problema, sino que representan el ruido estadístico por estar trabajando con una cantidad finita de datos. Por el contrario, en el tercer gráfico se puede ver que si el ancho de banda es muy grande se tendrá una distribución muy suavizada. Esto implica que se pierdan detalles de las características de la distribución. Por lo tanto debe existir un ancho

de banda óptimo intermedio que genere una distribución más similar a la del segundo gráfico. De esta manera, se eliminará el ruido estadístico con una buena descripción de la física del problema presente en los datos.

2.2. Componentes y estructura de la herramienta

La herramienta KDSource cuenta con bibliotecas en Python y C, las cuales permiten modelar las fuentes distribucionales de partículas. En ambos casos se define la estructura **KDSouce**, que se compone de dos subestructuras **Plist** y **Geometry**. La primera se encarga de la lectura y escritura de los archivos MCPL, formato en el que se encuentran originalmente las partículas. Además, se pueden realizar traslaciones y rotaciones de los ejes de coordenadas luego de leerlas, para poder acoplar simulaciones con distintos sistemas de referencia.

La subestructura **Geometry** utiliza el objeto generado por **PList** y define la manera en la que se tratarán la geometría espacial, la dirección de las partículas y su energía. De esta manera se genera un vector de parametrización donde se encuentran las variables que se utilizarán para el método KDE. Existen distintas subestructuras del tipo **Metric**, que permiten realizar distintos tratamientos espaciales, direccionales y energéticos a las partículas. Las métricas implementadas son:

- **Energy**: tratamiento energético con la energía, variable parametrizada $[E]$.
- **Lethargy**: tratamiento energético con la letargía, variable parametrizada $[u]$.
- **Vol**: tratamiento espacial volumétrico, variables parametrizadas $[x, y, z]$.
- **SurfXY**: tratamiento espacial en plano XY, variables parametrizadas $[x, y]$.
- **Guide**: geometría de guía rectangular, variables parametrizadas $[z, t, \mu, \phi]$.
- **Isotrop**: tratamiento direccional con versor unitario, variables parametrizadas $[u_x, u_y, u_z]$.
- **Polar**: tratamiento direccional en coordenadas polares con versor unitario, variables parametrizadas $[\theta, \phi]$.
- **PolarMu**: tratamiento direccional en coordenadas polares con $\mu = \cos\theta$, variables parametrizadas $[\mu, \phi]$.

Como se explicó previamente, dado que la API de Python tiene como objetivo generar la fuente de KDE, existen funciones implementadas en estas métricas (**transform** e **inverse_transform**) que generan un vector de variables parametrizadas del tipo **numpy.array**. Esta API tiene también la capacidad de elejir el ancho de banda óptimo

para cada variable con distintas metodologías. Por otro lado, en la biblioteca de C se deben generar las nuevas partículas a partir del modelo de KDE, y para eso existen funciones que perturban las partículas existentes con el *kernel* y el ancho de banda provistos, respetando la métrica correspondiente.

Un caso de interés particular en este trabajo es el análisis del muestreo de partículas fuera de las condiciones de borde de las variables. Estas condiciones pueden darse por motivos matemáticos o por decisión del usuario dándole un sentido físico del problema. Esto es una dificultad en el método KDE, principalmente en el uso del *kernel* gaussiano, ya que no se encuentra limitado espacialmente. Se realiza es un filtro, mediante el cual se refleja a la partícula que fue perturbada por fuera de los límites permitidos de manera que no existan partículas incoherentes con el sistema. Se podría pensar que la utilización de otro *kernel* podría minimizar la cantidad de partículas que caen fuera de los bordes permitidos para cada variable. Esta implementación será desarrollada más adelante en este trabajo.

2.3. Flujo de trabajo

El flujo de trabajo para la utilización de KDSource como paso intermedio entre simulaciones Monte Carlo es como el que se muestra en la Figura 2.4. Se parte de una primera simulación, con la cual se va a obtener un archivo de *tracks* conteniendo las partículas que pasaron por la superficie ubicada en una zona intermedia entre la fuente original y el punto de interés. Este archivo es, en primer lugar, transformado al formato MCPL ya que es el tipo de archivos que es capaz de leer KDSource. Luego, se procede a utilizar la API de Python donde a partir del archivo de *tracks* se ajusta el modelo de KDE con cierto *kernel* y se obtienen los anchos de banda óptimos. Esto luego se guarda en un archivo XML que contiene los anchos de banda óptimos, el *path* al archivo MCPL original y otros parámetros de interés para el muestreo.

En una segunda etapa, se procede a leer con la API de C el archivo XML generado en Python. Esto permite reconstruir el modelo de KDE planteado previamente y usarlo para generar nuevas partículas a partir de perturbaciones. Una opción para el sampleo es generar un nuevo archivo MCPL con la nueva cantidad de partículas que se quiera obtener, que luego puede ser transformado en el formato que se desee para una nueva simulación Monte Carlo. Por otra parte, existe la posibilidad de generar partículas en lo que se denomina “*on-the-fly source*”. Esto permite realizar la nueva simulación de Monte Carlo con las partículas que se generan en KDSource, sin necesidad de guardar un nuevo archivo MCPL que podría resultar de gran tamaño. Esta metodología se encuentra actualmente implementada para los códigos McStas y TRIPOLI-4 ya que permiten ser compilados con archivos MCPL como input.

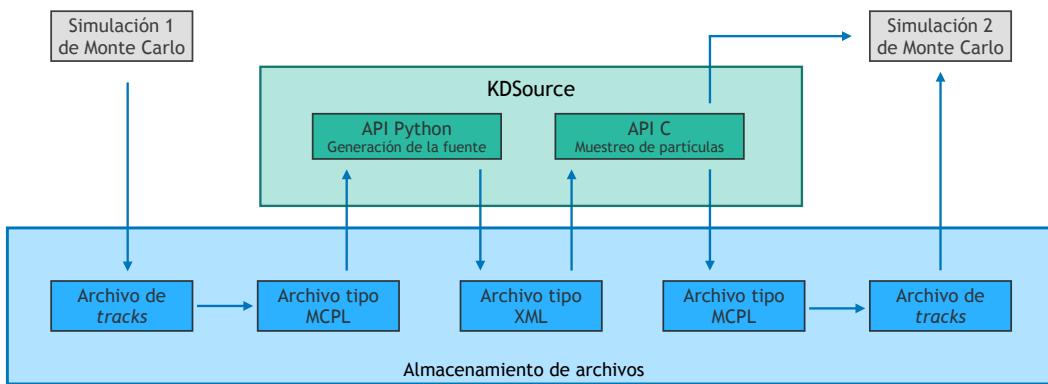


Figura 2.4: Flujo de trabajo de la herramienta KDSource, con su respectivo manejo de archivos.

2.4. Nuevas implementaciones

En base a recomendaciones realizadas en trabajos anteriores, surgen naturalmente dos nuevas implementaciones transversales a la herramienta que proveerán mejoras en los resultados de los cálculos para distintos problemas. En primer lugar, como se explicó previamente (en la Sección 2.2), resulta interesante estudiar el método de KDE con una función *kernel* alternativa a la gaussiana. De esta manera se agregó al código la posibilidad de utilizar la función de Epanechnikov, la cual generó cambios intrínsecos a la herramienta.

Dentro del código, tanto para el modelo de KDE en Python como para el posterior muestreo en C se utilizaba de manera implícita un *kernel* gaussiano. Para poder implementar la opción de Epanechnikov en la API de Python, se agregó una nueva variable `kernel` en la definición de la clase `KDSource`. Esto permite modelar el problema con las distintas distribuciones de probabilidad. Esta información es luego transferida a la API de C a través de un archivo XML, para poder generar nuevas partículas perturbadas según el *kernel* que haya sido elegido mediante una función llamada `rand_type`.

Otra implementación realizada en el código fue la adición de una métrica para poder parametrizar las variables espaciales en coordenadas cilíndricas. Para eso se definió una nueva clase `SurfCirc` que recibe las coordenadas espaciales $\{x, y, z\}$ y realiza la transformación de manera que las nuevas variables parametrizadas resultan ser $\{\rho, \psi, z\}$ calculadas según las Ecuaciones 2.8 y 2.9:

$$\rho = \sqrt{x^2 + y^2} \quad (2.8)$$

$$\psi = \arctan \frac{y}{x} \quad (2.9)$$

donde ρ es la distancia radial de la partícula desde el origen y ψ el ángulo tomado

desde el eje x positivo.

Esta última implementación resulta interesante porque es habitual contar con haces de radiación cuyas geometrías son cilíndricas y así los mismos se podrán representar con mayor exactitud.

Cabe destacar que estas implementaciones no fueron realizadas directamente en el código de la herramienta. Con el objetivo de visualizar si realmente había diferencias con el uso de estas modificaciones, se desarrollaron en primer lugar códigos independientes capaces de realizar esta metodología de manera satisfactoria. Luego de realizar esta verificación, se procedió a generar los cambios en la herramienta KDSource. Estos cambios requirieron un conocimiento exhaustivo del código y del manejo de las distintas clases y variables dentro del mismo. Los cambios realizados en el código fuente se pueden observar en el Apéndice C.

Capítulo 3

LINAC

3.1. Descripción general

Se llevaron a cabo simulaciones en base a experimentos de medición de espectro de neutrones realizados en una fuente de neutrones pulsados generados en un acelerador lineal de electrones (LINAC). Estos experimentos, descriptos con mayor detalle en [10], fueron llevados a cabo en el Centro Atómico Bariloche (CAB) y consisten en la medición de espectro de neutrones en agua liviana a 25°C utilizando el método de tiempo de vuelo. En la Figura 3.1 se puede observar la geometría de los dispositivos involucrados en el experimento.

Un haz de electrones de 25 MeV incide sobre un blanco liberando radiación por bremsstrahlung, lo cual genera neutrones rápidos a partir de una reacción fotonuclear. Estos neutrones monodireccionales son emitidos entonces por una fuente de geometría cilíndrica plana de 3 cm de radio, con un espectro de Watt:

$$S(E) dE = c e^{-E/a} \sinh \sqrt{b/E} dE \quad (3.1)$$

donde la energía E se encuentra en MeV, $a = 0,38$ MeV, $b = 7$ MeV $^{-1}$ y c es un parámetro de normalización constante de 1 n/s.

Estos neutrones inciden en un tanque cúbico de agua liviana de 25 mm de lado, rodeado de un blindaje de parafina borada. Este tiene un orificio semipasante en el centro, por el cual saldrán los neutrones moderados para luego dirigirse por un tubo de vuelo de 17 m de largo y ser detectados en el extremo.

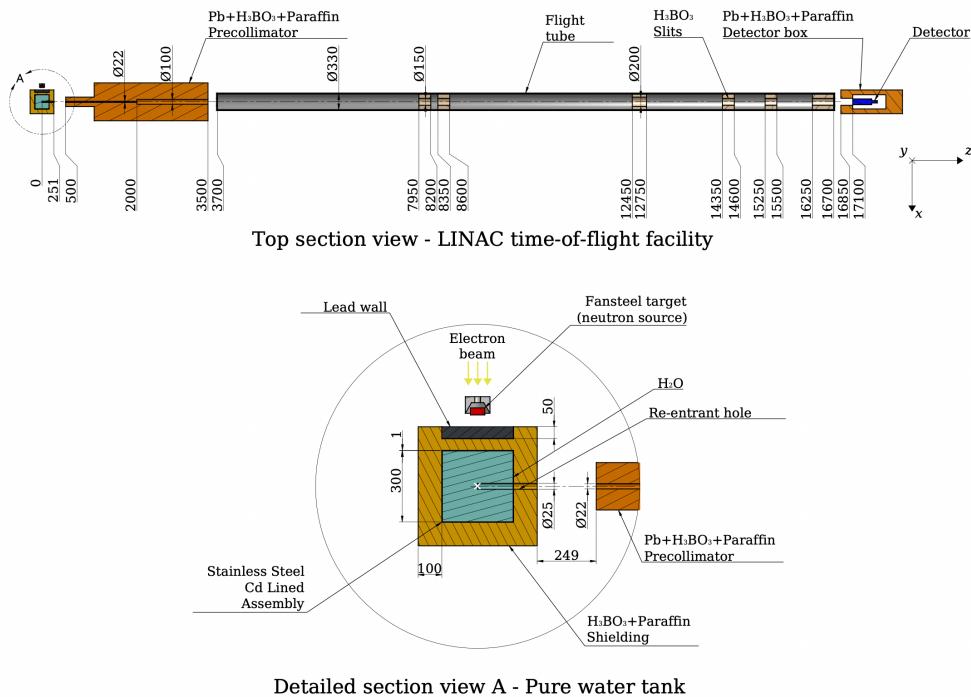


Figura 3.1: Esquema de la geometría utilizada para la medición del espectro de neutrones en agua liviana. En la parte superior se observa el dispositivo de tiempo de vuelo utilizado. En la parte inferior se ve con mayor detalle la fuente de neutrones, que inciden en el moderador por arriba y se extraen por el tubo reentrantante hacia la derecha para incidir en el tubo de 17 m de largo. Todas las unidades son en mm. Fuente: [1]

Se hizo una primera simulación en OpenMC, utilizando la biblioteca de datos nucleares ENDF/B-VII [5]. Las partículas que atravesaron la superficie del centro donde se encuentra el tubo reentrantante se guardaron en un archivo HDF5, y luego se filtraron aquellas con dirección saliente para ser finalmente almacenadas en un archivo MCPL. El tubo de tiempo de vuelo fue modelado en McStas. El origen de coordenadas tanto para OpenMC como para McStas se encuentra en el centro del tanque de agua (cruz blanca de la Figura 3.1), mismo lugar donde se registró la superficie de *tracks*. El eje *z* positivo se encuentra en dirección hacia el tubo de tiempo de vuelo.

En la primera simulación de Monte Carlo en OpenMC se registraron los flujos rápido y térmico dentro del tanque de agua. Esto se puede observar en la Figura 3.2.

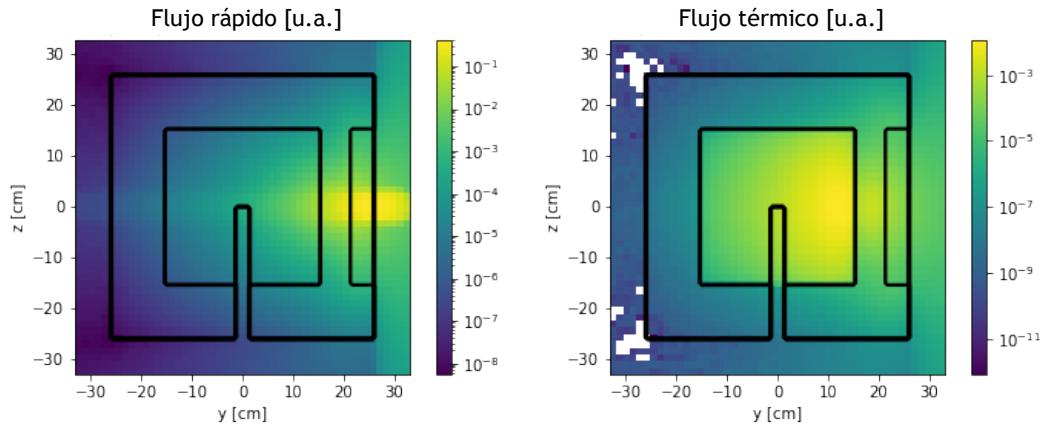


Figura 3.2: Flujo neutrónico rápido y térmico, obtenidos a partir de la simulación con OpenMC para el experimento del LINAC.

En la imagen de la izquierda se observa que el flujo rápido tiene mayor intensidad desde que sale de la fuente hasta que es moderado por el agua del tanque. Por otra parte, el flujo térmico tiene una intensidad mayor en todo el volumen del tanque de agua. De esta manera, se esperaría obtener en el tubo reentrantne neutrones moderados con el espectro característico del agua.

Además de observar los flujos neutrónicos, se registraron las partículas que atravesaron dos superficies de interés, las cuales se observan en la Figura 3.3.

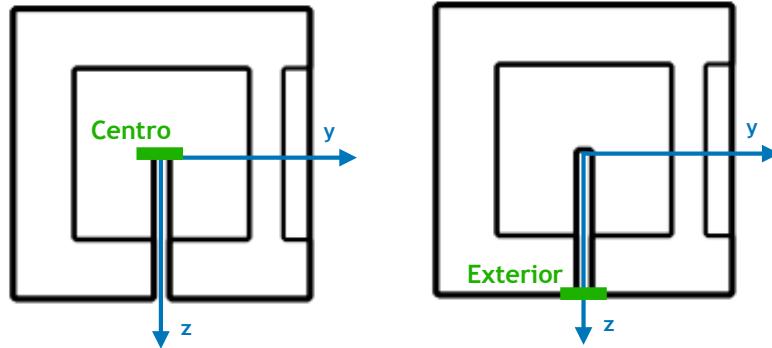


Figura 3.3: Superficies de *tracks* para la simulación del LINAC en OpenMC.

Los archivos de *tracks* tomados en estas superficies son los que se utilizaron para el análisis de datos posterior. En la Tabla 3.1 se ve la relación entre partículas de fuente y aquellas que se registraron en los archivos de *tracks* para ambas superficies, junto con el tiempo de simulación.

Fuente	Centro	Exterior	Tiempo
10^{10} part	10^7 part	$4 \cdot 10^4$ part	$\approx 23,2$ h

Tabla 3.1: Relación entre partículas simuladas de fuente, partículas en archivo de *tracks* y tiempo de simulación en OpenMC.

La relación entre las partículas que se grabaron en ambas superficies deja en evidencia que si se realizara una superficie de *tracks* en el detector que se encuentra a 17m se obtendría una cantidad mínima de partículas, las cuales no tendrían la estadística necesaria como para ser analizadas. Esto realza la motivación de generar una nueva fuente de radiación a la salida del tanque de agua.

A partir de la superficie de *tracks* registrada en el centro, se aplicaron los algoritmos de KDE para generar la fuente distribucional a utilizar en la segunda simulación. Para eso se utilizaron dos metodologías: la primera desarrollada de manera independiente y la otra utilizando la herramienta KDSource. Finalmente, se realizó la segunda simulación en McStas para obtener el espectro del agua liviana.

A continuación se analizarán los resultados obtenidos para las distintas etapas de cálculo. Para el próximo análisis se utilizó la superficie de *tracks* del centro.

3.2. Resultados

3.2.1. Elección de ancho de banda

En el desarrollo de una metodología independiente que utilice los algoritmos de KDE en la generación de una fuente distribucional de radiación, fue necesario analizar la elección de un ancho de banda BW acorde al problema que se está analizando. Para eso se analizaron las distribuciones de las distintas variables del espacio de fases con distintos valores de ancho de banda, las cuales se observan en las Figuras 3.4, 3.5 y 3.6. Se utilizó un kernel del tipo gaussiano.

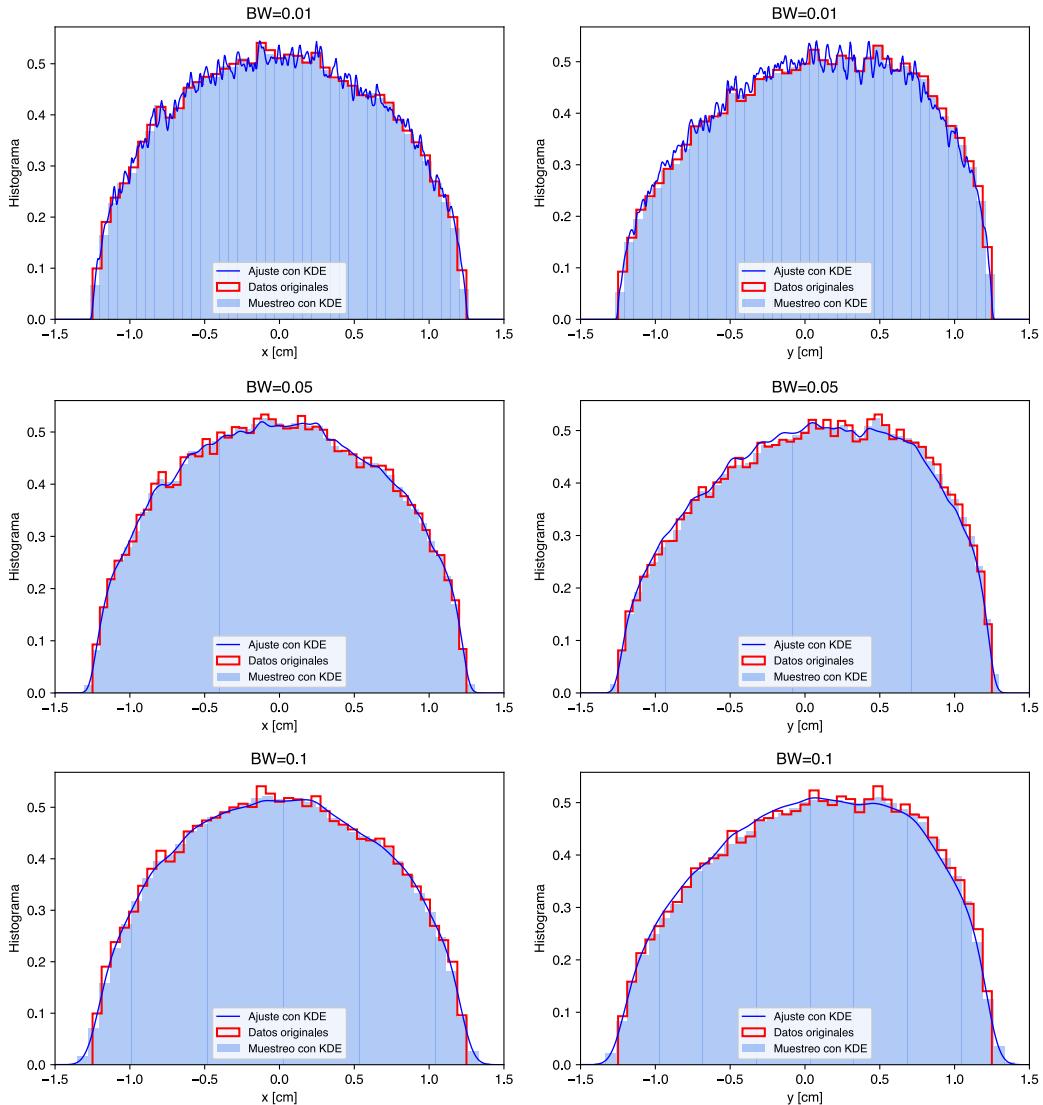


Figura 3.4: Distribuciones de coordenadas espaciales usando algoritmos de KDE con *kernel* gaussiano y variando el ancho de banda.

Utilizando un ancho de banda $BW = 0,01$ se observa que el ajuste de los datos con KDE presenta una gran componente de ruido. Este ruido es únicamente estadístico y no representa ningún comportamiento asociado a la física del problema. Por otra parte, si se aumenta mucho el ancho de banda se obtiene un ajuste que sobresuaviza la distribución en cada variable. El aumento de este suavizado también se observa en los bordes: al haber usado un *kernel* gaussiano se puede ver que hay una disminución gradual del ajuste que no coincide con los datos originales. Este efecto tampoco es deseable, y es por eso que se elige utilizar un valor intermedio de ancho de banda, de manera de optimizar la relación de compromiso que existe entre el ruido estadístico y el sobresuavizado. El ancho de banda elegido es entonces $BW = 0,05$, que se puede observar en la fila intermedia de las figuras.

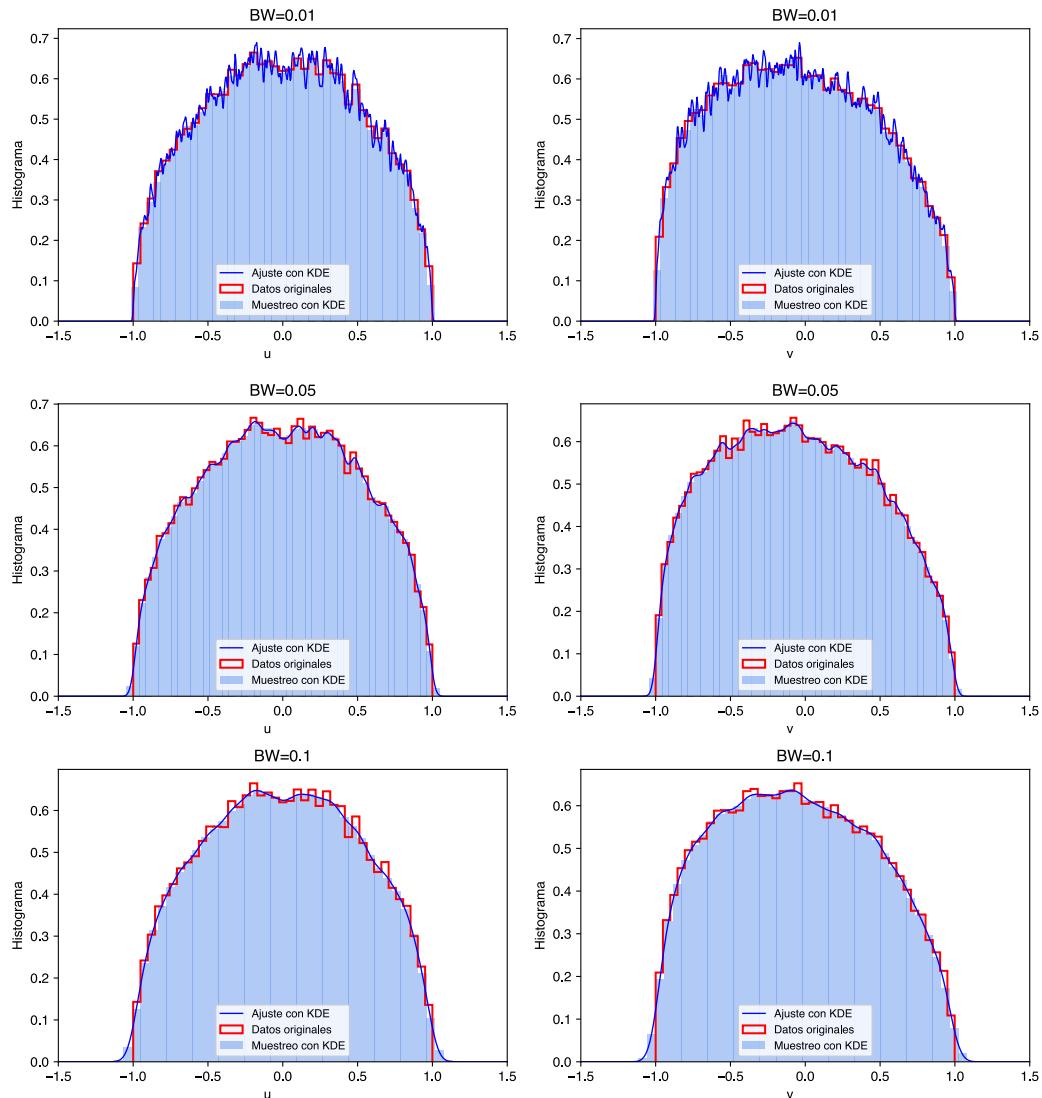


Figura 3.5: Distribuciones de coordenadas direccionales u y v usando algoritmos de KDE con *kernel* gaussiano y variando el ancho de banda.

En la Figura 3.6 se observa que la distribución en w es lineal. Esto se debe a que los neutrones en el seno del moderador presentan una distribución de flujo angular isotrópico, lo cual genera una corriente con distribución coseno, es decir, lineal en w .

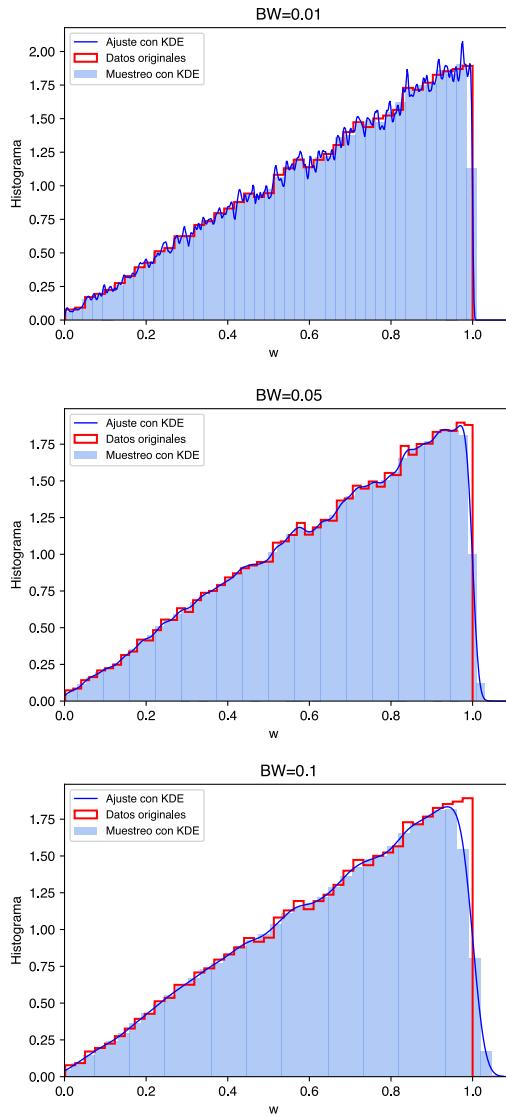


Figura 3.6: Distribuciones de variable w usando algoritmos de KDE con *kernel* gaussiano y variando el ancho de banda.

Esta elección de ancho de banda será utilizada para futuros ajustes y muestreos dentro de la metodología desarrollada independientemente. La herramienta KDSource ya cuenta con una implementación que elige el ancho de banda óptimo automáticamente para cada problema de distintas maneras, por lo que no será necesario hacer lo mismo que se realizó en este caso.

3.2.2. Variación de función *kernel*

Una vez elegido el ancho de banda, se buscó analizar de manera simplificada el impacto de la utilización de un nuevo *kernel* en los algoritmos de KDE. Para eso, se desarrolló un muestreo que siga una distribución del tipo Epanechnikov en la me-

dología independiente, y así observar los resultados preliminares de las distribuciones espaciales que se muestran en la Figura 3.7.

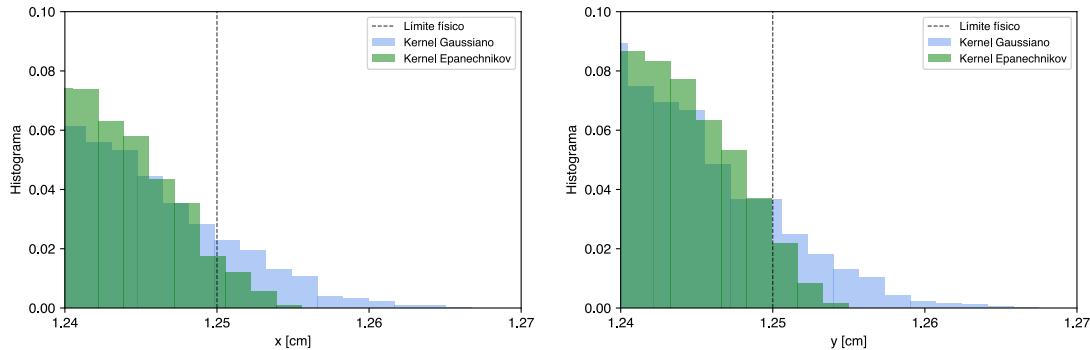


Figura 3.7: Comparación de distribuciones espaciales utilizando algoritmos de KDE con distintas funciones *kernel*.

Se puede observar que la distribución generada a partir de la utilización de *kernel* gaussiano se anula en valores mayores que en el caso Epanechnikov. Esto se debe al tipo de funciones que representan las distribuciones. Como se explicó previamente, el *kernel* gaussiano, al ser una distribución normal, se anula en el infinito. Esto hace que no haya límites discretos para las partículas muestreadas. Por otra parte, la función cuadrática que representa el *kernel* Epanechnikov sí se anula en un valor concreto. Esto hace que el muestreo de partículas sea mucho más preciso cerca de los bordes.

Este comportamiento puede solucionarse utilizando un filtro que no permita que se muestreen partículas fuera de los bordes del problema. Sin embargo, este no es el único problema que presenta la distribución normal. Otro fenómeno que se observa es la compensación de muestreo para valores cercanos al borde, del lado interno. Para el caso gaussiano se ve que hay una menor cantidad de partículas muestreadas en esta zona, que se debe nuevamente a la forma funcional de la distribución. Este efecto se agranda cuando se tratan distribuciones que no se anulan en los bordes, tal como la distribución en *w* que se muestra en la Figura 3.8.

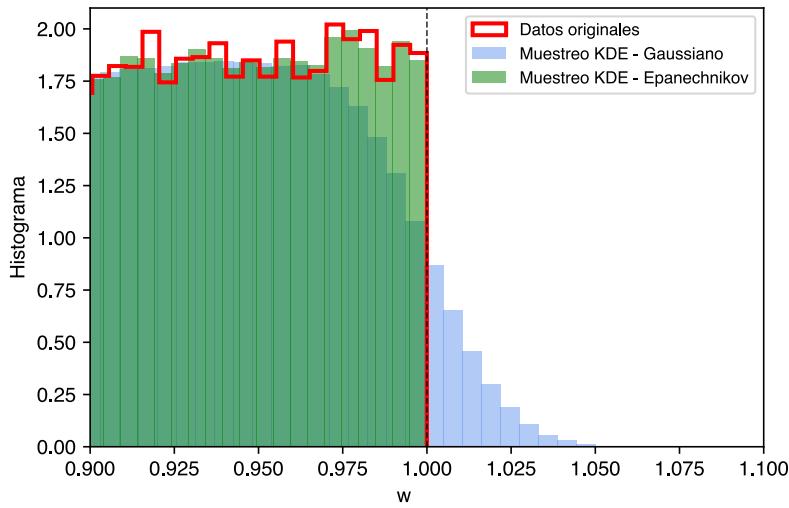


Figura 3.8: Comparación de distribución en w utilizando algoritmos de KDE con distintas funciones *kernel*.

Se puede ver que la suavidad presente en el *kernel* gaussiano resulta en una menor cantidad de partículas muestreadas cerca del borde. Este efecto no se observa con el *kernel* Epanechnikov, lo cual resulta muy beneficioso.

Habiendo analizado los resultados preliminares satisfactorios de las distribuciones utilizando distintos *kernel* en la metodología desarrollada independientemente, se consideró beneficioso implementar este mismo cambio en la herramienta KDSource. Para eso se desarrolló la opción de que el usuario elija el tipo de *kernel* que desea utilizar: gaussiano o Epanechnikov. La nueva implementación permite ajustar los datos y luego muestrear partículas utilizando la nueva distribución. Los resultados obtenidos para el análisis de distribuciones utilizando KDSource se observan en las Figuras 3.9 y 3.10.

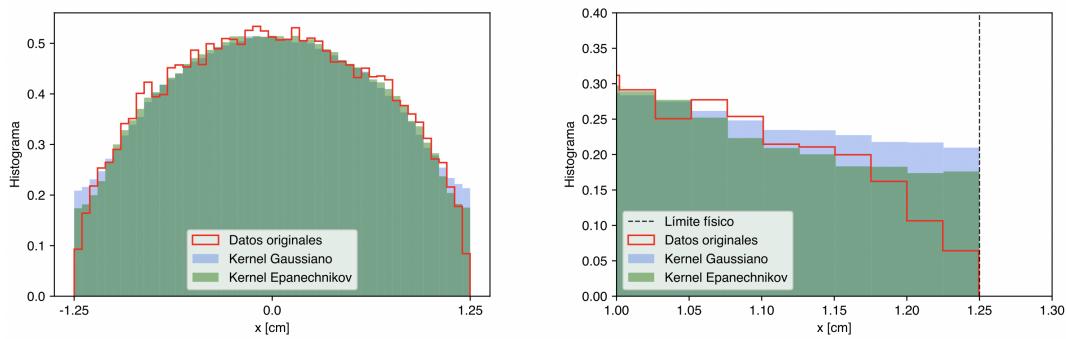


Figura 3.9: Comparación de distribución en x utilizando herramienta KDSource con distintas funciones *kernel*. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.

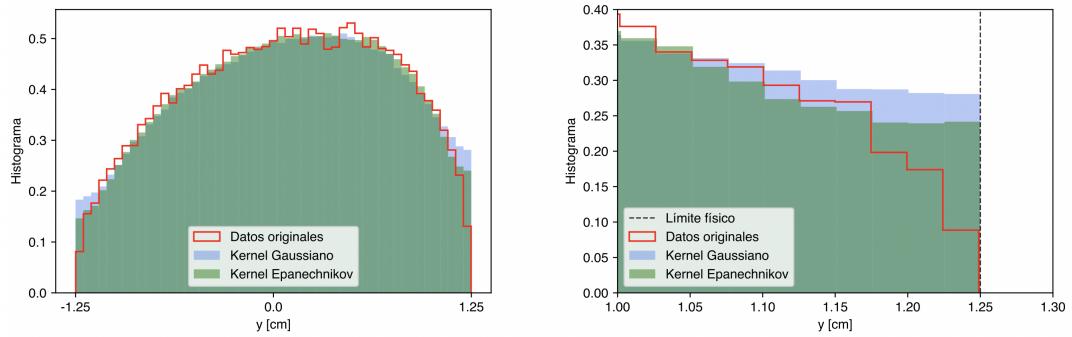


Figura 3.10: Comparación de distribución en y utilizando herramienta KDSource con distintas funciones *kernel*. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.

Se puede ver que ambas distribuciones espaciales coinciden lejos de los bordes. Sin embargo, al utilizar el *kernel* Epanechnikov se observa una mejora cerca de los límites de las distribuciones, tal como sucedía en el caso anterior. A pesar que la nueva distribución ajusta de mejor manera los datos, todavía se puede observar una diferencia con las partículas originales. Esta se buscará minimizar aún más con otra implementación en las variables espaciales, dada la geometría cilíndrica del problema.

3.2.3. Cambio de variables

Es común encontrar haces de radiación con geometría cilíndrica. Es por esto que se propuso implementar un cambio de variables espaciales que permitan aprovechar este tipo de geometrías al máximo. Nuevamente, se realizó esta modificación en la metodología independiente, para ver si realmente se presentan cambios al pasar de coordenadas $\{x, y\}$ a $\{\rho, \psi\} = \{\sqrt{x^2 + y^2}, \arctan(y/x)\}$.

En la Figura 3.11 se puede observar la distribución en términos del radio del orificio semipasante que se encuentra en el tanque de agua liviana en el LINAC. Se utilizó un *kernel* Epanechnikov para que las variaciones generadas por el gaussiano no contribuyan a las diferencias observadas por el cambio de variables.

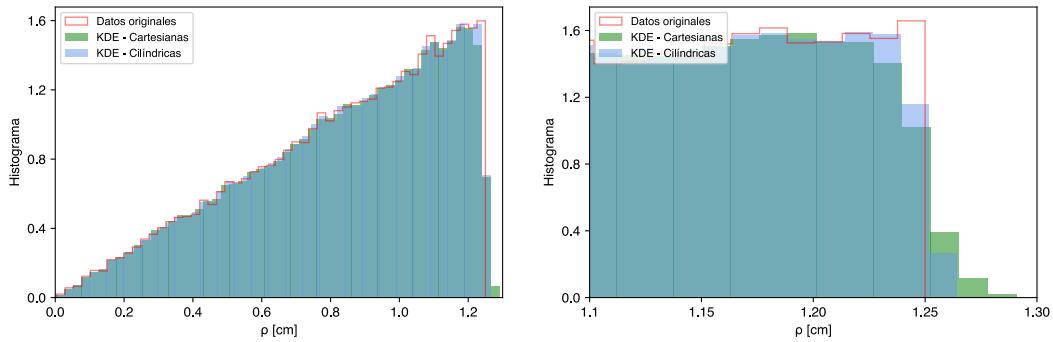


Figura 3.11: Distribución en el radio ρ utilizando distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.

Se puede observar en la imagen de la izquierda que, acorde a la teoría, esta distribución es lineal en $(0, R)$ con pendiente $m = 2/R^2$, siendo R el radio del orificio semipasante. Lejos del radio máximo no se observa una gran diferencia entre el muestreo realizado para ambos sistemas de coordenadas. Sin embargo, se ve en la imagen de la derecha que el muestreo en coordenadas cartesianas es mayor que aquel en coordenadas cilíndricas fuera del límite físico admisible del problema. Esta diferencia se da porque en el caso cartesiano el radio se calcula como:

$$\rho_{cart} = \sqrt{(x + \delta_x)^2 + (y + \delta_y)^2} \quad (3.2)$$

donde δ_x y δ_y son las perturbaciones realizadas en x e y respectivamente. Mientras que para el caso de coordenadas cilíndricas, el radio será:

$$\rho_{cil} = \sqrt{(x^2 + y^2)} + \delta_\rho \quad (3.3)$$

con δ_ρ la perturbación realizada en ρ . De esta manera, aumenta la probabilidad de encontrar mayores radios si se generan perturbaciones en las coordenadas $\{x, y\}$.

Esta diferencia es una ventaja que tiene el sistema de coordenadas cilíndricas por sobre el cartesiano. Consecuentemente, se decidió implementar en KDSource la posibilidad de realizar este cambio de variables. Esto se realizó generando un nuevo sistema de coordenadas llamado **SurfCirc**. De esta manera, el usuario podrá elegir las coordenadas espaciales que mejor se adapten al problema que quiere resolver. Los resultados para el LINAC de la utilización de **SurfCirc** se pueden observar en las Figuras 3.12 y 3.13. Nuevamente, se utilizó en este caso un *kernel* Epanechnikov.

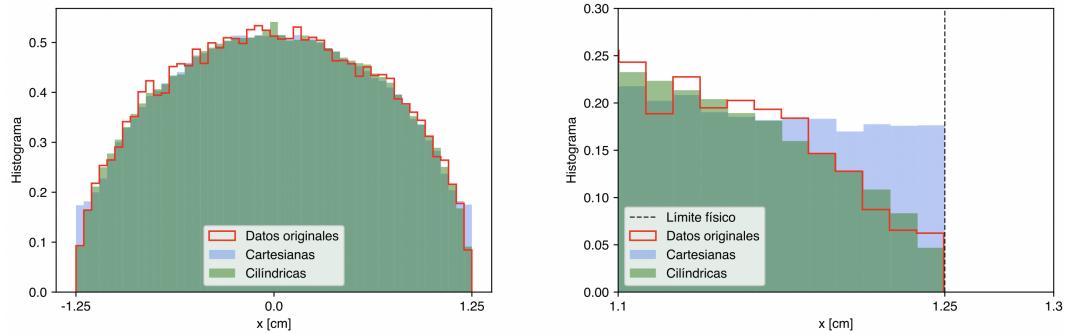


Figura 3.12: Comparación de distribución en x utilizando herramienta KDSource con distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.

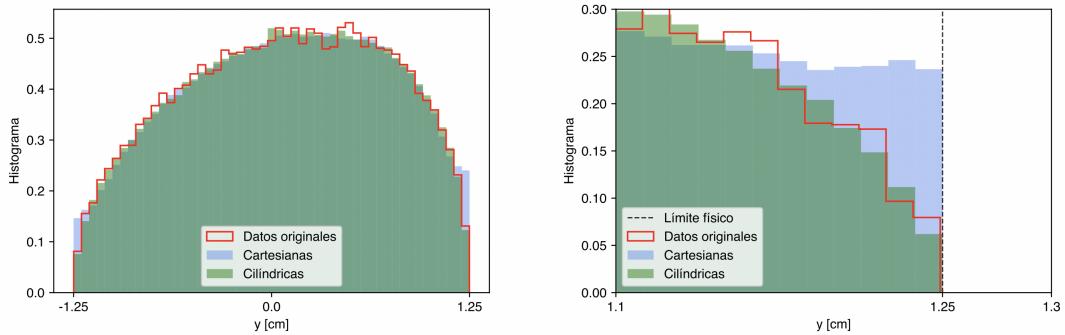


Figura 3.13: Comparación de distribución en y utilizando herramienta KDSource con distintos sistemas de coordenadas. La imagen de la derecha se enfoca en el comportamiento cercano al borde de la distribución.

Se ve que las distribuciones espaciales se ajustan con mayor precisión si se utilizan coordenadas cilíndricas.

3.2.4. Espectro

Como se explicó previamente, los experimentos realizados en el LINAC resultaron en la medición del espectro de neutrones en el agua liviana [10]. En primer lugar, utilizando KDSource se generó una fuente en la superficie externa del orificio semipasante del tanque de agua liviana. Esto se realizó tanto con la versión original de la herramienta computacional, como con sus nuevas implementaciones: *kernel* Epanechnikov y coordenadas espaciales cilíndricas. Los espectros resultante de estas fuentes se observan en las Figuras 3.14 y 3.15 respectivamente.

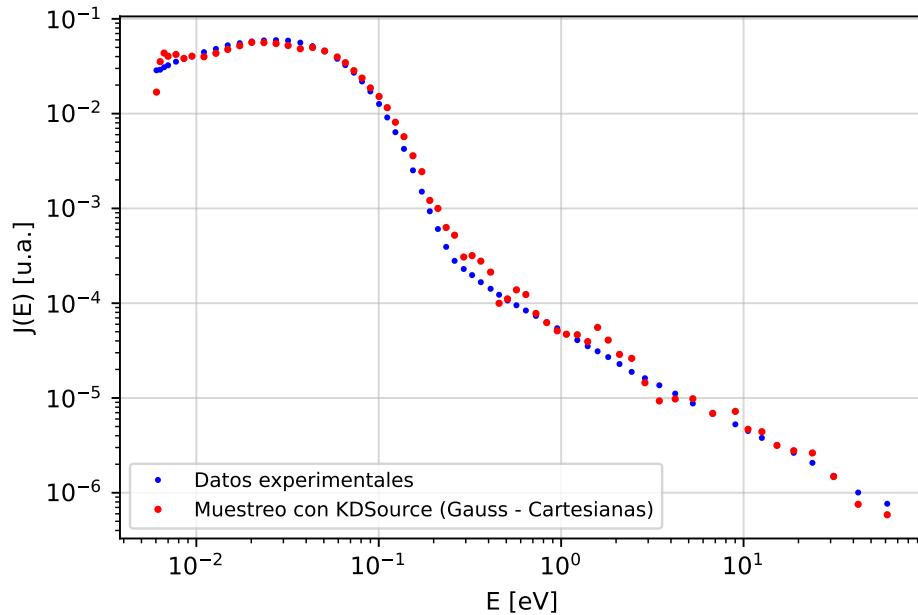


Figura 3.14: Espectro neutrónico de la fuente generada con la herramienta KDSource, en un extremo del tubo de vuelo del LINAC.

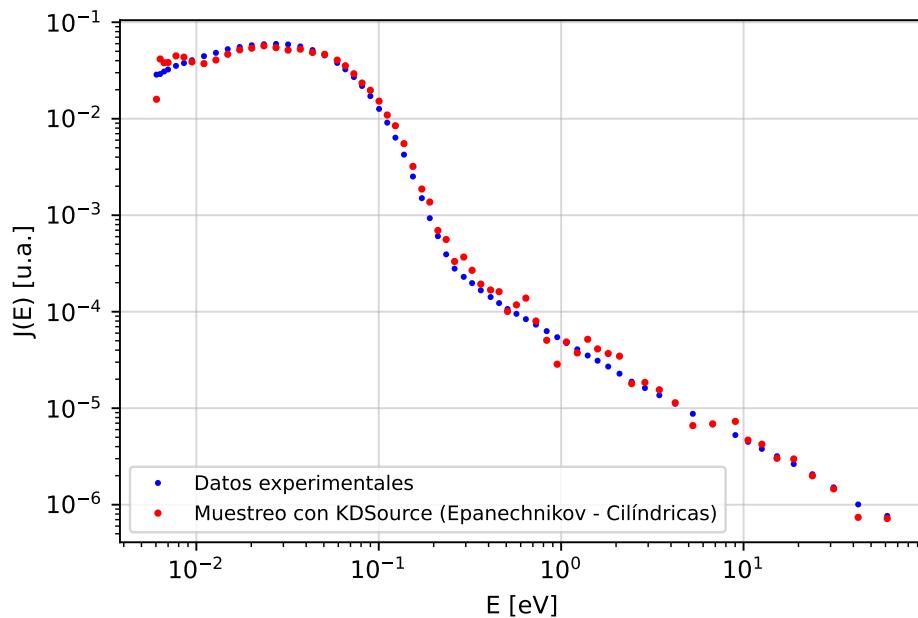


Figura 3.15: Espectro neutrónico de la fuente generada con la herramienta KDSource, en un extremo del tubo de vuelo del LINAC.

Se puede ver que en ambos casos el espectro neutrónico generado mediante algoritmos KDE se aproxima con gran exactitud a los datos experimentales en la mayor parte del rango energético. Sin embargo, estos resultados se generaron a la salida del tanque de agua, sin tener en cuenta el tubo de tiempo de vuelo de 17 m restante para llegar

al detector. Es por esto que, con el objetivo de analizar los resultados computacionales equivalentes a aquellos experimentales, se realizó una simulación del tubo de tiempo de vuelo en McStas, utilizando como fuente la generada por KDSource con $N_{part} = 10^{10}$ partículas. El espectro obtenido en el detector se puede observar en la Figura 3.16.

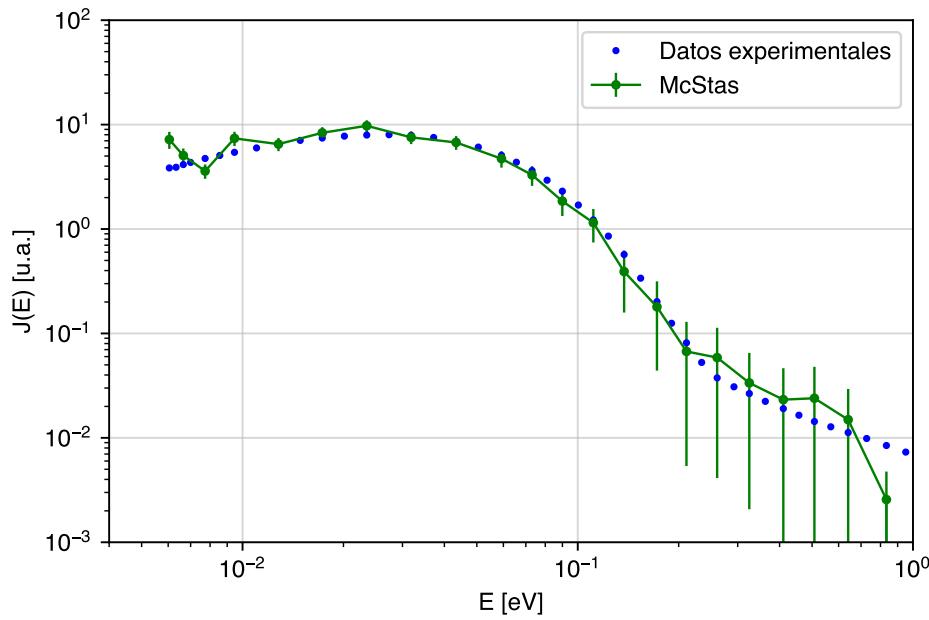


Figura 3.16: Espectro neutrónico obtenido en el extremo del tubo de vuelo del LINAC, simulado con McStas y utilizando la fuente generada en KDSource.

El espectro de neutrones de McStas se normalizó a 1 neutrón térmico para que sea comparable con los datos experimentales. En un primer análisis, cabe destacar que el flujo térmico simulado coincide en gran parte con el espectro medido. Sin embargo, al aumentar la energía se observa que tanto la exactitud como la precisión de la simulación disminuyen.

Capítulo 4

Haz de Neutrografía RA-6

4.1. Descripción general

En el presente capítulo, se continuará con la verificación de la herramienta KDSource, junto con sus nuevas implementaciones. Para eso, se utilizó como caso de estudio el dispositivo experimental de neutrografía, instalado en el conducto de radiación N1 del reactor RA-6 [11]. Para realizar esta validación, se utilizó un modelado en OpenMC realizado en un trabajo anterior [12]. En el esquema de la Figura 4.1 se pueden observar las distintas componentes del dispositivo.

El dispositivo consta de un conducto pasante que va desde la piletita del núcleo hasta un recinto donde se colocan las muestras que serán irradiadas. En su trayecto pasa por dos cámaras: una interna y otra externa. La primera se encuentra a la salida de la piletita. Su función es colimar los neutrones que ingresan al conducto, seleccionando aquellos que tengan posiciones y direcciones favorables. También posee un filtro de Zarifo para atenuar la componente de radiación γ y neutrones rápidos. Por otra parte, la cámara externa se utiliza para darle la forma definitiva al haz. Externo al colimador, se tiene un *shutter* móvil que se utiliza para blindar el haz cuando el dispositivo no se está utilizando, colocándolo frente al conducto. Como blindaje biológico del reactor, se tiene una pared de concreto en forma de octágono que rodea la piletita del mismo. Por otra parte, se tiene el blindaje principal con un recinto interno. Este es el lugar donde se alojan las muestras que se utilizarán para hacer neutrografía. Finalmente, se encuentra un *beam catcher* que se utiliza para blindar el haz de radiación que está saliendo del núcleo del reactor.

La fuente inicial utilizada es la que se denomina como superficie de *tracks*. Esta surge de un archivo que contiene partículas generadas a partir de una primera simulación del núcleo. Esta simulación inicial fue realizada con una cantidad total de partículas $N_{nuc} = 10^9$ y el núcleo en configuración 16 a 1 MW de potencia.

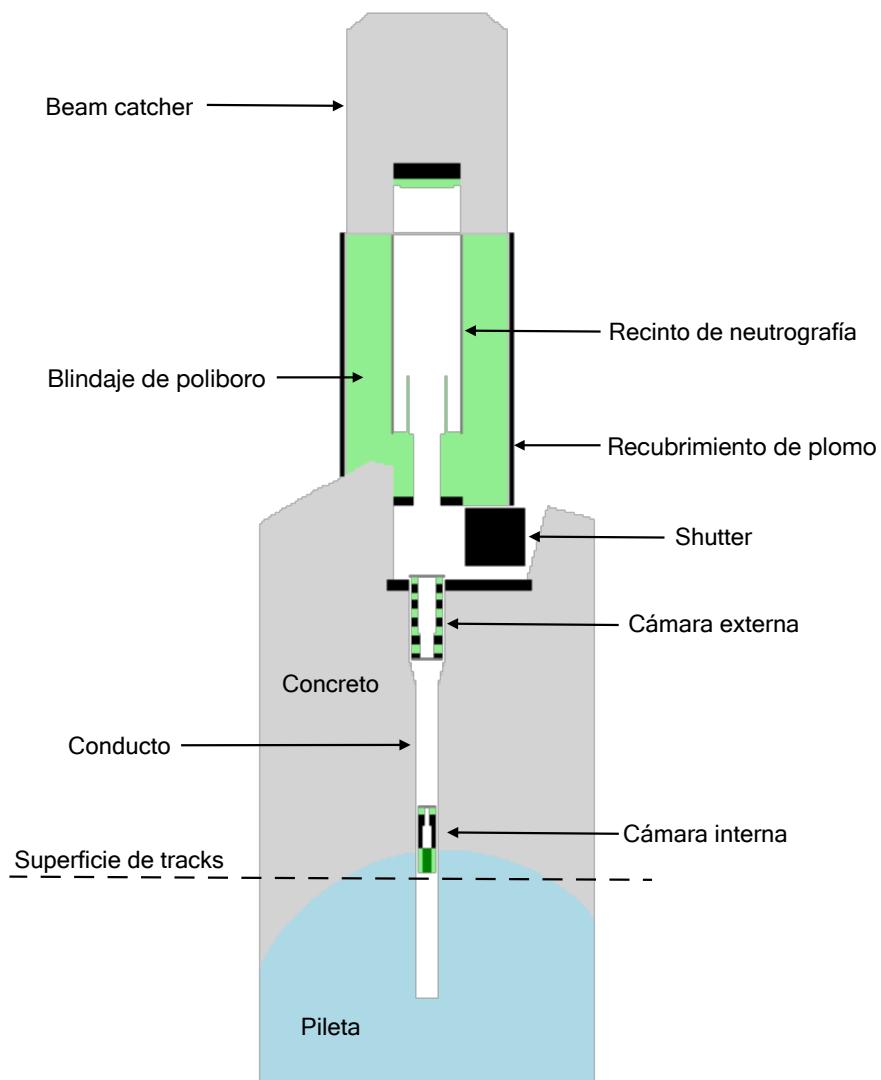


Figura 4.1: Esquema del dispositivo del neutrografía perteneciente al conducto N1 del reactor RA-6, modelada en OpenMC en vista superior

4.2. Simulaciones utilizando KDSource

Para obtener resultados sobre la tasa de dosis en distintos puntos de interés en el dispositivo de neutrografía, se dividieron las simulaciones en distintas etapas utilizando también KDSource en el proceso. En la Figura 4.2 se observan las geometrías y superficies utilizadas en estas etapas.

En primer lugar, se utilizó KDSource para generar una nueva fuente en la misma superficie $S1$ donde se encontraban las partículas originales. Es decir, a partir la superficie de *tracks* se produjo una nueva fuente con $N_{kds1} = 10^8$ neutrones y fotones. Estas partículas fueron simuladas en OpenMC con el modelo propuesto en la imagen de la izquierda de la Figura 4.2, guardando aquellas partículas que pasen por una nueva superficie $S2$, más próxima al extremo del haz. Con estas nuevas partículas se volvieron a

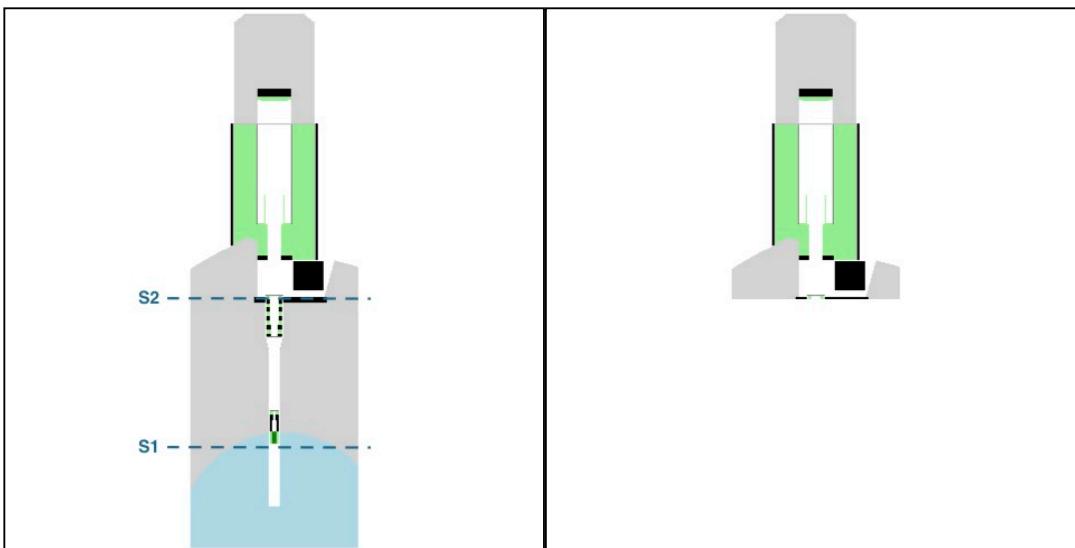


Figura 4.2: Geometrías del haz de neutrografía utilizadas en las simulaciones en OpenMC.

utilizar los algoritmos de KDE para obtener una nueva fuente con $N_{kds2} = 10^8$ neutrones y fotones. Finalmente, con la geometría de la imagen de la derecha, se realizó una simulación en OpenMC guardando los *tallies* necesarios en las afueras del dispositivo. Un esquema simplificativo de estas simulaciones y la generación de las distintas fuentes se observa en la Figura 4.3.

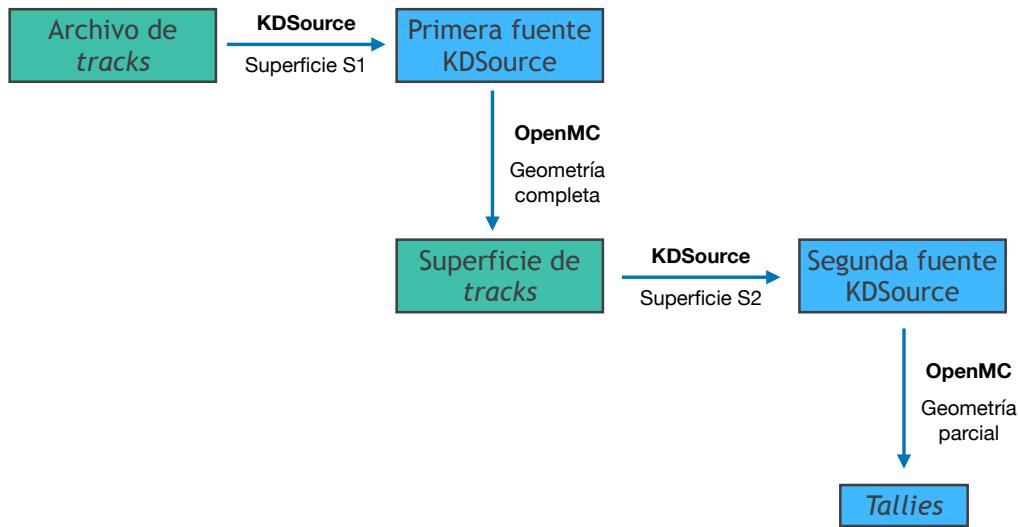


Figura 4.3: Esquema de trabajo realizado para el haz de neutrografía, utilizando OpenMC y KDSource.

En este caso, se contaba con un archivo de *tracks* inicial que contenía neutrones y fotones. Lo que se realizó fue tratar de manera independiente las fuentes de neutrones y fotones en la generación de fuentes de radiación con KDSource ya que se querían utilizar

métricas distintas para neutrones y fotones. Es por eso que se realizaron distintas simulaciones, conservando la relación entre las partículas con una debida normalización de los datos. El tratamiento del flujo de partículas en las distintas simulaciones puede observarse en la Figura 4.4.

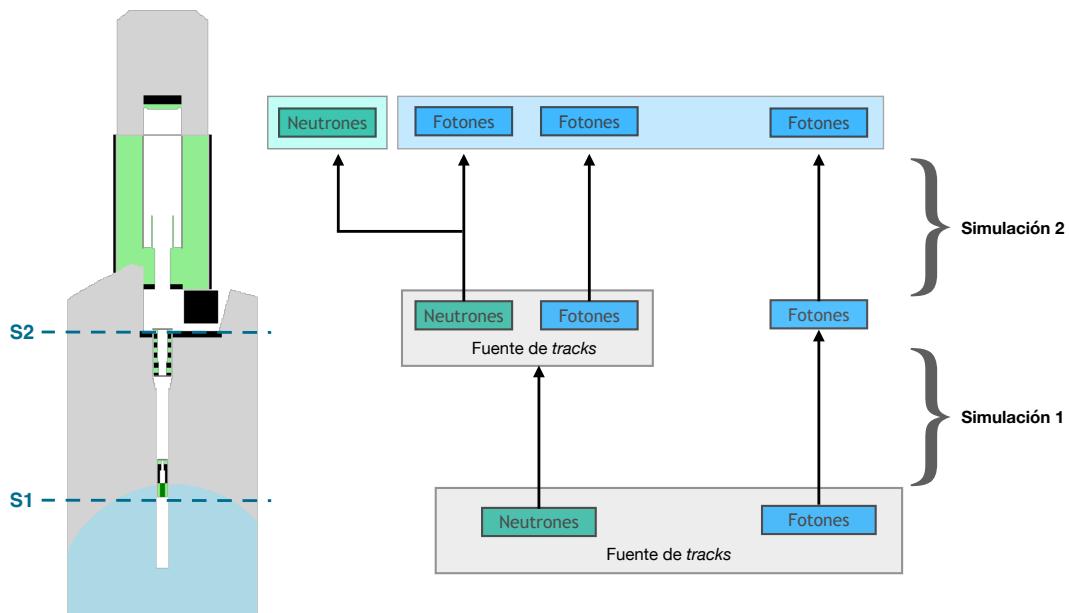


Figura 4.4: Esquema del flujo de partículas simuladas para el haz de neutrografía, utilizando OpenMC y KDSource.

4.2.1. Flujos

Para poder llegar a un valor de flujo es necesario calcular los factores de normalización. La tasa de neutrones que atraviesan una superficie se puede expresar como:

$$\frac{dN}{dt} = S_0 \cdot \frac{\sum_{i=1}^{N_{sup}} w_i}{N_{fuente}} \quad (4.1)$$

donde $S_0 = 7,73 \cdot 10^{16}$ n/s es el factor de fuente de la primera simulación del núcleo, w_i es el peso del i -ésimo neutrón que atraviesa la superficie, N_{sup} es la cantidad de neutrones en la superficie de estudio y N_{fuente} la cantidad de neutrones utilizados como fuente. De esta manera, se puede extender esta ecuación para calcular la tasa de neutrones que atraviesan una celda en el *tally* de flujo, resultando en la siguiente expresión:

$$\frac{dN_{tally}}{dt} = S_0 \cdot \frac{\sum_{i=1}^{N_{S1}} w_i}{N_{nuc}} \cdot \frac{\sum_{j=1}^{N_{S2}} w_j}{N_{kds1}} \quad (4.2)$$

donde N_{S1} son los neutrones de la superficie de *tracks*, N_{nuc} los neutrones utilizados como fuente en la primera simulación, N_{S2} los que fueron registrados en la superficie $S2$ y N_{kds1} aquellos que se generaron mediante KDSource en la superficie $S1$. Esta

normalización se extiende de forma análoga a los fotones. En la Figura 4.5 se pueden observar los flujos de neutrones y fotones obtenidos en la Simulación 1 realizada con los neutrones del archivo de *tracks* original, a partir del análisis del *tally* correspondiente. Se utilizó KDSource con las nuevas modificaciones realizadas: función *kernel* Epanechnikov y coordenadas espaciales cilíndricas.

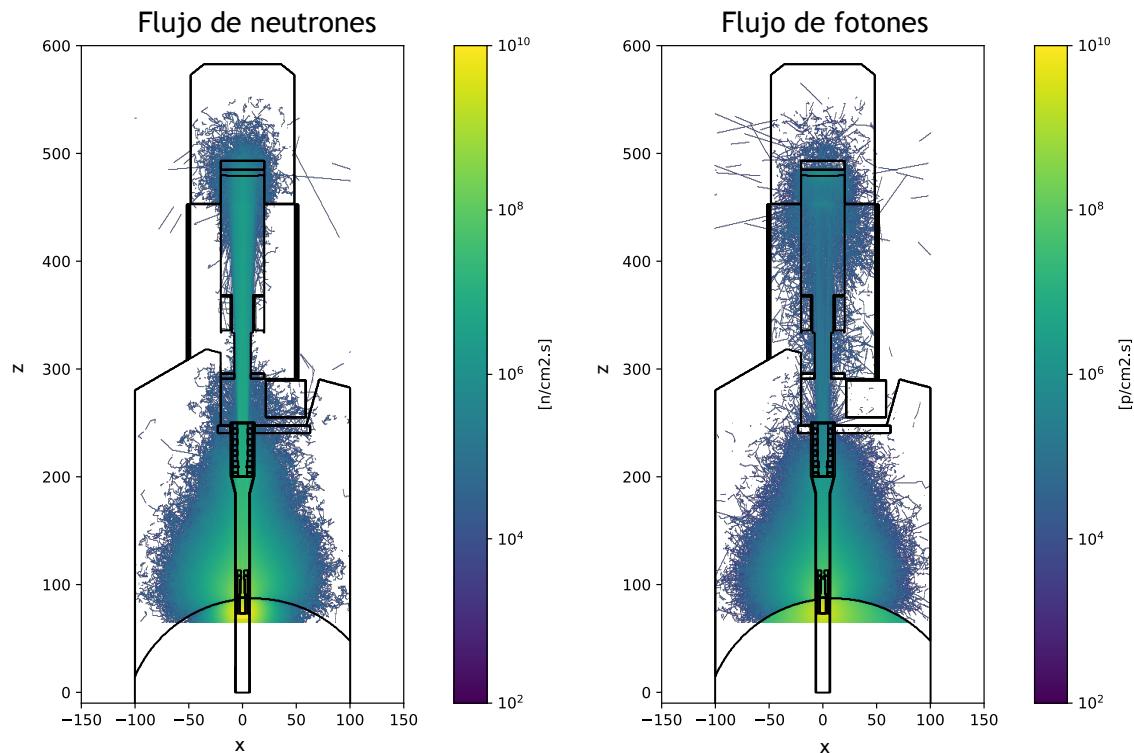


Figura 4.5: Flujos de neutrones y fotones obtenidos en la primera simulación de OpenMC con fuente de neutrones en la superficie S1.

Por otra parte, el flujo de fotones obtenido para la Simulación 1 a partir de los fotones de la fuente de *tracks* se observa en la Figura 4.6.

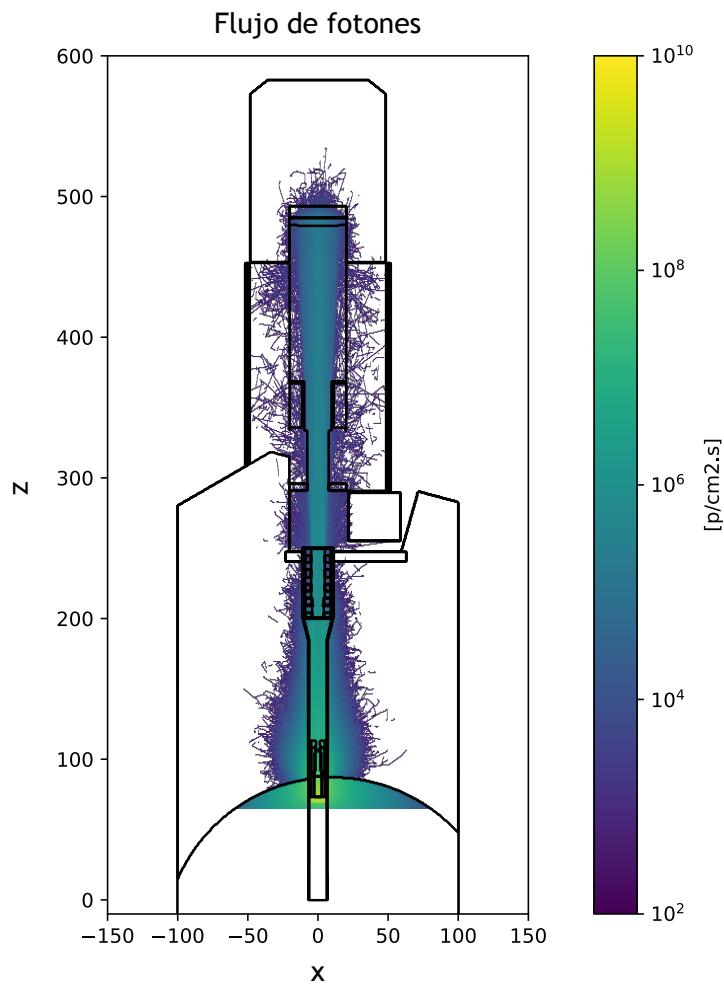


Figura 4.6: Flujo de fotones obtenido en la primera simulación de OpenMC con fuente de fotones en la superficie *S1*.

Se puede ver en ambas las Figuras 4.5 y 4.6 que sería imposible determinar un valor de flujo en las afueras del blindaje debido a la escasez de partículas presentes. Sin embargo, se ve que en la superficie *S2* dentro del conducto sí se pueden obtener datos estadísticos sobre las distribuciones de las partículas. A partir de estos resultados, se continuó con la metodología propuesta.

A continuación se observan los flujos de neutrones y fotones obtenidos luego de la Simulación 2 en la Figura 4.7. Cabe destacar que el flujo de fotones en este caso es la suma de los flujos normalizados generados por las distintas fuentes.

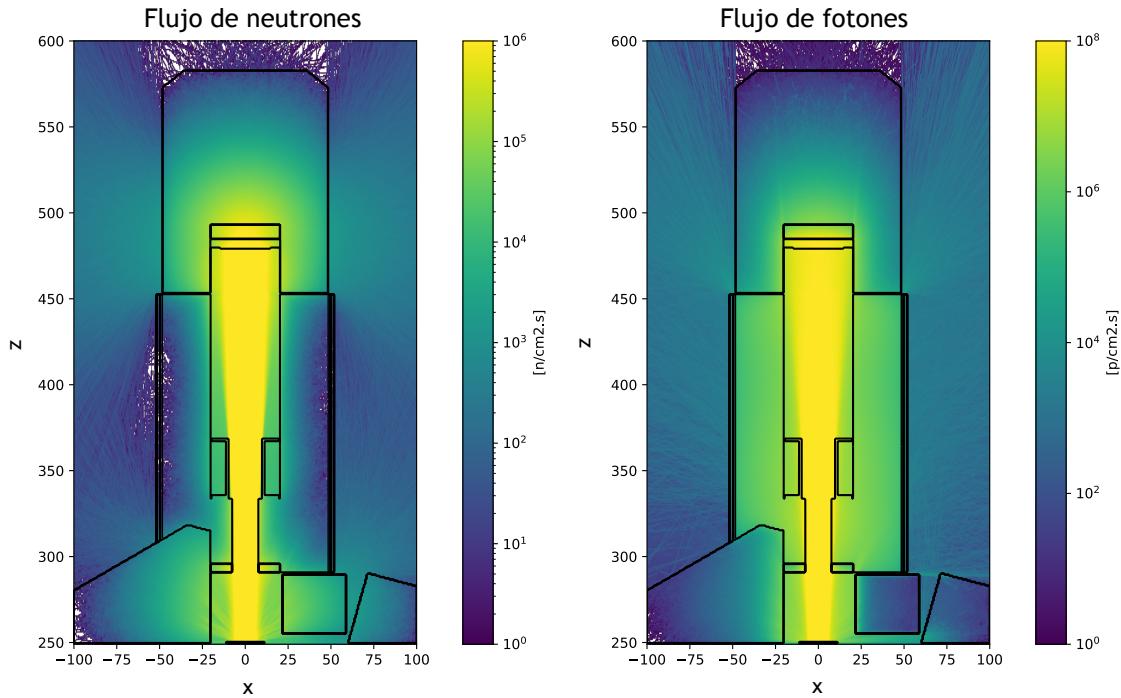


Figura 4.7: Flujos de neutrones y fotones obtenido en segunda simulación de OpenMC con fuentes en la superficie $S2$.

Se puede observar el aumento en la estadística de partículas en el área de interés para la segunda simulación, provocado por la proximidad de la fuente en este último caso. Esto permite visualizar la atenuación generada por el recubrimiento de plomo. Por otra parte, en ambos gráficos se observan zonas en blanco para grandes valores de z , lo cual evidencia la ausencia de partículas en estas zonas. Esto es por la presencia del *beam catcher*, el cual requiere que para obtener valores confiables de flujo en esa zona se simulen aún más partículas. Sin embargo, se esperaría que los valores tanto de flujo como posteriormente de dosis no sean muy elevados en esas zonas, ya que por diseño la tasa de dosis no debería superar la máxima tasa de dosis ambiental admitida para operarios.

En la imagen de la izquierda de la Figura 4.7 se puede ver que en las proximidades del exterior del recubrimiento de plomo también hay poca estadística sobre los valores de flujo de neutrones, asociada a los espacios en blanco por ausencia de partículas. Esto es por la presencia de políboro como blindaje de este tipo de partículas. Sin embargo, nuevamente se espera que los valores de flujo en esas zonas no sean muy grandes.

4.2.2. Tasa de dosis ambiental

Para el cálculo de dosis ambiental se utilizó un *tally* de flujo, junto con una función nativa de OpenMC que permite integrar el resultado de la multiplicación entre el flujo escalar y factores de conversión de fluencia a dosis equivalente ambiental, $H^*(10)$. Estos

últimos se pueden encontrar en la guía AR1 de la ARN [13]. Además, la grilla utilizada contiene celdas cuyos volúmenes corresponden a los de los monitores de las partículas que se deseen analizar en cada caso. Esto permite obtener la tasa de dosis ambiental para neutrones y fotones, las cuales se observan en la Figura 4.8. La dosis de fotones corresponde a la suma de las dosis de las distintas simulaciones realizadas con las fuentes que aportan este tipo de partículas.

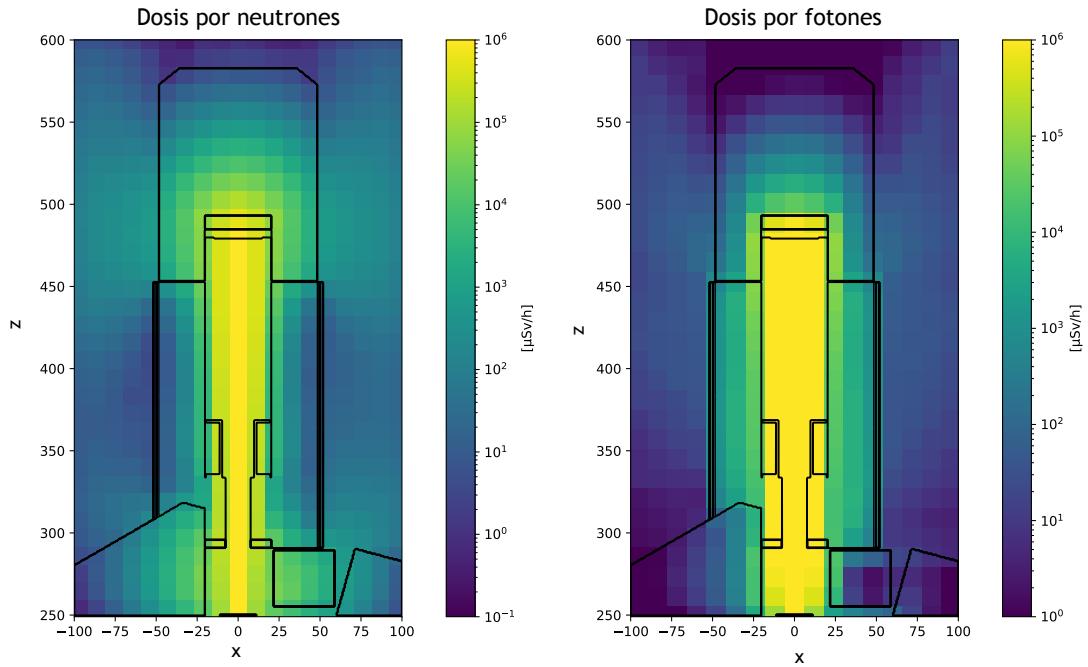


Figura 4.8: Tasa de dosis ambiental aportada por neutrones y fotones, obtenida a partir de la simulación de OpenMC con fuente en la superficie S2.

A primera vista, se puede observar que los órdenes de magnitud de dosis en las afueras de la instalación son acordes a lo esperado. La diferencia entre los gráficos de flujo y dosis tiene que ver con que no solo importa la cantidad de partículas presentes en el espacio sino también el espectro que presentan y la relación con el coeficiente de conversión a dosis. Es por eso que se observa que hay mayor dosis por neutrones en las afueras del dispositivo de neutrografía en comparación con la de fotones, mientras no se veían grandes diferencias para el caso de los flujos.

De manera más precisa, para comparar estos resultados se utilizaron datos de mediciones experimentales, las cuales se detallan en el informe de relevamiento radiológico [14], y también resultados obtenidos a partir de trabajos anteriores donde se realizaron metodologías independientes de algoritmos de KDE para el mismo caso de estudio [12]. Los puntos de interés son los que se observan en la Figura 4.9.

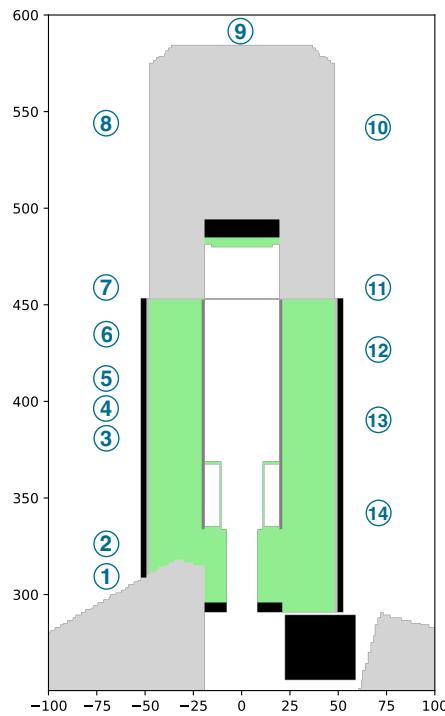


Figura 4.9: Puntos de interés para la comparación del cálculo de tasa de dosis ambiental en las afueras del dispositivo de neutrografía.

Con el objetivo de validar la herramienta KDSource se realizaron las simulaciones con las nuevas implementaciones de *kernel* y cambio de variables. En la Figura 4.10 se ve la comparación para los puntos de interés en el caso de las mediciones de dosis de neutrones. Los puntos graficados son aquellos donde se tenía una medición experimental y ergo se podía realizar una comparación entre las distintas metodologías.

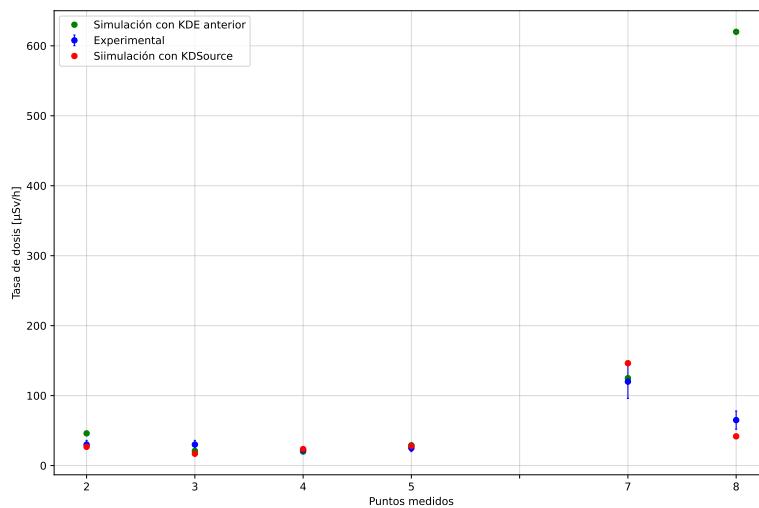


Figura 4.10: Comparación de tasa de dosis ambiental por neutrones obtenidas con mediciones experimentales y simulaciones.

Los resultados de la comparación de la tasa de dosis por fotones en los puntos de interés se observa en la Figura 4.11.

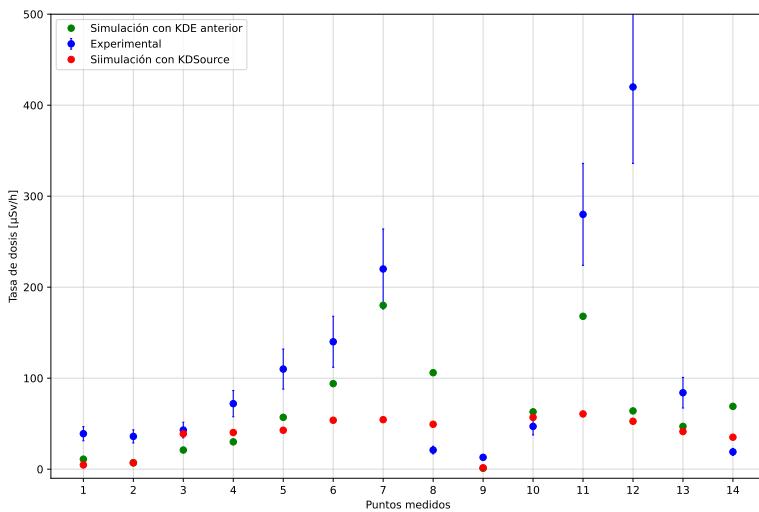


Figura 4.11: Comparación de tasa de dosis ambiental por fotones obtenidas con mediciones experimentales y simulaciones.

Para ambos gráficos se ve una diferencia entre los puntos calculados por simulación utilizando KDSource, en comparación con aquellos calculados con la metodología de KDE anterior. Se puede observar que los datos experimentales utilizados para distintos puntos análogos en la geometría, como por ejemplo los puntos 5 y 12 para el caso de los fotones, tienen valores muy distintos en tasa de dosis. Esto motivaría a incluir nuevas mediciones experimentales en trabajos futuros para poder obtener datos más confiables. En la Tabla 4.1 se observan los errores relativos a los datos experimentales asociados a cada punto para dosis por neutrones y fotones.

Punto	Error relativo fotones		Error relativo neutrones	
	(C-E)/E *100 %		(C-E)/E *100 %	
	KDE	KDSource	KDE	KDSource
1	71,8	88,1	-	-
2	80,6	80,6	53,3	10,7
3	51,2	9,6	30,0	43,4
4	58,3	44,1	9,1	7,2
5	48,2	61,1	16,0	12,5
6	32,9	61,6	-	-
7	18,2	75,3	79,2	21,9
8	404,8	134,9	853,8	35,7
9	92,3	88,6	-	-
10	34,0	21,1	-	-
11	40,0	78,3	-	-
12	84,8	87,5	-	-
13	44,0	50,7	-	-
14	263,2	84,8	-	-
Promedio ponderado	60,1	72,1	233,2	24,1

Tabla 4.1: Tabla comparativa de valores de errores relativos para el cálculo de dosis por las distintas metodologías de generación de fuentes de radiación para las simulaciones, con valores calculados (C) y experimentales (E). El promedio fue ponderado con la dosis obtenida para cada punto.

En base al análisis de promedios de errores ponderados con la dosis, se puede observar que los errores de los neutrones mejoraron considerablemente utilizando KDSource. Por otra parte, si bien el promedio para fotones es mayor, estos se mantienen en el mismo orden de magnitud. Las Tablas de los resultados obtenidos sobre las dosis calculadas se presentan en el Apéndice B.

4.2.3. Cociente dosis-flujo

Como análisis adicional, si calculamos el cociente entre la tasa de dosis y el flujo obtenidos en las simulaciones, se podría ver la dependencia energética del factor de conversión de la fluencia en dosis de equivalente ambiental. Las expresiones tanto para la tasa de dosis como para el flujo se observan en las Ecuaciones 4.3 y 4.4 respectivamente:

$$\dot{H}^*(10) = \int_0^\infty \phi(E) h^*(10; E) dE \quad (4.3)$$

$$\Phi = \int_0^\infty \phi(E) dE \quad (4.4)$$

donde $h^*(10; E)$ es el factor de conversión de fluencia a dosis equivalente ambiental expresado en $[pSv.cm^2]$, y ϕ la distribución de flujo en energía en $[n/cm^2.s]$. Entonces,

si calculamos el cociente entre la tasa de dosis y el flujo obtenemos la Ecuación 4.5.

$$\overline{H}^*(10) = \frac{\dot{H}^*(10)}{\Phi} = \frac{\int_0^\infty \phi(E) h^*(10; E) dE}{\int_0^\infty \phi(E) dE} \quad (4.5)$$

Esto se puede pensar como un valor de factor efectivo teniendo en cuenta la distribución energética del flujo. Para realizar un análisis del cociente propuesto, es necesario en primer lugar conocer las distribuciones energéticas de $h^*(10; E)$ y $\phi(E)$. En la Figura 4.12 se puede observar el comportamiento del factor de conversión tanto para neutrones como para fotones, en el rango de interés para cada caso.

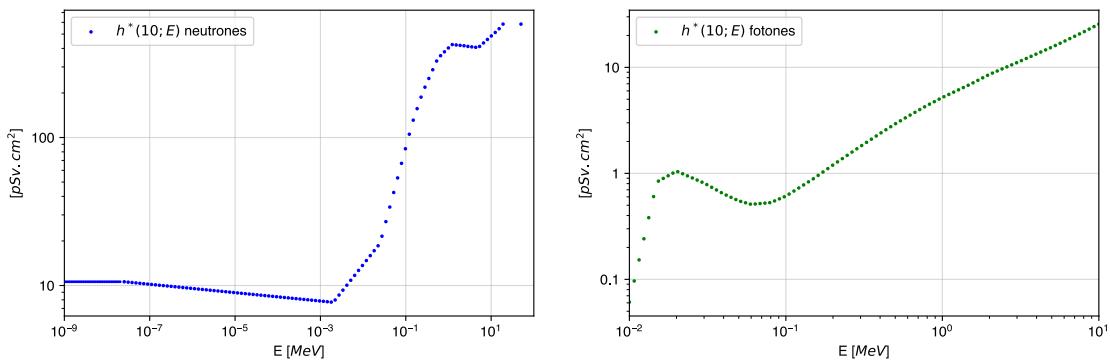


Figura 4.12: Gráfica de la relación del factor de conversión de fluencia en dosis equivalente ambiental con la energía, para neutrones y fotones.

Se observa que no necesariamente los factores aumentan con la energía de las partículas. Sin embargo, para energías muy grandes sí se observa este comportamiento creciente. Por otra parte, en la Figura 4.13 se muestran los espectros de estas partículas en el haz del Conducto 1 del reactor RA-6.

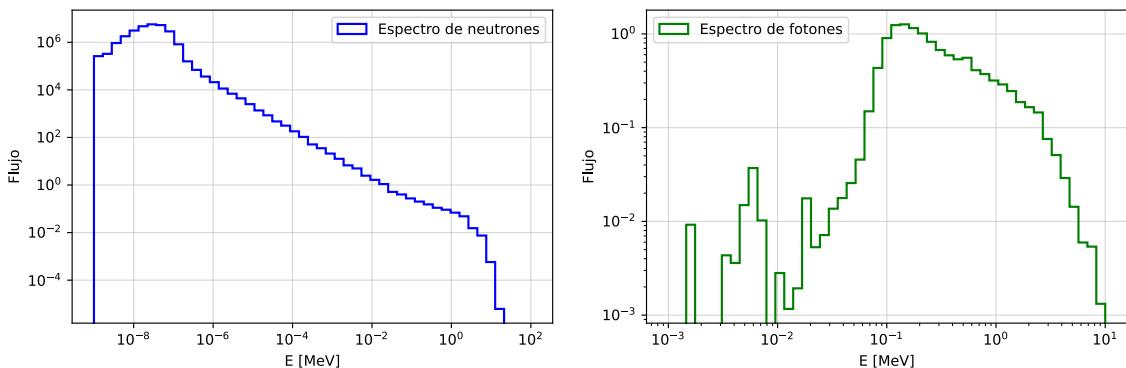


Figura 4.13: Espectro de neutrones y fotones en la entrada del colimador del haz, en el Conducto 1 del reactor RA-6.

Habiendo visto estos comportamientos, ahora sí se analizarán los resultados del cociente entre la tasa de dosis y el flujo. Estos se pueden observar en la Figura 4.14.

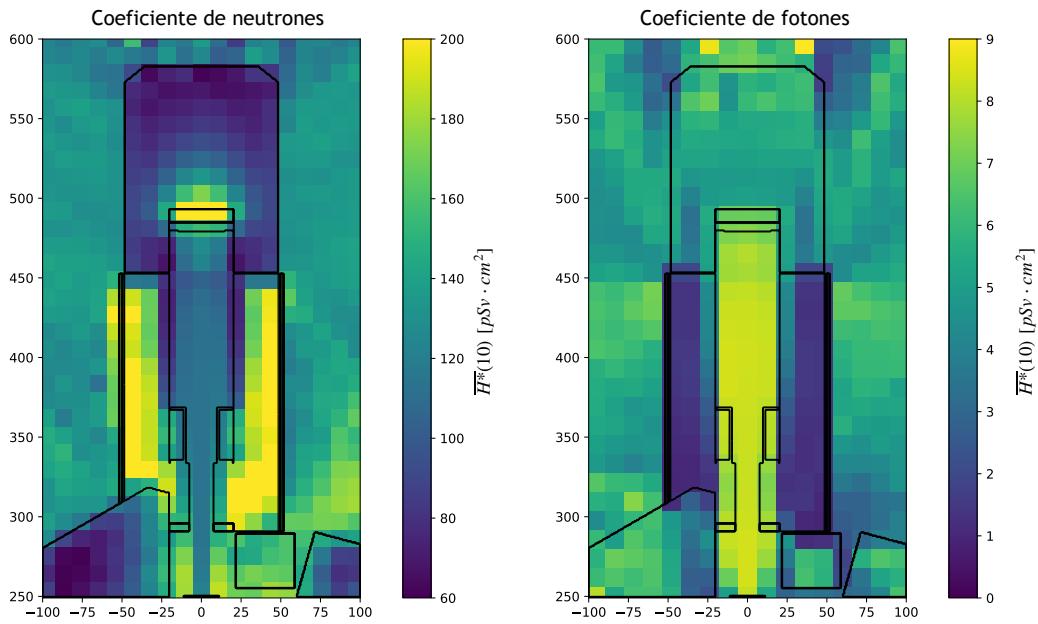


Figura 4.14: Cociente entre la tasa de dosis y el flujo obtenido para las simulaciones del dispositivo de neutrógrafía utilizando OpenMC junto con KDSource.

En el gráfico de la Figura 4.14, se muestra el cociente de la Ecuación 4.5. Se puede observar un aumento del coeficiente de conversión conforme el haz ingresa a la zona donde se encuentra el poliboro. En este lugar la absorción de neutrones térmicos endurecería el espectro, resultando en valores de $\bar{H}^*(10)$ mayores. Por otra parte, en la imagen de la derecha se observa que el menor factor de conversión efectivo se da en las proximidades del plomo. En este caso, el material tiene la capacidad de atenuar el haz de fotones incidente, generando un espectro más blando de radiación gamma.

Capítulo 5

Conclusiones

5.1. Conclusiones

Se optimizaron algoritmos de estimación de densidades para el cálculo de haces de neutrones y fotones, aplicados a la herramienta KDSource.

En primer lugar, se emplearon algoritmos de KDE en la cadena de cálculo de Monte Carlo mediante desarrollos independientes que se aplicaron al caso de estudio del acelerador lineal LINAC. Se realizó un estudio paramétrico con el objetivo de optimizar el valor del ancho de banda acorde al problema a resolver, para el ajuste de las distribuciones y futuro muestreo de partículas. Utilizando el valor de ancho de banda óptimo se desarrollaron las dos implementaciones propuestas: alternativa de *kernel* Epanechnikov y de variables espaciales cilíndricas. Luego del análisis de resultados preliminares se concluyó que estas nuevas implementaciones generaban cambios positivos en las distribuciones de las variables que conforman el espacio de fases. Es decir, se obtuvieron partículas muestreadas cuyas distribuciones se asemejaban más a las partículas originales del experimento. En particular, se obtuvo el espectro neutrónico del agua liviana utilizando una cadena de cálculo que constaba en la simulación en OpenMC, la utilización de algoritmos KDE y una última simulación con McStas. Los resultados fueron comparables con datos experimentales.

Se estudió la herramienta KDSource, junto con sus componentes, estructuras y flujo de información. Esto permitió agregar de manera coherente y efectiva las optimizaciones desarrolladas.

En otra etapa, se utilizó el Conducto 1 del reactor RA-6 para validar los cambios realizados en la herramienta KDSource. Se agregó la utilización de esta herramienta en dos superficies para aumentar la estadística de las partículas. Se analizaron los flujos de neutrones y fotones, y se llegaron a valores de tasa de dosis ambiental en las afueras del dispositivo de neutrografía ubicado a la salida de este haz. Se compararon estos resultados con aquellos experimentales y con otros obtenidos con una metodología similar.

Si bien existen errores considerables entre los valores obtenidos y los experimentales, estos tienen un error ponderado con la tasa de dosis de 72,1 % para fotones y 24,1 % para neutrones. Para finalizar, se analizó el cociente entre la tasa de dosis y el flujo. Con esto se pudo visualizar como varía el factor efectivo de conversión de flujo a dosis en zonas donde ocurren cambios en el espectro de neutrones y fotones debido a los materiales presentes, como el poliboro y el plomo. Se encontró que las zonas geográficas donde el factor efectivo de conversión adopta valores más altos se corresponden con espectros de partículas más endurecidos energéticamente.

5.2. Trabajo a futuro

Como trabajo a futuro se presentan alternativas analizadas pero no desarrolladas durante el trabajo que podrían ser beneficiosas para la herramienta, así como mejoras sobre los principales obstáculos que se presentaron durante el desarrollo:

- Optimizar el flujo de información entre los códigos de cálculo y KDSouce. Actualmente, los datos generados en OpenMC deben transformarse del formato `h5` a `mcpl` para utilizarlo en KDSouce. Luego, si se quiere realizar la segunda simulación en OpenMC o algún otro código que no acepte como input el formato `mcpl` se debe hacer el procedimiento inverso, lo cual es muy costoso computacionalmente cuando se tratan grandes cantidades de partículas. Como solución se propone realizar una opción *on-the-fly* para poder optimizar los tiempos y pasos en la cadena de cálculo. Así, las partículas generadas en KDSouce serían directamente simuladas en el código que se desee, sin necesidad de almacenar esta información en archivos.
- Continuar la validación de la herramienta. A pesar de que en este trabajo ya se llevó a cabo una validación, resultaría útil contar con otros casos de estudio donde se utilice la herramienta y se comparén los resultados.

Apéndice A

Validación del código OpenMC: Benchmark de criticidad

Con el objetivo de realizar una validación de la herramienta OpenMC, se realizó una simulación de criticidad (cálculo de autovalores) utilizando el *benchmark* IEU-COMP-FAST-001 del *International Criticality Safety Benchmark Evaluation Project* (ICSBEP) [2] provisto por la *Nuclear Energy Agency* (NEA) [15]. Dicho benchmark llamado ZPR-6 ASSEMBLY 6A, se trata de un reactor de espectro rápido de geometría cilíndrica, con óxido de Uranio como combustible y refrigerado por Sodio. En la Figura A.1 se observa la geometría junto con los materiales utilizados. Las regiones IC1 y OC1 corresponden al núcleo, IAB1 e IAB2 a los blankets internos, OAB1 y OAB2 a los blankets externos, RR1 y RR2 a los reflectores radiales, y MAT a la matriz externa del ZPR-6/6A.

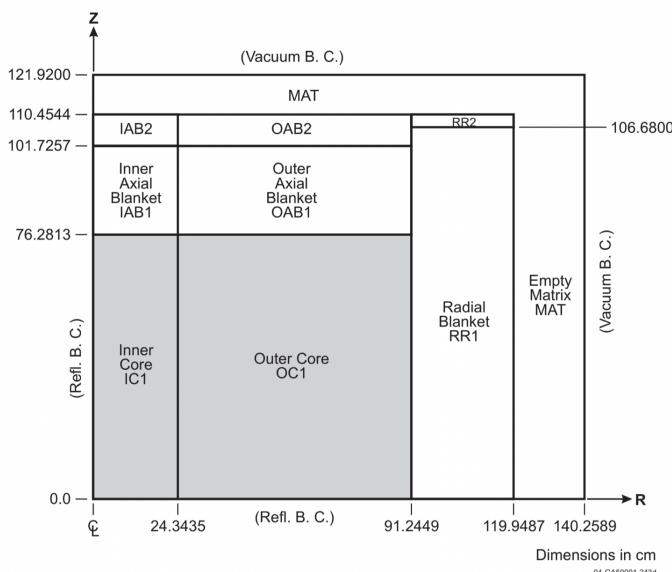


Figura A.1: Esquema de la geometría utilizada para el *benchmark* de criticidad ZPR-6/6A. Imagen obtenida de documento un documento de NEA [2].

En la Tabla A.1 se pueden observar los resultados obtenidos con el código propio en OpenMC, junto con los datos oficiales del *benchmark* en MCNP.

Experimental $k_{eff} \pm \sigma$	OpenMC ((ENDF/B-VII.1)) $k_{eff} \pm \sigma$	C-E [pcm]
$0,9939 \pm 0,0023$	$0,9929 \pm 0,0001$	100 ± 230

Tabla A.1: Tabla comparativa de los resultados experimentales, obtenidos con los datos del *benchmark* de MCNP y con la simulación propia realizada en OpenMC, junto con la diferencia de los resultados.

A partir de los resultados de la Tabla A.1, se verifica la correcta utilización del código y su biblioteca por parte del usuario.

Apéndice B

Cálculos de dosis del haz de neutrografía

A continuación se presentan las tablas con los valores de tasa de dosis ambiental. Los datos “Experimentales” se obtuvieron del informe de relevamiento radiológico [14]. Los datos de “KDE anterior” corresponden a los resultados obtenidos mediante la metodología de KDE desarrollada en un trabajo anterior para el mismo caso de estudio [12]. En la Tabla B.1 se tienen los datos para los neutrones, mientras que la Tabla B.2 aquellos de los fotones.

Punto	Tasa de dosis [$\mu\text{Sv/h}$]				
	Experimental	Error	KDE anterior	KDSource	Error
2	30	6	46	26,8	2,3
3	30	6	21	17,0	1,1
4	22	4,4	20	23,6	1,2
5	25	5	29	28,13	1,4
7	120	24	215	146,3	3,8
8	65	13	620	41,8	3,1

Tabla B.1: Comparación entre valores de tasa de dosis ambiental por neutrones, obtenidos con las distintas simulaciones.

Punto	Tasa de dosis [$\mu\text{Sv/h}$]				
	Experimental	Error	KDE anterior	KDSource	Error
1	39	7,8	11	4,6	2,9
2	36	7,2	7	7,0	3,9
3	43	8,6	21	38,9	4,0
4	72	14,4	30	40,3	3,9
5	110	22	57	42,8	2,9
6	140	28	94	53,8	1,6
7	220	44	180	54,4	1,0
8	21	4,2	106	49,3	2,5
9	13	2,6	1	1,5	3,1
10	47	9,4	63	56,9	1,3
11	280	56	168	60,7	2,0
12	420	84	64	52,6	1,6
13	84	16,8	47	41,4	2,1
14	19	3,8	69	35,1	1,8

Tabla B.2: Comparación entre valores de tasa de dosis ambiental por fotones, obtenidos con las distintas simulaciones.

Los valores de error obtenidos para el cálculo con KDSource fueron calculados a partir de un *tally* de eventos.

Apéndice C

Modificaciones del código

A continuación se detallan las modificaciones realizadas en el código de la herramienta KDSource, con el objetivo de implementar las optimizaciones propuestas.

Cambio de variables

A continuación se observan las nuevas clases implementadas para posibilitar el cambio de variables espaciales a cilíndricas. Estas modificaciones fueron realizadas en los archivos asociados a la geometría, tanto en Python como en C.

```
1 class SurfCirc(Metric):
2     def __init__(self, rho_min=0, rho_max=np.inf, psi_min=-np.pi, psi_max=np.pi, z=0):
3         """
4             Polar parametrization for position.
5
6             Polar variables are defined as follows:
7                 rho:
8                     psi: azimuthal angle, starting from x direction, in [deg].
9
10            Spatial variable rho is delimited between a min and max
11            value. By default these are zero and infinity,
12            respectively. All positions in the particle list should be
13            inside these limits.
14
15            z has the fixed value given as argument.
16        """
17        super().__init__([1, 2, 3], ["rho", "psi"], ["cm", "deg"], "deg.cm")
18        self.z = z
19        self.rho_min = rho_min
20        self.rho_max = rho_max
21        self.psi_min = psi_min
22        self.psi_max = psi_max
```

```

25
26     def transform(self, poss):
27         """Transform volume position (x,y,z) to circular flat position (rho,psi)."""
28         rhos = np.sqrt(poss[:,0]**2+poss[:,1]**2)
29         psis = np.arctan2(poss[:,1], poss[:,0])*180/np.pi
30         return np.stack((rhos, psis), axis=1)
31
32     def inverse_transform(self, poss):
33         """Transform polar flat position (rho,psi) to volume position (x,y,z)."""
34         z_col = np.broadcast_to(self.z, (*poss.shape[:-1], 1))
35         x_col = poss[:,0]*np.cos(poss[:,1]* np.pi / 180)
36         y_col = poss[:,0]*np.sin(poss[:,1]* np.pi / 180)
37         return np.stack((x_col, y_col, z_col), axis=1)
38
39     def save(self, mtree):
40         """Save SurfCirc parameters into XML tree."""
41         ET.SubElement(mtree, "dim").text = str(self.dim)
42         paramsel = ET.SubElement(mtree, "params")
43         paramsel.set("nps", "5")
44         paramsel.text = "{} {} {} {} {}".format(
45             self.rho_min, self.rho_max, self.psi_min, self.psi_max, self.z
46         )
47
48     def jac(self, poss):
49         """Jacobian of cylindrical transformation."""
50         rhos = np.sqrt(poss[:,0]**2+poss[:,1]**2)
51         return 1 / rhos.reshape(-1)
52
53     @staticmethod
54     def load(mtree):
55         """Load parameters from XML tree and build SurfXY."""
56         dim = int(mtree[0].text)
57         params = np.array(mtree[1].text.split(), dtype="float64")
58         if dim != 2 or len(params) != 5 or int(mtree[1].attrib["nps"]) != 5:
59             raise Exception("Invalid metric tree.")
60         return SurfXY(*params)

```

Listing C.1: Clase SurfCirc creada como opción de variables espaciales cilíndricas. Agregada al archivo `geom.py`.

```

1 int SurfCirc_perturb(const Metric* metric, mcpl_particle_t* part, double bw, char kernel){
2     double rho_min=metric->params[0], rho_max=metric->params[1];
3     double psi_min=metric->params[2], psi_max=metric->params[3];
4     double rho, psi, rho2, psi2;
5
6     rho = sqrt(part->position[0]*part->position[0]+part->position[1]*part->position[1]);
7     psi = atan2(part->position[1], part->position[0]);
8
9     rho2 = rho + bw*metric->scaling[0] * rand_type(kernel);
10    while((rho2 < rho_min) || (rho2 > rho_max)){
11        if(rho2 < rho_min) rho2 += 2 * (rho_min - rho2);
12        else                 rho2 -= 2 * (rho2 - rho_max);
13    }
14
15    psi2 = psi + bw*metric->scaling[1]*M_PI/180 * rand_type(kernel);
16    while((psi2 < psi_min) || (psi2>psi_max)){
17        if(psi2 < psi_min) psi2 += 2 * (psi_min - psi2);
18        else                 psi2 -= 2 * (psi2 - psi_max);
19    }
20
21    part->position[0] = rho2*cos(psi2);
22    part->position[1] = rho2*sin(psi2);
23    return 0;
24 }

```

Listing C.2: Clase SurfCirc_perturb creada para el muestreo de partículas en variables cilíndricas. Agregado al archivo geom.c.

Cambio de kernel

Con respecto al cambio de *kernel*, los cambios realizados en el código se presentan en modificaciones a todas las clases y estructuras utilizadas. Esto se debe a que la información sobre la elección de esta función debe ser accesible en todo momento durante la utilización de esta herramienta. Se realizaron modificaciones en las estructuras y funciones de Python y C, así como en la lectura y escritura de los archivos XML. Es por esto que no se muestran a continuación todas las líneas en las cuales surgieron modificaciones por esta implementación. Sin embargo, se pueden ver a continuación las funciones `rand_epan` y `rand_type`, desarrolladas para el muestreo de partículas.

```

1 double rand_epan(){
2     double x1=(double)rand()/(double)RAND_MAX/2.0) - 1.0;
3     double x2=(double)rand()/(double)RAND_MAX/2.0) - 1.0;
4     double x3=(double)rand()/(double)RAND_MAX/2.0) - 1.0;
5     if(abs(x3)>=abs(x2) && abs(x3)>=abs(x1)) return x2;
6     else return x3;
7 }
8

```

```
9 double rand_type(char kernel){  
10     if(kernel=='g') return rand_norm();  
11     if(kernel=='e') return rand_epan();  
12     else {  
13         printf("Cannot perturbate with current kernel. \n");  
14         return 0;  
15     }  
16     return 0;  
17 }
```

Listing C.3: Funciones implementadas en el muestreo de nuevas partículas. Agregado al archivo `utils.c`.

Apéndice D

Actividades relacionadas a la Práctica Profesional Supervisada

La Práctica Profesional Supervisada (PPS) se llevó a cabo en el Departamento de Física de Reactores y Radiaciones del Centro Atómico Bariloche durante los últimos 2 semestres de la carrera Ingeniería Nuclear, bajo la supervisión del Magister Ariel Marquez y el Magister Norberto Schmidt.

Las actividades desarrolladas a lo largo de la PPS fueron las siguientes:

- Se estudiaron las herramientas OpenMC y McStas.
- Se tomó conocimiento integral sobre la herramienta KDSource.
- Se modificaron estructuras y componentes ya existentes con el fin de optimizar dicha herramienta.
- Se realizaron desarrollos sobre algoritmos de KDE.
- Se realizaron simulaciones sobre distintas facilidades como el LINAC y el dispositivo de neutrografía del RA6.
- Se elaboró el informe final.

Total de horas: 420hs.

Bibliografía

- [1] Kdsource, a tool for the generation of monte carlo particle sources using kernel density estimation, 2022.
- [2] Smith, M. A. Zpr-6 assembly 6a: A cylindrical assembly with uranium oxide fuel and sodium with a thick depleted-uranium blanket. *International Handbook of Evaluated Criticality Safety Benchmark Experiments (ICSBEP)*, IEU-COMP-FAST-001. Organization for Economic Co-operation and Development-Nuclear Energy Agency (OECD-NEA), 2016.
- [3] Romano, P. K., Horelik, N. E., Herman, B. R., Nelson, A. G., Forget, B., Smith, K. OpenMC: A state-of-the-art Monte Carlo code for research and development, tomo 82. Elsevier, 2015.
- [4] OpenMC. <https://docs.openmc.org/en/stable/>.
- [5] ENDF/B-VII. 0: next generation evaluated nuclear data library for nuclear science and technology. *Nuclear data sheets 107*, pág. 2931 3060, 2006.
- [6] Abbate, O. I. Kdsource: Desarrollo de una herramienta computacional para el cálculo de blindajes. Tesis de carrera de maestría en ingeniería, Instituto Balseiro, Diciembre 2021.
- [7] McStas. McStas - A neutron ray-trace simulation package. <https://www.mcstas.org>.
- [8] B.W., S. Density estimation for statistics and data analysis. London: Chapman and Hall, 1986.
- [9] Kernel density estimation method for monte carlo global flux tallies. Nuclear science and engineering, 2012.
- [10] Facility for the measurement of neutron spectra by the time of flight method. Technical Report, Comisión Nacional de Energía Atómica, 1974.
- [11] Descripción de la instalación experimental de radiografía de neutrones del ra6. Inf. Téc. ITE-EN-GIN-RI-PL-010, CNEA, 2021.

- [12] Z., P. Incorporación de algoritmos de kde al modelado de fuentes de radiación en cálculos de monte carlo. Proyecto Integrador de la carrera de Ingeniería Nuclear, Instituto Balseiro, Junio 2021.
- [13] ARN. Factores dosimétricos para irradiación externa y contaminación interna, y niveles de intervención para alimentos. https://www.argentina.gob.ar/sites/default/files/gr1-r1_0.pdf, 2003.
- [14] Marín J., A. J. Relevamiento radiológico preliminar de la instalación experimental de neutrografía del ra-6 a 1mw. Inf. Tec. ITE-EN-GIN-RI-PL-012, CNEA, 2021.
- [15] Verification and validation benchmarks. *Nuclear Engineering and Design*, 238 (3), 716-743, 2008.

Agradecimientos

A mamá y papá, por su amor y apoyo incondicional. Ellos siempre fueron los primeros en creer en mi, hasta cuando ni yo lo hacía. Porque si no fuese por ellos no sería quien soy.

A mis hermanas. A Martu, por tener más ganas de que termine que yo mismo. A Barbie, por ser el mejor ejemplo de superación y de que ningún objetivo es inalcanzable.

A Ariel y Norbert, que me guiaron y ayudaron en esta última etapa. Por la predisposición infinita que tuvieron desde donde sea que estaban y por motivarme a aprender constantemente. A Zoe, que siempre estuvo para aconsejarme y orientarme.

A Yayi, Bruce, Renat y todo el matsi que hicieron del tiempo en Bariloche una experiencia inolvidable. Mención especial a Maxi y Fran que siempre me abrieron las puertas del cuarto para charlar y descargarme. A Joaco, por la ganas de hacer siempre algo diferente para despejarnos del estudio.

A mis amigos de toda la vida, que me recibieron con un abrazo siempre que volví a casa.

Al IB y al ITBA por la oportunidad que me dieron de acceder a una educación de excelencia.

