UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
DEPARTAMENTO DE ELECTRÓNICA E COMPUTACIÓN

Tesis Doctoral

Optimización de un simulador 3D paralelo aplicado al estudio de fluctuaciones de parámetros intrínsecos en dispositivos HEMT

Natalia Seoane Iglesias
Santiago de Compostela, Octubre 2006

Dr. **Antonio Jesús García Loureiro**, Profesor Contratado Doctor del Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela.

INFORMA:

Que la memoria titulada "**Optimización de un simulador 3D paralelo aplicado al estudio de fluctuaciones de parámetros intrínsecos en dispositivos HEMT**", ha sido realizada por Dña. **Natalia Seoane Iglesias** bajo mi dirección en el Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela y constituye la Tesis que presenta para optar al grado de Doctora en Ciencias Físicas.

Santiago, Octubre de 2006

Fdo. **Antonio J. García Loureiro**
Director de la tesis

Fdo. **Diego Cabello Ferrer**,
Director del Departamento de Electrónica e Computación.

Fdo. **Natalia Seoane Iglesias**,
Doctoranda

Agradecimientos

Deseo expresar mi agradecimiento a todas aquellas personas e instituciones que de una forma u otra han posibilitado la realización de este trabajo.

En primer lugar a mi director de tesis, Antonio García Loureiro, por su apoyo incondicional, su preocupación, su entusiasmo y su energía contagiosa.

Al Departamento de Electrónica e Computación, y en especial al Grupo de Arquitectura de Computadores, por posibilitarme los medios necesarios para la realización de este trabajo. A mis compañeros, tanto a los que ya se han ido, Alex, Javi y Marcos Boullón, que tanto me ayudaron en mis principios, como a los que aún se quedan, Diego, Xulio, Juan Ángel, Marcos Suárez y Óscar, que dan alegría a cada día de trabajo. En especial, quiero agradecer a Manuel su amistad, sus consejos sabios, sus correcciones estrictas y su ayuda impagable.

También tengo que agradecer a los miembros del Device Modelling Group de la Universidad de Glasgow su cálida acogida durantes mis estancias en Escocia. De forma especial quiero mencionar al Profesor Asen Asenov, por su contribución al desarrollo de esta tesis doctoral, y a Karol Kalna, por su ayuda, sus ideas y sus explicaciones pacientes.

Al EPCC (Edinburgh Parallel Computing Centre) y sobre todo al CESGA (Centro de Supercomputación de Galicia) por posibilitar y facilitar el acceso a sus recursos computacionales. Al Ministerio de Ciencia y Tecnología (MCYT) por la financiación a través del proyecto TIN2004-07797-C02.

Me gustaría agradecer a mis padres y a Patri su rescate cuando todo me parecía oscuro y difícil. Os lo debo todo.

Ana, Anita, Cris, gracias por estar siempre ahí, no puedo esperar a las aventuras que nos aguardan el año que viene. Azu y Alex, sois mi tesoro.

Jose, gracias por iluminar mi vida, quién diría que todo empezó con un café.

Por último, y aún a riesgo de ser repetitiva, me gustaría volver a agradecer a Antonio todos estos años de viaje en común. Ha sido maravilloso trabajar contigo.

A mis abuelos

Índice general

Índice de figuras

Índice de tablas

Introducción

Los dispositivos semiconductores están siendo escalados a dimensiones del orden de los nanómetros con el objetivo de mejorar cada vez más su rendimiento. El escalado tan drástico de los dispositivos aumenta la importancia de determinadas fuentes de fluctuaciones relacionadas con la naturaleza atomística de la carga y la materia, que se convierten en factores determinantes de la fiabilidad y el rendimiento de los dispositivos, y por lo tanto de los circuitos fabricados con ellos.

En el pasado, el desajuste en las curvas características de los transistores estaba principalmente asociado con variaciones en los parámetros de fabricación, lo que generaba variaciones macroscópicas en el grosor de las capas, en la geometría y en el dopado. En cambio, las fuentes de fluctuaciones que cobran importancia con la reducción del tamaño de los dispositivos son independientes de los procesos litográficos y no pueden ser eliminadas a través de mejoras en el proceso de fabricación. Así que, mientras que en las simulaciones numéricas convencionales los dispositivos se trataban como dispositivos perfectos, con interfaces y fronteras suaves y distribuciones continuas de dopado, ahora ya no será posible considerar un único dispositivo perfecto sino que será necesario simular un conjunto de transistores diferentes a nivel microscópico, puesto que, si se considera un conjunto de dispositivos, la inclusión de diversos tipos de fluctuaciones de parámetros intrínsecos provocará variaciones estadísticas entre ellos.

Las fuentes de fluctuaciones de parámetros intrínsecos que afectan a los dispositivos son de naturaleza tridimensional, por lo que, para capturar correctamente los efectos que provocan es necesario realizar simulaciones 3D. Otra consideración a tener en cuenta es la necesidad de simular un conjunto estadístico de dispositivos lo suficientemente grande que permita extraer con precisión suficiente los parámetros que caracterizan la distribución estadística. Por lo tanto, la técnica elegida para la simulación de estos efectos debe ser rápida y eficiente, permitiendo la simulación de un conjunto grande de dispositivos dentro de un período de tiempo relativamente corto. Por todo

ello, el simulador de dispositivos utilizado en este trabajo es tridimensional y se basa en la aproximación de arrastre–difusión, que representa el modelo más simple y menos costoso, desde el punto de vista de los recursos computacionales necesarios, usado en simulaciones multidimensionales.

El principal problema asociado con la simulación tridimensional es su elevado coste computacional, determinado por la gran cantidad de información implicada y el enorme número de cálculos a realizar. Además, en este caso particular, se suma la necesidad de realizar análisis estadísticos, lo que multiplica el coste computacional por el tamaño de la muestra estadística. Estos dos factores provocan que el uso de computadores convencionales en la resolución de estos problemas sea prohibitivo y generan la necesidad de la utilización de máquinas paralelas. Por ello, el simulador 3D de dispositivos utilizado en este trabajo está paralelizado a través de la librería MPI de paso de mensajes, usando los lenguajes de programación C y Fortran, garantizándose así la portabilidad del código.

Esta memoria se enmarca en una de las líneas de investigación del grupo de Arquitectura de Computadores del Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela. Es necesario comentar que este trabajo de investigación parte de un simulador, ya desarrollado, para transistores bipolares de homounión (BJT) y de heterounión (HBT). A partir de este proyecto[1], se desarrolla una extensión y optimización de este simulador para el estudio de dispositivos HEMT (High Electron Mobility Transistor) y la realización de análisis estadísticos de las fluctuaciones de parámetros intrínsecos que afectan a estos transistores.

El trabajo ha sido dividido en cinco capítulos, en los que se pretende dar, en primer lugar, una visión global del proceso de simulación de dispositivos semiconductores y en segundo lugar, el resultado de su aplicación al estudio de fluctuaciones de parámetros intrínsecos en dispositivos HEMT.

En el primer capítulo se introducen los principios básicos de los dispositivos HEMT, cuyo comportamiento es posteriormente estudiado en el simulador tridimensional de dispositivos. Los transistores HEMT están compuestos por heteroestructuras de semiconductores. Para ello, inicialmente se describen los principios físicos en los que se basan las heteroestructuras, para abordar a continuación la descripción de los dispositivos de efecto campo basados en heteroestructuras y su funcionamiento. Por último se resumen algunas de las aplicaciones y utilidades actuales de los dispositivos basados

[1]Información sobre este proyecto se encuentra en la tesis doctoral de Antonio Jesús García Loureiro, de título: **BIPS3D: Un simulador 3D paralelo de dispositivos bipolares BJT y HBT**, realizada en el Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela.

en heteroestructuras.

En el segundo capítulo inicialmente se describen las principales técnicas de simulación de dispositivos semiconductores utilizadas actualmente. A continuación se trata el modelo de arrastre–difusión, puesto que es el utilizado en el simulador 3D de dispositivos, describiéndose las ecuaciones matemáticas que componen este modelo, las ecuaciones de Poisson y de continuidad de portadores. Seguidamente se introduce el método de elementos finitos, utilizado en la discretización de las ecuaciones que forman el modelo de arrastre–difusión. La ecuación de Poisson es discretizada utilizando el método estándar, mientras que en las ecuaciones de continuidad de electrones y huecos se ha empleado la discretización de Scharfetter–Gummel puesto que conduce a mejores resultados.

En el tercer capítulo inicialmente se mencionan las principales etapas que componen el proceso de simulación. A continuación, en las diferentes secciones del capítulo se describe en profundidad cada una de estas etapas. En primer lugar se trata el proceso de generación y particionamiento de las mallas de elementos finitos que modelan el dispositivo discretizado. En segundo lugar se estudian las técnicas de linealización del sistema discretizado que se suelen aplicar en este ámbito, el método de Newton–Raphson y el método de Gummel. Seguidamente se tratan los métodos de resolución que se pueden utilizar para resolver los sistemas lineales, métodos directos e iterativos, junto con los precondicionadores usados para acelerar la convergencia de los métodos iterativos, puesto que en nuestro caso particular son los más eficientes en la resolución de los sistemas lineales. A continuación se introduce el concepto de reordenamiento de matrices dispersas y se muestran diversos formatos de almacenamiento de estas matrices. Para finalizar se resume la implementación de las diferentes etapas que componen el proceso de simulación de dispositivos en el simulador 3D paralelo basado en el modelo de arrastre–difusión.

En el cuarto capítulo se presentan diversas optimizaciones del simulador 3D de dispositivos utilizado en este trabajo, que tratan de minimizar el tiempo de simulación. Para la optimización se presentan diferentes estrategias y las mejoras en el tiempo de ejecución obtenidas después de su aplicación. Así, en primer lugar se enumeran las características del computador paralelo[2], un cluster HP Integrity Superdome, utilizado en la obtención de todos los resultados presentados en las secciones posteriores. En segundo lugar se tratan de encontrar los algoritmos de resolución de sistemas linea-

[2]El supercomputador paralelo utilizado en este trabajo pertenece al CESGA (Centro de Supercomputación de Galicia).

les más apropiados para nuestro problema, realizando para ello un análisis de los métodos de resolución y técnicas de precondicionamiento disponibles, así como de los parámetros que tienen una mayor importancia en el tiempo de ejecución. Una vez obtenidos los métodos de resolución más eficientes, en tercer lugar se busca optimizar toda la etapa de resolución de los sistemas lineales de ecuaciones implementada en el simulador, puesto que esta es la etapa más costosa de todo el proceso. Por último, se presenta una nueva estrategia de particionamiento de las mallas de dispositivos HEMT utilizadas, de tal forma que se tenga en cuenta la naturaleza física del fenómeno objeto de estudio.

En el quinto capítulo se trata el impacto de las fluctuaciones de parámetros intrínsecos en las curvas características de los dispositivos HEMT. Para ello, tal y como se comentó previamente, es necesario realizar análisis estadísticos de los resultados obtenidos. Por lo tanto, en este capítulo, inicialmente se definen ciertos conceptos estadísticos básicos que serán utilizados en apartados posteriores. En segundo lugar se comentan las principales ventajas y desventajas del uso de la aproximación de arrastre–difusión para la simulación de los efectos de las fluctuaciones de parámetros intrínsecos. En tercer lugar se describe la estructura de los dos dispositivos utilizados en el estudio, un dispositivo PHEMT de 120 nm de longitud de puerta con un canal de $In_{0.2}Ga_{0.8}As$ y un dispositivo HEMT de 50 nm de longitud de puerta con un canal de $In_{0.3}Ga_{0.7}As$. A continuación se muestra detalladamente el proceso de calibración de estos dispositivos, comparando para ello las curvas características obtenidas con el simulador 3D con las dadas por los resultados experimentales y por un simulador Monte Carlo 2D, desarrollado en el Device Modelling Group de la Universidad de Glasgow. Seguidamente se mencionan las principales fuentes de fluctuaciones intrínsecas que pueden aparecer en dispositivos MOSFET, puesto que en estos dispositivos estos efectos han sido ampliamente estudiados. Tomando esto como base, en el siguiente apartado se describen, de una forma más detallada, las principales fuentes de fluctuaciones que afectan a los HEMTs y que han sido analizadas en este trabajo. Para finalizar este capítulo se presentan los resultados numéricos obtenidos de este análisis. Este estudio se realiza por separado para los dos dispositivos estudiados, tratando tres fuentes de fluctuaciones diferentes, la variación aleatoria de la composición de los compuestos ternarios que forman el canal del dispositivo, la influencia de la naturaleza discreta de los átomos dopantes y la variación aleatoria en la carga interfacial presente entre dos fronteras del dispositivo. Todo este análisis se centra en la influencia de las fluctuaciones de parámetros intrínsecos en la corriente de drenador, aunque hay que tener en cuenta que las fluctuaciones pueden

afectar a otras variables. Por último, se muestra un estudio del impacto de las fluctuaciones de parámetros intrínsecos en la frecuencia de corte de los dispositivos.

Para finalizar se indican las principales conclusiones y aportaciones de este trabajo, así como las líneas de investigación abiertas por el mismo.

Capítulo 1

Introducción a los transistores de heteroestructura de efecto campo

En el año 1965 Gordon E. Moore hizo la observación de que, en el pasado, había habido un crecimiento exponencial en el número de transistores usados en los circuitos integrados y postuló que esta tendencia continuaría en el futuro [1]. En la actualidad esta afirmación es conocida como la ley de Moore. De forma sencilla, la ley de Moore expresa que cada tres años se duplica el rendimiento de los transistores y se cuadriplica el número de dispositivos presentes en un chip [2]. El espectacular progreso que supone la ley de Moore ha sido alcanzado gracias al escalado drástico de los dispositivos, ganando así en velocidad y en densidad de integración. El documento conocido como *SIA's International Technology Roadmap for Semiconductors* (ITRS) es una guía para la industria de los semiconductores que revisa los parámetros de diseño necesarios para poder ajustarse tanto a la ley de Moore como a las barreras tecnológicas a superar. Con la reducción de los dispositivos a dimensiones nanométricas las reglas de escalado utilizadas hasta el momento de forma exitosa han empezado a fallar. Muchos de los problemas que ahora surgen son intrínsecos a la naturaleza de los semiconductores y no pueden ser eliminados por medio de mejoras en el procesamiento o en el equipamiento [3]. Por ejemplo, el escalado de los dispositivos MOSFET (*Metal Oxide Semiconductor Field Effect Transistor*) a nodos tecnológicos inferiores al de 50 nm requiere la superación de barreras de naturaleza física que limitan su rendimiento [4, 5, 6, 7]. Algunos de los problemas más

frecuentes son:

- Corrientes de fuga a través del óxido de puerta.
- Efecto túnel de portadores desde la fuente al drenador o desde el drenador hacia el interior del dispositivo.
- Control de la densidad y posición de los átomos dopantes en el canal y en las regiones de fuente y drenador para poder proporcionar una elevada relación entre las corrientes de conducción y de corte.

Los límites asociados con el escalado y el rango de aplicación de los MOSFETs convencionales han dado un impulso a la investigación y desarrollo de propuestas alternativas en el diseño de transistores y en el uso de nuevos materiales. Estas propuestas pueden ser clasificadas en función de la técnica que usan para intentar mejorar el rendimiento del dispositivo. Así es posible:

- Inducir una densidad de carga más elevada para una tensión de puerta dada. Por ejemplo, esto puede lograrse reduciendo la temperatura de funcionamiento del sistema [2] o usando un dispositivo FET de doble puerta (*Double Gate FET*) [8, 9].
- Aumentar el transporte de portadores elevando la movilidad, la velocidad de saturación o el transporte balístico. Para ello, entre otras técnicas, es posible disminuir la temperatura de funcionamiento, reducir la influencia de factores que degradan la movilidad, minimizando el campo eléctrico transversal o la dispersión culombiana debida a átomos dopantes, o utilizar materiales de alta movilidad y velocidad de saturación, como pueden ser Ge, InGaAs o InP. Entre los ejemplos de dispositivos que utilizan estas técnicas se encuentran los HEMT [10], PHEMT [11, 12], MHEMT [13], SiGe MOSFET [14], HBT [15] y DHBT [16].
- Asegurar la escalabilidad del dispositivo a una longitud de puerta más pequeña. Esto se puede conseguir utilizando perfiles de dopado más bruscos, a través de una capacidad de puerta elevada o manteniendo un buen control electrostático del potencial en el canal. Esto ocurre en dispositivos Double Gate FET, Ground Plane FET y Ultra Thin Body SOI MOSFET [17] entre otros.
- Reducir capacidades y resistencias parásitas. Estas técnicas son empleadas en dispositivos SOI [18] y Double Gate FET.

Este trabajo se centra en los dispositivos HEMT y PHEMT, basados en la formación de heteroestructuras de semiconductores. Este tipo de dispositivos están reemplazando rápidamente a las tecnologías convencionales en aplicaciones que requieran alta ganancia y ruido reducido, sobre todo para frecuencias superiores a los 10 GHz. Así en este capítulo, inicialmente se describen los principios físicos en los que se basan las heteroestructuras, para abordar a continuación la descripción de los dispositivos de efecto campo basados en heteroestructuras y su funcionamiento, análogo en cierto modo al de los transistores MOSFET. Por último se muestran algunas de las aplicaciones y utilidades actuales de los dispositivos basados en heteroestructuras.

1.1. Heteroestructuras de semiconductores

Una heteroestructura se forma al poner en contacto dos materiales semiconductores pertenecientes al grupo IV (por ejemplo Si y SiGe) o semiconductores compuestos de los grupos III–V, siendo una de las técnicas más utilizadas para ello el crecimiento epitaxial. Las principales propiedades que afectan al comportamiento de la heteroestructura son las constantes de red, concentraciones de dopado y energías de la banda prohibida (E_{gap}).

La figura 1.1 muestra la dependencia de la energía de la banda prohibida y de la longitud de onda asociada frente a la constante de red para los principales semiconductores que dan lugar a heteroestructuras. Además, también se evalúa el desajuste de la constante de red de cada uno de los materiales con respecto a la del silicio. En la formación de heteroestructuras es posible utilizar, además de los elementos de la figura, combinaciones de estos elementos dando lugar, a compuestos ternarios si se forman por tres elementos, o cuaternarios si están formados a partir de cuatro elementos. Los elementos que puedan ser unidos en la gráfica por una línea aproximadamente vertical proporcionarán aleaciones de constantes de red muy similares, siendo sus energías de banda prohibida diferentes en función de la composición. Las líneas continuas de la figura unen elementos que darían lugar a materiales con banda de separación directa, en los que el mínimo de la banda de conducción coincide con el máximo de la banda de valencia en una representación energía–momento. En cambio, elementos unidos por líneas discontinuas formarían semiconductores con banda de separación indirecta. Al desplazarse a lo largo de las líneas se varía la fracción molar que aporta cada uno de los componentes en la formación del compuesto ternario. En el caso de materiales cuaternarios se obtendría una superficie delimitada por tres compuestos (puntos en la gráfica).

Una de las heteroestructuras más utilizadas es la formada entre GaAs

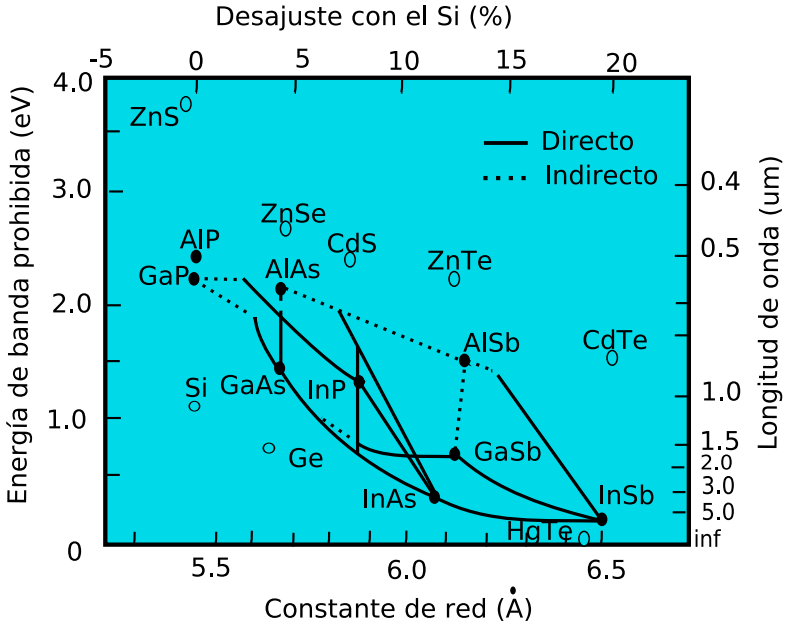


Figura 1.1: Representación de la energía de banda prohibida y de la longitud de onda asociada frente a la constante de red para las aleaciones de semiconductores más comunes.

y AlAs, o su compuesto relacionado el $Al_xGa_{1-x}As$, siendo x la razón de mezcla. En la heteroestructura $GaAs - Al_xGa_{1-x}As$ los dos materiales que la componen tienen constantes de red prácticamente idénticas, propiedad muy importante puesto que evita la aparición de tensiones o dislocaciones perjudiciales. En la figura 1.2 se representan las bandas de conducción y valencia de los dos componentes de la heteroestructura. En la heterounión se cumple que $\Delta E_{gap} = \Delta E_c + \Delta E_v$. La anchura de la banda prohibida del GaAs es 1.42 eV a 300K mientras que en el caso del $Al_xGa_{1-x}As$ varía con la composición de la siguiente forma:

$$E_{gap}(x) = \begin{cases} 1.424 + 1.247x & 0 < x < 0.45 \\ 1.9 + 0.125x + 0.143x^2 & x > 0.45 \end{cases}$$

En general, la energía de la banda prohibida de los materiales semiconductores que forman la heteroestructura es diferente, por lo que las bandas de conducción (BC) y valencia (BV) de los dos materiales no pueden ser continuas simultáneamente en la heterointerfaz, sino que, en el caso más general, ambas son discontinuas en la superficie de separación. Dependiendo de la

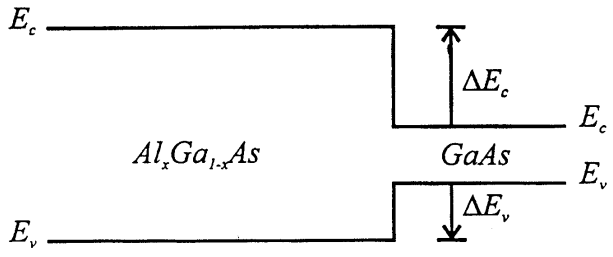


Figura 1.2: Representación de las bandas de conducción y valencia en una heterounión AlGaAs–GaAs.

aplicación el valor de la energía de la banda prohibida se ajustará cambiando los elementos que componen la heteroestructura, por ejemplo cambiando Galio por Indio o Aluminio, o bien variando la composición de la aleación.

Una heterounión se crea al poner en contacto dos materiales semiconductores distintos. En la figura 1.3 se representa el diagrama de bandas de energía de una heteroestructura formada por dos materiales semiconductores, teniendo uno de ellos una energía de banda prohibida más elevada que el otro. En el equilibrio el nivel de Fermi es constante y lejos de la unión se recuperan las propiedades masivas de los materiales.

La función de trabajo φ se define de tal forma que $q\varphi$ es la energía necesaria para promocionar a un electrón desde el nivel de Fermi hasta el nivel de vacío. La diferencia de las funciones de trabajo de dos materiales es conocida como potencial de contacto V_{bi}. Así en la heterounión se cumple que:

$$V_{bi} = \varphi_2 - \varphi_1 \tag{1.1}$$

donde φ_2 y φ_1 son las funciones de trabajo de los semiconductores de banda prohibida estrecha y ancha respectivamente. La diferencia entre las funciones de trabajo puede expresarse como:

$$V_{bi} = \varphi_2 - \varphi_1 = \frac{\Delta E_c}{q} + \frac{k_B T}{q} ln \frac{n_{10} N_{c2}}{n_{20} N_{c1}} \tag{1.2}$$

donde n_{10} y n_{20} son las concentraciones en el equilibrio de los semiconductores y N_{c1} y N_{c2} sus densidades efectivas de estados.

Un factor que influye notablemente en la formación de las heteroestructuras es la tensión en las heterointerfaces. Las fronteras abruptas entre distintas capas semiconductoras se forman utilizando principalmente dos métodos: MBE (*Molecular Beam Epitaxy*) y MOCVD (*Metal–Organic Chemical*

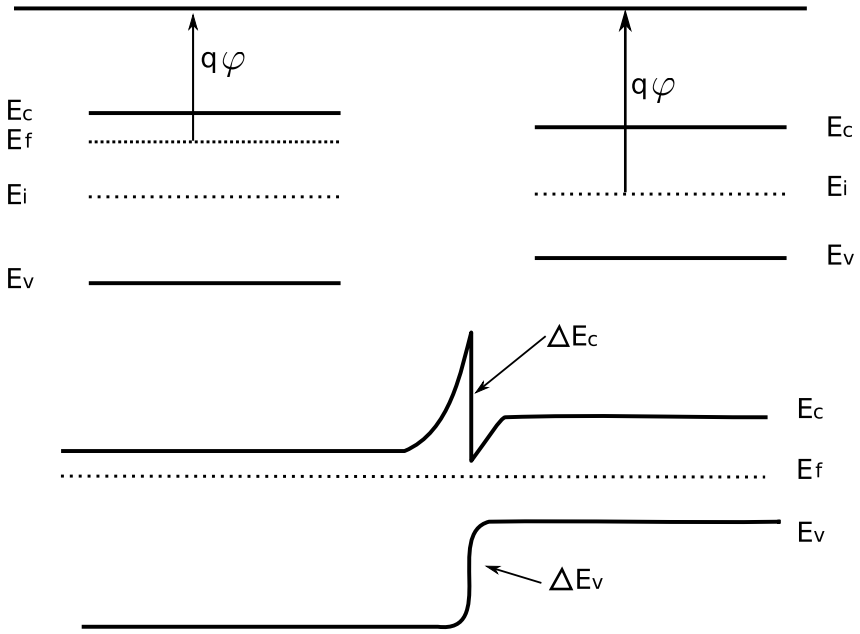


Figura 1.3: Ejemplo de diagrama de bandas de dos materiales semiconductores por separado y de la heterounión resultante de su unión.

Vapor Deposition). Estos métodos se basan en el crecimiento epitaxial sobre un sustrato con una constante de red adecuada. La tecnología de crecimiento cristalino permite actualmente crear capas muy finas de materiales semiconductores heterogéneos, lo que ha posibilitado el desarrollo de heteroestructuras. Este crecimiento es posible aún en el caso de que las constantes de red de los materiales sean diferentes. En ese caso, la capa fina adoptará la constante de red del material que la rodea, teniendo que expandirse o contraerse para adaptarse, abandonando para ello su forma cristalina masiva. Se dice entonces que las constantes de red se acomodan por tensión. A partir de un cierto grosor no se consigue el alineamiento de las constantes de red, por lo que cada capa mantendrá su constante de red inicial. Esto provocará la formación de dislocaciones en la superficie de separación que pueden degradar significativamente la fiabilidad y el rendimiento del dispositivo. Este efectos pueden observarse, para dos capas de distinto espesor, en la figura 1.4.

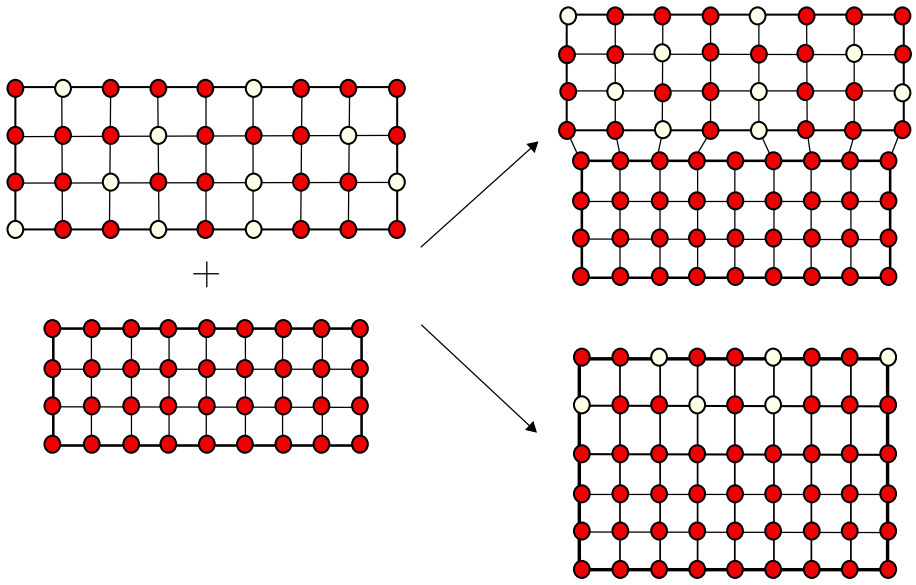


Figura 1.4: Posibles crecimientos resultantes de la unión de dos capas de materiales cristalinos. En la parte inferior derecha de la figura la capa epitaxial que se crece es lo suficientemente fina como para que se adapten las constantes de red, mientras que en la parte superior derecha de la figura esta capa es más ancha que el grosor crítico, lo que provoca la aparición de dislocaciones.

1.2. Transistores de efecto campo

Los transistores FET (transistores de efecto campo) son dispositivos unipolares que involucran principalmente el transporte de los portadores mayoritarios, ya sean electrones o huecos, en una capa paralela a la superficie. El más utilizado de los transistores de efecto campo es el MOSFET (transistor de efecto campo metal–óxido semiconductor). En estos transistores el semiconductor más empleado es el Si. Este material tiene la ventaja de ser oxidado fácilmente para formar SiO_2 de una manera altamente controlable y reproducible. La superficie de separación $Si - SiO_2$ se puede crear con una muy buena regularidad, produciendo muy pocos defectos. Esto permite que los MOSFETs de Si se puedan fabricar en grandes cantidades, siendo fácilmente integrables para formar circuitos a gran escala. Por otro lado, su principal limitación se encuentra en que el Si es un material de baja movilidad. La mayoría de los semiconductores compuestos (GaAs, InP, etc.)

tienen movilidades mayores que la del silicio pero en la actualidad, a nivel comercial, no son utilizados en la fabricación de MOSFETs a causa de la dificultad de encontrar aislantes válidos [19, 20].

Una alternativa para lograr la acción de la puerta en un FET sin utilizar estructuras Metal–Óxido–Semiconductor se encuentra en el uso de barreras Schottky, lo que da lugar a los dispositivos conocidos como MESFETs. Para su construcción se pueden utilizar semiconductores compuestos puesto que no son necesarias capas de aislante. A diferencia de los dispositivos MOSFET, en los que el flujo de corriente se produce próximo a la superficie en la capa de inversión formada entre el Si y el SiO_2, en los MESFETs el flujo de corriente tiene lugar en una zona del volumen del semiconductor, surgiendo los portadores de la concentración de dopado de esta zona. Como la corriente en un MESFET se da en la zona masiva, los portadores y los átomos donadores comparten el mismo espacio. Dado que el donador se encuentra ionizado, un centro fijo y cargado positivamente está presente en el cristal produciendo una dispersión culombiana bastante grande sobre los electrones libres conocida como dispersión por impurezas ionizadas. La importancia de este fenómeno depende de la separación espacial entre los centros de dispersión (en este caso los átomos donadores ionizados) y los electrones. Al incrementar la concentración de donadores, la dispersión por impurezas ionizadas aumenta, reduciéndose la movilidad electrónica en el dispositivo. Una técnica para reducir la dispersión culombiana consiste en separar físicamente los portadores de los átomos donadores. Esto es posible a través de la técnica de modulación del dopado, empleada en transistores de heteroestructura de efecto campo (HFETs) [21]. La modulación del dopado es una alternativa eficaz a las técnicas convencionales puesto que la separación espacial entre los portadores libres y los dopantes reduce la acción de la dispersión por impurezas ionizadas, lográndose aumentar la concentración de los portadores sin comprometer la movilidad.

En los HFETs las heteroestructuras se forman al poner en contacto un semiconductor dopado tipo n con anchura de banda prohibida elevada (p.ej. AlGaAs) y otro semiconductor intrínseco de anchura de banda prohibida reducida (p.ej. InGaAs). En la figura 1.5 se representan los diagramas de bandas de estos materiales por separado y en contacto. Inicialmente el nivel de Fermi en el caso del AlGaAs, al estar dopado tipo n, se encuentra más próximo a la banda de conducción que a la de valencia. Después de poner los dos materiales en contacto y alcanzarse el equilibrio, el nivel de Fermi se debe alinear a lo largo de toda la estructura. Para ello se deben transferir electrones desde la capa del AlGaAs a la del InGaAs, puesto que en el material de banda prohibida más estrecha hay estados de menor energía.

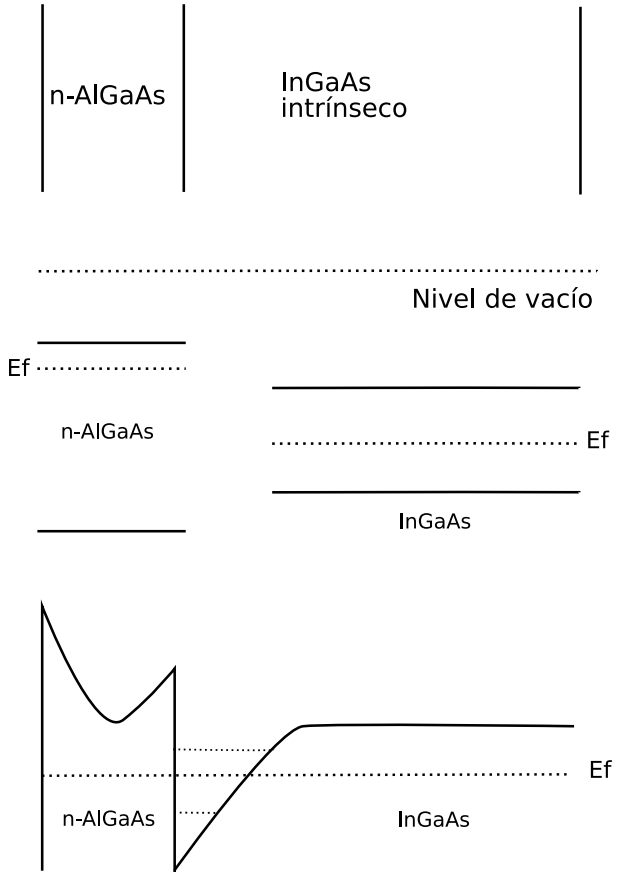


Figura 1.5: Estructura de bandas de una heterounión de n–AlGaAs e InGaAs intrínseco teniendo en cuenta modulación de dopado.

Esto provoca un aumento de la concentración electrónica dentro del InGaAs, sin que se incrementen las impurezas ionizadas dentro de esta zona. Así, los átomos donadores ionizados en el interior del AlGaAs tienen una carga neta positiva provocada por la transferencia de electrones a la capa de InGaAs. Aunque los átomos ionizados influyen sobre los electrones transferidos, la separación espacial entre ambos mitiga el efecto de la atracción culombiana entre ellos.

En esta situación, los electrones se encuentran confinados en una capa extremadamente fina, muy próxima a la heterounión, donde la energía de Fermi es superior a la energía de la banda de conducción. Esto confiere al canal una resistividad muy baja. La separación espacial de los donadores (cargados positivamente) y los electrones produce un campo eléctrico, dando lugar a una distorsión de la banda. Dependiendo del grado de curvatura de

la banda en la capa de InGaAs se puede dar cuantización espacial. Es decir, si la curvatura de la banda en la superficie es considerable se puede formar un pozo de potencial de dimensiones comparables a la longitud de onda de De Broglie de los electrones. Se producirán entonces niveles cuantizados de energía y el sistema se comportará como un gas de electrones bidimensional (2DEG).

Los HFETs utilizan esta técnica de modulación de dopado para alcanzar altas densidades de corriente, manteniendo al mismo tiempo una elevada movilidad de los portadores. Los principales ejemplos de HFETs son [22]:

1. HEMTs (High Electron Mobility Transistors), compuestos de capas de materiales con distintas energías de banda prohibida sobre un sustrato con la misma constante de red que las capas. Los portadores en estos dispositivos son proporcionados por una capa altamente dopada y el transporte tiene lugar en una capa adyacente sin dopar, dando lugar a una movilidad muy elevada. Dos ejemplos de los sustratos más utilizados en los dispositivos HEMT son el de InP y el de GaN.

2. PHEMTs (Pseudomorphic High Electron Mobility Transistors), compuestos de capas de materiales con distintas energías de banda prohibida sobre un sustrato con una constante de red similar a la de las capas pero no idéntica. Este tipo de dispositivos suelen tener una movilidad más elevada que la de los HEMTs y suelen estar basados en GaAs.

3. MHEMTs (Metamorphic High Electron Mobility Transistors), formados por capas con materiales de diferentes energías de banda prohibida, siendo sus constantes de red diferentes a la del sustrato. La tensión provocada por esta diferencia en las constantes de red es sobrellevada por una capa buffer especialmente diseñada. Estos dispositivos permiten mayor flexibilidad en el diseño y alcanzan rendimientos más elevados. Generalmente se fabrican a partir de sustratos de GaAs para aprovechar la mayor madurez de los materiales y las tecnologías de procesamiento.

1.3. Funcionamiento de los dispositivos de heteroestructura de efecto campo

En la estructura básica de un dispositivo de heteroestructura de efecto campo se crece epitaxialmente una capa de un semiconductor compuesto perteneciente al grupo III–V dopado tipo n sobre otro intrínseco. La variante

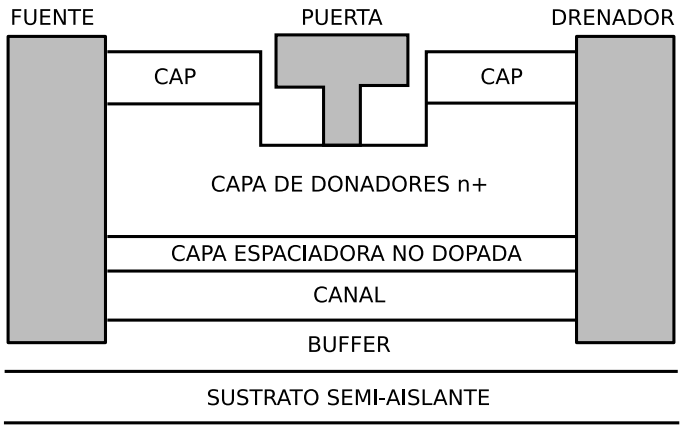


Figura 1.6: Estructura epitaxial básica de un dispositivo de heteroestructura de efecto campo.

más sencilla de la estructura básica de este tipo de transistores, representada en la figura 1.6, consiste en introducir una capa espaciadora no dopada entre las capas de los materiales dopado tipo n e intrínseco. De esta forma se consigue reducir la dispersión por impurezas ionizadas además de incrementar la movilidad electrónica. El grosor de esta capa se encuentra típicamente entre los 20 y los 50 Å. Cuanto más gruesa sea la capa espaciadora mayor realce de la movilidad electrónica se produce, pero al mismo tiempo se reduce la densidad de los portadores, efecto no deseable porque implica un descenso de la transferencia electrónica.

A través de la modulación del dopado se producen los portadores de carga sin por ello dopar el semiconductor intrínseco. La concentración total de carga es dependiente de la tensión de puerta y del modo de funcionamiento del dispositivo, que se puede encontrar en acumulación o en vaciamiento. Los dispositivos en modo acumulación están en corte cuando la tensión de puerta V_G es nula, por lo tanto, en el equilibrio no tenemos canal. Si el canal es tipo n es necesaria una cierta tensión de puerta positiva para inducir el canal. Por el contrario, en los dispositivos en modo vaciamiento existe canal para una tensión de puerta nula, en este caso será necesaria una tensión de puerta negativa para vaciar el canal y cortar el dispositivo.

Para examinar la situación dentro de la estructura en función de la tensión aplicada a la puerta, partimos de una tensión de drenador V_D nula, para un dispositivo en modo acumulación. Cuando la tensión de puerta supera un determinado voltaje umbral V_T, en la heterounión se forma un capa de inversión que contiene electrones móviles. Naturalmente, cuanto mayor sea

la polarización de inversión, mayor será la cantidad de electrones presentes en esa capa, y mayor será la conductancia de la capa de inversión.

De cualquier modo, una vez que el gas de electrones 2D se induce en la superficie de separación de la heterounión, se establece un canal de conducción entre la fuente y el drenador. La aplicación de una tensión positiva en el drenador provocará un flujo de corriente en el dispositivo, que surgirá del movimiento de los electrones de la fuente al drenador en el gas 2D. Se pueden lograr velocidades y movilidades de los electrones muy elevadas para tensiones aplicadas de drenador muy reducidas.

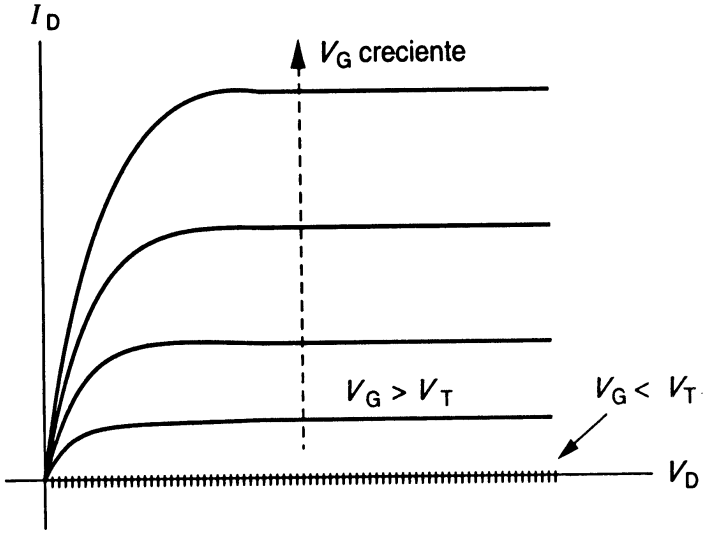


Figura 1.7: Curva característica $I_D - V_D$ para un dispositivo HEMT.

Cuando la tensión de drenador se aumenta poco a poco a partir de $V_D = 0$, el canal actúa como una simple resistencia y empieza a fluir corriente de drenador I_D proporcional a la tensión de drenador aplicada. Una vez que V_D aumenta por encima de unas pocas décimas de voltio, se produce un aumento de la zona de vaciamiento a lo largo del canal, que es más pronunciado en la zona de drenador, y por lo tanto disminuye la cantidad de portadores en la capa de inversión. El reducido número de portadores disminuye la conductancia del canal, que se refleja en una disminución de la pendiente de la curva característica $I_D - V_D$. El descenso en el número de portadores es más marcado en las proximidades del drenador, hasta que llegado un momento la capa de inversión desaparece en esta zona, produciéndose entonces el estrangulamiento del canal. Esto ocurre cuando la tensión de drenador es igual a $V_{D,sat}$. En este momento la pendiente de la

curva característica $I_D - V_D$ es nula y el dispositivo entra en zona de saturación. A partir de este momento I_D se mantiene constante para voltajes de drenador superiores a $V_{D,sat}$. La curva característica $I_D - V_D$ se representa en la figura 1.7, en la que podemos ver claramente las distintas regiones de funcionamiento.

Se define un potencial de superficie ψ_s tal que $q\psi_s$ es la diferencia de energía entre las bandas de conducción del semiconductor intrínseco en la zona masiva y en la heterointerfaz. Cuando se aplica una tensión en el drenador la curvatura de las bandas será diferente cerca del drenador o cerca de la región de fuente. Al aplicar tensiones de puerta y drenador positivas, cerca del drenador el potencial de superficie es menos positivo que cerca de la fuente, lo que hace que la curvatura de las bandas sea más pronunciada cerca de la región de fuente, haciendo que en esta zona el pozo sea más estrecho. Esto lleva a una diferencia en las sub–bandas de energía entre las dos regiones, puesto que la estructura energética de los electrones cambiará constantemente entre la fuente y el drenador.

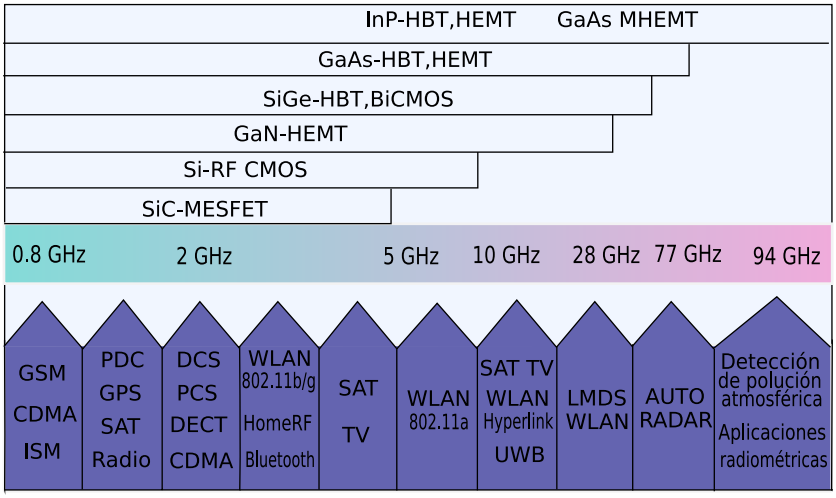


Figura 1.8: Rango de frecuencias de aplicación de los dispositivos semiconductores actuales.

1.4. Aplicaciones y utilidades de los transistores de heteroestructura de efecto campo

Los dispositivos de efecto campo basados en heteroestructuras son principalmente utilizados en el rango de las ondas milimétricas para aplicaciones

inalámbricas. En la actualidad el grupo IV de semiconductores (como por ejemplo Si y SiGe) dominan en las aplicaciones por debajo de los 10 GHz y los semiconductores compuestos de los grupos III–V por encima de los 10 GHz. El rango de frecuencias en el que se centra la competición entre los semiconductores elementales y compuestos cambia con el tiempo, moviéndose hacia frecuencias más altas. No obstante, es poco probable que el SiGe reemplace a los dispositivos de los grupos III–V en aplicaciones donde se requiera un ruido muy bajo o alta ganancia de potencia. En la figura 1.8 se muestra el rango de frecuencias de aplicación de los diferentes dispositivos actuales. Las fronteras que se presentan en esta figura no están tan claramente definidas en la actualidad, sino que son difusas y cambiantes con el tiempo. Esto es debido a que los consumidores del mercado de las comunicaciones inalámbricas están fuertemente influenciados por el coste, siendo este uno de los factores determinantes para la elección de la tecnología a utilizar. En el futuro, el eje de frecuencias de la figura perderá su importancia a la hora de definir fronteras entre tecnologías, puesto que se espera que la mayoría de los dispositivos citados en la figura alcancen frecuencias de operación muy elevadas. Así, las fronteras futuras estarán más dominadas por otros parámetros tales como el ruido, la potencia o la linealidad.

Ejemplos de áreas de aplicación de los dispositivos de efecto campo basados en heteroestructuras son, entre otros, redes inalámbricas de área local, redes personales inalámbricas, telefonía móvil, sistemas de radio–frecuencia, radares, aplicaciones radiométricas (como por ejemplo detección medioambiental de la polución o la monitorización no invasiva de la actividad sub–celular), etc.

1.5. Resumen

En este capítulo se han introducido los dispositivos HEMT. Estos transistores de efecto campo están basados en heteroestructuras de materiales semiconductores. Para facilitar su comprensión, en los diferentes apartados que forman el capítulo se han tratado de responder a las siguientes preguntas:

- ¿Qué es una heteroestructura de semiconductores y cómo se forma?
- ¿Qué es un dispositivo de efecto campo?
- ¿Cómo funciona un dispositivo de efecto campo basado en heteroestructuras?

- ¿Cuales son los principales usos y aplicaciones de los dispositivos de efecto campo basados en heteroestructuras?

Capítulo 2

Simulación de dispositivos semiconductores

El uso de herramientas software en el desarrollo de nuevos procesos o dispositivos electrónicos permite obtener de forma eficiente información de la física de los procesos de fabricación, del comportamiento de los dispositivos y del rendimiento de los circuitos. De esta forma se reducen de forma significativa los costes de desarrollo de los nuevos circuitos integrados.

La simulación numérica de dispositivos semiconductores se basa en desarrollar un programa informático capaz de predecir su comportamiento físico. Ejemplos de simuladores comerciales son, por ejemplo, Taurus–Medici [23] y Sentaurus [24] de SYNOPSYS, ATLAS [25] de Silvaco, BIPOLE3 [26] de BIPSIM y APSYS [27] de Crosslight. Estos programas han sido desarrollados para trabajar con diferentes tipos de dispositivos, por lo que presentan características distintas e implementan diferentes aproximaciones a la resolución del problema.

La simulación de dispositivos semiconductores puede hacerse a distintos niveles de complejidad computacional en función del modelo empleado. Así, en este capítulo inicialmente se introducen las principales técnicas de simulación utilizadas en el campo de los dispositivos semiconductores. A continuación se describen las ecuaciones que componen el modelo de arrastre–difusión, puesto que es este el modelo matemático implementado en el simulador tridimensional de dispositivos HEMT utilizado en este trabajo. Para ello se explica en detalle el proceso de obtención de la ecuación de Poisson y de continuidad de huecos y electrones, así como los cálculos necesarios para la obtención de los valores de la concentración de portadores, el factor de generación–recombinación y las condiciones de contorno del problema. Seguidamente, en esta sección se introduce el concepto de factor de escala y

se muestra el escalado realizado de las principales variables físicas del proceso de simulación. Por último se introduce el método de elementos finitos y el proceso de discretización de las ecuaciones pertenecientes al modelo de arrastre–difusión.

2.1. Técnicas de simulación de dispositivos semiconductores

En la actualidad existe una jerarquía bien establecida de técnicas que pueden ser utilizadas en la simulación de dispositivos semiconductores modernos [28]. En la figura 2.1 se presenta una escalera de jerarquía que los clasifica en base a dos parámetros, la complejidad computacional y el tiempo de simulación.

En la parte inferior de la escalera de jerarquía están los modelos compactos que contienen muy poca información sobre la física presente en el sistema y en general imitan el comportamiento del dispositivo utilizando aproximaciones analíticas y parámetros empíricos. Estos modelos requieren poco tiempo de cálculo pero su validez es limitada puesto que ignoran la naturaleza distribuida de los parámetros y la compleja geometría del dispositivo.

El siguiente nivel de técnicas de simulación es la aproximación de arrastre–difusión (D–D, drift–diffusion) a la ecuación de transporte de Boltzmann (BTE) [29]. La aproximación D–D considera sólo los primeros dos momentos de la BTE, la ecuación de continuidad de corriente y la ecuación de conservación del momento. Estas ecuaciones están acopladas a la ecuación de Poisson por el potencial electrostático. La aproximación D–D incluye una relación local entre la velocidad y el campo eléctrico y no puede representar apropiadamente efectos de transporte fuera del equilibrio.

El siguiente nivel por encima del modelo D–D es la aproximación hidrodinámica a la ecuación de transporte de Boltzmann. En este caso se incluye el tercer momento de la BTE, la ecuación de conservación de la energía, lo que hace más compleja la ecuación de conservación del momento. Esto permite el tratamiento de efectos fuera del equilibrio ya que incluye una relación no local entre el campo eléctrico y la velocidad.

El siguiente nivel consiste en utilizar una técnica Monte Carlo para la resolución de la BTE. En esta técnica un conjunto de partículas evoluciona a través de un espacio de aceleración real y de eventos de dispersión escogidos aleatoriamente. Estos métodos requieren un elevado tiempo de computación, por lo que son utilizados principalmente para dispositivos de dimensiones

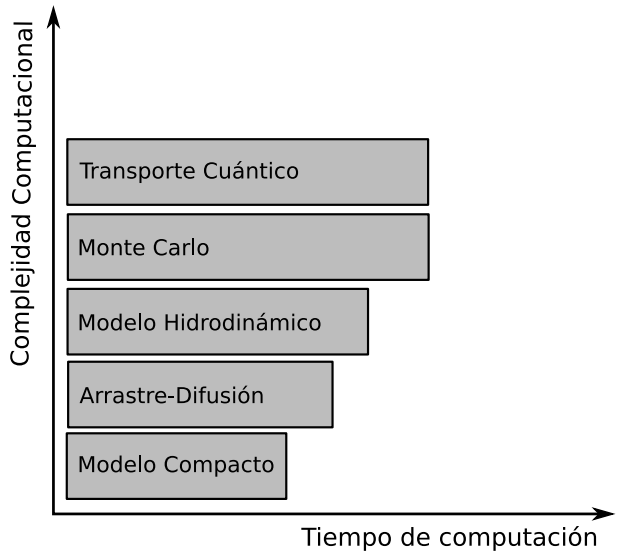


Figura 2.1: Escalera de jerarquía para diferentes métodos de simulación en función de su complejidad computacional y su tiempo de computación.

muy pequeñas en los que los modelos de arrastre–difusión no son válidos.

El nivel más elevado en la jerarquía está ocupado por las aproximaciones de transporte cuántico que utilizan las ecuaciones acopladas de Poisson y de Schödinger independiente del tiempo, la matriz de densidad o la función de distribución Wigner. Todas estas técnicas son extremadamente costosas computacionalmente. Otra técnica que está ganando popularidad es el uso del formalismo de las funciones de Green fuera del equilibrio (NEGF Non–Equilibrium Green Functions), que permite la inclusión de la dispersión en la formulación del transporte cuántico. A continuación se describen brevemente las diferentes técnicas de simulación numéricas, destacando sus puntos fuertes y sus limitaciones para el estudio de fluctuaciones de parámetros intrínsecos.

2.1.1. Arrastre–difusión

El modelo de arrastre–difusión es el más simple empleado en simulaciones numéricas multidimensionales [30, 31]. Utiliza los dos momentos más bajos de la ecuación de transporte de Boltzmann. Las densidades de corriente de electrones J_n y huecos J_p se obtienen por medio de la suma de dos componentes, una de arrastre gobernada por el campo eléctrico y otra de difusión dirigida por el gradiente de la densidad de portadores. Esta aproximación

no tiene en cuenta la temperatura de los portadores y de forma estricta es solamente válida para campos en los que la velocidad de los portadores esté directamente relacionada con el campo eléctrico. Sin embargo, la validez de la aproximación de arrastre–difusión puede extenderse empíricamente introduciendo modelos de movilidad dependientes del campo eléctrico, lo que permite el uso de campos eléctricos más elevados en el proceso de simulación. Aunque incluso con esta extensión el sistema arrastre–difusión sólo funciona en el cuasi–equilibrio, donde el campo eléctrico varía lentamente y la velocidad está relacionada localmente con el campo. La principal ventaja de la aproximación de arrastre–difusión es su menor coste computacional en comparación con las otras técnicas de simulación, lo que la hace adecuada para realizar simulaciones estadísticas tridimensionales a gran escala, necesarias para la caracterización del impacto de diversas fuentes de fluctuaciones de parámetros intrínsecos. Además, para las ecuaciones de arrastre–difusión hay una gran variedad de técnicas numéricas plenamente desarrolladas.

2.1.2. Modelo hidrodinámico

Con el escalado de los dispositivos a dimensiones muy reducidas se produce una disparidad entre el rápido descenso de las dimensiones físicas y la mucho más lenta reducción de la tensión aplicada. Esto provoca un aumento en los campos en el interior de los dispositivos. Para tener en cuenta estos campos elevados el modelo de arrastre–difusión introduce modelos de movilidad dependientes del campo que relacionan de un modo local la velocidad de los portadores con la componente del campo en la dirección del flujo de corriente. Esta aproximación ignora fenómenos de transporte no locales en los que la velocidad de portadores en un punto está determinada por la distribución del campo a lo largo del camino de corriente. Una aproximación que supera estas limitaciones es el modelo hidrodinámico [32, 33] al considerar el tercer momento de la ecuación de transporte de Boltzmann. En este caso la corriente tiene un término adicional proporcional al gradiente de la temperatura de los electrones y además se añade al sistema una ecuación de balance energético. Un problema que supone añadir momentos más elevados de la BTE es la estabilidad numérica de la solución, lo que lleva a un gran aumento de la complejidad computacional.

2.1.3. Monte Carlo

Un método alternativo de simular el transporte de portadores que no implica la discretización de la BTE o de sus momentos es la aproximación Monte Carlo (MC) [34, 35]. El método MC es una técnica estocástica

que utiliza números aleatorios para obtener una aproximación estadística a la solución exacta de la BTE. El método Monte Carlo traza las trayectorias clásicas de los portadores en un dispositivo simulado por medio de la dispersión de cada partícula después de un período de vuelo libre determinado estocásticamente a través de las tasas de dispersión acumulativas. Estas tasas de dispersión se calculan cuánticamente e incluyen, entre otras, interacciones electrón–fonón, electrón–donador y electrón–electrón. Una simulación Monte Carlo es por lo tanto una serie de vuelos libres intercalados con eventos de dispersión que cambian el momento y posiblemente la energía de la partículas. El movimiento de las partículas está acoplado a la solución de la ecuación de Poisson para permitir el cálculo actualizado de la fuerza que las dirige. La aproximación Monte Carlo es ampliamente usada en la simulación de dispositivos semiconductores en general [36, 37] y en particular para la simulación de HEMTs [38, 39]. Sin embargo el gran número de partículas que deben ser simuladas, la enorme cantidad de números aleatorios y el acoplamiento con la ecuación de Poisson hacen que este método sea muy costoso computacionalmente.

2.1.4. Transporte cuántico

En la cima de la jerarquía de los métodos de simulación están las técnicas de transporte cuántico. La modelización del transporte dentro de un sólido usando para ello una aproximación cuántica completa es muy costosa computacionalmente y poco práctica para simulaciones realistas de dispositivos. Una técnica que está ganando popularidad en el campo del transporte cuántico es la aproximación de las funciones de Green fuera del equilibrio (NEGF) [40, 41]. Utiliza un método matemático conocido como Funciones de Green para obtener la solución de un Hamiltoniano independiente del tiempo. La función de Green, a una energía dada, tiene dos entradas que pueden ser relacionadas con dos posiciones del espacio real permitiendo simular áreas de un transistor. Esta función considera la influencia de una perturbación que tiene lugar en una entrada sobre la otra entrada, y tiene, en teoría, la habilidad de modelar las propiedades físicas del sistema, tales como, la densidad electrónica, la densidad de corriente y densidad de estados. Una de las principales limitaciones del uso de NEGF es su elevadísimo coste computacional.

Como se acaba de comentar, las simulaciones completamente mecano–cuánticas son prohibitivas en términos de tiempo computacional, pero es posible incluir efectos cuánticos en simulaciones clásicas en lo que se conoce como correcciones cuánticas [42], con un coste computacional mucho menor.

Estas correcciones permiten considerar efectos de confinamiento cuántico y ciertos aspectos del efecto túnel. Las correcciones cuánticas desempeñan un papel muy importante conforme los dispositivos son escalados más agresivamente a dimensiones nanométricas. Los dos métodos más conocidos para incluir correcciones cuánticas en simulaciones clásicas de dispositivos son la aproximaciones *density gradient* [43] y potencial efectivo [44].

La aproximación *density gradient* introduce un potencial cuántico que proporciona un término adicional de arrastre a la expresión de la densidad de corriente. Este potencial cuántico es proporcional a la segunda derivada de la densidad de portadores y aleja a los electrones de variaciones pronunciadas en el potencial clásico.

La técnica del potencial efectivo representa a los portadores por medio de un paquete de ondas gaussiano de dispersión mínima. El potencial efectivo está relacionado con el potencial a través de una integral de convolución. El suavizado del potencial asociado con la operación de convolución representa los efectos mecánico–cuánticos que alejan a la concentración de electrones de la interfaz y reducen picos bruscos en el potencial.

2.2. Modelo matemático: aproximación arrastre–difusión

El simulador 3D de dispositivos HEMT desarrollado en este trabajo se basa en el modelo de arrastre–difusión. Las ecuaciones básicas a resolver en este modelo son la ecuación de Poisson y las ecuaciones de continuidad de huecos y de electrones. Estas ecuaciones que describen el comportamiento eléctrico del semiconductor se determinan a partir de las ecuaciones de Maxwell.

Se considera que un dispositivo semiconductor real ocupa un dominio cerrado y conectado en R^3, denominado Ω. Las ecuaciones de Maxwell definen la evolución del campo electromagnético en un medio arbitrario [47]:

$$\nabla \times H = J + \frac{\partial D}{\partial t} \tag{2.1}$$

$$\nabla \times E = -\frac{\partial B}{\partial t} \tag{2.2}$$

$$\nabla \cdot D = \varrho \tag{2.3}$$

$$\nabla \cdot B = 0 \tag{2.4}$$

donde E y D son los vectores campo y desplazamiento eléctrico, H y B el campo y la inducción eléctrica, J es el vector densidad de corriente, y ϱ es

la densidad de carga espacial. Es preciso además considerar una variable espacial en R^3, $x = (x_0, x_1, x_2)$ y una temporal, t, donde $t \in \mathrm{R}^+$.

Junto con estas ecuaciones, se considera que existe una relación de proporcionalidad entre los vectores campo y desplazamiento eléctricos, es decir:

$$D = \epsilon E \tag{2.5}$$

donde ϵ es la permitividad del medio. En un medio arbitrario ϵ es un tensor con dependencia temporal, de forma que E y D no son paralelos. Sin embargo, en las aplicaciones más comunes, es posible considerar los medios homogéneos e isótropos y tratar ϵ como un escalar independiente del tiempo, que es lo que ocurre en nuestro caso.

2.2.1. Ecuación de Poisson

Para la deducción de la ecuación de Poisson se parte de las ecuaciones de Maxwell. Como la inducción magnética tiene divergencia cero, puede representarse siempre como el rotacional de un potencial vector A. Por lo tanto, la ecuación 2.4 puede satisfacerse si se expresa la inducción magnética como:

$$B = \nabla \times A \tag{2.6}$$

donde para un B dado, el vector A no es único. Si se sustituye en la ecuación 2.2 el valor de B por el correspondiente a la ecuación 2.6 se obtiene la siguiente expresión:

$$\nabla \times \left(E + \frac{\partial A}{\partial t} \right) = 0 \tag{2.7}$$

Si el rotacional de un vector es cero, se puede expresar dicho vector como el gradiente de un campo escalar ψ. Así, se obtiene:

$$E + \frac{\partial A}{\partial t} = -\nabla \psi \tag{2.8}$$

Reemplazando E en la ecuación 2.5 e introduciendo el resultado en 2.3, considerando ε constante, se obtiene:

$$\varepsilon \frac{\partial \nabla \cdot A}{\partial t} + \varepsilon \nabla^2 \psi = -\varrho \tag{2.9}$$

El teorema de Helmholtz indica que el vector A no estará completamente determinado hasta que no se hayan especificado tanto $\nabla \times A$ como ∇A. Hasta ahora sólo se ha especificado el rotacional a través de la ecuación 2.6 y en consecuencia, existe libertad para escoger la divergencia de A de

la manera más conveniente. Si se quiere que las ecuaciones permanezcan invariantes bajo la transformación de Lorentz, es necesario requerir que:

$$\nabla \cdot A = -\frac{1}{c^2}\frac{\partial \psi}{\partial t} \tag{2.10}$$

donde c es la velocidad de la luz. A este requisito se le conoce con el nombre de *condición de Lorentz*. Insertando esta condición en la ecuación 2.9 se obtiene la ecuación de onda para el potencial:

$$-\frac{\varepsilon}{c^2}\frac{\partial^2 \psi}{\partial^2 t} + \varepsilon \nabla^2 \psi = -\varrho \tag{2.11}$$

Usualmente se considera que la velocidad de la luz es muy grande en comparación con todas las velocidades que son relevantes en el dispositivo. Esto es equivalente a considerar $\nabla \cdot A = 0$. De esta forma, se puede despreciar el primer término de la ecuación 2.11, obteniéndose así la *ecuación de Poisson* que define el comportamiento del potencial eléctrico:

$$\varepsilon \nabla^2 \psi = -\varrho \tag{2.12}$$

En el interior Ω de un semiconductor la densidad espacial de carga viene dada por:

$$\varrho = q(p - n + C) \tag{2.13}$$

donde q es la unidad de carga elemental, p la concentración de huecos, n la concentración de electrones y C el perfil predefinido de impurezas eléctricamente activas. Si se supone que todas las impurezas están ionizadas a 300 K se tiene:

$$C = N_D^+ - N_A^- \tag{2.14}$$

donde N_D^+ y N_A^- representan respectivamente, la concentración de impurezas dadoras y aceptoras.

Por lo tanto, la ecuación de Poisson en un semiconductor genérico puede expresarse como sigue:

$$\varepsilon \nabla^2 \psi = q(n - p - C), \qquad \forall x \in \Omega \tag{2.15}$$

2.2.2. Ecuaciones de continuidad de huecos y electrones

Volviendo a las ecuaciones de Maxwell, se observa que aplicando el operador divergencia a la ecuación 2.1 y usando la ecuación 2.3 se obtiene:

$$0 = \nabla J + \frac{\partial \varrho}{\partial t} \tag{2.16}$$

ya que la divergencia de un rotacional es nula.

Se puede expresar la densidad de corriente J en función de dos términos, uno asociado con la corriente de electrones, J_n, y otro con la de huecos, J_p, quedando de esta forma $J = J_n + J_p$. Así, considerando que el perfil de impurezas es invariante en el tiempo y usando 2.16 y 2.13, se obtiene que en el interior Ω del semiconductor se verifica la siguiente expresión:

$$-\nabla J_p - q\frac{\partial p}{\partial t} = \nabla J_n - q\frac{\partial n}{\partial t} \tag{2.17}$$

Es posible obtener una ecuación para la densidad de corriente de electrones y otra equivalente para la de huecos si se hacen ambas partes de 2.17 iguales a una cantidad, que se denotaran por qR:

$$\nabla J_n - q\frac{\partial n}{\partial t} = qR \tag{2.18}$$

$$\nabla J_p + q\frac{\partial p}{\partial t} = -qR \tag{2.19}$$

El término R puede interpretarse físicamente como una función que describe la generación o recombinación neta de electrones y huecos. Valores de R positivos implican que la recombinación de pares electrón–hueco prevalece sobre la generación de los mismos en el semiconductor. Valores negativos de R indican que la generación predomina.

Si se está bajo condiciones isotérmicas y si el transporte es por arrastre–difusión, entonces se podrían considerar las densidades de corriente de electrones y huecos proporcionales a los gradientes de los cuasipotenciales de Fermi de los electrones y huecos (ϕ_n y ϕ_p):

$$J_n = -q\mu_n n\nabla(\phi_n) \tag{2.20}$$

$$J_p = -q\mu_p p\nabla(\phi_p) \tag{2.21}$$

donde μ_n y μ_p son las movilidades de electrones y de huecos respectivamente. Físicamente, las movilidades están relacionadas con los tiempos medios de relajación de electrones y huecos, τ_n^r y τ_p^r, que representan el tiempo medio entre dos procesos consecutivos de dispersión. Por ello la movilidad puede considerarse como una medida de la facilidad de movimiento de los portadores en el cristal, por lo que es evidente que las movilidades serán inversamente proporcionales a la cantidad de colisiones. En el simulador el valor de la movilidad se fija en función del tipo de material y de otros efectos, como pueden ser, campos elevados, impurezas, etc.

2.2.3. Concentración de portadores

En este apartado se tratan de obtener unas expresiones para las concentraciones de portadores, n y p, en función del potencial electrostático ψ y de los cuasipotenciales de Fermi de electrones y de huecos ϕ_n y ϕ_p.

En un semiconductor con diferentes bandas (o valles) no equivalentes que intervengan en el transporte, la concentración de electrones en cada una de esas bandas, $n_j(E)$, depende del número de estados por unidad de volumen del cristal con esa energía $g_j(E)$, que es la función de densidad de estados, y de la probabilidad de que esos estados estén ocupados por un electrón $f_e(E)$, que viene dada por la función de Fermi–Dirac. Es posible escribir la concentración de electrones en la banda j como:

$$n_j = \int_{E_{cj}}^{E_{sup_j}} f_e(E) g_j(E) dE \tag{2.22}$$

donde E_{cj} es el mínimo de energía de la banda j y E_{sup_j} el nivel máximo de energía de la banda de conducción. Teniendo en cuenta que el valor del integrando anterior disminuye muy rápidamente a medida que la energía aumenta, y que es esencialmente cero para energías apenas pocos KT por encima de E_c, es posible extender el límite superior de la ecuación anterior a ∞ y seguir obteniendo un valor similar de la integral. Dicha integral quedaría como,

$$n_j = \int_{E_{cj}}^{\infty} f_e(E) g_j(E) dE \tag{2.23}$$

La función de Fermi–Dirac para los electrones es:

$$f_e(E) = \frac{1}{1 + \exp^{\frac{E - E_{Fn}}{KT}}} \tag{2.24}$$

donde E_{Fn} es el cuasinivel de Fermi para los electrones. Este valor se relaciona con el cuasipotencial de Fermi para los electrones ϕ_n mediante la expresión:

$$\nabla E_{Fn} = -\nabla q \phi_n \tag{2.25}$$

Si se tiene presente la no parabolicidad de la banda j, mediante un factor de no parabolicidad B_j, la expresión de la densidad de estados $g_j(E)$ se puede escribir como:

$$g_j = \frac{\sqrt{2} m_{dj}^{3/2}}{\pi^2 \hbar^3} \left[(E - E_{cj})^{1/2} + B_j (E - E_{cj})^{3/2} \right] \tag{2.26}$$

siendo $\hbar = \frac{h}{2\pi}$ la constante reducida de Planck y m_{dj} la masa efectiva para la densidad de estados en la banda (o en el valle).

Sustituyendo las fórmulas anteriores en la concentración de portadores en la banda j se puede poner como:

$$n_j = N_{cj} \left[F_{1/2}(\eta_{cj}) + \frac{3}{2} KT B_j F_{3/2}(\eta_{cj}) \right] \tag{2.27}$$

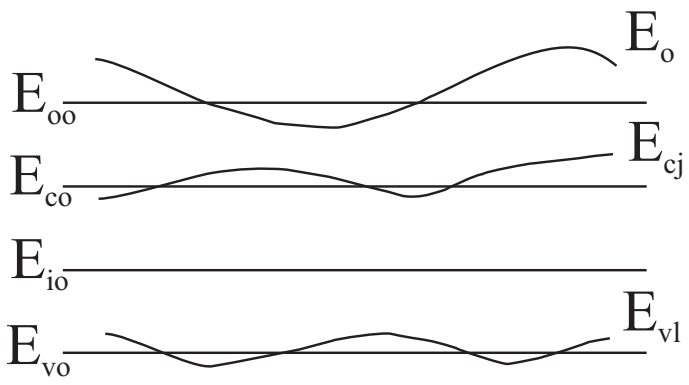


Figura 2.2: Niveles de energía del material de referencia junto con los niveles de energía de una región del semiconductor.

donde $F_{1/2}(\eta_{cj})$ y $F_{3/2}(\eta_{cj})$ son las integrales de Fermi–Dirac de orden 1/2 y 3/2, respectivamente.

En la ecuación 2.27, N_{cj} representa la densidad efectiva de estados. Este parámetro se puede relacionar con la masa efectiva para la densidad de estados por:

$$N_{cj} = 2\left(\frac{2m_{dj}KT\pi}{h^2}\right)^{3/2} \tag{2.28}$$

donde η_{cj} viene dado por la siguiente expresión:

$$\eta_{cj} = \frac{E_{Fn} - E_{cj}}{KT} \tag{2.29}$$

Se puede considerar por tanto, la concentración de electrones como la suma de las concentraciones para cada una de las bandas o valles:

$$n = \sum_j N_{cj}\left[F_{1/2}(\eta_{cj}) + \frac{3}{2}kTB_jF_{3/2}(\eta_{cj})\right] \tag{2.30}$$

En el modelo físico utilizado se mide el valor de la energía de los distintos niveles respecto a un nivel de energía constante que se toma como referencia. Este nivel de energía, igual a cero, coincide con el nivel de vacío de un material tomado a su vez como referencia. Se utiliza el subíndice o para las constantes físicas y los niveles de energía constantes del material tomado como referencia, que no tiene porque corresponder con ninguno de los puntos de las regiones presentes en el dispositivo, pero que deben ser coherentes entre sí.

En la figura 2.2 se muestran los niveles de energía del material de referencia E_{oo}, E_{co}, E_{io} y E_{vo} junto con el nivel de vacío, E_o, el nivel mínimo

de la banda de conducción j, E_{cj}, y el máximo de una banda de valencia l, E_{vl}, en el interior de cualquier región del dispositivo.

En el diagrama de energías de esta figura, la energía del mínimo de la banda de conducción del material de referencia, E_{co}, y del máximo de la banda de valencia, E_{vo}, se pueden expresar en función de su afinidad electrónica, χ_o, y de la anchura de su banda prohibida, E_{go}. Por lo tanto, $E_{co} = -\chi_o$ y $E_{vo} = -(\chi_o + E_{go})$.

La concentración intrínseca del material de referencia, elegido generalmente en su estado intrínseco, n_{io}, está relacionada con las densidades efectivas de estados de las bandas de valencia y de la banda de conducción y con la anchura de la banda prohibida mediante la ecuación:

$$n_{io} = \sqrt{N_{co}N_{vo}} \exp\left(-\frac{E_{go}}{2KT}\right) \tag{2.31}$$

siendo N_{co} la densidad efectiva de estados en la banda de conducción, y N_{vo} la densidad efectiva de estados en la banda de valencia del material de referencia.

Por otro lado, el nivel intrínseco del material de referencia, E_{io}, depende de la densidad efectiva de estados y de la concentración intrínseca. Se puede escribir como:

$$E_{io} = E_{co} + KT \ln\left(\frac{n_{io}}{N_{co}}\right) \tag{2.32}$$

Para obtener las expresiones de las concentraciones de portadores n y p, en primer lugar, se relacionan los niveles de energía de las bandas en un semiconductor, esto es, nivel de vacío, E_o, mínimo de la banda de conducción j, E_{cj} y máximo de la banda de valencia l, E_{vl} y los cuasiniveles de Fermi para los electrones y los huecos, E_{Fn} y E_{Fp}, con el potencial electrostático, ψ, los cuasipotenciales de Fermi, ϕ_n y ϕ_p y los niveles de energía del material de referencia.

El nivel de vacío del semiconductor se obtiene a partir del potencial electrostático por:

$$E_o = -q\psi \tag{2.33}$$

El nivel de energía mínimo de la banda de conducción j es:

$$E_{cj} = -q\psi - \chi_j \tag{2.34}$$

y el máximo de la banda de valencia l:

$$E_{vl} = -q\psi - \chi_l - E_{gl} \tag{2.35}$$

El cuasinivel de Fermi para los electrones es:

$$E_{Fn} = E_{io} - q\phi_n \tag{2.36}$$

y el cuasinivel de Fermi para los huecos:

$$E_{Fp} = E_{io} - q\phi_p \tag{2.37}$$

Además, la ecuación 2.32 se puede expresar como:

$$\frac{n_{io}}{N_{co}\exp\left(\frac{E_{io}-E_{co}}{KT}\right)} = 1 \tag{2.38}$$

de tal manera que si se multiplica la parte derecha de la ecuación 2.30 por el factor anterior se obtiene la siguiente expresión:

$$n = n_{io}\exp\left(\frac{E_{co}-E_{io}}{KT}\right)\sum_j \frac{N_{cj}}{N_{co}}\left[F_{1/2}(\eta_{cj}) + \frac{3}{2}kTB_jF_{3/2}(\eta_{cj})\right] \tag{2.39}$$

Teniendo en cuenta la ecuación 2.29 y que $E_{co} = \psi_o$ y $E_{cj} = -q\psi - \chi_j$, se puede reescribir la ecuación anterior como:

$$\begin{aligned} n &= n_{io}\sum_j \frac{N_{cj}}{N_{co}}\frac{\left[F_{1/2}(\eta_{cj}) + \frac{3}{2}kTB_jF_{3/2}(\eta_{cj})\right]}{\exp(\eta_{cj})} \\ &\quad \exp\left(\frac{\chi_j - \chi_o}{KT}\right)\exp\left(\frac{q\psi + E_{Fn} - E_{io}}{KT}\right) \end{aligned} \tag{2.40}$$

Expresando el cuasinivel de Fermi de los electrones en función de su cuasipotencial de Fermi $E_{Fn} = E_{io} - q\phi_n$ se obtiene la siguiente expresión para n:

$$\begin{aligned} n &= \left\{n_{io}\sum_j \frac{N_{cj}}{N_{co}}\frac{\left[F_{1/2}(\eta_{cj}) + \frac{3}{2}kTB_jF_{3/2}(\eta_{cj})\right]}{\exp(\eta_{cj})}\right. \\ &\quad \left.\exp\left(\frac{\chi_j - \chi_o}{KT}\right)\right\}\exp\left(\frac{q\psi - q\phi_n}{KT}\right) \end{aligned} \tag{2.41}$$

Del mismo modo, se puede obtener una expresión similar para la concentración de huecos p:

$$\begin{aligned} p &= \left\{n_{io}\sum_l \frac{N_{vl}}{N_{vo}}\frac{\left[F_{1/2}(\eta_{vl}) + \frac{3}{2}kTB_jF_{3/2}(\eta_{vl})\right]}{\exp(\eta_{vl})}\right. \\ &\quad \left.\exp\left(-\frac{\chi_j - \chi_o + E_{gl} - E_{go}}{KT}\right)\right\}\exp\left(\frac{q\phi_p - q\psi}{KT}\right) \end{aligned} \tag{2.42}$$

donde el subíndice j o l se refiere a las diferentes bandas que pueden intervenir en la población de portadores.

El cuasipotencial de Fermi para los huecos se escribe como $q\phi_p = E_{io} - E_{Fp}$, y el parámetro $\eta_{vl} = \frac{E_{vl}-E_{Fp}}{KT}$.

En las ecuaciones anteriores correspondientes a las concentraciones de electrones 2.41 y huecos 2.42 se han indicado entre llaves las expresiones de las concentraciones intrínsecas efectivas de los electrones y huecos, n_{ien} y n_{iep}. Estas expresiones son:

$$n_{ien} = n_{io} \sum_j \frac{N_{cj}}{N_{co}} \frac{\left[F_{1/2}(\eta_{cj}) + \frac{3}{2} kTB_j F_{3/2}(\eta_{cj})\right]}{\exp(\eta_{cj})} \exp\left(\frac{\chi_j - \chi_o}{KT}\right) \quad (2.43)$$

$$\begin{aligned} n_{iep} &= n_{io} \sum_l \frac{N_{vl}}{N_{vo}} \frac{\left[F_{1/2}(\eta_{vl}) + \frac{3}{2} kTB_j F_{3/2}(\eta_{vl})\right]}{\exp(\eta_{vl})} \\ &\quad \exp\left(-\frac{\chi_j - \chi_o + E_{gl} - E_{go}}{KT}\right) \end{aligned} \quad (2.44)$$

Los valores de las concentraciones intrínsecas efectivas de electrones y huecos, n_{ien} y n_{iep}, permiten tener en cuenta los efectos de degeneración del semiconductor, de la variación de los parámetros con la composición, e incluso de la existencia de varias bandas o valles, incluyendo efectos de no parabolicidad de esas bandas. Además, se pueden comprobar que dichas concentraciones son función de la posición, dependen de las características de los materiales en cada punto, N_c, N_v, χ y E_g, y además, variarán a medida que se modifique la polarización del dispositivo puesto que, el potencial y los cuasipotenciales de Fermi de los portadores cambian al variar la polarización. Por tanto, n_{ien} y n_{iep}, tienen un papel primordial en el modelo presentado.

Las ecuaciones 2.43 y 2.44 permiten escribir las siguientes expresiones para las concentraciones de electrones y de huecos de forma compacta como:

$$n = n_{ien} \exp\left(\frac{q\psi - q\phi_n}{KT}\right) \quad (2.45)$$

$$p = n_{iep} \exp\left(\frac{q\phi_p - q\psi}{KT}\right) \quad (2.46)$$

Las ecuaciones 2.43 y 2.44, junto con las ecuaciones 2.45 y 2.46, constituyen el modelo utilizado para expresar la concentración de portadores en un punto cualquiera de un semiconductor degenerado o no degenerado, en el que pueden intervenir varias bandas o valles en la población de electrones y huecos, permitiendo además estudiar bandas no parabólicas.

Las ecuaciones anteriores para la concentraciones intrínsecas de electrones y huecos se pueden simplificar si cada tipo de portador se sitúa en una

sola banda de energía y si se prescinde de los factores de no parabolicidad de las bandas, entonces:

$$n_{ien} = n_{io}\frac{N_c}{N_{co}}\exp\left(\frac{\chi-\chi_o}{KT}\right)\frac{F_{1/2}\left(\frac{\chi-\chi_o}{KT}-\ln\left(\frac{N_{co}}{n_{io}}\right)+\frac{q\psi-q\phi_n}{KT}\right)}{\exp\left(\frac{\chi-\chi_o}{KT}-\ln\left(\frac{N_{co}}{n_{io}}\right)+\frac{q\psi-q\phi_n}{KT}\right)} \quad (2.47)$$

$$\begin{aligned} n_{iep} &= n_{io}\frac{N_v}{N_{vo}}\exp\left(-\frac{\chi-\chi_o+E_g-E_{go}}{KT}\right) \\ &\quad \frac{F_{1/2}\left(-\frac{\chi_j-\chi_o}{KT}-\frac{E_g-E_{go}}{KT}-\ln\left(\frac{N_{vo}}{n_{io}}\right)-\frac{q\phi_p-q\psi}{KT}\right)}{\exp\left(-\frac{\chi_j-\chi_o}{KT}-\frac{E_g-E_{go}}{KT}-\ln\left(\frac{N_{vo}}{n_{io}}\right)-\frac{q\phi_p-q\psi}{KT}\right)} \end{aligned} \quad (2.48)$$

2.2.4. Factor de generación–recombinación

En la derivación de las ecuaciones de continuidad (sección 2.2.2) se ha introducido el parámetro R que representa la diferencia entre las tasas de recombinación y generación de pares electrón–hueco. La recombinación ocurre cuando un electrón en la banda de conducción salta a la banda de valencia neutralizando un hueco mientras que, la generación sucede cuando un electrón de valencia pasa a la banda de conducción produciendo un hueco. La generación necesita energía mientras que la recombinación la libera.

Cuando un semiconductor extrínseco está en equilibrio térmico existe un equilibrio dinámico entre los procesos de generación y recombinación, con lo que $R = 0$ (principio del balance detallado). Si el equilibrio se rompe, por ejemplo al aplicar un potencial externo al semiconductor, las concentraciones de portadores se modifican respecto a sus valores en el equilibrio. Empiezan a producirse entonces varios procesos de generación–recombinación que tienden a restaurar el equilibrio de forma que el exceso o déficit de portadores es estabilizado si la perturbación se mantiene, o eliminado si la causa perturbadora desaparece. Así, si la perturbación produce un exceso de portadores la recombinación domina sobre la generación (es decir, $R > 0$) para eliminar dicho exceso, en caso contrario, la generación domina y $R < 0$.

La generación–recombinación es provocada por diversos tipos de procesos. Para cada tipo de proceso es necesario obtener una expresión que determine el valor de R en función de las variables características del dispositivo. Entre los principales procesos responsables de la generación–recombinación se destacan:

- Recombinación intrínseca también llamada directa, banda–a–banda o de transición de fotones.

- Recombinación extrínseca o de transición de dos partículas, de transición de fonones o Shockley–Read–Hall.
- Recombinación Auger o de transición de tres partículas.
- Recombinación superficial.

El primer mecanismo de generación–recombinación implica un proceso directo en el cual un electrón pasa de la banda de conducción a la de valencia (o viceversa) liberando (o absorbiendo) un fotón. Este proceso es importante para semiconductores de banda estrecha y semiconductores, como el GaAs, cuya estructura de bandas permite transiciones directas. La tasa de recombinación banda–banda, R_{BB}, es proporcional a la diferencia entre el producto de las concentraciones de portadores en ese momento y el que habría en equilibrio. A la constante de proporcionalidad se le conoce como coeficiente banda–banda, C_{BB}. Teniendo en cuenta que n_o y p_o representan las concentraciones de electrones y de huecos en equilibrio, esta recombinación se puede expresar como:

$$R_{BB} = C_{BB}(np - n_o p_o) \tag{2.49}$$

La forma más importante de generación–recombinación es la extrínseca. Este proceso tiene lugar a través de centros de recombinación (impurezas, imperfecciones de la red, etc). Así, la recombinación ocurre en dos pasos, en el primero un electrón de la banda de conducción es atrapado por el centro, y en el segundo dicho electrón pasa a la banda de valencia neutralizando un hueco. Este proceso es típicamente no–radiactivo. En él se libera (o absorbe en caso de generación) energía térmica o, equivalentemente, vibraciones de la red (fonones). El proceso de recombinación extrínseca se describe a través del denominado término de Shockley–Read–Hall:

$$R_{SRH} = \frac{np - n_o p_o}{\tau_p(n + n_T) + \tau_n(p + p_T)} \tag{2.50}$$

que relaciona la recombinación global con las concentraciones de portadores, siendo τ_n y τ_p los tiempos de vida media de electrones y huecos, respectivamente. El término n_T (p_T) representaría la concentración de electrones (de huecos) que habría en la banda de conducción (en la banda de valencia), si el nivel de Fermi en equilibrio coincidiese con el nivel de energía del centro de recombinación. Su valor es:

$$n_T = n_o \exp \frac{E_T - E_F}{KT} \tag{2.51}$$

$$p_T = p_o \exp \frac{E_F - E_T}{KT} \tag{2.52}$$

La recombinación Auger es un proceso no–radiactivo en el que la recombinación directa o vía centros ocurre simultáneamente a la colisión entre dos portadores del mismo tipo. Existen dos tipos de procesos Auger. En el primero la energía que se obtiene de la destrucción de un par electrón–hueco se emplea en aumentar la energía de un electrón de la banda de conducción mientras que, en el segundo esta energía se emplea en aumentar la de un hueco de la banda de valencia. Como en el primer proceso intervienen dos electrones la recombinación es proporcional a n^2p y en el segundo, como intervienen dos huecos ésta es proporcional a np^2 por lo tanto:

$$R_A = C_{An}(n^2p - n_o^2p_o) + C_{Ap}(np^2 - n_op_o^2) \tag{2.53}$$

donde C_{An} es el coeficiente Auger para los electrones y C_{Ap} es el coeficiente Auger para los huecos.

2.2.5. Condiciones de contorno

La frontera $\partial\Omega$ del dominio semiconductor puede dividirse generalmente en dos partes disjuntas:

$$\partial\Omega = \partial\Omega_R \cup \partial\Omega_A, \qquad \partial\Omega_R \cap \partial\Omega_A = \emptyset \tag{2.54}$$

donde $\partial\Omega_R$ representa las fronteras físicas reales, como por ejemplo los contactos metálicos y las interfases con material aislante y $\partial\Omega_A$ se refiere a las fronteras artificiales, que no se corresponden con fronteras físicas propiamente dichas y que se utilizan para separar el dispositivo de todos sus vecinos en un circuito integrado.

Las fronteras artificiales se introducen para simplificar la simulación numérica mediante la eliminación de regiones del dispositivo que tengan poca importancia en su comportamiento eléctrico. Para aislar completamente el dispositivo se considera como condición de contorno que la componente normal a estas fronteras del vector campo eléctrico sea nula. También deben ser nulas las componentes normales de las densidades de corriente:

$$E \cdot \nu|_{\partial\Omega_A} = J_n \cdot \nu|_{\partial\Omega_A} = J_p \cdot \nu|_{\partial\Omega_A} = 0 \tag{2.55}$$

donde ν representa la normal exterior a la frontera $\partial\Omega$ del dominio semiconductor.

Estas expresiones se trasladan al potencial y a las concentraciones de portadores, imponiendo condiciones naturales (tipo Neumann) sobre sus valores en la frontera:

$$\left.\frac{\partial\psi}{\partial\nu}\right|_{\partial\Omega_A} = \left.\frac{\partial n}{\partial\nu}\right|_{\partial\Omega_A} = \left.\frac{\partial p}{\partial\nu}\right|_{\partial\Omega_A} = 0 \tag{2.56}$$

Por otro lado, desde un punto de vista matemático los contactos metálicos Ω_M constituyen un subconjunto cerrado y conectado de la frontera real $\partial\Omega_R$ del dominio semiconductor. Estos contactos se fabrican mediante una unión íntima del metal con el semiconductor. En los dispositivos se utilizan dos tipos de contactos: óhmicos y Schottky.

Un contacto óhmico presenta una resistencia relativa al sustrato despreciable y no perturba de forma significativa la eficiencia del dispositivo. Usualmente en estos contactos se asumen las condiciones de equilibrio térmico:

$$np|_{\partial\Omega_C} = n_i^2 \tag{2.57}$$

y de neutralidad de carga:

$$(n - p - C)|_{\partial\Omega_C} = 0 \tag{2.58}$$

De las ecuaciones 2.57 y 2.58 se obtienen fácilmente condiciones de contorno tipo Dirichlet para n y p:

$$n(x) = n_o, \quad \forall x \in \partial\Omega_M \tag{2.59}$$

$$p(x) = p_o, \quad \forall x \in \partial\Omega_M \tag{2.60}$$

El valor del potencial en el contacto óhmico se obtiene como la suma del potencial externo V_C aplicado al contacto y del potencial interno ψ_{bi} producido por el dopado:

$$\psi|_{\partial\Omega_C} = \psi_{bi}|_{\partial\Omega_C} + V_C(t) \tag{2.61}$$

El potencial ψ_{bi} es el que existiría en el interior del semiconductor en una situación de equilibrio térmico sin los potenciales externos aplicados.

Las condiciones de contorno para n y p son independientes del tiempo. Las condiciones para el potencial dependen del tiempo si y sólo si el potencial externo es variable.

Un contacto Schottky presenta una barrera de potencial importante en la unión metal–semiconductor. Esto influye en el funcionamiento del dispositivo ya que se crea una capa delgada de carga espacial. La física de los contactos Schottky es muy compleja y su simulación implica imponer grandes simplificaciones. Para un contacto ideal se usa la siguiente condición tipo Dirichlet para el potencial:

$$\psi|_{\partial\Omega_S} = \psi_{bi}|_{\partial\Omega_S} - \varphi_S + V_S(t) \tag{2.62}$$

donde φ_S denota la altura de la barrera metal–semiconductor.

2.2.6. Escalado de las variables

Para mejorar el tratamiento computacional de los modelos matemáticos en la simulación de semiconductores se introduce un escalado de las variables tal que se obtengan cantidades adimensionales y que se aislen los parámetros dimensionales relevantes de los que depende el modelo.

Si $\widetilde{\omega}$ es el factor de escala de la variable ω, el valor de la variable escalada y adimensional ω^* vendrá dado por:

$$\omega^* = \omega/\widetilde{\omega} \tag{2.63}$$

Existen diferentes formas de escalar las variables, algunas de ellas recogidas en [45, 46, 29, 47], en nuestro caso el escalado utilizado está recogido en [47] y parte de las siguientes premisas:

- Las longitudes están referidas a la longitud característica del dispositivo $\widetilde{l}$ elegida de tal forma que l sea del mismo orden de magnitud que el diámetro del dominio semiconductor:

$$\widetilde{l} = O(\mathrm{diam}(\Omega)) \tag{2.64}$$

- Los potenciales están escalados por el potencial térmico $V_T = \frac{KT}{q}$.

- Las concentraciones de portadores se escalan por una concentración característica $\widetilde{C}$ que se elige de tal forma que resulte del mismo orden de magnitud que la máxima concentración de impurezas:

$$\widetilde{C} = O\left(\max_{\Omega} |C(x)|\right) \tag{2.65}$$

  De esta forma, las concentraciones escaladas máximas serán del orden de la unidad.

- Las movilidades se escalan por una movilidad de referencia $\widetilde{\mu}$ elegida de tal forma que $\mu_n/\widetilde{\mu}$ sea como máximo del orden de la unidad.

Los valores de los factores de escala resultantes para los distintos parámetros pueden encontrarse en [47]. En la tabla 2.1 se indican algunos de los principales factores de escalado utilizados y su valor típico.

Es posible reescribir, en función de las variables escaladas, las ecuaciones que modelan el comportamiento de los dispositivos en la aproximación de arrastre–difusión. Denotando las nuevas variables escaladas con los mismos

Símbolo	Significado	Factor escala	Valor típico
x	posición	$\widetilde{l}$	5×10^{-3} cm
t	tiempo	$\frac{\widetilde{l}^2}{\widetilde{\mu} V_T}$	9.7×10^{-7} s
ϕ	potencial	V_T	$0.0259V$
n	concent. de e^-	$\widetilde{C}$	$10^{17} cm^{-3}$
p	concent. de h^+	$\widetilde{C}$	$10^{17} cm^{-3}$
J_n	dens. corriente de e^-	$\frac{qV_T \widetilde{C} \widetilde{\mu}}{\widetilde{l}}$	$83 A cm^{-2}$
J_p	dens. corriente de h^+	$\frac{qV_T \widetilde{C} \widetilde{\mu}}{\widetilde{l}}$	$83 A cm^{-2}$
μ_n	movilidad de e^-	$\widetilde{\mu}$	$1000 cm^2 V^{-1} s^{-1}$
μ_p	movilidad de h^+	$\widetilde{\mu}$	$1000 cm^2 V^{-1} s^{-1}$
C	concent. de dopantes	$\widetilde{C}$	$10^{17} cm^{-3}$
R	recombinación	$\frac{V_T \widetilde{\mu} \widetilde{C}}{\widetilde{l}^2}$	$1.1 \times 10^{23} cm^{-3} s^{-1}$
τ_n	tiempo vida media e^-	$\frac{\widetilde{l}^2}{\widetilde{\mu} V_T}$	$9.7 \times 10^{-7} s$
τ_p	tiempo vida media h^+	$\frac{\widetilde{l}^2}{\widetilde{\mu} V_T}$	$9.7 \times 10^{-7} s$

Tabla 2.1: Factores de escalado

símbolos que las antiguas, las ecuaciones 2.15, 2.18 y 2.19 y las ecuaciones auxiliares 2.20 y 2.21 se expresan como:

$$\lambda^2 \nabla^2 \psi = (n - p - C) \tag{2.66}$$

$$\nabla J_n - \frac{\partial n}{\partial t} = R \tag{2.67}$$

$$\nabla J_p + \frac{\partial p}{\partial t} = -R \tag{2.68}$$

$$J_n = -\mu_n n \nabla \phi_n \tag{2.69}$$

$$J_p = -\mu_p p \nabla \phi_p \tag{2.70}$$

El problema estático escalado se obtiene haciendo $\frac{\partial n}{\partial t} = 0$ en las ecuaciones 2.67 y 2.68. El parámetro λ, cuyo cuadrado multiplica al Laplaciano del potencial en la ecuación de Poisson escalada 2.66, viene dado por:

$$\lambda = \frac{\lambda_D}{\widetilde{l}}, \qquad \lambda_D = \sqrt{\frac{\varepsilon V_T}{q \widetilde{C_o}}} \tag{2.71}$$

siendo λ_D la longitud de Debye característica del dispositivo. En la mayoría de las situaciones, λ es un parámetro muy pequeño, típicamente del orden de 10^{-3} a 10^{-5}.

Como conclusión, el sistema de ecuaciones obtenido, para su uso en el simulador, a través del modelo de arrastre–difusión está formado por 3 ecuaciones no lineales (2.66, 2.67 y 2.68) con 3 incógnitas (ψ, ϕ_n y ϕ_p) que están acopladas entre sí y que no pueden resolverse directamente, por lo cual es necesario recurrir a técnicas numéricas para su resolución.

2.3. Discretización de las ecuaciones de arrastre–difusión

El modelo matemático descrito en la sección anterior está formado por las ecuaciones diferenciales que modelan el comportamiento de los dispositivos objeto de estudio. Para su resolución es preciso encontrar una solución numérica de estas ecuaciones junto con unas condiciones de contorno dando lugar a un sistema no lineal de ecuaciones diferenciales parciales elípticas. Para ello se discretiza el sistema de ecuaciones aplicando el método de elementos finitos (FEM).

El método de elementos finitos es una técnica numérica que resuelve, de forma aproximada, problemas descritos por ecuaciones diferenciales en derivadas parciales o que pueden ser descritos como minimización de un funcional. En lugar de aproximar la ecuación directamente, como ocurre en otros métodos, se reformula el problema de forma variacional, utilizando una integral sobre el dominio del problema que involucra la ecuación diferencial. Este dominio se divide en subdominios llamados elementos finitos, que en nuestro caso serán tetraedros. En estos subdominios se aproxima la solución por funciones sencillas, dando como resultado un sistema algebraico de dimensión finita.

A continuación se muestra una breve introducción al método de elementos finitos, para posteriormente aplicarlo a las ecuaciones de Poisson y de continuidad de electrones y huecos.

2.3.1. Método de elementos finitos

Se parte de una ecuación diferencial parcial elíptica definida sobre un dominio Ω:

$$-\nabla^2 u = f \quad en \quad \Omega \tag{2.72}$$

$$u = 0 \quad sobre \quad \partial\Omega \tag{2.73}$$

Para su resolución se realiza una formulación débil del problema, en la que se multiplica por una función de ponderación v y se integra sobre el

dominio:

$$-\int_\Omega \nabla^2 uv\, dx = \int_\Omega fv\, dx, \quad \forall v \in H_0^1(\Omega) \tag{2.74}$$

siendo H_0^1 un espacio de dimensión infinita cuyas funciones tienen derivadas cuadrado integrables y que se anulan en la frontera $\partial\Omega$.

Considerando la siguiente relación:

$$\nabla(h \cdot \nabla g) = \nabla h \nabla g + h\nabla^2 g \tag{2.75}$$

y el teorema de Green–Gauss:

$$\int_\Omega \nabla g \nabla h\, dx = -\int_\Omega g\nabla^2 h\, dx + \int_{\partial\Omega} g\frac{\partial h}{\partial \vec{n}}\, dS \tag{2.76}$$

en el que $\vec{n}$ es un vector normal a la superficie S, y suponiendo que $\dfrac{\partial h}{\partial \vec{n}} = 0$ en $\partial\Omega$, la ecuación 2.74 se puede expresar como:

$$\int_\Omega \nabla u \nabla v\, dx = \int_\Omega fv\, dx, \quad \forall v \in H_0^1 \tag{2.77}$$

Seguidamente, el método de elementos finitos toma un subespacio de dimensión finita H_0^K del espacio original $H_0^1(\Omega)$ en el que sus funciones son polinomios de grado bajo definidos sobre regiones pequeñas, conocidas como elementos finitos, del dominio del problema Ω. Es posible construir una base del espacio finito H_0^K de modo que cualquier función perteneciente a este espacio puede ponerse como combinación lineal de las funciones de dicha base. Según esto, el problema aproximado se formula como:

$$\textit{Encontrar}\, u_K \in H_0^K \,\textit{tal que}\, \int_\Omega \nabla u_K \nabla v_K\, dx = \int_\Omega f v_K\, dx \tag{2.78}$$

donde u_K y v_K son aproximaciones de u y v utilizando una base ϕ_j del espacio H_0^K de dimensión n:

$$u_K = \sum_{j=1}^{n} u_j \phi_j \tag{2.79}$$

$$v_K = \sum_{j=1}^{n} v_j \phi_j \tag{2.80}$$

Al sustituir estas aproximaciones en el problema 2.78 se obtiene el siguiente sistema lineal de ecuaciones:

$$\sum_{j=1}^{n} \alpha_{ij} u_j = \beta_i, \quad i = 1, \ldots, n \tag{2.81}$$

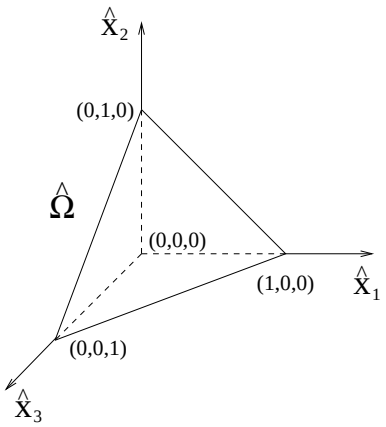


Figura 2.3: Tetraedro elemental.

donde α_{ij} y β_i son:

$$\begin{aligned} \alpha_{ij} &= \int_\Omega \nabla\phi_i \nabla\phi_j \, dx \\ \beta_i &= \int_\Omega f\phi_i \, dx \end{aligned}$$

El método de elementos finitos proporciona una forma general y sistemática para generar las funciones de la base del espacio de dimensión finita H_0^K. Da lugar a unas funciones básicas que se definen a trozos sobre subregiones del dominio llamadas elementos finitos que en el caso tridimensional pueden ser tetraedros, cubos, prismas, pirámides, etc.

Con el fin de simplificar la construcción de las funciones base ϕ_i se definen un conjunto de funciones elementales φ_l^e, conocidas como funciones de forma, sobre cada elemento finito. El índice $e = 1, \ldots, N$ referencia los elementos de la partición y el índice l el vértice P del elemento que se está tratando, con $l = 1, \ldots, 4$ si se usan elementos tetraédricos. Las funciones φ_l^e se pueden definir como:

$$\varphi_l^e(P_i) = \begin{cases} 1 & i = l \\ 0 & i \neq l \end{cases}$$

De esta forma, cualquier función base ϕ_i se puede definir sumando las contribuciones de todas las funciones de forma definidas sobre el vértice i–ésimo,

$$\phi_i = \sum_{e \ni P_i} \varphi_{P_i}^e \tag{2.82}$$

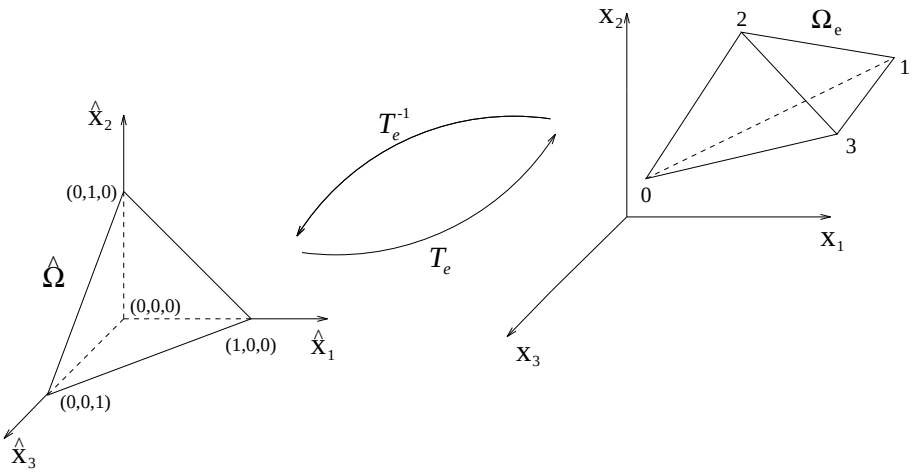


Figura 2.4: Transformación entre el elemento patrón y uno genérico.

Las funciones de forma para un tetraedro situado sobre el origen de coordenadas, denominado tetraedro elemental o patrón, como el representado en la figura 2.3 son:

$$\begin{aligned} &\widehat{\varphi}_0 = 1 - \widehat{x}_1 - \widehat{x}_2 - \widehat{x}_3 \\ &\widehat{\varphi}_1 = \widehat{x}_1 \\ &\widehat{\varphi}_2 = \widehat{x}_2 \\ &\widehat{\varphi}_3 = \widehat{x}_3 \\ &con\ \widehat{x}_1,\ \widehat{x}_2,\ \widehat{x}_3\ \in\ [0,1] \end{aligned} \tag{2.83}$$

El cálculo de las funciones de forma y la obtención de las matrices elementales se realizan sobre cada elemento. Estos cálculos sobre un elemento genérico Ω_e suelen ser complicados si se realizan directamente en función de las coordenadas originales $\{x_1, x_2, x_3\}$ puesto que son dependientes del elemento de la malla que se esté procesando. Para la resolución de estas integrales es preferible introducir una serie de transformaciones invertibles $\{T_e\}$ entre el elemento patrón $\widehat{\Omega}$ y cada uno de los elementos genéricos Ω_e. De esta forma, será posible transformar las integrales sobre Ω_e en integrales sobre $\widehat{\Omega}$.

En la figura 2.4 se muestra una transformación $T_e : \widehat{\Omega} \rightarrow \Omega_e$, en la que se realiza una transformación entre un elemento genérico Ω_e y el elemento patrón $\widehat{\Omega}$, siendo este el tetraedro de vértices $(0,0,0)$, $(1,0,0)$, $(0,1,0)$ y $(0,0,1)$.

En una transformación de este tipo, dada una función g, $g : \Omega_e \rightarrow \mathrm{R}$,

$$\int_{\Omega_e} g d\Omega_e = \int_{\widehat{\Omega}} \widehat{g} |J_e| d\widehat{\Omega} \tag{2.84}$$

siendo $\widehat{g} = g \circ T_e$, y $|J_e|$ el determinante de la matriz jacobiana de la transformación T_e:

$$J_e = \begin{pmatrix} x_1^1 - x_1^0 & x_1^2 - x_1^0 & x_1^3 - x_1^0 \\ x_2^1 - x_2^0 & x_2^2 - x_2^0 & x_2^3 - x_2^0 \\ x_3^1 - x_3^0 & x_3^2 - x_3^0 & x_3^3 - x_3^0 \end{pmatrix} \tag{2.85}$$

donde $P_i = (x_1^i, x_2^i, x_3^i)$, con $i = 0, 1, 2, 3$, son los cuatro vértices del tetraedro Ω_e. El determinante del jacobiano se puede obtener como:

$$|J_e| = (\vec{v}_{01} \times \vec{v}_{02}) \cdot \vec{v}_{03} \tag{2.86}$$

donde $\vec{v}_{ij}$ es el vector que va del vértice P_i al P_j. Se puede comprobar que el determinante $|J_e|$ es igual a seis veces el volumen del tetraedro Ω_e.

Además, se verifica que

$$\nabla g = J_e^{-t} \widehat{\nabla} \widehat{g} \tag{2.87}$$

siendo J_e^{-t} la transpuesta de la inversa de la matriz jacobiana.

La integral sobre $\widehat{\Omega}$ suele ser evaluada utilizando fórmulas de cuadratura numérica. En general, una fórmula de cuadratura se define especificando las coordenadas $(\widehat{x}_1^l, \ldots, \widehat{x}_d^l)$ de N_l puntos de integración en el dominio sobre el que se evaluará la integral y un conjunto de N_l números w_l, denominados pesos de la cuadratura. Así, si $\widehat{G} = \widehat{g}|J_e|$, la integral viene dada por:

$$\int_{\Omega_e} g d\Omega_e = \int_{\widehat{\Omega}} \widehat{G} d\widehat{\Omega} = \sum_{l=0}^{N_l - 1} \widehat{G}(\widehat{x}_1^l, \ldots, \widehat{x}_d^l) w_l + \widehat{E} \tag{2.88}$$

donde $\widehat{E}$ es el error de cuadratura.

En el cálculo de integrales de superficie también es posible el uso de una transformación sobre el elemento patrón. Para ello se introduce un conjunto adicional de transformaciones $\{w_s\}$ que proyectan un elemento patrón del espacio $(d-1)$ dimensional sobre cada una de las superficies $\partial\widehat{\Omega}$.

En la figura 2.5 se proyecta un triángulo patrón $\widehat{S}$ definido en el plano, con vértices $(0,0)$, $(1,0)$ y $(0,1)$, mediante w_s sobre una de las caras del tetraedro patrón $\widehat{\Omega}$. La transformación t_e proyecta esta cara $\partial\widehat{\Omega}$ sobre una cara $\partial\Omega_e$ del elemento original.

Dada una función h, $h : \partial\Omega_e \to \mathrm{R}$ definida sobre $\partial\Omega_e$ se verifica que:

$$\int_{\partial\Omega_e} h dS_e = \int_{\widehat{S}} (\widehat{h} \circ w_s)|j| dS \tag{2.89}$$

siendo $\widehat{h} = h \circ t_e$, y $|j|$ el determinante de la matriz jacobiana de la transformación. Se puede demostrar que $|j| = 2S_{\partial\Omega_e}$, donde $S_{\partial\Omega_e}$ es el área de la cara $\partial\Omega_e$ del tetraedro original.

A continuación se aplica el método de elementos finitos a las ecuaciones de Poisson y de continuidad de electrones.

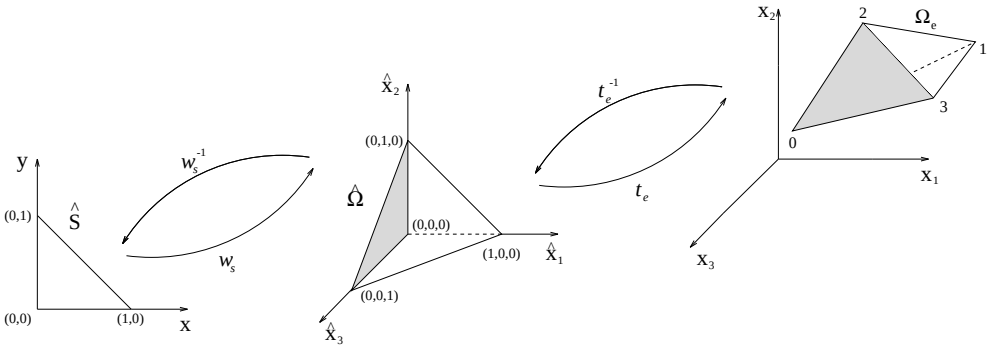


Figura 2.5: Transformación de coordenadas para una cara del tetraedro.

2.3.2. Ecuación de Poisson

La ecuación de Poisson escalada en estado estacionario a discretizar es:

$$\lambda^2 \nabla^2 \psi = (n - p - C), \qquad x \in \Omega \tag{2.90}$$

sujeta a las condiciones de frontera:

$$\left.\frac{\partial \Psi}{\partial \nu}\right|_{\partial\Omega_N} = 0 \tag{2.91}$$

$$\Psi\Big|_{\partial\Omega_D} = \Psi_D\Big|_{\partial\Omega_D} \tag{2.92}$$

Si además se tiene en cuenta la presencia de carga interfacial entre el material semiconductor y el aislante es necesario considerar otra condición de contorno:

$$\varepsilon_{sc}\left.\frac{\partial \psi}{\partial \nu}\right|_{\partial\Omega_I} - \varepsilon_{ais}\left.\frac{\partial \psi_{ais}}{\partial \nu}\right|_{\partial\Omega_I} = Q_I \tag{2.93}$$

siendo ε_{sc} y ε_{ais} las permitividades de los materiales semiconductor y aislante y Q_I la carga acumulada en la superficie de separación.

Si se considera que:

$$\varepsilon_{ais}\left.\frac{\partial \psi_{ais}}{\partial \nu}\right|_{\partial\Omega_I} = 0 \tag{2.94}$$

entonces la condición de contorno se expresa como:

$$\varepsilon_{sc}\left.\frac{\partial \psi}{\partial \nu}\right|_{\partial\Omega_I} = Q_I \tag{2.95}$$

Sea $H_0^1(\Omega \cup \partial\Omega_N)$ el espacio de funciones con derivada cuadrado integrable que se anulan en sentido débil en la frontera $\partial\Omega_D$. La formulación

débil del problema anterior se obtiene multiplicando la ecuación 2.90 por una función de test arbitraria $\xi \in H_0^1(\Omega \cup \partial\Omega_N)$.

$$\lambda^2 \nabla^2 \psi \xi = (n - p - C)\xi, \qquad \forall \xi \in H_0^1(\Omega \cup \partial\Omega_N) \tag{2.96}$$

Integrando la ecuación anterior en el dominio Ω y aplicando el teorema de Green al primer término de la ecuación, junto con las condiciones de contorno se obtiene:

$$-\int_\Omega \lambda^2 (\nabla\psi)^t \nabla\xi d\Omega + \int_{\partial\Omega} \lambda^2 \xi \frac{\partial\psi}{\partial\nu} dS = \int_\Omega (n - p - C)\xi d\Omega, \qquad \forall \xi \in H_0^1 \tag{2.97}$$

El método de Galerkin consiste en buscar una solución aproximada del problema variacional anterior en un subespacio de dimensión finita H_0^K del espacio infinito $H_0^1(\Omega \cup \partial\Omega_N)$. Tomando una base $\{\theta_i\}_{i=1}^K$ de $H_0^K \subset H_0^1$ el problema anterior se puede formular como:

$$-\int_\Omega \lambda^2 (\nabla\psi)^t \nabla\theta_i d\Omega + \int_{\partial\Omega} \lambda^2 \theta_i \frac{Q_I}{\varepsilon_{sc}} dS = \int_\Omega (n - p - C)\theta_i d\Omega, \qquad \forall i = 1, \ldots, K \tag{2.98}$$

Expresando la aproximación del potencial definido sobre los nodos de la malla P_j como combinación lineal de las funciones de la base:

$$\psi = \sum_{j=1}^K \psi(P_j)\theta_j \tag{2.99}$$

y expresando de la misma forma la concentración de electrones se obtiene,

$$\lambda^2 \sum_{j=1}^K \int_\Omega (\nabla\theta_j)^t \nabla\theta_i d\Omega \psi_j + \lambda^2 \int_{\partial\Omega} \frac{Q_I}{\varepsilon_{sc}} \theta_i dS +$$

$$+ \sum_{j=1}^K \int_\Omega (n_j - p_j - C_j)\theta_j \theta_i d\Omega = 0 \qquad \forall i = 1, \ldots, K \tag{2.100}$$

Con el fin de realizar de forma más simple y efectiva las integrales anteriores, las funciones de la base se expresan en términos de las funciones de forma y se realizan las integrales respecto al elemento patrón. Teniendo en cuenta que $n_j = n_{ienj} \exp\left(\psi_j - \phi_{nj}\right)$, la fórmula 2.100 se expresa como:

$$\lambda^2 \sum_{j=1}^K \int_{\widehat{\Omega}} (\nabla\widehat{\varphi_j})^t \nabla\widehat{\varphi}_i |J_e| d\widehat{\Omega} \psi_j + \lambda^2 \int_{\partial\Omega} \frac{Q_I}{\varepsilon_{sc}} \varphi_i dS +$$

$$+ \sum_{j=1}^K \int_{\widehat{\Omega}} (n_j - p_j - C_j)\widehat{\varphi_j}\widehat{\varphi}_i |J_e| d\widehat{\Omega} = 0, \qquad \forall i = 1, \ldots, K \tag{2.101}$$

2.3.3. Ecuaciones de continuidad

En la discretización de las ecuaciones de continuidad de electrones y huecos en estado estacionario se toman como punto de partida las siguientes expresiones escaladas:

$$\nabla J_n = R \tag{2.102}$$

$$\nabla J_p = -R \tag{2.103}$$

$$J_n = -\mu_n n \nabla(\phi_n) \tag{2.104}$$

$$J_p = -\mu_p p \nabla(\phi_p) \tag{2.105}$$

$$R = R_{BB} + R_A + R_{SRH} \tag{2.106}$$

$$\forall x \in \Omega$$

junto con las condiciones contorno,

$$n|_{\partial\Omega_D} = n_{eq}, \quad p|_{\partial\Omega_D} = p_{eq} \tag{2.107}$$

$$\left.\frac{\partial n}{\partial \nu}\right|_{\partial\Omega_N} = 0, \quad \left.\frac{\partial p}{\partial \nu}\right|_{\partial\Omega_N} = 0 \tag{2.108}$$

Una primera aproximación en la búsqueda de la solución numérica de estas ecuaciones consistiría en proceder como en el caso general explicado en la sección 2.3.1, de tal forma que, tomando como ejemplo la ecuación de continuidad de electrones, se realizaría una formulación variacional del problema y se aplicaría el teorema de Green:

$$\int_\Omega \mu_n n \nabla(\phi_n) \nabla\theta_i d\Omega = \int_\Omega R\theta_i d\Omega \tag{2.109}$$

A partir de esta ecuación se realizaría una formulación en términos de las funciones de forma y se integraría sobre el elemento patrón. Pero es necesario tener en cuenta que la discretización de las ecuaciones de continuidad requiere un cuidado especial, debido al comportamiento de capa que presenta el potencial y las concentraciones de portadores. Se puede demostrar [47] que, para un dispositivo unidimensional, la discretización estándar de las ecuaciones sólo conduce a soluciones aceptables si se verifica que:

$$\max_i |\psi_{i+1} - \psi_i| \ll 1 \tag{2.110}$$

donde ψ_i representa el potencial en el vértice i–ésimo de la discretización. Esta es una condición muy restrictiva, pues requiere que las zonas de transición, en las que el potencial varía de forma brusca, sean discretizadas con un tamaño de malla muy fino. Así, si un número pequeño de puntos de la

malla cae dentro de esta capa, $|\psi_{i+1} - \psi_i|$ puede tomar valores muy grandes y el esquema estándar no obtiene resultados satisfactorios.

En un caso multidimensional la situación es la misma por lo que es necesario que el potencial varíe lentamente en cada elemento finito. Esto puede requerir mallas excesivamente finas con un número de vértices muy elevado, lo que hace muy costosa, en términos de memoria y tiempo de computación, y a veces imposible su realización práctica. Por tanto, es necesario utilizar otros métodos de discretización que eviten este problema.

Uno de los métodos más importantes utilizado para evitar este problema fue propuesto inicialmente por Scharfetter y Gummel [48] para un dispositivo unidimensional usando una aproximación en diferencias finitas, aunque posteriormente fue aplicado a varias dimensiones y tipos de elementos no sólo usando diferencias finitas sino también elementos finitos.

Con esta aproximación es posible obtener discretizaciones apropiadas para las ecuaciones de continuidad en sistemas unidimensionales teniendo en cuenta que las densidades de corriente J_n y J_p son unas funciones que varían lentamente, es decir, no presentan comportamiento de capa como el potencial. De esta forma, es posible aproximar J_n y J_p mediante funciones con valor constante sobre cada elemento. Calculando estas funciones, y aproximando la densidad de corriente como combinación lineal de las mismas, se pueden obtener nuevas formulaciones del problema que conducen a mejores resultados. Es preciso tener en cuenta que en casos multidimensionales las densidades de corriente pueden exhibir también comportamiento de capa, pero sin embargo, su variación va a ser mucho menor que la de ψ, n o p.

El procedimiento de este método se detalla a continuación para la ecuación de continuidad de electrones, puesto que la formulación sería análoga para la ecuación de continuidad de huecos. Sea G_e el centro de gravedad de un elemento Ω_e y $\psi_{\triangle}(x)$ la función del potencial eléctrico definido sobre el elemento finito. Es posible obtener una aproximación de la densidad de corriente sobre el elemento Ω_e aproximando μ_n por su valor μ_{n,G_e} en el centro, y ϕ_n por funciones definidas a trozos sobre el dominio. Haciendo estas consideraciones se expresa J_n sobre el elemento finito como el vector:

$$J_{n,G_e} = -\mu_{n,G_e} n_{ien_{\triangle},G_T}(x) \exp(\psi_{\triangle}(x) - \phi_{n,\triangle}(x)) \nabla\phi_n(x), \quad \forall x \in \Omega_e \tag{2.111}$$

De esta forma:

$$J_n = \sum_{\Omega_e} J_{n,G_e} \tag{2.112}$$

Pasando a integrar sobre el elemento patrón y usando $\nabla g = J_e^{-t} \widehat{\nabla} \widehat{g}$ se

transforma la ecuación anterior en,

$$J_{n,G_e} = -\mu_{n,G_e} n_{ien_\triangle,G_T}(\widehat{x}) \exp(\psi_\triangle(\widehat{x}) - \phi_{n,\triangle}(\widehat{x})) J_e^{-t} \widehat{\nabla}\phi_n(\widehat{x}), \quad \forall \widehat{x} \in \widehat{\Omega} \tag{2.113}$$

Si se multiplican ambos términos de la ecuación anterior por $e^{-\psi_\triangle(\widehat{x})} J_e^t$, integrando la j–ésima componente con respecto a $\widehat{x}_j$ en el intervalo $[0,1]$ y teniendo en cuenta que $\frac{\partial x}{\partial \widehat{x}}$ es constante sobre el elemento patrón, tras invertir el cambio de variables se obtiene,

$$J_{n,G_e} = \mu_{n,G_e} n_{ien_\triangle,G_T} \exp(\psi_\triangle(P_0)) J_e^{-t} B_T(\psi_\triangle) J_e^t \nabla(\exp(-\phi_{n,\triangle}(x))) \tag{2.114}$$

siendo $\{P_0, \ldots, P_d\}$ los vértices del elemento Ω_e, $\psi(P_i)$ el valor del potencial en el vértice i–ésimo y $B_e(\psi)$ la matriz diagonal:

$$B_T(\psi) = \begin{pmatrix} B(\psi(P_0) - \psi(P_1)) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & B(\psi(P_0) - \psi(P_d)) \end{pmatrix} \tag{2.115}$$

donde B es la función de Bernoulli definida como [29]:

$$B(z) = \begin{cases} \frac{z}{(e^z - 1)} & \text{si } |z| \geq 10^{-4} \\ \\ 1 - \frac{z}{2}\left(1 + \frac{z}{6}\right)\left(1 - \frac{z^2}{60}\right) & \text{si } |z| < 10^{-4} \end{cases} \tag{2.116}$$

Expresando la ecuación de continuidad de electrones en función del valor de la densidad de corriente en cada elemento se obtiene,

$$\sum_{\Omega_e} \nabla J_{n,Ge} \quad = \quad R \tag{2.117}$$

Teniendo en cuenta las condiciones de contorno, la formulación variacional de esta ecuación se puede expresar como:

$$\sum_{\Omega_e} \int_{\Omega_e} J_{n,Ge} \nabla \xi d\Omega_e = -\sum_{\Omega_e} \int_{\Omega_e} R\xi d\Omega_e, \qquad \forall \xi \in H_0^1(\Omega \cup \partial\Omega_N) \tag{2.118}$$

Sustituyendo en la expresión anterior el valor de la densidad de corriente expresado en la fórmula 2.114 y tomando una base finita, $\left\{\theta_i\right\}_{i=1}^K$, del espacio $H_0^1(\Omega \cup \partial\Omega_N)$ se obtiene:

$$\sum_{\Omega_e} \int_{\Omega_e} \mu_{n,G_e} n_{ien_\triangle,G_T} \exp(\psi_\triangle(P_0)) J_e^{-t} B_T(\psi_\triangle) J_e^t \nabla(\exp(-\phi_{n,\triangle}(x))) \nabla\theta_i d\Omega_e$$

$$= -\sum_{\Omega_e} \int_{\Omega_e} R\theta_i d\Omega_e, \quad \forall i = 1, \ldots, K \tag{2.119}$$

A continuación, se expresa el término de recombinación R como combinación lineal de las funciones de la base, sabiendo que su valor en el nodo i–ésimo es:

$$\begin{aligned} R_i = C_{BBi}(n_i p_i - n_{oi} p_{oi}) + \frac{n_i p_i - n_{oi} p_{oi}}{\tau_{p_i}(n_i + n_{Ti}) + \tau_{ni}(p_i + p_{Ti})} \\ + C_{Ani}(n_i^2 p_i - n_{oi}^2 p_{oi}) + C_{Ap_i}(n_i p_i^2 - n_{oi} p_{oi}^2) \end{aligned} \tag{2.120}$$

Por último, utilizando la expresión anterior y reemplazando las funciones elementales por las funciones de forma, es posible expresar la expresión 2.119 como:

$$\sum_{\Omega_e} \int_{\Omega_e} \mu_{n,G_e} n_{ien_\triangle,G_T} \exp(\psi_\triangle(P_0)) \nabla\varphi_j J_e^{-t} B_T(\psi_\triangle) J_e^t \nabla(\exp(-\phi_{ni})) \nabla\varphi_i d\Omega_e$$

$$= -\sum_{\Omega_e} \int_{\Omega_e} R_j \varphi_j \varphi_i d\Omega_e, \quad \forall i = 1, \ldots, K \tag{2.121}$$

En el caso de la ecuación de continuidad de huecos se realizaría el mismo procedimiento y la expresión final obtenida sería:

$$\sum_{\Omega_e} \int_{\Omega_e} \mu_{p,G_e} n_{iep_\triangle,G_t} \exp(-\psi_{p_j}) \nabla\varphi_j J_e^{-t} B_T(-\psi_\triangle) J_e^t \nabla(\exp(\phi_{p,\triangle}(x))) \nabla\varphi_i d\Omega_e$$

$$= -\sum_{\Omega_e} \int_{\Omega_e} R_j \varphi_j \varphi_i d\Omega_e, \quad \forall i = 1, \ldots, K \tag{2.122}$$

2.4. Resumen

En este capítulo se han descrito las principales técnicas de simulación de dispositivos semiconductores utilizadas en la actualidad, ordenándolas en base a su complejidad y al tiempo de computación que consumen. La aproximación de arrastre–difusión ha sido la técnica elegida para la implementación en el simulador 3D de dispositivos en base a que permite alcanzar resultados adecuados en un tiempo de simulación asumible. Por lo tanto, la segunda parte del capítulo se ha centrado en este modelo, en las ecuaciones que lo componen, las ecuaciones de Poisson y de continuidad de electrones y huecos, y en la obtención de las expresiones discretizadas de estas ecuaciones.

Capítulo 3

Implementación paralela del simulador y resolución de sistemas

El simulador 3D paralelo de dispositivos semiconductores utilizado en este trabajo se basa en el modelo de arrastre–difusión (drift–diffusion o también D–D). En la figura 3.1 se muestra un esquema de las etapas básicas del proceso de simulación. En el modelo D–D se resuelven las ecuaciones de Poisson y de continuidad de huecos y electrones con el fin de obtener el potencial electrostático y los cuasi–potenciales de Fermi. Para la resolución de las ecuaciones diferenciales no lineales que componen este modelo es preciso encontrar una solución numérica que cumpla unas determinadas condiciones de contorno. Para ello se discretiza el sistema de ecuaciones por medio del método de elementos finitos.

Las ecuaciones no lineales de Poisson y de continuidad de portadores están acopladas entre sí, por lo que una vez discretizadas son desacopladas por medio del método iterativo de Gummel [49] y resueltas por separado. Para su resolución se utiliza el método iterativo de Newton–Raphson [50]. Este método linealiza las ecuaciones, debiendo resolver en cada iteración un sistema lineal de ecuaciones. Estos sistemas lineales son dispersos y están mal condicionados a causa de las altas variaciones en las variables implicadas en la simulación. Además, en el caso de las ecuaciones de continuidad de portadores las matrices no son diagonal dominantes [51].

Los métodos utilizados en la resolución de los sistemas lineales se pueden agrupar en dos categorías, métodos directos, si están basados en la factorización de la matriz dispersa, y métodos iterativos, si tratan de encontrar la solución al sistema por medio de aproximaciones sucesivas tomando como

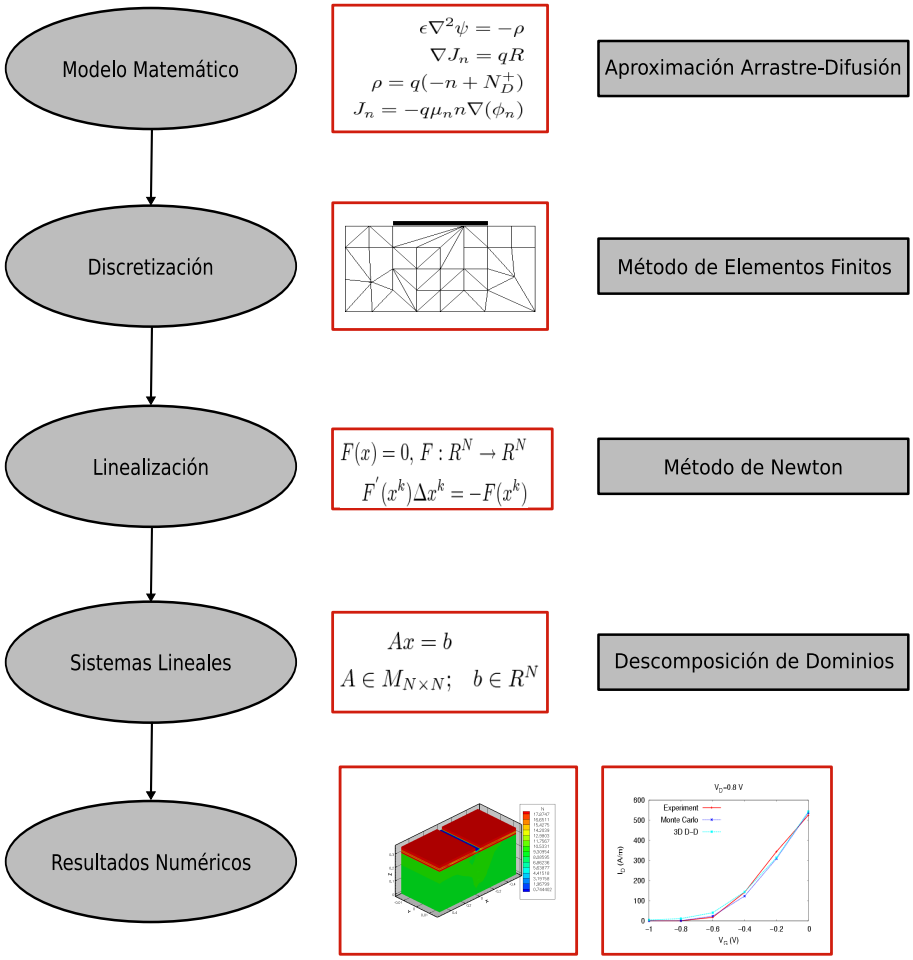


Figura 3.1: Esquema de las etapas del proceso de simulación de dispositivos semiconductores usando la aproximación de arrastre–difusión.

partida una solución inicial. La conveniencia de usar los diferentes tipos de métodos en el ámbito de la simulación de dispositivos semiconductores ha sido analizada en [52]. En nuestro caso se ha optado por el uso de métodos iterativos, concretamente técnicas basadas en descomposición de dominios [53, 54]. Estas técnicas son altamente eficientes en computación paralela, lo que las hace idóneas para su uso en el simulador.

El objetivo es utilizar el simulador tridimensional en el estudio del impacto de las fluctuaciones de parámetros intrínsecos de diferente naturaleza en las curvas características de dispositivos HEMT y PHEMT. Este estudio consiste en la simulación de un conjunto estadístico de dispositivos lo suficientemente grande como para que permita extraer con una precisión

adecuada los parámetros que caracterizan la distribución estadística resultante. Esto incrementa la complejidad computacional, haciendo esencial el uso de computación paralela.

Por lo tanto el simulador tridimensional está diseñado para ser empleado en sistemas de memoria distribuida utilizando una estrategia MIMD (múltiple flujo de instrucciones, múltiple flujo de datos) bajo el paradigma SPMD (un programa, múltiples datos) [55]. La comunicación entre procesadores se realiza por medio de la librería de paso de mensajes MPI estándar [56, 57].

A continuación, en este capítulo se describe la implementación paralela del simulador 3D de dispositivos HEMT, empezando por el proceso de generación y particionamiento de las mallas utilizadas para representar a estos dispositivos. Seguidamente se describen algunos de los métodos de resolución de los sistemas de ecuaciones que surgen de la discretización de las ecuaciones de arrastre–difusión. Además, se menciona brevemente la técnica de reordenamiento de matrices utilizada y algunos de los formatos más conocidos de almacenamiento de matrices dispersas. Para finalizar, como resumen del proceso de simulación, se describen las tres etapas básicas en las que se divide el simulador 3D paralelo, preprocesado, procesado y postprocesado.

3.1. Mallado y técnicas de particionamiento

En el capítulo 2 se obtuvieron las expresiones discretizadas de la ecuación de Poisson y de las ecuaciones de continuidad de portadores. Para ello se utilizó una descomposición del dominio objeto de estudio en tetraedros sobre la que se aplica el método de elementos finitos.

Teniendo esto en cuenta, la simulación de un dispositivo empieza con un preprocesado en el que se genera una malla de elementos finitos a partir de la estructura de capas del dispositivo y se leen los ficheros de entrada que contienen los parámetros de la simulación. A continuación, para una óptima ejecución en paralelo, se realiza un particionamiento de la malla en tantos subdominios como se desee y se reordenan las variables de cada subdominio.

3.1.1. Mallado

El simulador tridimensional utiliza mallas generadas a partir de dos programas diferentes, el QMG [58, 59], desarrollado por S. A. Vavasis en la Universidad de Cornell, y el MMG [60], desarrollado en el Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela. Estos malladores permiten obtener mallas tetraédricas no estructuradas

para una geometría del dispositivo dada.

Ambos programas están basados en el uso de octrees como estructura para la generación de los tetraedros. A nivel interno el método de generación de los tetraedros a partir del octree es diferente, aunque a nivel de usuario la principal diferencia se encuentra en la especificación del tamaño de los tetraedros. En el programa QMG es posible especificar el refinado en los planos, aristas y vértices de la estructura, mientras que el MMG permite una mayor versatilidad al posibilitar el refinado en cualquier parte del volumen del dispositivo.

El proceso de mallado se divide en varias fases. En primer lugar, a partir de las dimensiones de la estructura que se desea simular, se genera el modelo geométrico. En segundo lugar, en la etapa de mallado se fija un tamaño inicial para la malla, se decide el tipo de malla que se quiere generar, por ejemplo estructurada o no estructurada, y el tipo de elemento a usar. Después, en algunas zonas, como por ejemplo en algunas fronteras, o debajo de la zona de puerta del dispositivo, se establece un valor más pequeño del parámetro que controla el tamaño del elemento finito en esa región, con el fin de obtener una malla con más elementos en esa zona. También es posible realizar un refinamiento total de la malla obtenida de modo que cada tetraedro se divide en ocho nuevos tetraedros. La etapa de refinado es muy importante puesto que permite obtener mallas que obtengan mejores soluciones al problema físico planteado, disminuyéndose de esta forma el error de la solución.

3.1.2. Técnicas de particionamiento

Antes de estudiar las técnicas de particionamiento es necesario introducir el concepto de grafo. Un grafo dirigido o digrafo es un conjunto de vértices, también conocidos como nodos, y de aristas dirigidas que unen los nodos. El patrón de cualquier matriz cuadrada dispersa tiene asociado un digrafo y cada digrafo genera un patrón de dispersión.

Para una matriz cuadrada dispersa A, se asocia un vértice del digrafo con cada una de sus filas. Si a_{ij} es una entrada, esto es, un elemento no nulo de la matriz, entonces existe una arista desde el vértice i hasta el vértice j en el digrafo. En el caso de grafos procedentes de mallas de elementos finitos, en los cuales una conexión del vértice i al j va a implicar necesariamente otra conexión del j al i, no es necesario que las aristas que conecten los nodos estén dirigidas. Se obtiene así otro tipo de grafos, conocidos como no dirigidos, o simplemente grafos.

En la figura 3.2 se representa un dominio bidimensional, y debajo de él,

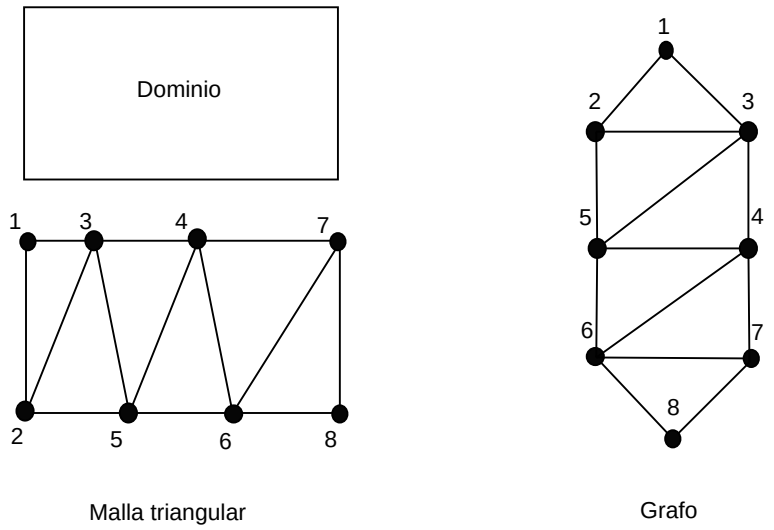
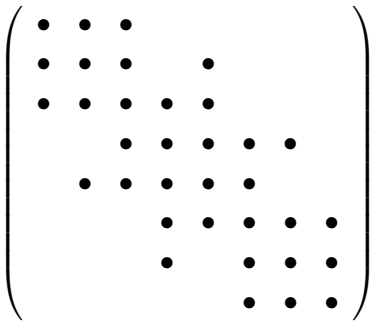


Figura 3.2: Dominio, mallado triangular resultante y grafo asociado.

se muestra el dominio mallado, usando para ello 6 elementos triangulares y 8 nodos. A la derecha de la figura se indica el grafo asociado a dicha malla.

Un grafo de v vértices genera una matriz cuadrada A de $v \times v$ elementos, donde $a_{ij} \neq 0$ si y sólo si existe una arista que une los vértices i y j. Hay que tener en cuenta que $a_{ii} \neq 0$, $0 \leq i < v$, ya que siempre existe una arista que une un vértice consigo mismo, aunque no se represente en el diagrama. La matriz A tendrá una estructura simétrica si el grafo es no dirigido. Entonces se dice que la estructura de la matriz A ha sido generada por el grafo y se denomina matriz de adyacencias del grafo. Por ejemplo, la matriz asociada al grafo representado en la figura 3.2 sería:



El problema que se plantea en definitiva en el particionamiento de grafos de elementos finitos es el de distribuir una malla de elementos finitos entre los procesadores disponibles de tal modo que se optimice el tiempo de ejecución. Generalmente, en el particionamiento se intentan verificar al máximo los

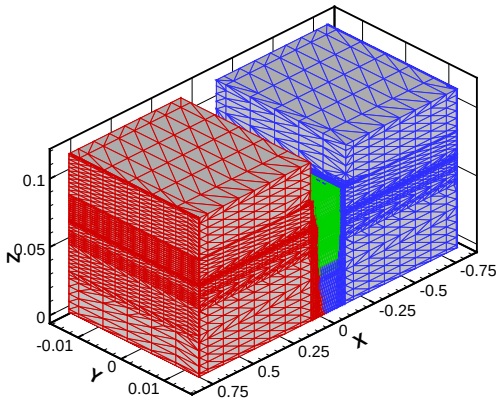


Figura 3.3: Malla de 29012 nodos dividida en 3 subdominios utilizando el programa METIS para un dispositivo HEMT.

siguientes criterios:

1. La relación computación–comunicación dentro de un procesador debe ser maximizada.

2. La carga computacional debe estar balanceada entre todos los procesadores.

3. El coste temporal asociado al control de las comunicaciones y a la complejidad en los almacenamientos debe ser minimizado.

Siguiendo estos criterios, generalmente las técnicas de particionamiento se basan en dividir los nodos correspondientes a la malla de elementos finitos en p partes aproximadamente iguales, de tal manera que el número de aristas que unen los nodos pertenecientes a partes diferentes se minimice. Por lo tanto, el dominio original del problema se divide en p subdominios. Estas técnicas son empleadas por diferentes librerías, entre las que caben destacar Chaco [61] y METIS [62]. En concreto esta última librería ha sido la utilizada en el simulador a causa de la calidad de sus particionamientos y de su rapidez. La librería METIS emplea algoritmos de particionamiento de grafos multinivel junto con otros métodos como bisección espectral, bisección geométrica, etc [63]. Un ejemplo de un dispositivo HEMT dividido en 3 subdominios utilizando este paquete puede encontrarse en la figura 3.3.

3.2. Linealización del sistema discretizado

En los dispositivos HEMT y PHEMT los portadores mayoritarios son electrones. Por lo tanto, lejos de la zona de ruptura es posible ignorar, tal y como se hará a partir de este momento, la contribución de la ecuación de continuidad de huecos, así como los términos de generación y recombinación, y resolver únicamente las ecuaciones de Poisson y de continuidad de electrones. De esta forma se logra acelerar el proceso de simulación.

Por lo tanto, especificando para el caso concreto de estos dispositivos, las ecuaciones del modelo de arrastre–difusión a resolver son 2.101 y 2.121, considerando en esta última ecuación la recombinación nula. Estas ecuaciones constituyen un sistema acoplado de ecuaciones diferenciales no lineales en derivadas parciales que definen el comportamiento del transistor. Si el dispositivo está discretizado en K nodos, es necesario resolver un sistema de $2K$ ecuaciones con $2K$ incógnitas $\Psi = (\psi, \phi_n)$ de la forma:

$$F(\Psi) = \begin{pmatrix} F_\psi(\psi, \phi_n) \\ F_{\phi_n}(\psi, \phi_n) \end{pmatrix} = 0 \tag{3.1}$$

donde $F_\psi(\psi, \phi_n)$ y $F_{\phi_n}(\psi, \phi_n)$ representan las ecuaciones de Poisson y de continuidad de electrones respectivamente.

Para la resolución del sistema 3.1 se pueden utilizar métodos acoplados o desacoplados. Los primeros son métodos tipo Newton [64], en los que se resuelve el sistema completo formado por las $2K$ ecuaciones directamente, actualizando a la vez los valores de ψ y de ϕ_n. Los métodos desacoplados se basan en el método de Gummel [49] y resuelven las dos ecuaciones secuencialmente, siendo preciso resolver dos sistemas de ecuaciones de dimensión K. De esta forma se obtienen de forma consecutiva actualizaciones para ψ y de ϕ_n. A continuación se estudian en mayor profundidad estos dos métodos.

3.2.1. Método de Newton–Raphson

El método de Newton es un método iterativo que se aplica a la resolución de sistemas no lineales.

Dada una función $F : \Re^N \to \Re^N$ se busca un vector $x^* \in \Re^N$ tal que $F(x^*) = 0$. Para resolver el sistema se parte de un vector solución inicial $x^0 \in \Re^N$ y se busca la serie x^k, tal que $\lim_{k\to\infty} x^k = x^*$. Los elementos de dicha serie se obtienen mediante la siguiente iteración:

$$F'(x^k)\Delta x^k = -F(x^k) \tag{3.2}$$

$$x^{k+1} = x^k + \Delta x^k \tag{3.3}$$

la cual se repite hasta que se cumple alguno de los criterios de parada, siendo $F'(x)$ el jacobiano del sistema, que se obtiene como:

$$F'(x) = \left(\frac{\partial F_i}{\partial x_j} \right) \tag{3.4}$$

La importancia del método de Newton se debe a que, con ciertas condiciones naturales de F, la iteración presenta una convergencia cuadrática

$$\|x^{k+1} - x^*\| \leq c \cdot \|x^k - x^*\|^2 \tag{3.5}$$

siempre que el vector inicial x^0 esté suficientemente próximo a la solución buscada. Esto indica que el $(k+1)$–ésimo error cometido es proporcional al cuadrado del k–ésimo error, por lo que la convergencia es muy rápida, siempre que los errores sean pequeños.

La aplicación del método de Newton a las ecuaciones de los semiconductores implica resolver el siguiente sistema en cada iteración:

$$F'(\Psi^k)\Delta\Psi^k = -F(\Psi^k) \tag{3.6}$$

y actualizar las variables haciendo $\Psi^{k+1} = \Psi^k + \Delta\Psi^k$.

Suele ser necesario mejorar la convergencia del método de Newton para lo cual se utiliza una versión modificada, que emplea un parámetro de amortiguamiento o sobrerrelajación, t_k, en la actualización de Ψ, es decir:

$$\Psi^{k+1} = \Psi^k + t_k \cdot \Delta\Psi^k \tag{3.7}$$

siendo $0 < t_k \leq 1$. El parámetro de amortiguamiento t_k se suele ajustar a partir de resultados experimentales, aunque también existen métodos teóricos para obtenerlo [50, 65]. En el caso de que la matriz de coeficientes esté muy mal condicionada se puede utilizar otra versión que suma al jacobiano el término $s_k I$, donde $s_k \in \Re^+$ e I es la matriz identidad, de modo que se mejoran las propiedades del sistema.

Aunque el método de Newton es atractivo teóricamente, su implementación práctica puede ser complicada, debido a que en cada paso es preciso resolver un sistema no lineal y no simétrico de ecuaciones y, si el número N de variables es muy elevado (como ocurre en la simulación 3D), esto puede ser muy costoso tanto en tiempo de computación como en consumo de memoria. Existe una variante de este método conocida como Newton–Raphson modificado en la que se evita el cálculo de la matriz jacobiana en cada iteración haciendo uso de una matriz calculada en alguna iteración anterior y sólo se calcula el término independiente que es mucho menos costoso. El coste de este procedimiento es mucho menor por iteración pero tiene la desventaja de tener una convergencia más lenta y, en algunos casos, ni siquiera llega a converger [66].

3.2.2. Método de Gummel

El método de Gummel [49] ha sido aplicado ampliamente a la resolución de las ecuaciones básicas de los semiconductores. Matemáticamente, corresponde a un algoritmo iterativo por bloques no lineal tipo Gauss–Seidel [67]. En el caso particular de los dispositivos HEMT, usando este método se desacoplan las ecuaciones de Poisson y de continuidad de electrones y se resuelve cada una de ellas por separado.

El método de Gummel realizaría la siguiente iteración: Dado (ψ^k, ϕ_n^k) para $k = 0$, ψ^{k+1} se computa resolviendo la ecuación de Poisson, con ϕ_n^k constante. El valor de ϕ_n^k se obtiene a continuación, resolviendo la ecuación de continuidad de electrones con ψ^{k+1} constante incluyendo las condiciones de contorno.

Este método se ha mostrado muy útil en la práctica. La convergencia puede alcanzarse incluso empezando con malas condiciones iniciales, y puede ser muy rápida en muchas ocasiones. Por otro lado, en algunas aplicaciones, como por ejemplo en situaciones de muy alto nivel de inyección en el dispositivo semiconductor y en los casos en los que el factor de generación–recombinación tiene un peso elevado, el método podría tener problemas de convergencia. Sin embargo, en situaciones cercanas al equilibrio térmico y con $R = 0$, numerosos autores avalan la utilidad de este método, ya que la ecuación de continuidad de electrones puede considerarse como una pequeña perturbación de la ecuación del potencial.

Existen diferentes trabajos destinados a mejorar la convergencia del método de Gummel, pero evitando tener que resolver el sistema acoplado. En algunas ocasiones el método de Gummel converge rápidamente en las primeras iteraciones, pero luego la convergencia se hace más lenta. En este caso se pueden combinar los métodos de Gummel y de Newton. De esta forma primero se aproxima la solución haciendo uso del método de Gummel y luego se cambia al método de Newton para aprovechar su propiedad de convergencia cuadrática cerca de la solución. También existen otras aproximaciones basadas en combinar los métodos de Gummel y Newton usando diversas estrategias [68, 69].

3.3. Resolución de sistemas de ecuaciones lineales

Los sistemas de ecuaciones lineales se pueden expresar como $Ax = b$, donde dada una matriz A de orden n x n y un vector b n–dimensional, se trata de determinar el vector solución x de dimensión n.

Además, en nuestro caso, la matriz A es dispersa, por lo que será ven-

tajoso trabajar con métodos que aprovechen esta propiedad. Una matriz A de orden n x n es dispersa si una gran cantidad de sus términos son nulos. Se denomina índice de dispersión, β, a la relación entre el número de elementos no nulos de la matriz (α) y el número total de elementos ($n \times n$); es decir $\beta = \frac{\alpha}{n \times n}$. El valor del índice de dispersión necesario para que la matriz pueda ser considerada dispersa depende del problema a resolver, del patrón de la matriz y de la arquitectura de la máquina en la cual se implementa el código.

3.3.1. Implementación paralela: estructura de los sistemas locales

El dominio formado por el conjunto de nodos incógnitas del problema se descompone en una serie de subdominios, de tal forma que se obtienen tantos subdominios como procesadores. Para entender esto es necesario distinguir entre tres tipos de nodos incógnitas, tal y como aparece representado en la figura 3.4:

- **Nodos internos:** nodos locales que sólo tienen relación con otros nodos interiores al subdominio y por tanto sin ningún contacto con nodos pertenecientes a otros subdominios.

- **Nodos frontera internos:** nodos locales pero acoplados con nodos frontera pertenecientes a otros subdominios.

- **Nodos frontera externos:** nodos pertenecientes a otros subdominios pero acoplados con nodos frontera internos de este subdominio.

Las ecuaciones locales a cada subdominio no tienen porque corresponder con ecuaciones contiguas en el sistema original. Las filas de la matriz asignadas a un procesador se pueden dividir en dos partes: en una *submatriz local* A_i que actúa sobre los nodos internos y una *submatriz de interfaz* X_i que se refiere a las entradas correspondientes a nodos frontera externos que interaccionan con nodos frontera internos al subdominio. Las variables remotas deben ser recibidas de los otros procesadores antes de que el producto matriz–vector se pueda completar en estos procesadores.

Los nodos frontera se numeran siempre después de los nodos internos. Esta ordenación local de los datos presenta varias ventajas, entre las que se incluye una eficiente comunicación entre procesadores. De igual modo, cada vector local de incógnitas x_i es dividido en dos partes: un subvector u_i de nodos internos seguido por el subvector y_i correspondiente a los nodos

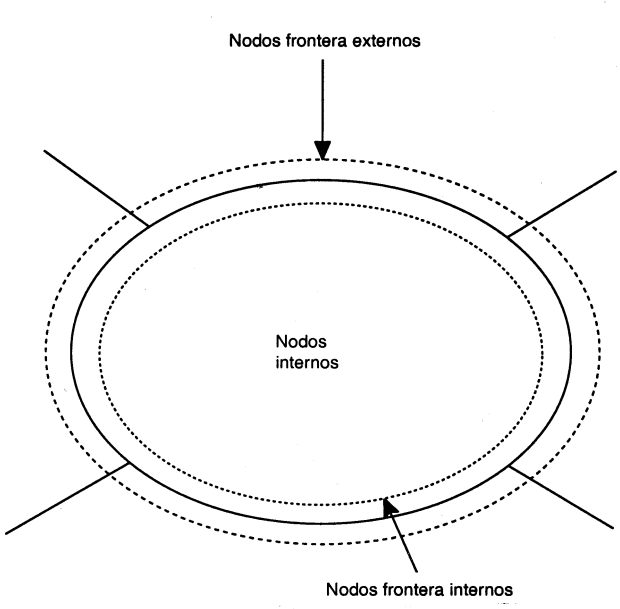


Figura 3.4: Representación local de una matriz dispersa distribuida.

frontera internos. Siguiendo el mismo procedimiento el vector independiente también es dividido en los subvectores f_i y g_i.

Así:

$$x_i = \begin{pmatrix} u_i \\ y_i \end{pmatrix}, \qquad b_i = \begin{pmatrix} f_i \\ g_i \end{pmatrix} \tag{3.8}$$

La matriz local A_i, perteneciente al procesador i, también puede particionarse por bloques, de tal modo que se separen las contribuciones de cada clase de nodo a dicha matriz de la siguiente forma:

$$A_i = \begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} \tag{3.9}$$

siendo B_i la parte asociada a los nodos internos del subdominio, E_i y F_i las contribuciones dadas por los acoplamientos de los nodos internos y frontera internos a un subdominio y C_i muestra el acoplamiento de los nodos frontera internos entre sí. Según esto, las ecuaciones locales de un subdominio i pueden ser expresadas como:

$$\begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \begin{pmatrix} 0 \\ \sum_{j \epsilon N_i} E_{ij} y_i \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix} \tag{3.10}$$

donde el término $E_{ij}y_j$ es la contribución a las ecuaciones locales del subdominio vecino j–ésimo, siendo N el conjunto de subdominios vecinos al subdominio i. Para implementaciones prácticas, los subvectores de los nodos frontera externos se agrupan en un vector $y_{i,ext}$. Por lo tanto la contribución al sistema local por parte de los nodos frontera externos puede expresarse:

$$\sum_{j \epsilon N_i} E_{ij} y_j = X_i y_{i,ext} \tag{3.11}$$

3.3.2. Métodos de resolución de sistemas de ecuaciones lineales

Los métodos de resolución de sistemas de ecuaciones lineales se pueden dividir en dos grupos:

1. **Métodos directos:** están basados en la factorización de la matriz A para convertir el sistema lineal en otro con formato de resolución más simple. Durante la factorización, un elemento de la matriz con un valor inicial nulo, puede pasar a tener un valor distinto de cero; sufre entonces un proceso de llenado. Cuantos más elementos sufran llenado, más operaciones tendrá que realizar el algoritmo aumentando así la carga computacional. A causa de esto, los métodos directos necesitan más memoria, puesto que al producirse el llenado se tiene que almacenar en memoria, además del sistema original todas las nuevas entradas no nulas, y tienen una complejidad computacional mayor que los métodos iterativos. Esto supone la imposibilidad de usar métodos directos al menos en gran parte de las aplicaciones donde se va trabajar con matrices muy grandes como ocurre en muchas ocasiones en simulaciones en dos y tres dimensiones. Existen diferentes métodos de factorización de una matriz, siendo uno de los más habituales la factorización LU.

2. **Métodos iterativos:** son técnicas que tratan de encontrar la solución de un sistema mediante sucesivas aproximaciones a partir de una solución inicial. Los métodos iterativos se pueden clasificar en dos tipos:

   a) **Métodos estacionarios:** son los más sencillos y fáciles de implementar, pero en general menos efectivos que los métodos no estacionarios [70]. Se basan en la relajación de coordenadas, empezando con una solución aproximada y modificando los componentes de la aproximación hasta que se alcanza la convergencia. Se trata de obtener:

$$x_k = Ax_{k-1} + b \tag{3.12}$$

donde ni la matriz A ni el vector b dependen del contador de iteraciones. Ejemplos de estos métodos son:

1) *Jacobi*: Basado en la computación de cada variable del vector solución con respecto al resto de las variables. Es un método sencillo de implementar pero la convergencia es lenta.
2) *Gauss–Seidel*: Método similar al anterior exceptuando que utiliza los valores actualizados de la solución tan pronto como estén disponibles. En general, converge más rápido que Jacobi.
3) *SOR(Sobrerrelajación sucesiva)*: Se deriva del método Gauss–Seidel introduciendo un parámetro de relajación w. Con una correcta elección de w el método converge más rápido que el Gauss–Seidel en un orden de magnitud.
4) *SSOR(Sobrerrelajación sucesiva simétrica)*: Aunque no presenta ventajas como método iterativo respecto al SOR, es muy útil como precondicionador para métodos no estacionarios.

b) **Métodos no estacionarios:** Son más complejos que los anteriores pero altamente eficientes [54]. Se diferencian de los métodos estacionarios en que las computaciones implican información que cambia en cada iteración. En la actualidad los más populares pertenecen al conjunto de los métodos del subespacio de Krylov [71]. El subespacio de Krylov $K^i(A, r_o)$ de dimensión i, asociado con un sistema lineal $Ax = b$, para un vector solución inicial x_0 y un vector residuo $r_0 = b - Ax_0$ se define como el subespacio cubierto por los vectores $r_0, Ar_0, A^2r_0, ..., A^{i-1}r_0$. Dependiendo de las características de la matriz que define el problema, es posible clasificar estos métodos en varios grupos:

1) Si la matriz es simétrica y definida positiva el método de *Gradiente Conjugado (CG)* es el más apropiado. Utiliza una secuencia de vectores ortogonales x_i para los que se minimiza $(x_i - x)^T A(X_i - x)$ sobre todos los vectores en el espacio de Krylov actual $K^i(A, r_0)$.
2) Si la matriz es simétrica pero no es definida positiva, una posibilidad es considerar el método de *Lanczos* o los métodos de *MINRES*. En los métodos *MINRES*, los elementos $x_i \in K^i(A, r_0)$ se determinan minimizando la norma cuadrática de los residuos $||b - Ax_i||^2$, mientras que en el método de *Lanczos*

los elementos x_i son determinados por los residuos $b - Ax_i$ perpendiculares al subespacio de Krylov. En estos casos es necesario almacenar toda la secuencia, lo que conlleva un consumo elevado de memoria.

3) Si la matriz no es simétrica, en general no se puede determinar un conjunto óptimo de soluciones $x_i \in K^i(A, r_0)$ con pocas secuencias de vectores. Sin embargo, es posible computar el conjunto de vectores $x_i \in K^i(A, r_0)$ para los que se cumple la condición $b - Ax_i \perp K^i(A^T, r_0)$ (normalmente se elige $s_0 = r_0$). Así se generan dos secuencias de vectores, una con la matriz de coeficientes A y otra con A^T, y en vez de ortogonalizar cada secuencia lo hacen mutuamente: este es el método *Gradiente Biconjugado (BiCG).* Requiere un almacenamiento limitado aunque la convergencia puede ser irregular. Una variante de este método es el *Residuo Cuasi–Mínimo (QMR)* que aplica un resolutor por mínimos cuadrados y una actualización de la solución a los residuos del BiCG, suavizando el comportamiento de la convergencia y haciendo estos métodos más robustos.

4) Si A no es simétrica, se puede computar la secuencia de vectores $x_i \in K^i(A, r_0)$ para los que los residuos sean minimizados usando una norma euclídea (mínimos cuadrados). Esto es lo que se implementa en el método *Mínimo Residuo Generalizado (GMRES).* Esta implementación requiere almacenar toda la secuencia, lo que conlleva un consumo elevado de memoria. Una variante del GMRES es el *Mínimo Residuo Generalizado Flexible (FGMRES)* que permite que el precondicionamiento varíe a cada paso.

5) Las operaciones con la matriz A^T en el método BiCG pueden ser sustituidas por operaciones con la matriz original A teniendo en cuenta que $\langle x, A^T y\rangle = \langle x, Ay\rangle$ donde el operador $\langle ...\rangle$ representa el producto escalar de dos vectores. Como la secuencia de vectores que usan A^T en el método BiCG se utiliza sólo para mantener el espacio dual con el que los residuos se ortogonalizan, reemplazar en las operaciones A por A^T permite la expansión del subespacio de Krylov y encontrar mejores aproximaciones a la solución virtualmente con el mismo coste computacional por iteración. Esta idea da lugar a los métodos iterativos conocidos como métodos híbridos: el *Gradiente Conjugado Cuadrático (CGS),* el *Gradiente Biconjugado Esta-*

bilizado (BCGSTAB), TFQMR, etc.

Todos los métodos iterativos presentan, en general, una convergencia demasiado lenta, así que es necesario introducir mejoras en el esquema numérico que aceleren la convergencia. Esto se realiza aplicando el precondicionador adecuado al sistema lineal a resolver.

A continuación se describe brevemente en qué consiste una factorización LU, para definir después el concepto de precondicionador y su utilidad.

3.3.3. Factorización LU

Consiste en factorizar la matriz A como el producto de una matriz triangular inferior L y una matriz triangular superior U:

$$A = LU \tag{3.13}$$

Se considera que la matriz L está normalizada ($l_{ii} = 1$) y almacenada por columnas mientras que la matriz U está almacenada por filas, de tal forma que siempre se mantenga la siguiente igualdad:

$$\sum_{j=1}^{n} l_{ij} u_{jk} = a_{ik} \tag{3.14}$$

El algoritmo básico para la factorización LU es el siguiente:

DO $i = 1, n$
  $l_{ii} = 1$
  DO $k = 1, i-1$
    $l_{ik} = (a_{ik} - \sum_{j=1}^{k-1} l_{ij} \cdot u_{jk})/u_{kk}$
  END DO
  DO $k = 1, i$
    $u_{ki} = (a_{ki} - \sum_{j=1}^{k-1} l_{kj} \cdot u_{ji})$
  END DO
END DO

Figura 3.5: Factorización LU básica.

Una vez factorizada la matriz el sistema de ecuaciones $Ax = b$ podría representarse como $LUx = b$. Para su resolución se realiza la siguiente operación:

$$L^{-1}LUx = L^{-1}b \tag{3.15}$$

obteniendo:

$$Ux = L^{-1}b \tag{3.16}$$

si se define $z = L^{-1}b$ el sistema a resolver será:

$$Ux = z \tag{3.17}$$

Para ello primero se obtiene z, resolviendo la ecuación $Lz = b$, cuya solución es:

$$z_i = \frac{1}{l_{ii}}(b_i - \sum_{j=1}^{i-1} l_{ij}z_j) \qquad i = 1, 2, ...n \tag{3.18}$$

Para resolver por último $Ux = z$:

$$x_i = \frac{1}{u_{ii}}(z_i - \sum_{j=i+1}^{n} u_{ij}x_j) \qquad i = n, n-1, ..., 1 \tag{3.19}$$

Obteniendo así el vector solución x.

3.3.4. Precondicionadores

Los precondicionadores se usan para mejorar las propiedades de los sistemas lineales con el fin de acelerar la convergencia de los métodos iterativos. Partiendo del sistema $Ax = b$, se busca una matriz M que transforme el sistema en otro cuyas propiedades sean más favorables y, por tanto, más fácil de resolver. El sistema lineal podría representarse como:

$$M^{-1}Ax = M^{-1}b \tag{3.20}$$

En la búsqueda de la matriz M se pueden seguir dos caminos:

1. Encontrar una matriz M que se aproxime a A, de tal forma que resolver el sistema con esta matriz sea más fácil que hacerlo con la matriz A, resolviendo por lo tanto:
$$Mx = b \tag{3.21}$$

2. Lograr una matriz M aproximada a A^{-1} de tal forma que la expresión AM sea tan cercana a la identidad como sea posible en algún sentido, como por ejemplo minimizando $|AM - I|$ en la norma de Frobenius. De esta forma sólo sería necesario realizar el producto de M por el vector independiente para obtener la solución. Es decir:
$$MAx = Mb \Longrightarrow x \simeq Mb \tag{3.22}$$

Los precondicionadores se pueden aplicar por la izquierda, por la derecha y/o por ambos lados. Partiendo del sistema $Ax = b$, el precondicionamiento por la izquierda se basa en realizar la operación:

$$M^{-1}Ax = M^{-1}b \tag{3.23}$$

En cambio, el precondicionamiento por la derecha se aplica así:

$$AM^{-1}u = b \tag{3.24}$$

con $u = Mx$, donde u una nueva variable que nunca necesitará ser invocada explícitamente.

El precondicionamiento por ambos lados no es más que una combinación de los dos métodos de precondicionamiento anteriores.

Los precondicionadores a estudiar se pueden clasificar en tres tipos principales: los llamados precondicionadores clásicos, precondicionadores multimalla y los precondicionadores basados en descomposición de dominios. Seguidamente se describen las principales características de cada uno de ellos.

Precondicionadores clásicos

Se basan en manipulaciones algebraicas de la matriz para obtener algún tipo de aproximación de la inversa de la matriz de coeficientes. Ejemplos de estos precondicionadores son las factorizaciones incompletas. Se trata de factorizar la matriz A (por ejemplo a través de una factorización LU) pero sin introducir todo el llenado que se produce durante este proceso [72]. Es decir, en una factorización incompleta LU se computa la matriz triangular inferior (L) y superior (U) de tal forma que la matriz residuo $R = LU - A$ satisfaga ciertas limitaciones, como puede ser tener entradas nulas en ciertas posiciones. Existen básicamente cuatro versiones de factorizaciones incompletas:

1. *Factorizaciones sin llenado, ILU(0):* son las más sencillas y no introducen llenado alguno, es decir, la factorización incompleta tiene la misma cantidad de elementos no nulos y en las mismas posiciones que la matriz A [73]. Este tipo de precondicionadores no suelen ser lo suficientemente potentes.

   Un ejemplo de este tipo de factorizaciones incompletas se puede observar en la figura 3.6, en la que está representada la matriz A (imagen inferior izquierda) y unas matrices triangulares L y U con la misma estructura que las partes inferior y superior de la matriz A. La matriz

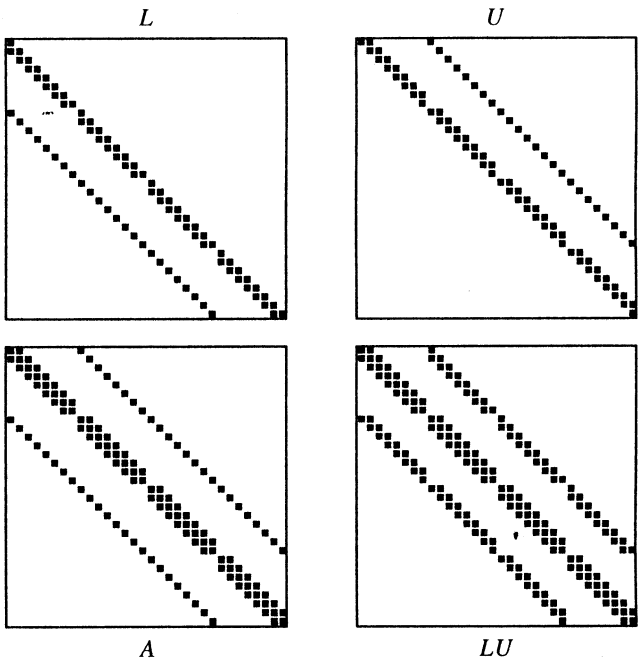


Figura 3.6: Ejemplo de factorización incompleta LU.

A representada en la figura está generada a partir de una malla 8x4. Realizando el producto LU, la matriz resultante tendrá el patrón dado en la figura inferior derecha. Generalmente es imposible que coincidan las matrices A y LU, a causa de la aparición de entradas diagonales extra al realizar el producto (elementos de llenado). Ignorando estos elementos es posible lograr unas matrices L y U cuyo producto sea aproximadamente igual a A en las otras diagonales.

2. *Factorizaciones con llenado, ILU(k):* en las que se usa como criterio para la introducción del llenado la posición dentro de la matriz [73]. El parámetro k de una factorización ILU(k) indica el número de columnas alrededor de la diagonal en las que se permite llenado. Este tipo de factorizaciones tienen el defecto de considerar que la importancia numérica de un llenado depende únicamente de la proximidad topológica sin tener en cuenta el fenómeno físico que la matriz A representa. Se podrían dar otras definiciones del parámetro k.

3. *Factorizaciones con llenado, ILU(τ):* Estas factorizaciones deciden introducir o no llenado en función de si el elemento es superior o inferior a un umbral numérico determinado τ, relativo al valor de los elementos

de la fila i–ésima de A usando cierta medida (como puede ser el valor medio de los elementos de la fila i–ésima) [74].

4. *Factorizaciones con llenado, ILU($fill, \tau$):* en este caso se usan dos criterios para la introducción de llenado; la posición dentro de la matriz y un umbral numérico [75]. Así al factorizar la fila i–ésima se introducen todos los llenados que superen un umbral numérico relativo a esa fila (parámetro τ); una vez acabada la factorización de la fila i–ésima, sólo se almacena en la estructura de datos de salida tantos elementos como tuviese la matriz A en la fila i–ésima más $2 \cdot fill$ (fill elementos en la parte L y fill elementos en la parte U). Se eligen para su almacenamiento aquellos con un valor absoluto mayor. Por lo tanto, usando este método τ controla el umbral numérico de cálculo y el parámetro $fill$ la cantidad efectiva de llenado.

Precondicionadores multimalla

Son un grupo de métodos que trabajan sobre varias mallas para alcanzar la solución del problema[76]. Sobre la malla refinada se aplica un método iterativo suave para eliminar las componentes de error oscilatorias en ese nivel. Seguidamente se restringen los errores a una malla más gruesa, en la cual las componentes suaves se convierten en oscilatorias, y se aplica de nuevo el método suave sobre esta malla. Se repite esto de modo recursivo hasta llegar al nivel de malla donde se resuelve directamente el sistema. Este método acelera la convergencia de los métodos iterativos clásicos aplicando métodos iterativos suaves sobre una jerarquía de mallas relacionadas por una serie de operadores de restricción y prolongación.

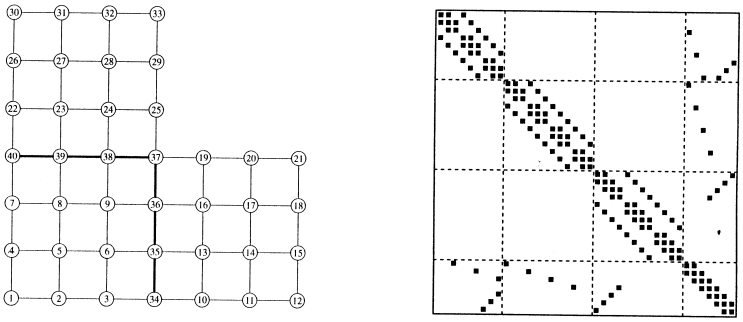
Presentan una complejidad en operaciones lineal con el tamaño del sistema, frente a la complejidad cuadrática de los métodos clásicos. Pero por otro lado consumen más memoria y son menos versátiles en su aplicación [77, 78].

Precondicionadores basados en descomposición de dominios

Las técnicas de descomposición de dominios [79] intentan resolver el problema sobre todo el dominio a partir de la solución en cada subdominio Ω_i.

Para un dominio genérico Ω dividido en s subdominios, en el que se considera que no se produce solapamiento entre subdominios, se cumple:

$$\Omega = \bigcup_{i=1}^{s} \Omega_i \qquad \Omega_i \cap \Omega_j = \emptyset \tag{3.25}$$



(a) (b)

Figura 3.7: (a) Malla asociada a un dominio dividido en tres subdominios, (b) matriz asociada a la malla anterior.

Aunque puede darse el caso de que exista solapamiento. Esto implica que los subdominios son tales que:

$$\Omega = \bigcup_{i=1}^{s} \Omega_i \qquad \Omega_i \cap \Omega_j \neq \emptyset \tag{3.26}$$

Para problemas discretizados, es típico cuantificar la extensión del solapamiento a través del número de líneas de la malla que son comunes a los dos subdominios.

Partiendo de un dominio Ω discretizado es posible etiquetar los nodos por cada subdominio de tal forma que se etiqueten primero los nodos internos a cada subdominio y por último los nodos frontera. La matriz obtenida presentará un patrón de bloques no nulos en forma de flecha, representándose el sistema lineal como:

$$\begin{pmatrix} B_1 & 0 & 0 & \dots & E_1 \\ 0 & B_2 & 0 & \dots & E_2 \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & B_s & E_s \\ F_1 & F_2 & \dots & F_s & C \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_s \\ y \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_s \\ g \end{pmatrix} \tag{3.27}$$

El dominio está dividido en s subdominios, por lo tanto cada x_i será el subvector de incógnitas internas al subdominio Ω_i y el subvector y indicará todas las incógnitas pertenecientes a la interfaz entre los subdominios.

Las submatrices B_i indican el acoplamiento de las ecuaciones de cada subdominio con los nodos internos al mismo, las E_i indican el acoplamiento de las ecuaciones de cada subdominio con los nodos frontera, las F_i se refieren

al acoplamiento de los nodos frontera de cada subdominio con los nodos frontera indicados por y. Por último, la matriz C indica el acoplamiento de los nodos frontera entre sí.

En la figura 3.7(a) se muestra un ejemplo de una malla en forma de L asociada a un dominio divido en tres subdominios. El etiquetado de los nodos es de tal forma que primero se numeran los nodos internos a cada uno de los subdominios y luego los nodos frontera, además, el solapamiento entre subdominios es de orden uno. En la figura 3.7(b) se representa la matriz asociada a esta malla, que presenta un patrón en forma de flecha.

En el simulador 3D se han implementado distintas técnicas de precondicionamiento basadas en descomposición de dominios, que serán descritas a continuación:

1. Método de Schwarz aditivo

   Se puede considerar como una forma de iteración por bloques de Jacobi, en la que cada bloque está referido al sistema de ecuaciones asociado a cada subdominio. La iteración de Jacobi por bloques es un método en el cual en el paso de una iteración a la siguiente se busca la actualización de un conjunto (un bloque) de componentes a la vez. Para ello se realiza un particionamiento de la matriz A y de los vectores solución e independiente en bloques:

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1p} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2p} \\ A_{31} & A_{32} & A_{33} & \dots & A_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{p1} & A_{p2} & A_{p3} & \dots & A_{pp} \end{pmatrix}; \quad x = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \vdots \\ \rho_p \end{pmatrix}; \quad b = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_p \end{pmatrix} \tag{3.28}$$

   Por otro lado se descompone la matriz A, en $A = D - E - F$ con :

$$A = \begin{pmatrix} A_{11} & & & \\ & A_{22} & & \\ & & \ddots & \\ & & & A_{pp} \end{pmatrix}; \qquad E = - \begin{pmatrix} 0 & & & \\ A_{21} & 0 & & \\ \vdots & \vdots & \ddots & \\ A_{p1} & A_{p2} & \dots & 0 \end{pmatrix} \tag{3.29}$$

$$F = -\begin{pmatrix} 0 & A_{12} & \dots & A_{1p} \\ & 0 & \dots & A_{2p} \\ & & \ddots & \\ & & & 0 \end{pmatrix} \tag{3.30}$$

En una iteración k, x^k sería el vector solución obtenido en esa iteración para un determinado bloque. A partir de esto la iteración de Jacobi calcula su componente i–ésima en la siguiente iteración $(k+1)$ con el objetivo de anular la componente i del vector residuo. Es decir conseguir que:

$$(b - Ax^{(k+1)})_i = 0 \tag{3.31}$$

Para que esto sea posible debe darse que:

$$a_{ii}x_i^{(k+1)} = -\sum_{j=1_{j\neq i}}^{p} a_{ij}x_j^k + \beta_i \qquad i = 1, ..., p \tag{3.32}$$

despejando se obtiene:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}(\beta_i - \sum_{j=1_{j\neq i}}^{p} a_{ij}x_j^k) \qquad i = 1, ..., p \tag{3.33}$$

Reescribiéndolo todo en notación matricial:

$$x_i^{(k+1)} = A_{ii}^{-1}((E+F)x_i^k + A_{ii}^{-1}\beta_i \qquad i = 1, ..., p \tag{3.34}$$

se obtiene la ecuación:

$$x_i^{(k+1)} = D^{-1}(E+F)x^k + D^{-1}b \tag{3.35}$$

Este método determina el vector solución, para un determinado bloque, en la iteración $k+1$ utilizando para ello los valores obtenidos en la iteración anterior k. Es interesante tener en cuenta que el número de iteraciones necesarias para obtener la convergencia suelen disminuir rápidamente al aumentar el tamaño del bloque.

Lo visto por el momento sería a nivel de un solo bloque. En conjunto el algoritmo que se sigue es:

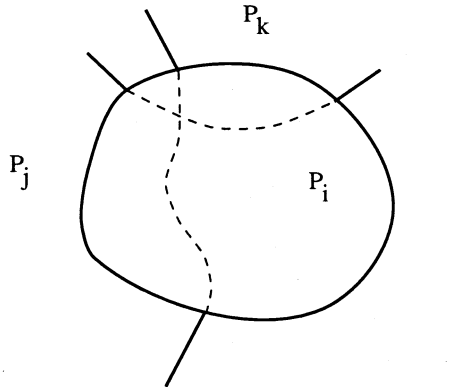


Figura 3.8: Solapamiento de dominios.

1.- For i=1,...,s s= número de dominios existentes

2.- Obtener los datos externos $y_{i,ext}$

3.- Calcular el residuo local:

$$r_i = (b - Ax)_i = b_i - A_i x_i - X_i y_{i,ext} \tag{3.36}$$

4.- Resolver

$$A_i \delta_i = r_i \tag{3.37}$$

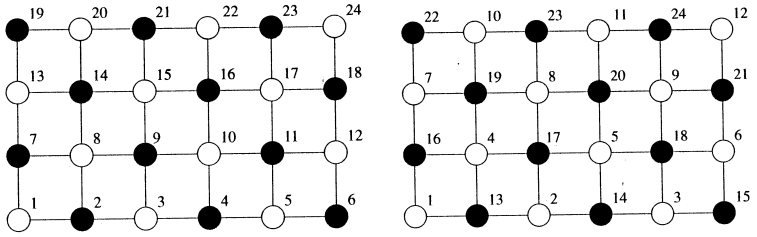
5.- EndDo

6.- Obtener

$$x_{new} = x + \sum_{i=1}^{s} \delta_i \tag{3.38}$$

Es decir, en una iteración, cada subdominio calcularía sus incógnitas locales y utilizaría para ello los valores de las incógnitas externas obtenidos por los otros subdominios en la iteración anterior. Al final de la iteración se produciría una actualización de la solución obteniendo así una nueva aproximación.

Un caso a tener en cuenta sería la aplicación del método de Jacobi con solapamiento. Utilizar solapamiento es una buena estrategia para reducir el número de iteraciones. Hay varias formas posibles de implementar el solapamiento en las iteraciones por bloques de Jacobi. Tomando como ejemplo la figura 3.8 en la que se muestran tres subdominios, se observa que en ciertas zonas, por ejemplo en la región triangular, los datos se solapan tres veces y existirán por lo tanto tres



(a) (b)

Figura 3.9: (a) Etiquetado natural para una malla bicolor, (b) reordenamiento blanco–negro de los nodos.

versiones distintas de ellos, una por P_k, otra por P_j y una versión local asociada a P_i. Cuando se intercambian datos durante la fase de iteración es posible reemplazar la versión local de los datos por una externa o utilizar algún promedio de los datos de las distintas versiones.

2. Método Schwarz multiplicativo

   Este método es un algoritmo de Gauss–Seidel por bloques multicolor. Se puede considerar como una secuencia de eliminación de las componentes del residuo de los sistemas locales a cada procesador. Cada eliminación da lugar a una corrección de las variables locales del vector incógnita, y posteriormente a la corrección del vector residuo global.

   Es preciso tener un criterio de ordenamiento global de los subdominios de modo que subdominios vecinos tengan etiquetas diferentes. La opción seleccionada en este caso, el ordenamiento multicolor, maximiza el paralelismo.

   El ordenamiento multicolor consiste en colorear un grafo, de tal forma que no haya dos nodos adyacentes del mismo color. Su objetivo es la obtención de un grafo que utilice el menor número posible de colores. A continuación se toma como ejemplo el caso más simple, en el que sólo hay dos colores, blanco y negro.

   El grafo de partida muestra el etiquetado natural en el caso de nodos adyacentes de colores distintos (figura 3.9(a)). En el se puede ver que los colores están alternados y la numeración es consecutiva. A continuación se modifica el etiquetado de los nodos numerando primero todos los nodos de un color para a continuación hacer lo mismo con los del otro (figura 3.9(b)). Como los nodos de un mismo color no están acoplados entre sí, el sistema resultante de este reordenamiento tendrá la estructura:

$$\begin{pmatrix} D_1 & F \\ E & D_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{3.39}$$

en la que D_1 y D_2 son matrices diagonales.

En nuestro caso concreto, se utiliza el ordenamiento multicolor en un método basado en descomposición de dominios, dividiendo para ello el sistema en tantos subdominios como número de procesadores, coloreándolos de tal forma que subdominios vecinos tengan colores distintos. Así todos los procesadores de un mismo color podrían computar sus partes locales del vector solución en paralelo porque no existe ninguna dependencia entre ellas. Al finalizar enviarían los valores obtenidos a sus vecinos, permitiendo así que otro color empiece su fase de computación.

Así, el reordenamiento multicolor aplica un algoritmo que tiene el siguiente esquema:

1.- For col=1,...,numcolor

2.- If (col.eq.mycol) then

3.- Obtener los datos externos $y_{i,ext}$

4.- Calcular el residuo local:

$$r_i = (b - Ax)_i \tag{3.40}$$

5.- Resolver

$$A_i \delta_i = r_i \tag{3.41}$$

6.- Actualizar la solución:

$$x_i = x_i + \delta_i \tag{3.42}$$

7.- EndIf

siendo numcolor el número total de colores.

Esta implementación puede permitir también solape entre los dominios, un parámetro de relajación w, etc.

Un problema asociado con el ordenamiento multicolor es el hecho de que si los subdominios asociados con un color dado están activos, todos los otros subdominios deberán estar inactivos, limitando la eficiencia alcanzable por 1/numcolor. Para solucionar esto, es posible dividir la matriz local en dos bloques, el primero asociado a los nodos internos y el segundo asociado a los nodos frontera.

De este modo para la matriz local A_i se obtiene,

$$A_i = \begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} = \begin{pmatrix} B_i & 0 \\ 0 & C_i \end{pmatrix} + \begin{pmatrix} 0 & E_i \\ F_i & 0 \end{pmatrix} \tag{3.43}$$

Utilizando esta técnica las computaciones que involucran a los nodos internos de cada subdominio pueden ser realizadas en paralelo, de modo que el límite de la eficiencia antes establecido se aumenta.

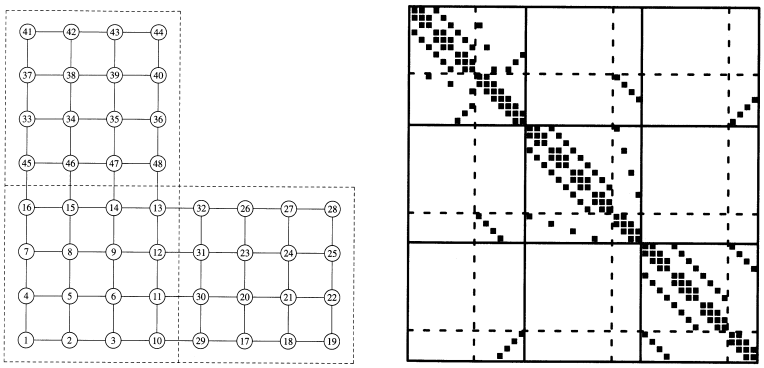
Desde el punto de vista matemático la diferencia existente entre las iteraciones por bloques de Gauss–Seidel y Jacobi es mínima. Gauss–Seidel actualiza inmediatamente (nada más obtenerlas) las componentes corregidas en el paso i de una cierta iteración y utiliza la solución aproximada actualizada para computar el residuo necesario para corregir las siguientes componentes a calcular en esa iteración. Mientras tanto, la iteración de Jacobi utiliza la misma aproximación antigua del vector solución durante toda la iteración.

3. Métodos basados en el complemento de Schur

   Estas técnicas se refieren a métodos que sólo trabajan sobre las variables de interfaz, utilizando implícitamente las variables internas como variables intermedias. En este caso se parte de un particionamiento de la matriz basado en vértices. Se denominan aristas de interfaz todas las aristas que unen vértices que no pertenecen al mismo subdominio, siendo los vértices de interfaz aquellos que en un subdominio dado son adyacentes a una arista de interfaz. Un vértice no es compartido por 2 particiones con la excepción de que exista solapamiento entre subdominios. Este tipo de particionamiento es totalmente diferente al realizado en la figura 3.7(a), en la que si dos vértices están acoplados tienen que pertenecer al mismo subdominio (particionamiento basado en arista).

   Un ejemplo de particionamiento basado en vértices se encuentra en la figura 3.10(a), los vértices de interfaz para el subdominio 1 (parte inferior de la figura, cuadrado izquierdo) son aquellos etiquetados desde el 10 al 16. La matriz resultante (figura 3.10(b)) es diferente de la obtenida con un particionamiento por aristas (gráfica 3.7), puesto que en esta ocasión los nodos de interfaz no se numeran al final en el etiquetado global, sino que se numeran como los últimos nodos de cada subdominio.

   Es posible escribir el sistema basado en este nuevo etiquetado. La matriz asociada con el particionamiento de las variables en subdominios



(a) (b)

Figura 3.10: (a)Malla asociada a un subdominio dividido en tres subdominios según un particionamiento basado en vértice, (b) matriz asociada a la malla anterior.

tendrá una estructura de bloques con un número de subdominios s. Tomando como ejemplo la figura 3.10(b), $s = 3$ y la matriz tendrá una estructura de bloques de la forma:

$$A = \begin{pmatrix} A_1 & A_{12} & A_{13} \\ A_{21} & A_2 & A_{23} \\ A_{31} & A_{32} & A_3 \end{pmatrix} \tag{3.44}$$

En cada subdominio las variables serán:

$$z_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{3.45}$$

representando x_i los nodos internos e y_i los nodos de interfaz asociados con el subdominio i. Cada matriz A_i será la matriz local del subdominio i, siendo su estructura:

$$A_i = \begin{pmatrix} B_i & E_i \\ F_i & C_i \end{pmatrix} \tag{3.46}$$

donde B_i representa la matriz asociada con los nodos internos del subdominio i, E_i y F_i representan los acoplamientos de los nodos internos con los nodos de interfaz, por último C_i es la parte local de la matriz de interfaz C y representa el acoplamiento entre los nodos locales de interfaz.

Observando la matriz de la figura 3.10(b) se encuentra que en la estructura de los bloques A_{ij} con $j \neq i$ se encuentra un sub–bloque de ceros en la parte que actúa sobre la variable x_j. Esto es lo esperado puesto que x_i y x_j no están acoplados. Por lo tanto:

$$z_i = \begin{pmatrix} 0 \\ E_{ij} \end{pmatrix} \tag{3.47}$$

La mayoría de las matrices E_{ij} son cero, puesto que sólo los índices j de los subdominios que tengan acoplamiento con el subdominio i presentarán un $E_{ij} \neq 0$. Por lo tanto la parte del sistema lineal que es local al subdominio i es de la forma:

$$B_i x_i + E_i y_i = f_i \tag{3.48}$$

$$F_i x_i + C_i y_i + \sum_{j \in N_i} E_{ij} y_j = g_i \tag{3.49}$$

El término $E_{ij}y_j$ es la contribución a la ecuación del subdominio vecino j y N_i es el conjunto de subdominios adyacentes al subdominio i. Además, es posible eliminar la variable x_i del sistema, extrayéndola de la ecuación 3.48:

$$x_i = B_i^{-1} \left(f_i - E_i y_i \right) \tag{3.50}$$

y sustituyéndola en la ecuación 3.49:

$$\left(C_i - F_i B_i^{-1} \right) y_i + \sum_{j \in N_i} E_{ij} y_j = g_i - F_i B_i^{-1} f_i \qquad i = 1, ..., s \tag{3.51}$$

Al primer término de la ecuación 3.51 se le denomina matriz de complemento de Schur local $S_i = C_i - F_i B_i^{-1} E_i$. Tras resolver la ecuación anterior se obtendrían los valores en los nodos de la interfaz y_i para una cierta iteración $(k+1)$, utilizando valores de los nodos frontera obtenidos en la iteración anterior:

$$y_i^{(k+1)} = S_i^{-1} \left[g_i - F_i B_i^{-1} f_i - \sum_{j \in N_i} E_{ij} y_j^{(k)} \right] \tag{3.52}$$

que son sustituidos a continuación en la ecuación 3.50 para obtener los valores de las variables internas x_i en la iteración $(k+1)$. Por lo tanto el procedimiento a seguir comprende tres pasos básicos:

a) Calcular el vector independiente $g' = g - FB^{-1}f$.

b) Resolver el sistema reducido a través de un método iterativo, como puede ser un método basado en subespacios de Krylov (por ejemplo GMRES), obteniendo así el vector y.

c) Obtener x vía $x = B^{-1}(f - Ey)$.

Para todos los subdominios las ecuaciones 3.51 se convierten en un sistema de ecuaciones que implican sólo a los puntos de interfaz y_j con $j = 1, 2, ..., s$, siendo s el número de subdominios.

Este sistema presenta una estructura por bloques asociada al vector de incógnitas de cada subdominio, donde los bloques de la diagonal, conocidos como matrices S_i, son generalmente densos mientras que los bloques E_{ij}, correspondientes a la relación entre incógnitas del subdominio i–ésimo con las del j–ésimo, son dispersos. De hecho $E_{ij} \neq 0$ sólo si existe alguna ecuación que los acople. La estructura de dicha matriz sería la siguiente:

$$S = \begin{pmatrix} S_1 & E_{12} & E_{13} & \dots & E_{1s} \\ E_{21} & S_2 & E_{23} & \dots & E_{2s} \\ E_{31} & E_{32} & S_3 & \dots & E_{3s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ E_{s1} & E_{s2} & E_{s3} & \dots & S_s \end{pmatrix} \tag{3.53}$$

La estructura del complemento global de Schur S se obtiene teniendo en cuenta que para particionamientos basados en vértice, la matriz de complemento de Schur se puede formar a partir de las matrices de complemento de Schur local (las S_i) y la información interfaz–interfaz (los E_{ij}).

3.3.5. Técnicas de reordenamiento

En la resolución de sistemas lineales dispersos usando precondicionadores basados en factorizaciones incompletas, es necesario tener en cuenta el reordenamiento de las variables con el fin de reducir el número de nuevas entradas en la matriz (*llenado*), disminuir el tiempo de computación, etc.

Existen multitud de técnicas de reordenamiento, tanto simétricas, en las que la misma permutación se aplica sobre las filas y columnas de la matriz, como no simétricas, sin que se pueda afirmar que un método dado sea el más adecuado para todos los casos [80].

En el simulador se han utilizado las funciones de reordenamiento de matrices que incluye el paquete METIS [62] con el fin de obtener una permutación que permita reducir el *llenado*. Estas funciones están basadas en *nested dissection multinivel* [81, 82]. Este algoritmo se basa en realizar sobre el grafo de adyacencias de la matriz una operación de partición, buscando un conjunto pequeño de nodos tales que su eliminación del grafo de adyacencias deje a este dividido en dos partes sin conexión entre sí. Seguidamente se reetiquetan los nodos de cada grupo consecutivamente, dejando para el final los nodos que pertenecen al separador. Estos nodos son los únicos adyacentes a la vez a miembros de los dos subgrafos. A continuación se procede recursivamente sobre cada uno de los subgrafos del mismo modo. Tras aplicar este algoritmo se obtiene una nueva matriz que presenta un patrón de dispersión bastante alto, con una forma de "punta de flecha".

3.3.6. Técnicas de almacenamiento de matrices dispersas

En el caso de las matrices dispersas un factor muy importante a tener en cuenta es el modo de almacenamiento. Una matriz densa $N \times N$ se suele almacenar en un vector bidimensional de dimensiones $N \times N$. Sin embargo, si la matriz es dispersa, este almacenamiento desperdicia mucho espacio en memoria porque la mayoría de los elementos de la matriz son nulos y no necesitan ser almacenados explícitamente, es común almacenar sólo las entradas diferentes de cero añadiendo información sobre la localización de estas entradas. Existen muchos esquemas de almacenamiento de matrices dispersas [83], entre los más utilizados están el CRS (Compressed Row Storage), el CCS (Compressed Column Storage) y el formato MSR (Modified Compressed Sparse Row). La elección del esquema más adecuado dependerá de las características del problema a resolver y del patrón (situación de las entradas no nulas) de la matriz. Para ello se toma como ejemplo la matriz representada a continuación, cuya dimensión es $8x8$ y el número de elementos no nulos α=15.

$$A = \begin{pmatrix} 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 3 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & 6 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 \\ 9 & 10 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 0 & 12 & 0 & 0 & 0 & 13 & 0 \end{pmatrix} \tag{3.54}$$

Formato CRS (Compressed Row Storage)

El esquema CRS representa la matriz por medio de tres vectores (Da, Colind y Rowptr). El vector Da almacena las entradas no nulas de la matriz al recorrerla por filas, Colind almacena el índice de columna de cada una de las entradas y Rowptr almacena la posición del vector Da en la que empieza cada nueva fila. El resultado del almacenamiento de la matriz en tres vectores se representa en la tabla 3.1.

Este tipo de almacenamiento representa un ahorro considerable de memoria pues para almacenar una matriz de orden $N \times N$ no se necesitan N^2 posiciones de memoria, sino únicamente $2\alpha + N + 1$.

Da	1 2 2 3 4 5 6 6 7 8 9 10 11 12 13
Colind	1 4 6 8 2 5 4 5 7 8 1 2 8 3 7
Rowptr	1 3 5 6 7 10 11 14 16

Tabla 3.1: Modo de almacenamiento del formato CRS para la matriz dispersa.

Formato CCS (Compressed Column Storage)

El esquema CCS representa la matriz por medio de tres vectores (Da, Rowind y Colptr). El vector Da almacena las entradas no nulas de la matriz al recorrerla por columnas, y en este caso Rowind almacena el índice de fila de cada una de las entradas y Colptr almacena la posición del vector Da en la que empieza cada nueva columna. El resultado del almacenamiento de la matriz en tres vectores se representa en la tabla 3.2.

Da	1 9 4 10 12 2 6 5 6 2 7 13 3 8 11
Rowind	1 7 3 7 8 1 5 4 5 2 5 8 2 6 7
Colptr	1 3 5 6 8 10 11 13 16

Tabla 3.2: Modo de almacenamiento del formato CCS para la matriz dispersa.

Formato MSR (Modified Compressed Sparse Row)

El esquema MSR representa la matriz por medio de dos únicos vectores (Da, Index). El vector Da almacena las entradas de la matriz, empezando

por toda la diagonal, dejando a continuación una entrada en blanco (en la tabla 3.3 marcado con -1), para seguir con el resto de los valores no nulos al recorrer la matriz por filas. El vector Index almacena en las $N+1$ primeras entradas la posición del dato en el vector Da que comienza cada una de las filas, y a continuación la columna correspondiente a cada dato de las entradas no diagonales del vector Da. El resultado del almacenamiento de la matriz en los dos vectores se representa en la tabla 3.3.

Da	1 0 0 0 6 0 0 0 -1 2 2 3 4 5 6 7 8 9 10 11 12 13
Index	10 11 13 14 15 17 18 21 21 4 6 8 2 5 4 7 8 1 2 8 3 7

Tabla 3.3: Modo de almacenamiento del formato MSR para la matriz dispersa.

Formato HB (Harwell Boeing)

Este formato de almacenamiento [84] está basado en el CCS. A la matriz almacenada en formato CCS se le añaden simplemente unas cabeceras que contienen información relativa al formato de almacenamiento y a los requerimientos de espacio. Suponiendo que no se almacena el vector independiente, la matriz en formato H/B estará formada por tres bloques de datos consecutivos que contendrán los vectores Da, Rowind y Colptr, y cuatro líneas iniciales de cabecera que nos darán el número de líneas ocupadas por cada uno de los vectores, el número total de filas y de elementos no nulos, el tipo de matriz, etc.

Si también se almacena el vector independiente habría que añadir un cuarto bloque de datos y una quinta línea de cabecera. Este bloque contendría los valores numéricos del vector independiente (aunque también se podría escoger almacenarlo en el mismo formato utilizado para la matriz), y la nueva línea de cabecera informaría de la dimensión del vector independiente, del número de filas que ocupa y del formato escogido para su almacenamiento.

3.4. Estructura del simulador 3D paralelo

Es posible dividir el código del simulador 3D en tres partes diferentes: preprocesado, procesado y postprocesado. De estas tres etapas, sólo la etapa de procesamiento se realiza completamente en paralelo. A continuación

se describirán detalladamente cada una de las tres etapas del proceso de simulación.

3.4.1. Preprocesado

Inicialmente en la etapa de preprocesado, se genera la malla y los ficheros de entrada, que contienen información sobre parámetros geométricos, físicos y de polarización del dispositivo que serán considerados posteriormente en la simulación.

Las características geométricas del dispositivo fueron utilizadas previamente por los programas QMG o MMG para la generación de la malla de elementos finitos. Además, también son usadas, junto con las características físicas para asignarle el valor de los parámetros de entrada a cada nodo de la malla.

La entrada de datos se realiza a través de un fichero de texto que se divide en una serie de partes. En primer lugar se define el material de referencia, teniendo en cuenta que las características eléctricas del dispositivo no dependen del material elegido como referencia, proporcionando las siguientes constantes relativas a dicho material: temperatura, concentración intrínseca, densidades efectivas de estados, afinidad electrónica y anchura de la banda prohibida.

Una vez fijado el material de referencia se introducen las características de cada región del dispositivo indicándose para cada zona: el material que lo forma, el perfil y el tipo de dopado, las movilidades de los portadores para cada tipo de portador, la constante dieléctrica relativa del material, la afinidad electrónica, la anchura de la banda prohibida, la masa efectiva para la densidad de estados de electrones, la masa efectiva para la densidad de estados de huecos, etc. A continuación estas magnitudes son escaladas y se almacena su valor en cada uno de los nodos.

Por último, para poder polarizar el dispositivo, se especifican en otro fichero el número de potenciales y su valor en cada uno de los contactos para todas las polarizaciones a estudiar, además de especificar el incremento de la polarización para cada uno de ellos.

3.4.2. Procesado

Durante esta etapa se procede a resolver las ecuaciones que modelan el comportamiento del dispositivo. Un diagrama de flujo de este proceso se muestra en la figura 3.11.

En primer lugar se calcula una solución en equilibrio térmico para el potencial electrostático. Seguidamente se polariza el dispositivo, para lo cual

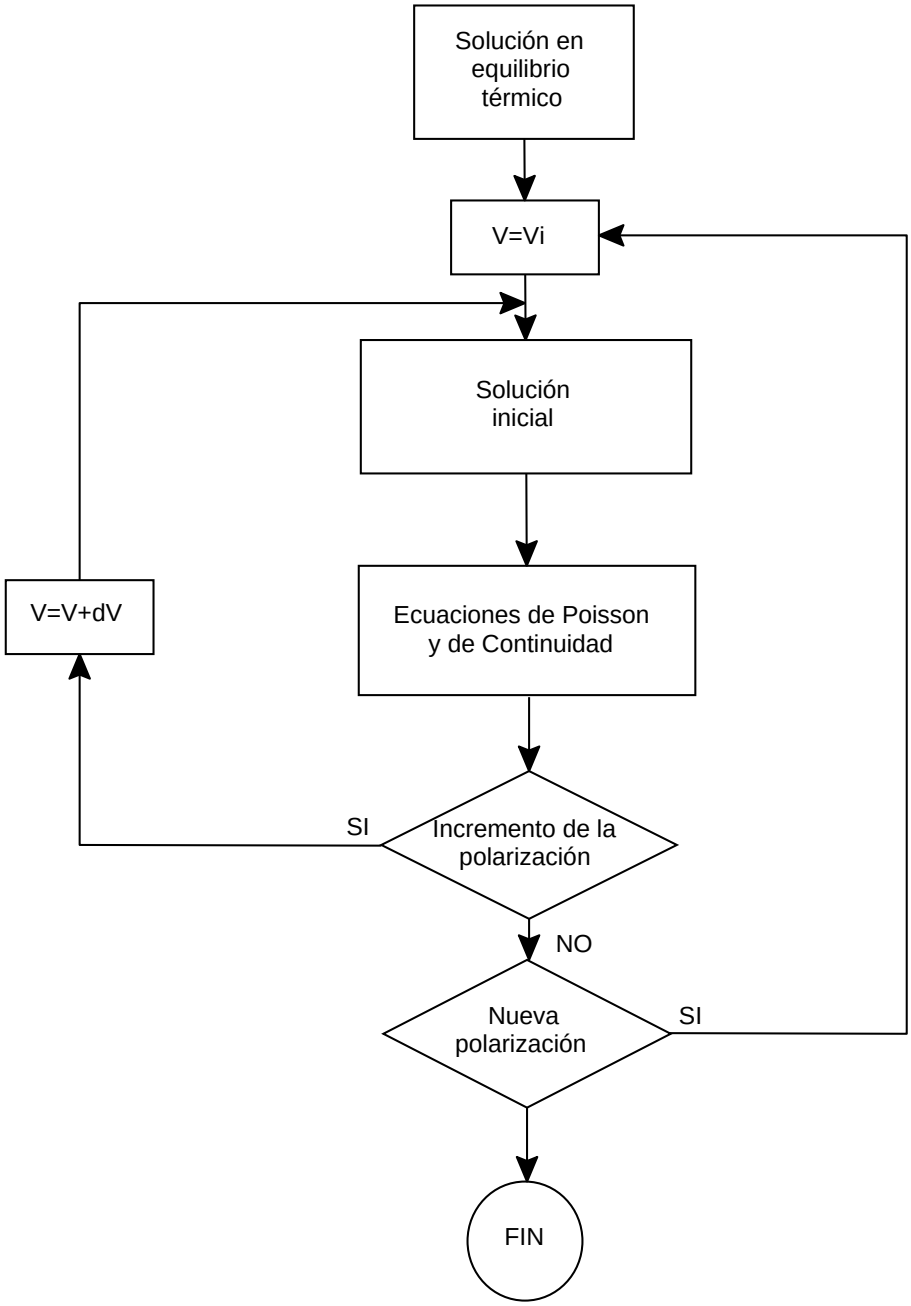


Figura 3.11: Diagrama de flujo de la etapa de procesamiento del simulador 3D paralelo basado en el modelo de arrastre–difusión.

se van produciendo pequeños incrementos de las tensiones en cada uno de los contactos hasta alcanzar el valor de la polarización. Para cada una de estas tensiones se calcula una solución inicial que se utilizará en la resolución de

las ecuaciones de Poisson y de continuidad de electrones y huecos. Una vez acabada esta etapa, si no se ha alcanzado el valor de polarización, se procede a incrementar el valor de la tensión en los contactos. En caso contrario se comprueba si hay que alcanzar una nueva polarización y entonces, se continúa todo el proceso o se finaliza la parte de procesamiento.

Solución en equilibrio térmico

Antes de polarizar el dispositivo se procede a calcular una solución en equilibrio térmico para lo cual, en primer lugar, se calcula una solución aproximada para el potencial electrostático, que servirá para resolver posteriormente la ecuación de Poisson, la cual dará el valor del potencial en equilibrio térmico.

El cálculo de la solución aproximada está basado en suponer que la concentración de portadores mayoritarios coincide con el dopado en el nodo en el que se va a calcular el valor del potencial ψ.

Bajo esta aproximación si se supone que la región es de tipo N, a una temperatura de 300 K, y la concentración de electrones es aproximadamente igual al dopado, puesto que en equilibrio $\phi_n = 0$, a partir de la expresión 2.45 se tiene que:

$$N_D = n = n_{ien} \exp\left(\frac{q\psi}{KT}\right) \tag{3.55}$$

teniendo en cuenta que la concentración intrínseca efectiva de los electrones, calculada en la ecuación 2.47, es:

$$n_{ien} = n_{io} \frac{N_c}{N_{c0}} \exp\left(\frac{\chi - \chi_0}{KT}\right) \frac{F_{1/2}(\eta_c)}{\exp(\eta_c)} \tag{3.56}$$

donde η_c se puede expresar en función del potencial electrostático como,

$$\eta_c = -ln\frac{N_{c0}}{n_{i0}} + \left(\frac{\chi - \chi_0}{KT}\right) + \frac{q\psi}{KT} \tag{3.57}$$

A partir de estas expresiones se puede sustituir la ecuación 3.56 en la ecuación 3.55. Además, utilizando la expresión 3.57 para el término exponencial, es posible despejar el valor de la integral de Fermi–Dirac de orden 1/2 resultando:

$$F_{1/2}(\eta_c) = \frac{N_D}{N_C} \tag{3.58}$$

Utilizando las aproximaciones para la integral de Fermi de orden 1/2 de la tabla 3.4 [85], la ecuación no lineal 3.58 se resuelve analíticamente cuando $|\eta| \geq 10$, y en el otro caso se aplica el método de Newton.

$\eta \leq -10$	$\exp(\eta)$
$-10 < \eta < 10$	$\exp(-0.3288 + 0.7404\eta - 0.0454\eta^2 - 8.79710^{-4}\eta^3 + 1.511710^{-4}\eta^4)$
$10 \leq \eta$	$4\frac{\eta^{3/2}}{3\sqrt{\pi}}$

Tabla 3.4: Aproximaciones para la integral de Fermi–Dirac de orden 1/2.

A partir de esa solución y teniendo en cuenta la expresión 3.57, es posible obtener una primera aproximación para el potencial en equilibrio para cada nodo ψ_i.

El valor obtenido del potencial con esta primera aproximación no es lo suficientemente preciso como para poder ser utilizado como potencial en equilibrio por lo que se va a aplicar una segunda aproximación, la cual está basada en resolver la ecuación de Poisson.

Teniendo en cuenta que el cuasipotencial de Fermi de electrones es igual a cero en el equilibrio y usando el valor obtenido en esta primera aproximación para el potencial se resuelve la ecuación de Poisson en el dispositivo en equilibrio. De este modo se procede a resolver la ecuación de Poisson discretizada y linealizada usando el método de Newton hasta obtener la convergencia.

Estrategias de aproximación inicial

Dado que la convergencia de los métodos de resolución de ecuaciones no lineales tipo Newton dependen fuertemente de que la solución inicial escogida esté lo suficientemente próxima a la solución final, es muy interesante buscar esta aproximación inicial a la solución. Además en el caso del método de Newton nos garantiza la convergencia cuadrática. Estos métodos y algunas de sus propiedades se han tratado en la sección 3.2.1.

En la rutina de aproximación inicial se calculan los incrementos del potencial y de los cuasipotenciales de huecos y electrones cada vez que se aplica un incremento de la tensión en los contactos, de modo que sumando el resultado obtenido al valor previo del potencial se obtiene la aproximación inicial a la solución buscada. Esta aproximación está basada en despreciar el incremento de recombinación producido por el aumento de las tensiones aplicadas a los contactos del dispositivo y considerar que no varía la concentración de mayoritarios.

Para una cierta polarización del dispositivo hay asociado a cada nodo un valor del potencial y de los cuasipotenciales de huecos y de electrones. Al aumentar las tensiones en los contactos ΔV_S, ΔV_G y ΔV_D, se busca una aproximación a la solución mediante la resolución secuencial de dos sistemas

de ecuaciones lineales, uno para los incrementos del cuasipotencial de Fermi de electrones y otro para los incrementos del cuasipotencial de Fermi de huecos.

Estos sistemas se crean de modo similar al usado para resolver las ecuaciones de continuidad, pero utilizando los incrementos de las densidades de corriente ΔJ_n, ΔJ_p y ΔR en lugar de utilizar las magnitudes totales y haciendo uso de algunas aproximaciones numéricas. Además, se puede suponer que el incremento de la recombinación es nula, por lo que se cumple que $\Delta R = 0$. Teniendo en cuenta las consideraciones anteriores se obtiene un sistema de ecuaciones lineales.

Una vez resueltas las ecuaciones anteriores y obtenido la aproximación para los cuasipotenciales de Fermi en la nueva polarización, se procede a realizar una aproximación para el potencial electrostático suponiendo que la concentración de mayoritarios no se ve modificada por el incremento de la polarización.

Solución de las ecuaciones de Poisson y de continuidad

Para una polarización dada y a partir de la aproximación inicial a la solución calculada anteriormente se procede a resolver las ecuaciones de Poisson 2.101 y de continuidad de electrones 2.121 y de huecos 2.122, discretizadas en la sección 2.3, utilizando el método iterativo de Gummel (explicado en la sección 3.2.2).

Aplicando el método de Gummel se desacoplan las ecuaciones de Poisson y de continuidad de electrones y de huecos, y se resuelven por separado de forma iterativa hasta obtener la convergencia de las tres ecuaciones a la vez. En el caso particular de la aplicación del simulador al estudio estadístico de la influencia de las fluctuaciones de parámetros intrínsecos, en el que los requerimientos computacionales son extremadamente elevados, es posible prescindir de la resolución de la ecuación de continuidad de huecos y de la recombinación, y resolver únicamente las ecuaciones de Poisson y de continuidad de electrones.

El algoritmo alcanza la convergencia si al final se llega a la solución del sistema de ecuaciones planteado. Para cumplir este objetivo se pueden utilizar varias estrategias [29, 86]. En general no basta con observar si el incremento de las variables es menor en la última iteración que en la anterior e ir repitiendo el proceso, sino que también se tiene que comprobar que la sustitución de los nuevos valores de las variables en las ecuaciones provoca que el error sea siempre menor. Con el fin de intentar cumplir estos objetivos se utilizan diferentes criterios. En primer lugar se limita el valor máximo del

incremento de las variables en la nueva iteración al valor obtenido en la iteración anterior. Además, si el error en la nueva iteración es superior al anterior se utiliza un factor de amortiguamiento en la solución t_k en el paso k–ésimo, con $t_k \in (0, 1]$ calculado como,

$$t_k = \frac{1}{2^i} \qquad i = 0, 1, 2, \ldots \tag{3.59}$$

Además, es preciso tener en cuenta que si se utiliza un paso muy alto en los incrementos de la polarización o si la malla de elementos finitos no es de buena calidad, por ejemplo por presentar pocos nodos o por tener tetraedros con ángulos obtusos, se pueden producir soluciones espúreas, sobre todo en el caso de las ecuaciones de continuidad, por lo cual también se limita el incremento máximo de la solución al valor del incremento del potencial aplicado.

Se utilizan diferentes estrategias para fijar el criterio de convergencia de la solución, como exigir que la norma del vector residual o del vector incremento de la solución sea menor que el valor de un parámetro dado, y que no se supere un número máximo de iteraciones para cada una de las ecuaciones ni para el sistema desacoplado.

Resolución de los sistemas de ecuaciones lineales

Tanto en el caso de la aproximación lineal, como en el de las ecuaciones de Poisson y de continuidad de huecos y de electrones, es preciso resolver un sistema lineal de ecuaciones de dimensión el número de nodos de la malla de elementos finitos utilizada.

Para poder realizar esto se ha utilizado la librería PSPARSLIB [87]. Los motivos para la elección de esta librería y sus principales características se explican en la sección 4.2.1.

En el simulador se han implementado varios métodos de resolución, y se han seleccionado para su uso el método de Schwarz aditivo y el método del complemento de Schur, debido a que son los más eficientes para resolver los sistemas lineales que surgen de la simulación. En cualquiera de los casos en los que hay que resolver sistemas de ecuaciones lineales es posible seleccionar el tipo de resolutor que se desea utilizar y, además, asignarle diferentes parámetros como, por ejemplo, el nivel de llenado o la dimensión del subespacio de Krylov. De este modo es posible seleccionar el resolutor y los parámetros que mejor se adapten a cada caso.

3.4.3. Postprocesado

Durante esta etapa se procede a analizar y visualizar los datos obtenidos por el simulador. Estos datos son grabados en diferentes ficheros. El usuario también puede escoger guardar los datos que más le interesan y evitar de esta manera generar demasiada información no útil.

Entre otros parámetros se muestran las densidades de corriente en todos los contactos para cada unos de los incrementos de la polarización, los parámetros de pequeña señal para las tensiones que se indiquen en el fichero de polarizaciones, los valores del potencial, de los cuasipotenciales de Fermi de los portadores, concentraciones, etc., en cada nodo del dispositivo, en un plano de corte, o a lo largo de una línea.

Los valores anteriores se graban en diferentes ficheros en formato ASCII, fácilmente modificables, para poder ser procesados posteriormente por programas gráficos, como por ejemplo Tecplot [88].

El simulador permite además hacer un seguimiento opcional por el usuario de los pasos necesarios para alcanzar la solución, como por ejemplo, el número de iteraciones, las normas de los incrementos de la solución y del error, etc. Además, es posible guardar los valores de la matriz de coeficientes y del vector independiente en un momento dado de la simulación, lo cual resulta útil para analizar las propiedades de dichos sistemas.

3.5. Resumen

Una vez obtenidas, en el capítulo 2, las expresiones matemáticas que componen el modelo de arrastre–difusión, en este capítulo se ha descrito la implementación paralela de estas ecuaciones en el simulador. Para ello ha sido necesaria la introducción de las técnicas de mallado y particionamiento de las mallas tetraédricas de elementos finitos que representan el problema a estudiar. A continuación, se han mostrado las diferentes estrategias utilizadas en la linealización de los sistemas de ecuaciones no–lineales acoplados que forman estas ecuaciones, como son el método de Newton–Raphson y el método de Gummel, y en la posterior resolución de los sistemas de ecuaciones lineales generados con el proceso de linealización. Para ello se han presentado los métodos de resolución directos e iterativos, las principales técnicas de precondicionamiento aplicadas como complemento a los métodos iterativos y se ha introducido el concepto de técnicas de reordenamiento de las matrices dispersas, muy útiles para acelerar el proceso de resolución de los sistemas lineales. Como conclusión del capítulo se ha descrito brevemente la implementación en el simulador de las ecuaciones de arrastre–difusión y

de su proceso de resolución.

Capítulo 4

Optimización del simulador 3D paralelo basado en arrastre–difusión

Uno de los principales problemas relacionados con la simulación de dispositivos semiconductores, sobre todo cuando se requiere alta precisión y simulaciones tridimensionales, es su elevado coste computacional. Esto es debido a la alta cantidad de datos a procesar y a la lentitud de las técnicas numéricas empleadas. Además, en este tipo de simulaciones es necesaria la disponibilidad de una gran cantidad de memoria. En general las estaciones de trabajo comunes tienen problemas para llevar a cabo este tipo de simulaciones correctamente. Además, si es necesario realizar análisis estadísticos, el coste computacional se multiplica por el tamaño de la muestra estadística. Para resolver este problema es fundamental el uso de máquinas paralelas y algoritmos apropiados de tal forma que se obtenga el máximo rendimiento posible con una precisión adecuada.

En este trabajo se utiliza un simulador 3D paralelo de dispositivos HEMT en el estudio de la influencia de las fluctuaciones de parámetros intrínsecos en las curvas características de los dispositivos y en los parámetros de pequeña señal. Para ello, es necesario llevar a cabo estudios estadísticos en los que el número de simulaciones a realizar es muy elevado. Todo esto hace necesaria la optimización del simulador de tal forma que se minimice el tiempo de ejecución y el consumo de memoria. El uso del paralelismo en este caso es sumamente necesario puesto que permite reducir el tiempo de computación del problema, ya que la idea básica del procesamiento en paralelo consiste en la subdivisión del problema en un conjunto de partes resolubles de forma concurrente, de manera que el tiempo total de resolución del problema quede

dividido por el número de procesadores utilizados. El grado de consecución de este objetivo depende básicamente de dos factores, la sobrecarga computacional generada por la paralelización del problema y el equilibrio de carga conseguido entre el conjunto de procesadores.

Por lo tanto, se intentan alcanzar dos objetivos principalmente. En primer lugar se trata de encontrar los algoritmos de resolución de sistemas lineales más apropiados para nuestro problema en particular, de tal forma que se minimice el tiempo de ejecución. Una vez conseguido, en segundo lugar se analizan diversos procedimientos para la optimización del simulador tridimensional de tal forma que se mejore la eficiencia paralela del proceso de simulación.

En este capítulo inicialmente se mencionan las características del computador paralelo utilizado en la obtención de todos los resultados presentados en este trabajo, a continuación se muestra un análisis de algunos de los métodos de resolución y técnicas de precondicionamiento disponibles para la solución de los sistemas de ecuaciones lineales dispersos a resolver en el simulador de dispositivos, así como de los parámetros que tienen una mayor importancia en el tiempo de ejecución. Por último se presentan dos estrategias de optimización del simulador 3D paralelo de dispositivos HEMT basado en el modelo de arrastre–difusión. En primer lugar se trata de optimizar la etapa de resolución de los sistemas lineales de ecuaciones implementada en el simulador, puesto que esta es la etapa más costosa de toda la simulación. En segundo lugar se presenta una nueva estrategia de particionamiento de las mallas de dispositivos HEMT utilizadas, de tal forma que se tenga en cuenta su comportamiento físico.

4.1. Supercomputadores Paralelos

A lo largo de los años en los que se ha desarrollado este trabajo se han utilizado varios supercomputadores paralelos, pertenecientes a diferentes instituciones.

En trabajos relacionados con el estudio de los métodos de resolución de sistemas lineales dispersos en el contexto de la simulación de dispositivos semiconductores [89, 90, 91] se ha utilizado un cluster Beowulf, perteneciente al CESGA (Centro de Supercomputación de Galicia) [92]. Este cluster estaba formado por 16 procesadores Pentium III a 1GHz, cada uno de ellos con 512MB de RAM. Las comunicaciones se realizaban a través de una red Myrinet 2000.

Además de este computador, también se han utilizado, en estudios iniciales de la eficiencia paralela del simulador de dispositivos [93], una máqui-

na Sun Fire E15K, perteneciente al EPCC (Edinburgh Parallel Computing Centre) [94], y el supercomputador Mare Nostrum perteneciente al BSC (Barcelona Supercomputer Centre) [95].

En la calibración de uno de los dispositivos HEMT utilizados en este estudio [96] se ha utilizado un cluster con 100 procesadores AMD Opteron a 2.2 GHz, perteneciente al grupo de modelado de dispositivos de la Universidad de Glasgow [97]. Por último, la máquina paralela más utilizada en este trabajo es el cluster HP Superdome, perteneciente al CESGA, cuyas principales características serán descritas en el siguiente apartado.

4.1.1. Cluster HP Integrity Superdome

Todos los resultados numéricos que se presentan a continuación, han sido obtenidos en un cluster HP Superdome formado por dos servidores HP Integrity Superdome, cada uno con 64 procesadores Itanium 2 a 1.5 GHz y 6 MBbytes de cache integrados en el propio chip. En total el sistema dispone de 384 Gbytes de memoria y 4.6 Terabytes para almacenamiento temporal o *scratch* repartidos en 128 discos SCSI, además de otros 16 discos de 72 Gbytes para el sistema operativo (1.1 Terabyte). El rendimiento pico del sistema es de 768 Gflops.

4.2. Análisis de los métodos de resolución de sistemas dispersos de ecuaciones lineales

En el estudio de las técnicas de resolución de sistemas lineales dispersos asociados con la simulación de dispositivos semiconductores hemos utilizado una serie de librerías numéricas paralelas, que implementan diversos métodos de resolución y técnicas de precondicionamiento. Inicialmente, en este apartado, se describen las principales características de cada una de las librerías numéricas utilizadas, para a continuación presentar un análisis de la eficiencia de algunos de los diferentes métodos de resolución implementados. Por último se muestran las principales conclusiones extraídas de este análisis.

4.2.1. Librerías numéricas

Existen un gran número de librerías numéricas para la resolución de sistemas lineales tanto densos como dispersos. Considerando las librerías especializadas en la resolución de sistemas dispersos, existen tanto librerías

secuenciales como paralelas. Ejemplos de librerías secuenciales son las librerías HSL [98], YSMP (Yale Sparse Matrix Package) [99] y SPARSKIT [100]. Por otro lado, entre las principales librerías paralelas que permiten la resolución de sistemas lineales dispersos se encuentran Aztec [101], BlockSolve [102], PETSc [103, 104], PSPARSLIB [87, 105], SuperLU [106], pARMS [107] y MUMPS [108].

En este trabajo se han utilizado solamente librerías paralelas, basadas algunas de ellas en métodos de resolución directos y otras en métodos iterativos. A continuación se describen brevemente las características principales de las librerías utilizadas en este trabajo.

SuperLU

La librería SuperLU fue desarrollada entre el NERSC (National Energy Research Scientific Computing Center) y la Universidad de Berkeley. Tiene como objetivo la resolución de una factorización LU completa en diversas arquitecturas. Se utiliza generalmente en la solución directa de grandes sistemas dispersos y no–simétricos de ecuaciones lineales en máquinas de alto rendimiento. Está formada por los siguientes módulos:

- SuperLU Secuencial: diseñada para procesadores secuenciales con uno o más niveles en la jerarquía de memoria (cachés).
- SuperLU Multithreaded: planteada para multiprocesadores de memoria compartida.
- SuperLU Distribuida: está diseñada para procesadores paralelos de memoria distribuida.

En nuestro caso se utilizará el módulo SuperLU Distribuido. Está implementado en ANSI C, aunque es sencillo compaginarlo con Fortran, y utiliza MPI para las comunicaciones, siendo su modelo de programación SPMD. La librería incluye rutinas que le permiten manejar matrices tanto reales como complejas en doble precisión. Incorpora una serie de ideas algorítmicas que explotan las características de las arquitecturas de computadores modernas, particularmente la organización multinivel de la caché. Puede utilizarse con un número de procesadores elevado, alcanzándose una tasa de factorización de 10.2 Gigaflops en un Cray T2E de 512 procesadores. A continuación se describe su algoritmo básico de funcionamiento:

1. Equilibrar la matriz A: computar las matrices diagonales D_r y D_c de tal forma que $\widehat{A} = D_r A D_c$ esté mejor condicionada, es decir que $\widehat{A}^{-1}$ sea menos sensible a perturbaciones de lo que sería A^{-1}.

2. Reordenar las filas de $\widehat{A}$: reemplazar $\widehat{A}$ por $\widehat{A'} = P_r\widehat{A}$, donde P_r es la matriz de permutación.

3. Ordenar las columnas de $\widehat{A'}$: para incrementar la dispersidad de los factores L y U computados, e incrementar el paralelismo. En otras palabras, reemplazar $\widehat{A'}$ por $\widehat{A''} = P_c\widehat{A}P_c^T$.

4. Computar la factorización LU de $\widehat{A''}$.

5. Resolver el sistema utilizando los factores triangulares computados.

6. Computar los márgenes de error.

La matriz de entrada A se encuentra distribuida entre los procesadores, que utilizan una distribución basada en bloques de filas. Es decir, cada procesador posee un bloque de filas consecutivas de A. En el caso de las matrices L y U se puede decir que están divididas entre todos los procesadores por medio de un mapeado cíclico por bloques.

El tamaño concreto de cada asignación de bloques a procesadores depende de la estructura de no ceros de la diagonal. Todos los bloques diagonales serán cuadrados y contendrán sólo elementos no nulos, requisito no exigible para los bloques no diagonales. Al utilizar un mapeado cíclico por bloques se desacoplan los procesadores en filas para la matriz L y en columnas para la matriz U. En este mapeado 2D, cada bloque de columna de L pertenece a más de un procesador. Además de los valores numéricos almacenados en un vector por columnas, *nzval*, es necesaria información para interpretar la localización y el subíndice de fila de cada no cero, esto se almacena en un vector de enteros llamado *index* que incluye información para la columna de bloques completa y para cada bloque individual de ella. Muchos bloques no diagonales son ceros y por lo tanto no son almacenados, y tampoco se incluyen los ceros en un bloque de no ceros. Por otro lado tanto los triángulos inferior y superior que forman los bloques de la diagonal se almacenan en la estructura de datos de L. Para U se utiliza un almacenamiento orientado por bloques filas, aunque los valores numéricos dentro de cada bloque siguen siendo por columnas. De forma similar a L también se emplean un par de vectores $indice - nzval$ para el almacenamiento de los bloques de filas de U.

MUMPS

MUMPS (Multifrontal Massively Parallel Solver) es una librería para la resolución de sistemas lineales de ecuaciones del tipo $Ax = b$, siendo A una matriz dispersa que puede ser no–simétrica, simétrica definida positiva,

o simétrica en general. MUMPS utiliza una técnica multifrontal que no es más que un método directo basado en la factorización de la matriz.

Este paquete permite la solución del sistema transpuesto, análisis de error, refinamiento adaptativo, escalado de la matriz original, y obtención de la matriz del complemento de Schur. Ofrece varios algoritmos propios de reordenamiento, una interfaz que permite utilizar paquetes externos de ordenamiento, como por ejemplo METIS, e incluso permite que el usuario proporcione su propio ordenamiento.

La librería está escrita en Fortran 90 y su versión paralela utiliza MPI para el paso de mensajes. MUMPS distribuye el trabajo entre los procesadores, pero un procesador maestro es necesario para realizar la mayoría de la fase de análisis, para la distribución de la matriz de entrada entre los procesadores esclavos en el caso de que esta esté centralizada y para recopilar la solución. El sistema lineal es resuelto en tres etapas:

- Análisis. El procesador maestro realiza un ordenamiento basado en el patrón $(A + A^T)$ y lleva a cabo una factorización simbólica. A continuación se realiza un mapeado del grafo computacional multifrontal y se transfiere la información simbólica desde el master hacia los otros procesadores. Utilizando esta información los procesadores estiman la memoria necesaria para la factorización y la solución.

- Factorización. La matriz original se distribuye a los procesadores que participan en la factorización numérica. La factorización en cada matriz es dirigida por el procesador maestro y por uno o más procesadores esclavos, determinados dinámicamente. Cada procesador destina un array para los bloques y los factores obtenidos, que deben conservarse para la fase siguiente.

- Solución. El vector independiente b es distribuido desde el maestro hacia los otros procesadores. Estos procesadores computan la solución x utilizando para ello los factores distribuidos que han sido calculados en la etapa anterior, y finalmente la solución es reconstruida en el master o permanece distribuida en los procesadores.

Tanto para los algoritmos simétricos y no–simétricos utilizados en el código se ha escogido una aproximación completamente asíncrona con una distribución dinámica de las tareas computacionales. La comunicación asíncrona es utilizada para permitir el solapamiento entre comunicaciones y computaciones. La asignación dinámica de tareas permite al algoritmo adaptarse en tiempo de ejecución y redistribuir trabajo y datos a los procesadores más apropiados. En realidad, se combinan las características básicas de las

aproximaciones dinámicas y estáticas, puesto que se utiliza la estimación obtenida durante la fase de análisis para mapear algunas de las principales tareas computacionales, mientras que las otras tareas se asignan en tiempo de ejecución. De igual forma, las principales estructuras de datos, como por ejemplo la matriz original, se mapean parcialmente de acuerdo con la fase de análisis.

PSPARSLIB

La librería PSPARSLIB es una librería de resolutores paralelos iterativos que proporciona una serie de módulos utilizados para simplificar el desarrollo y la implementación de resolutores iterativos dispersos en computadores de memoria distribuida. PSPARSLIB resuelve sistemas lineales dispersos que se encuentran distribuidos entre varios procesadores, pudiendo trabajar tanto con matrices simétricas o no–simétricas, incluso con patrones irregulares. Está escrita básicamente en Fortran aunque incluye un número pequeño de módulos en C, utilizando la librería MPI para paso de mensajes.

Hay dos formas distintas de preparar la matriz para los resolutores iterativos. En la primera, el particionamiento puede determinarse de antemano y cada nodo crear su propia parte local del sistema lineal. En la segunda, un nodo lee el sistema lineal completo, para a continuación particionar la matriz y distribuir el sistema entre los procesadores participantes.

La librería PSPARSLIB está compuesta de los siguientes módulos:

- Un particionador de grafos simple.
- Rutinas para reordenar y formar las matrices locales.
- Aceleradores Krylov: CG, GMRES, FGMRES, DQGMRES, BCGSTAB, QMR y TFQMR.
- Precondicionadores basados en descomposición de dominios, como el método de Jacobi por bloques, el método SOR multicolor o técnicas de complemento de Schur. Para cada uno de los precondicionadores es posible elegir si se desea que exista o no solapamiento.
- Herramientas de preprocesamiento, como pueden ser rutinas de particionamiento, de mapeado, rutinas de datos locales y algunas rutinas de color.

Realizando un estudio del código desarrollado en esta librería, el procedimiento a seguir sería el siguiente: el procesador maestro lee la matriz

completa, particiona el grafo y reparte las matrices locales a cada procesador. Una vez que cada uno recibe su submatriz, crean un vector de mapeado que contiene la lista nodo–procesador al que pertenece, determinan la información frontera, es decir, el número de procesadores adyacentes, la lista de procesadores vecinos, los nodos frontera internos ordenados por procesador, etc. Una vez formada la matriz local, que no será más que una forma reordenada de las ecuaciones iniciales, se hace una factorización incompleta LU de la matriz, como precondicionamiento. El paso siguiente es proponer una solución inicial y resolver el problema utilizando un resolutor, de los disponibles en la librería, junto con un precondicionador basado en descomposición de dominios.

Si como precondicionador se utiliza el método Schwarz aditivo con solapamiento, en la fase de solución se dan dos casos posibles:

- Si el número máximo de iteraciones del precondicionador, fijado como parámetro de entrada, es igual a cero, el resolutor interno resuelve simplemente una factorización ILU(fill,τ). Es decir se implementa una factorización incompleta LU en la que se permite controlar el llenado (fill) de la matriz, que tiene en cuenta además el parámetro τ que indica un valor mínimo a partir del cual se permite el llenado.

- Si el número máximo de iteraciones del precondicionador es mayor de cero se resuelve el problema utilizando GMRES precondicionado con ILU(fill,τ). Este caso se utiliza sólo cuando el resolutor elegido es el FGMRES, que permite la variación del precondicionador en cada paso.

Si por el contrario se utiliza como precondicionador el método SOR multicolor, se observa que el código es prácticamente idéntico al del caso anterior, aunque en este caso se utiliza la rutina *multicD*, que será la encargada del ordenamiento multicolor. El algoritmo empleado en la implementación de esta rutina está basado en una ordenación topológica tal que a nivel de programación el paralelismo sea del orden del diámetro del grafo. La rutina *multicD* proporciona el número de colores distintos asignados a los procesadores adyacentes y el color asignado al procesador local. En el caso del resolutor interno, se llaman a las rutinas *msorlu* (*msorlut*) que resuelven el sistema por medio de una factorización incompleta LU de la matriz (o de su traspuesta).

PETSc

PETSc (Portable Extensible Toolkit for Scientific Computing) es una librería utilizada en la resolución numérica de ecuaciones diferenciales par-

ciales y problemas similares en computadoras de alto rendimiento. Contiene un conjunto de estructuras y rutinas que combinadas componen los bloques empleados en implementación de códigos para aplicaciones a gran escala en ordenadores paralelos. Entre sus herramientas están incluidas un grupo de resolutores de ecuaciones lineales y no–lineales, que se pueden utilizar en códigos escritos en Fortran, C y C++. Utiliza paso de mensajes vía MPI y no asume compartición física de datos o un espacio de direcciones global. Alguno de los módulos de PETSc están relacionados con:

- Conjuntos de índices, incluyendo permutaciones, renombrado, etc.
- Vectores.
- Matrices (generalmente dispersas).
- Vectores distribuidos.
- Métodos de resolución de sistemas de ecuaciones lineales basados en subespacios de Krylov.
- Precondicionadores.
- Resolutores no–lineales.
- Marcadores de tiempo para resolver ecuaciones no lineales dependientes del tiempo.

Cada módulo consiste en una interfaz abstracta y una o más implementaciones usando estructuras de datos particulares, puede decirse que PETSc consiste en un conjunto de librerías (parecidas a las clases de C++), donde cada una de ellas manipula una familia de objetos (por ejemplo vectores), y las operaciones que se pueden realizar sobre estos objetos. Al estar escrita en un modelo orientado a objetos, todas las estructuras de datos están ocultas para el usuario. Su infraestructura crea una base para producir aplicaciones de gran escala, por lo cual es útil considerar las interrelaciones entre los diferentes módulos de PETSc.

Las matrices se encuentran almacenadas por defecto en formato RCS, aunque cabe la posibilidad de utilizar otros formatos, como pueden ser BCRS (Block Compressed Row Storage) o BDS (Block Diagonal Storage), que puedan resultar más eficientes en problemas con múltiples grados de libertad por nodo. En la distribución paralela de la matriz cada proceso posee localmente una submatriz formada por filas contiguas en la matriz global. Siempre hay que tener en cuenta que las estructuras de datos son internas, pasándose los distintos elementos a través de llamadas a funciones.

Para la resolución de sistemas de ecuaciones lineales el objeto más importante es SLES, puesto que proporciona un acceso uniforme y eficiente a los resolutores de sistemas lineales, paralelos y secuenciales, directos e iterativos. Como la base de la mayoría de los códigos actuales para la resolución iterativa de sistemas lineales se encuentra en la combinación de un método de resolución basado en subespacios de Krylov y un precondicionador, cada objeto SLES contiene normalmente a otros dos objetos:

- KSP (Krylov Space Method), formado por el método iterativo y cuyo contexto contiene información relacionada con el método elegido.
- PC (Preconditioners), contiene información sobre los parámetros relativos al precondicionador elegido.

Los métodos iterativos de los que dispone esta librería son: Richardson, Chebyshev, CG, GMRES, TCQMR, BCGS, CGS, TFQMR, CR y LSQR. Estos métodos iterativos se suelen emplear en combinación con un precondicionador, siendo posible utilizar: Jacobi, Jacobi por bloques, Gauss–Seidel por bloques (pero sólo en el caso secuencial), SOR, ILU (sólo en el caso secuencial), Schwarz aditivo, una factorización completa (también se encuentra sólo disponible en el caso secuencial), y un precondicionamiento proporcionado por el usuario. Por defecto, todas las implementaciones KSP utilizan precondicionamiento por la izquierda. También existe la posibilidad de utilizar un precondicionamiento combinado, es decir, utilizar una combinación de los precondicionadores o resolutores definidos para lograr una eficiencia mejor que la obtenida con un único método, aunque en muchos casos utilizar un solo precondicionador es mejor que una combinación de ellos.

Aztec

Aztec es una librería iterativa que busca simplificar el proceso de paralelización cuando se resuelven sistemas lineales de ecuaciones. Dispone de una serie de herramientas de transformación de los datos que permiten una rápida creación de matrices dispersas distribuidas para una solución paralela. El uso de una matriz distribuida global permite al usuario especificar fragmentos (diferentes filas para diferentes procesadores) de su matriz de aplicación de igual forma que si estuviera trabajando en un caso secuencial (es decir, utilizando un esquema de numeración global). Cuestiones como la numeración local o los mensajes son ignorados por el usuario, pero en cambio son computados por funciones de transformación automatizadas, así se obtiene un buen rendimiento utilizando técnicas estándar de memoria distribuida.

Las submatrices etiquetadas localmente y los mensajes informativos computados por la función de transformación son conservados por cada procesador para que las computaciones y comunicaciones de las dependencias de datos sean más rápidas.

La librería está escrita en ANSI C estándar, y aunque puede trabajar con matrices generales, el paquete fue diseñado para las matrices que surgen de la aproximación de ecuaciones diferenciales parciales (PDEs). Aztec puede trabajar con dos formatos específicos de matrices dispersas, el formato MSR (Modified Sparse Row) o el VBR (Variable Block Row).

Incluye una serie de métodos iterativos basados en subespacios de Krylov para la resolución de sistemas de ecuaciones, así dispone de los siguientes resolutores: CG, GMRES, CGS, TFQMR, BCGSTAB, LU (válida sólo en el caso secuencial). Los métodos iterativos son utilizados conjuntamente con varios precondicionadores como pueden ser: Jacobi por bloques, Gauss–Seidel, series polinomiales de Neumann y métodos basados en descomposición de dominios con solapamiento (Schwarz aditivo). Otra opción a tener en cuenta, en el caso de escoger como precondicionamiento métodos basados en descomposición de dominios, es el resolutor a utilizar en cada subdominio. Entre las posibilidades en esta situación se pueden destacar una factorización LU completa, una factorización incompleta LU con un cierto nivel de llenado y una factorización ILUT. Esta última factorización no utiliza las mismas definiciones vistas anteriormente, sino que usa dos criterios diferentes para determinar el número de no ceros a introducir en las factorizaciones aproximadas resultantes, por un lado el parámetro *ilut_fill* indica que la factorización resultante puede contener como máximo *ilut_fill* veces el número de no ceros de la matriz original. Por otro lado no se tienen en cuenta aquellos elementos de la factorización resultante cuyo valor sea inferior a un límite fijado (*drop*). Cuando este límite esté fijado a cero no se eliminará ningún elemento y sólo se tendrá en cuenta la contribución de *ilut_fill*. Sin embargo, la utilización del parámetro *drop* puede implicar que la matriz resultante contenga un número significativamente menor de elementos no nulos.

A continuación se describe los formatos de la matriz y de los vectores usados internamente por Aztec. El producto de la matriz dispersa por un vector $y = Ax$ es el principal núcleo de computación de esta librería. Para realizar esta operación en paralelo los vectores x e y así como la matriz A deben estar distribuidos a través de los procesadores. Cuando se realiza una operación que involucra a un vector, por ejemplo y, cada procesador computa sólo aquellos elementos (entradas particulares de un vector) de y que tiene asignados. Estos elementos del vector se encuentran almacenados explícitamente en el procesador y se definen por medio de un conjunto de índices a

Nombre	Nodos	Elementos	Núm.Condición	Norma Frobenius
Poisson_A	29,012	398,102	$1.3\,10^5$	$1.0\,10^3$
Electron_A	29,012	398,102	$2.4\,10^{21}$	$1.5\,10^4$

Tabla 4.1: Información general que caracteriza a las matrices Poisson_A y Electron_A.

los que se refieren como *conjunto de actualización* del procesador. El conjunto de actualización por su parte se divide en dos subconjuntos: *interno* y *frontera*. Un componente correspondiente a un índice en el subconjunto *interno* se actualiza usando sólo información perteneciente al propio procesador, en cambio el subconjunto *frontera* define elementos que requerirían valores de otros procesadores para poder ser actualizados durante el producto matriz–vector. El conjunto de índices que identifican los elementos exteriores al procesador necesarios para actualizar componentes del conjunto frontera se denominan *externos* y son obtenidos de otros procesadores vía comunicación mientras se realiza el producto matriz–vector.

En cuanto a las matrices, cada procesador almacena un subconjunto de los elementos no–nulos de la matriz. En particular, cada procesador almacena sólo aquellas filas que corresponden a su *conjunto de actualización*. Además el etiquetado local de los elementos de los vectores en un procesador específico induce un etiquetado de las filas y columnas de la matriz. Es decir, cada procesador contiene una submatriz cuyas entradas en filas y columnas corresponden a variables definidas en este procesador.

4.2.2. Resultados numéricos

Utilizando como punto de partida cada una de las librerías numéricas descritas en la sección anterior, a continuación se muestra un resumen del análisis realizado a los métodos de resolución y técnicas de precondicionamiento implementadas en cada una de estas librerías, tratando de encontrar aquellos parámetros con un mayor impacto en la eficiencia paralela y en el tiempo de resolución de los sistemas lineales.

En este estudio se han utilizado dos matrices diferentes, una de ellas surge de la discretización de la ecuación no lineal de Poisson, Poisson_A, y la otra de la discretización de la ecuación no lineal de continuidad de electrones, Electron_A. Las características principales de estas matrices se muestran en la tabla 4.1.

Inicialmente se ha estudiado el comportamiento de los métodos directos, usando para ello las librerías SuperLU y MUMPS. Con la versión de

		2 proc	3 proc	4 proc	5 proc	6 proc	7 proc
SuperLU	tiempo(s)	6.964	5.171	4.802	4.122	4.120	4.309
	eficiencia	0.842	0.756	0.610	0.569	0.474	0.388
MUMPS	tiempo(s)	1.568	1.531	1.364	1.226	1.107	1.089
	eficiencia	0.634	0.433	0.364	0.324	0.299	0.261
PSPARS.	tiempo(s)	0.282	0.241	0.169	0.163	0.159	0.155
	eficiencia	1.108	0.864	0.924	0.768	0.654	0.578
PETSc	tiempo(s)	0.551	0.254	0.181	0.129	0.108	0.127
	eficiencia	1.127	1.628	1.715	1.927	1.908	1.397
Aztec	tiempo(s)	0.482	0.307	0.251	0.230	0.183	0.199
	eficiencia	1.063	1.097	1.020	0.887	0.944	0.729

Tabla 4.2: Tiempo total y eficiencia para las condiciones óptimas de cada librería, usando la matriz Poisson_A.

memoria distribuida de la librería SuperLU se ha resuelto una factorización LU completa en paralelo. Como era de esperar el tiempo de resolución disminuye con el número de procesadores, aunque esta disminución se va haciendo cada vez menos pronunciada. Se ha repetido el mismo proceso con la librería MUMPS, en la que también se ha encontrado un descenso en el tiempo de ejecución con el número de procesadores, pero sin obtener una alta escalabilidad.

Utilizando la definición estándar de la eficiencia paralela, dada por la expresión:

$$E(p) = \frac{t_1}{t_p\, p} \tag{4.1}$$

con t_1 y t_p los tiempos de ejecución de la carga de trabajo en un único procesador o en p procesadores respectivamente. En la tabla 4.2 se muestra una comparativa entre los tiempos de resolución y la eficiencia obtenida y su dependencia con el número de procesadores empleados para cada una de las librerías analizadas en este trabajo. Los valores representados en la tabla han sido obtenidos para la matriz Poisson_A y para cada librería se han utilizado los métodos de resolución y técnicas de precondicionamiento óptimas, en el sentido de que minimicen el tiempo de ejecución. Inicialmente se comparan tan sólo los valores obtenidos con las librerías SuperLU y MUMPS. Se observa que la librería MUMPs obtiene los menores tiempos de ejecución hasta 7 procesadores. Esta librería necesita menos de la mitad del tiempo que necesita SuperLU para obtener la solución. Pero por otro lado, es la librería SuperLU la que obtiene mayores valores de eficiencia paralela.

En segundo lugar, se han estudiado los métodos iterativos, analizando para ello los resolutores basados en subespacios de Krylov, los precondicio-

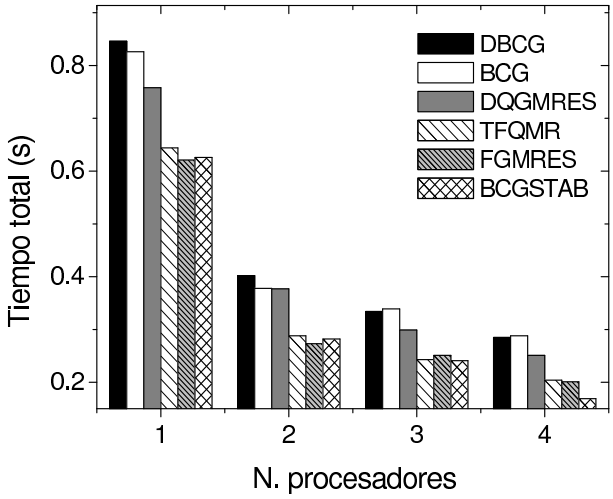


Figura 4.1: Comparativa de resolutores iterativos implementados en la librería PSPARSLIB usando la matriz Poisson_A.

nadores paralelos, los procesos de llenado en las factorizaciones LU incompletas, la dimensión del subespacio de Krylov y el número de procesadores empleados. Se han considerado dos criterios de parada diferentes, o bien que el residuo de salida del precondicionador se reduzca un factor 10^{-7}, o que se alcance un máximo de 500 iteraciones.

Usando la librería PSPARSLIB se realiza una comparación de resolutores iterativos, representada en la figura 4.1, encontrándose que los resolutores TFQMR, FGMRES y BCGSTAB obtienen los menores tiempos de resolución. Además, también se ha realizado una comparativa entre los tres precondicionadores basados en descomposición de dominios implementados en la librería. Así, la figura 4.2 representa el tiempo total necesario para obtener la solución de un sistema local en función del número de procesadores empleados para los precondicionadores Schwarz aditivo, SOR multicolor y para técnicas de precondicionamiento basadas en el cálculo de la matriz de complemento de Schur. De esta figura se deduce que el método Schwarz aditivo es el más apropiado para la resolución de estos sistemas.

En los métodos Schwarz aditivo y SOR multicolor se ha utilizado como resolutor interno el método FGMRES y en el método basado en el complemento de Schur se ha utilizado una técnica LSCHUR, que resuelve cada matriz de complemento de Schur local por medio de un resolutor FGMRES precondicionado con una factorización ILU. En el cómputo de los nodos internos a cada subdominio se ha utilizado una factorización LU incompleta

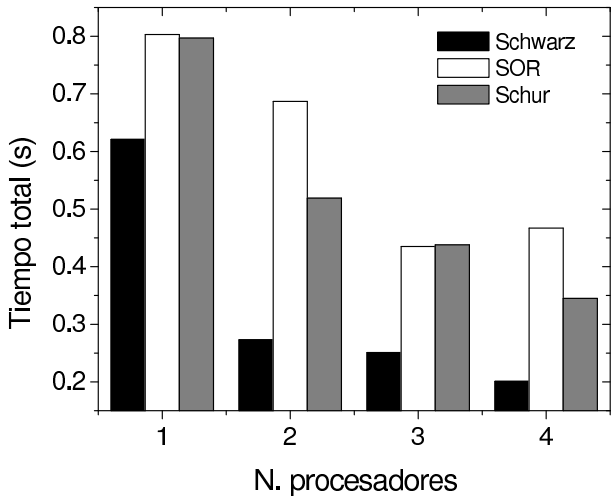


Figura 4.2: Comparativa entre los precondicionadores basados en descomposición de dominios implementados en la librería PSPARSLIB. Para ello se ha usado la matriz Poisson_A.

dependiente de dos parámetros, el llenado, teniendo en cuenta que $2 \cdot llenado$ es el número máximo de elementos de llenado por fila que pueden ser introducidos en la estructura de datos de salida, y un cierto valor umbral, fijado a 10^{-4}.

Los menores tiempos de resolución, para la matriz Poisson_A, independientemente del número de procesadores utilizado, se obtuvieron para valores bajos de llenado, entre 5 y 15. Este comportamiento puede observarse en la figura 4.3, en la que se representa el tiempo de resolución de un sistema lineal en función del número de procesadores y del llenado para el resolutor BCGSTAB. En cambio, para la matriz Electron_A, el valor óptimo de llenado es más elevado, sobre 25, tal y como se muestra en la figura 4.4, obtenida para el resolutor FGMRES. Esto da una idea de la diferente naturaleza física de las dos matrices analizadas, a pesar de tener ambas la misma estructura.

Es necesario tener en cuenta que el tiempo de resolución de un sistema lineal está compuesto por la suma de dos contribuciones, el tiempo empleado en la factorización LU y el tiempo necesario por el resolutor iterativo para alcanzar la convergencia. Por lo tanto, un aumento en el valor de llenado provoca un importante incremento en el tiempo necesario para realizar la factorización incompleta, tal y como se puede observar en la figura 4.5. Esta figura muestra la dependencia del tiempo de factorización ILU con el llenado y con el número de procesadores para la matriz Poisson_A, utilizando para

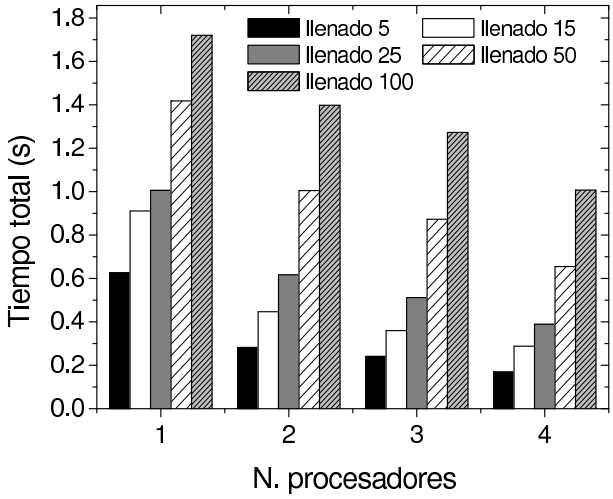


Figura 4.3: Dependencia del tiempo de resolución de un sistema lineal con el llenado, usando el resolutor BCGSTAB, para la matriz Poisson_A.

ello el resolutor BCGSTAB. Este aumento en el tiempo de factorización implica generalmente un incremento en el tiempo total. Además, la figura 4.6 muestra el tiempo empleado por el resolutor iterativo para alcanzar la convergencia y su dependencia con el número de procesadores y el llenado.

La influencia de la dimensión del subespacio de Krylov utilizada es muy pequeña, puesto que sólo se encuentran variaciones en el tiempo de ejecución del orden del 5 % al cambiar dos órdenes de magnitud el tamaño del subespacio de Krylov. Sin embargo, es necesario tener en cuenta que no se puede fijar este parámetro a un valor excesivamente bajo, puesto que no se lograría la convergencia del sistema lineal. Por lo tanto, para las medidas tomadas tanto con esta librería como con las otras librerías que se muestran a continuación se utilizó un valor 50 de la dimensión del subespacio de Krylov.

Otro factor a tener en cuenta es el número de iteraciones realizadas y su dependencia con el llenado y con el número de procesadores. El número de iteraciones disminuye al aumentar el llenado, y por el contrario, aumenta con el número de procesadores utilizados en la resolución del sistema.

Respecto al número de procesadores utilizados, se encuentra un descenso en el tiempo total al aumentar el número de procesadores. Este descenso se hace menos pronunciado al ir aumentado el número de procesadores, debido al tamaño de las matrices locales, que son demasiado pequeñas al aumentar mucho el número de procesadores. La reducción en el tamaño de las matrices

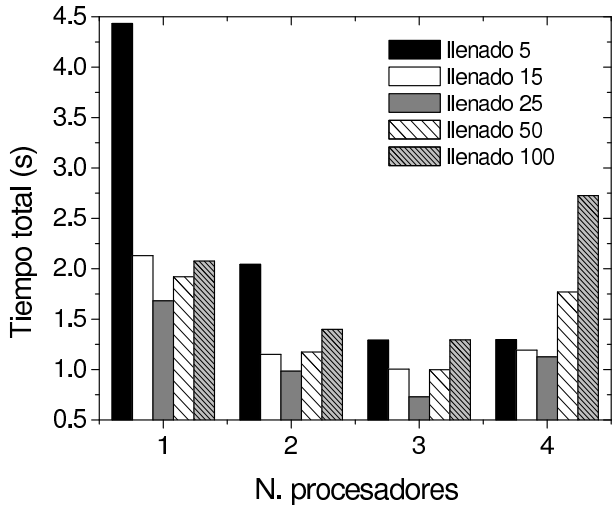


Figura 4.4: Dependencia del tiempo de resolución de un sistema lineal con el llenado, usando el resolutor FGMRES, para la matriz Electron_A.

Matriz	Proc	$\text{Efic}_{llen.5}$	$\text{Efic}_{llen.15}$	$\text{Efic}_{llen.25}$	$\text{Efic}_{llen.50}$	$\text{Efic}_{llen.100}$
Poisson_A	2	1.108	1.019	0.815	0.706	0.615
	3	0.864	0.843	0.655	0.541	0.450
	4	0.924	0.792	0.646	0.542	0.427
Electron_A	2	1.085	0.925	0.854	0.818	0.742
	3	1.145	0.707	0.769	0.641	0.534
	4	0.855	0.446	0.373	0.271	0.190

Tabla 4.3: Dependencia de la eficiencia paralela con el llenado para los resolutores BCGSTAB, en el caso de la matriz Poisson_A, y FGMRES, en el caso de la matriz Electron_A.

implica un descenso en el número de nodos internos y un incremento de los nodos de interfaz, lo que se traduce en un mayor coste tanto en tiempos de computación como de comunicación. Con respecto a la eficiencia del código paralelo, se encuentran los valores más elevados al usar dos procesadores. Este comportamiento puede observarse en la tabla 4.3 en la que se muestra la dependencia de la eficiencia paralela con el valor del llenado para las matrices Poisson_A y Electron_A. Además se encuentra que los valores más elevados de la eficiencia paralela se logran para valores bajos de llenado, entre 5 y 25. Estas medidas fueron obtenidas con los resolutores BCGSTAB y FGMRES para las matrices Poisson_A y Electron_A respectivamente. Finalmente, es necesario comentar que en el código utilizado para la resolución

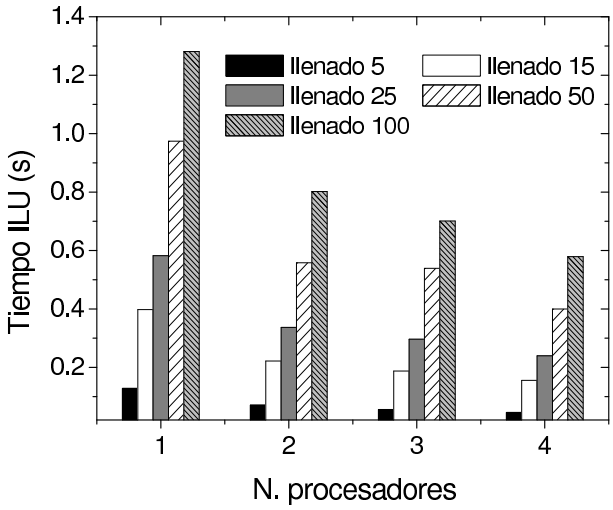


Figura 4.5: Dependencia del tiempo de factorización con el llenado, usando el resolutor BCGSTAB, para la matriz Poisson_A.

de los sistemas lineales se ha aplicado un escalado diagonal de la matriz y se ha utilizado solapamiento entre subdominios.

En la librería PETSc existe una amplia variedad de técnicas de precondicionamiento. Aquí, se limita el estudio a los precondicionadores basados en descomposición de dominios, en concreto, se utiliza el método Schwarz aditivo porque es la técnica de precondicionamiento más eficiente para este caso en particular. En esta librería, en la resolución de los nodos internos dentro de cada subdominio se han utilizado factorizaciones LU incompletas dependientes de un cierto nivel de llenado. El nivel de llenado indica el número de columnas alrededor de la diagonal en las que están permitidas entradas de llenado. También se ha fijado a 10^{-4} el valor de tolerancia umbral, de tal forma que entradas con valores inferiores a este sean rechazadas. En el código utilizado con esta librería se ha aplicado un escalado de la matriz.

Utilizando PETSc los menores tiempos de ejecución fueron encontrados para los resolutores BCGSTAB y GMRES, aunque para bajos niveles de llenado, inferiores a 10, el resolutor BCGSTAB obtiene menores tiempos de resolución que los dados por GMRES. Para la matriz Electron_A los menores tiempos de ejecución se encuentran para un nivel 5 de llenado mientras que la matriz Poisson_A obtiene sus tiempos de resolución mínimos para niveles de llenado muy bajos, 1 ó 2. Pero hay que tener en cuenta que en el caso secuencial la tendencia se invierte, y los menores tiempos de ejecución se encuentran para niveles de llenado elevados, en torno a 10.

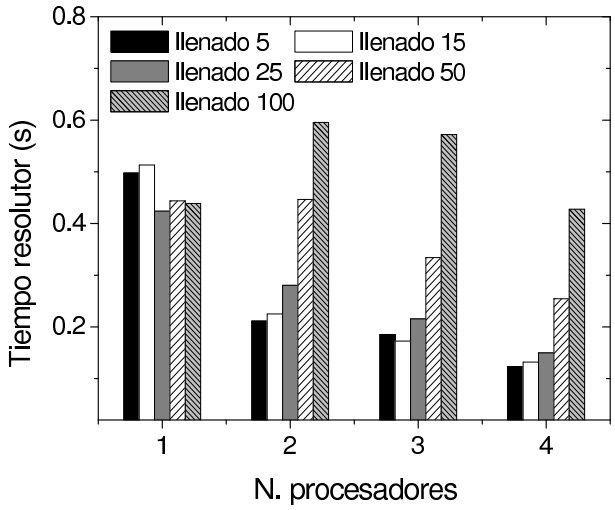


Figura 4.6: Dependencia del tiempo del método iterativo FGMRES con el llenado, usando el resolutor BCGSTAB, para la matriz Poisson_A.

Como ejemplo, la figura 4.7 muestra el tiempo de resolución en función del número de procesadores empleados y su dependencia con el nivel de llenado. Estas medidas fueron obtenidas con el resolutor BCGSTAB para la matriz Poisson_A.

En la librería Aztec, al igual que en los casos anteriores se ha analizado una serie de resolutores iterativos disponibles, utilizando como resolutor externo el método basado en descomposición de dominios Schwarz aditivo y como precondicionador interno una factorización incompleta LU dependiente del parámetro ilut–fill. Este parámetro indica que la factorización final puede contener como máximo ilut–fill veces el número de elementos no nulos existentes en la matriz original. Se ha escogido este criterio para introducir el llenado, en lugar del utilizado en la librería PETSc que también está disponible en Aztec, porque permite obtener menores tiempos de ejecución en todos los casos estudiados. Además la factorización LU incompleta es dependiente de un cierto valor de tolerancia umbral, fijado a 10^{-4}. El código en el que se han realizado todos los cálculos se ha aplicado un escalado diagonal de la matriz y se ha trabajado con solapamiento, siendo uno el número de líneas de la malla que pueden ser común a dos de los subdominios.

La figura 4.8 representa, para la matriz Poisson_A, el tiempo de resolución de un sistema lineal en función del número de procesadores empleado y del parámetro ilut–fill, utilizando para ello el resolutor iterativo GMRES. Para esta matriz, los menores tiempos de ejecución, se obtuvieron para va-

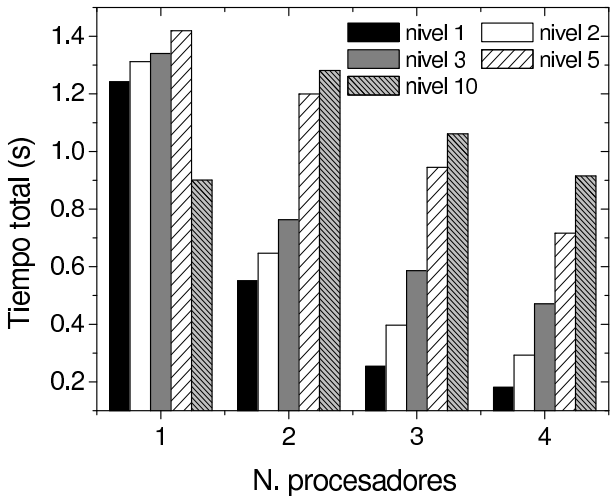


Figura 4.7: Dependencia del tiempo de resolución de un sistema lineal con el nivel de llenado en la librería PETSc. Los resultados han sido obtenidos para el resolutor BCGSTAB precondicionado con el método Schwarz aditivo. Se ha usado la matriz Poisson_A.

lores bajos de ilut–fill, entre 1.2 y 1.5, para todos los resolutores estudiados. Así, por ejemplo, en la figura se encuentra un mínimo, sea cual sea el número de procesadores utilizado, en ilut–fill=1.2.

Comparando los resolutores iterativos entre sí, se encuentra que los métodos BCGSTAB y GMRES son los que obtienen menores tiempos de ejecución. Esto también es cierto al utilizar la matriz Electron_A, pero se encuentra que para valores pequeños de ilut–fill, entre 1.0 y 1.3, el resolutor BCGSTAB obtiene sus menores tiempos de ejecución, mientras que el resolutor GMRES los obtiene para valores más elevados de ilut–fill, entre 1.5 y 2.0.

4.2.3. Conclusiones del análisis de los métodos de resolución de sistemas dispersos de ecuaciones lineales

A continuación se resumen los resultados obtenidos en este estudio. Con respecto a las técnicas de precondicionamiento basadas en descomposición de dominios, el método Schwarz aditivo es el más eficiente de los tres métodos analizados, independientemente del valor de llenado, de la dimensión del subespacio de Krylov o del número de procesadores utilizado. Por otro lado, los resolutores BCGSTAB, FGMRES y GMRES son los más veloces en todas las librerías analizadas, con diferencias en tiempo entre ellos pequeñas

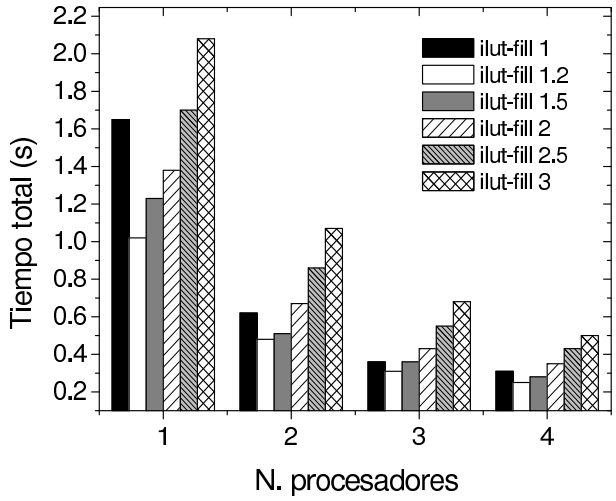


Figura 4.8: Dependencia del tiempo de resolución de un sistema lineal con el parámetro ilut–fill para la librería Aztec. Los resultados han sido obtenidos para el resolutor GMRES precondicionado con el método Schwarz aditivo. Se ha usado la matriz Poisson_A.

generalmente. Sin embargo, en las librerías PETSc y Aztec, BCGSTAB es más rápida a valores bajos de llenado.

Con respecto a las factorizaciones incompletas LU, a pesar de las diferentes implementaciones del proceso de llenado desarrolladas en las librerías, en todos los casos analizados, se encuentran los menores tiempos de resolución para valores bajos de llenado. Al aumentar el llenado, el tiempo de resolución del sistema lineal aumenta a causa del importante incremento en el tiempo de factorización, que no logra ser compensado con un menor número de iteraciones del resolutor interno.

Finalmente, se ha estudiado la influencia del número de procesadores utilizado. Lógicamente, se ha encontrado una reducción en el tiempo total de ejecución al aumentar el número de procesadores utilizados. Un ejemplo de este comportamiento se puede observar en la tabla 4.2 en la que, para la matriz Poisson_A, se muestran los tiempos totales de resolución de un sistema lineal y la eficiencia paralela para cada una de las librerías estudiadas, considerando tanto el resolutor como las condiciones de llenado en las que cada librería obtiene los menores tiempos de ejecución.

La eficiencia del código paralelo utilizado en las librerías basadas en métodos directos disminuye notablemente al aumentar el número de procesadores. Estas librerías obtienen sus valores más elevados de la eficiencia

paralela al utilizar 2 procesadores. Además, los tiempos de resolución del sistema lineal dados por estas librerías son mucho más elevados que los obtenidos por cualquiera de las librerías estudiadas basadas en métodos iterativos. Esto nos permite alcanzar la conclusión de que las técnicas basadas en métodos directos no son las más adecuadas para implementar en el simulador 3D de dispositivos.

Con respecto a las librerías basadas en métodos iterativos los resultados muestran que los tiempos de ejecución están próximos en todos los casos estudiados, destacando que la librería PSPARSLIB obtiene los menores tiempos de resolución hasta 4 procesadores, mientras que a partir de 5 procesadores es PETSc la que obtiene los tiempos más reducidos. Estudiando la eficiencia paralela, las librerías Aztec y PSPARSLIB obtienen sus máximos de eficiencia entre 2 y 4 procesadores, mientras que la librería PETSc los obtiene entre 4 y 6 procesadores.

Por último es necesario recordar que este estudio ha sido realizado para matrices de dimensión relativamente pequeña, aproximadamente 30.000 nodos. La dimensión de la matriz influirá en los resultados de eficiencia paralela. En este análisis de los métodos de resolución y las técnicas de precondicionamiento más adecuados para implementar en el simulador tridimensional no se han utilizado matrices de dimensión más elevada ni un mayor número de procesadores puesto que el principal objetivo es el uso de simulador para el cálculo del impacto de las fluctuaciones de parámetros intrínsecos en las curvas características de los dispositivos. Para ello se busca utilizar una malla pequeña que proporcione valores correctos de las variables y que permita obtener los resultados de la simulación en un tiempo asumible. Además, para la optimización del uso de los recursos disponibles es muy importante emplear un número óptimo de procesadores en cada simulación. La librería PSPARSLIB es la más adecuada, conforme a los resultados obtenidos en este estudio y en otros previos [52, 90, 91], para ser implementada en el simulador, puesto que obtiene los menores tiempos de ejecución y los valores óptimos de eficiencia para un número de procesadores pequeño, adecuado para el tamaño de malla que se empleará en los estudios posteriores.

4.3. Optimización de la etapa de resolución de los sistemas lineales de ecuaciones

Para el cálculo del impacto de las fluctuaciones de parámetros intrínsecos se ha utilizado, como se ha comentado anteriormente, un simulador 3D

paralelo basado en el modelo de arrastre–difusión. En la resolución de los sistemas lineales de ecuaciones locales a cada procesador se emplea la librería PSPARSLIB, puesto que en general obtuvo los menores tiempos de simulación de todas las librerías analizadas y altos valores de eficiencia paralela.

De todas las técnicas de precondicionamiento basadas en descomposición de dominios implementadas en esta librería, el método Schwarz aditivo obtiene tiempos de simulación considerablemente inferiores al resto. El algoritmo básico de este método fue descrito anteriormente en la sección 3.3.4. Esta técnica se utiliza para actualizar el valor de los nodos frontera entre subdominios.

En cambio, en la resolución del sistema lineal local para cada uno de los subdominios se utiliza una factorización LU incompleta dependiente de un cierto llenado y un umbral numérico (ILUT) como técnica de precondicionamiento. Este técnica se combina con un resolutor iterativo basado en subespacios de Krylov, el método FGMRES (Flexible Generalised Minimal Residual Method). Este método es una variante del método GMRES que permite que el precondicionamiento varíe cada cierto número de iteraciones.

4.3.1. Propuesta de optimización

El principal objetivo de este estudio es analizar como mejorar la eficiencia paralela del simulador 3D paralelo de tal forma que se reduzca el tiempo de simulación. Por ello, se ha analizado en detalle la etapa de resolución de sistemas lineales implementada en el simulador, puesto que es esta la parte que más tiempo consume de todo el proceso de simulación, del orden del 90 % del tiempo total. Así, esta propuesta de optimización se centra en esta etapa tratando de minimizar su tiempo de ejecución.

Para un número pequeño de procesadores, las factorizaciones LU incompletas son una de las más importantes contribuciones al tiempo total de simulación, limitando el rendimiento de la eficiencia paralela. Como ejemplo, los resultados marcados con rayas inclinadas en la figura 4.9 (*Original*) muestran, para una malla de 126.166 nodos dividida en cuatro subdominios, el tiempo de factorización LU utilizado por cada uno de los procesadores. El tiempo de resolución de una factorización LU incompleta es en promedio del orden de los 150 segundos, siendo el tiempo total de solución del sistema lineal completo 250 segundos. Además el comportamiento entre procesadores está muy desbalanceado, encontrándose una diferencia en tiempo del orden del 48 % entre el procesador más rápido y el más lento.

Este comportamiento tan desbalanceado es debido principalmente a un

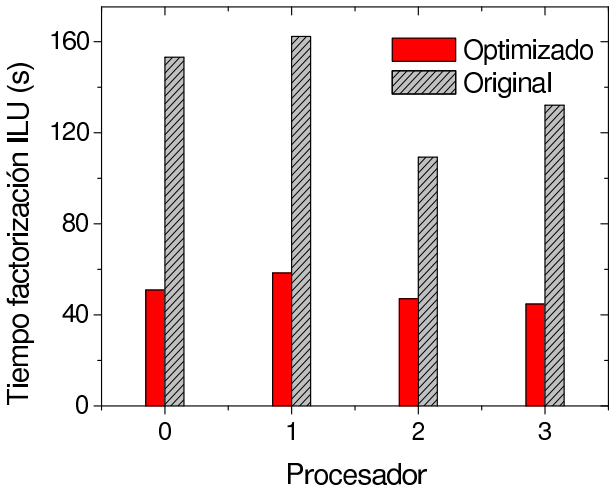


Figura 4.9: Tiempo utilizado por cada procesador en realizar una factorización LU incompleta utilizando las dos versiones del código, original y optimizada. Estos resultados han sido obtenidos utilizando una malla de 126.166 nodos particionada en cuatro subdominios.

reordenamiento interno realizado por la librería PSPARSLIB. Esta librería necesita llamar a una función, SETUP, previamente al cálculo de la factorización LU. Esta función reordena los nodos de las matrices locales de tal forma que etiqueta en primer lugar los nodos internos, a continuación los nodos de frontera locales y por último los nodos frontera externos. Este reordenamiento cambia el patrón de la matriz local, que había sido optimizado en formato de "punta de flecha" a través del reordenamiento hecho por la librería METIS [62] en la etapa de preprocesado del proceso de simulación. El nuevo reordenamiento dado por la función SETUP aumenta el llenado de la matriz. Las figuras 4.10 y 4.11 representan el patrón de la matriz antes y después de la llamada a la función SETUP, respectivamente.

La librería PSPARSLIB utiliza la función SETUP para solapar comunicaciones y computaciones en los posteriores productos matriz–vector, puesto que este reordenamiento no sólo coloca los nodos frontera externos al final de la estructura de datos, sino que además los ordena por procesadores. La figura 4.12 muestra un diagrama de flujo de la etapa de resolución de sistemas lineales implementada por la librería PSPARSLIB. Como se ha mencionado anteriormente, en la solución de los nodos locales internos a cada subdominio se emplea el algoritmo iterativo FGMRES. Este método está precondicionado por el método iterativo PGMRES, que es una versión sencilla del algoritmo GMRES precondicionado con una ILUT.

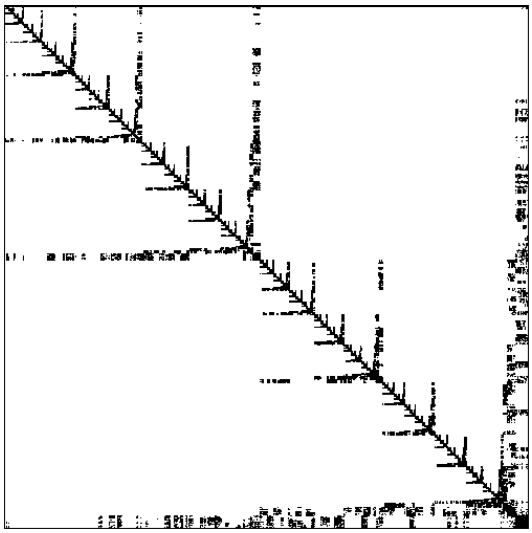


Figura 4.10: Patrón de una matriz local reordenada con el programa METIS, obtenida previamente a la llamada a la función SETUP.

En nuestra propuesta de optimización se intenta reducir el tiempo de la factorización LU incompleta sin comprometer por ello el solapamiento de computaciones y comunicaciones. Por lo tanto, para realizar la factorización LU, se utiliza la matriz inicial, previa a la originada con la función SETUP, mientras que para la resolución del método iterativo FGMRES se utiliza la nueva ordenación de la matriz generada con la función SETUP. Por lo tanto, después de cada factorización LU es necesario permutar la matriz resultante de este proceso para adaptarla al nuevo etiquetado originado con el reordenamiento dado por SETUP. La figura 4.13 muestra un diagrama de flujo de la etapa de resolución de los sistemas lineales optimizada con nuestra propuesta.

Esta técnica de optimización disminuye de forma muy considerable el tiempo de factorización ILU por procesador y mejora el balanceo computacional entre procesadores. Por ejemplo, la figura 4.9 también muestra el tiempo de factorización incompleta en la versión optimizada para cada uno de los procesadores utilizados. El tiempo de la factorización incompleta es en promedio 50 segundos y entre el procesador más rápido y el más lento hay una diferencia en tiempo de aproximadamente el 30 %.

Por lo tanto, el principal objetivo de esta optimización es el uso, durante la resolución de una factorización LU, de una matriz con un ancho de banda por fila inferior al de la matriz original. Esto es importante puesto que

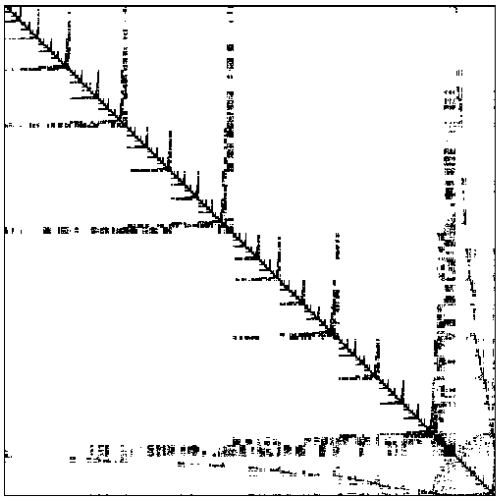


Figura 4.11: Patrón de una matriz local reordenada después de la llamada a la función SETUP.

el ancho de banda por fila fija las posiciones extremas de una fila en las que los elementos de llenado pueden ser introducidos. Por lo tanto, esta aproximación permite trabajar con valores más bajos de llenado, ahorrando tiempo computacional y consumo de memoria.

En la versión optimizada, como se ha visto en el diagrama de flujo, la función ILU recibe como entrada una matriz reordenada dada por el programa METIS (A), diferente a la dada por la función SETUP (A'), con el objetivo de reducir el llenado. El único coste de esta optimización, por iteración, es el tiempo computacional necesario para permutar el vector de la solución obtenida con la función Lusol0, para adaptarlo al ordenamiento original exigido por PSPARSLIB. En la función Lusol0 se resuelven consecutivamente sistemas lineales compuestos por unas matrices triangulares superiores e inferiores para una matriz LU, por lo tanto esta función recibe como entrada la salida de la función ILU (LU).

4.3.2. Resultados numéricos

El mallado para el simulador 3D se genera con dos programas diferentes, uno desarrollado en la Universidad de Cornell, el QMG [59], y otro en el departamento de Electrónica y Computación de la Universidad de Santiago de Compostela, el MMG [60]. Para llevar a cabo este estudio se han empleado cuatro mallas de diferentes tamaños. Sus principales características, tales como el número de nodos, tetraedros, elementos no nulos (NNZ) y mallador

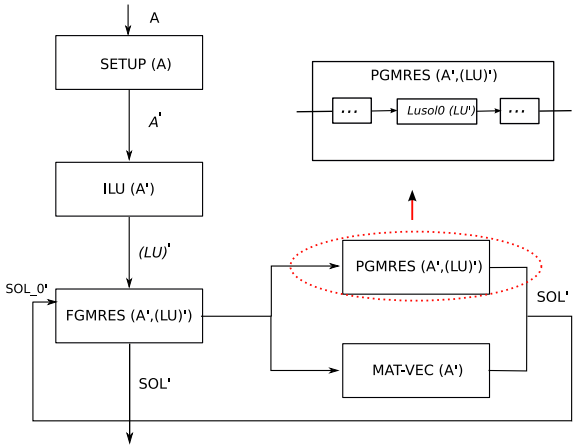


Figura 4.12: Diagrama de flujo de la etapa de resolución de sistemas de ecuaciones implementada por la librería PSPARSLIB, previo a cualquier optimización.

Malla	Nodos	Tetraedros	NNZ	Mallador
S	29,012	147,682	398,102	QMG
M	76,446	433,824	1,116,664	MMG
L	126,166	723,040	1,852,656	MMG
H	221,760	1,253,760	3,223,110	MMG

Tabla 4.4: Información general sobre las cuatro mallas utilizadas en la simulación.

utilizado se muestran en la tabla 4.4.

En el apartado anterior se comentó que las ecuaciones de Poisson y de continuidad de electrones dan lugar a matrices de características muy diferentes, por ello, en este estudio se presentan por separado resultados de tiempo de resolución de la ecuación de Poisson y resultados de tiempo totales, en los que también afecte la resolución de la ecuación de continuidad de electrones. Más información sobre la propuesta de optimización y sobre los resultados obtenidos se puede encontrar en [109, 110].

Consideraciones previas

Los resultados que se presentan de ahora en adelante son comparativas entre las dos versiones del código, inicial y optimizada. Ambas versiones son distinguidas, en las tablas representadas a continuación, a través del símbolo

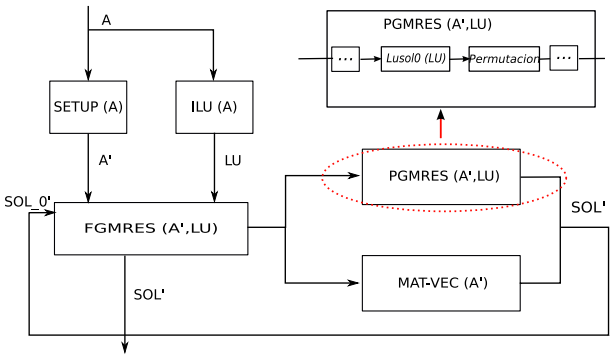


Figura 4.13: Diagrama de flujo de la etapa de resolución de sistemas de ecuaciones optimizada.

$-O$ en los resultados de la versión optimizada.

Se ha utilizado un valor constante del llenado de 700. Este valor tan elevado parece ir en contraposición con las conclusiones obtenidas en la sección anterior, en las que se comentaba que los menores tiempos de ejecución se obtenían para valores bajos de llenado. Hay que tener en cuenta que ese estudio tenía como objetivo encontrar los valores óptimos de llenado y de otros parámetros tales que el tiempo de ejecución se minimizase. Sin embargo, en nuestro caso particular, las conclusiones obtenidas con respecto al valor de llenado óptimo no pueden ser aplicadas, a causa de tres razones principalmente. Por un lado, en este análisis se utilizan mallas de tamaños muy diferentes y se utiliza un valor fijo de llenado para todas ellas, puesto que no se desea que una variación en el valor del llenado interfiera en el estudio de la eficiencia paralela. Por otro lado, para hacer este estudio y también por propósitos comparativos, se fija el mismo valor de llenado para la resolución de la ecuación de Poisson y para la resolución de la ecuación de continuidad de electrones. Esta última ecuación necesita valores de llenado bastante más elevados que la ecuación de Poisson para poder ser resuelta de una forma correcta. Además, en el simulador tridimensional se puede trabajar con polarizaciones elevadas, que requieren altos valores del llenado para que los sistemas converjan.

Influencia de la ecuación de Poisson

El estudio que se presenta a continuación está centrado en la resolución de la ecuación de Poisson en el equilibrio. Las tablas 4.5, 4.6 y 4.7 muestran para las mallas S, M y L respectivamente, la influencia del número

proc	t_{ILU}	t_{ILU-O}	$t_{met.iter}$	$t_{met.iter-O}$	t_{resol}	$t_{resol-O}$	it_{resol}
1	20.76	20.61	0.45	0.45	21.21	21.06	2
2	9.82	6.71	5.28	7.55	15.10	14.27	33
4	4.62	1.64	7.21	3.25	11.84	4.90	39
8	1.95	0.51	4.38	1.01	6.34	1.52	41
16	0.67	0.15	1.38	0.37	2.05	0.52	55
32	0.21	0.05	0.49	0.17	0.70	0.22	67
62	0.06	0.03	0.13	0.08	0.19	0.11	61

Tabla 4.5: Dependencia del número de procesadores en los tiempos promedio por iteración del resolutor externo necesarios para: una factorización incompleta LU (t_{ILU}), el resolutor FGMRES en alcanzar la convergencia ($t_{met.iter}$) y la solución de un sistema lineal local (t_{resol}=t_{ILU}+$t_{met.iter}$). Estos tiempos se muestran para las dos versiones del código, inicial y optimizada (indicada en la tabla por el símbolo $-O$). También se muestra el número promedio de iteraciones del resolutor interno (it_{resol}), idéntico para las dos implementaciones del código. Los resultados presentados en esta tabla corresponden a la resolución de la ecuación de Poisson en el equilibrio, utilizando para ello la malla S.

de procesadores empleados en: el tiempo promedio por iteración del resolutor externo en realizar una factorización LU incompleta (t_{ILU}), el tiempo promedio necesario por el resolutor FGMRES para alcanzar la convergencia ($t_{met.iter}$), el tiempo promedio en resolver un sistema lineal local (t_{resol}), siendo $t_{resol} = t_{ILU} + t_{met.iter}$, y el número de iteraciones promedio del resolutor interno (it_{resol}). Este último valor es común para las dos versiones del código.

El número mínimo de procesadores que pueden ser empleados en cada uno de los casos analizados depende del tamaño de la malla y de los requerimientos de memoria. Por ello, para la malla M es posible obtener resultados sólo para más de un procesador y para la malla L es necesario utilizar al menos cuatro procesadores. Como era de esperar, en el caso secuencial, las dos versiones del código, inicial y optimizada, producen los mismos tiempos ya que la optimización no cambia nada en este caso.

En el caso secuencial, t_{ILU} es la principal contribución a t_{resol}, sin embargo, con el aumento del número de procesadores, y a causa de la reducción en el tamaño de las matrices locales, hay un descenso drástico en el tiempo de factorización t_{ILU}. La resolución de una factorización LU incompleta por subdominios es una tarea muy paralelizable en las dos versiones del códi-

proc	t_{ILU}	t_{ILU-O}	$t_{met.iter}$	$t_{met.iter-O}$	t_{resol}	$t_{resol-O}$	it_{resol}
2	113.20	75.90	39.11	30.47	152.32	106.37	32
4	56.04	23.40	46.89	32.09	102.93	55.49	48
8	30.54	5.27	58.92	17.13	89.46	22.40	60
16	9.35	1.23	18.87	5.11	28.22	6.34	69
32	2.63	0.35	8.86	1.25	11.49	1.60	82
62	0.65	0.11	1.84	0.47	2.49	0.58	93

Tabla 4.6: Representación de los mismos parámetros que en la tabla 4.5 pero utilizando la malla M.

proc	t_{ILU}	t_{ILU-O}	$t_{met.iter}$	$t_{met.iter-O}$	t_{resol}	$t_{resol-O}$	it_{resol}
4	139.19	50.29	115.28	62.83	254.47	113.12	83
8	62.88	14.47	83.78	44.08	146.66	58.55	104
16	24.69	3.45	51.28	21.32	75.97	24.77	110
32	7.70	0.85	26.10	6.92	33.80	7.77	129
62	1.93	0.26	7.31	1.16	9.24	1.42	130

Tabla 4.7: Representación de los mismos parámetros que en la tablas 4.5 y 4.6 pero utilizando la malla L.

go, aunque los tiempos de factorización son siempre menores en la versión optimizada del código. Por otro lado, $t_{met.iter}$ en el caso secuencial es casi despreciable en comparación la contribución del tiempo de factorización ILU. Esto es debido al pequeño número de iteraciones que el método iterativo tiene que realizar hasta alcanzar la convergencia del sistema. Sin embargo, la influencia de este tiempo en t_{resol} se incrementa al aumentar el número de procesadores a causa de un incremento en el número de iteraciones del resolutor interno.

El incremento del número de iteraciones con el aumento del número de procesadores utilizados provoca que $t_{met.iter}$ para dos procesadores sea más elevado que el obtenido en el caso secuencial, aunque este tiempo escala correctamente con el aumento del número de procesadores. Para la malla S, considerando la versión optimizada del código, $t_{met.iter}$ llega a ser inferior al obtenido en el caso secuencial para más de ocho procesadores. Esto no ocurre en la versión inicial del código, en la que el tiempo obtenido con un procesador es siempre el menor hasta 32 procesadores. Las figuras 4.14 y 4.15 son útiles para visualizar todos los resultados explicados anteriormente para esta malla. Así, estas figuras representan la influencia de t_{ILU} y $t_{met.iter}$

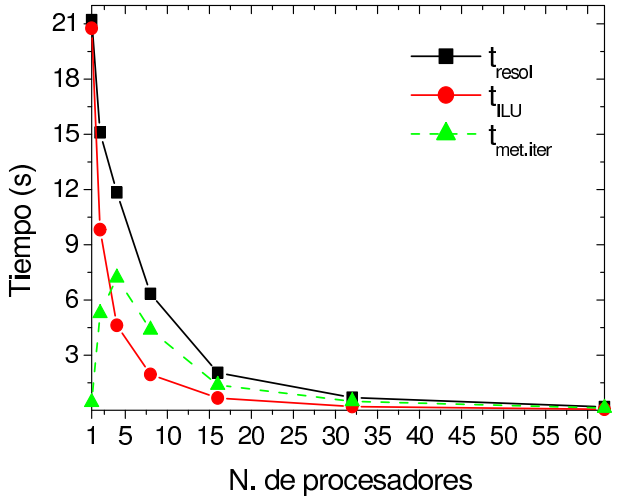


Figura 4.14: Representación de t_{ILU}, $t_{met.iter}$ y t_{resol} frente al número de procesadores para la versión inicial del código. Estos resultados han sido obtenidos para la malla S.

en el tiempo total de resolución del sistema (t_{resol}), para las versiones inicial y optimizada del código respectivamente, y su dependencia con el número de procesadores empleados.

Las figuras 4.16 y 4.17 representan, para las mallas M y L respectivamente, una comparativa gráfica entre t_{ILU}, $t_{met.iter}$ y t_{resol} para las dos versiones del código. Estas figuras son extremadamente útiles para apreciar las mejoras, en el tiempo de simulación y por lo tanto en el rendimiento, dadas por la versión optimizada del código. Esta mejoría no es solamente importante en la resolución de las factorizaciones incompletas sino que también se encuentra en la resolución del método iterativo, lográndose una más rápida convergencia. A pesar de que las dos versiones necesitan realizar el mismo número de iteraciones del resolutor interno, el coste por iteración en la versión optimizada del código es menor, puesto que se reduce el tiempo de resolución de Lusol0.

La tabla 4.8 muestra la influencia del número de procesadores empleados en la solución de la ecuación de Poisson para las mallas S, M y L. Estos tiempos han sido obtenidos para las dos versiones del código, inicial y optimizada, y no representan tiempos promedio, sino los tiempos totales usados en la resolución de los sistemas no–lineales necesarios para obtener la solución global del problema. Estos resultados resaltan las mejoras en tiempo y escalabilidad de la versión optimizada con respecto a la inicial.

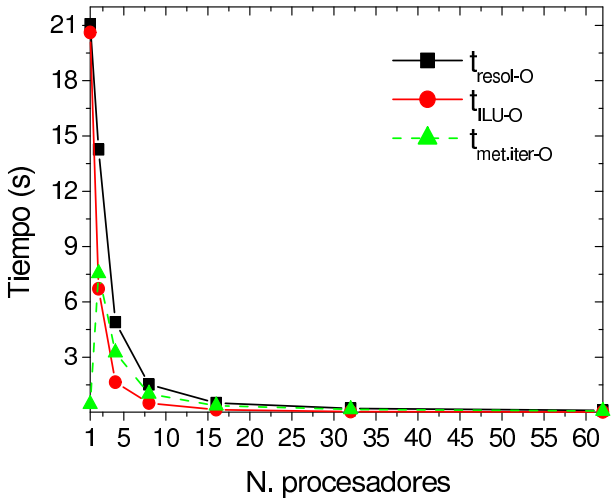


Figura 4.15: Representación de t_{ILU}, $t_{met.iter}$ y t_{resol} frente al número de procesadores para la versión optimizada del código. Estos resultados han sido obtenidos para la malla S.

La figura 4.18 representa la eficiencia paralela para estas tres mallas y su dependencia con el número de procesadores. Las dos versiones del código se muestran en la figura para permitir la comparación. La versión optimizada obtiene eficiencia superlineal hasta 32 procesadores, mientras que en la versión inicial el aumento en la eficiencia es menos importante y solamente usando 32 procesadores la eficiencia alcanza valores superlineales. La eficiencia paralela para la malla S alcanza un valor de saturación para un número elevado de procesadores. Este comportamiento es debido a la reducción del tamaño de las matrices locales al aumentar el número de procesadores y al aumento de las comunicaciones.

Finalmente, la figura 4.19 muestra para las cuatro mallas estudiadas y para las dos versiones del código, utilizando ocho procesadores, el tiempo de simulación y el uso de memoria frente al tamaño del problema. La memoria empleada aumenta linealmente con el número de nodos de la malla, y ambas versiones del código dan aproximadamente resultados idénticos. Sin embargo, el aumento en el tiempo de simulación es mucho más pronunciado en la versión inicial del código que en la optimizada, especialmente para un número de nodos de la malla elevado.

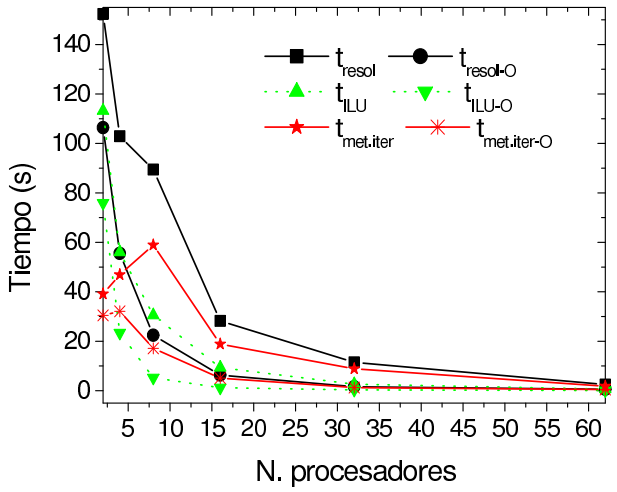


Figura 4.16: Comparativa, para la malla M, entre t_{resol}, t_{ILU} y $t_{met.iter}$ frente al número de procesadores, para las dos versiones del código.

Influencia de la ecuación de continuidad de electrones

Los resultados mostrados hasta ahora sólo analizan el comportamiento del simulador cuando se trata de obtener la solución en el equilibrio. En cambio, en la tabla 4.9 se muestran, para la malla S, los tiempos necesarios para realizar una simulación completa ($t_{total.simul}$) en las dos versiones del código. $t_{total.simul}$ está formado por la suma de dos tiempos diferentes, el tiempo necesario para obtener la solución de equilibrio y el tiempo de obtención de un punto de la curva característica $I_D - V_G$. De esta forma se trata no sólo de analizar la influencia de la ecuación de Poisson, la única que se resuelve en el equilibrio, sino también la influencia de la ecuación de continuidad de electrones, presente junto con la ecuación de Poisson en el resto de la simulación. En la tabla también se incluyen el número de iteraciones promedio del resolutor interno, realizándose este promedio en una simulación completa y las eficiencias paralelas obtenidas para las dos versiones del código.

Las dos versiones de código estudiadas obtienen aproximadamente los mismos tiempos totales de simulación en el caso secuencial. A partir de ahí, si se aumenta el número de procesadores el comportamiento de ambas versiones es radicalmente diferente. En la versión inicial del código se observa una rápida caída de la eficiencia paralela al aumentar el número de procesadores, lo cual no ocurre en la versión optimizada, en la que la eficiencia paralela aumenta con el número de procesadores, llegando a alcanzar valores

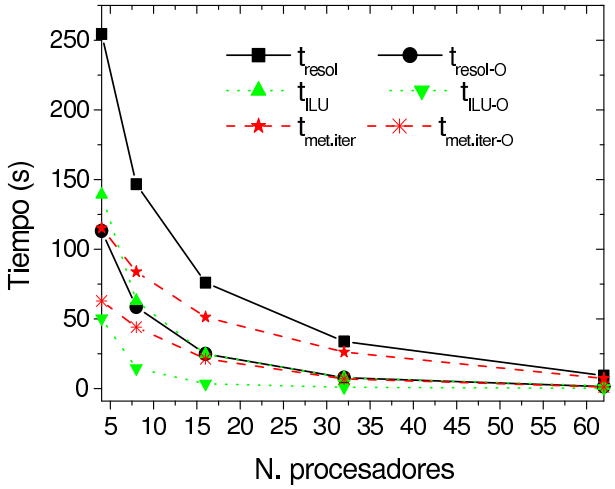


Figura 4.17: Comparativa, para la malla L, entre t_{resol}, t_{ILU} y $t_{met.iter}$ frente al número de procesadores, para las dos versiones del código.

de supereficiencia para ocho procesadores.

El número promedio de iteraciones del resolutor interno es igual para ambas versiones del código exceptuando para 4 procesadores, indicado en la tabla por (*). En este caso el resolutor interno necesita en promedio, para la versión original del código, 30 iteraciones más para converger en cada iteración del resolutor externo que las realizadas por la versión optimizada del código.

En la tabla 4.10 se representa, para la malla S y para las dos versiones del código, considerando una simulación completa, el tiempo promedio en realizar una factorización LU incompleta (t_{ILU}), el tiempo promedio necesario por el resolutor FGMRES para alcanzar la convergencia ($t_{met.iter}$) y el tiempo promedio en resolver un sistema lineal local (t_{resol}). Al igual que en las tablas anteriores, el símbolo $-O$ se refiere a los resultados de la versión optimizada.

Los resultados muestran que los tiempos promedio de la factorización ILU son mucho menores en la versión optimizada del código que en la original. En cambio, para 1 y 2 procesadores los tiempos promedio del resolutor interno son ligeramente superiores en la versión optimizada que en la original, aunque para un número de procesadores superior a 2 esta tendencia ya no se cumple y el resolutor FGMRES necesita mucho menos tiempo de resolución en la versión optimizada. Este comportamiento influye en el incremento de la eficiencia paralela con el número de procesadores en la versión optimizada del código.

proc	mesh S		mesh M		mesh L	
	t_{equi}	t_{equi-O}	t_{equi}	t_{equi-O}	t_{equi}	t_{equi-O}
1	570.53	568.68	–	–	–	–
2	460.34	377.45	4236.52	3665.68	–	–
4	339.26	132.81	3728.74	1481.93	6216.41	2930.07
8	164.02	43.49	2088.04	678.61	3654.45	1537.60
16	55.35	16.54	829.02	150.19	2148.47	534.75
32	19.47	7.71	265.75	38.88	868.13	169.43

Tabla 4.8: Influencia del número de procesadores en el tiempo necesario para obtener la solución de la ecuación de Poisson en el equilibrio, para las dos versiones del código, inicial y optimizada (indicada en la tabla por el símbolo $-O$). Estos resultados se representan para las mallas S, M y L.

proc	$t_{total.simul}$	$Efic.$	$t_{total.simul-O}$	$Efic.-O$	it_{resol}
1	2710.20		2722.41		2
2	2074.85	0.65	1769.59	0.77	35
4	1388.60	0.49	759.60	0.89	98-68(*)
8	1163.17	0.29	335.33	1.01	82

Tabla 4.9: Influencia, para la malla S y las dos versiones del código analizadas, del número de procesadores en el tiempo necesario para realizar una simulación completa y obtener así un punto de la curva característica $I_D - V_G$ y en el número de iteraciones promedio del resolutor interno en esas circunstancias. También para ambas versiones, se muestra la eficiencia paralela de la simulación.

Por último, es posible comparar los tiempos promedio por iteración al resolver la ecuación de Poisson en equilibrio, representados en la tabla 4.5, y los dados al resolver una simulación completa. Tanto en la versión original como en la optimizada, los tiempos promedio al resolver un sistema lineal local son inferiores cuando se resuelve tan sólo la ecuación de Poisson en equilibrio. Esto es debido principalmente al menor tiempo que necesita el resolutor interno para converger en este caso. Este aumento de los tiempos promedio en ambas versiones del código está provocado por el aumento del número de iteraciones del resolutor interno, sobre todo a partir de cuatro procesadores. Este comportamiento es debido a las matrices que se generan a partir de la ecuación de continuidad de electrones, peor condicionadas y

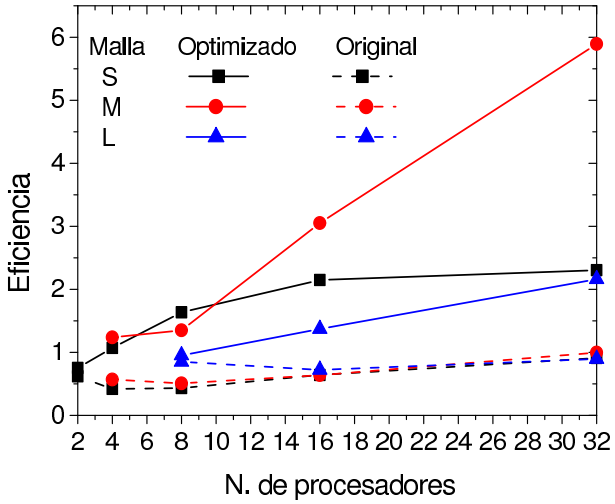


Figura 4.18: Eficiencia paralela obtenida, para las mallas S, M y L, en la resolución de la ecuación de Poisson en equilibrio para las dos versiones del código, original y optimizada.

con una mala dominancia diagonal.

4.4. Nueva estrategia de particionamiento de las mallas utilizadas en el simulador

La librería METIS, usada en el simulador como técnica de particionamiento de las mallas tetraédricas, se basa en la división de la malla en p partes iguales, o subdominios, tratando de minimizar el número de aristas que unen los vértices que pertenecen a subdominios diferentes. Por lo tanto, la estrategia de particionamiento realizada por METIS tiene como objetivo principal la minimización de los nodos frontera por procesador y el reparto equitativo de los nodos locales. En el proceso de simulación se utilizan métodos iterativos para la resolución de los sistemas de ecuaciones que surgen de la discretización del problema. Con este tipo de particionamientos se minimizan las comunicaciones entre procesadores por iteración y se mejora el balanceo.

Nuestra propuesta provoca un desbalanceo en el particionamiento en comparación con el dado por METIS, pero trata de compensarlo con una mejora en la convergencia del método iterativo utilizado y con la disminución del tiempo de factorización.

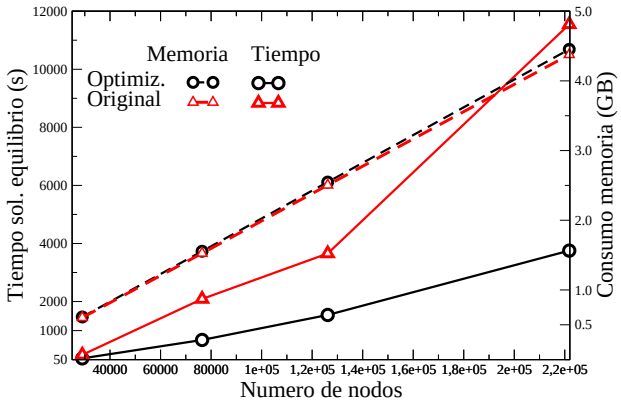


Figura 4.19: Dependencia del tiempo de simulación y del consumo de memoria con el número de nodos de la malla. En la figura se comparan las dos versiones del código.

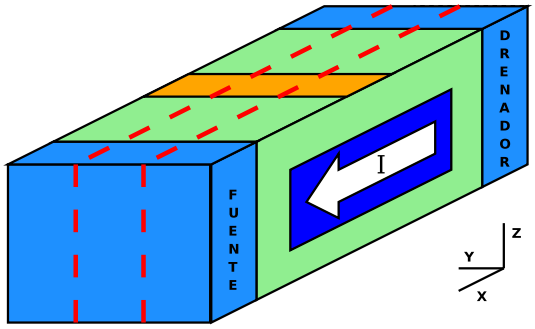


Figura 4.20: Ejemplo de dispositivo semiconductor genérico en el que el flujo de corriente viene indicado por una flecha.

4.4.1. Propuesta de particionamiento

La nueva propuesta de particionamiento de la malla se basa en incluir un nuevo criterio a la hora de realizar el particionamiento. Este consiste en mantener, después del particionamiento, el mismo tipo de estructura física, de tal manera que en cada subdominio se tenga un problema físico con idénticas características que el global. El objetivo es poder aplicar un nivel más alto de paralelismo y resolver los problemas de cada dominio de manera casi desacoplada, reduciendo así las comunicaciones [111].

En la figura 4.20 se muestra el esquema de un dispositivo semiconductor

proc	t_{ILU}	t_{ILU-O}	$t_{met.iter}$	$t_{met.iter-O}$	t_{resol}	$t_{resol-O}$
1	23.55	23.62	0.54	0.57	24.09	24.19
2	10.23	6.98	8.66	9.01	18.89	15.99
4	4.61	1.65	17.59	5.15	22.20	6.80
8	1.96	0.51	7.34	1.96	9.30	2.47

Tabla 4.10: Dependencia, para la malla S, del número de procesadores en los tiempos promedio necesarios para: una factorización incompleta LU (t_{ILU}), el resolutor FGMRES en alcanzar la convergencia ($t_{met.iter}$) y la solución de un sistema lineal local (t_{resol}). Estos tiempos se muestran para las dos versiones del código y corresponden a la obtención de un punto de la curva característica, realizándose para ello una simulación completa.

genérico con un flujo de corriente de drenador a fuente, indicado en la figura por una flecha, controlado por las condiciones de contorno en las zonas de fuente, drenador, puerta, etc. A esta estructura, dependiendo de su forma geométrica, pueden asimilarse gran cantidad de dispositivos reales como por ejemplo los HEMTs, MOSFETs, HBTs, etc. En este caso las principales variaciones de los parámetros físicos se producen en el plano XZ. Estas características físicas de los sistemas a particionar no son consideradas por las técnicas de particionamiento genéricas. Usando este método, un ejemplo de particionamiento del dispositivo en tres dominios consistiría en realizar las divisiones en planos con Y constante, mostradas en la figura 4.20 a través de las líneas discontinuas.

En esta situación, pueden obtenerse soluciones próximas a la global en cada subdominio, permitiendo la convergencia del resolutor de los sistemas de ecuaciones interno a cada subdominio en menos iteraciones y acelerando así el proceso de simulación. Este método presenta la ventaja de permitir simular dispositivos más anchos, ya que la resolución de los subdominios de forma desacoplada da lugar a un paralelismo de grano grueso que permite trabajar con mallas más grandes y con un mayor número de procesadores.

4.4.2. Resultados numéricos

Los resultados que se presentan en este apartado fueron obtenidos con las mismas mallas que las utilizadas en el apartado anterior, cuyas características fueron descritas en la tabla 4.4.

A continuación se comparan los resultados obtenidos por un lado con el mejor algoritmo proporcionado por el programa METIS y por el otro

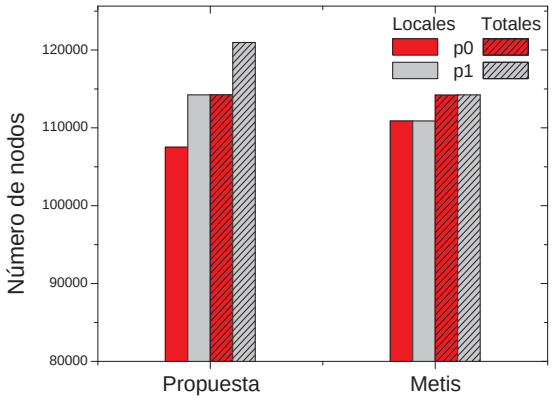


Figura 4.21: Número de nodos locales y totales de los subdominios obtenidos usando nuestra propuesta de particionamiento y el particionamiento dado por METIS. Estos resultados fueron obtenidos, para 2 procesadores, a partir de la malla H.

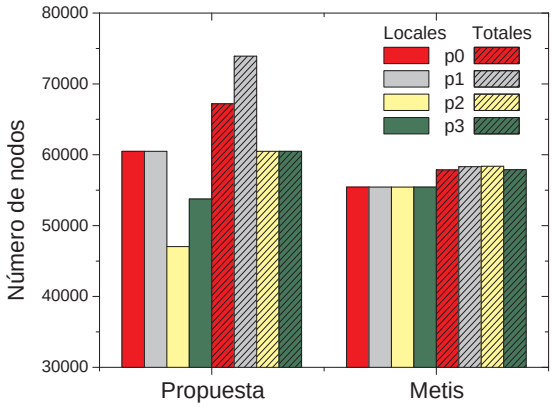


Figura 4.22: Número de nodos locales y totales de los subdominios obtenidos usando nuestra propuesta de particionamiento y el particionamiento dado por METIS. Estos resultados fueron obtenidos, para 4 procesadores, a partir de la malla H.

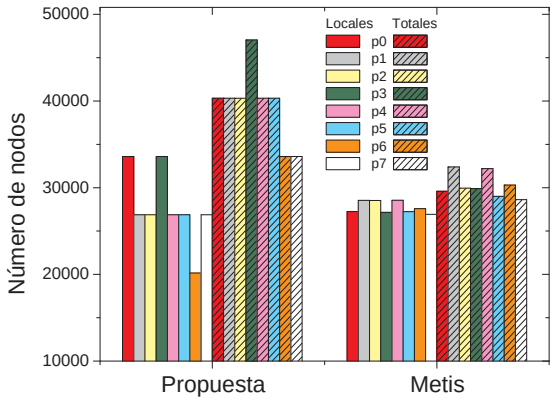


Figura 4.23: Número de nodos locales y totales de los subdominios obtenidos usando nuestra propuesta de particionamiento y el particionamiento dado por METIS. Estos resultados fueron obtenidos, para 8 procesadores, a partir de la malla H.

con el particionamiento propuesto. En las figuras 4.21, 4.22 y 4.23 se representan el número de nodos locales y totales de los subdominios para los particionamientos dados por METIS y por nuestra propuesta, usando 2, 4 y 8 procesadores respectivamente. Estos resultados se obtuvieron a partir de la malla H. El desbalanceo que se genera en nuestra propuesta en algunas situaciones es fácilmente mejorable permitiendo, posteriormente al particionamiento, un reequilibrio del número de nodos de cada subdominio.

Los nodos totales de un subdominio se definen como la suma de los nodos locales a ese subdominio y los nodos frontera externos pertenecientes a otros subdominios pero conectados con nodos internos a éste. En el caso del particionamiento dado por METIS se encuentra que el número de nodos locales es prácticamente igual en todos los subdominios independientemente del número de procesadores, además el número de nodos totales por subdominio está también muy equilibrado y la contribución dada por el número de nodos frontera externos es muy pequeña. Sólo a partir de 8 procesadores se nota un cierto desbalanceo en el número de nodos por procesador. Como ejemplo de este particionamiento, en la figura 4.24 se muestra el patrón de una matriz, originaria de la simulación de dispositivos HEMT, particionada en tres subdominios por medio del programa METIS. Esta figura nos permite observar el reducido número de nodos frontera por subdominio.

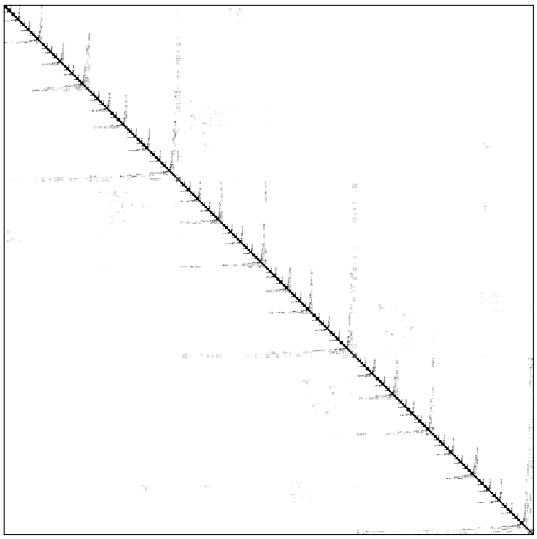


Figura 4.24: Patrón de una matriz particionada con el programa METIS en tres subdominios.

Por otro lado, utilizando nuestra estrategia de particionamiento se obtiene un comportamiento mucho más desbalanceado, encontrando grandes diferencias entre el número de nodos locales por procesador, sobre todo al aumentar el número de subdominios. Además el número de nodos totales del dominio es notablemente superior al número de nodos locales y a los dados por METIS en las mismas circunstancias. Esto es debido a la gran cantidad de nodos frontera externos presentes en cada subdominio. En la figura 4.25 se representa el patrón de una matriz particionada en tres subdominios utilizando para ello nuestra propuesta de particionamiento. En este caso, a diferencia del patrón dado con el particionamiento de METIS, se observa un número considerable de nodos frontera entre subdominios, estando además este número desbalanceado entre los diferentes procesadores.

En las figuras 4.26 y 4.27 se representan las mallas resultantes de un dispositivo HEMT particionadas en 4 procesadores usando METIS y nuestra propuesta respectivamente.

Las tablas 4.5 y 4.11 muestran para la malla S y para los particionamientos dados por METIS y por nuestra propuesta respectivamente, los tiempos promedio por iteración necesarios para realizar una factorización ILU (t_{ILU-O}), para que el resolutor interno alcance la convergencia ($t_{met.iter-O}$), y para obtener la solución del resolutor externo ($t_{resol-O}$). También muestran el número de iteraciones promedio del resolutor interno ($it_{resol-O}$).

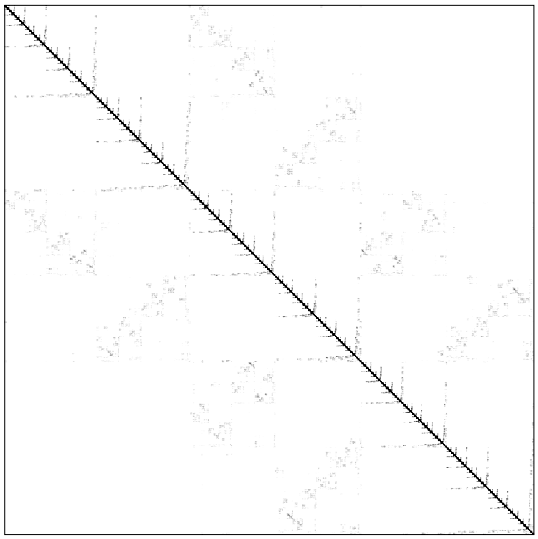


Figura 4.25: Patrón de una matriz particionada en tres subdominios utilizando nuestra propuesta de particionamiento.

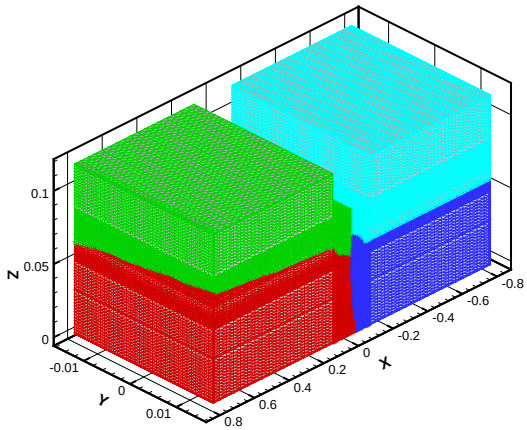


Figura 4.26: Malla de 221760 nodos dividida en 4 subdominios utilizando el programa METIS para un dispositivo HEMT.

Estos resultados se obtuvieron para la versión optimizada de la etapa de resolución de ecuaciones lineales implementada en el simulador, descrita en el apartado anterior. Los tiempos presentados en estas tablas se obtuvieron en el cluster Superdome cuyas características *hardware* fueron comentadas

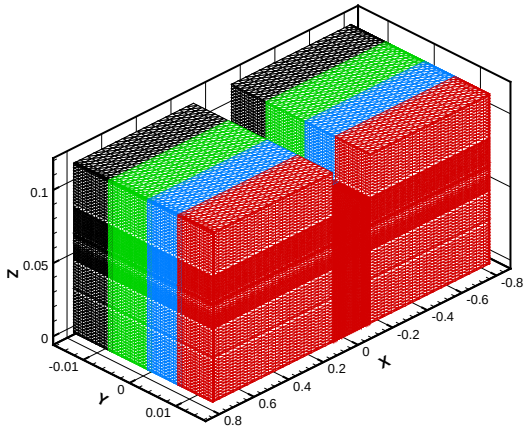


Figura 4.27: Malla de 221760 nodos dividida en 4 subdominios en planos con Y constante para un dispositivo HEMT.

proc	t_{ILU-O}	$t_{met.iter-O}$	$t_{resol-O}$	$it_{resol-O}$
1	20.61	0.45	21.06	2
2	3.18	3.26	6.44	27
4	1.27	2.78	4.05	43
8	0.85	0.62	1.47	57

Tabla 4.11: Tiempos promedio, usando la nueva propuesta de particionamiento de la malla, para: una factorización ILU (t_{ILU}), el resolutor en alcanzar la convergencia ($t_{met.iter}$) y resolver un sistema lineal local (t_{resol}). También se muestra el número promedio de iteraciones del resolutor interno (it_{resol}). Resultados correspondientes a la resolución de la ecuación de Poisson en el equilibrio, utilizando para ello la malla S y la versión optimizada del código.

previamente.

Estos resultados muestran que los tiempos de ejecución obtenidos en el simulador son menores con la nueva propuesta de particionamiento hasta ocho procesadores. Por un lado, esto es debido a la reducción del tiempo de la factorización ILU para un número pequeño de procesadores, aunque este tiempo empeora respecto al obtenido con METIS a partir de 8 procesadores. Hay que tener en cuenta que cuando se realiza la factorización de la matriz, los nodos internos son los únicos involucrados en esta operación,

con lo que no tiene importancia que el número de nodos frontera aumente considerablemente respecto al particionamiento dado por METIS. Por otro lado, el tiempo del resolutor interno también disminúye. Este descenso en el tiempo del método iterativo se consigue incluso para ocho procesadores, a pesar de que a partir de cuatro procesadores aumenta el número promedio de iteraciones por encima del valor dado con el particionamiento METIS. Es necesario considerar que estos resultados son muy dependientes del tamaño del problema, para mallas más grandes las ventajas de nuestro particionamiento se extienden a un mayor número de procesadores.

Para solucionar el problema del empeoramiento de la eficiencia paralela para un número de procesadores elevado es necesaria la implementación de un nuevo algoritmo de resolución de los sistemas de ecuaciones en el simulador 3D de dispositivos. Esta nueva implementación, que es parte de nuestro trabajo futuro, consiste en aprovechar en una mayor medida el paralelismo en grano grueso que permite este tipo de particionamiento, evitando realizar comunicaciones en cada iteración del lazo externo, tal y como se hace actualmente, y resolviendo cada subdominio de forma mucho más independiente. De esta forma se lograría minimizar las comunicaciones entre procesadores, puesto que sólo serían necesarias cada cierto número de iteraciones.

4.5. Resumen

Básicamente, en este capítulo se ha formulado una pregunta:

- ¿Cómo mejorar la eficiencia paralela del simulador 3D de dispositivos?

Para obtener la respuesta a esta pregunta ha sido necesario en primer lugar la resolución de los siguientes interrogantes:

- ¿Cual es la parte más costosa del proceso de simulación?
- ¿Cómo se puede minimizar el tiempo de simulación empleado por esa parte?

Se ha encontrado que la etapa de resolución de los sistemas de ecuaciones lineales dispersos es la que consume aproximadamente un 90 % del tiempo total de simulación. Por lo tanto la optimización del simulador 3D se ha centrado básicamente en esta etapa del proceso de simulación y se han estudiado diferentes estrategias para lograr minimizar su tiempo de computación.

En primer lugar se ha realizado un análisis de los métodos de resolución implementados en una serie de librería numéricas con el objetivo de obtener aquellos que sean más eficientes para nuestro problema particular. En

segundo lugar, los métodos de resolución más eficientes son utilizados en el simulador. A continuación se han presentado dos estrategias para mejorar la eficiencia paralela del simulador. En la primera se modifica la implementación realizada de la etapa de resolución de los sistemas de ecuaciones lineales, lográndose reducir considerablemente los tiempos de ejecución. En la segunda, se presenta otra alternativa para la optimización del simulador basada en utilizar una nueva estrategia de particionamiento de las mallas tetraédricas de elementos finitos.

Capítulo 5

Fluctuaciones en los parámetros intrínsecos de dispositivos HEMT

En el pasado, el desajuste en las características de los transistores fabricados bajo idénticas condiciones estaba relacionado principalmente con variaciones en los parámetros asociados con los procesos de fabricación, lo que daba lugar a variaciones macroscópicas principalmente en el grosor de las capas, la geometría y el perfil del dopado en el interior del dispositivo.

El escalado de los dispositivos a dimensiones del orden de los nanómetros ha provocado la aparición de nuevos problemas asociados con la naturaleza discreta de la materia y de la carga. Además, estos efectos no pueden ser eliminados por medio de mejores etapas de procesado o avances en el equipamiento ya que son intrínsecos a las propiedades de los materiales [3].

Los efectos combinados de todas las fluctuaciones en los parámetros intrínsecos de los dispositivos tienen un impacto significativo en la funcionalidad, rendimiento y fiabilidad de sistemas y circuitos [112].

Convencionalmente, cuando se realizaban simulaciones numéricas de transistores se consideraban dispositivos perfectos con fronteras suaves y distribuciones continuas de dopado. La inclusión de diversos tipos de fluctuaciones en los parámetros intrínsecos provoca variaciones estadísticas dentro de un conjunto de dispositivos. Por lo tanto, la simulación de un único dispositivo perfecto ya no es suficiente, siendo preciso simular un conjunto de dispositivos diferentes a nivel microscópico para considerar estadísticamente el efecto de diferentes fuentes de fluctuaciones. Así, para poder entender y predecir el comportamiento de un dispositivo en el interior de un circuito, será necesario conocer los valores medios y varianzas de las distribuciones estadísticas de

parámetros de diseño como la corriente, la transconductancia o la tensión umbral.

En la primera parte de este capítulo se definen brevemente los parámetros estadísticos que serán utilizados posteriormente en los resultados numéricos presentados. En el segundo apartado se justifica la necesidad de utilizar el modelo de arrastre–difusión para el cálculo de las fluctuaciones de parámetros intrínsecos, además, se citan las principales limitaciones de este modelo. A continuación se describen las características de los dispositivos utilizados en el estudio, así como el proceso de calibración seguido. En el siguiente apartado se presentan las principales fuentes de fluctuaciones en los dispositivos MOSFET, en los que estos efectos han sido ampliamente estudiados y documentados. A continuación se introducen las principales fuentes de fluctuaciones de parámetros intrínsecos para dispositivos HEMT analizadas en este trabajo, que son, la variación aleatoria en la composición de los compuestos ternarios que constituyen el canal del dispositivo, la consideración de la naturaleza discreta de los dopantes y la variación aleatoria de la carga interfacial presente en la frontera de alguna zona del dispositivo. Por último se presentan los resultados numéricos en los que, en primer lugar, se analiza la influencia de la presencia de carga interfacial en el comportamiento de los dispositivos. A continuación se estudia el impacto, por medio de análisis estadísticos, de las fuentes de fluctuaciones antes mencionadas en las curvas características de los dispositivos HEMT. Para finalizar se muestra un estudio del impacto de las fluctuaciones de parámetros intrínsecos en la frecuencia de corte de los dispositivos.

5.1. Introducción a la estadística básica

En esta sección, se realiza una breve introducción sobre estadística básica, centrándose en las definiciones de los términos utilizados en el estudio del impacto de las fluctuaciones de parámetros intrínsecos presentado en la parte final del capítulo. Es posible encontrar una descripción más detallada de las variables definidas en esta sección y sus propiedades en [113].

Partiendo de un experimento aleatorio, con espacio muestral asociado Ω, una variable aleatoria es cualquier función, X, que asocia a cada suceso elemental un número real, verificando que, para cualquier número real r, es un suceso el conjunto:

$$\{\omega \in \Omega / X(w) \leq r\} = X^{-1}((-\infty, r]) \tag{5.1}$$

Hay que tener en cuenta que la probabilidad definida en el espacio muestral

no influye en la definición anterior. Únicamente influye el tipo de sucesos de la respectiva álgebra asociada.

Se define la función de distribución de una variable aleatoria X, como una función real que a cada número real x, le asocia una probabilidad de que la variable tome valores menores o iguales a dicho número, esto es:

$$F(x) = P(X \leq x) = P(\{\omega \in \Omega / X(w) \leq x\}) = P(X^{-1}((-\infty, x])) \quad (5.2)$$

La definición de F está garantizada porque el conjunto $X^{-1}((-\infty, x])$ es un suceso, según la definición de variable aleatoria.

Las propiedades de una función de distribución de una variable aleatoria son:

1. $0 \leq F(X) \leq 1$
2. F es no decreciente.
3. $F(+\infty) = 1 \qquad (\lim_{x \to +\infty} F(x) = 1)$
4. $F(-\infty) = 0 \qquad (\lim_{x \to -\infty} F(x) = 0)$
5. F es continua por la derecha.

5.1.1. Variables aleatorias discretas

Una variable aleatoria discreta es aquella que sólo puede tomar valores dentro de un conjunto finito o infinito numerable. Sea X una variable aleatoria discreta que toma valores $x_1, x_2, ..., x_n, ...$ Al conjunto de probabilidades $p_1, p_2, ..., p_n, ...$, de forma que $p_i = P(X = x_i)$ con $\sum_i p_i = 1$ se le denomina función de masa de probabilidad o función de probabilidad de la variable X.

La función de distribución de una variable discreta es una función real, de variable real, que asocia a cada número x la probabilidad acumulada hasta ese valor. Está dada como suma de la función de masa de probabilidad, del siguiente modo:

$$F(x) = P(X \leq x) = \sum_{x_i \leq x} p_i \quad (5.3)$$

La función F es escalonada, no decreciente, con saltos de discontinuidad en los puntos x_i iguales a la probabilidad en dichos puntos p_i.

5.1.2. Variables aleatorias continuas

Una variable aleatoria es continua si toma valores en uno o varios intervalos de la recta real. Es necesario tener en cuenta que la medida de los

resultados de muchos experimentos es aproximada, pues no se puede precisar la exactitud mediante un proceso de medición físico. Así, cuando se recogen los datos de una variable estadística se clasifican los resultados en clases con sus respectivas frecuencias, al hacer las clases más y más finas, se observa que los histogramas de frecuencias correspondientes se aproximan a una curva. De este modo surge el concepto de función de densidad como la función límite de los histogramas.

Así, dada una variable aleatoria continua X, la probabilidad de un intervalo (a,b) será el área limitada por esta función de densidad, las rectas $x = a$, $x = b$ y el eje de abscisas. La probabilidad de que la variable aleatoria tome un valor concreto es igual a cero, esto es $P(X = x_0) = 0$. Sin embargo, tiene sentido analizar lo densa que está repartida la probabilidad en torno a ese valor, dando lugar a la siguiente definición.

Dada una variable aleatoria continua X, la función de densidad es la función real de variable real:

$$f(x) = \lim_{h \to 0^+} \frac{P(x - h \leq X \leq x + h)}{2h} \tag{5.4}$$

Dada una variable continua X, con función de distribución F, la función de densidad f es la función que resulta de derivar la función de distribución:

$$f(x) = F'(x); \qquad F(x) = \int_{-\infty}^{\infty} f(t)dt \tag{5.5}$$

La función f describe el comportamiento idealizado de la variable aleatoria continua asociada, reflejando para cada intervalo real, sobre el que tome valores la variable, su densidad de probabilidad. Geométricamente $F(x_0)$ mide el área de la región limitada por la función de densidad, el eje X y la recta de ecuación $x = x_0$.

Las principales propiedades de la función de densidad son:

1. $f(x) \geq 0$; $-\infty < x < \infty$

2. El área total bajo la curva de la función de densidad es 1:

$$\int_{-\infty}^{\infty} f(x)dx = F(+\infty) = 1 \tag{5.6}$$

3. La probabilidad del intervalo $[a,b]$ es el área bajo la curva $f(x)$ entre las rectas $x = a$, $x = b$ y el eje de abscisas.

$$P(a \leq X \leq b) = \int_a^b f(t)dt = F(b) - F(a) \tag{5.7}$$

4. La probabilidad de que una variable continua tome un único valor es nula.

$$P(X = x_0) = \int_{-x_0}^{x_0} f(t)dt = 0 \tag{5.8}$$

En general, cualquier función real que verifica las propiedades 1 y 2 es la función de densidad de alguna variable aleatoria continua X.

5.1.3. Características de una variable aleatoria

La esperanza matemática es un concepto que surge ligado a los juegos de azar, como forma de conocer la ganancia esperada de un juego. A modo de introducción, considerando como ejemplo el juego de la ruleta, si se apuesta a *par* e *impar* un total de n veces, suponiendo que sale n_p veces *par* con una ganancia de x_p por cada vez y n_i veces *impar* con una ganancia de x_i, la ganancia total en n apuestas, apostando siempre a *par*, vendría dada por:

$$n_p x_p - n_i x_i \tag{5.9}$$

La ganancia media se obtiene dividiendo la expresión anterior por el número de pruebas, es decir:

$$\overline{x} = \frac{n_p x_p - n_i x_i}{n} = \frac{n_p}{n} x_p - \frac{n_i}{n} x_i = f_p x_p - f_i x_i \tag{5.10}$$

siendo $\overline{x}$ la media de una variable estadística.

Para extender este concepto a una variable aleatoria basta con cambiar las frecuencias f_p y f_i por las respectivas probabilidades para la variable aleatoria X, definida por:

$$X(\omega) = \begin{cases} x_p & \text{con} \quad P(X = x_p) = p_p \\ -x_i & \text{con} \quad P(X = x_i) = p_i \end{cases} \tag{5.11}$$

Así, la media de la variable aleatoria $X = \textit{ganancia del juego}$ es:

$$E(X) = x_p p_p + (-x_i) p_i \tag{5.12}$$

Generalizando lo anterior, dada X una variable aleatoria discreta que toma los valores $x_1, x_2, ..., x_n, ...$ con probabilidades $p_1, p_2, ..., p_n, ...$, la media o esperanza matemática de la variable X es el número real:

$$\mu = E(X) = \sum_i x_i p_i \tag{5.13}$$

Si X es una variable aleatoria continua con función de densidad f su media o esperanza matemática es el número real:

$$\mu = E(X) = \int_{-\infty}^{+\infty} x f(x) dx \tag{5.14}$$

En ambos casos se cumplen las siguientes propiedades:

1. $E(aX + b) = aE(X) + b$
2. $E(X + Y) = E(X) + E(Y)$
3. Si g es una función real, es posible extender la definición de media para la variable $g(X)$ del modo siguiente:

$$E(g(X)) = \sum_i g(x_i)p_i \qquad (Xdiscreta) \tag{5.15}$$

$$E(g(X)) = \int_{-\infty}^{+\infty} g(x)f(x)dx \qquad (Xcontinua) \tag{5.16}$$

4. Si tenemos una variable no negativa con media 0, la variable tomará sólo el valor 0.

Sea X una variable aleatoria con media $\mu = E(x)$, la varianza de X es el valor esperado de los cuadrados de las diferencias con la media:

$$\sigma^2 = Var(X) = E((X - E(X))^2) \tag{5.17}$$

Además, se define la desviación típica de la variable X como la raíz positiva de la varianza:

$$\sigma = +\sqrt{E((X - E(X))^2)} \tag{5.18}$$

La desviación típica es la medida de dispersión más utilizada en la práctica. Al contrario de la varianza, presenta la ventaja de dar los resultados en las mismas unidades que los valores de la variable.

Una variable aleatoria se dirá que está estandarizada o tipificada si su media es 0 y su varianza es 1. Para transformar una variable X con media μ y desviación típica σ en otra tipificada, basta con aplicar un cambio de variable tipo lineal $Y = \frac{X-\mu}{\sigma}$. Esta nueva variable Y tiene por media 0 y por varianza 1. Tipificar una variable es útil para la comparación de distintas variables medidas con unidades diferentes.

La varianza es una medida de dispersión que no es invariante a cambios de escala, por lo que resulta útil construir otra medida que mida la dispersión y permanezca inalterada si se produce un cambio de escala de los datos. De esta forma se define el coeficiente de variación de la variable X con media μ y desviación típica σ como:

$$CV(X) = \frac{\sigma}{\mu} \tag{5.19}$$

siempre que $\mu \neq 0$.

El momento central de orden k respecto al origen, representado por α_k, de una variable aleatoria X se define como:

$$\alpha_k = E(X^k) \tag{5.20}$$

De igual forma, el momento central de orden k, o momento respecto a la media de orden k:

$$\mu_k = E((X - E(X))^k) \tag{5.21}$$

La mediana es otra medida muy usada que caracteriza una variable aleatoria. La mediana de una variable aleatoria es una medida de centralización que divide la función en dos partes de igual probabilidad. La denotaremos por el valor M_e, que para una variable con distribución F, verifica:

$$F(M_e) = \frac{1}{2} \tag{5.22}$$

La moda de una variable es el valor M_0 que maximiza la función de probabilidad o la función de densidad, según se trate de una variable discreta o continua, respectivamente.

El coeficiente de asimetría se define como el cociente:

$$C.A. = \frac{\mu_3}{\sigma^3} \tag{5.23}$$

Si $C.A$ es 0, o próximo a 0, la distribución es simétrica. En otro caso presentará asimetría positiva o negativa de acuerdo con el signo de $C.A$.

El coeficiente de apuntamiento o kurtosis se define como el número:

$$Kurt. = \frac{\mu_4}{\sigma^4} - 3 \tag{5.24}$$

Si el valor de la kurtosis es 0, o próximo a 0, la distribución presenta una forma similar a la distribución normal.

La distribución normal o gaussiana es la más importante y de mayor uso de todas las distribuciones continuas de probabilidad. Existen multitud de ejemplos cuyo resultado se ajusta a esta distribución, por ejemplo: el peso de una persona y su talla, datos meteorológicos, errores de medición, calificaciones de pruebas de aptitud, etc.

Una variable aleatoria tiene distribución normal de parámetros (μ,σ) si su función de densidad es:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \qquad si -\infty < x < \infty \tag{5.25}$$

Las principales características de esta distribución son:

1. $E(X) = \mu$
2. $Var(X) = \sigma^2$
3. El coeficiente de kurtosis de una normal vale 0.

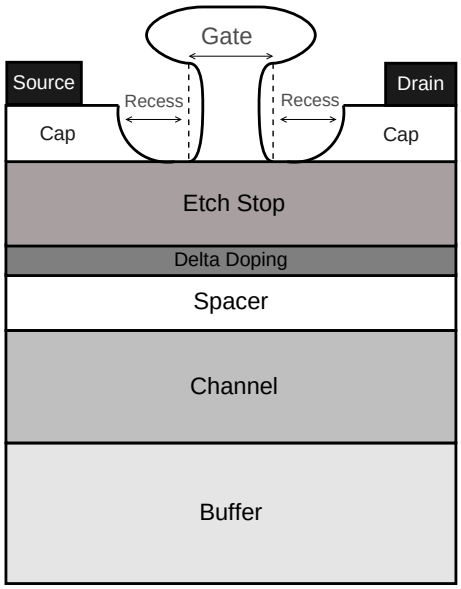


Figura 5.1: Representación esquemática de un dispositivo HEMT genérico.

5.2. Modelo de arrastre–difusión: necesidad y limitaciones

La mayoría de las fuentes de fluctuaciones de parámetros intrínsecos son de naturaleza tridimensional y, por lo tanto, para capturar correctamente sus efectos, es necesario realizar simulaciones 3D, que son, computacionalmente mucho más costosas que las simulaciones en dos dimensiones. Además, la necesidad de realizar simulaciones estadísticas multiplica la complejidad del problema por el tamaño de la muestra estadística [112, 114].

	$N_{eff}(cm^{-3})$	$\Delta X(\mu m)$	$\Delta Y(\mu m)$	$\Delta Z(\mu m)$
Cap ($GaAs$)	$4.0\,10^{18}$	0.690	0.030	0.030
Etch Stop ($Al_{0.3}Ga_{0.7}As$)	$1.0\,10^{14}$	1.600	0.030	0.018
δ-doping ($Al_{0.3}Ga_{0.7}As$)	$1.75\,10^{19}$	1.600	0.030	0.002
Spacer ($Al_{0.3}Ga_{0.7}As$)	$1.0\,10^{14}$	1.600	0.030	0.006
Canal ($In_{0.2}Ga_{0.8}As$)	$2.0\,10^{14}$	1.600	0.030	0.010
Buffer ($GaAs$)	$1.0\,10^{14}$	1.600	0.030	0.300

Tabla 5.1: Dimensiones, composiciones y dopados de las diferentes capas del dispositivo PHEMT de 120 nm de longitud de puerta.

	$N_{eff}(cm^{-3})$	$\Delta X(\mu m)$	$\Delta Y(\mu m)$	$\Delta Z(\mu m)$
Cap ($In_{0.53}Ga_{0.47}As$)	$1.0\,10^{19}$	0.500	0.030	0.020
Etch Stop ($In_{0.52}Al_{0.48}As$)	$1.0\,10^{14}$	1.070	0.030	0.009
δ-doping ($In_{0.52}Al_{0.48}As$)	$2.0\,10^{19}$	1.070	0.030	0.002
Spacer ($In_{0.52}Al_{0.48}As$)	$1.0\,10^{14}$	1.070	0.030	0.004
Canal ($In_{0.7}Ga_{0.3}As$)	$2.0\,10^{14}$	1.070	0.030	0.012
Buffer ($In_{0.52}Al_{0.48}As$)	$1.0\,10^{15}$(*)	1.070	0.030	0.300

Tabla 5.2: Dimensiones, composiciones y dopados de las diferentes capas del dispositivo HEMT de 50 nm de longitud de puerta. El símbolo (*) indica que en esta capa, a diferencia de en el resto de las regiones del dispositivo, el dopado es tipo P.

La técnica empleada en este tipo de simulaciones debe ser rápida y eficiente permitiendo la simulación de un conjunto grande de dispositivos en un tiempo de computación lo menor posible. Teniendo esto en cuenta, la aproximación de arrastre–difusión es la más adecuada para este tipo de análisis, ya que proporciona un compromiso entre la calidad de la solución y el tiempo computacional necesario para alcanzarla [3, 112, 115, 116].

Por otro lado, hay que tener en cuenta que la aproximación arrastre–difusión no captura correctamente el transporte de portadores fuera del equilibrio en dispositivos de dimensiones inferiores a los 100 nm y por lo tanto subestima su corriente de drenador [39, 117]. Sin embargo, siendo siempre conscientes de sus limitaciones, es perfectamente adecuada para dar una estimación de la influencia de diferentes fuentes de fluctuaciones asociadas con la electrostática del dispositivo en las curvas características y en su funcionamiento, con un coste computacional medianamente asequible para poder realizar la gran cantidad de simulaciones que implican el estudio de fluctuaciones de parámetros intrínsecos.

5.3. Transistores HEMT: estructura y calibración

El estudio del efecto de las fluctuaciones de parámetros intrínsecos en HEMTs realizado en este trabajo se centra en dos dispositivos diferentes, un dispositivo PHEMT de 120 nm de longitud de puerta y un dispositivo HEMT de 50 nm de longitud de puerta. A continuación, en este apartado se describe la estructura de los dos dispositivos HEMT utilizados en este estudio y el proceso de calibración seguido hasta lograr su correcto funcionamiento.

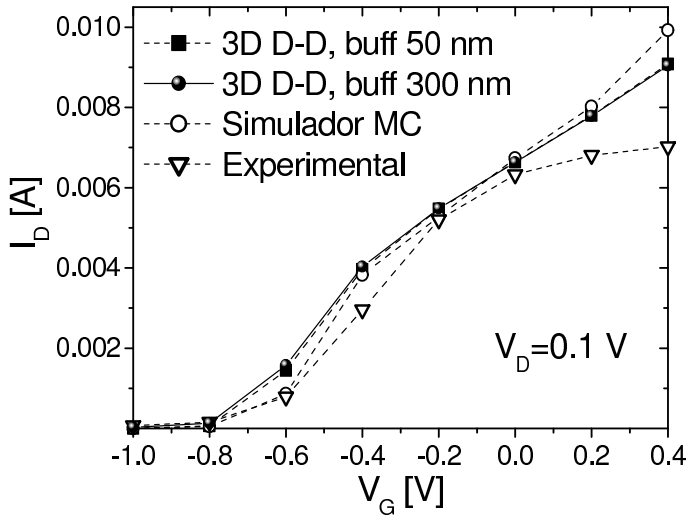


Figura 5.2: Curvas características, en escala lineal, a una tensión de drenador de 0.1 V para el dispositivo PHEMT de 120 nm de longitud de puerta. Los resultados obtenidos con el simulador tridimensional de arrastre–difusión son comparados con los obtenidos experimentalmente y con un simulador 2D Monte Carlo.

5.3.1. Estructura de los dispositivos

Un esquema básico de los dos dispositivos HEMT utilizados en este trabajo se muestra en la figura 5.1.

La estructura vertical del dispositivo PHEMT de 120 nm de longitud de puerta incluye una capa *cap* de 30 nm de n^+GaAs dopada con Si, con una concentración de 4×10^{18}cm^{-3}, una capa *etch stop* de 18 nm de $Al_{0.3}Ga_{0.7}As$, una capa δ*–doping* dopada con Si, con una concentración de 3.5×10^{12}cm^{-2}, una capa *spacer* de 6 nm de $Al_{0.3}Ga_{0.7}As$ que separa la capa δ*–doping* del canal de $In_{0.2}Ga_{0.8}As$ con 10 nm de grosor. Todo el dispositivo se crece sobre un *buffer* de 300 nm de grosor de GaAs. En la tabla 5.1 se muestran en detalle el dopado, la composición y las dimensiones de las diferentes regiones que componen este dispositivo.

En el simulador tridimensional basado en arrastre–difusión se utiliza un modelo de movilidad de campo elevado [118]. Entre los parámetros de este modelo dependientes del material utilizado se encuentra la movilidad a campo pequeño y la velocidad de saturación. Los valores que se han utilizado son una movilidad a campo pequeño de 3800 cm^2/Vs y una velocidad de saturación de 2.5×10^7 cm/s para el canal de $In_{0.2}Ga_{0.8}As$, y de 3000 cm^2/Vs

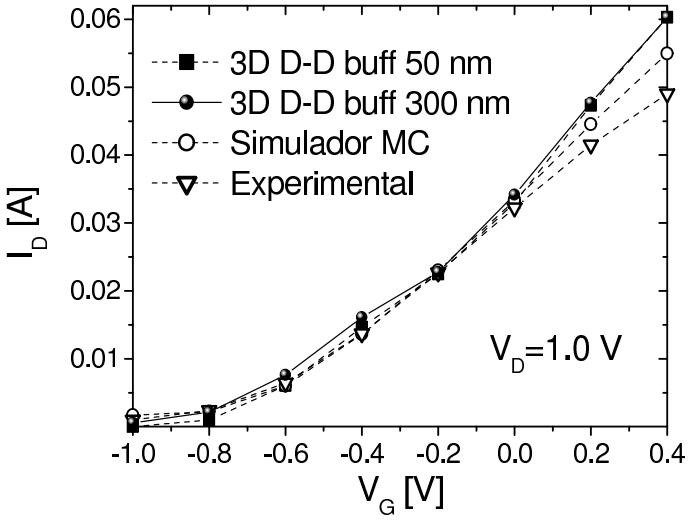


Figura 5.3: Curvas características, en escala lineal, a una tensión de drenador de 1.0 V para el dispositivo PHEMT de 120 nm de longitud de puerta.

y 1.0×10^7 cm/s para las capas de $Al_{0.3}Ga_{0.7}As$.

El dispositivo HEMT de 50 nm de longitud de puerta está formado por una capa *cap* de 20 nm de $In_{0.53}Ga_{0.47}As$ dopada con Si, siendo su concentración de $10^{19}cm^{-3}$, una capa *etch stop* de $In_{0.52}Al_{0.48}As$ de 9 nm de grosor, una capa δ*–doping* con un dopado de $4\times10^{12}cm^{-2}$ situada encima de una capa *spacer* de $In_{0.52}Al_{0.48}As$ de 4 nm de grosor y un canal de 12 nm de $In_{0.7}Ga_{0.3}As$. Todas las capas activas en el dispositivo se crecen sobre una capa *buffer* dopada tipo P muy gruesa, de 300 nm de ancho, de $In_{0.52}Al_{0.48}As$. En la tabla 5.2 se muestran los valores del dopado, la composición y las dimensiones de las diferentes capas que constituyen este dispositivo.

En este dispositivo HEMT, de igual modo que en el PHEMT de 120 nm, se utiliza una movilidad a campo pequeño de 5000 cm^2/Vs y una velocidad de saturación de 4.0×10^7 cm/s para el canal de $In_{0.7}Ga_{0.3}As$, y de 3000 cm^2/Vs y 1.0×10^7 cm/s para las capas de $In_{0.52}Al_{0.48}As$, respectivamente.

5.3.2. Calibración de los dispositivos

El estudio de las fluctuaciones de parámetros intrínsecos se basa en la obtención de las curvas características, I_D–V_G, de los dos dispositivos simulados a diferentes valores de la tensión de drenador.

La validez de estos resultados se justifica por medio de la calibración de

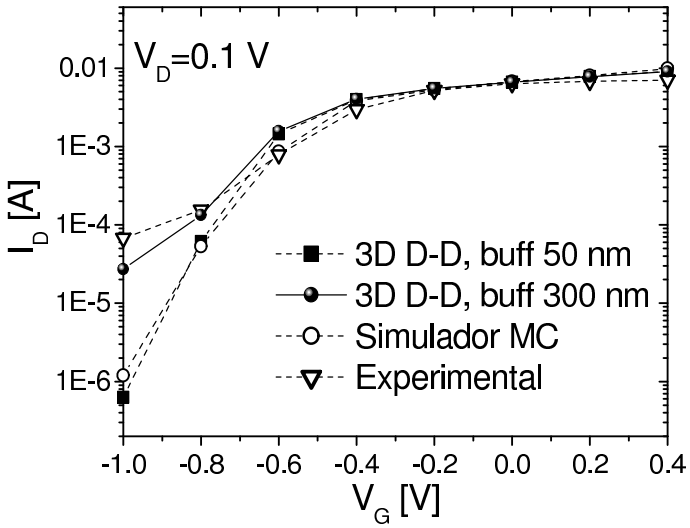


Figura 5.4: Curvas características, en escala logarítmica, a una tensión de drenador de 0.1 V para el dispositivo PHEMT de 120 nm de longitud de puerta.

las curvas características logradas con el simulador, por medio de su comparación con los resultados experimentales obtenidos con los dispositivos físicos, diseñados y fabricados por el Centro de Investigación Nanotecnológica de la Universidad de Glasgow, y con los resultados de un simulador Monte Carlo H2F/MC [39, 119] desarrollado en el Device Modelling Group perteneciente también a la Universidad de Glasgow. La calibración se realiza tanto en escala lineal como en escala logarítmica.

Las figuras 5.2 y 5.3 comparan las curvas características I_D–V_G para valores de tensión de drenador de 0.1 V y 1.0 V en escala lineal, para el dispositivo PHEMT de 120 nm de longitud de puerta. Se presentan resultados del simulador 3D para dos valores de ancho del buffer, 50 y 300 nm, aunque en escala lineal apenas se aprecian diferencias entre ambas medidas. Los resultados obtenidos directamente del simulador 3D paralelo basado en el modelo de arrastre–difusión no incluyen resistencias en los contactos, por lo que su corriente de drenador es superior a la obtenida con los resultados experimentales para valores de tensión de puerta elevados. Por lo tanto, la curva característica obtenida con el simulador tridimensional es calibrada utilizando las curvas características intrínsecas obtenidas directamente de simulaciones MC. Estas simulaciones no incluyen efectos de deplexión del potencial en superficie en las regiones de *recess* del dispositivo puesto que el simulador 3D no permite la inclusión de este efecto. Hay que tener en cuen-

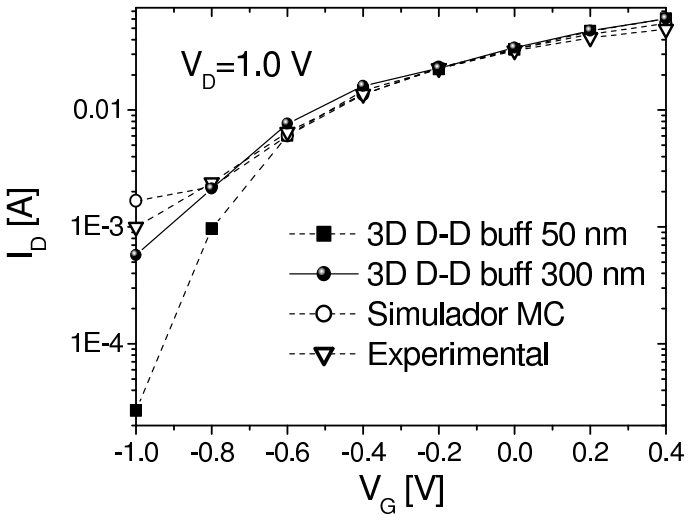


Figura 5.5: Curvas características, en escala logarítmica, a una tensión de drenador de 1.0 V para el dispositivo PHEMT de 120 nm de longitud de puerta.

ta que el impacto de las resistencias externas empieza a tener importancia para $V_G > 0.2$ V tanto para valores de tensión de drenador reducida como elevada y puede ser introducido en las simulaciones en una posterior etapa de post–procesado [120].

Las figuras 5.4 y 5.5 muestran las curvas características obtenidas, para el dispositivo de 120 nm de longitud de puerta, a una V_D de 0.1 y 1.0 V respectivamente, en escala logarítmica. En esta escala es posible apreciar la diferencia que supone el ancho del buffer en la corriente de drenador de la zona sub–umbral. Así, para una $V_D = 0.1$ V los resultados obtenidos, con el simulador MC y con el simulador 3D paralelo con un ancho de buffer de 50 nm, se alejan de los experimentales para tensiones de puerta inferiores a -0.6 V. En cambio, si se utiliza un buffer con un ancho de 300 nm, los resultados dados por el simulador 3D paralelo también son muy próximos a los experimentales a valores muy bajos de la tensión de puerta. Un comportamiento muy similar se encuentra a $V_D = 1.0$ V. En esta situación, tanto el simulador MC como el simulador 3D paralelo con un ancho de buffer de 300 nm, reflejan correctamente los resultados experimentales, mientras que los resultados dados por el simulador paralelo con un buffer de 50 nm se desvían bastante de los experimentales a V_G menores de -0.6 V. A pesar de ello, en el estudio del impacto de las fluctuaciones intrínsecas en este dispositivo, el simulador utiliza el buffer de 50 nm. Esta elección es debida a que una

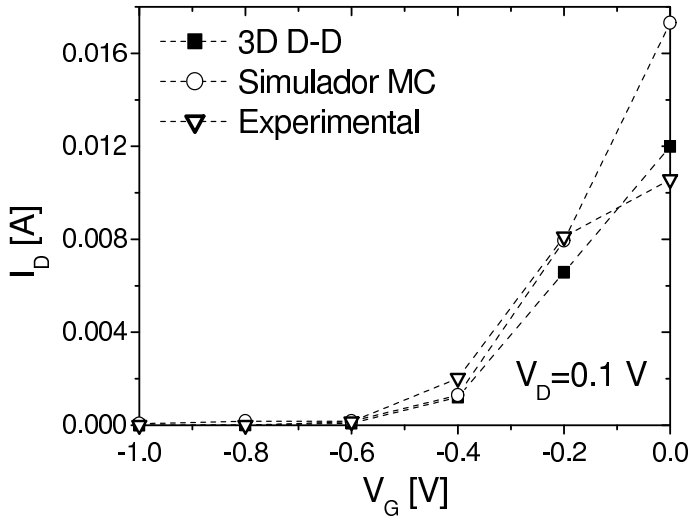


Figura 5.6: Curvas características, en escala lineal, a una tensión de drenador de 0.1 V para el dispositivo HEMT de 50 nm de longitud de puerta. Los resultados obtenidos con el simulador tridimensional de arrastre–difusión son comparados con los obtenidos experimentalmente y con un simulador 2D Monte Carlo.

menor dimensión del buffer consige simulaciones más rápidas, puesto que el menor tamaño de los tetraedros de la malla de elementos finitos en esta región permite una más rápida convergencia del problema. Además, el uso de este tamaño de buffer no perjudica a la calidad del estudio elaborado, puesto que el análisis de las fluctuaciones se realiza tan sólo para valores de tensión de puerta positivos, en los que apenas se encuentran diferencias en los resultados para las dos dimensiones de buffer utilizadas.

En escala lineal, una calibración similar se muestra en las figuras 5.6 y 5.7 para el dispositivo HEMT de 50 nm de longitud de puerta a tensiones de drenador de 0.1 V y 0.8 V respectivamente. A causa del elevado transporte fuera del equilibrio en el HEMT de 50 nm, la corriente de drenador intrínseca obtenida con el simulador 3D de arrastre–difusión es ligeramente inferior, para bajo y alto voltaje de drenador, que la obtenida de simulaciones MC para tensiones de puerta superiores a -0.2 V y 0.0 V respectivamente. Las figuras 5.8 y 5.9 muestran las curvas características para este dispositivo en escala logarítmica. Se observa que a un valor bajo de tensión de drenador de 0.1 V, el simulador 3D paralelo modela adecuadamente el comportamiento experimental, incluso mejor que el simulador MC, en la zona sub–umbral. En cambio al aumentar la tensión de drenador hasta 0.8 V, el simulador paralelo

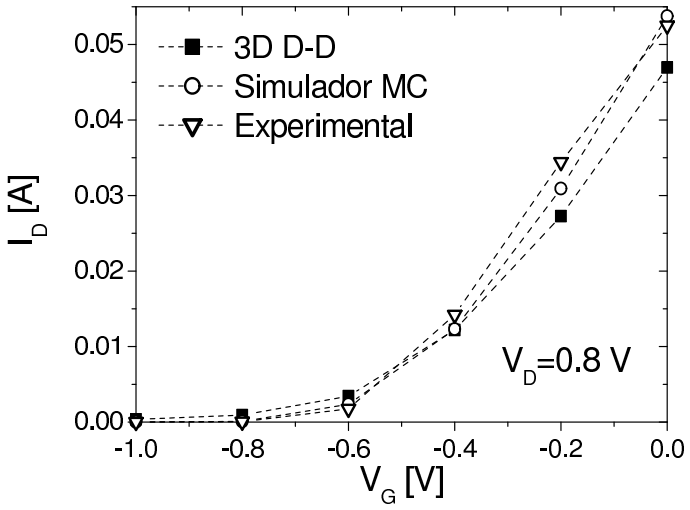


Figura 5.7: Curvas características, en escala lineal, a una tensión de drenador de 0.8 V para el dispositivo HEMT de 50 nm de longitud de puerta.

no consigue ajustarse a los resultados experimentales para una tensión de puerta inferior a -0.6 V.

Los valores de calibración presentados hasta el momento, para los dos dispositivos analizados, han sido los utilizados en el estudio de la influencia de las fluctuaciones de parámetros intrínsecos en los dispositivos presentado en las secciones siguientes.

En el caso del dispositivo HEMT de 50 nm de longitud de puerta, dados los desajustes obtenidos en la calibración tanto a bajo como a alto voltaje de drenador, sobre todo en escala logarítmica, se presenta a continuación una calibración más meticulosa con el objetivo de obtener valores de la corriente de drenador más próximos a los experimentales. Para ello se han realizado variaciones de una serie de parámetros con una importante influencia en el comportamiento del simulador. Así, se ha modificado el ancho de la zona de buffer y la movilidad y la velocidad de saturación en la zona del canal.

Las figuras 5.10 y 5.11 muestran las curvas características a tensión de drenador de 0.1 V en la escala lineal y logarítmica respectivamente. En escala lineal tan sólo se representan los valores hasta un valor de V_G=-0.6 V, puesto que las principales variaciones entre las distintas curvas características se localizan hasta este valor de tensión de puerta. En estas figuras se representan, además de los resultados dados por el dispositivo experimental, el simulador MC y el simulador 3D con los parámetros antes mencionados, dos nuevas curvas, denotadas en las gráficas por $p0$ y $p1$ que representan

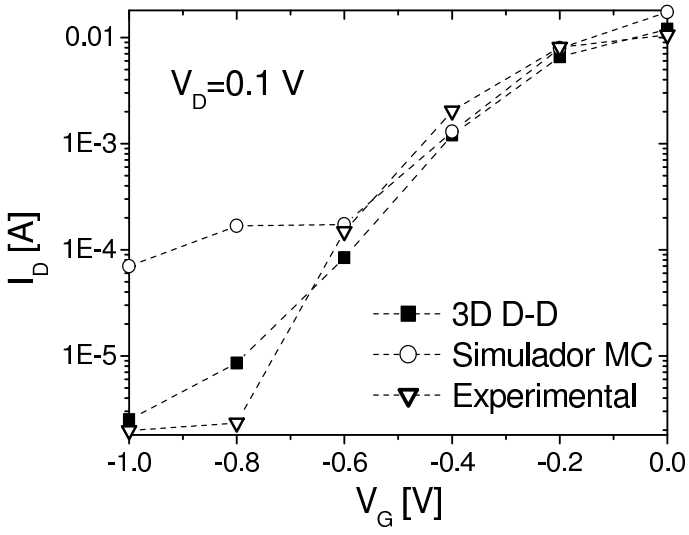


Figura 5.8: Curvas características, en escala logarítmica, a una tensión de drenador de 0.1 V para el dispositivo HEMT de 50 nm de longitud de puerta.

las curvas de calibración más aproximadas a los resultados experimentales obtenidas. Para ello se ha aumentado el ancho del buffer, de 300 a 500 nm, la movilidad a campo pequeño en el canal, de 5000 a 10000 $\mathrm{cm}^2/\mathrm{Vs}$ y la velocidad de saturación en el caso de la curva $p1$, de 4.0×10^7 a 4.5×10^7 cm/s. Los resultados mostrados en las gráficas reflejan que las dos nuevas curvas, $p0$ y $p1$, se ajustan mejor a la experimental que la curva ($3D$ *D-D*), a todos los valores de tensión de puerta excepto a $\mathrm{V_G} = -1.0$ V.

Por último, las figuras 5.12 y 5.13 muestran las curvas características a tensión de drenador de 0.8 V en la escala lineal y logarítmica. En estas figuras se ha realizado un calibrado idéntico al presentado en el párrafo anterior. En esta ocasión se incluyen dos nuevas curvas, $p2$ y $p3$, en las que la única diferencia que presentan en sus características con respecto a las dadas para las curvas $p0$ y $p1$, es el valor de la velocidad de saturación. Así, la curva $p2$ modifica el valor de la velocidad de saturación de 4.0×10^7 a 5.0×10^7 cm/s y la curva $p3$ incrementa aún más la velocidad de saturación hasta 6.0×10^7 cm/s. En escala lineal, la gráfica muestra que la curva $p3$ se ajusta correctamente a los resultados experimentales, incluso mejor que la curva del simulador MC. Se observa que al ir disminuyendo la velocidad de saturación se obtienen valores cada vez más bajos de la corriente de drenador, por lo tanto las curvas $p2$ y sobre todo $3D$ *D-D* se alejan progresivamente del valor experimental. En escala logarítmica tanto la curva $p2$ como la $p3$ se ajustan

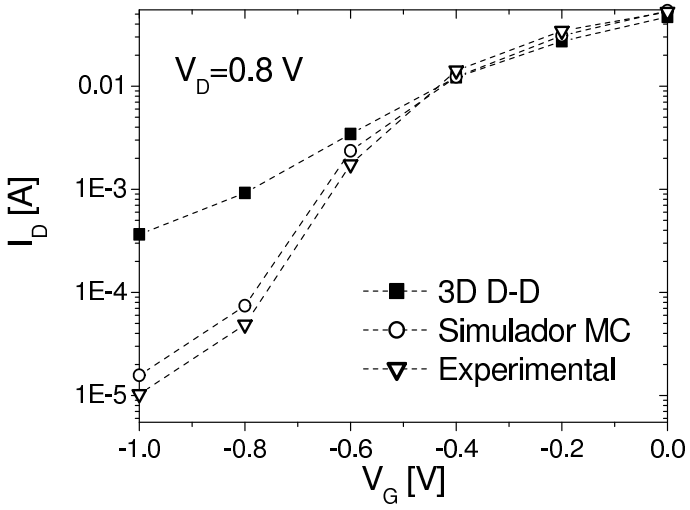


Figura 5.9: Curvas características, en escala logarítmica, a una tensión de drenador de 0.8 V para el dispositivo HEMT de 50 nm de longitud de puerta.

más a la experimental que la curva $3D$ *D-D*.

5.4. Fluctuaciones de parámetros intrínsecos en dispositivos HEMT

La influencia de las fluctuaciones de parámetros intrínsecos en dispositivos MOSFET está ampliamente estudiada y documentada. Sin embargo, su importancia es igualmente notable en otros dispositivos menos comerciales y que por lo tanto han sido menos analizados. Este es el caso de los dispositivos HEMT y PHEMT. A continuación se describen algunas de las más importantes fuentes de fluctuaciones en los dispositivos MOSFET, teniendo en cuenta que también pueden aparecer en otros dispositivos.

- Naturaleza discreta de los dopantes. Las fluctuaciones de parámetros intrínsecos al tener en cuenta la naturaleza discreta de los dopantes en el canal, la fuente y el drenador de los dispositivos MOSFET se predijo al inicio de los años 70 [121] y han sido confirmadas en varios estudios experimentales [122, 123] y en simulaciones [124, 125]. Considerar la naturaleza discreta de los dopantes en el canal, y en otras zonas del dispositivo, provoca fluctuaciones en la tensión umbral y en la corriente de los dispositivos [3, 115, 116, 126].

- Carga atrapada. En MOSFETs con longitudes de puerta del orden de

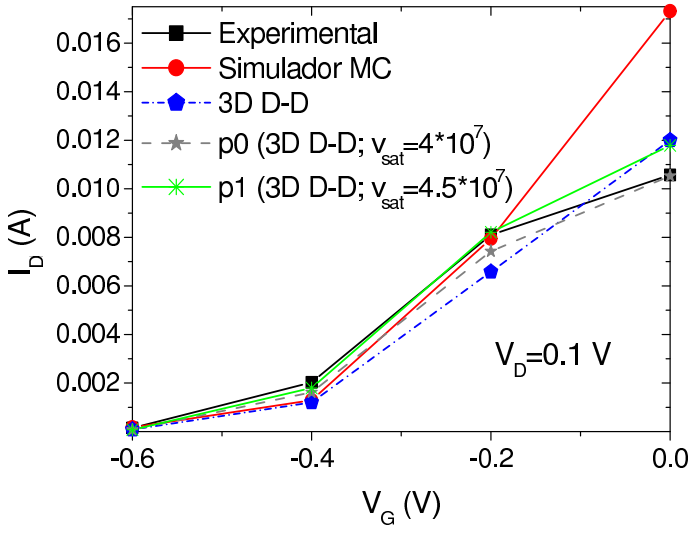


Figura 5.10: Curvas características, en escala lineal, a una tensión de drenador de 0.1 V para el dispositivo HEMT de 50 nm de longitud de puerta, entre las que se incluyen datos experimentales, datos de un simulador MC, y datos del simulador 3D paralelo: iniciales y de dos nuevas calibraciones, $p0$ y $p1$.

las decenas de nm, la aparición de una carga atrapada en las proximidades de la frontera Si/SiO_2 provoca una modulación en la densidad de portadores y en la movilidad [127] en un área comparable a las dimensiones características del dispositivo. La influencia de este fenómeno es muy importante en las corrientes de drenador y puerta de los dispositivos [128]. Las fluctuaciones en la corriente a estas escalas se convierten en fuentes de un excesivo ruido a baja frecuencia tanto en circuitos analógicos y mixtos [129] como en memorias dinámicas [130] y posiblemente en aplicaciones digitales.

- Fluctuaciones en el grosor del óxido. Se están fabricando industrialmente dispositivos en los que el grosor del óxido de puerta ha alcanzado casi los 1.5 nm [131] mientras que en los dispositivos de investigación el grosor es inferior a 1 nm. Con el escalado de los dispositivos el grosor del óxido de puerta comienza a ser equivalente a varias capas atómicas con una rugosidad de interfaz típica del orden de 1 o 2 capas atómicas [132], lo que provocará variaciones en el grosor de óxido de puerta entre dispositivos superiores al 50 %, haciendo cada dispositivo microscópicamente diferente de otro. Esto provocará fluctuaciones

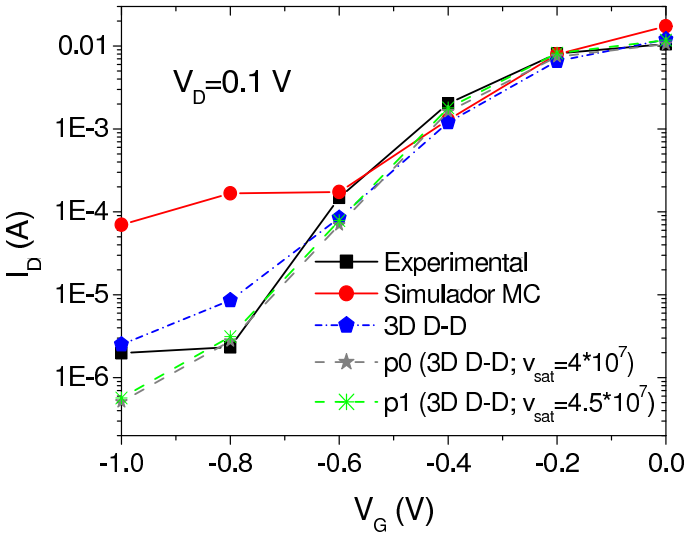


Figura 5.11: Curvas características, en escala logarítmica, a una tensión de drenador de 0.1 V para el dispositivo HEMT de 50 nm. Se incluyen datos experimentales, datos de un simulador MC, y datos del simulador 3D paralelo: iniciales y de dos nuevas calibraciones, $p0$ y $p1$.

en la movilidad, corriente túnel a través de la puerta o en la tensión umbral [133].

- Rugosidad de los extremos de la puerta (*Line Edge Roughness*). Las rugosidades en los extremos de la puerta aparecen de forma inherente al proceso litográfico. Este fenómeno no era considerado en el pasado ya que las dimensiones críticas de los dispositivos eran varios órdenes de magnitud superiores a las rugosidades generadas. Sin embargo, con el drástico escalado de los dispositivos las rugosidades en los extremos de la puerta no escalan de una forma proporcional, convirtiéndose cada vez en una fracción más grande de la longitud de puerta, lo que provoca variaciones en la tensión umbral y en las corrientes en conducción y corte de los dispositivos [132, 134].

Al igual que en el caso de los dispositivos MOSFET, la progresiva evolución del escalado en los dispositivos HEMT provoca que los efectos debidos a las fluctuaciones de parámetros intrínsecos sean cada vez más importantes, pudiendo afectar seriamente al comportamiento de los dispositivos.

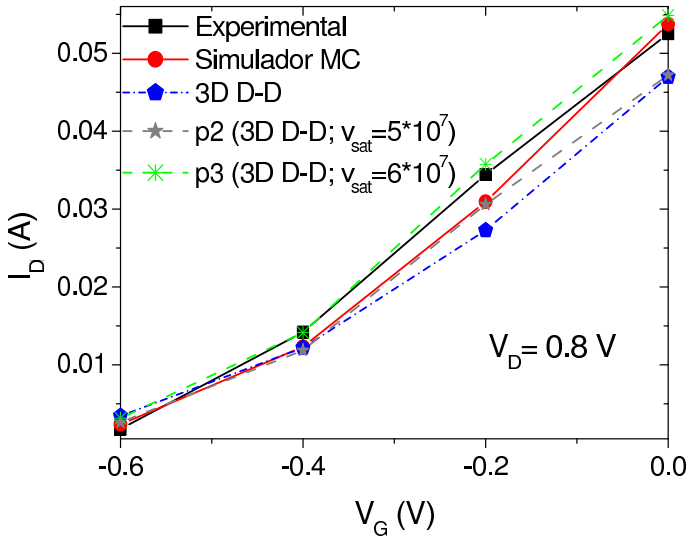


Figura 5.12: Curvas características, en escala lineal, a una tensión de drenador de 0.8 V para el dispositivo HEMT de 50 nm de longitud de puerta, entre las que se incluyen datos experimentales, datos de un simulador MC, y datos del simulador 3D paralelo: iniciales y de dos nuevas calibraciones, $p2$ y $p3$.

5.4.1. Fuentes de fluctuaciones de parámetros intrínsecos

Las principales fuentes de fluctuaciones de parámetros intrínsecos que hacen a los dispositivos HEMT microscópicamte diferentes entre sí son la consideración de la naturaleza discreta de los dopantes y las variaciones de la composición de los materiales.

La primera fuente de fluctuaciones considerada en este estudio es la naturaleza discreta de los dopantes situados en la capa δ–*doping*. En el resto del dominio de simulación se supone que la carga tiene una distribución continua. El número de dopantes esperado en la capa δ–*doping* se estima como el producto de la concentración de hoja en la capa por el área del HEMT simulado. El número real de dopantes en un dispositivo simulado concreto es escogido aleatoriamente a partir de una distribución de Poisson con media el número de dopantes estimado. A continuación se aplica una "técnica de rechazo" para situar los dopantes en los nodos de la malla cristalina.

Un ejemplo de la concentración del dopado generada aleatoriamente en la capa δ–*doping* del dispositivo HEMT de 50 nm de longitud de puerta se muestra en la figura 5.14, en la que las posiciones de los dopantes se indican mediante nodos blancos.

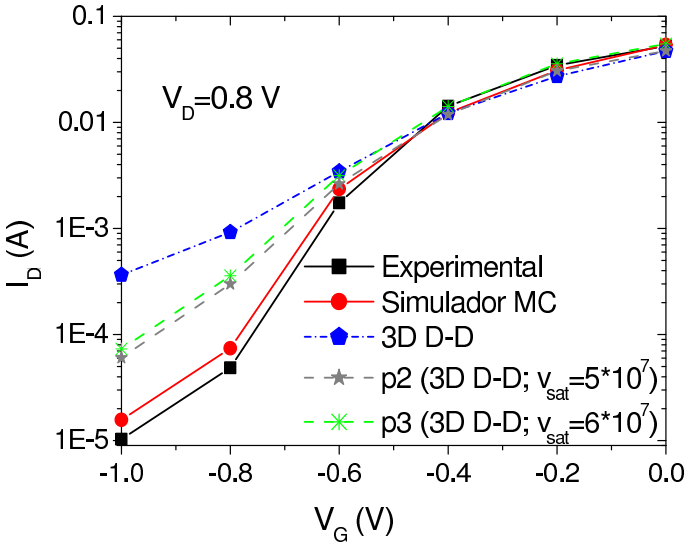


Figura 5.13: Curvas características, en escala logarítmica, a una tensión de drenador de 0.8 V para el dispositivo HEMT de 50 nm. Se incluyen datos experimentales, datos de un simulador MC, y datos del simulador 3D paralelo: iniciales y de dos nuevas calibraciones, $p2$ y $p3$.

La segunda fuente de fluctuaciones considerada en este trabajo es la variación en el contenido en Indio en el canal del dispositivo. Para generar de forma realista las fluctuaciones en la composición del canal se utiliza una red cristalina tridimensional con las mismas dimensiones a las del canal. Las posiciones de los átomos de In, Ga, As en la red cristalina son generadas aleatoriamente de acuerdo con la composición del canal. Cada nodo de la red cristalina es a continuación asignado al nodo más próximo de la malla tetraédrica, creando una distribución aleatoria en el contenido en Indio dentro del canal. El contenido en Indio obtenido utilizando esta aproximación varía entre 0.1 y 0.3 en el dispositivo de 120 nm de longitud de puerta con un canal de $In_{0.2}Ga_{0.8}As$. Para la capa de $In_{0.7}Ga_{0.3}As$ correspondiente al dispositivo HEMT de 50 nm de longitud de puerta la variación en el contenido en Indio está entre 0.6 y 0.8. Así, la figura 5.15 ilustra un ejemplo de la variación en la concentración de Indio en el interior del canal del PHEMT de 120 nm. Las grandes variaciones en el contenido en Indio que se encuentran en la figura bajo la puerta y las zonas de *recess* del dispositivo están asociadas con el reducido tamaño de los elementos de las mallas en esas regiones. En el resto de las zonas, el tamaño de los elementos que forman la malla es más grande, con lo cual a cada elemento le corresponde un número conside-

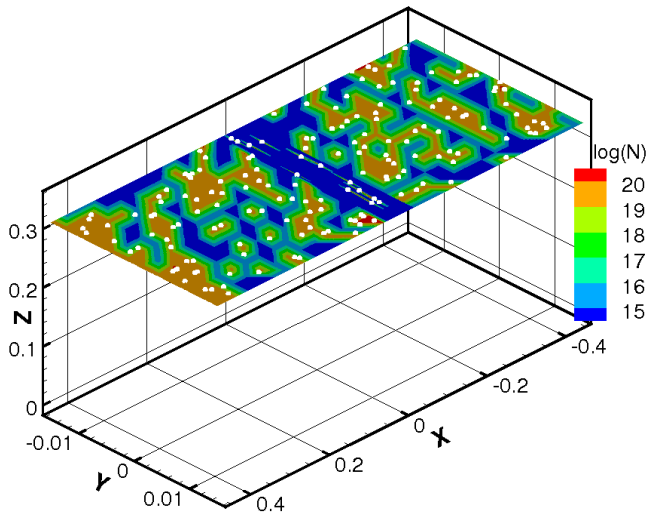


Figura 5.14: Ejemplo de una sección en el interior de la capa δ–*doping* perteneciente al dispositivo HEMT de longitud de puerta 50 nm. La posición de los dopantes se indican mediante círculos a lo largo del plano.

rable de dopantes. Teniendo en cuenta que en la figura se muestran valores promedio por elemento de la malla de la variación de la concentración de Indio, en estas regiones el efecto creado por las fluctuaciones está suavizado por la suma de todas las contribuciones. En cambio, debajo de la puerta y de la zona de *recess* el menor tamaño de los elementos provoca que a cada uno de ellos le corresponda un número mucho menor de dopantes, con lo cual al promediar en cada elemento, las variaciones que se observan son más bruscas, apreciándose a causa de ello picos en el valor de la concentración.

El contenido en Indio, asignado a cada nodo de la malla tetraédrica, se utiliza para actualizar diversos parámetros físicos como la movilidad, la constante dieléctrica y la energía de la banda prohibida. Por ejemplo, la figura 5.16 ilustra un ejemplo de la variación en la movilidad provocada por las fluctuaciones en el contenido en Indio en el interior del canal del PHEMT de 120 nm. Teniendo en cuenta que x es la fracción molar final de In de cada nodo de la malla tetraédrica, en la actualización de las variables mencionadas anteriormente se utilizan las siguientes expresiones:

- Energía de la banda prohibida:

$$E_{gap} = 1.43 - 1.543x + 0.48x^2 \tag{5.26}$$

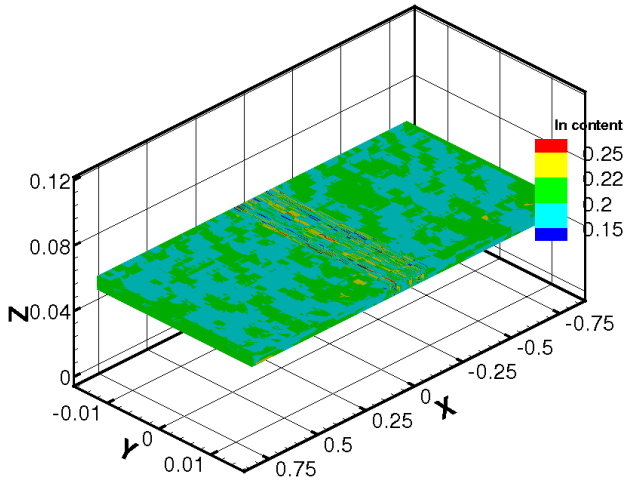


Figura 5.15: Ejemplo de las fluctuaciones en el contenido en Indio en el interior del canal de $In_{0.2}Ga_{0.8}As$ del dispositivo PHEMT de 120 nm de longitud de puerta.

- Constante dieléctrica:

$$\varepsilon = 12.65 - 1.73x + 3.90x^2 \tag{5.27}$$

- Movilidad electrónica:

$$\mu_n = \mu_n(x_{unif})\frac{(m^*(x_{unif}))^2}{(m^*(x))^2} \tag{5.28}$$

  siendo x_{unif} la fracción molar inicial de In de cada nodo de la malla, previamente al cálculo de las fluctuaciones, y $m^*(x_{unif})$ y $m^*(x)$ son, respectivamente, las masas efectivas electrónicas en el caso uniforme y una vez calculadas las fluctuaciones. Para el cálculo de las masas efectivas se utiliza la siguiente expresión:

$$m^*(x) = 0.066(1.0 - x) + 0.023x \tag{5.29}$$

Además de estas dos fuentes de fluctuaciones también se ha estudiado la influencia de una tercera fuente, la variación aleatoria de la carga superficial en la interfaz de la zonas *recess* de los dispositivos. Esta fuente de fluctuaciones tiene su origen en el proceso de fabricación. El proceso de cálculo del

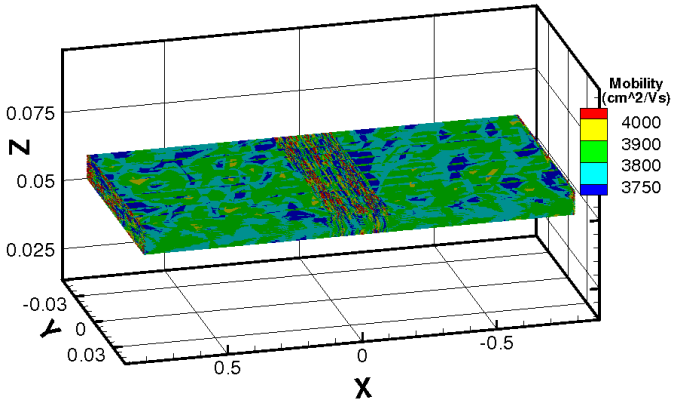


Figura 5.16: Ejemplo de las fluctuaciones en la movilidad creadas por la variación en el contenido en Indio en el interior del canal de $In_{0.2}Ga_{0.8}As$ del dispositivo PHEMT de 120 nm de longitud de puerta.

impacto de este efecto es análogo al empleado en el caso de las cargas dopantes en la capa δ–*doping*, descrito previamente. La figura 5.17 representa el potencial en el plano de las zonas de *recess*, considerando la presencia de fluctuaciones debidas a la carga interfacial en estas zonas.

5.5. Resultados numéricos

En esta sección, inicialmente se presenta un análisis de la influencia en las curvas características de los dos dispositivos HEMT estudiados de la presencia de carga interfacial en las zonas de *recess* de los dispositivos.

A continuación, se muestran algunos de los resultados numéricos obtenidos a lo largo del estudio de las tres fuentes de fluctuaciones de parámetros intrínsecos descritas previamente. Para ello se divide el análisis en tres partes, en primer lugar se estudian las fluctuaciones en las curvas características para el dispositivo PHEMT de 120 nm, a continuación se realiza el mismo proceso para el dispositivo HEMT de 50 nm, para finalizar mostrando un análisis del impacto de las fluctuaciones en la frecuencia de corte de ambos dispositivos.

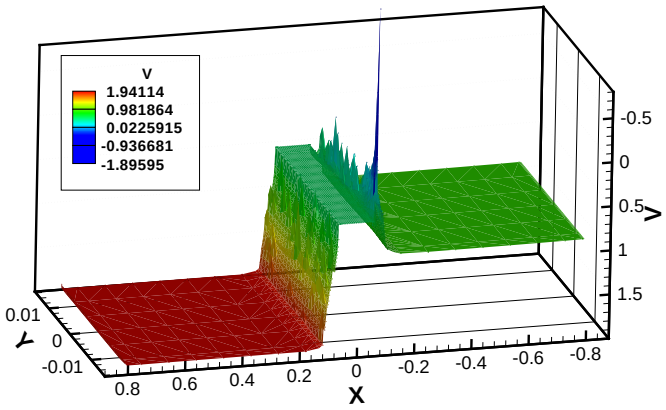


Figura 5.17: Distribución del potencial en el plano que contiene a las zonas de *recess* del dispositivo PHEMT de 120 nm considerando únicamente fluctuaciones de la carga interfacial en estas zonas.

5.5.1. Efecto de la presencia de carga interfacial en las curvas características de los dispositivos

Se ha calculado la influencia de la presencia de carga interfacial en las regiones *recess* de los dispositivos, situadas entre la puerta y las regiones de fuente y drenador.

Las figuras 5.18 y 5.19 comparan las curvas características, para V_D=0.1 y V_D=1.0 V respectivamente, obtenidas con el simulador tridimensional para el PHEMT de 120 nm bajo dos situaciones: sin considerar la presencia de carga interfacial y considerando un valor uniforme de carga de -2.0×10^{-12} cm^{-2}. La presencia de carga interfacial en estas zonas reduce la corriente de drenador, siendo la reducción más considerable a una tensión de drenador de 0.1 V que a la tensión de drenador de 1.0 V . La figura 5.20 muestra la diferencia entre las densidades electrónicas en equilibrio, para el dispositivo PHEMT de 120 nm, con carga interfacial y sin ella. La densidad electrónica está representada a lo largo de un plano en el medio del canal en el que se asume una densidad uniforme de carga interfacial en la superficie de las regiones *recess* del dispositivo.

En el caso del dispositivo HEMT de 50 nm se ha realizado el mismo

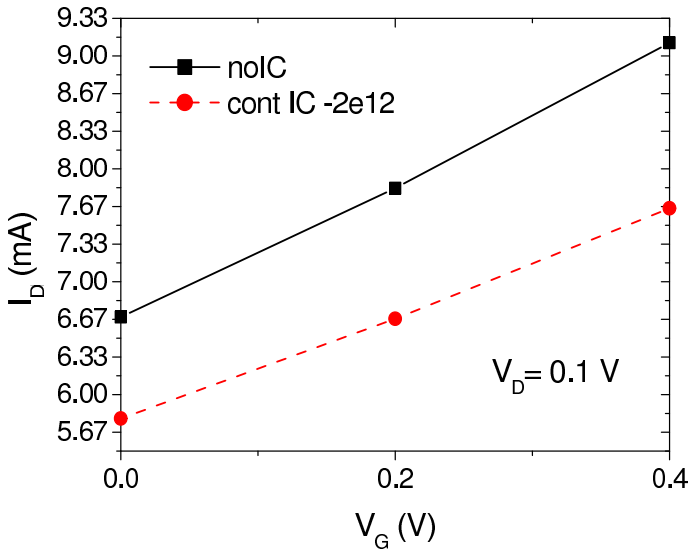


Figura 5.18: Comparación de las curvas características, I_D–V_G, obtenidas con el simulador tridimensional sin considerar y considerando la presencia de cargas en la interfaz de las regiones *recess* del PHEMT de 120 nm. Estas curvas se obtuvieron para un V_D=0.1 V.

estudio, cuyos resultados están representados en las figuras 5.21 y 5.22. El efecto de la carga interfacial es menor que el obtenido con el dispositivo de 120 nm, aunque el comportamiento cualitativo es similar.

5.5.2. Efecto de las fluctuaciones de parámetros intrínsecos en el PHEMT de 120 nm de longitud de puerta

El impacto de las dos primeras fuentes de fluctuaciones analizadas, la consideración de la naturaleza discreta de los dopantes situados en el interior de la capa *δ–doping* y las variaciones en el contenido en Indio en el interior del canal, han sido estudiadas de forma separada y en conjunto, en transistores con un ancho de canal de 30 nm.

La dependencia de la desviación estándar normalizada de la corriente de drenador, $\sigma I_D/I_D$, como una función de la tensión de puerta a tensiones de drenador de 0.1 V y 1.0 V para el 120 nm PHEMT se muestra en las figuras 5.23 y 5.24 respectivamente. La suma estadística de las dos fuentes de fluctuaciones también se representa en esas figuras. Esta suma viene dada por la siguiente expresión:

$$\sigma_{\text{indep}} = \sqrt{\sigma^2_{\text{delta}} + \sigma^2_{\text{channel}}} \tag{5.30}$$

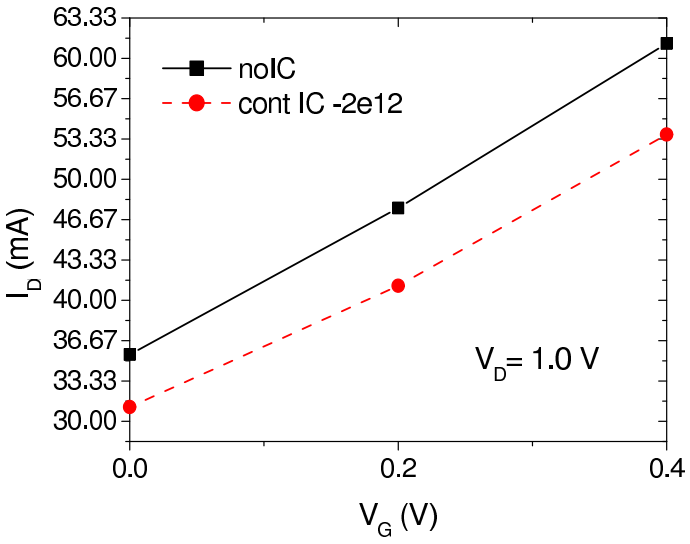


Figura 5.19: Comparación de las curvas características obtenidas con el simulador tridimensional sin considerar y considerando la presencia de cargas en la interfaz de las regiones *recess* del PHEMT de 120 nm. Estas curvas se obtuvieron para un V_D=1.0 V.

siendo σ_{delta} y σ_{channel} las desviaciones estándar normalizadas debidas a variaciones en la capa δ*–doping* y en el contenido en Indio, respectivamente.

La presencia de dopantes en el interior de la capa δ*–doping* es la principal fuente de fluctuaciones en la corriente del PHEMT. La variación en el contenido en Indio en el canal juega un papel menos importante, dando lugar a desviaciones estándar normalizadas de la corriente menores que el 30 % de las provocadas por los dopantes en el δ*–doping*.

En el caso de las fluctuaciones creadas por tener en cuenta la naturaleza discreta de los dopantes presentes en la capa δ*–doping* se observa una pronunciada disminución en la variación de la corriente al aumentar la tensión de puerta aplicada. Esto es debido al efecto pantalla creado por la presencia de los dopantes, que inducen fluctuaciones en el potencial por medio de portadores libres en el canal, cuya densidad aumenta con la tensión de puerta. Sin embargo, las fluctuaciones provocadas por las variaciones en la composición del In en el canal son más importantes al aumentar la tensión de puerta. Para las dos fuentes de fluctuaciones estudiadas, la desviación estándar normalizada de la corriente de drenador se incrementa con la tensión de drenador, puesto que al aumentar el voltaje de drenador, aumenta el campo eléctrico, provocando variaciones más pronunciadas en la movilidad

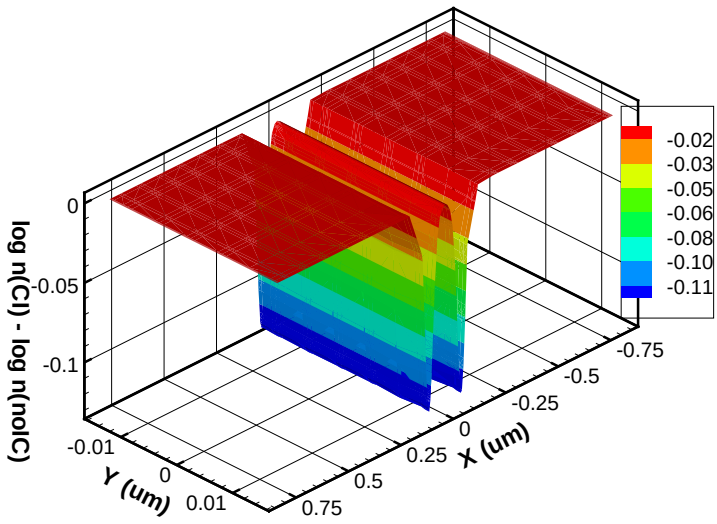


Figura 5.20: Diferencia en la densidad electrónica en equilibrio entre un dispositivo con carga de interfaz de -2.0×10^{-12} cm^{-2} en las regiones *recess* y otro sin carga de interfaz. La medida está realizada en un plano a lo largo del canal de $In_{0.2}Ga_{0.8}As$ del dispositivo PHEMT de 120 nm de longitud de puerta.

que son la principal contribución a la variación de la corriente de drenador. Un estudio más detallado sobre la influencia de las fluctuaciones en la composición del canal en las características del dispositivo se puede encontrar en [135].

Al considerar ambas fuentes de fluctuaciones en conjunto se observa que la desviación estándar normalizada disminuye al aumentar la tensión de puerta, lo que es completamente lógico, ya que, como se comentó anteriormente, considerar la naturaleza discreta de los dopantes en la capa δ*–doping* supone una mucho mayor influencia en las fluctuaciones que las variaciones en el Indio. Además, se observa una mayor variación en la corriente al simular las dos fuentes de fluctuaciones conjuntamente que la obtenida con la suma estadística de las dos fuentes de fluctuaciones por separado.

Para este dispositivo, también ha sido analizada la variación de la desviación estándar normalizada con la corriente de drenador dependiendo del ancho del canal del dispositivo. Para ello, se han simulado dispositivos con anchos de canal de 30, 60, 90 y 120 nm. La desviación estándar normalizada

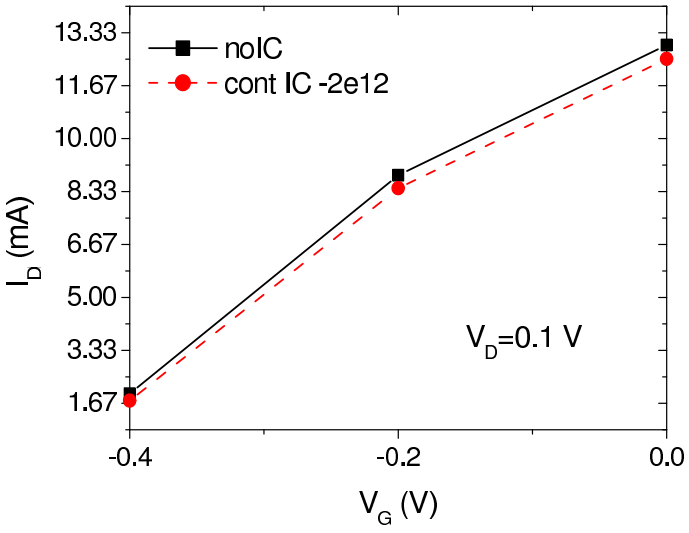


Figura 5.21: Comparación de las curvas características, I_D–V_G, obtenidas con el simulador tridimensional sin considerar y considerando la presencia de cargas en la interfaz de las regiones *recess* del HEMT de 50 nm. Estas curvas se obtuvieron para un V_D=0.1 V.

de la corriente de drenador como función del ancho del canal a baja y alta tensión de drenador se ilustra en la figura 5.25. Se observa un descenso no–lineal en la desviación estándar normalizada de la corriente de drenador al aumentar el ancho del dispositivo [132].

$V_D(V)$	$V_G(V)$	$I_{D,(0.2,\text{unif})}(mA)$	$I_{D,media}(mA)$	$\sigma I_D/I_D$	*kurtosis*	*C.A.*
0.1V	0.0	6.638	6.597	0.035	-0.177	-0.018
	0.2	7.799	7.858	0.036	0.269	-0.181
	0.4	9.075	9.264	0.039	0.627	-0.297
1.0V	0.0	31.387	30.186	0.045	-0.550	0.207
	0.2	45.488	44.624	0.046	-0.841	-0.107
	0.4	59.975	60.822	0.061	6.599	-1.934

Tabla 5.3: Parámetros estadísticos que caracterizan las distribuciones relativas a las variaciones aleatorias en el contenido de In en el interior del canal del PHEMT de 120 nm a $V_G = 0.0, 0.2$ y 0.4 V.

La figura 5.26 muestra un ejemplo, en escala semilogarítmica, de la concentración de electrones en el equilibrio teniendo en cuenta fluctuaciones en la composición del Indio del canal del dispositivo. Por otro lado, en la figura 5.27 se representa el potencial electrostático obtenido, en la malla 3D, en

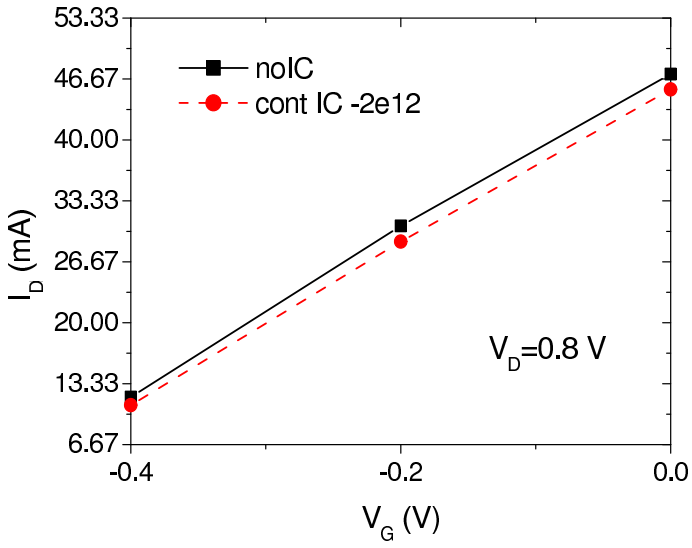


Figura 5.22: Comparación de las curvas características obtenidas con el simulador tridimensional sin considerar y considerando la presencia de cargas en la interfaz de las regiones *recess* del HEMT de 50 nm. Estas curvas se obtuvieron para un V_D=0.8 V.

$V_D(V)$	$V_G(V)$	$I_{D,(0.2,\mathrm{unif})}(mA)$	$I_{D,media}(mA)$	$\sigma I_D/I_D$	*kurtosis*	*C.A.*
0.1V	0.0	6.638	5.922	0.121	0.777	0.400
	0.2	7.799	7.094	0.109	1.063	0.686
	0.4	9.075	8.178	0.108	1.114	0.762
1.0V	0.0	31.387	28.213	0.156	-0.316	0.181
	0.2	45.488	41.666	0.137	-0.004	0.400
	0.4	59.975	54.729	0.127	0.064	0.482

Tabla 5.4: Parámetros estadísticos que caracterizan las distribuciones aleatorias de dopantes en el interior de la capa δ–*doping* a $V_G = 0.0, 0.2$ y 0.4 V para el PHEMT de 120 nm.

las mismas condiciones que las representadas en la figura anterior.

Los parámetros que caracterizan la forma de las distribuciones estadísticas para el dispositivo PHEMT de 120 nm de longitud de puerta, incluyendo el valor medio, la desviación estándar normalizada, la kurtosis y el coeficiente de asimetría (C.A.) de la distribución de corriente de drenador, son mostrados para las variaciones en el contenido en Indio, las fluctuaciones en la carga en la capa δ–*doping*, la combinación de ambas fuentes de fluctuaciones y para las variaciones con el ancho del dispositivo, en las tablas 5.3, 5.4,

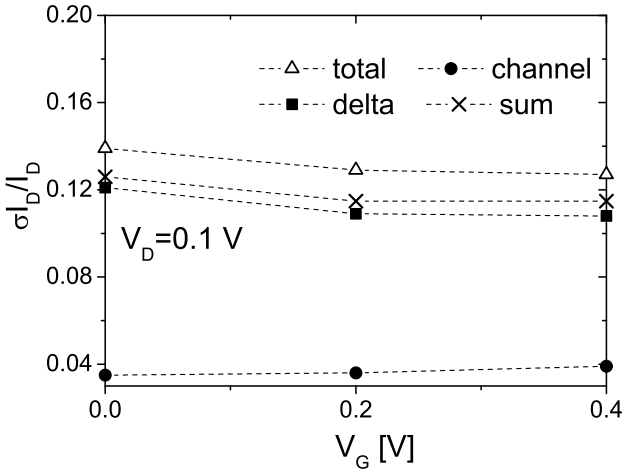


Figura 5.23: Desviación estándar normalizada de la corriente de drenador en función de la tensión de puerta calculada teniendo en cuenta la presencia de dopantes en la capa δ–*doping* (delta), variaciones en el contenido en Indio del canal (channel) o ambos efectos a la vez (total) a $V_{\mathrm{D}} = 0.1$ V para el PHEMT de 120 nm. La suma estadística de las dos fuentes de fluctuaciones también está incluida (sum).

$\mathrm{V_D}(V)$	$\mathrm{V_G}(V)$	$I_{D,(0.2,\mathrm{unif})}(mA)$	$I_{D,media}(mA)$	$\sigma I_D/I_D$	*kurtosis*	*C.A.*
0.1V	0.0	6.638	5.970	0.139	0.016	0.499
	0.2	7.799	7.300	0.129	0.937	0.843
	0.4	9.075	8.572	0.127	1.215	0.910
1.0V	0.0	31.387	27.275	0.183	-0.534	0.364
	0.2	45.488	41.102	0.162	-0.043	0.590
	0.4	59.975	55.622	0.148	0.262	0.725

Tabla 5.5: Parámetros estadísticos que caracterizan las fluctuaciones de parámetros intrínsecos debidas tanto a la presencia de cargas dopantes en la capa δ–*doping* como a variaciones en la composición del canal para el PHEMT de 120 nm a $\mathrm{V_G} = 0.0, 0.2$ y 0.4 V.

5.5 y 5.6, respectivamente. Los parámetros estadísticos han sido extraídos de un conjunto de 40 configuraciones diferentes del dispositivo. En las tablas también se muestran las corrientes de drenador obtenidas usando una distribución continua en la capa δ–*doping* y un contenido uniforme de Indio

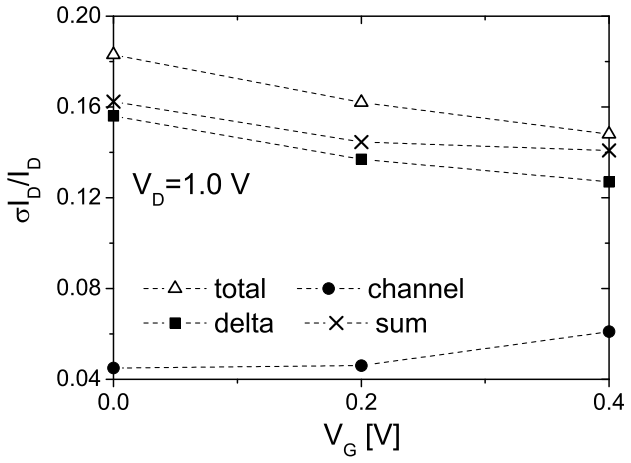


Figura 5.24: Desviación estándar normalizada de la corriente de drenador en función de la tensión de puerta calculada a $V_{\mathrm{D}} = 1.0$ V para el PHEMT de 120 nm.

en el canal.

La presencia de fluctuaciones lleva a un descenso en la corriente de drenador promedio tanto para las dos fuentes de fluctuaciones por separado como en combinación, aunque al considerar sólo las fluctuaciones debidas

V_{D}	*Ancho*(nm)	$I_{D_{unif}}(mA)$	$I_{D,media}(mA)$	$\sigma I_D/I_D$	*kurtosis*	*C.A.*
0.1V	30	6.636	5.795	0.133	2.172	0.735
	60	6.624	5.788	0.123	-0.550	0.575
	90	6.608	5.736	0.116	-0.796	0.260
	120	6.595	5.517	0.101	0.079	0.334
1.0V	30	31.419	28.024	0.170	-0.517	0.274
	60	31.384	27.833	0.159	-0.808	0.404
	90	31.205	27.267	0.146	-0.469	0.421
	120	29.901	26.516	0.124	-0.282	0.175

Tabla 5.6: Parámetros estadísticos que caracterizan las fluctuaciones de parámetros intrínsecos debidas tanto a la presencia de cargas dopantes en el *δ–doping* como a variaciones en la composición del canal para el PHEMT de 120 nm de longitud de puerta, para anchos del dispositivo de 30, 60, 90 y 120 nm.

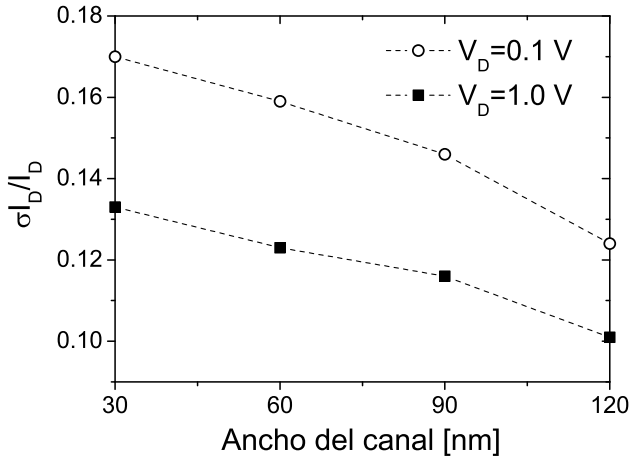


Figura 5.25: Desviación estándar normalizada de la corriente de drenador en función del ancho del PHEMT de 120 nm, calculada a $V_{\mathrm{D}} = 0.1$ y 1.0 V.

$\mathrm{V_D}(v)$	$\mathrm{V_G}(v)$	$I_{D_{noIC}}(mA)$	$I_{D_{unif}}(mA)$	$I_{D_{med}}(mA)$	$\frac{\sigma I_D}{I_D}$	$Kurt$	$C.A.$
0.1	0.0	6.687	5.788	5.782	0.016	0.212	-0.070
	0.2	7.828	6.671	6.665	0.018	0.234	-0.030
	0.4	9.117	7.650	7.648	0.021	0.233	-0.035
1.0	0.0	35.517	31.177	31.097	0.020	1.019	-0.542
	0.2	47.626	41.202	41.442	0.023	0.334	-0.316
	0.4	61.227	53.704	53.550	0.025	0.579	-0.467

Tabla 5.7: Parámetros estadísticos que caracterizan la naturaleza de las distribuciones obtenidas para el PHEMT de 120 nm debidas a la presencia de carga interfacial en las zonas de *recess* del dispositivo.

a variaciones en el contenido de In en el canal se encuentran excepciones a esta afirmación, cuando la tensión de drenador es de 0.1 V y la tensión de puerta es superior a 0.0 V, o bien para una tensión de drenador de 1.0 V y un valor de $V_G = 0.4$ V. En estas situaciones se obtiene una corriente de drenador promedio superior a la lograda en el caso uniforme, en el que no se considera ninguna fuente de fluctuaciones. El descenso en la corriente de drenador promedio, a causa de las dos fuentes de fluctuaciones analizadas, es debido a un filtrado de la corriente a través de los valles en la fluctuación del potencial en el canal. La distribución estadística está próxima a la distribución normal pero para una precisa afirmación de esta hipótesis

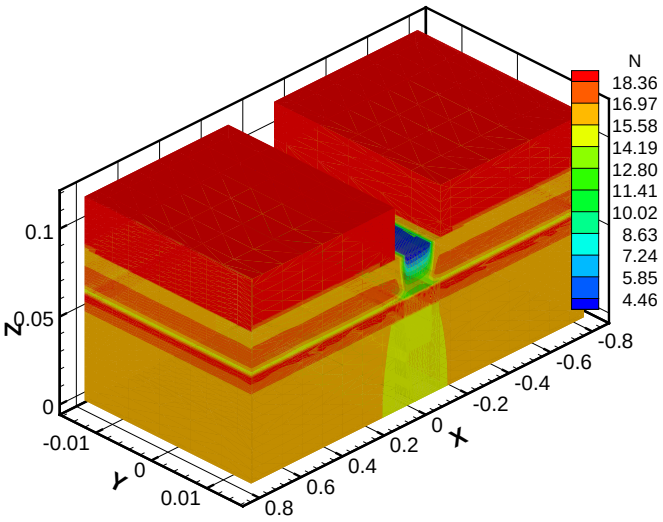


Figura 5.26: Concentración de electrones en el equilibrio considerando la presencia de fluctuaciones debidas a variaciones en la composición del canal del dispositivo PHEMT de 120 nm de longitud de puerta.

se necesitaría un mayor número de muestras estadísticas. Estudiando los parámetros estadísticos obtenidos al considerar conjuntamente las dos fuentes de fluctuaciones (ver tabla 5.5) se observa que el coeficiente de asimetría siempre toma valores positivos, aumentando su valor con la tensión de puerta aplicada y disminuyendo al incrementar la tensión de drenador. Los valores obtenidos llevan a la conclusión de que existe, en algunos casos, una muy pequeña asimetría que provoca que los valores extremos de la distribución se encuentren desplazados hacia la derecha de la media. Con respecto a la kurtosis, se encuentra que la mayoría de los valores obtenidos son positivos, exceptuando para $V_G = 0.0$ y 0.2 V cuando la tensión de drenador es de 1.0 V.

Además de las dos fuentes de fluctuaciones analizadas hasta ahora, es necesario estudiar también el impacto de la presencia de carga interfacial aleatoria en las regiones *recess* del dispositivo. La tabla 5.7 muestra los parámetros estadísticos que caracterizan la naturaleza de las distribuciones obtenidas para este dispositivo, incluyendo el valor medio de la corriente de drenador ($I_{D_{med}}$) y su desviación estándar normalizada ($\frac{\sigma I_D}{I_D}$), kurtosis ($Kurt$) y coeficiente de asimetría ($C.A.$). Las corrientes de drenador obteni-

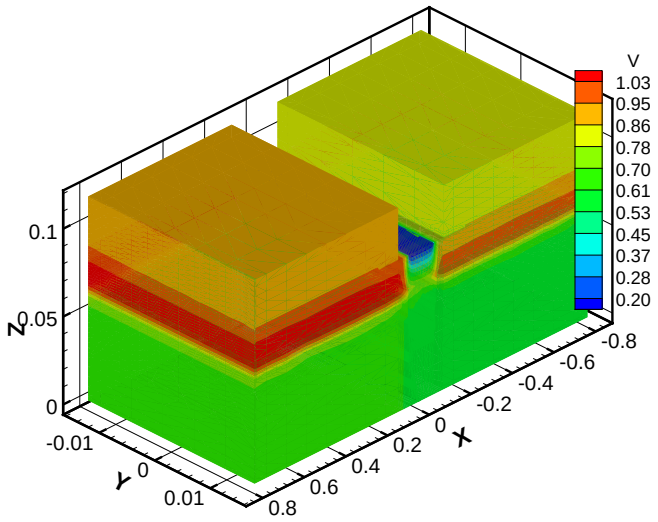


Figura 5.27: Potencial electrostático en el equilibrio considerando la presencia de fluctuaciones debidas a variaciones en el contenido en In el interior del canal del dispositivo PHEMT de 120 nm de longitud de puerta.

das sin carga interfacial ($I_{D_{noIC}}$) y utilizando una distribución uniforme de carga de en las regiones *recess* ($I_{D_{unif}}$) también se muestran en la tabla. Las distribuciones estadísticas han sido obtenidas a partir de 60 configuraciones del dispositivo microscópicamente diferentes.

En este dispositivo, a causa de las fluctuaciones en la carga interfacial presente en las regiones *recess*, la desviación estándar normalizada de la corriente de drenador a tensión de drenador 1.0 V es siempre más elevada que la desviación estándar correspondiente a la corriente de drenador a tensión de drenador 0.1 V. Además, la desviación estándar normalizada de la corriente de drenador aumenta conforme aumenta la tensión de puerta. Por otro lado, la presencia de fluctuaciones debidas a variaciones en la carga interfacial lleva a un descenso de la corriente drenador promedio con respecto al valor de la corriente obtenida a un valor uniforme de carga interfacial en las zonas de *recess*. Las distribuciones estadísticas obtenidas presentan valores negativos del coeficiente de asimetría, lo que indica que los valores extremos de estas distribuciones se encuentran situados hacia la izquierda de la media. Además, este coeficiente, en valor absoluto, aumenta con la tensión de drenador aplicada. En cambio, el valor de la kurtosis, es positivo

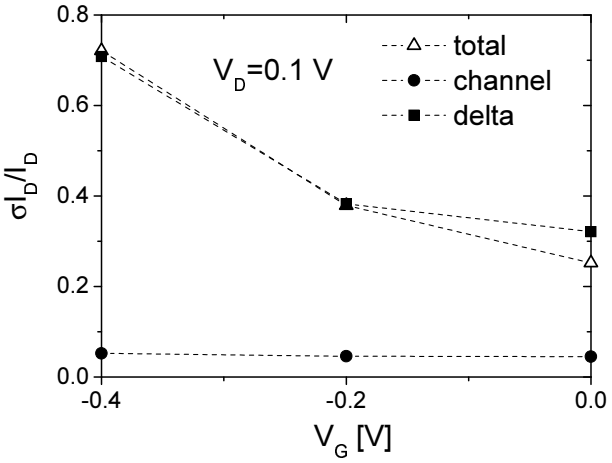


Figura 5.28: Desviación estándar normalizada de la corriente de drenador en función de la tensión de puerta calculada teniendo en cuenta la presencia de dopantes en la capa *δ–doping* (delta), variaciones en el contenido en Indio del canal (channel) o ambos efectos a la vez (total) a $V_{\mathrm{D}} = 0.1$ V para el HEMT de 50 nm.

en todos los casos analizados.

$\mathrm{V_D}(V)$	$\mathrm{V_G}(V)$	$I_{\mathrm{D},(0.7,\mathrm{cont})}(A/m)$	$I_{\mathrm{D,media}}(A/m)$	$\sigma I_{\mathrm{D}}/I_{\mathrm{D}}$	Kur	$C.A.$
0.1V	-0.4	12.023	13.069	0.052	-0.178	0.285
	-0.2	65.934	65.133	0.046	0.004	0.253
	0.0	119.623	116.896	0.045	0.000	0.439
0.8V	-0.4	121.213	107.141	0.049	-0.293	-0.117
	-0.2	269.752	235.283	0.074	13.141	2.927
	0.0	458.786	369.020	0.048	0.133	-0.511

Tabla 5.8: Parámetros estadísticos que caracterizan las distribuciones relacionadas con la variación del contenido en In en el interior del canal del HEMT de 50 nm a $V_{\mathrm{D}} = 0.1$ V y $V_{\mathrm{D}} = 0.8$ V. Estos parámetros han sido evaluados a $\mathrm{V_G} = -0.4, -0.2$ y 0.0 V.

5.5.3. Efecto de las fluctuaciones de parámetros intrínsecos en el HEMT de 50 nm de longitud de puerta

En el dispositivo HEMT de 50 nm de longitud de puerta, de igual modo que en el dispositivo de 120 nm de longitud de puerta, los dopantes en la

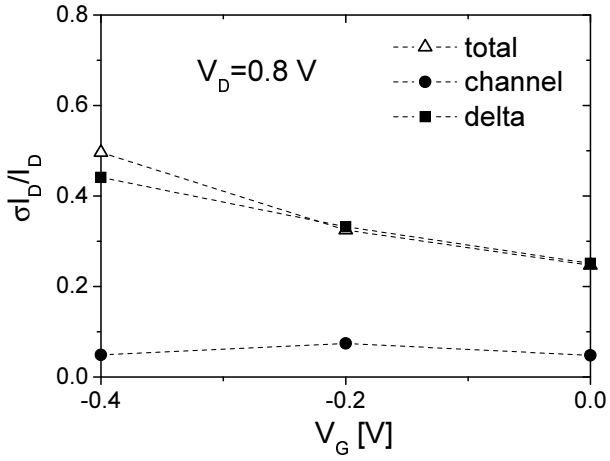


Figura 5.29: Desviación estándar normalizada de la corriente de drenador en función de la tensión de puerta calculada a $V_{\mathrm{D}} = 0.8$ V para el HEMT de 50 nm.

$\mathrm{V_D}(V)$	$\mathrm{V_G}(V)$	$I_{\mathrm{D},(0.7,\mathrm{cont})}(A/m)$	$I_{\mathrm{D,media}}(A/m)$	$\sigma I_{\mathrm{D}}/I_{\mathrm{D}}$	Kur	$C.A.$
0.1V	-0.4	12.023	18.013	0.708	-0.683	0.579
	-0.2	65.934	63.353	0.383	-1.217	-0.345
	0.0	119.623	115.044	0.321	2.511	-0.015
0.8V	-0.4	121.213	129.371	0.441	-0.402	0.316
	-0.2	269.752	268.895	0.332	-0.906	-0.068
	0.0	458.786	432.784	0.251	-0.624	-0.135

Tabla 5.9: Parámetros estadísticos que caracterizan la distribución de dopantes en el interior de la capa δ–*doping* a $\mathrm{V_G} = -0.4, -0.2$ y 0.0 V para el HEMT de 50 nm.

capa δ–*doping* y la variación en la composición del canal se simulan de forma separada y en conjunto.

La desviación estándar normalizada de la corriente de drenador frente a la tensión de puerta a tensiones de drenador de 0.1 V y de 0.8 V para el dispositivo de HEMT de 50 nm de longitud de puerta se muestra en las figuras 5.28 y 5.29 respectivamente. Las tablas 5.8, 5.9, y 5.10 presentan los parámetros que caracterizan las distribuciones estadísticas evaluadas a $\mathrm{V_G} = -0.4, -0.2$ y 0.0 V para las dos fuentes de fluctuaciones de parámetros intrínsecos actuando por separado y de forma conjunta. Para ello se utilizaron 40 configuraciones diferentes del dispositivo.

$V_D(V)$	$V_G(V)$	$I_{D,(0.7,cont)}(A/m)$	$I_{D,media}(A/m)$	$\sigma I_D/I_D$	Kur	$C.A.$
0.1V	-0.4	12.023	17.790	0.721	-0.443	0.684
	-0.2	65.934	61.019	0.379	-0.992	-0.186
	0.0	119.623	109.440	0.252	3.030	1.121
0.8V	-0.4	121.213	120.075	0.496	1.312	0.943
	-0.2	269.752	231.170	0.325	-0.853	-0.059
	0.0	458.786	352.787	0.247	-0.795	-0.150

Tabla 5.10: Parámetros estadísticos que caracterizan las fluctuaciones de parámetros intrínsecos debidas tanto a la presencia de cargas dopantes en la capa *δ–doping* como a variaciones en la composición del canal para el HEMT de 50 nm a $V_G = -0.4, -0.2$ y 0.0 V.

Las fluctuaciones inducidas debidas a la naturaleza discreta de los dopantes en la capa *δ–doping* son también en este dispositivo la fuente más dominante de fluctuaciones de parámetros intrínsecos, siendo la desviación estándar normalizada un orden de magnitud superior que la asociada con variaciones en la composición en el canal de InGaAs. Un estudio más profundo sobre la influencia de las fluctuaciones debidas a la naturaleza discreta de los dopantes en las curvas características de este dispositivo se encuentra en [96].

La desviación estándar normalizada se reduce con el aumento de la tensión de puerta tanto si se considera cada fuente de fluctuaciones por separado como si ambas fuentes de fluctuaciones son tenidas en cuenta conjuntamente, aunque esto no es cierto en todos los casos a un valor de tensión de drenador de 0.8 V si se considera sólo la presencia de fluctuaciones debidas a variaciones de composición en el canal.

Es interesante destacar que en el dispositivo de 50 nm de longitud de puerta la desviación estándar normalizada de la corriente de drenador a una tensión de drenador de 0.1 V es siempre superior que la respectiva desviación estándar en la corriente de drenador a una tensión de drenador más elevada de 0.8 V, al tener en cuenta las dos fuentes de fluctuaciones conjuntamente o al considerar sólo la presencia de dopantes en el *δ–doping*. Este comportamiento es opuesto al observado en el PHEMT de 120 nm y está relacionado con efectos de canal corto que son muy pronunciados en el dispositivo de 50 nm a un voltaje de drenador elevado. En cambio, al considerar solamente las fluctuaciones debidas a variaciones de composición en el canal observamos el comportamiento opuesto al descrito anteriormente, exceptuando para una $V_G = -0.4$. En el resto de los casos la desviación estándar normalizada de la corriente aumenta con la tensión de drenador.

$V_D(v)$	$V_G(v)$	$I_{D_{noIC}}(\frac{A}{m})$	$I_{D_{un}}(\frac{A}{m})$	$I_{D_{me}}(\frac{A}{m})$	$\frac{\sigma I_D}{I_D}$	Kur	$C.A$
0.1	-0.4	19.692	17.516	18.261	0.0167	-0.534	-0.129
	-0.2	88.492	84.391	86.083	0.0082	0.071	-0.337
	0.0	129.470	125.087	126.524	0.0064	1.318	-0.552
0.8	-0.4	118.689	109.789	113.022	0.0120	-0.496	-0.255
	-0.2	305.898	288.762	296.281	0.0079	-0.383	-0.229
	0.0	472.073	455.196	455.197	0.0067	0.132	-0.360

Tabla 5.11: Parámetros estadísticos que caracterizan la naturaleza de las distribuciones obtenidas para el HEMT de 50 nm debidas a la presencia de carga interfacial en las zonas de *recess* del dispositivo.

La presencia de fluctuaciones lleva, generalmente, a un descenso en la corriente de drenador promedio, tanto para las dos fuentes de fluctuaciones por separado como en combinación. Sin embargo, en las tres situaciones analizadas, se encuentra que, para un valor de $V_G = -0.4$ V y a un valor bajo de tensión de drenador, la corriente de drenador en el caso uniforme es menor que la corriente de drenador promedio. Además, considerando solamente las fluctuaciones creadas por considerar la naturaleza discreta de los dopantes en la capa *δ–doping*, este comportamiento también se da a alta tensión de drenador. Analizando los parámetros estadísticos que caracterizan la naturaleza de las distribuciones obtenidas, considerando conjuntamente las dos fuentes de fluctuaciones (ver tabla 5.10), se observa una pequeña asimetría en las distribuciones, siempre teniendo en cuenta que el tamaño de nuestra muestra es pequeño, lo que no nos permite realizar afirmaciones concluyentes. Además, no es posible encontrar, a diferencia de lo que ocurría con el dispositivo de 120 nm, una relación clara del coeficiente de asimetría con la tensión de puerta o drenador aplicada. Con respecto a la kurtosis, generalmente en este dispositivo se encuentran, en valor absoluto, valores más elevados que los dados por el dispositivo de 120 nm.

Al comparar los resultados obtenidos con los dos dispositivos analizados, se observa que la influencia de las fluctuaciones de parámetros intrínsecos es más pequeña en el dispositivo PHEMT de 120 nm, obteniéndose como máximo fluctuaciones en la corriente de drenador del 18 %. Sin embargo, este valor aumenta al reducirse el tamaño del dispositivo. Así, en el HEMT de 50 nm las fluctuaciones en la corriente de drenador alcanzan el 70 %, lo que puede afectar de forma considerable al comportamiento de estos dispositivos HEMT de alto rendimiento.

Con respecto a la tercera fuente de fluctuaciones analizada, la presencia de carga interfacial en las regiones *recess* del dispositivo HEMT de 50 nm,

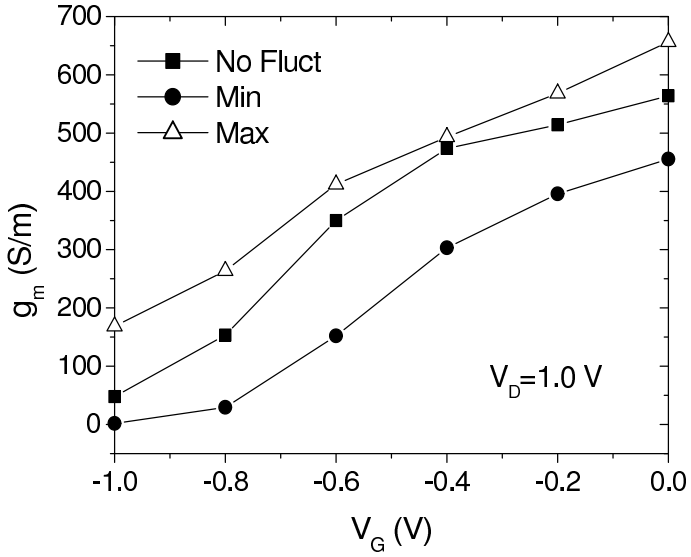


Figura 5.30: Dependencia de la transconductancia con la tensión de puerta aplicada para el dispositivo PHEMT de 120 nm a una tensión de drenador de 1.0 V. Los resultados de frecuencia se muestran para el caso continuo (*No Fluct*) y para las configuraciones de los dispositivos en las que la influencia de las fuentes de fluctuaciones en las características de los dispositivos es más (*Max*) o menos (*Min*) importante.

en la tabla 5.11 se representan los parámetros estadísticos que caracterizan la naturaleza de las distribuciones obtenidas para este dispositivo. Estas distribuciones estadísticas han sido obtenidas a partir de 60 configuraciones del dispositivo microscópicamente diferentes.

En este dispositivo, al contrario que en el PHEMT de 120 nm, considerando sólo la influencia de la presencia de carga interfacial en las zonas de *recess*, la desviación estándar normalizada de la corriente de drenador a un valor alto de tensión de drenador de 0.8 V es siempre más pequeña que la desviación estándar correspondiente a la corriente de drenador a un valor bajo de tensión de drenador de 0.1 V. Además, también de forma opuesta a lo que ocurre en el PHEMT de 120 nm, la desviación estándar normalizada de la corriente de drenador disminuye conforme aumenta la tensión de puerta. Por último, se observa que las fluctuaciones en la carga interfacial, a diferencia de las fluctuaciones creadas por las variaciones en la composición y por la consideración de la naturaleza discreta de los dopantes, provocan valores de la corriente de drenador promedio superiores a los obtenidos con un valor constante de carga interfacial.

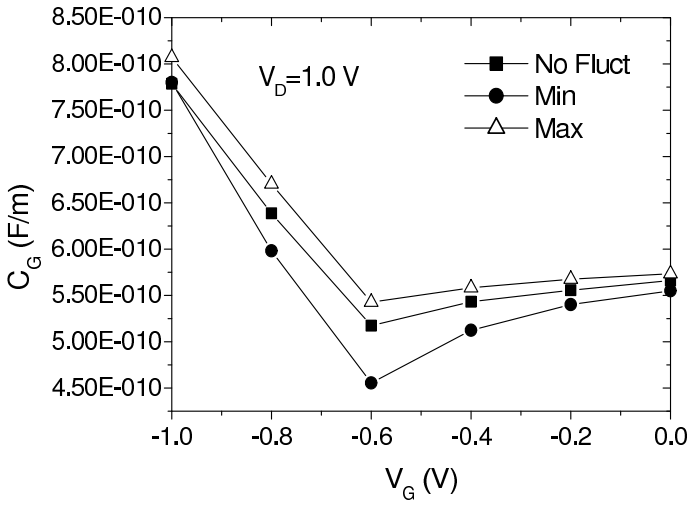


Figura 5.31: Dependencia de la capacitancia extrínseca total de la puerta con la tensión de puerta aplicada para el dispositivo PHEMT de 120 nm a una tensión de drenador de 1.0 V.

Por otro lado, el efecto de las fluctuaciones debidas a la presencia de carga de interfaz en las regiones *recess* es más importante en el PHEMT de 120 nm que en el HEMT de 50 nm. Sin embargo, en ambos dispositivos, el impacto de esta fuente de fluctuaciones en las curvas características es mucho menos importante que el efecto inducido por fluctuaciones de carga en la capa δ–*doping* y por variaciones en la composición del material ternario del canal. Un estudio más detallado sobre el impacto de las fluctuaciones de la carga interfacial en las regiones de *recess* sobre las curvas características de los dos dispositivos analizados se encuentra en [136].

5.5.4. Fluctuaciones en la frecuencia de corte

Una medida importante del factor de calidad de un dispositivo HEMT es la frecuencia de corte de la ganancia en corriente, que viene dada aproximadamente por [137]:

$$f_T \simeq \frac{g_m}{2\pi C_G} \tag{5.31}$$

siendo g_m la transconductancia y C_G la capacitancia extrínseca total de la puerta, que se puede calcular a través de la siguiente aproximación cua-

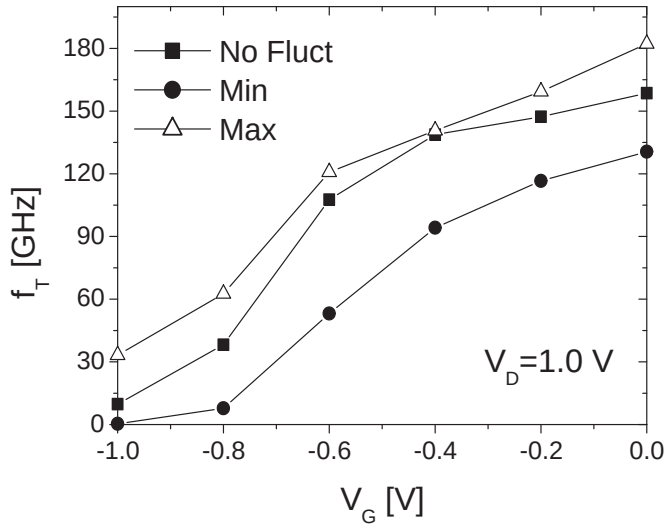


Figura 5.32: Dependencia de la frecuencia de corte con la tensión de puerta aplicada para el dispositivo PHEMT de 120 nm a un valor fijo de tensión de drenador de 1.0 V.

siestática:

$$C_G(V_G, V_D) = \left.\frac{\partial Q_G(V_G)}{\partial V_G}\right|_{V_D=const} \tag{5.32}$$

donde Q_G es la carga total en el electrodo metálico de la puerta a un valor de tensión de drenador dado.

Para el cálculo de la variación de la carga bajo la puerta se utiliza la siguiente integral de línea, dada en [138]:

$$\Delta Q_i = \oint_{gate} \varepsilon(\Delta \vec{E})\ \vec{n}\ dl \tag{5.33}$$

siendo ΔQ_i la carga incremental inducida en el contacto de puerta a causa de la perturbación de tensión, ε la permitividad del material de la puerta, $(\Delta \vec{E})$ el incremento en el campo eléctrico y $\vec{n}$ el vector normal al contorno de la puerta.

Hay que tener en cuenta que para calcular la componente del campo en cada nodo de la puerta se utiliza tan solo la componente vertical del campo eléctrico, siendo en este caso la componente z.

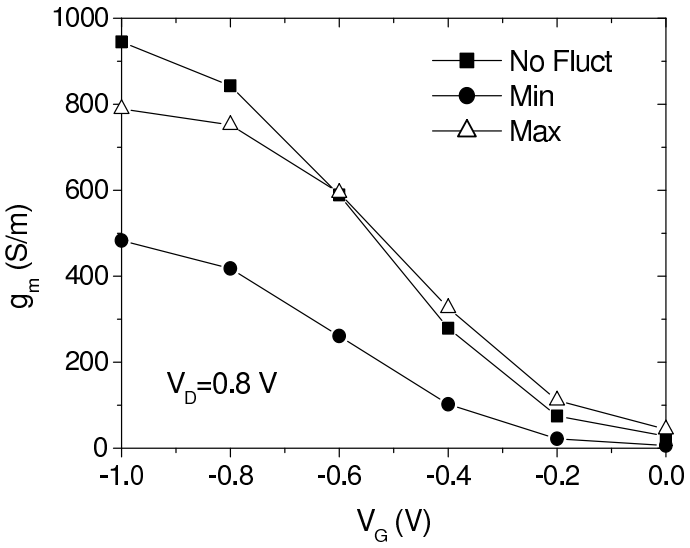


Figura 5.33: Dependencia de la transconductancia con la tensión de puerta aplicada para el dispositivo HEMT de 50 nm a un valor de tensión de drenador de 0.8 V.

También es interesante estudiar el efecto de las fluctuaciones de parámetros intrínsecos en la frecuencia de corte de los dispositivos. Para ello se han considerado conjuntamente las dos fuentes de fluctuaciones de parámetros intrínsecos que tienen una mayor influencia en la corriente de drenador de los dispositivos, es decir, se considera la naturaleza discreta de los dopantes situados en la capa δ*–doping* y las variaciones en el contenido en Indio en el interior del canal.

Las figuras 5.30, 5.31 y 5.32 muestran la dependencia de la transconductancia, g_m, de la capacitancia de la puerta, C_G, y de la frecuencia de corte, f_T, con la tensión de puerta aplicada para el PHEMT de longitud de puerta de 120 nm. Estos resultados han sido obtenidos para un valor fijo de tensión de drenador de 1.0 V.

En las figuras los resultados de frecuencia obtenidos se muestran para el caso continuo (*No Fluct*) y para las configuraciones de los dispositivos en las que el impacto de las fluctuaciones en las curvas características de los dispositivos obtienen el valor máximo (*Max*) o mínimo (*Min*) de corriente de drenador.

Como era de esperar, las fluctuaciones de parámetros intrínsecos afectan muy severamente a la frecuencia de corte de los dispositivos HEMT. Por ejemplo, a un valor de tensión de puerta 0.0 V para el dispositivo de 120 nm

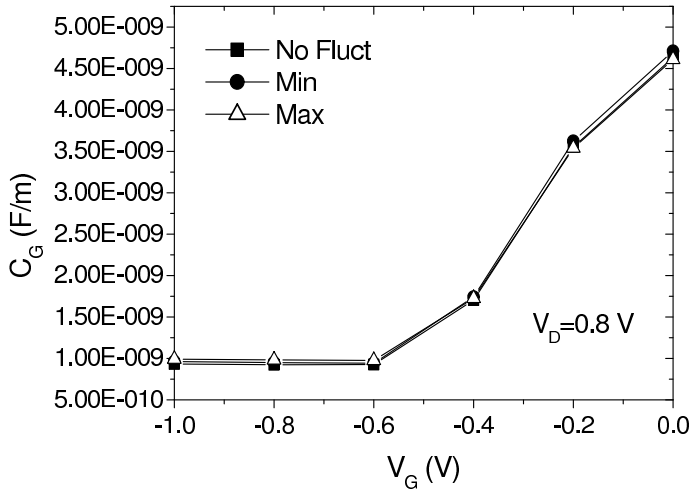


Figura 5.34: Dependencia de la capacitancia de puerta con la tensión de puerta aplicada para el dispositivo HEMT de 50 nm a un valor de tensión de drenador de 0.8 V.

de longitud de puerta se encuentra una excursión del 32 % en los valores de frecuencia al compararlos con el caso continuo.

En el caso del dispositivo HEMT de 50 nm de longitud de puerta, se ha realizado el mismo estudio. Las figuras 5.33, 5.34 y 5.35 representan la influencia de la tensión de puerta en la transconductancia, capacitancia de puerta y frecuencia de corte del dispositivo respectivamente. Para ello se consideran tres posibles situaciones, el caso continuo, es decir, sin fluctuaciones, y las dos configuraciones del dispositivo en las que el impacto de las fluctuaciones es máximo o mínimo. Estos resultados han sido obtenidos para un valor fijo de tensión de drenador de 0.8 V.

Las fluctuaciones de parámetros intrínsecos en el dispositivo de 50 nm de longitud de puerta no presentan el comportamiento simétrico que se encontraba en el dispositivo de 120 nm de longitud de puerta. Se observa que a tensiones de puerta superiores a -0.4 V, el impacto de las fluctuaciones siempre provoca una disminución en f_T con respecto al valor obtenido en el caso uniforme. En este caso, a un valor de tensión de puerta de 0.0 V, la reducción en la frecuencia puede llegar a ser del orden del 50 % del valor obtenido considerando valores continuos de los parámetros materiales.

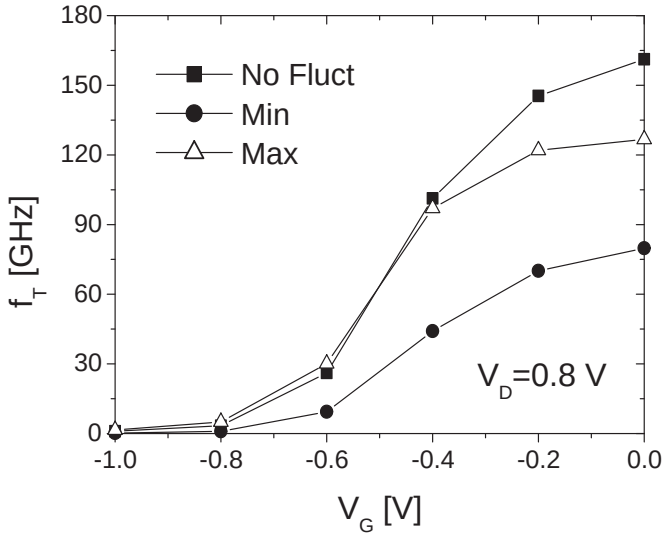


Figura 5.35: Dependencia de la frecuencia de corte con la tensión de puerta aplicada para el dispositivo HEMT de 50 nm a un valor fijo de tensión de drenador de 0.8 V.

5.6. Resumen

En este capítulo se ha presentado un estudio de la influencia de diversas fuentes de fluctuaciones de parámetros intrínsecos en el comportamiento de los dispositivos HEMT. Para ello, se han descrito detalladamente las características de los dos dispositivos HEMT utilizados en el estudio, un PHEMT con una longitud de puerta de 120 nm y un HEMT con una longitud de puerta de 50 nm, y su calibración. También se han introducido las tres fuentes de fluctuaciones de parámetros intrínsecos a estudiar, la influencia de considerar la naturaleza discreta de la materia, las variaciones en la composición de los materiales ternarios que componen el canal del dispositivo y la presencia de carga interfacial en la interfaz en una cierta región. Así, básicamente en este capítulo se trata de encontrar respuesta a las siguientes preguntas:

- ¿Cúal es el efecto de las fuentes de parámetros intrínsecos en el comportamiento de los dispositivos?

- ¿Cúal es la fuente de parámetros intrínsecos con un mayor impacto en

las curvas características de los dispositivos?

- ¿Cúal es la dependencia de las fluctuaciones de parámetros intrínsecos con el tamaño del dispositivo?
- ¿Cúal es el impacto de las fluctuaciones de parámetros intrínsecos en la frecuencia de corte de los dispositivos?

En resumen, en este capítulo se ha estudiado la influencia de diferentes fuentes de fluctuaciones de parámetros intrínsecos en la fiabilidad de los dos dispositivos HEMT analizados.

Conclusiones y principales aportaciones

La simulación de dispositivos electrónicos se ha convertido en una de las herramientas imprescindibles en el diseño de circuitos integrados, puesto que su uso permite abaratar las fases de diseño e investigar nuevos dispositivos, evitando realizar costosos y numerosos experimentos.

El escalado de los dispositivos hacia tamaño cada vez más reducidos hace necesaria la simulación de nuevos fenómenos que cobran cada vez una mayor importancia. Este es el caso de las fluctuaciones de parámetros intrínsecos en los dispositivos, que están asociadas con la naturaleza discreta de la carga y la materia. El estudio de este tipo de fluctuaciones requiere el uso de simuladores tridimensionales, puesto que estos fenómenos son de naturaleza 3D. El coste computacional de este tipo de simulaciones es muy elevado, por lo que es aconsejable el uso de simuladores paralelos.

En este trabajo se utiliza un simulador 3D paralelo de dispositivos HEMT para el estudio del impacto de diversas fuentes de fluctuaciones intrínsecas en las curvas características de estos dispositivos. El número de simulaciones a realizar en este tipo de estudios es elevado, puesto que es necesario simular un conjunto lo suficientemente grande de configuraciones diferentes de los dispositivos tal que permita realizar un análisis estadístico válido de los resultados. Por lo tanto, el problema a tratar exige el uso de un elevado número de recursos e implica mucho consumo de tiempo de CPU.

El simulador de dispositivos utilizado en esta memoria estaba inicialmente desarrollado para su funcionamiento con dispositivos BJT y HBT. Por lo tanto el punto de partida de este trabajo ha sido la adaptación del código para lograr su correcto funcionamiento con dispositivos HEMT y la implementación de las diferentes fuentes de fluctuaciones a estudiar. A causa de la necesidad de optimizar el uso de los recursos computacionales disponibles se ha realizado un estudio de la eficiencia paralela del simulador 3D y se han propuesto tres alternativas diferentes para mejorarla.

En primer lugar se ha realizado un estudio general de la eficiencia obtenida con algunos de los métodos de resolución y técnicas de precondicionamiento disponibles para la resolución de los sistemas de ecuaciones lineales dispersos que surgen de la discretización de las ecuaciones de arrastre–difusión relativas tanto a dispositivos HEMT como a otros dispositivos. Este estudio ha permitido la elección de los métodos de resolución más adecuados para nuestro problema particular, siempre teniendo presente el objetivo de la minimización del tiempo de computación.

En segundo lugar se ha encontrado que la etapa de resolución de ecuaciones lineales implementada en el simulador consumía más del 90 % del tiempo total de simulación. Por lo tanto se ha analizado esta etapa en profundidad, con el objetivo de buscar su parte más costosa. Se encontró que una gran parte del tiempo utilizado era empleado en la realización de las factorizaciones incompletas LU, por lo que se ha modificado el código para reducir la influencia de este tiempo, lográndose un importante incremento en la eficiencia paralela del simulador 3D de dispositivos, tanto en la resolución de la ecuación de Poisson como en la resolución de la ecuación de continuidad de electrones. Esto es de suma importancia, puesto que anteriormente al resolver la ecuación de continuidad de electrones se producía un descenso muy importante en la eficiencia paralela del proceso de simulación.

En tercer lugar se ha tratado de aprovechar, en el momento de realizar el particionamiento de la malla tetraédrica de elementos finitos, que el flujo de corriente en el interior de estos dispositivos, y por supuesto en otros de la misma naturaleza, se produce en una única dirección. Se han presentado resultados de tiempo satisfactorios utilizando la nueva propuesta de particionamiento, que mejoran para un número pequeño de procesadores los tiempos obtenidos usando el particionador METIS. Es necesario decir que esta propuesta aún no ha sido plenamente desarrollada, puesto que el siguiente paso sería la adaptación del código del simulador para el pleno aprovechamiento del paralelismo de grano grueso inherente en él. Esta es una de nuestras propuestas para un trabajo futuro.

Una vez optimizado el simulador 3D paralelo de dispositivos HEMT, este ha sido utilizado en el análisis de la influencia de las fluctuaciones de parámetros intrínsecos en el comportamiento de los dispositivos. En este estudio se han utilizado dos dispositivos diferentes, un dispositivo PHEMT de 120 nm de longitud de puerta con un canal de $In_{0.2}Ga_{0.8}As$ y un dispositivo HEMT de 50 nm de longitud de puerta con un canal de $In_{0.3}Ga_{0.7}As$. Se han tratado tres fuentes de fluctuaciones diferentes, la variación aleatoria de la composición de los materiales ternarios que forman el canal del dispositivo, la influencia de la naturaleza discreta de los átomos dopantes y la variación

aleatoria en la carga interfacial presente entre dos fronteras del dispositivo.

Los resultados obtenidos en este trabajo muestran que, en los dispositivos analizados, la distribución aleatoria de los dopantes en el interior de la capa *δ–doping* es la principal fuente de fluctuaciones en la corriente del HEMT, por ejemplo las variaciones en el contenido en Indio en el canal dan lugar a desviaciones estándar normalizadas de la corriente menores que el 30 % de las provocadas por los dopantes en el *δ–doping*. Además, el impacto de las fluctuaciones debidas a la presencia de carga interfacial en las regiones *recess* del dispositivo es mucho menos importante que el causado por las otras dos fuentes de fluctuaciones mencionadas anteriormente.

Se han comparado conjuntamente los resultados obtenidos con los dos dispositivos analizados, encontrándose que la influencia de las fluctuaciones de parámetros intrínsecos es relativamente pequeña en el dispositivo PHEMT de 120 nm, obteniéndose como máximo fluctuaciones en la corriente de drenador del 18 %. Sin embargo, este valor aumenta al reducirse el tamaño del dispositivo. Así, en el HEMT de 50 nm las fluctuaciones en la corriente de drenador han alcanzado el 70 %, lo que puede afectar de forma considerable al comportamiento de los dispositivos HEMT de alto rendimiento. En cambio, el efecto de las fluctuaciones debidas a la presencia de carga interfacial en las regiones *recess* del dispositivo es más importante en el dispositivo PHEMT de 120 nm que en el HEMT de 50 nm.

Por último, además de la influencia de las fluctuaciones de parámetros intrínsecos en la corriente de drenador de los dispositivos, también se ha estudiado el impacto de las fluctuaciones en la frecuencia de corte. Para ambos dispositivos las fluctuaciones de parámetros intrínsecos afectan muy severamente a la frecuencia de corte. En el dispositivo PHEMT de 120 nm se ha llegado a encontrar una excursión del 32 % en los valores de frecuencia al compararlos con el caso continuo. En el HEMT de 50 nm la influencia de las fluctuaciones es más importante, puesto que se observa que a tensiones de puerta superiores a -0.4 V, el impacto de las fluctuaciones siempre provoca una disminución en la frecuencia de corte con respecto al valor obtenido en el caso uniforme, pudiendo llegar la reducción en la frecuencia a un 50 % del valor obtenido considerando valores continuos de los parámetros materiales.

La continuación de este trabajo en un futuro inmediato puede orientarse en varias direcciones. Para ello, se distinguen los dos principales aspectos en los que se ha centrado el desarrollo de esta memoria, los métodos numéricos y la simulación de dispositivos semiconductores. En el marco de los métodos numéricos se pretende continuar el desarrollo de la optimización del simulador basada en una nueva estrategia de particionamiento de la malla tetraédrica, descrita en el capítulo 4, modificando para ello el método en el

que se realizan las comunicaciones de los nodos frontera entre procesadores. Además, se desea implementar la librería numérica PETSc en el simulador 3D paralelo para utilizarla cuando el número de procesadores sea elevado. Por otra parte también se pretende estudiar la adaptación de la aplicación a entornos de computación Grid. En el marco de los dispositivos semiconductores se tratará de avanzar en varios frentes. Inicialmente se buscará ampliar el simulador utilizando modelos de simulación más precisos, como el modelo hidrodinámico o el Monte Carlo. Además, se tratará de adecuar el simulador para estudiar otros fenómenos físicos, tales como el ruido de los dispositivos o las fluctuaciones debidas a variaciones en el grosor de las capas de los materiales. Por último, se utilizará el simulador para modelar el comportamiento de HEMTs formados por otros materiales diferentes a los estudiados.

Bibliografía

[1] G. E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38:114–117, (1965).

[2] H. S. P. Wong. Beyond the conventional transistor. *IBM Journal of Research and Development*, 46:133–165, (2002).

[3] A. Asenov. Random dopant induced threshold voltage lowering and fluctuations in Sub–0.1 μm MOSFETs: a 3D "atomistic" study. *IEEE Transactions on Electron Devices*, 12:2505–2513, (1998).

[4] R. W. Keyes. Fundamental limits of silicon technology. *Proc. IEEE*, 89:259–288, (2001).

[5] Y. Taur, D. Buchanan, W. Chen, D. Frank. CMOS scaling into nanometer regime. *Proc. IEEE*, 85:486–503, (1997).

[6] D. Frank, R. Dennard, E. Nowak, P. Solomon, Y. Taur, H. S. Wong. Device scaling limits of Si MOSFETs and their application dependencies. *Proc. IEEE*, 89:227–239, (2001).

[7] M. V. Fischetti. Scaling MOSFETs to the limit: A physicists's perspective. *Journal of Computational Electronics*, 2:73–79, (2003).

[8] P.M. Solomon, K. W. Guarini, K. Chan, E. C. Jones, et al. Two gates are better than one: double–gate MOSFET process. *IEEE Circuits and Devices Magazine*, 19:48–62, (2003).

[9] M. Ieong, H.–S. P. Wong, E. Nowak, J. Kedzierski, E. C. Jones. High performance double–gate device technology challenges and opportunities. *International Symposium on Quality Electronic Design, Proceedings*, 492–495, (2002).

[10] Y. Yamashita, A. Endoh, K. Shinohara. Pseudomorphic $In_{0.52}Al_{0.48}As/In_{0.7}Ga_{0.3}As$ HEMTs with an ultrahigh f_T of 562 GHz. *Electron Device Letters, IEEE*, 23:573–575, (2002).

[11] K. Kalna, A. Asenov. Scaling of pHEMTs to Decanano Dimensions. *VLSI Design*, 13:435–439, (2001).

[12] K. Kalna, A. Asenov. Role of multiple delta doping in PHEMTs scaled to sub–100 nm dimensions. *Solid–State Electronics*, 48:1223–1232, (2004).

[13] C. S. Whelan, P. F Marsh, W. E. Hoke. GaAs metamorphic HEMT (MHEMT): an attractive alternative to InP HEMTs for high performance low noise and power applications. *Indium Phosphide and Related Materials, 2000. Conference Proceedings*, 337–340, (2000).

[14] S. G. Badcock, A. G. O'Neill, E. G. Chester. Device and circuit performance of SiGe/Si MOSFETs. *Solid–State Electronics*, 46:1925–1932, (2002).

[15] R. J. Welty, K. Mochizuki, C. R. Lutz, R. E. Welser, P. M. Asbeck. Design and performance of tunnel collector HBTs for microwave power amplifiers. *IEEE Transactions on Electron Devices*, 50:894–900, (2003).

[16] M. Ida, K. Kurishima, N. Watanabe. Over 300 GHz f_T and f_{max} InP/InGaAs double heterojunction bipolar transistors with a thin pseudomorphic base. *IEEE Electron Device Letters*, 23:694–696, (2002).

[17] J. T. Park, J. P. Colinge. Multiple–Gate SOI MOSFETs: Device desing guidelines. *IEEE Transactions on Electron Devices*, 49:2222–2229, (2002).

[18] H. S. Wong, D. Frank, P. Solomon. Device design considerations for Double–Gate, Ground–Plane and Single Gated Ultra–Thin SOI MOSFETs at the 25nm channel length generation. *Electron Devices Meeting. IEDM'98 Technical Digest.*, 407–410, (1998).

[19] P. D. Ye, G. D. Wilk, J. Kwo, et al. GaAs MOSFET with oxide gate dielectric grown by atomic layer deposition. *IEEE Electron Device Letters*, 24:209–211, (2003).

[20] K. Kalna, M. Boriçi, L. Yang, A. Asenov. Monte Carlo simulations of III–V MOSFETs, *Semicond. Sci. Technol.*, 19:S202–S205, (2004).

[21] P. Roblin, H. Rohdin. High–Speed heteroestructure devices. Cambridge University Press. (2002).

[22] International Technology Roadmap for Semiconductors, 2005 Edition. Radio Frequency and Analog/Mixed–Signal Technologies for Wireless Communications.

[23] Taurus Medici, User Guide. Version W–2004.09 (2004).

[24] Sentaurus Device, Versatile, Multifunctional Device Simulator. SYNOPSYS (2005).

[25] ATLAS User's Manual. Device Simulator Software (2002).

[26] BIPOLE3 Tutorial Guide. Version 5.20 (2005).

[27] APSYS: Advanced Physical Models of Semiconductor Devices. Crosslight Device Simulation, Software Information (2005).

[28] U. Ravaioli. Hierarchy of simulation approaches for hot carrier transport in deep submicron devices. *Semicond. Sci. Technol.*, 13:1–10, (1998).

[29] S. Selberherr. Analysis and Simulation of Semiconductor Devices. Springer (1984).

[30] A.J.García Loureiro, K.Kalna, A.Asenov. Efficient three–dimensional parallel simulations of PHEMTs. *Electronics Networks, Devices and Fields*, 18:327–340 (2005).

[31] Y. P. Zhao, J. R. Watling, S. Kaya, A. Asenov, J. R. Barker. Drift Diffusion and hydrodynamic simulations of Si/SiGe p–MOSFETs. *Materials Science and Engineering (B)–Solid State Materials for Advanced Technology*, 72:180–183 (2000).

[32] N. R. Aluru, K. H. Law, R. W. Dutton. Simulation of the Hydrodynamic Device Model on Distributed Memory Parallel Computers. *IEEE Trans. on Computer–Aided Design of Integrated Circuits and Systems*, 15,9:1029–1047, (1996).

[33] C. L. Gardner. The Quantum Hydrodynamic Model for Semiconductor Devices. *SIAM Journal on Applied Mathematics*, 54:409–427 (1994).

[34] C. Jacoboni, P. Lugli. The Monte Carlo method for semiconductor device simulation. Springer–Verlag, (1989).

[35] C. Jacoboni, L. Reggiani. The Monte Carlo method for the solution of charge transport in semiconductors with applications to covalent materials. *Reviews of Modern Physics*, 55:645–705 (1983).

[36] F. M. Bufler, Y. Asahi, H. Yoshimura, et al. Monte Carlo simulation and measurement of nanoscale n–MOSFETs. *IEEE Transactions on Electron Devices*, 50:418–424, (2003).

[37] G. F. Formicone, M. Saraniti, D. Z. Vasileska, D.K. Ferry. Study of a 50 nm nMOSFET by ensemble Monte Carlo simulation including a new approach to surface roughness and impurity scattering in the Si inversion layer. *IEEE Transactions on Electron Devices*, 49:125–132, (2002).

[38] J. Mateos, T. Gonzalez, D. Pardo, V. Hoel, H. Happy, A. Cappy. Improved Monte Carlo algorithm for the simulation of δ–doped AlInAs/GaInAs HEMTs. *IEEE Transactions on Electron Devices*, 47:250–253 (2000).

[39] K. Kalna, S. Roy, A. Asenov, K. Elgaid, I. Thayne. Scaling of pseudomorphic high electron mobility transistors to decanano dimensions. *Solid–State Electronics*, 46:631–638 (2002).

[40] S. Datta. Nanoscale device modeling: the Green's function method. *Superlattices and Microstructures*, 28:253–278 (2000).

[41] A. Martinez, A. Svizhenko, M. P. Anantram, J. R. Barker, A. R. Brown, A. Asenov. A Study of the Interface Roughness on a DG–MOSFET using a Full 2D NEGF Technique. *International Electron Devices Meeting (IEDM) Technical Digest*, 627–630 (2005).

[42] M. G. Ancona, G. J. Iafrate. Quantum correction to the equation of state of an electron gas in a semiconductor. *Physical Review B*, 39:9536–9540 (1989).

[43] A. Wettstein, A. Schenk, W. Fichtner. Quantum device–simulation with the density–gradient model on unstructured grids. *IEEE Transactions on Electron Devices*, 279–284 (2001).

[44] D.K. Ferry, R. Akis, D. Vasileska. Quantum effects in MOSFETs: use of an effective potential in 3D Monte Carlo simulation of ultra–short channel devices. *Electron Devices Meeting, 2000. IEDM Technical Digest. International*, 287–290 (2000).

[45] A. DeMari. An Accurate Numerical Steady–State One–Dimensional Solution of the P–N Junction. *Solid–State Electronics*, 11:33–58 (1968).

[46] A. DeMari. An Accurate Numerical Steady–State One–Dimensional Solution of the P–N Junction under Arbitrary Transient Conditions. *Solid–State Electronics*, 11:1021–1053 (1968).

[47] P. A. Markowich. The Stationary Semiconductor Device Equations. Springer–Verlag (1986).

[48] D. L. Scharfetter, H. K. Gummel. Large–Signal Analysis of a Silicon Read Diode Oscillator. *IEEE Trans. on Electron Devices*, 64–77 (1969).

[49] H. K. Gummel. A self–consistent iterative scheme for one–dimensional steady state transistor calculations. *IEEE Trans. on Electron Devices*, 11 (1964).

[50] R. E. Bank, D. J. Rose. Global approximate Newton methods. *Numerische Mathematik*, 37 (1981).

[51] S. Rollin, O. Schenk, A. Gupta. The effects of unsymmetric matrix permutations and scalings in semiconductor device and circuit simulation. *IEEE Trans. on Computer–Aided Design of Integrated Circuits and Systems*, 23:400–411 (2004).

[52] N. Seoane, A.J.García Loureiro. Study of parallel numerical methods for semiconductor device simulation. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, 19:15–32 (2006).

[53] Y. Saad, G.C. Lo. Iterative solution of general sparse linear systems on clusters of workstations. Technical report, Univ. of Minnesota, Dept. of Computer Science (1996).

[54] Y. Saad. Iterative methods for sparse linear systems. PWS Publishing Co., (1996).

[55] M. J. Flynn. Some computers organizations and their effectiveness. *IEEE Trans. on Computers*, 21:948–960 (1972).

[56] MPI: A Message–Passing Interface Standard, University of Tennessee (1995).

[57] W. Group, E. Lusk, A. Skjellum. Using MPI. The MIT Press (1996).

[58] S. A. Mitchell, S. A. Vavasis. Quality mesh generation in higher dimensions. *SIAM J. on Computing*, 29:1334–1370 (2000).

[59] S. A. Vavasis. QMG 1.1 reference manual, Computer Science Department, Cornell University, (1996).

[60] M. Aldegunde, J.J. Pombo, A. J. García Loureiro. Octree–based mesh generation for the simulation of semiconductor devices. *XX Conference on Design of Circuits and Integrated Systems (DCIS 2005)*, (2005).

[61] B. Hendrickson, R. Leland. The CHACO user's guide, version 2.0. Sandia National Laboratories, (1994).

[62] G. Karypis, V. Kumar. METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill–reducing orderings of sparse matrices. Univ. of Minnesota, (1997).

[63] G. Karypis, V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. Dept. of Computer Science. Univ. of Minnesota, (1995).

[64] C. S. Rafferty, M. R. Pinto, R. W. Dutton. Iterative methods in semiconductors device simulation. *IEEE Trans. on Computer–Aided Design*, 4:462–471, (1985).

[65] R.E. Bank, D. J. Rose. Parameter selection for Newton–like methods applicable to nonlinear partial differential equations. *SIAM J. Numer. Anal.*, 17:806–822, (1980).

[66] J. M. Ortega, W. C. Rheinboldt. Iterative solution of nonlinear equation in several variables. Academic Press, (1970).

[67] R. Barrett, M. Berry, T. Chan, J. Demmel et al. Templates for the solution of linear systems: building blocks for iterative methods. SIAM, (1994).

[68] C. Ringhofer, C. Schmeiser. A modified Gummel method for the basic semiconductor device equations. *IEEE Trans. on Computer–Aided Design*, 7:251–253, (1988).

[69] Ke–Chih Wu, R. F. Lucas, Z. Y. Wang, R. W. Dutton. New approaches in a 3–D one–carrier device solver. *IEEE Trans. on Computer-Aided Design*, 8:528–537, (1989).

[70] D. Kincaid, W. Cheney. Numerical analysis. Brooks/Cole, (1991).

[71] Y. Saad. Krylov subspace methods on parallel computers. Computer Science Department. Univ. Minnesota, (1995).

[72] T. F. Chan, H. A. van der Vorst. Approximate and incomplete factorizations. University of California, (1994).

[73] J. A. Meijerink, H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M–matrix. *Math. Compt.*, 31:148–162, (1997).

[74] Z. Zlatev. Use of iterative refinement in the solution of sparse linear systems. *SIAM Jour. Numer. Analysis*, 19:381–399, (1982).

[75] Y. Saad. ILUT: a dual threshold incomplete LU factorization. *Numerical Linear Algebra with Applications*, 4, (1994).

[76] C. Douglas. Multigrid methods in science and engineering. *IEEE Computational Science and Engineering*, 55–68, (1996).

[77] W. L. Briggs. A multigrid tutorial. *SIAM*, (1987).

[78] P. Wesseling. Introduction to multigrid methods. NASA ICASE, NASA Langley Research Center, Virginia, (1995).

[79] B. F. Smith, P. E. Bjorstad, W. D. Gropp. Domain decomposition: parallel multilevel methods for elliptic partial differential equations. Cambridge University Press, (1996).

[80] J. J. Pombo. Estudio y comparación de las técnicas de reordenamiento para matrices dispersas. Memoria de Licenciatura, Departamento de Electrónica e Computación (1997).

[81] J. A. George. Nested dissection of a regular finite element mesh. *SIAM J. Numerical Analisys*, 10:345–363 (1973).

[82] J. A. George, J. W. H. Liu. An automatic nested dissection algorithm for irregular finite element problems. *SIAM J. Numerical Analisys*, 15:1053–1069 (1978).

[83] J. Dongarra. Sparse Matrix Storage Formats. Templates for the Solution of Algebraic Eigenvalue Problems: A practical guide. SIAM (2000).

[84] Página Web: http://math.nist.gov/MatrixMarket/formats.html#hb

[85] M. Shur. Physics of Semiconductor Devices. Prentice Hall, (1990).

[86] K. Hwang, D. H. Navon, T. W. Tang, M. A. Osman. Improved Convergence of Numerical Device Simulation Iterative Algorithms. *IEEE Trans. on Electron Devices*,32,6:1143–1145 (1985).

[87] Y. Saad, G. C. Lo, S. Kuznetsov. PSPARSLIB users manual: A portable library of parallel iterative solvers. Technical report, Univ. of Minnesota, Dept. of Computer Science, (1997).

[88] Página Web: http://www.tecplot.com

[89] N. Seoane, A. J. García Loureiro. Estudio de métodos iterativos aplicados a la resolución paralela de la ecuación de Poisson. Congresso de Métodos computacionais em Engenharia, (2004).

[90] N. Seoane, A. J. García Loureiro. Study of numerical libraries to solve the 3D Poisson equation in advanced transistors. High Performance Computing for Computational Science (VECPAR), (2004).

[91] N. Seoane, A. J. García Loureiro. Analysis of Parallel Numerical Libraries to Solve the 3D Electron Continuity Equation. *Lecture Notes in Computer Science*, 3036:590–593, (2004).

[92] Centro de Supercomputación de Galicia.
Página Web: http://www.cesga.es

[93] N. Seoane, A. J. García Loureiro, K. Kalna, A. Asenov. A high-performance parallel device simulator for high electron mobility transistors. Parallel Computing (ParCo), (2005).

[94] Edinburgh Parallel Computing Centre.
Página Web: http://www.epcc.ed.ac.uk

[95] Barcelona Supercomputing Centre.
Página Web: http://bsc.es

[96] N. Seoane, A. J. García Loureiro, K. Kalna, A. Asenov. Atomistic effect of delta doping layer in a 50 nm Inp HEMT. *Journal of Computational Electronics*, 5: 131–135 (2006).

[97] Device Modelling Group at Glasgow Universtity.
Página Web: http://www.elec.gla.ac.uk/groups/dev_mod

[98] HSL Library. Numerical Analysis Group of the Computational Science & Engineering Department (CCLRC), United Kingdom. Página Web: http://www.cse.clrc.ac.uk/nag/hsl/contents.shtml, (2004).

[99] S. Eisenstat, M. Gursky, M. Schultz, A. H. Sherman. Yale sparse matrix package: The symmetric codes. *Int. J. Numer. Meth. in Eng.*, 18:1145–1151, (1982).

[100] Y. Saad. SPARSKIT: a Basic Tool Kit for Sparse Matrix Computations. University of Illinois, Urbana, IL, (1994).

[101] S. A. Hutchinson, J. Shadid, R. S. Tuminaro. Aztec User's Guide. Sandia National Laboratories, (1999).

[102] M. T. Jones, P. E. Plassmann. BlockSolve95 Users Manual: Scalable Library Software for the Parallel Solution of Sparse Linear Systems. Argonne National Laboratory, (1997).

[103] PETSc. Página Web: http://www.mcs.anl.gov/petsc

[104] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp et al. PETSc Users Manual. Argonne National Laboratory, (2004).

[105] Y. Saad, A. V. Malevsky. Data structures, computational, and communication kernels for distributed memory sparse iterative solvers. Technical report, Univ. of Minnesota, Dept. of Computer Science, (1995).

[106] J. W. Demmel, J. R. Gilbert, X. S. Li. SuperLU Users Guide, (2003).

[107] Y. Saad, M. Sosonkina. pARMS: a package for the parallel iterative solution of general large sparse linear systems. User's guide, (2003).

[108] Multifrontal Massively Parallel Solver, MUMPS version 4.6.3. User's guide, (2006).

[109] N. Seoane Iglesias, A. García Loureiro. Optimisation of the Parallel Performance of a 3D Device Simulator for High Electron Mobility Transistors. *Lecture Notes in Computer Science*, 4330:859–868, (2006).

[110] N. Seoane Iglesias, A. García Loureiro. Simulación tridimensional paralela de dispositivos HEMT usando el modelo de arrastre difusión. XVII Jornadas de Paralelismo, (2006).

[111] A. García Loureiro, N. Seoane Iglesias, M. Aldegunde Rodríguez. Estudio de técnicas de particionamiento aplicadas a la simulacion paralela de transistores. XVII Jornadas de Paralelismo, (2006).

[112] A. Asenov, A. R. Brown, J. H. Davies, S. Saini. Hierarchical approach to "atomistic" 3–D MOSFET simulation. *IEEE Transactions on Computer–Aided Design of Integrated Circuits and Systems*, 18,11:1558–1565, (1999).

[113] R. Cao et al. Estadística básica aplicada. Tórculo Artes Gráficas, (1998).

[114] A. Asenov, G. Slavcheva, A. R. Brown, R. Balasubramaniam, J. H. Davies. Statistical 3–D "atomistic" simulation of decanano MOSFETs. *Superlattices and Microstructures*, 27:215–227, (2000).

[115] A. Asenov, S. Saini. Suppression of random dopant induced threshold voltage fluctuations in sub–0.1 μm MOSFET's with epitaxial and δ–doped channels. *IEEE Trans. on Electron Devices*, 46:1718–1723, (1999).

[116] A. Asenov, S. Saini. Polisilicon gate enhancement of the random dopant induced threshold voltage fluctuations in sub 100 nm MOSEFTs with ultrathin gate oxides. *IEEE Trans. on Electron Devices*, 47:805–812, (2000).

[117] K. Kalna, A. Asenov. Nonequilibrium transport in scaled high electron mobility transistors. *Semicond. Sci. Technol.*, 17:579–584, (2002).

[118] D. M. Caughey, R. E. Thomas. Carrier mobilities in silicon empirically related to doping and fields. *Proc. IEEE*,55:2192–2193, (1967).

[119] Kalna K, Elgaid K, Thayne I, Asenov A. Modelling of InP HEMTs with high Indium content channels. *Proc. Indium Phosphide and Related Materials Conf. (IPRM)*, 61–65, (2005).

[120] Babiker S, Asenov A, Cameron N, and Beaumont SP. A simple approach to include external resistances in the Monte Carlo simulation of MESFETs and HEMTs. *IEEE Trans. on Electron Devices*, 43:2032–2034 (1996).

[121] R. W. Keys. Physical limits in digital electronics. *Proceedings of the IEEE*, 63,5:740–766, (1975).

[122] T. Mizuno, J. Okumtura, A. Toriumi. Experimental study of threshold voltage fluctuation due to statistical variation of channel dopant number in MOSFET's. *IEEE Trans. on Electron Devices*, 41,11:2216–2221, (1994).

[123] J. T. Horstmann, U. Hilleringmann, K. F. Goser. Matching analysis of deposition defined 50–nm MOSFET's. *IEEE Trans. on Electron Devices*, 45,1:299–306, (1998).

[124] K. Nishinohara, N. Shigyo, T. Wada. Effects of microscopic fluctuations in dopant distributions on MOSFET threshold voltage. *IEEE Trans. on Electron Devices*, 39,3:634–639, (1992).

[125] P. A. Stolk, F. P. Widdershoven, D. B. M. Klaassen. Modeling statistical dopant fluctuations in MOS transistors.*IEEE Trans. on Electron Devices*, 45,9:1960–1971, (1998).

[126] A. Asenov, G. Slavcheva, A. R. Brown, J. H. Davies, S. Saini. Increase in the random dopant induced threshold fluctuations and lowering in sub–100 nm MOSFETs due to quantum effects: a 3D density–gradient simulation study. *IEEE Trans. on Electron Devices*, 48,4:722–729, (2001).

[127] S. T. Martin, G. P Li, E. Worley, J. White. The gate bias and geometry dependence of random telegraph signal amplitudes. *IEEE Electron Device Letters* , 18,9:444–446, (1997).

[128] A. Avellan, W. Krautschneider, S. Schwantes. Observation and modeling of random telegraph signals in the gate and drain currents of tunneling metal–oxide–semiconductor field–effect transistors. *Applied Physics Letters*, 78,18:2790–2792, (2001).

[129] K. S. Ralls, W. J. Skocpol, L. D. Jackel, R. E. Howard, L. A. Fetter, R. W. Epworth, D. M. Tennant. Discrete resistance switching in submicrometer silicon inversion layers: individual interface traps and low–frequency (1/f) noise. *Phys. Rev. Lett.*, 52:228–231, (1984).

[130] P. J. Restle, J. W. Park, B. F. Lloyd. DRAM variable retention time. *International Electron Devices Meeting, Technical Digest.*, 807–810, (1992).

[131] S. Thompson, M. Alavi, R. Arghavani, et al. An enhanced 130 nm generation logic technology featuring 60 nm transistors optimized for high performance and low power at 0.7–1.4 V. *International Electron Devices Meeting, 2001. IEDM Technical Digest.*, 1161–1164, (2001).

[132] A. Asenov, A. R. Brown, J. H. Davies, S. Kaya, G. Slavcheva. Simulation of intrinsic parameter fluctuations in decananometer and nanometer–scale MOSFETs. *IEEE Trans. on Electron Devices*, 50,9:1837–1852, (2003).

[133] A. Asenov, S. Kaya. Effect of oxide interface roughness on the threshold voltage fluctuations in decanano MOSFETs with ultrathin gate oxides. *International Conference on Simulation of Semiconductor Processes and Devices, SISPAD*,135–138, (2000).

[134] A. Asenov, S. Kaya, A. R. Brown. Intrinsic parameter fluctuations in decananometer MOSFETs introduced by gate line edge roughness. *IEEE Trans. on Electron Devices*, 50,5:1254–1260, (2003).

[135] N. Seoane, A. J. García Loureiro, K. Kalna, A.Asenov. Current Variations in pHEMTs introduced by channel composition fluctuations. *Journal of Physics: Conference Series.*, 38: 212–215 (2006).

[136] N. Seoane, A. J. García Loureiro, K. Kalna, A.Asenov. A 3D parallel simulation of the effect of interface charge fluctuations in HEMTs. 11th International Workshop on Computational Electronics, (2006).

[137] K. Brennan. Introduction to Semiconductor Devices: For Computing and Telecommunications Applications. Cambridge Univ. Press (2005).

[138] S. E. Laux. Techniques for small–signal analysis of semiconductor devices. *IEEE Trans. on Electron Devices*,32,10:2028–2037, (1985).