



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
PROGRAMA DE MAESTRÍA Y DOCTORADO EN CIENCIAS
MATEMÁTICAS Y DE LA ESPECIALIZACIÓN EN ESTADÍSTICA
APLICADA.

UN MODELO DE OPTIMIZACIÓN PARA EL DISEÑO DE UNA PALETA
DE COLORES

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN CIENCIAS

PRESENTA:
CALEB ERUBIEL ANDRADE SERNAS

DIRECTOR DE TESIS:
DR. DAVID ROMERO VARGAS
[INSTITUTO DE MATEMÁTICAS U.N.A.M.](#)

MÉXICO, D. F. AGOSTO 2014

A la memoria de mi madre,

Juana Sernas Sayavedra
(1951 – 2012)

*...mujer llena de fe y perseverancia; con ánimo incansable e inspirador siempre me
exhortaste a seguir adelante y no desfallecer. Siempre te llevaré en mi mente y
corazón.*

Un modelo de optimización para el diseño de una paleta de colores

Prefacio

En términos generales, la ciencia de la optimización consiste en hacer la mejor elección dentro de un grupo de posibles opciones cuando hay un conjunto de requerimientos en conflicto a satisfacer. Esto puede resultar en un problema bastante complejo de resolver. En este trabajo, tenemos que elegir una paleta de n colores de entre todas las posibles combinaciones en un espacio de color de cardinalidad finita, de tal manera que todos los colores sean bien distinguibles entre sí para un observador promedio. La manera de medir la diferencia entre colores será a través de la métrica euclíadiana, puesto que los espacios de color se modelan en \mathbb{R}^3 .

El planteamiento del problema resulta ser sencillo, pero la primera dificultad surge al darnos cuenta que es necesario contar con un espacio de color donde las diferencias entre sus elementos sean percibidas de manera uniforme en todo el espacio de color. Es importante que exista una correlación directa entre la distancia medida de dos colores y la diferencia que percibe un observador promedio de los tales, de ahí surge el concepto de que un espacio de color sea perceptualmente uniforme. Otra dificultad es que el espacio de color que utilizaremos no es convexo, y su cardinalidad de 256^3 elementos hace totalmente impráctico el querer resolver el problema mediante una búsqueda exhaustiva del espacio de soluciones factibles.

No se conocen métodos exactos eficientes hasta el momento para resolver este problema, y los métodos analíticos resultan ser poco apropiados para resolverlo, como lo veremos en el segundo capítulo. En este sentido, los métodos heurísticos serán los más idóneos para abordar y solucionar este problema.

El trabajo está dividido básicamente en dos partes, una teórica y una experimental. La primera parte tiene la intención de proveer un marco teórico para el planteamiento del modelo de optimización y el método heurístico empleado para la resolución del problema. En la segunda parte, se muestran los procedimientos y resultados de las implementaciones desarrolladas en un lenguaje de programación (Python), así como la interpretación de los resultados mediante el uso de herramientas estadísticas.

Puesto que este trabajo se desarrolla alrededor del concepto del color, el primer capítulo es una introducción a la teoría del color que busca dar un panorama general de cómo fueron los inicios y la evolución de esta teoría, lo cual dió como resultado un modelo

matemático que describe la manera en que los seres humanos percibimos el color.

Resultaría ambicioso tratar de abordar con detalle los conceptos y formulaciones centrales en cuanto a la teoría del color, que permitirían al lector profundizar en la disciplina de la colorimetría, sin embargo, el primer capítulo contiene toda la información necesaria para comprender la formulación del problema y el enfoque que se ha dado en este trabajo para su resolución.

El capítulo dos tiene como objetivo dar una justificación razonable para el empleo de técnicas metaheurísticas para atacar el problema propuesto. En este capítulo, se plantea el problema de la colocación de n puntos en el cubo unitario de manera que la distancia mínima entre ellos sea lo más grande posible. Se provee de una demostración analítica para resolver el caso $n = 3$, la cual resulta ser demasiado compleja para la aparente sencillez del problema. La exposición de este ejemplo hace ver que esta vía no es muy prometedora para tratar de resolver el caso general para cualquier valor de n .

El capítulo tres introduce a la optimización combinatoria como una subrama de la optimización para espacios discretos. Aquí, se explica formalmente el concepto de metaheurística como un mecanismo de búsqueda local guiado por una estrategia global, que permite aceptar soluciones factibles que, en lugar de aportar una mejora a la resolución del problema, parecen empeorar las cosas. Se aborda un caso particular, el algoritmo de recocido simulado y una variante del mismo conocida como algoritmo de umbral de aceptación. En este capítulo y en adelante, la distribución de probabilidad empleada para la selección de soluciones vecinas será la uniforme.

El capítulo cuatro está dedicado a darle un soporte teórico al problema de optimización que nos concierne, al modelarlo mediante una teoría muy socorrida con respecto a procesos estocásticos, las cadenas de Markov. Asimismo, se provee en este capítulo de resultados formales de convergencia que garantizan la efectividad del algoritmo de recocido simulado para resolver un problema combinatorio de esta magnitud. El resultado de convergencia requiere un número infinito de iteraciones para garantizar el hallazgo de la solución óptima. Por cuestiones prácticas, nos limitaremos a encontrar aproximaciones de la solución óptima, ya que la implementación del algoritmo estará limitada a un numero finito de iteraciones.

El capítulo cinco es una breve descripción de trabajos previos que se han realizado para resolver el mismo problema que concierne a este trabajo. Se hace mención de algunos inconvenientes que presentan los enfoques que se tomaron en estos trabajos.

El capítulo seis trata acerca de la descripción del algoritmo de umbral empleado para resolver el problema, primeramente en el cubo unitario, y las dos etapas en las que consiste: la construcción de la solución inicial y la etapa de búsqueda local.

En el capítulo siete se trasladan las ideas que se desarrollaron en el capítulo seis al espacio de color perceptualmente uniforme CIE $L^*u^*v^*$, donde debemos resolver el problema de optimización. Es importante señalar que, a pesar de que el problema es de naturaleza discreta, para hacer más eficiente el programa que se implementó se trabaja en el espacio como si fuera continuo, y al final, las coordenadas de los puntos de la

solución aproximada se redondean a valores enteros.

El capítulo ocho está dedicado a describir la manera en que se determinaron los parámetros que gobiernan el comportamiento del algoritmo implementado.

En el capítulo nueve se exponen los resultados experimentales, habiendo hecho pruebas con ocho variantes del algoritmo. Se utilizan pruebas estadísticas no paramétricas para sacar conclusiones importantes acerca de los datos recabados, más concretamente, del desempeño de las distintas variantes del algoritmo de umbral que se implementaron. Al final de este capítulo, se despliegan las paletas de colores para algunos valores de n , encontradas al utilizar la mejor variante del algoritmo.

Las referencias bibliográficas, tanto para citar texto como para indicar la autoría de un gráfico, están indicadas entre corchetes con las siglas del autor y el año de la publicación del trabajo.

Agradecimientos

Quiero externar primeramente mi gratitud a Dios por la oportunidad tan especial que me concede de concluir esta etapa tan importante de mi vida. A mis padres y hermanas, su amor y apoyo constante e incondicional han sido un pilar fundamental en mi vida. Por siempre les estaré agradecido, les amo profundamente.

Agradezco también al Dr. David Romero Vargas por haberme concedido el privilegio y la oportunidad de trabajar bajo su dirección en este proyecto de tesis. Debo un agradecimiento especial al Dr. Gilberto Calvillo Vives por su valioso consejo y apoyo durante la elaboración de este trabajo.

Al Dr. Emilio Marmolejo Olea le agradezco grandemente el haber sido mi tutor durante los últimos tres años. Siempre me concedió de su tiempo y consejo para consultarle sobre distintas inquietudes mías.

En su conjunto, al personal del Posgrado en Ciencias Matemáticas de la Universidad Nacional Autónoma de México, mil gracias. Haber formado parte del cuerpo estudiantil de esta noble institución ha sido un gran privilegio y honor.

Asimismo, quiero externar mi gratitud a mis sinodales por haberse dado a la tarea de revisar mi trabajo y evaluarme: Dr. David Romero Vargas (Instituto de Matemáticas, U.N.A.M.), Dr. Gilberto Calvillo Vives (Instituto de Matemáticas, U.N.A.M.), Dr. Rafael López Bracho (Departamento de Sistemas, U.A.M.), Dr. David Flores Peñaloza (Facultad de Ciencias, U.N.A.M.) y Dr. Francisco Javier Zaragoza Martínez (Departamento de Sistemas, U.A.M.). Gracias por su tiempo y sus valiosas observaciones.

Por último, agradezco al Consejo Nacional de Ciencia y Tecnología por el apoyo económico recibido durante la realización de este trabajo, como Ayudante de Investigador. Exp. Inv. 5367 y Exp. Ayte. 10458.

Índice general

Prefacio	III
Agradecimientos	VII
I. Marco teórico	3
1. Introducción a la teoría del color	5
1.1. La luz y el color	5
1.2. Modelos de sistemas de color: CIE RGB y CIE XYZ	7
1.3. Diagrama de cromaticidad	15
1.4. Espacios de color perceptualmente uniformes	17
1.5. Color de 24 bits y corrección gamma	21
2. Justificación para el empleo de métodos heurísticos	25
2.1. Un ejemplo de optimización convexa	25
3. Métodos heurísticos	33
3.1. Optimización Combinatoria	33
3.2. Metaheurísticas	35
3.3. Recocido Simulado y Umbral de Aceptación	36
4. Resultados de convergencia	43
4.1. Modelos de Markov	43
4.2. Teorema de convergencia	48
4.3. Estrategia de enfriamiento	50
5. Trabajos previos	55
6. Descripción del método de optimización en el cubo RGB	59
6.1. Formulación del problema	59

6.2. Métodos constructivos	61
6.3. Generación de mallas y descomposiciones tetraédricas recursivas	62
6.4. Metaheurística con búsqueda local	65
7. Aplicación del método de optimización en el espacio de color CIE L*u*v*	71
7.1. Formulación del problema	72
7.2. Solución inicial	72
II. Experimentos computacionales	75
8. Selección de parámetros	77
8.1. Tamaño de la δ -caja-vecindad.	78
8.2. Valor para la temperatura inicial t_0	80
8.3. Número de lote λ y factor de enfriamiento φ	81
9. Resultados experimentales	83
10. Conclusiones	95
III. Apéndice	99
A. Pruebas estadísticas no paramétricas	101
A.1. Prueba de Friedman	102
A.2. Prueba de Wilcoxon	105
B. Tablas	109
B.1. Matrices δ_0 contra θ para valores de la función objetivo	109
B.2. Matrices θ contra δ_0 para valores de la función objetivo	114
B.3. Matrices λ contra φ para valores de la función objetivo y tiempo de cálculo	119
B.4. Matrices φ contra λ para valores de la función objetivo	122
B.5. Matrices de observaciones de las ocho variantes del algoritmo para va- lores de la función objetivo	125
B.6. Matrices de observaciones de las ocho variantes del algoritmo para va- lores de tiempo de cálculo	128
Bibliografía	135

Parte I

Marco teórico

Capítulo 1

Introducción a la teoría del color

1.1. La luz y el color

La luz visible es una porción del espectro de radiación electromagnética que es perceptible por el sentido de la vista en los humanos. El rango de luz visible abarca aproximadamente longitudes de onda desde los 380 hasta los 780 nanómetros ($1 \text{ nm} = 10^{-9} \text{ m}$). La figura 1.1 nos da una representación gráfica del rango del espectro de luz visible, así como los colores que percibimos en las distintas longitudes de onda dentro del mismo. Estos colores originados por una radiación luminosa de una única longitud de onda se conocen como *colores espectrales*. Cuando se superponen radiaciones de distintas longitudes de onda se da lugar a nuevos estímulos de color. La superposición de todas las longitudes de onda da como resultado la luz blanca.

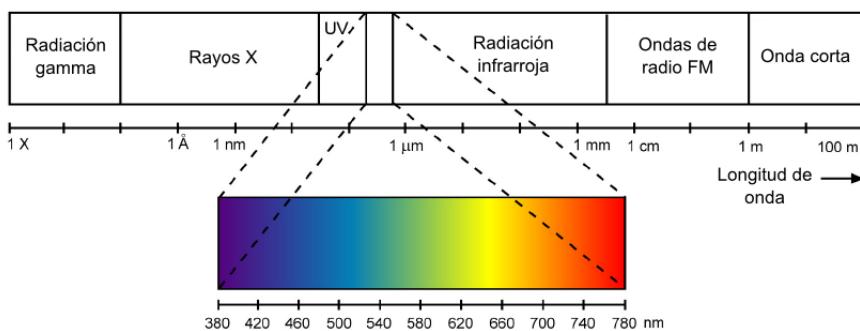


Figura 1.1 Representación de la porción de luz visible en el espectro electromagnético. Los colores percibidos por radiaciones con una única longitud de onda se denominan colores espectrales [Ste09].

La primera pregunta fundamental que debemos responder en esta sección es ¿qué es el color? Una respuesta es la siguiente: “*Empezaremos diciendo que el color en sí no*

existe, no es una característica del objeto, es más bien una apreciación subjetiva nuestra. Por tanto, podemos definirlo como una sensación que se produce en respuesta a la estimulación del ojo y de sus mecanismos nerviosos, por la energía luminosa de ciertas longitudes de onda. El color es pues un hecho de la visión que resulta de las diferencias de percepciones del ojo a distintas longitudes de onda que componen lo que se denomina el espectro de luz visible" [Net]. El primero en darse cuenta que el color es una percepción subjetiva fue Isaac Newton en 1675: "De cierto los rayos, propiamente expresados, no tienen color;...el color únicamente existe en el ojo y en el cerebro" [Poy06].

La retina en el ojo humano tiene tres tipos de células fotoreceptoras que distinguen el color, los *conos*, y cada uno de estos tres tipos de célula tiene una respuesta distinta al espectro de luz visible con puntos críticos de sensibilidad en tres zonas de longitud de onda: *corta* (420–440 nm) la zona donde percibimos el estímulo del color azul; *media* (530–540 nm), la zona donde percibimos el estímulo del color verde; *larga* (560–580 nm), la zona donde percibimos el estímulo del color rojo. En la figura 1.2 se muestra un diagrama donde se aprecian las zonas de sensibilidad para cada uno de los tres tipos de conos en el ojo humano, a tales tipos denominaremos cono C, cono M y cono L. Un trabajo detallado sobre la sensibilidad espectral de los conos en el ojo humano se puede encontrar en [SMJ93].

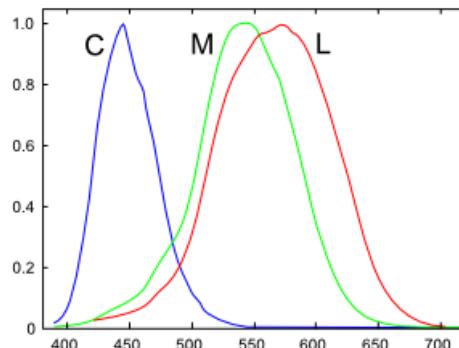


Figura 1.2 Diagrama normalizado de la respuesta espectral de los tres tipos de cono en el ojo humano en sus respectivas zonas de longitud de onda: corta, mediana y larga [Eze07].

Ahora bien, ya que existen exactamente tres tipos de células fotoreceptoras que distinguen el color, entonces sólo se requieren tres componentes numéricas para representarlo, por lo cual, un espacio que describa cómo el color es percibido por la visión humana será un espacio tridimensional y le llamaremos *espacio de color*. En un espacio de color se definen tres estímulos de color fundamentales que denominaremos primarios, cuya intensidad varía de 0 a algún máximo, que denotaremos como 1. Podemos visualizar a los colores como puntos en dicho espacio, o bien como vectores centrados en el origen. Al esquema de generación de color por medio de haces luminosos se le denomina *aditivo*

ya que, puesto que las ondas electromagnéticas obedecen al principio físico de la superposición, entonces la suma vectorial de los primarios en distintas intensidades da lugar a nuevos estímulos de color. La intensidad de un estímulo luminoso está determinada por la *amplitud* de la onda en el sentido físico, véase la figura 1.3.

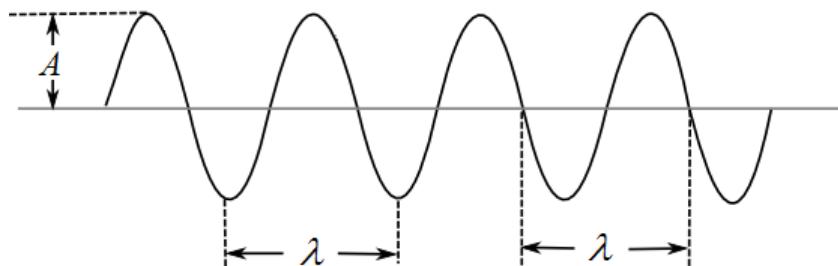


Figura 1.3 En este gráfico A representa la amplitud de onda. La longitud de onda λ nos da el color espectral.

1.2. Modelos de sistemas de color: CIE RGB y CIE XYZ

Uno de los primeros espacios de color determinado matemáticamente fue creado en 1931 por la Comisión Internacional de Iluminación (CIE, *Commission Internationale de l'Éclairage*). La primera etapa para construir este espacio consistió en realizar una serie de experimentos visuales llevados a cabo de manera independiente por William David Wright y John Guild. Uno de los experimentos consistía en poner a un observador frente a dos muestras de color distintas, generadas por fuentes luminosas independientes, una de ellas era un color espectral puro (una radiación con una única longitud de onda) y la otra era generada por tres haces luminosos primarios, véase la figura 1.4. Pero, ¿cuáles eran los primarios utilizados? Veamos: “*Los tres primarios elegidos para el experimento tenían espectros compuestos solamente por una sola longitud de onda. Aquellas longitudes de onda fueron 700 nm, 546.1 nm y 435.8 nm. Los dos últimos se eligieron ya que se podían generar con precisión a partir de las emisiones de una lámpara de mercurio de arco. El primero de ellos no podía ser generado directamente de manera precisa en aquel entonces, pero fue elegido porque la respuesta del ojo en esa región espectral no requería que este primario se generase con gran precisión*” [Ker10]. A esta combinación de primarios se le denomina *triestímulo RGB* (siglas en inglés para rojo, verde y azul). El observador tenía que igualar la muestra triestimular *RGB* con la muestra de color espectral pura, esto lo lograba al ir variando la intensidad de cada uno de los tres haces de color primario. Así pues, se determinaba la proporción de intensidades de los tres estímulos primarios necesaria para igualar un color espectral con una intensidad y

longitud de onda determinada. En adelante nos referiremos a los primarios con la notación R, G y B, mientras que para referirnos a las variables utilizaremos la notación en cursivas R , G y B .

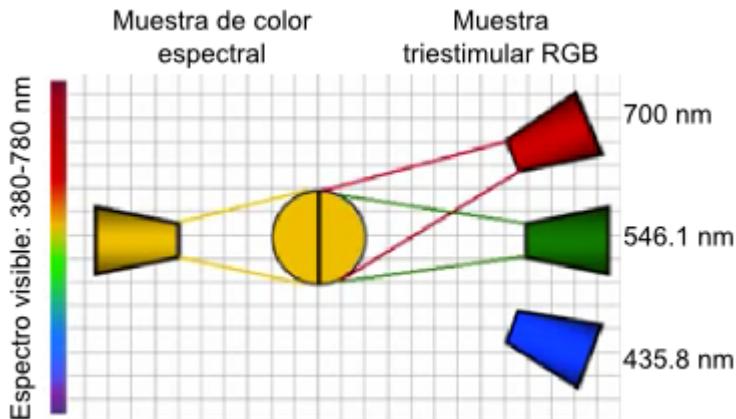


Figura 1.4 Experimento para determinar los valores de intensidad de cada uno de los tres estímulos primarios para igualar un color espectral [Bla08].

Ahora bien, resulta que no se podían reproducir todos los colores espectrales sin tener la necesidad de introducir valores triestimulares *RGB* con alguna componente negativa. Esta necesidad surgía al tener que mezclar la muestra de color espectral con alguno de los haces primarios, para entonces poder igualar esta muestra modificada con los otros dos primarios. Al anotar la medición, se ponía el valor de la intensidad del haz primario que se combinó con el color espectral con signo negativo, véase la figura 1.5. Para mayores detalles consultese [Gui32] y [Wri29].

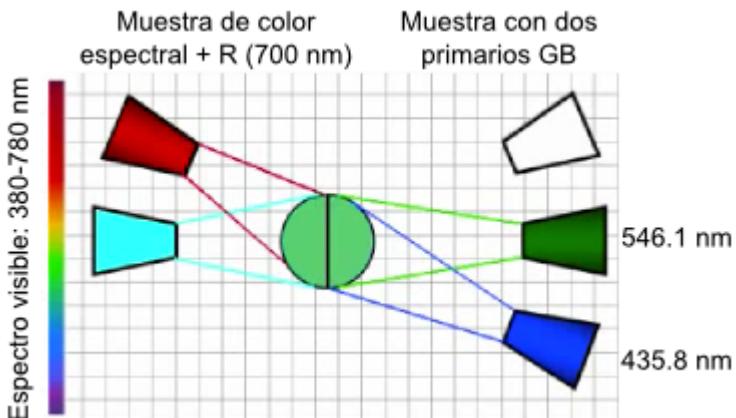


Figura 1.5 Experimento de Wright y Guild que requería el manejo de valores negativos [Bla08].

La información recabada en estos experimentos fue presentada en forma de tres curvas, una por cada primario, que mostraban la intensidad necesaria de cada primario para hacer la mezcla que igualara a un colorpectral de longitud de onda e intensidad determinada. Cada una de estas curvas es entonces una función de la longitud de onda λ , y se les llamó *funciones de emparejamiento* (*matching functions*), véase la figura 1.6. Es importante señalar que estas funciones se comportan de manera muy similar a la respuesta espectral de los conos pero no son exactamente lo mismo, en el primer caso no tenemos valores negativos y en el segundo caso sí existe dicha posibilidad.

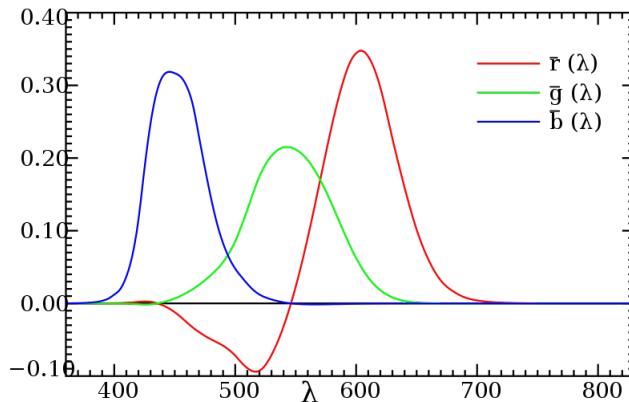


Figura 1.6 Funciones de emparejamiento de los colores espetrales [Wik14a].

Con las funciones de emparejamiento se puede definir una función vectorial

$$\vec{F}(\lambda) = \bar{r}(\lambda) \mathbf{R} + \bar{g}(\lambda) \mathbf{G} + \bar{b}(\lambda) \mathbf{B},$$

que define una trayectoria en \mathbb{R}^3 con parámetro λ . Al conjunto de puntos que forman dicha trayectoria se le denomina el *lugar espectral*. En la figura 1.7 se muestran dos vistas distintas del lugar espectral en \mathbb{R}^3 .

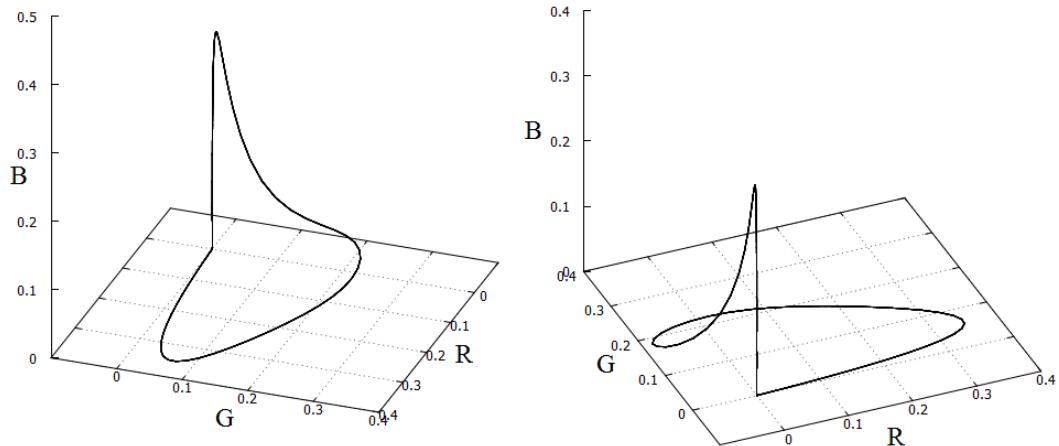


Figura 1.7 El lugar espectral es una trayectoria que sale del octante de los triestímulos positivos.

Como ya hemos mencionado, en los espacios de color aditivos los estímulos luminosos se pueden superponer para formar nuevos colores, entonces todos los colores que son perceptibles por la visión humana se pueden interpretar como una combinación lineal de los primarios R, G y B. Este conjunto de colores es un sólido tridimensional que recibe el nombre de *gamut* de la visión humana, delimitado por la superficie que se genera por las curvas que representan lugares espectrales a distintas intensidades (Figura 1.8), y generado por la suma convexa de puntos tomados en dicha superficie. A este espacio se le denominó CIE RGB.

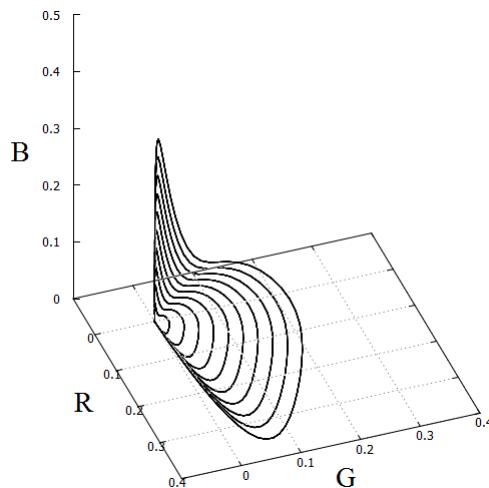


Figura 1.8 La superficie generada por curvas que representan lugares espectrales a distintas intensidades forma parte de la frontera del espacio CIE RGB.

Definición 1.1. La *suma de Minkowski* de dos conjuntos de vectores A y B en un espacio Euclíadiano se forma al sumar a cada vector de A con cada vector de B , esto es

$$A + B = \{a + b \mid a \in A, b \in B\}$$

Cabe señalar que al utilizar tres haces luminosos primarios para generar color en el mundo físico, como en los monitores de las computadoras cuyos pixeles tienen tres componentes (rojo, verde y azul), el gamut de colores generados corresponde a un cubo sólido contenido propiamente en el espacio CIE RGB, esto es, el volumen delimitado por la suma de Minkowski de los primarios R, G y B $\{(1,0,0), (0,1,0), (0,0,1)\}$ (la intensidad luminosa máxima está dada por 1, mientras que el 0 representa intensidad luminosa nula), véase la figura 1.9. Esto se debe a que sólo se pueden generar físicamente intensidades luminosas con valor no negativo.

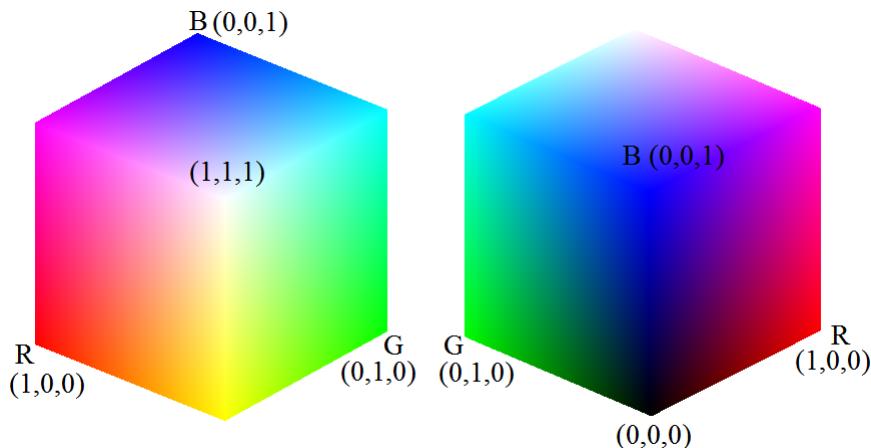


Figura 1.9 El cubo RGB representa el gamut de colores que pueden ser generados físicamente mediante la combinación de distintas intensidades de tres primarios luminosos, R, G y B. El blanco es el punto $(1, 1, 1)$, el negro es el punto $(0, 0, 0)$.

Como acabamos de ver, el espacio CIE RGB requiere valores negativos en algunos triestímulos RGB , sin embargo, la Comisión Internacional de Iluminación consideró que no era apropiado utilizar un sistema de color con valores negativos como estándar internacional, así que se definieron tres nuevos primarios X, Y y Z para que el gamut de la visión humana pudiera ser definido con triestímulos con coordenadas no negativas. La suma de Minkowski de estos tres primarios tendría que ser un poliedro que contuviera propiamente al espacio CIE RGB, como lo indica la figura 1.10, para lo cual se podía escoger una infinidad de ternas de primarios que cumplieran esta condición, sin embargo, se tomaron en cuenta también otras propiedades para escoger la terna de primarios, como por ejemplo, que el valor de la coordenada Y represente la luminancia de un triestímulo XYZ (esto se explicará más adelante). Estos primarios no corresponden a colores

reales sino que se trata más bien de un artificio matemático, pues tales primarios representados como vectores en el espacio CIE RGB tienen valores negativos en algunas de sus entradas, y puesto que no se pueden reproducir físicamente intensidades negativas de luz, quedan únicamente como entes imaginarios.

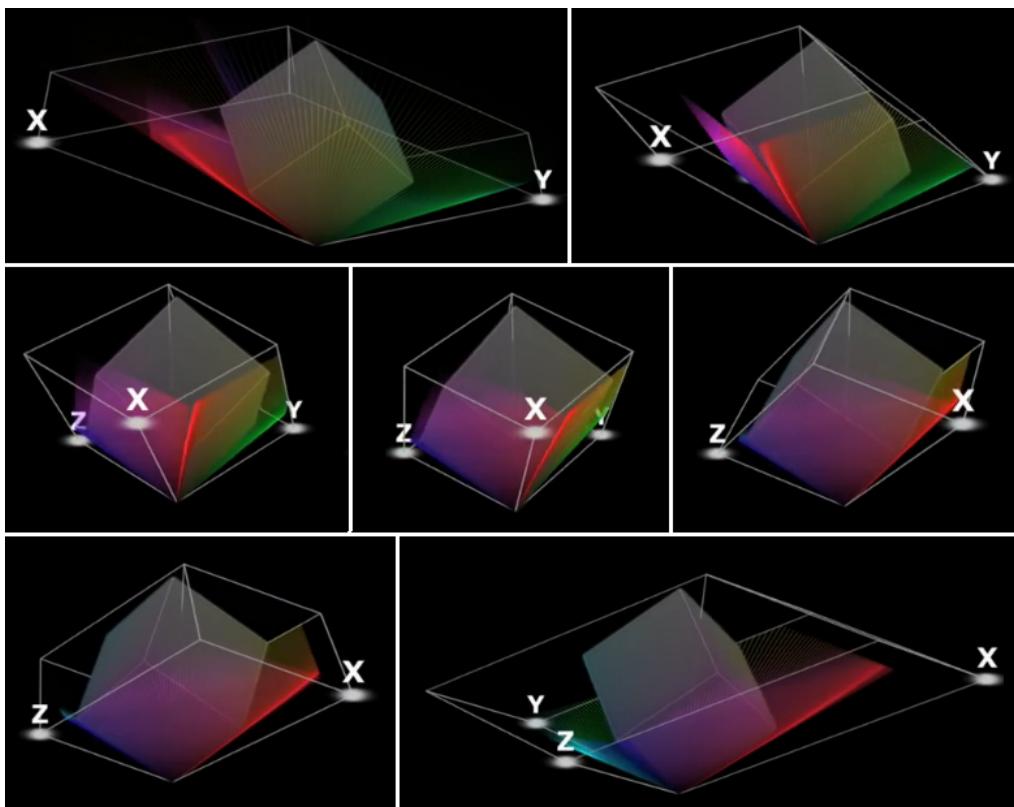


Figura 1.10 En este gráfico apreciamos el cubo unitario del espacio CIE RGB, la frontera del espacio CIE RGB generada por los colores espectrales y los vectores primarios X, Y y Z cuya suma de Minkowski define un volumen que contiene a todo el espacio CIE RGB [Sel13].

Ahora bien, se ha comprobado que el ojo humano responde mejor a ciertas longitudes de onda que a otras, y la mayor eficacia de percepción se encuentra en el rango de 530-540 nm, alcanzando su máximo en $\lambda = 555$. Esto quiere decir que los colores que el humano percibe como más brillantes se encuentran en dicha zona del espectro, esto se ve reflejado en la función definida por CIE que describe la variación del brillo percibido en función de la longitud de onda, ya que es similar a la respuestapectral de los conos M, como lo ilustra la figura 1.11.

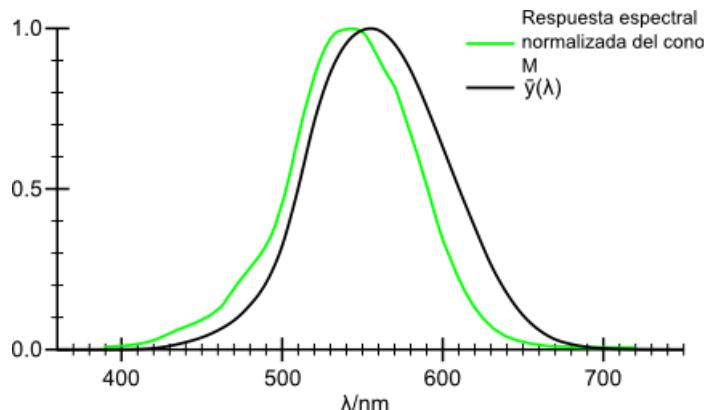


Figura 1.11 La función de luminosidad definida por CIE $\bar{y}(\lambda)$ tiene un comportamiento muy similar a la respuesta espectral del cono que es mayormente sensible al estímulo de la luz verde [Wik14a].

A partir de esta observación se definió la luminosidad en función de los valores de las coordenadas de un triestímulo RGB [Ker10], como una suma convexa que otorga a la coordenada G el mayor peso:

$$L(R, G, B) = 0.212671R + 0.715160G + 0.072169B$$

Se puede observar entonces que las contribuciones a la luminosidad de un color (R, G, B) son aproximadamente de 72 % por parte de su componente de luz verde, 21 % por parte de su componente de luz roja y 7 % por parte de su componente de luz azul ($R, G, B \in [0, 1]$). Así pues, el color blanco $(1, 1, 1)$ tiene la máxima intensidad luminosa.

Ahora bien, al definir la transformación lineal $T_1 : CIE RGB \rightarrow CIE XYZ$ se determinó que los coeficientes que aparecen en $L(R, G, B)$ fueran también las entradas del segundo renglón de la matriz asociada con dicha transformación, con lo cual se logra que $Y = L(R, G, B)$. De este modo el gamut de la visión humana queda definido únicamente con triestímulos no negativos y la coordenada Y corresponde precisamente a la luminosidad del color (R, G, B) . La transformación lineal es la siguiente

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix},$$

y su transformación inversa es

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.240479 & -1.537150 & -0.498535 \\ -0.969256 & 1.875992 & 0.041556 \\ 0.055648 & -0.204043 & 1.057311 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}.$$

Así nació el primer modelo matemático formal para describir el color, el modelo CIE XYZ de 1931 que representa todas las sensaciones de color que una persona promedio puede experimentar, el gamut de la visión humana. En la figura 1.12 se muestra el gamut de la visión humana en el espacio CIE XYZ. Nótese como todos los colores están en el primer octante con coordenadas positivas. El lugar espectral se encuentra en la frontera de dicho sólido que asemeja a un cono. En la figura 1.13 se ilustra la imagen del cubo bajo la transformación T_1 .

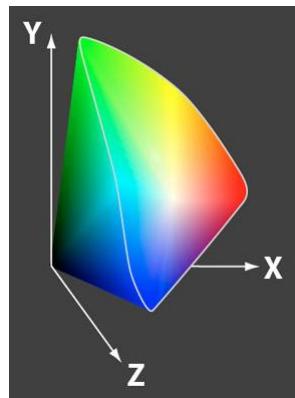


Figura 1.12 El sólido que representa el gamut de la visión humana en el espacio CIE XYZ. Este sólido se genera al aplicar la transformación: $T_1 : \text{CIE RGB} \rightarrow \text{CIE XYZ}$ [Inc09].

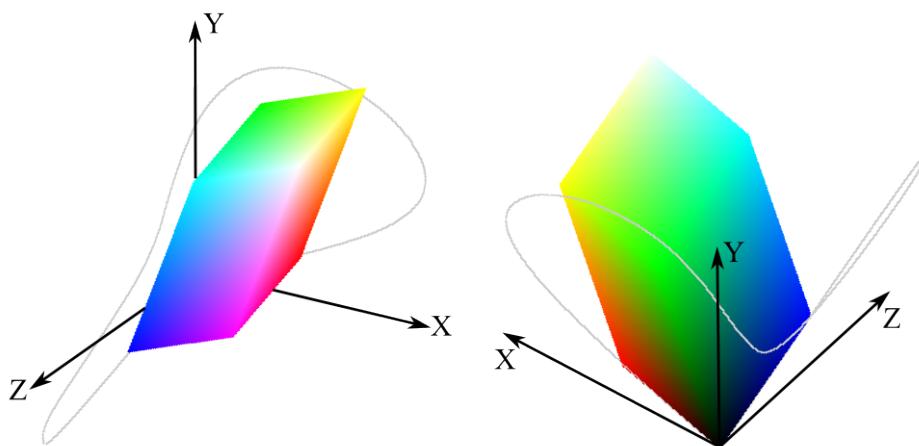


Figura 1.13 El cubo RGB en el espacio CIE XYZ y la curva del lugar espectral.

1.3. Diagrama de cromaticidad

El concepto del color puede ser dividido básicamente en dos partes: *cromaticidad* y *luminancia*.

Definición 1.2. La *luminancia* es una medida de la intensidad luminosa por unidad de área de luz que viaja en cierta dirección. Describe la cantidad de luz que atraviesa o es emitida por cierta área. Su unidad en el Sistema Internacional es la *candela*.

Definición 1.3. La *cromaticidad* es una especificación objetiva de la calidad del color independientemente de su luminosidad. Consiste en dos parámetros independientes, comúnmente especificados como *hue* y *saturación*.

El concepto de luminancia es intuitivamente claro, pero el concepto de cromaticidad tal vez necesite un poco más de trabajo para ser entendido. El valor de *hue* se refiere a la gama de color a la que pertenece el color en cuestión, es decir, a la gama de los azules, los violetas, los rojos, los verdes, los amarillos, etc. El valor de *saturación* se refiere a si el color es opaco (poco saturado) o intenso (muy saturado). Una analogía que ayuda a entender este último parámetro es la siguiente: una prenda de vestir nueva se dice que tiene un color saturado, pero con el paso del tiempo y el uso el color se va desgastando volviéndose cenizo y opaco, entonces se dice que tiene un color poco saturado, pero el hue del color no ha cambiado, si era roja la prenda cuando era nueva, seguirá siendo roja aún después de volverse opaca por el uso.

Para describir la cromaticidad de un color se derivaron dos parámetros *x* y *y* como función de sus tres valores triestimulares *X, Y, Z*:

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}.$$

Tales parámetros representan la proyección de un color con valor triestimular *X, Y, Z* sobre el plano definido por las coordenadas de los primarios X, Y y Z, como lo ilustra la figura 1.14. Con estos parámetros se elaboró un espacio de color con dos coordenadas para describir la cromaticidad *x, y*, y una tercera para describir la intensidad luminosa *Y*, véase la figura 1.15. Este último espacio de color, el CIE xyY, es utilizado ampliamente en la práctica para especificar colores, ya que la manera de hacerlo resulta más natural e intuitiva para el usuario, al seleccionar en un plano la cromaticidad deseada y la intensidad luminosa manejarla como un parámetro independiente en una tercera coordenada.

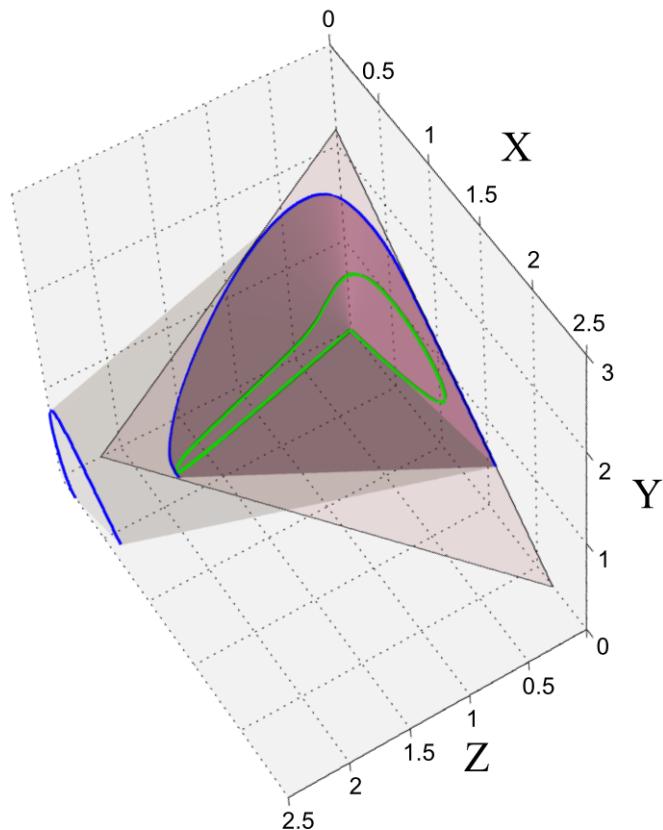


Figura 1.14 En el gráfico se representa la manera de obtener el diagrama de cromaticidad xy [Ado08].

En la figura 1.15 se muestra a la izquierda el espacio tridimensional CIE xyY, y a la derecha su proyección, es decir, el *diagrama de cromaticidad xy*, donde la frontera es una curva que representa a los colores espectrales con sus longitudes de onda especificadas en nanómetros. El segmento de línea recta que está en la parte baja, llamada la línea de los púrpuras, aunque está en la frontera del gamut no representa colores espectrales, es decir, tales colores no pueden ser generados por una radiación electromagnética con una única longitud de onda, sino que se forman mediante la combinación convexa de rojo con azul.

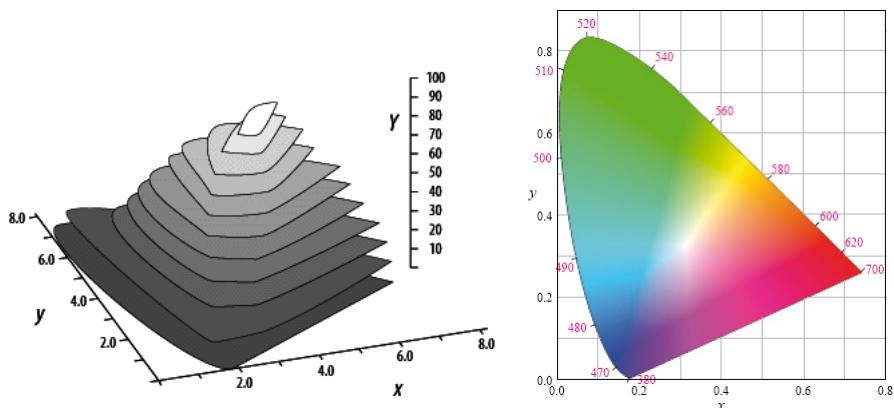


Figura 1.15 El diagrama de cromaticidad xy [Ste09], a la derecha, es una proyección del sólido que representa todo el espacio de color CIE xyY [Ado00c], a la izquierda.

En resumen, el diagrama de cromaticidad xy ilustra algunas propiedades interesantes del espacio de color CIE XYZ.

- El diagrama representa todas las cromaticidades visibles para una persona promedio.
- Si se escogen dos puntos dentro del diagrama de cromaticidad, todos los colores que están en el segmento de línea recta que une a tales puntos pueden ser generados por la combinación convexa de estos dos colores.
- Se puede notar que dados tres colores reales, que se pueden generar físicamente y que corresponderían a tres puntos dentro del diagrama de cromaticidad xy , no es posible generar con ellos todos los colores del diagrama de cromaticidad xy . En términos geométricos, no existe un triángulo dentro del diagrama de cromaticidad xy que contenga a todos los puntos del mismo.

1.4. Espacios de color perceptualmente uniformes

Ahora bien, en el estudio de la percepción del color surge de manera natural la pregunta ¿qué tan diferentes son dos colores? Para responder a esta pregunta, David Mac Adam [Ada42] condujo una serie de experimentos en 1942 para determinar qué tanto se podía mover la posición de un color dado, en distintas direcciones dentro el diagrama de cromaticidad xy , de modo que el color de la nueva posición no se percibiera como distinto al color de referencia. Los datos recolectados en sus experimentos revelaron que dado un color de referencia en el diagrama de cromaticidad xy existe una elipse centrada en dicho punto tal que los colores dentro de ella no se perciben distintos con respecto

al color de referencia. A estas elipses se les conoce como *elipses de Mac Adam*, véase figura 1.16.

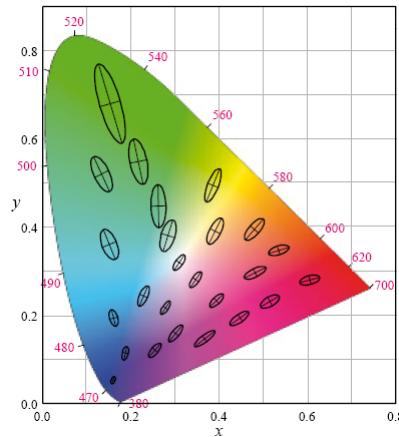


Figura 1.16 Las elipses de Mac Adam delimitan los colores que son indistinguibles para un observador promedio con respecto al color de referencia ubicado al centro de la ellipse [Ste09].

CIE Yuv

Esta última propiedad del espacio de color CIE Yxy le ganó la clasificación de ser un espacio *perceptualmente no uniforme*. Esto quiere decir que distancias euclidianas iguales entre parejas de colores no se perciben en general como la misma diferencia visual. Para solucionar este inconveniente se propuso modificar el espacio de color CIE Yxy logrando con ello que presentara cierta uniformidad al percibir diferencias entre colores. Para ello se transformaron las coordenadas x , y en un nuevas coordenadas u , v , que permitían una descripción más uniforme de las diferencias perceptuales en el nuevo espacio de color. Surgió así en 1960 el espacio de color CIE Yuv, con su respectivo diagrama de cromaticidad uv , ilustrado en la figura 1.17. La coordenada Y se mantiene pero las transformaciones correspondientes de las coordenadas x y y son las siguientes

$$u = \frac{4x}{12y - 2x + 3}$$

$$v = \frac{6y}{12y - 2x + 3}$$

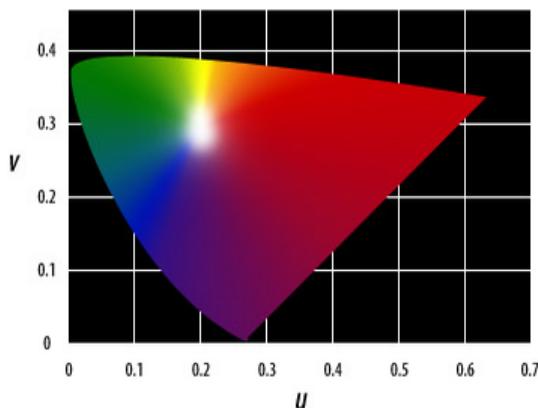


Figura 1.17 Diagrama de cromaticidad CIE uv de 1960.

CIE Yu'v'

En 1975 la CIE propuso modificar los valores de u y v , reemplazándolos por nuevas coordenadas $u' = u$ y $v' = 1.5v$, para mejorar aún más la uniformidad en la percepción. El resultado fue el espacio de color CIE $Yu'v'$ de 1976, véase la figura 1.18. Las transformaciones quedan de la siguiente manera

$$u' = u = \frac{4x}{12y - 2x + 3} = \frac{4X}{X + 15Y + 3Z}$$

$$v' = 1.5v = \frac{9y}{12y - 2x + 3} = \frac{9Y}{X + 15Y + 3Z},$$

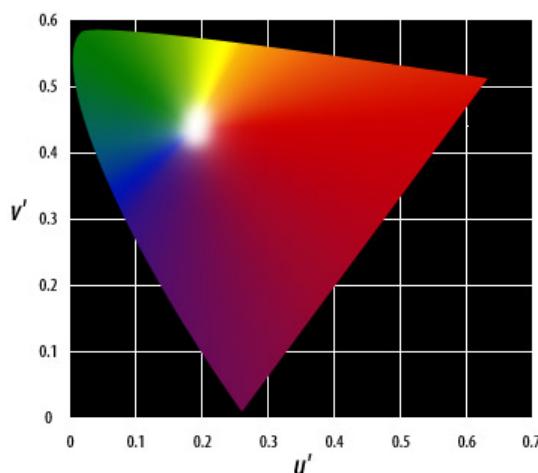


Figura 1.18 Diagrama de cromaticidad CIE u'v' de 1976.

Mientras que el resultado de uniformidad no es perfecto en este último espacio, la mejora es sustancial con respecto al espacio CIE XYZ. Esto se puede apreciar en los diagramas de cromaticidad respectivos ilustrados en la figura 1.19. En ambos casos los segmentos de línea representan la misma medida de diferencia perceptual, pero es claro notar que la distancia euclídea varía, y que estas variaciones son notablemente mayores en el primer diagrama.

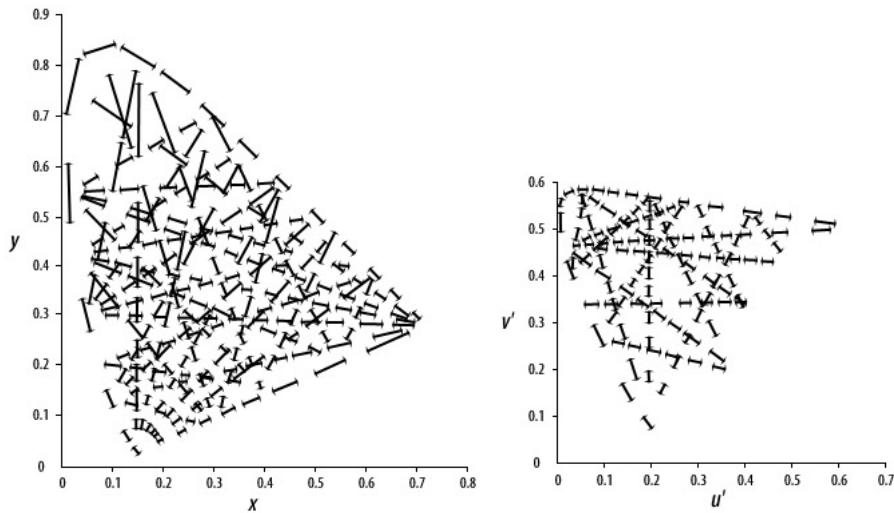


Figura 1.19 Comparación de la uniformidad en dos diagramas de cromaticidad xy y $u'v'$.

CIE $L^*u^*v^*$

Una última modificación que se hizo al espacio de color fue el reemplazar la coordenada de luminosidad Y con una nueva coordenada L^* . Este cambio fue con la intención de imitar la respuesta logarítmica del ojo a los cambios de intensidad luminosa. Las transformaciones y sus respectivas inversas se describen a continuación.

$$L^* = \begin{cases} 116Y^{1/3} - 16 & \text{si } Y > 0.008856 \\ 903.3Y & \text{si } Y \leq 0.008856 \end{cases}$$

$$u^* = 13(L^*)(u' - u'_n)$$

$$v^* = 13(L^*)(v' - v'_n)$$

Las transformaciones del espacio CIE $L^*u^*v^*$ al espacio CIE XYZ son

$$Y = \begin{cases} \left(\frac{L+16}{116}\right)^3 & \text{si } L \leq 7.999 \\ \frac{L^*}{903.3} & \text{si } L > 7.999 \end{cases}$$

$$X = \frac{9Y(u^* + 13L^*u'_n)}{4(v^* + 13L^*v'_n)}$$

$$Z = \frac{52L^*X}{3(u^* + 13L^*u'_n)} - \frac{15Y + X}{3}$$

Los valores u'_n y v'_n son las imágenes correspondientes, bajo las transformaciones descritas, del punto blanco D65 en el diagrama de cromaticidad xy con coordenadas $x_n = 0.3127$, $y_n = 0.3290$. Más información sobre la definición y significado del punto blanco D65 se puede encontrar en [Poy95]. Las imágenes de las figuras 1.17, 1.18 y 1.19, así como la información relacionada con ellas, se encuentra en [Ado00b].

1.5. Color de 24 bits y corrección gamma

Una herramienta indispensable para lograr los objetivos del presente trabajo será la computadora, tanto para la implementación de los algoritmos en un lenguaje de programación, como para el despliegue de los resultados gráficos, por lo cual resulta importante entender la manera en que se interpreta el color en una computadora. En primer lugar debemos mencionar que el espacio de color que utilizan las computadoras para interpretar el color es el cubo RGB, descrito en la sección 1.2. La razón básica que justifica el uso de este modelo de color es el hecho de que sólo se pueden generar físicamente intensidades luminosas con valores positivos.

La manera en que se logra generar triestímulos luminosos en un monitor contemporáneo es a través de pequeñas unidades denominadas *pixeles* (acrónimo en inglés de *picture element*) que consisten en un arreglo de tres diodos LED (*Light Emission Diode*), correspondientes a los tres colores primarios: rojo, verde y azul; esto se ilustra en la figura 1.20. Ahora, puesto que la unidad básica de memoria en una computadora es 1 byte, un número de 8 bits (cifras) en sistema binario, una manera natural de representar el color en un monitor a base de pixeles es asignar 3 bytes a cada pixel, un byte por cada coordenada R , G y B , para definir una combinación de intensidades luminosas de los primarios R,G,B. Se tienen entonces 24 bits para representar un color.

Ahora bien, en el sistema binario se pueden generar los primeros 2^8 enteros (contando el cero) utilizando un número de ocho cifras, entonces, se dispone del rango de valores enteros $[0, 255]$ para definir intensidades luminosas en cada uno de los diodos LED del pixel (0 representa intensidad luminosa nula, 255 representa intensidad luminosa máxima). Por tanto, se pueden representar un total de $256^3 = 16,777,216$ colores distintos de

este modo. Es necesario reescalar el rango de valores $[0, 255]$, mediante el mapeo biyectivo natural, al intervalo $[0, 1]$ para apegarnos al modelo estándar del cubo RGB donde los primarios son los vectores unitarios.

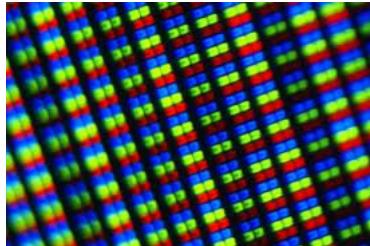


Figura 1.20 Pixeles en un monitor contemporáneo.

La respuesta del ojo a la variación de la luminosidad no es lineal, es decir, la eficacia del ojo para distinguir intensidades luminosas varía dependiendo de la zona donde se tomen valores así que es necesario aplicar una transformación no lineal para reescalar el rango de valores $[0, 1]$ y corregir esta situación para que los colores se perciban en el monitor de manera correcta. A esto se le denomina *corrección gamma*, y consiste básicamente en aplicar una función potencia con parámetro γ . La transformación está dada por las siguientes fórmulas:

$$R = \frac{r}{255}, G = \frac{g}{255}, B = \frac{b}{255}$$

$$R_{lin} = f(R), G_{lin} = f(G), B_{lin} = f(B)$$

donde

$$f(v) = \begin{cases} \frac{v}{12.92} & \text{si } v \leq 0.04045 \\ \left(\frac{v+0.055}{1.055}\right)^{2.4} & \text{si } v > 0.04045 \end{cases}$$

para una terna de valores (r, g, b) que describen un color en la computadora con valores enteros en $[0, 255]$. La transformación inversa está dada por

$$R = f^{-1}(R_{lin}), G = f^{-1}(G_{lin}), B = f^{-1}(B_{lin})$$

$$r = 255R, g = 255G, b = 255B$$

donde

$$f^{-1}(w) = \begin{cases} 12.92w & \text{si } w \leq 0.0031308 \\ 1.055w^{1/2.4} - 0.055 & \text{si } w > 0.0031308 \end{cases}.$$

Hasta aquí hemos explorado un panorama general de la teoría del color, que explica la manera en que percibimos los estímulos luminosos que nuestro cerebro interpreta como colores, así como un modelo matemático que será la base para resolver el problema de optimización del que trata el presente trabajo, a saber, la manera de diseñar una paleta de colores bien diferenciados. En los capítulos subsiguientes se explorarán las ideas y conceptos necesarios para el correcto planteamiento del problema correspondiente, por lo que recomendamos al lector regresar a esta sección para consultar los conceptos sobre los espacios de color que volverán a ser mencionados en el capítulo siete al plantear el problema de optimización combinatoria en el espacio de color CIE L*u*v*.

Capítulo 2

Justificación para el empleo de métodos heurísticos

2.1. Un ejemplo de optimización convexa

En esta sección daremos una justificación de la utilización de métodos heurísticos en nuestro problema de optimización. Abordaremos primeramente el problema de colocar n puntos en el cubo unitario $\mathcal{C} = \{(x, y, z) \in \mathbb{R}^3 \mid 0 \leq x, y, z \leq 1\}$, de tal modo que la distancia mínima entre tales puntos sea lo más grande posible. Como función natural para medir la distancia entre dos puntos utilizaremos la métrica euclíadiana $d : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}$. Sea

$$S_n = \{p_1, \dots, p_n \mid p_i = (x_i, y_i, z_i); x_i, y_i, z_i \in [0, 1]\}$$

$$f(S_n) = \min_{p_i, p_j \in S_n} d(p_i, p_j).$$

El modelo de optimización para el cubo unitario queda de la siguiente manera:

$$\text{maximizar } f(S_n) \text{ sobre todos los posibles conjuntos } S_n \subset \mathcal{C}.$$

Una solución óptima global para este problema será el conjunto S_n^* que satisface

$$f(S_n^*) \geq f(S_n) \text{ para toda } S_n \subset \mathcal{C}.$$

La desigualdad de la expresión anterior no es estricta puesto que puede haber más de una solución óptima. A continuación daremos una solución analítica para el caso $n = 3$, lo cual será suficiente para hacer notorio que la resolución del problema para cualquier entero n no es trivial.

Si deseamos seleccionar $n = 2$ puntos en \mathcal{C} de modo que su distancia sea la mayor posible existen 4 soluciones triviales, se toman las parejas de vértices que conforman las diagonales principales del cubo unitario. Las parejas son: $(0,0,0), (1,1,1)$; $(1,0,1), (0,1,0)$; $(1,0,0), (0,1,1)$ y $(1,1,0), (0,0,1)$. La distancia entre estas parejas de puntos es lo que se conoce como el *diámetro* del cubo, el mínimo que una esfera puede tener para contenerlo, y que en este caso es $\sqrt{3}$. Tenemos entonces que una de las soluciones óptimas es $S_2^* = \{(0,0,0), (1,1,1)\}$, como lo indica la figura 2.1.

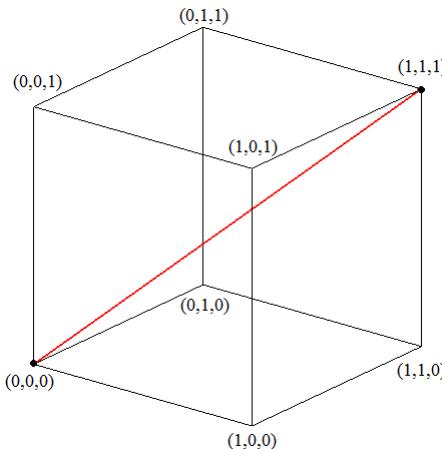


Figura 2.1 La diagonal principal que une a los vértices $(0,0,0)$ y $(1,1,1)$ es un segmento de recta con distancia máxima en \mathcal{C} para $n = 2$.

Ahora bien, para resolver el caso $n = 3$ una idea es utilizar a S_2^* como configuración de partida, procurando colocar un tercer punto en \mathcal{C} de manera que éste se encuentre lo más alejado posible de los puntos de S_2^* . Para lograr lo anterior, nos fijamos en el plano π ortogonal a la recta que cruza por $(0,0,0)$ y $(1,1,1)$ cuya ecuación es $x + y + z = \frac{3}{2}$. Este plano divide a \mathcal{C} en dos regiones convexas congruentes, entonces, por razones de simetría el tercer punto buscado debe estar en dicho plano. Este plano interseca en seis puntos medios de aristas del cubo, aquéllas que no son incidentes con S_2^* , formando así un hexágono inscrito en el cubo, véase la figura 2.2. Por el teorema de Pitágoras, podemos deducir que los puntos de $\pi \cap \mathcal{C}$ que se encuentran a la mayor distancia posible de los puntos $(0,0,0)$ y $(1,1,1)$ son los seis vértices del hexágono. Escogemos a $(0,1,0.5)$.

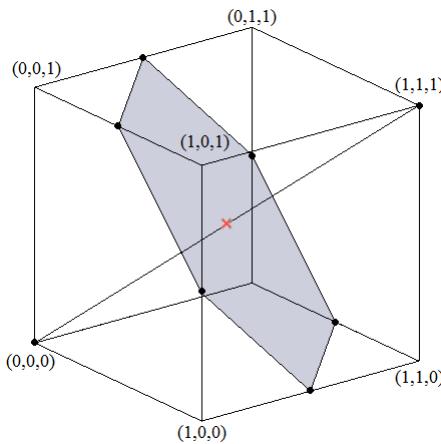


Figura 2.2 Los puntos medios de las aristas que no inciden en los puntos de S_2^ son equidistantes a $(0, 0, 0)$ y $(1, 1, 1)$.*

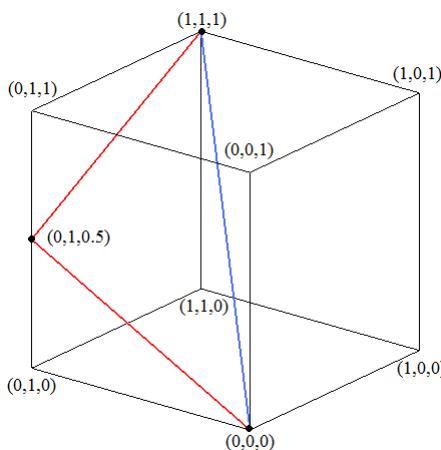


Figura 2.3 La distancia mínima en este conjunto de tres puntos tiene un valor de $\sqrt{\frac{5}{4}}$, la distancia entre $(0, 1, 0.5)$ y cualquiera de los puntos de S_2^ .*

Ahora, ¿será que la distancia mínima de esta configuración de tres puntos, figura 2.3, es la más grande posible en el cubo unitario? Obsérvese que al desplazar las posiciones de dos de los puntos, como lo indica la figura 2.4, la distancia mínima se va incrementando hasta llegar a una mejor configuración donde las distancias valen todas $\sqrt{2}$.

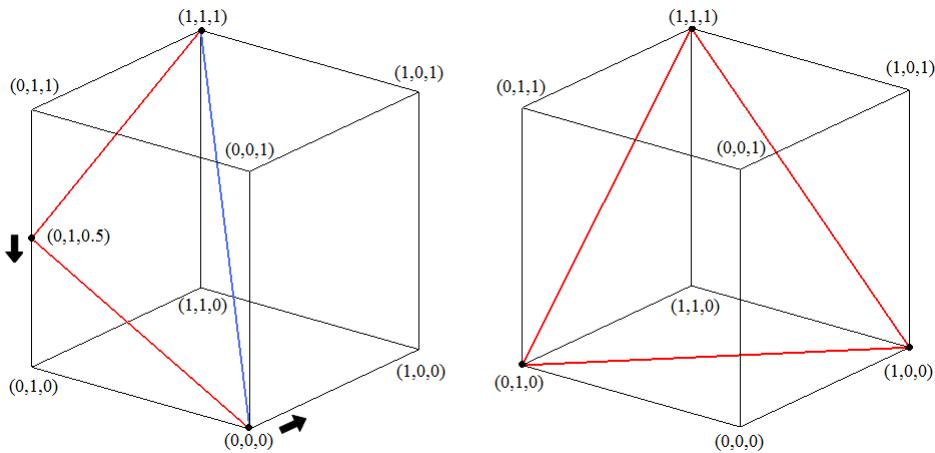


Figura 2.4 Al mover los vértices en la dirección indicada por las flechas se logra incrementar la distancia mínima.

Esta configuración es la que logra maximizar la distancia mínima en un conjunto de tres puntos en \mathcal{C} , es decir, no hay otra configuración de tres puntos cuya distancia mínima sea mayor que $\sqrt{2}$ en el cubo. Veamos la demostración de [Nic11], donde se propone acotar el producto de las distancias de un triángulo contenido en el cubo. Sean $p_1 = (x_1, y_1, z_1)$, $p_2 = (x_2, y_2, z_2)$ y $p_3 = (x_3, y_3, z_3)$ tres puntos en \mathcal{C} . Sea $\delta_{ij} = d(p_i, p_j)^2$, entonces

$$\delta_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2.$$

Por la desigualdad de la media aritmética-media geométrica tenemos que

$$\sqrt[3]{\delta_{12}\delta_{13}\delta_{23}} \leq \frac{\delta_{12} + \delta_{13} + \delta_{23}}{3}.$$

Nótese que al agrupar convenientemente los términos de $\delta_{12} + \delta_{13} + \delta_{23}$ se obtienen tres expresiones similares, una de las cuales es

$$(x_1 - x_2)^2 + (x_1 - x_3)^2 + (x_2 - x_3)^2,$$

las otras dos expresiones son análogas, sólo hay que intercambiar x por y y z . Al expandir y reducir la expresión anterior obtenemos

$$2x_1^2 + 2x_2^2 + 2x_3^2 - 2x_1x_2 - 2x_1x_3 - 2x_2x_3.$$

Ahora bien, podemos ver a la expresión anterior como una función $f : \mathbb{R}^3 \rightarrow \mathbb{R}$, a partir de la cual podemos construir otra función $F : \mathbb{R}^9 \rightarrow \mathbb{R}$ dada por

$$F(x_1, x_2, x_3, y_1, y_2, y_3, z_1, z_2, z_3) = f(x_1, x_2, x_3) + f(y_1, y_2, y_3) + f(z_1, z_2, z_3).$$

Para concluir la demostración, necesitamos encontrar el máximo de F dentro del cubo unitario, así que el problema a resolver es

$$\text{maximizar } F(x_1, x_2, x_3, y_1, y_2, y_3, z_1, z_2, z_3)$$

con las restricciones

$$0 \leq x_i \leq 1$$

$$0 \leq y_i \leq 1$$

$$0 \leq z_i \leq 1.$$

A este tipo de formulación se le denomina un problema de optimización con restricciones, y de manera más concreta, un problema de *optimización convexa*, pues está definido en un dominio convexo (el hipercubo de nueve dimensiones definido por las restricciones) y la función convexa F , es la función objetivo a maximizar.

Definición 2.1. Un conjunto S de un espacio Euclíadiano \mathbb{R}^n es *convexo* si para cualesquiera dos puntos de S el segmento de recta que los une está contenido en S .

Definición 2.2. Sea \mathbb{D} un dominio convexo. Una función real $f : \mathbb{D} \rightarrow \mathbb{R}$ es *convexa* si

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y),$$

para todo $x, y \in \mathbb{D}$.

Retomando el problema de optimización convexa de esta primera sección, y dada la simetría del mismo, se puede simplificar su resolución al enfocarnos en maximizar f en \mathcal{C} y multiplicar por tres el valor de su máximo debido a la manera en que se definió F . Utilizando técnicas propias de esta área de estudio, que involucran el uso de las condiciones de Kuhn-Tucker, se demuestra que el máximo de f en el cubo unitario es 2, luego se sigue que

$$\delta_{12} + \delta_{13} + \delta_{23} \leq 2 + 2 + 2,$$

y por lo tanto

$$\sqrt[3]{\delta_{12}\delta_{13}\delta_{23}} \leq \frac{\delta_{12} + \delta_{13} + \delta_{23}}{3} \leq \frac{6}{3} = 2,$$

$$\delta_{12}\delta_{13}\delta_{23} \leq 8,$$

$$\sqrt{\delta_{12}\delta_{13}\delta_{23}} \leq \sqrt{8} = 2\sqrt{2},$$

lo cual equivale a

$$d(p_1, p_2) d(p_1, p_3) d(p_2, p_3) \leq 2\sqrt{2}.$$

Concluimos entonces que no existe un triángulo inscrito en el cubo unitario cuya menor distancia sea estrictamente mayor que $\sqrt{2}$. Entonces, para $n = 3$ una de las soluciones óptimas es $S_3^* = \{(0, 1, 0), (1, 1, 1), (1, 0, 0)\}$.

Una solución para $n = 4$ se encuentra de manera inmediata a partir del resultado anterior, esto se logra al añadir un vértice a S_3^* , en este caso $(0, 0, 1)$, que se encuentra a distancia $\sqrt{2}$ de cualquiera de los puntos de S_3^* , como se muestra en la figura 2.5.

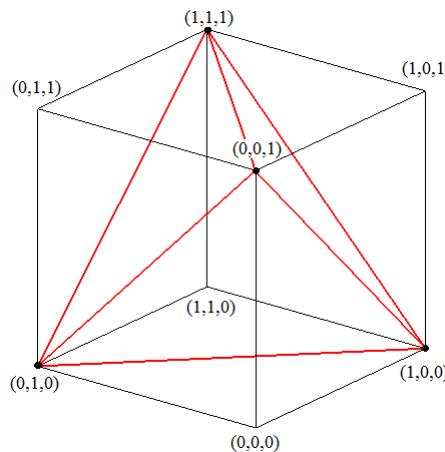


Figura 2.5 $S_4^* = \{(0, 1, 0), (1, 1, 1), (1, 0, 0), (0, 0, 1)\}$. La gráfica que representa las distancias entre los cuatro puntos es un tetraedro regular con longitud de arista $\sqrt{2}$.

Como hemos visto, encontrar la solución óptima para $n = 3$ no es tarea sencilla, involucra resolver un problema de optimización convexa en \mathbb{R}^9 , que en este caso se pudo simplificar a un problema en \mathbb{R}^3 por la simetría del planteamiento. En general, si se desea resolver el problema de colocar n puntos en el cubo unitario por medio de técnicas propias de la optimización convexa, se tiene que resolver un problema de $3n$ variables, donde cada variable representa una coordenada del hipercubo en $3n$ dimensiones, lo cual se vuelve evidentemente complejo para valores progresivamente más grandes de n . J. Dattorro expresa de manera sucinta la dificultad intrínseca de resolver un problema de este tipo: “*Es fácil minimizar una función convexa sobre cualquier subconjunto convexo de su dominio puesto que cualquier mínimo local será un mínimo global. Pero es difícil encontrar el máximo de una función convexa sobre un dominio convexo puesto que pueden existir muchos máximos locales que no son globales*” [Dat05]. Entonces, tratar de resolver este problema por medios analíticos no parece ser la mejor opción, por lo cual requerimos un enfoque distinto para resolverlo de manera aproximada.

El problema tratado en esta sección se le conoce, de manera más general, como el *problema de diversificación Máx-Min* (MMDP), el cual se plantea como la selección de un subconjunto de elementos de un conjunto dado, tal que la distancia mínima entre tales elementos sea máxima. La definición de distancia depende de cada aplicación específica. Ghosh demostró que este problema es *NP-duro* [Gho96], es decir, no se sabe si existe algún algoritmo que lo pueda resolver en tiempo polinomial. Otro nombre con el que se conoce a este problema es el de *problema de n-dispersión*, introducido por Chandrasekaran y Daughety [CD81], quienes proponen métodos heurísticos para resolver el caso especial de redes de árboles.

Se han utilizado distintas técnicas para atacar este problema, como el empleo de métodos heurísticos con dos etapas, donde la primera es constructiva y la segunda es de búsqueda local. Algunos autores han implementado el recocido simulado y la búsqueda tabú como metaheurísticas para resolver este problema [Kin92]. Otros han recurrido a técnicas híbridas utilizando una heurística como GRASP (Procedimientos de Búsqueda Glotona, Aleatoria y Adaptativa) con religado de trayectorias [RR07],[RMGD10]; y también se ha planteado este problema, en una versión todavía más general, como uno de programación lineal entera [JSW07].

En el método heurístico propuesto en este trabajo incorporamos dos etapas, la primera es un método constructivo para la obtención de una solución inicial, y la segunda etapa una metaheurística. A continuación revisaremos las ideas principales concernientes a este tipo de métodos.

Capítulo 3

Métodos heurísticos

3.1. Optimización Combinatoria

En términos generales, el objetivo de resolver un problema de optimización es encontrar un conjunto de valores adecuados para las variables del problema, con el fin de alcanzar alguna meta específica. Dicho conjunto de valores debe satisfacer ciertas restricciones, y cuando esto se logra, se obtiene lo que se denomina una *solución factible*. Comúnmente la meta es minimizar o maximizar el valor de una función a la que se denomina *función objetivo*.

Existen dos tipos de variables que se utilizan para formular un problema de optimización: variables continuas y variables discretas. Dependiendo de la naturaleza del problema a resolver, éste se puede plantear utilizando un solo tipo de variable, o bien una combinación de ambos tipos de variables.

Una clase de problemas que utilizan variables discretas para su planteamiento son los problemas de Optimización Combinatoria, donde el conjunto de soluciones factibles es un espacio discreto, es decir, un conjunto finito o a lo más infinito numerable. Una *solución óptima* en el espacio de soluciones, es una solución factible que satisface la meta del problema. De acuerdo con [AL97] “*la optimalidad está relacionada con algún criterio de costo que provee una medida cuantitativa de la calidad de cada solución*”. Una introducción más amplia a este tema de optimización combinatoria se puede encontrar en [PS82].

Una gran variedad de problemas de optimización combinatoria normalmente se encuentran en la categoría de problemas *NP-duros*, por lo cual los *algoritmos de aproximación* han cobrado gran interés para su resolución. Estos algoritmos se caracterizan por encontrar soluciones *cuasi-óptimas* (aproximadas a la solución óptima) cuya calidad está demostrada y cuyos tiempos de ejecución tienen cotas conocidas. Veamos ahora algunas definiciones y conceptos importantes, a la manera de [AL97], para el plantea-

miento de problemas de optimización combinatoria, no sin antes mencionar que, a lo largo de este trabajo, al referirnos a un problema de optimización combinatoria la meta será maximizar la función objetivo.

Definición 3.1. Una *instancia* de un problema de optimización combinatoria es una pareja (\mathcal{Y}, f) donde \mathcal{Y} es el conjunto de soluciones factibles y la función objetivo f es un mapeo $f : \mathcal{Y} \rightarrow \mathbb{R}$.

Definición 3.2. Sea (\mathcal{Y}, f) una instancia de un problema de optimización combinatoria. Una solución *óptima global* es una solución factible $i^* \in \mathcal{Y}$ tal que $f(i^*) \geq f(i)$ para toda $i \in \mathcal{Y}$.

Definición 3.3. Sea (\mathcal{Y}, f) una instancia de un problema de optimización combinatoria. Una *función de vecindad* es un mapeo $\mathcal{N} : \mathcal{Y} \rightarrow 2^{\mathcal{Y}}$, que define para cada solución $i \in \mathcal{Y}$ un conjunto $\mathcal{N}(i) \subseteq \mathcal{Y}$ de soluciones que se consideran cercanas a i . Al conjunto $\mathcal{N}(i)$ se le denomina la *vecindad* de la solución i , y cada $j \in \mathcal{N}(i)$ es un vecino de i .

La cercanía entre dos soluciones se puede definir de muchas maneras, y es importante señalar que este concepto no se relaciona necesariamente con la noción de cercanía derivada de una métrica. Por ejemplo, se puede definir una función de vecindad entre circuitos hamiltonianos de una gráfica, donde dos circuitos son vecinos entre sí si difieren a lo más en dos aristas.

Un *sistema de vecindades* es una colección $\{\mathcal{N}(i)\}_{i \in \mathcal{Y}}$ de subconjuntos del espacio de soluciones \mathcal{Y} , que satisfacen lo siguiente:

- $\forall i \in \mathcal{Y}, i \in \mathcal{N}(i)$
- $\forall i, j \in \mathcal{Y}, i \in \mathcal{N}(j) \Leftrightarrow j \in \mathcal{N}(i)$
- $\forall i, j \in \mathcal{Y}, |\mathcal{N}(i)| = |\mathcal{N}(j)|$
- $\forall i, j \in \mathcal{Y},$ existe una sucesión $\{u_1, \dots, u_m\}$ en \mathcal{Y} tal que

$$i \in \mathcal{N}(u_1), u_i \in \mathcal{N}(u_{i+1}), u_m \in \mathcal{N}(j).$$

Definición 3.4. Sea (\mathcal{Y}, f) una instancia de un problema de optimización combinatoria y sea \mathcal{N} una función de vecindad. Una solución \hat{i} es localmente óptima con respecto a \mathcal{N} si

$$f(\hat{i}) \geq f(j) \quad \text{para toda } j \in \mathcal{N}(\hat{i}).$$

En términos generales, los algoritmos de búsqueda local comienzan con una solución inicial, tratando de reemplazarla con alguna solución vecina si ésta última satisface algún criterio de aceptación. La versión más básica de búsqueda local es la *mejora sucesiva*.

Esta variante comienza con una solución inicial, luego busca entre las soluciones vecinas alguna que mejore el valor de la función objetivo (maximizando su valor). Si se encuentra tal solución, entonces ésta reemplaza a la solución inicial y la búsqueda continúa entre sus soluciones vecinas. De no encontrarse una mejor solución vecina que aquélla que está en curso, entonces se dice que ésta es localmente óptima.

3.2. Metaheurísticas

Durante las últimas cuatro décadas los algoritmos de aproximación, como el de búsqueda local, han cobrado gran interés e importancia tanto en la academia como en el mundo real para abordar problemas de tipo práctico. Lo que antes era un conjunto de técnicas empíricas para resolver problemas de optimización, sin tener una base teórica sólida, se ha convertido en un campo de investigación maduro y formal dentro de la optimización combinatoria. Glover en 1986 introdujo por primera vez el término *metaheurística* [Glo86] para referirse a un “método superpuesto sobre otra heurística”. Ésta palabra tiene su origen en la composición de dos palabras griegas: heurística que proviene del verbo *heuriskein* que significa “encontrar” y del prefijo *meta* que significa “más allá”, o bien “nivel superior”. Veamos algunas definiciones de este concepto.

- “*Metaheurísticas, en su definición original, son métodos de resolución que organizan una interacción entre procedimientos de mejora local y estrategias de alto nivel para crear un proceso capaz de escapar de óptimos locales y desempeñar una búsqueda robusta en un espacio de soluciones*” [GK03].
- “*Una metaheurística está formalmente definida como un proceso de generación iterativa que guía a una heurística subordinada al combinar de manera inteligente diferentes conceptos para explorar y explotar el espacio de búsqueda, estrategias de aprendizaje son utilizadas para estructurar información de manera que se puedan encontrar de manera eficiente soluciones cercanas al óptimo*” [OL96].
- “*Una metaheurística es un proceso maestro iterativo que guía y modifica las operaciones de heurísticas subordinadas para producir de manera eficiente soluciones de alta calidad. Puede ser que manipule una solución completa (o incompleta), o bien un conjunto de soluciones en cada iteración. Las heurísticas subordinadas pueden ser de alto (o bajo) nivel, búsqueda local o simplemente un método constructivo*” [VMOR99].

La filosofía detrás de las técnicas metaheurísticas puede ser descrita de manera sucinta, y por analogía, a través de la siguiente ilustración: “*Regularmente se cree que los humanos operan de acuerdo con algún tipo de elemento azaroso (probabilístico) que promueve cierto nivel de inconsistencia. La tendencia resultante de desviarse de un rumbo trazado, algunas veces lamentado como una fuente de error, puede también proveer una*

fuente de ganancia. Tal mecanismo se ha postulado como útil, e incluso fundamental, para el ingenio humano” [Glo86]. La similitud con las metaheurísticas se encuentra en el hecho de que las estrategias de *alto nivel* guían la búsqueda permitiendo en algunas etapas *desviarse* de lo que pareciera ser el mejor rumbo (mejora sucesiva) con el objetivo de escapar de óptimos locales, resultando así en ganancia el que exista entonces la posibilidad de encontrar un óptimo global.

Algunas de las técnicas metaheurísticas más populares y utilizadas incluyen: Búsqueda Tabú, Algoritmos Genéticos, Programación Genética, Colonias de Hormigas, Búsqueda de Entorno Variable, Búsqueda Local Guiada, Recocido Simulado, Búsqueda Local Iterada, Métodos Multi-Comienzo, Métodos de Redes Neuronales para Optimización, entre otras. Para una descripción detallada de éstas y otras técnicas véase [GK03].

De acuerdo con [BR03], las propiedades fundamentales que caracterizan a las metaheurísticas son:

- Las metaheurísticas son estrategias que “guían” el proceso de búsqueda.
- La meta es explorar de manera eficiente el espacio de soluciones (o espacio de búsqueda) para así encontrar soluciones quasi-óptimas.
- Las técnicas que constituyen los algoritmos metaheurísticos varían desde procedimientos simples de búsqueda local hasta complejos procesos con aprendizaje.
- Los algoritmos metaheurísticos son aproximados y usualmente no deterministas.
- Pueden incorporar mecanismos para evitar ser atrapados en áreas confinadas del espacio de búsqueda.
- Las metaheurísticas no son exclusivas de ciertos tipos de problemas.
- Las metaheurísticas pueden hacer uso de conocimiento específico del dominio en la forma de heurísticas que son controladas por una estrategia de alto nivel.
- Las metaheurísticas más avanzadas en la actualidad utilizan la búsqueda con experiencia previa, guardada en alguna forma de memoria, para guiar la búsqueda.

3.3. Recocido Simulado y Umbral de Aceptación

En 1983 apareció el artículo “Optimización por Recocido Simulado” [KGV83] en el cual sus autores, Kirkpatrick, Gelatt y Vecchi, exponen una técnica para aplicar métodos heurísticos en la búsqueda de soluciones óptimas, en problemas de optimización combinatoria, mediante la implementación de un proceso estocástico análogo al planteado en la mecánica estadística, el *recocido*. En este proceso se busca que un sólido fundido alcance su estado de mínima energía. Esto se logra al permitir que sus moléculas se

reacomoden mientras el material recibe un baño térmico que mantiene su temperatura cerca del punto de solidificación por un periodo prolongado, la circunstancia que permite a las moléculas tener cierta holgura para desplazarse, luego la temperatura se va disminuyendo lenta y progresivamente. El método de recocido simulado puede ser visto como una *caminata aleatoria controlada* en el espacio de soluciones factibles.

Este método fue descubierto también, de manera independiente, por Cerny [Cer85]. “*Este tipo de algoritmo ha cobrado gran interés puesto que ha demostrado ser bastante exitoso al ser aplicado en una gran variedad de problemas prácticos, porque existe una componente estocástica que facilita su estudio y el análisis de su convergencia asintótica, razón que le ha ganado popularidad entre la comunidad matemática*” [AKvL97].

En la figura 3.1 se da el seudocódigo del algoritmo de *umbral de aceptación*, el cual es una variante simplificada propuesta por Dueck y Scheuer [DS90] del algoritmo de recocido simulado. La diferencia principal entre éstos radica en que la aceptación de una solución vecina, en el algoritmo de recocido simulado, depende de una distribución de probabilidad, mientras que en el caso del algoritmo de umbral de aceptación, la aceptación de una solución vecina está dada por un criterio determinista. Este criterio consiste en que la diferencia de la función objetivo de la solución en curso y la solución vecina no sea mayor que cierto umbral.

En la iteración $k = 0$, el primer paso es seleccionar una solución inicial $i_0 \in \mathcal{Y}$. El siguiente paso consiste en generar una solución vecina $j \in \mathcal{N}(i_0)$. Si $f(i_0) - f(j) \leq u_k$ ($u_k \geq 0$ es el umbral correspondiente a la iteración $k = 0, 1, 2, \dots$), la solución vecina reemplaza a la solución en curso. Si no se satisface el criterio del umbral, entonces se continúa con la iteración $k + 1$ para escoger otra solución vecina. Cuando se logra un reemplazo con una solución vecina j la búsqueda prosigue, pero ahora en las soluciones vecinas de j . El algoritmo termina cuando se satisface una condición de paro.

```

Inicio :
Solución inicial  $i_0$ ;
 $i \leftarrow i_0$ ;
 $k \leftarrow 0$ ;

Repetir :
Sea  $j$  solución vecina de  $i$ ;
if  $f(i) - f(j) \leq u_k$  then  $i \leftarrow j$ ;
 $k \leftarrow k + 1$ 

Sair :
Cuando se satisface criterio de paro;

Fin;
```

Figura 3.1 Seudocódigo del algoritmo de umbral de aceptación.

- *Mejora sucesiva.* $u_k = 0, k = 0, 1, 2, \dots$ En esta variante únicamente se aceptan soluciones vecinas que mejoren la función objetivo. Véase la figura 3.2.

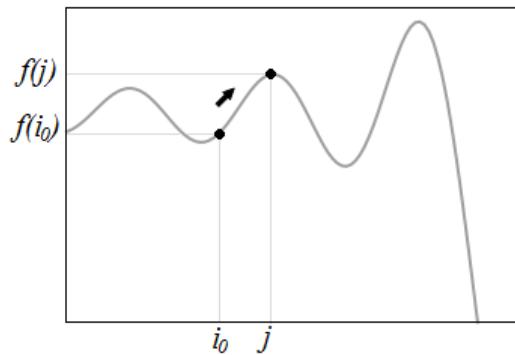


Figura 3.2 Mejora sucesiva. Se comienza en una solución inicial i_0 , luego, el algoritmo busca repetidamente vecinos que incrementen el valor de la función objetivo f , llegando así a un máximo local en j .

- *Recocido simulado.* Para este algoritmo, la aceptación de una solución vecina está dada por una distribución de probabilidad que depende de la temperatura t_k , y de la diferencia $f(i) - f(j)$, donde i es la solución en curso y j es la solución vecina. Esto quiere decir que cada solución vecina tiene cierta probabilidad de ser escogida. Tradicionalmente se escoge a F como la *distribución de Boltzmann*. Sea $i \in \mathcal{Y}$ y $j \in \mathcal{N}(i)$ en la iteración k . El criterio de aceptación para este método lo podemos describir de la siguiente manera

$$P(\text{aceptar } j) = \begin{cases} 1 & \text{si } f(j) \geq f(i) \\ \frac{1}{\exp\left(\frac{f(i)-f(j)}{t_k}\right)} & \text{si } f(j) < f(i) . \end{cases}$$

En esta expresión es claro que la probabilidad de aceptar un empeoramiento en la función objetivo depende de la temperatura t_k y de $f(i) - f(j)$. De acuerdo con el criterio anterior, la probabilidad de aceptación de una solución vecina j es pequeña si $f(i) - f(j)$ es un valor grande o si t_k es un valor pequeño. En este método, $t_k \geq 0$, $t_k \geq t_{k+1}$ y $\lim_{k \rightarrow \infty} t_k \rightarrow 0$, pues como se verá más adelante, la convergencia del algoritmo depende de que la temperatura t_k se reduzca asintóticamente hacia cero. Para valores de t_k cercanos a cero el algoritmo se comporta prácticamente como mejora sucesiva.

- *Umbral de aceptación.* En este caso los umbrales corresponderán a los valores de la temperatura, es decir, $u_k = t_k$ para toda k , donde t_k satisface las condiciones del método anterior. Se dice que en este algoritmo la posibilidad de aceptar soluciones que empeoran a la función objetivo es determinista y limitada, pues sólo se aceptan soluciones cuya función objetivo no empeore más allá de t_k , aunque con probabilidad 1. Asintóticamente los valores t_k se van acercando a cero, el algoritmo se comporta entonces como mejora sucesiva.

$$P(\text{aceptar } j) = \begin{cases} 1 & \text{si } f(i) - f(j) \leq t_k \\ 0 & \text{si } f(i) - f(j) > t_k \end{cases}.$$

En la figura 3.3 se ilustra la ventaja que posee el algoritmo de recocido simulado, y su variante umbral de aceptación, para escapar de soluciones óptimas locales.

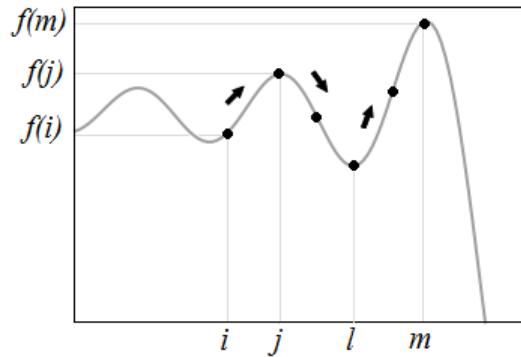


Figura 3.3 Comenzando en una solución inicial i , se aplica iterativamente el algoritmo con la expectativa de llegar a un máximo local j . Luego, como el algoritmo permite empeorar la función objetivo, se logra escapar del máximo local con la expectativa de llegar a un máximo global.

Ahora bien, a continuación veremos un ejemplo donde se aplica el algoritmo de umbral de aceptación, para ilustrar la importancia que tiene la elección adecuada de los umbrales ($u_k | k = 0, 1, 2, \dots$). Esto no resulta ser una tarea sencilla en general. Si los umbrales son demasiado grandes, el algoritmo se degenera en una caminata aleatoria; si los umbrales son demasiado pequeños, el algoritmo se comporta como mejora sucesiva, y en algunos casos no es posible determinar valores para los u_k de manera que se garantice la capacidad del algoritmo para llegar a una solución óptima global. Este ejemplo es una variante simplificada del problema propuesto por [LM86]. Sea $f : \mathcal{Y} \rightarrow \mathbb{R}$ la función objetivo definida como

$$f(i) = \left\lfloor \frac{i}{n} \right\rfloor \delta - i,$$

y el conjunto de soluciones dado por $\mathcal{Y} = \{0, 1, 2, \dots, N\}$, con $1 \ll N, n \in \mathbb{N}, \delta \in \mathbb{R}$ y $1 < \delta < n$. El problema consiste en encontrar un elemento en \mathcal{Y} que maximize el valor de la función objetivo. Sea \mathcal{N} una función de vecindad definida por

$$\mathcal{N}(i) = \begin{cases} \{i-1, i, i+1\} & \text{si } i \in \{1, \dots, N-1\} \\ \{0, 1\} & \text{si } i = 0 \\ \{N, N-1\} & \text{si } i = N \end{cases}.$$

Consideraremos dos situaciones: cuando $\delta \approx 1$, donde se requiere un pequeño empeoramiento de f para escapar de una solución óptima local; cuando $\delta \gg 1$, donde se requiere un gran empeoramiento de f para escapar de una solución óptima local. En el caso del algoritmo de mejora sucesiva no importa el valor de δ , pues al seleccionar aleatoriamente una solución inicial i en $\mathcal{Y} = \{0, 1, 2, \dots, N\}$ el algoritmo va a terminar en una solución óptima local $i' = \lfloor \frac{i}{n} \rfloor n$, como se ilustra en la figura 3.4.

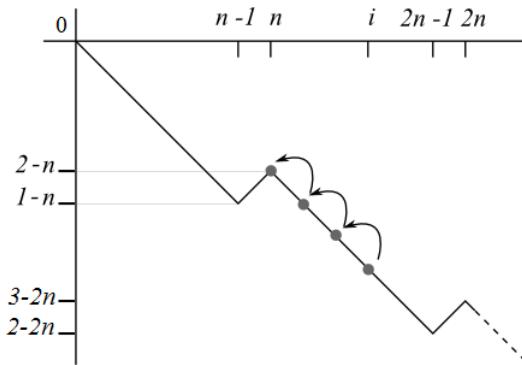


Figura 3.4 Al comenzar en i , se llega a un óptimo local en $\lfloor \frac{i}{n} \rfloor n = (1)n$.

Supongamos que $\delta \approx 1$. Si $t_k \geq 1$, el algoritmo de umbral de aceptación se comporta como una caminata aleatoria en el espacio de soluciones puesto que, dada una solución inicial i , se acepta tanto a la solución vecina $i - 1$ como a $i + 1$, ya que $|f(i) - f(j)| \leq 1$ para i, j soluciones vecinas. Véase figura 3.5.

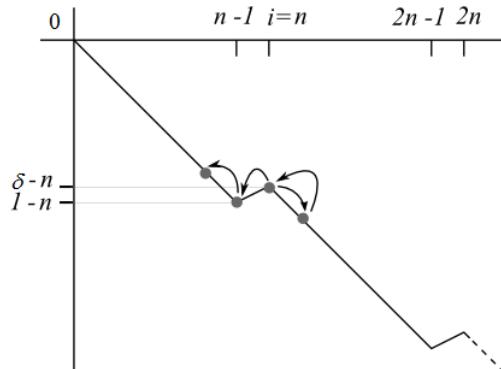


Figura 3.5 Si $t_k \geq 1$, el algoritmo se comporta como una caminata aleatoria.

Por otro lado, si $\delta - 1 \leq t_k < 1$ entonces el algoritmo no va a aceptar que la función objetivo empeore más de 1, por lo que en general aceptará mejoras hacia la izquierda. El único empeoramiento que aceptará es cuando la solución vecina a la izquierda sea

$j = mn - 1$ con $m \in \mathbb{N}$, pues en ese caso $|f(i) - f(j)| = \delta - 1$, lo cual está dentro de la tolerancia de aceptación. Aquí el algoritmo se comporta como un algoritmo de optimización, véase figura 3.6.

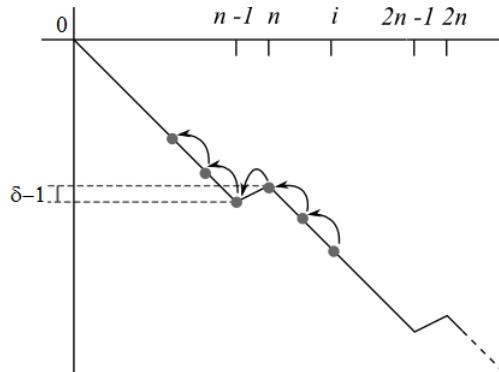


Figura 3.6 Si $\delta - 1 \leq t_k < 1$, el algoritmo de umbral de aceptación se comporta como un algoritmo de optimización.

Ahora bien, si $t_k < \delta - 1$ entonces el algoritmo únicamente aceptará soluciones vecinas hacia la izquierda cuando éstas mejoren la función objetivo, es decir, el algoritmo se comportará como mejora sucesiva y quedará atrapado en el máximo local que está en $\lfloor \frac{i}{n} \rfloor n$, dada una solución inicial i . Véase figura 3.7.

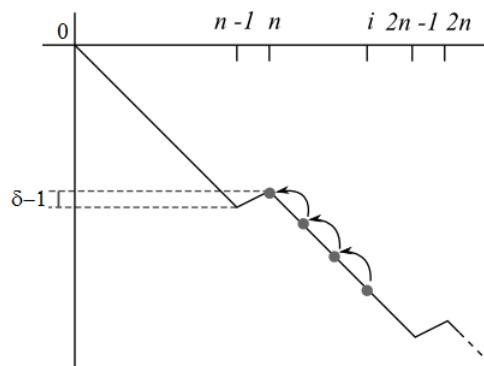


Figura 3.7 Si $t_k < \delta - 1$, el algoritmo de umbral de aceptación se comporta como mejora sucesiva.

Por último, sea $\delta \gg 1$. Si $1 < \delta - 1 \leq t_k$ el algoritmo se comporta como una caminata aleatoria, aceptando soluciones vecinas a la izquierda y a la derecha de manera indistinta. Véase figura 3.8.

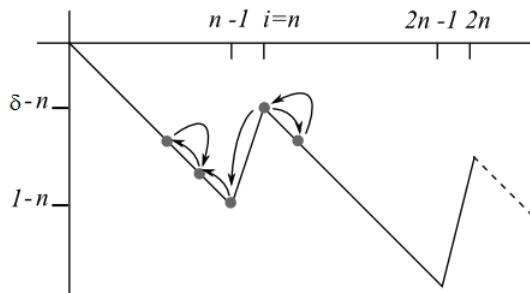


Figura 3.8 Si $\delta \gg 1$ y $1 < \delta - 1 \leq t_k$, el algoritmo se comporta como una caminata aleatoria.

Dada una solución inicial i , si $1 \leq t_k < \delta - 1$, el algoritmo siempre aceptará soluciones vecinas a la derecha y en la mayoría de las veces a la izquierda, excepto en las soluciones mn . Véase figura 3.9.

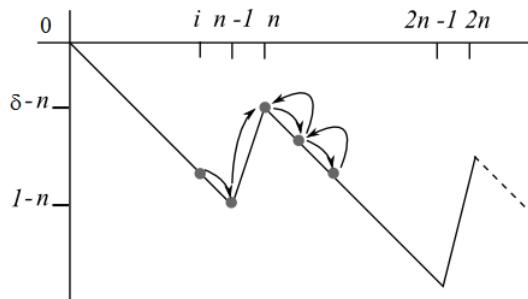


Figura 3.9 En este ejemplo $1 \leq t_k < \delta - 1$ e $i = n - 2$, una vez que se acepta la solución vecina n no es posible regresar a la solución $n - 1$.

En resumen, este sencillo ejemplo nos muestra que elegir el valor adecuado para los umbrales del algoritmo umbral de aceptación no es tarea sencilla. En algunas ocasiones simplemente no es posible determinar un valor que garantice el llegar a una solución óptima global, como lo vimos en los dos últimos casos. Por otro lado, veremos más adelante cómo en el caso del algoritmo recocido simulado existe siempre la posibilidad de escapar de soluciones óptimas locales y llegar así a una solución óptima global.

Capítulo 4

Resultados de convergencia

Como vimos anteriormente, el algoritmo de umbral de aceptación posee la diferencia sustancial con respecto al algoritmo de recocido simulado de que las aceptaciones de soluciones vecinas se dan de manera determinista en el primer caso, en tanto que en el segundo se sigue una regla probabilística. Esto impide que se pueda garantizar en general la convergencia del método de umbral de aceptación, ya que en el método de recocido simulado la virtud que le permite obtener resultados generales de convergencia es precisamente su fuerte componente estocástica, lo cual facilita hacer un análisis asintótico favorable en este sentido, como lo veremos en este capítulo.

Ahora bien, Althöfer & Koschnick han encontrado algunos resultados parciales de convergencia para el algoritmo de umbral de aceptación, no proveyendo una demostración constructiva, sino dando argumentos acerca de la existencia de una estrategia de enfriamiento que permita garantizar la convergencia. En palabras de los autores: “*Nuestras demostraciones son únicamente resultados de existencia y no nos dicen nada acerca de cómo construir sucesiones de umbrales que sean óptimas (o al menos buenas). Por lo tanto, los resultados no tendrán ningún impacto directo en la práctica del algoritmo de umbral de aceptación*” [AK91].

Sigue siendo un problema abierto el determinar una estrategia de enfriamiento adecuada, tema que abordaremos más adelante, pero primero veremos una descripción general de la teoría de cadenas de Markov, pues ésta es la herramienta básica para obtener los resultados generales de convergencia para el método de recocido simulado.

4.1. Modelos de Markov

Una cadena de Markov es un modelo matemático para describir un sistema estocástico cuyos estados están dados por una regla de transición probabilística, donde la probabilidad de alcanzar cierto estado depende exclusivamente de los estados previos más

recientes. A continuación examinaremos los conceptos básicos de esta teoría, así como algunos resultados, para tener un panorama general sobre la manera de modelar el algoritmo de recocido simulado por medio de cadenas de Markov. Una lectura con mayor profundidad y detalles en el tema de cadenas de Markov se encuentra en [IM76]. Las definiciones y resultados de las siguientes dos secciones están en acorde con [AKvL97].

Definición 4.1. Sea Ω el conjunto de posibles resultados de un proceso de muestreo. Una *cadena de Markov* es una sucesión de *ensayos*, donde la probabilidad de un ensayo dado depende únicamente del resultado del ensayo anterior. Sea $X(k)$ una variable aleatoria que denote el resultado del k -ésimo ensayo. Entonces la *probabilidad de transición* en el k -ésimo ensayo para cada par de resultados $i, j \in \Omega$ se define como

$$p_{ij}(k) = \mathbf{P}\{X(k) = j | X(k-1) = i\}$$

Definición 4.2. La matriz $P(k) = [p_{ij}(k)]$, cuyos elementos están dados por la definición anterior, se denomina la *matriz de transición*.

Definición 4.3. Una cadena de Markov es *finita* si el conjunto de resultados es finito.

Definición 4.4. Una cadena de Markov se denomina *no-homogénea* si las probabilidades de transición dependen del número de ensayo k . Si las probabilidades de transición no dependen del número de ensayo, la cadena de Markov es llamada *homogénea*.

Observación 4.1. Una propiedad importante que tienen las cadenas de Markov, como consecuencia directa de la definición 4.1, es que satisfacen la ecuación de Chapman-Kolmogorov

$$p_{ij}^{(n)} = \sum_{l \in \Omega} p_{il}^{(k)} p_{lj}^{(n-k)},$$

donde la notación $p_{ij}^{(n)}$ significa la probabilidad de llegar del estado i al estado j en exactamente n transiciones. A partir de esta ecuación se deducen de manera directa las siguientes desigualdades

$$p_{ij}^{(n+m)} \geq p_{il}^{(n)} p_{lj}^{(m)},$$

$$p_{ij}^{(n+m+r)} \geq p_{il}^{(n)} p_{ll}^{(m)} p_{lj}^{(r)}.$$

Definición 4.5. Un vector $x = (x_1, \dots, x_n)$ es *estocástico* si cumple que

$$x_i \geq 0, \quad i = 1, \dots, n, \quad \text{y} \quad \sum_{i=1}^n x_i = 1.$$

Una matriz X con n renglones y m columnas es llamada *estocástica* si sus componentes X_{ij} satisfacen

$$X_{ij} \geq 0 \quad \text{y} \quad \sum_{j=1}^m X_{ij} = 1, \quad \text{para } i = 1, \dots, n, \quad j = 1, \dots, m.$$

Aplicando el concepto de cadenas de Markov para el algoritmo de recocido simulado, el conjunto de posibles resultados Ω está dado por el espacio finito de soluciones factibles para una instancia de un problema de optimización combinatoria, así que cada transición entre las soluciones del sistema es un ensayo. Es claro notar que el resultado de un estado en particular depende únicamente del estado anterior, debido a la manera en que transicionan los estados en el algoritmo de recocido simulado, al seleccionar una solución dentro de la vecindad de la que está en curso.

Definición 4.6. *Probabilidad de transición.* Sea (\mathcal{Y}, f) una instancia de un problema de optimización combinatoria y \mathcal{N} una función de vecindad. Entonces las probabilidades de transición para el algoritmo de recocido simulado están definidas como

$$p_{ij}(k) = \begin{cases} g_{ij}(t_k) a_{ij}(t_k) & \text{si } i \neq j \\ 1 - \sum_{l \in \mathcal{Y} \setminus i} g_{il}(t_k) a_{il}(t_k) & \text{si } i = j \end{cases}, \quad \forall i, j \in \mathcal{Y},$$

donde $g_{ij}(t_k)$ denota la probabilidad de generar una solución j a partir de una solución i . La probabilidad de aceptar la solución j que fue generada a partir de la solución i está dada por $a_{ij}(t_k)$.

La matriz $[p_{ij}(k)]$ es estocástica, pero las matrices $[a_{ij}(t_k)]$ y $[g_{ij}(t_k)]$ no son necesariamente estocásticas, a éstas se les denomina *matriz de aceptación* y *matriz de generación* respectivamente. En el algoritmo de recocido simulado (en su versión de maximizar) las probabilidades se plantean de la siguiente manera:

Definición 4.7. La *probabilidad de generación* se define como

$$g_{ij}(t_k) = g_{ij} = \frac{1}{\Theta} \chi_{\mathcal{N}(i)}(j), \quad \forall i, j \in \mathcal{Y},$$

donde $\Theta = |\mathcal{N}(i)|$, para toda $i \in \mathcal{Y}$. La función característica χ_S para un subconjunto $S \subset X$ es un mapeo sobre el conjunto $\{0, 1\}$, tal que

$$\chi_S(x) = \begin{cases} 1 & \text{si } x \in S \\ 0 & \text{si } x \in X \setminus S \end{cases}.$$

La *probabilidad de aceptación* está definida como

$$a_{ij}(t_k) = \begin{cases} 1 & \text{si } f(j) \geq f(i) \\ \frac{1}{\exp\left(\frac{f(i)-f(j)}{t_k}\right)} & \text{si } f(j) < f(i) \end{cases}, \quad \forall i, j \in \mathcal{Y}.$$

La probabilidad de generación no depende del parámetro t_k y es uniforme en todas las vecindades $\mathcal{N}(i)$, que en principio deben tener el mismo tamaño, es decir, $\mathcal{N}(i) = \Theta$ para toda i .

Definición 4.8. Una *distribución estacionaria* de una cadena finita homogénea de Markov con matriz de transición P en un conjunto de resultados Ω se define como el $|\Omega|$ -vector estocástico q , cuyas componentes están dadas por

$$q_i = \lim_{k \rightarrow \infty} \mathbf{P}\{X(k) = i \mid X(0) = j\}, \quad \text{para toda } j \in \Omega,$$

o simplemente como

$$q_i = \lim_{k \rightarrow \infty} \mathbf{P}\{X(k) = i\}.$$

Los valores de q_i se refieren a la proporción de veces que el estado i aparecerá en la cadena de Markov a largo plazo, es decir, en un número infinito de ensayos. Por la definición de q y P se tiene que $qP = q$, ya que

$$(q_1, q_2, \dots, q_{|\Omega|}) \begin{vmatrix} p_{11} & \cdots & p_{1|\Omega|} \\ \vdots & \ddots & \vdots \\ p_{|\Omega|1} & \cdots & p_{|\Omega||\Omega|} \end{vmatrix} = \left(\sum_{i \in \Omega} q_i p_{i1}, \dots, \sum_{i \in \Omega} q_i p_{i|\Omega|} \right),$$

donde $q_j = \sum_{i \in \Omega} q_i p_{ij}$ para cada j , pues se están sumando los productos de la probabilidad de que aparezca el estado i con la probabilidad de transicionar del estado i al estado j , para toda i , lo cual naturalmente nos dice la probabilidad total de que aparezca el estado j en un número infinito de ensayos. A esta distribución estacionaria se le conoce también como *vector de probabilidad invariante*, que viene siendo el punto fijo de la transformación cuya matriz asociada es P . Se puede probar que para el algoritmo de recocido simulado dicha distribución estacionaria existe, pero veamos antes algunas definiciones necesarias.

Definición 4.9. Una cadena de Markov con matriz de transiciones P en un conjunto de resultados Ω es *irreducible* si para cada par de resultados $i, j \in \Omega$ existe una probabilidad positiva de llegar a j a partir de i en una cantidad finita de ensayos, es decir,

$$\forall i, j \in \Omega \quad \exists n \in \mathbb{Z}^+ \text{ tal que } (p^n)_{ij} > 0.$$

En otra terminología, se dice que dos estados están *comunicados* cuando se puede acceder del uno al otro, y viceversa, en un número finito de ensayos. Que una cadena sea irreducible quiere decir que todos los estados están comunicados.

Definición 4.10. Una cadena de Markov con matriz de transiciones P es *aperiodica* si para cada $i \in \Omega$ el máximo común divisor $\text{mcd}(\mathcal{D}_i) = 1$, donde \mathcal{D}_i es el conjunto de todos los enteros $n > 0$ tal que $p_{ii}^{(n)} > 0$.

Al número $d_i = \text{mcd}(\mathcal{D}_i)$ se le llama el *periodo* de i . De manera informal, el periodo se refiere a que, tomando como estado inicial i , la cantidad de estados por los que se tienen que pasar para regresar a i debe ser un múltiplo de d_i . La aperiodicidad requiere que todos los estados tengan periodo 1. El siguiente lema servirá para garantizar cuándo una cadena de Markov es aperiódica [SP06].

Lema 4.1. *En una cadena de Markov si los estados $i, j \in \Omega$ están comunicados entonces $d_i = d_j$.*

*Demuestra*ción. Por hipótesis tenemos que existen enteros $n, r, m \geq 1$ tales que $p_{jj}^{(n)} > 0$, $p_{ij}^{(r)} > 0$ y $p_{ji}^{(m)} > 0$. Ahora bien, por las desigualdades que se derivan de la ecuación de Chapman-Kolmogorov (observación 4.1) tenemos que

$$p_{ii}^{(r+m)} \geq p_{ij}^{(r)} p_{ji}^{(m)},$$

$$p_{ii}^{(r+n+m)} \geq p_{ij}^{(r)} p_{jj}^{(n)} p_{ji}^{(m)}.$$

Lo cual nos dice que $r+m$ y $r+n+m$ son múltiplos de d_i , lo cual implica que su diferencia $n = r+n+m - (r+m)$ también es múltiplo de d_i , lo cual implica que d_i es divisor de cada n que cumpla $p_{jj}^{(n)} > 0$, por lo cual $d_i \leq d_j$ ya que tales números son múltiplos de d_j . Por simetría obtenemos que $d_j \leq d_i$ y así terminamos la demostración. ■

El lema anterior nos lleva al siguiente corolario.

Corolario 4.2. *En una cadena de Markov irreducible todos los estados tienen el mismo periodo.*

Así que, si en una cadena de Markov irreducible uno de los estados es aperiódico, entonces ésta es aperiódica, por lo cual si $\exists j \in \Omega : p_{jj} > 0$ se tiene una cadena de Markov irreducible aperiódica.

El siguiente resultado es acerca de la existencia y unicidad de una distribución estacionaria, requiriendo como condición que la matriz de transición sea aperiódica e irreducible; véase [IM76] para su demostración.

Teorema 4.3. *Sea P la matriz de transición asociada con una cadena finita homogénea de Markov, irreducible y aperiódica, en un conjunto de posibles resultados Ω . Entonces existe una única distribución estacionaria q cuyas componentes q_j están determinadas únicamente por*

$$q_j = \sum_{i \in \Omega} q_i p_{ij}, \quad \text{para toda } j \in \Omega.$$

Definición 4.11. Una cadena de Markov homogénea con matriz de transición $P = [p_{ij}]$ en un conjunto de resultados Ω es *reversible* si existe una distribución de probabilidades q tal que

$$q_i p_{ij} = q_j p_{ji} \quad \text{para toda } i, j \in \Omega.$$

Las ecuaciones de esta definición se llaman *ecuaciones de balanceo detallado*. Del teorema 4.3 y la definición 4.11 obtenemos el siguiente corolario.

Corolario 4.4. *Sea una cadena de Markov finita homogénea, irreducible, aperiódica y reversible con q la distribución de probabilidades que satisface las ecuaciones de balanceo detallado, entonces q es la única distribución estacionaria.*

Demuestra. Como la cadena de Markov es reversible satisface las ecuaciones de balanceo detallado $q_i p_{ij} = q_j p_{ji}$. Entonces, si sumamos sobre i en ambos lados de la ecuación tenemos que

$$\begin{aligned} \sum_{i \in \Omega} q_i p_{ij} &= \sum_{i \in \Omega} q_j p_{ji} \\ &= q_j \sum_{i \in \Omega} p_{ji} = q_j, \end{aligned}$$

ya que P es matriz estocástica por lo que $\sum_{i \in \Omega} p_{ji} = 1$ para toda j . ■

A continuación veremos el resultado principal que garantiza la convergencia del algoritmo de recocido simulado.

4.2. Teorema de convergencia

Para demostrar la convergencia asintótica del algoritmo de recocido simulado, éste se puede modelar como una cadena de Markov no homogénea de longitud infinita que consiste en una sucesión de cadenas de Markov homogéneas de longitud finita. Cada sección de la cadena que representa una cadena de Markov homogénea contiene un número finito de transiciones donde las probabilidades de transición no dependen de k , es decir, t_k se mantiene constante en la cadena homogénea y su valor disminuye al comenzar la siguiente. Las probabilidades de transición son aquéllas de la definición 4.6. Se supondrá además que la sucesión $(t_k | k = 0, 1, 2, \dots)$ satisface las siguientes condiciones

$$\lim_{k \rightarrow \infty} t_k = 0, \text{ y}$$

$$t_k \geq t_{k+1}, \text{ para } k = 0, 1, \dots$$

El siguiente teorema nos dice que bajo ciertas condiciones en la velocidad de convergencia de la sucesión $(t_k | k = 0, 1, 2, \dots)$, la cadena no homogénea de Markov asociada

con el algoritmo de recocido simulado converge en probabilidad. La prueba de este teorema se puede consultar en [AKvL97].

Teorema 4.5. *Sea (\mathcal{Y}, f) una instancia de un problema de optimización combinatoria, \mathcal{N} una función de vecindad, y $P(k)$ la matriz de transición de la cadena de Markov no homogénea asociada con el algoritmo de recocido simulado planteado en las definiciones 4.6 y 4.7. Más aún, supongase que las siguientes condiciones se satisfacen*

$$\forall i, j \in \mathcal{Y} \exists r \geq 1 \exists l_0, l_1, \dots, l_r \in \mathcal{Y}$$

con

$$l_0 = i, l_r = j, \quad y \quad g_{l_k l_{k+1}} > 0, \quad k = 0, 1, \dots, r-1,$$

y además

$$t_k \geq \frac{\Gamma}{\log(k+k_0)}, \quad k = 0, 1, \dots,$$

para algún valor $\Gamma > 0$ y $k_0 > 2$.

Entonces la distribución asociada a la cadena de Markov converge al vector estocástico $q^* = \{q_i^*\}$, definido por

$$q_i^* = \lim_{t \rightarrow 0} q_i(t) = \frac{1}{|\mathcal{Y}^*|} \chi_{(\mathcal{Y}^*)}(i),$$

donde \mathcal{Y}^* representa el conjunto de soluciones óptimas, y

$$q_i(t) = \frac{\exp(-f(i)/t)}{\sum_{j \in \mathcal{Y}} \exp(-f(j)/t)} \text{ para toda } i \in \mathcal{Y},$$

en otras palabras:

$$\lim_{k \rightarrow \infty} \mathbf{P}\{X(k) \in \mathcal{Y}^*\} = 1.$$

Este resultado nos dice que el algoritmo de recocido simulado encuentra una solución óptima con probabilidad 1, si se le permite un número infinito de iteraciones. Por lo tanto, una implementación del algoritmo con un número finito de iteraciones no puede garantizar el hallazgo de una solución óptima, sólo es posible encontrar soluciones aproximadas en tiempos de cómputo razonables.

En la siguiente sección, abordaremos el tema de cómo implementar el algoritmo de recocido simulado para la resolución del problema de optimización de este trabajo, con un número finito de iteraciones, mediante la correcta elección del conjunto de parámetros que gobiernan el comportamiento del algoritmo, es decir, de lo que se denomina la estrategia de enfriamiento.

4.3. Estrategia de enfriamiento

Un aspecto muy importante en la implementación del algoritmo de recocido simulado o umbral de aceptación, es la determinación de una estrategia de enfriamiento adecuada. Como ya se señaló anteriormente, el parámetro de la temperatura desempeña un papel importante para garantizar la convergencia del algoritmo, siendo indispensable que la temperatura converja hacia el cero. Por otro lado, este parámetro es aquél que determina qué tanto se puede empeorar la función objetivo, de modo que el algoritmo tenga la posibilidad de escapar de mínimos locales. Esto va de acuerdo con la analogía con el proceso físico de recocido, que a través de baños térmicos, un material en el punto de fusión se va enfriando lentamente para lograr que sus moléculas se vayan acomodando en una configuración molecular óptima de baja energía.

Ahora bien, en la sección anterior se mencionó que para garantizar el hallazgo de una solución óptima global es necesario un número infinito de iteraciones, lo cual resulta imposible en la práctica, así que trabajaremos con implementaciones del algoritmo con un número finito de iteraciones, para obtener aproximaciones de la solución óptima en nuestra aplicación para la selección de una paleta de colores en el espacio de color CIE $L^*u^*v^*$. “*Una implementación del algoritmo de recocido simulado, con un número finito de iteraciones, se obtiene al generar una sucesión de cadenas de Markov homogéneas de longitud finita con valores descendientes de la temperatura. Para esto, un conjunto de parámetros que gobiernan la convergencia del algoritmo se tienen que especificar. Estos parámetros, y la manera en que gobiernan la convergencia del algoritmo, en conjunto forman lo que se denomina una estrategia de enfriamiento*” [AKvL97].

Definición 4.12. Un *estrategia de enfriamiento* especifica una sucesión finita de valores del parámetro temperatura t , y un número finito de transiciones para cada valor de la temperatura t , está definido por

- Un valor para la temperatura inicial t_0 .
- Una función $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ de decremento para disminuir el valor de la temperatura asintóticamente a cero.
- Una cota para la temperatura final ε , que llamaremos *cero virtual*.
- Una estrategia para definir la longitud de cada cadena de Markov homogénea.

Es importante señalar que para implementar el algoritmo con un número finito de iteraciones existen resultados teóricos sobre la aproximación de la distribución estacionaria, la velocidad de convergencia de la distribución, y el valor Γ para la función de decremento, sin embargo, realizar las estimaciones correspondientes puede tomar un tiempo considerable de cómputo. Aarts y Van Laarhoven probaron que para aproximar

arbitrariamente la distribución estacionaria se requiere un número de transiciones al menos tan grande como el cuadrado del tamaño del espacio de soluciones [AL85], así que en este caso, enumerar todas las soluciones tomaría menos tiempo que aproximar a la solución óptima de manera arbitrariamente cercana con el algoritmo de recocido simulado.

Puesto que el objetivo de este trabajo es encontrar una aproximación a la solución óptima de manera eficiente, en lugar de buscar una estrategia de enfriamiento que nos permita gran precisión, a un costo mayor de lo que sería explorar el espacio de soluciones exhaustivamente, utilizaremos el enfoque de las estrategias de enfriamiento heurísticas, de las cuales hay estáticas y dinámicas.

Una estrategia de enfriamiento estática es aquel cuyos parámetros no cambian durante la ejecución del algoritmo, uno dinámico es aquel cuyos parámetros varían durante la ejecución del algoritmo. Una estrategia de enfriamiento bastante popular es la estrategia *geométrica*. Esta estrategia se origina del trabajo de Kirkpatrick, Gelatt y Vecchi [KGV83], y todavía es utilizado ampliamente, a continuación daremos una descripción.

Valor inicial de la temperatura. Se escoge un valor suficientemente grande para t_0 , que puede ser

$$\max_{i,j \in \mathcal{Y}} f(i) - f(j),$$

donde i, j son soluciones vecinas. En principio, el escoger t_0 de esta manera es para incrementar la probabilidad de la aceptación de cualquier solución vecina, durante cada iteración donde la temperatura sea t_0 .

Función de decremento. Está definida por la siguiente regla:

$$t_{m+1} = \varphi t_m, m = 0, 1, \dots,$$

donde t_m es la temperatura de la m -ésima cadena de longitud finita homogénea de Markov; $\varphi \in [0, 1]$ es lo que se conoce como el *factor de enfriamiento*.

Temperatura final. Este valor se escoge pequeño, procurando que esté relacionado con la diferencia más pequeña posible entre las funciones objetivo de dos soluciones vecinas.

Longitud de la cadena de Markov. La longitud de las cadenas de Markov se fija en algún número que pueda estar relacionado con el tamaño de las vecindades de la instancia con la que se esté trabajando.

En el problema que nos concierne en este trabajo utilizaremos la estrategia geométrica, con algunas adaptaciones convenientes, pero también se explorará el uso de un estrategia con una función de decremento oscilante. Veamos ahora los detalles de las estrategias de enfriamiento utilizadas en este trabajo.

- Para determinar el valor inicial de la temperatura consideramos una *razón de aceptación inicial* del $95 \pm 0.1\%$, que se refiere al porcentaje de transiciones aceptadas durante la ejecución del algoritmo a una temperatura inicial t_0 . Para lograr esto, utilizamos un método de búsqueda binaria, descrito con mayor detalle en la

sección 8.2 de selección de parámetros. La justificación para tener un porcentaje de aceptación inicial alto es porque queremos darle flexibilidad al algoritmo para explorar el espacio de soluciones, permitiendo la aceptación de soluciones que empeoran la función objetivo con la esperanza de escapar de algún máximo local, pero no le conferimos un porcentaje de aceptación del 100% puesto que el algoritmo se degeneraría en una caminata aleatoria sin ningún control en el espacio de soluciones.

- Utilizaremos dos funciones de decremento, la función tradicional

$$g(\varphi, m) = t_{m+1} = \varphi^m t_0$$

y exploraremos el uso de una función alternativa, que se deriva de g pero con la característica de ser oscilante

$$h(\varphi, m) = t_{m+1} = \frac{1}{2} \varphi^m t_0 \cdot (\operatorname{sen}(m) + 2).$$

La razón para explorar el uso de esta última función de decremento está motivada por el trabajo de Boese & Kahng [BK94], donde se propone como alternativa el uso de estrategias de enfriamiento que utilicen funciones periódicas para regular la temperatura. La función de decremento h no es periódica en sí, sino oscilante, y el factor $\varphi^m t_0$ la obliga a converger a cero. En la parte experimental compararemos ambas funciones para determinar si existe o no alguna diferencia significativa en el empleo de ellas. En la figura 4.1 se muestra la gráfica de g y h con $t_0 = 20$ y $\varphi = 0.9$.

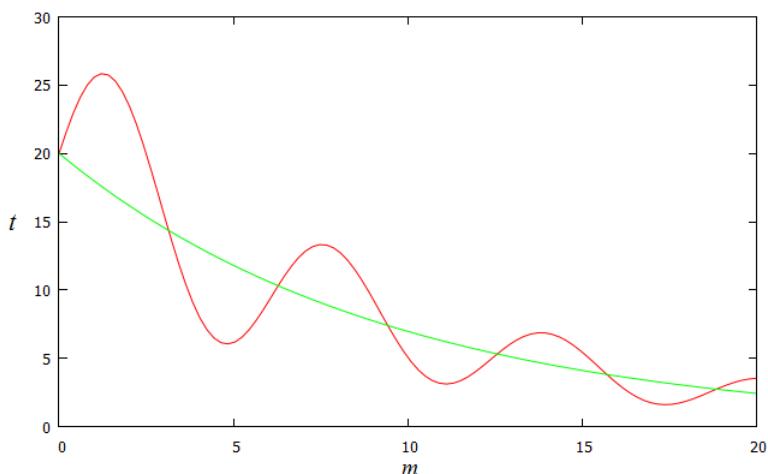


Figura 4.1 Gráfica de las funciones de decremento g y h .

- Para el cero virtual ε utilizaremos valores menores que 1, como 0.5 y 0.1. La justificación para utilizar estos valores se encuentra en la siguiente observación. En el cubo RGB la distancia mínima entre dos colores es 1, así que, se tomaron los ocho colores básicos (blanco, negro, amarillo, verde, cyan, magenta, azul y rojo) y se tomó un color vecino para cada uno, al variar una de las coordenadas de cada color básico en ± 1 . La distancia entre estas parejas de colores en el cubo RGB es 1. Las distancias de sus respectivas imágenes en el espacio CIE $L^*u^*v^*$ se encuentran en la tabla 4.1, donde se aprecia claramente que las distancias se contraen en el espacio CIE $L^*u^*v^*$. A partir de esta observación, podemos aproximar la distancia mínima entre dos colores en el espacio CIE $L^*u^*v^*$ con 0.5. Utilizar el valor 0.1 es para obtener una solución cuasi-óptima más refinada.

Colores RGB	d_{CIERGB}	$d_{CIEL^*u^*v^*}$
(255, 254, 255), (255, 255, 255)	1	0.85
(0, 0, 1), (0, 0, 0)	1	0.08
(255, 254, 0), (255, 255, 0)	1	0.79
(0, 254, 0), (0, 255, 0)	1	0.56
(0, 254, 255), (0, 255, 255)	1	0.89
(255, 0, 254), (255, 0, 255)	1	0.84
(0, 0, 254), (0, 0, 255)	1	0.59
(254, 0, 0), (255, 0, 0)	1	0.72
Promedios:	1	0.66

Tabla 4.1 Distancias entre colores en el cubo RGB y en el espacio CIE $L^*u^*v^*$.

- Determinar la longitud de las cadenas de Markov no es una tarea sencilla: cadenas demasiado largas pueden disminuir la eficiencia del algoritmo al incrementar considerablemente los tiempos de ejecución, sin que ello represente una mejoría significativa en la calidad de la solución; cadenas demasiado cortas podrían evitar que el algoritmo tuviera suficiente tiempo para alcanzar una solución aproximada de buena calidad. En nuestra implementación, la longitud de cada cadena de Markov homogénea será un parámetro dinámico. A continuación se dará una descripción de la estrategia utilizada para determinar la longitud de las cadenas de Markov homogéneas.

Un concepto importante que está relacionado con la longitud de las cadenas finitas de Markov homogéneas, y consecuentemente con el cambio de la temperatura, es el de *equilibrio dinámico*, que se refiere a que el algoritmo ha alcanzado un estado donde, en promedio, no mejora ni empeora la función objetivo, lo cual hace necesario el decremento de la temperatura. Para ello, se calcula el promedio de los valores de f evaluada en un lote, que es un conjunto de λ soluciones aceptadas consecutivamente, y se compara este promedio con el promedio del lote de aceptaciones anteriores. Si el promedio

correspondiente al último lote es igual al promedio que corresponde al anterior, entonces podemos suponer que se ha conseguido un estado de equilibrio dinámico. Por otro lado, si el promedio del último lote es menor que el promedio del lote anterior, entonces se están aceptando demasiadas soluciones que empeoran a la función objetivo. En ambos casos es necesario disminuir la temperatura para elevar el promedio de la función objetivo en las aceptaciones.

Ahora bien, si los promedios de los $q \in \mathbb{N}$ lotes subsecuentes son progresiva y estrictamente crecientes, entonces el algoritmo sigue trabajando con el mismo valor de t_m , prolongando así la longitud de la m -ésima cadena de longitud finita de Markov homogénea a $(q+1)\lambda$. En la figura 4.2 se ejemplifica la situación anterior. La función escalonada representa los valores de los promedios de cada lote de soluciones aceptadas. Los valores enteros en las abcisas corresponden a las soluciones aceptadas, mientras que su ordenada correspondiente es el valor de la función objetivo f evaluada en la solución aceptada.

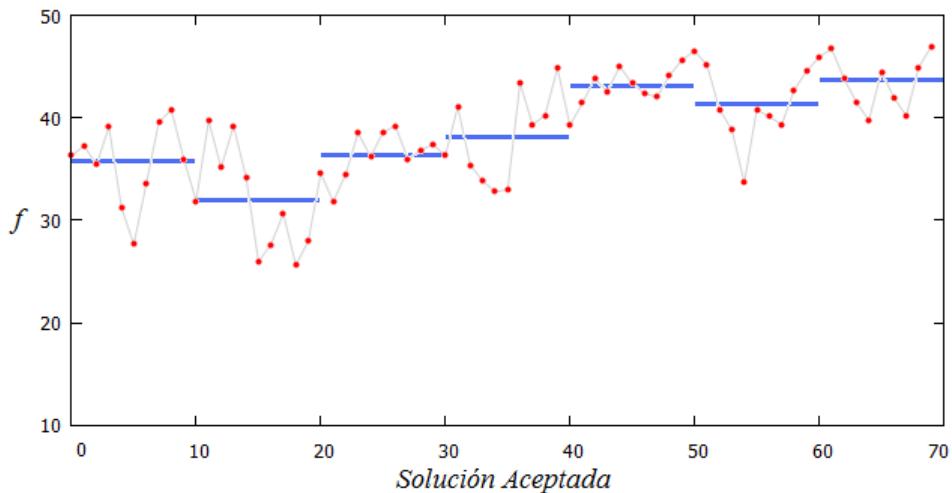


Figura 4.2 En este ejemplo $\lambda = 10$. La primera cadena de Markov tiene una longitud de 20, ya que al comparar los promedios del primer y segundo lote se nota que es necesario aplicar el factor de enfriamiento a la temperatura t_0 , logrando así que el promedio del siguiente lote se incremente. La siguiente cadena de Markov, con temperatura $t_1 = \phi t_0$, tiene de longitud 40.

En el capítulo ocho daremos una descripción detallada de cómo determinamos los parámetros del algoritmo para nuestra aplicación, siguiendo los lineamientos generales ya descritos.

Capítulo 5

Trabajos previos

El problema de encontrar un conjunto de n colores lo más diferenciados entre sí ha sido estudiado con anterioridad por varios autores. Uno de los primeros trabajos en este sentido es el de Carter & Carter, donde se plantea el problema como “*la colocación de n puntos tal que la distancia mínima entre cualesquiera dos de ellos, D , sea tan grande como sea posible dado el número de puntos deseados, n , y los límites de la región, R , que representa el gamut del medio de color utilizado*” [CC82].

En este trabajo, el primer paso es colocar aleatoriamente n puntos con una distribución uniforme en R (en este caso R es el cubo RGB). Los primeros movimientos para separar los puntos son hechos con vecindades grandes, para alcanzar el óptimo rápidamente. El método heurístico propuesto consiste en examinar primeramente todas las distancias entre las parejas formadas por los n colores, identificando aquélla que sea menor y la pareja de colores que se encuentren a dicha distancia, que se denomina D . Después se examinan los puntos adyacentes a estos dos colores, el autor utiliza vecindades de 26 puntos adyacentes en un empaquetamiento rectangular, y se consideran las alternativas de intercambio con los vecinos de ambos colores.

Las alternativas de intercambio se acomodan en una lista ordenada dependiendo de cuánto se expande D al hacer el intercambio. Se acepta la primera alternativa de intercambio de esta lista que mejore D y que no empeore a las demás distancias. Si no se logra un intercambio, el tamaño de la vecindad se reduce a la mitad y se repite el proceso de búsqueda, véase la figura 5.1. Si al llegar al tamaño más pequeño de vecindad que permite expresar la computadora no se han hecho intercambios, entonces se vuelve a incrementar el tamaño de la vecindad al máximo y se repite la búsqueda para revisar que no se llegó a un óptimo local. Si al pasar por todos los tamaños posibles para las vecindades no se realizan intercambios, se termina el algoritmo y se da la solución en curso como la óptima.

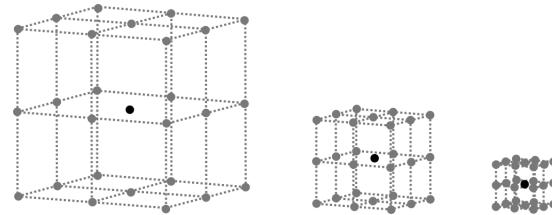


Figura 5.1 Empaquetamiento rectangular, vecindades utilizadas por Carter & Carter. Aquí se muestran en tamaños consecutivamente menores.

Por otro lado, Campadelli et al. [CPS99], toman un enfoque más general al plantear el problema como uno de optimización combinatoria en gráficas. En su planteamiento $G = \{V, E, W\}$ es una gráfica completa sin orientación, donde V es el conjunto de vértices (el conjunto de todos los colores del espacio de color), E el conjunto de aristas que representan las distancias entre los vértices, y W una función $W : E \rightarrow \mathbb{R}^+$ que asigna los pesos (distancias) a las aristas. Tenemos que $W_{ij} = W(\{i, j\}) = d_{ij}$ es la distancia entre los colores i, j , con d la función de distancia seleccionada.

El problema consiste en encontrar, entre todas las subgráficas completas de G con n vértices, aquélla que maximize la suma de los pesos (distancias) entre todas las parejas de vértices (colores) de tal subgráfica. La estrategia empleada para atacar este problema consiste en plantearlo como un problema de minimización, donde la función objetivo a minimizar es la suma de los recíprocos de las distancias elevadas a una potencia α

$$F_\alpha(\hat{G}) = \sum_{(i,j) \in \hat{E}} \frac{1}{(\hat{W}_{ij})^\alpha},$$

donde $\hat{G} = \{\hat{V}, \hat{E}, \hat{W}\}$ es una subgráfica completa de G con n vértices y \hat{W} es la restricción de W a \hat{G} . El mínimo global de este problema está dado por

$$\min \sum_{1 \leq i < j \leq |V|} \frac{1}{(W_{ij})^\alpha} x_i x_j \quad \text{tal que} \quad \sum_{i=1}^{|V|} x_i = n,$$

donde $x_i = 1$ si el vértice i está seleccionado y $x_i = 0$ si el vértice i no está seleccionado. Una solución se puede expresar entonces como un vector del hipercubo $\{0, 1\}^{|V|}$ con exactamente n coordenadas con valor igual a 1. Dos soluciones se consideran vecinas cuando comparten exactamente $n - 1$ índices con el valor 1 en la coordenada correspondiente, es decir, cuando difieren exactamente en dos coordenadas. El algoritmo empleado comienza con una solución aleatoria y va buscando mejoras entre las soluciones vecinas. Cuando una mejora se encuentra se reescribe la solución en curso y se continúa la búsqueda hasta que no haya soluciones vecinas que minimicen el valor de F_α .

Un inconveniente que tiene este enfoque es que no toma en cuenta la geometría del espacio de color, ya que su planteamiento es en abstracto mediante el uso de gráficas,

y la noción de vecindad no coincide con aquélla de \mathbb{R}^3 , donde la cercanía se mide en términos de la distancia euclíadiana.

Glasbey et al. [GvdHTG06] plantean el problema de la siguiente manera: “*encuéntrese un conjunto de colores $\{c_1, c_2, \dots, c_n\}$ tal que la distancia mínima entre cualquier par de ellos sea tan grande como sea posible*”. En su trabajo escogen como espacio de color el cubo RGB con la métrica de disimilitud de color CIELAB como función de distancia. Dos métodos son propuestos, a saber, un algoritmo de búsqueda secuencial que consiste básicamente en seleccionar aleatoriamente un color inicial y luego ir añadiendo sucesivamente colores (hasta alcanzar los n), donde cada color es añadido si su distancia al conjunto parcial de colores seleccionados es máxima. La desventaja de este método es que no hay ninguna garantía de encontrar soluciones óptimas.

Como los métodos de mejora sucesiva corren el riesgo de quedar atrapados en óptimos locales, el segundo método que proponen es uno estocástico para reducir este riesgo, en este caso el método de recocido simulado. El algoritmo comienza con una solución inicial aleatoria de n colores, luego, en cada iteración se considera reemplazar uno de los dos colores que se encuentran a distancia mínima D , con un color escogido al azar en una vecindad que consiste en una caja de $5 \times 5 \times 5$ unidades centrada en el color a reemplazar. Se acepta el reemplazo con probabilidad

$$p = \min \left\{ 1, \exp \left(\frac{D_{\text{NEW}} - D_{\text{OLD}}}{T} \right) \right\},$$

donde D_{NEW} y D_{OLD} representan las distancias mínimas de la solución propuesta y de la solución en curso, respectivamente, y $T > 0$ es la temperatura que va disminuyendo conforme las iteraciones aumentan. Si la función objetivo se mejora el reemplazo se acepta, de lo contrario se acepta con probabilidad $\exp(\frac{D_{\text{NEW}} - D_{\text{OLD}}}{T})$. El algoritmo se detiene cuando se alcanza un número predeterminado de iteraciones o cuando ningún reemplazo se acepta durante un ciclo. El inconveniente de esta implementación es que el tamaño de las vecindades se mantiene fijo durante todas las iteraciones y para toda n . Más adelante veremos que, cuando el tamaño de la vecindad depende del valor de n y del avance en la ejecución del algoritmo, el desempeño del algoritmo será más eficiente.

Otra aportación mucho más reciente en el estudio de este problema se encuentra en [Ste09, SP09], con un enfoque distinto al de los trabajos ya mencionados. El planteamiento del problema difiere un poco de los demás en lo siguiente: no sólo se busca un conjunto X de n colores lo más distintos entre sí, sino que también, dado un conjunto fijo F de k colores, se busca que los colores de X también se encuentren lo más lejos posible del conjunto F . Esta última condición surge de la idea de Steinruecken de crear un conjunto óptimo de colores para una aplicación visual que pueda evitar ciertos colores específicos, que un usuario con problemas de daltonismo no pueda distinguir. Entonces el planteamiento de Steinruecken es más general, y en este sentido nuestro problema es el caso particular cuando $F = \emptyset$.

El primer paso consiste en hacer una aproximación de la imagen del cubo RGB, en el espacio CIE $L^*u^*v^*$, con su poliedro envolvente convexo, como lo ilustra la figura 5.2. Despu s, se seleccionan n puntos de manera aleatoria en la superficie del poliedro como soluci n inicial, luego se mueven los puntos estrictamente sobre la frontera de manera que su distancia m nima sea maximizada, utilizando un m todo de programaci n cuadr\'atica secuencial basado en el m todo de optimizaci n continua de Newton-Lagrange (para mayor referencia sobre este m todo v ase [KKD11, AM93]). Habiendo optimizado la ubicaci n de los puntos estrictamente en la frontera del poliedro, se utiliza un diagrama de Voronoi tridimensional ([OAS⁺00]) para buscar soluciones sub ptimas y para detectar posibles mejoras. El diagrama de Voronoi se utiliza para encontrar la *bola* m s grande dentro del espacio de color que no contenga a ning n color del conjunto F . Si el radio de la bola excede la distancia m nima calculada hasta ese momento, entonces se ha encontrado una mejora a la soluci n y uno de los dos colores que minimiza la distancia se desplaza hacia el centro de dicha bola.

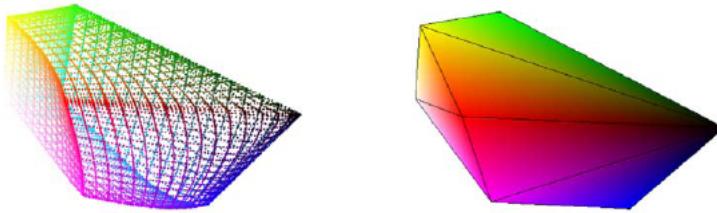


Figura 5.2 A la izquierda una representaci n de la imagen del cubo RGB en el espacio CIE $L^*u^*v^*$, a la derecha el poliedro convexo que lo approxima [SP09].

El inconveniente en este planteamiento es que la imagen del cubo RGB en el espacio CIE $L^*u^*v^*$ no representa un s olido convexo (esto se puede apreciar en las aproximaciones gr ficas, como la de la figura 5.2), as  que al trabajar en la superficie de la c scara convexa se estar n utilizando colores fuera del gamut del cubo RGB. Al ajustar los valores de la soluci n obtenida se podr a da ar la calidad de la misma.

En el siguiente cap tulo explicaremos la implementaci n del algoritmo de umbral para resolver el problema de dispersi n Max-Min en el cubo unitario.

Capítulo 6

Descripción del método de optimización en el cubo RGB

Habiendo examinado las ideas centrales de la optimización combinatoria, así como un primer acercamiento al problema que nos ocupa en el segundo capítulo de este trabajo, y la revisión de las ideas en trabajos previos relacionados, haremos ahora el planteamiento formal, primeramente en el cubo RGB por simplicidad, para después trasladar el cúmulo de ideas al espacio CIE L*u*v* y resolver el problema de optimización planteado en este trabajo.

6.1. Formulación del problema

Sea $\mathcal{C}_{RGB} = \{(r, g, b) \mid r, g, b \in \{0, 1, \dots, 255\}\}$ el cubo RGB discretizado y reescalado con 16,777,216 elementos. Cada elemento en \mathcal{C}_{RGB} representa las tres componentes de un color en el sistema RGB, descrito en el primer capítulo. La instancia correspondiente al problema de optimización combinatoria que nos concierne está dada por $(\mathcal{Y}(n, \mathcal{C}_{RGB}), f)$, donde $\mathcal{Y} = \mathcal{Y}(n, \mathcal{C}_{RGB})$ es el conjunto de todas las combinaciones posibles de n elementos de \mathcal{C}_{RGB} y f es la distancia mínima entre dos puntos de una solución de \mathcal{Y} . Es claro que la cardinalidad de \mathcal{Y} está dada por el coeficiente binomial $\binom{256^3}{n}$. Entonces, si quisieramos resolver el problema de manera exhaustiva para $n = 20$, por ejemplo, ¡se tendrían que examinar no menos de 2^{392} casos!

$$\binom{256^3}{20} = \frac{(256^3)!}{(256^3 - 20)!20!} > \frac{(256^3 - 20)^{20}}{20!} > \frac{(2^{23})^{20}}{2^{68}} = \frac{2^{460}}{2^{68}} = 2^{392}.$$

Esto nos hace ver con claridad la enorme cantidad de tiempo de cómputo que requeriría encontrar la solución óptima global examinando cada una de las posibilidades. Por

otro lado, debido a que no se conoce un método exacto para resolver este problema eficientemente, el camino de los métodos heurísticos es el más apropiado para resolverlo.

La función objetivo f en este caso es la distancia mínima entre los n puntos de una solución factible $V \in \mathcal{Y}$, es decir

$$f(V) = \min_{x,y \in V} d(x,y),$$

donde d es la métrica euclíadiana usual.

Ahora daremos la función de vecindad $\mathcal{N} : \mathcal{Y} \rightarrow 2^{\mathcal{Y}}$, pero para ello necesitamos una definición que nos será útil para construir las soluciones vecinas.

Definición 6.1. Una δ -caja-vecindad $C(r, \delta)$ para $r \in \mathbb{R}^n$ y $\delta > 0$ es el producto cartesiano

$$\prod_{i=1}^n B(r_i, \delta),$$

donde r_i es la i -ésima coordenada de r y $B(r_i, \delta) = \{s \in \mathbb{R} \mid d(r_i, s) \leq \delta\}$.

Definición 6.2. Sea $V = \{v_1, v_2, \dots, v_n \mid v_i \in \mathbb{R}^3\}$ una solución factible del espacio de soluciones \mathcal{Y} y sea $\delta > 0$. Decimos que $V' = \{v'_1, v'_2, \dots, v'_n \mid v'_i \in \mathbb{R}^3\}$ es una *solución vecina* de V si existe un j tal que

$$v'_i = v_i \text{ para } i \neq j \text{ y } v'_j \in C(v_j, \delta),$$

es decir, V y V' difieren exactamente en un elemento.

A partir de esto es claro ver que la definición de vecindad es simétrica. La figura 6.1 ilustra gráficamente la noción de vecindad mediante un ejemplo.

Definición 6.3. Definimos la función de vecindad $\mathcal{N}_\delta : \mathcal{Y} \rightarrow 2^{\mathcal{Y}}$ como

$$\mathcal{N}_\delta(V) = V \cup C(v_j, \delta).$$

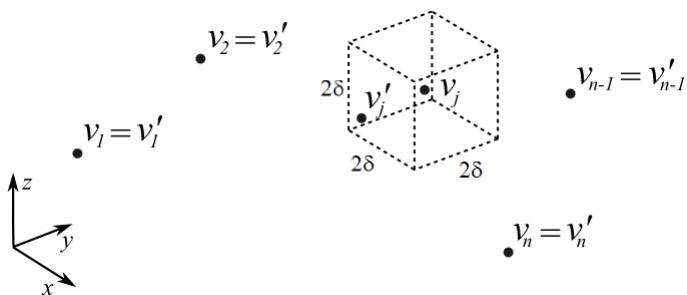


Figura 6.1 Una solución vecina de V es V' .

El problema consiste entonces en encontrar una solución óptima global, es decir, una $V^* \in \mathcal{Y}$ tal que $f(V^*) \geq f(V)$ para toda $V \in \mathcal{Y}$. El primer paso para resolver este problema consiste en encontrar una solución inicial. En este caso tenemos básicamente dos alternativas: una solución inicial aleatoria o una solución inicial determinada por un método constructivo. En la parte experimental examinaremos ambas alternativas, pero por ahora veamos un método que proponemos para generar eficientemente una solución inicial de manera constructiva. Este método se denomina *partición tetraédrica recursiva*.

6.2. Métodos constructivos

Existen varios métodos constructivos para generar soluciones iniciales para el problema de diversificación Max-Min. Erkut [Erk90] propone una heurística constructiva glotona que genera una solución inicial, primeramente, al escoger los dos elementos con la distancia más grande entre ellos. Luego, en las siguientes iteraciones el elemento que incrementa la función objetivo al máximo es añadido a la solución parcial. Esto se hace hasta que se acumulan los n elementos. En [GKD98] se proponen cuatro heurísticas constructivas, las dos primeras se aplican únicamente a conjuntos en espacios euclídeos, ya que están basadas en el concepto de centro de gravedad de un conjunto. Las otras dos heurísticas constructivas tienen que ver con la distancia entre a y un conjunto X definida como

$$d(a, X) = \sum_{b \in X} d(a, b),$$

donde en un primer paso se escoge un elemento inicial y se van seleccionando consecutivamente elementos del espacio de búsqueda de manera que la distancia entre tal elemento y la solución parcial acumulada hasta el momento sea la mayor posible. La otra heurística constructiva comienza seleccionando todos los elementos del espacio de búsqueda, después de lo cual comienza a descartar elementos sucesivamente uno por uno, aquellos cuya distancia al resto del conjunto sea mínima. En ambos casos el proceso termina hasta haber acumulado n elementos para la solución inicial.

Estos métodos iniciales constructivos son aplicables en general a una gran cantidad de problemas, sin embargo, en el problema que nos ocupa no resultan métodos eficaces para generar una solución inicial pues resultan ser muy costosos en términos de tiempo computacional, ya que el espacio de búsqueda es muy grande, y por tratarse de generar una solución inicial no deseamos dedicarle demasiado tiempo ni refinamiento. En cambio, el método de partición tetraédrica recursiva funciona bien puesto que la naturaleza del problema es geométrica, obteniéndose soluciones iniciales en apenas un par de segundos para valores de n cercanos a 100.

6.3. Generación de mallas y descomposiciones tetraédricas recursivas

Un tema que ha cobrado gran interés desde hace algunos años en el área de la geometría computacional es el de *técnicas de mallado*, donde el objetivo básico es construir una malla que aproxime a un dominio, ya sea en dos o en tres dimensiones, y que cumpla con algunos criterios de calidad que pueden variar dependiendo de la aplicación, como por ejemplo, que la forma de los elementos de la malla sea regular. La motivación para la utilización de mallas tiene que ver con el hecho de que resulta casi imposible hacer representaciones exactas de entornos físicos reales para modelar objetos en dos o tres dimensiones. Para resolver estas situaciones es preciso utilizar aproximaciones de *buenas calidad* que permitan trabajar con tiempos de cómputo razonables.

En términos informales, una malla es una colección de elementos simples que se yuxtaponen para construir la aproximación de un objeto sólido, normalmente complejo. Dos tipos de mallas populares para dominios sólidos son las tetraédricas y las hexaédricas, cuyos elementos son tetraedros y hexaedros (seis caras con cuatro aristas cada una), éstas últimas son famosas porque su implementación por medio de estructuras de datos resulta natural y relativamente sencilla para los programadores. Por otro lado, algunos autores consideran que la manera más natural para aproximar un sólido es descomponiéndolo por medio de tetraedros, ya que las mallas tetraédricas presentan algunas ventajas sobre las hexaédricas, pero profundizar en el tema queda fuera del enfoque de este trabajo, véase [Geo99, BKK11, Epp01] para mayor referencia.

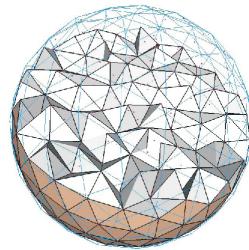


Figura 6.2 Aproximación de una esfera mediante una malla tetraédrica [YWLL11].

El método que veremos a continuación está en concordancia con un tipo de técnica automatizada para generar mallas denominado *subdivisión recursiva del dominio*. Una descripción más detallada sobre el tema se puede encontrar en [Moo92]. La idea principal consiste en subdividir el dominio utilizando alguna forma elemental, como los tetraedros por ejemplo. Una clase importante de este tipo de subdivisión se logra a través de triangulaciones. Una triangulación del cubo particularmente interesante es la *triangulación de Kuhn* [Kuh60]. Para construir esta triangulación, primeramente se selecciona una de las cuatro diagonales principales del cubo; luego, se buscan los recorridos mínimos sobre las aristas del cubo que unen a los dos vértices de esta diagonal. Por último,

las envolventes convexas de los caminos encontrados, son los tetraedros que definen la partición del cubo.

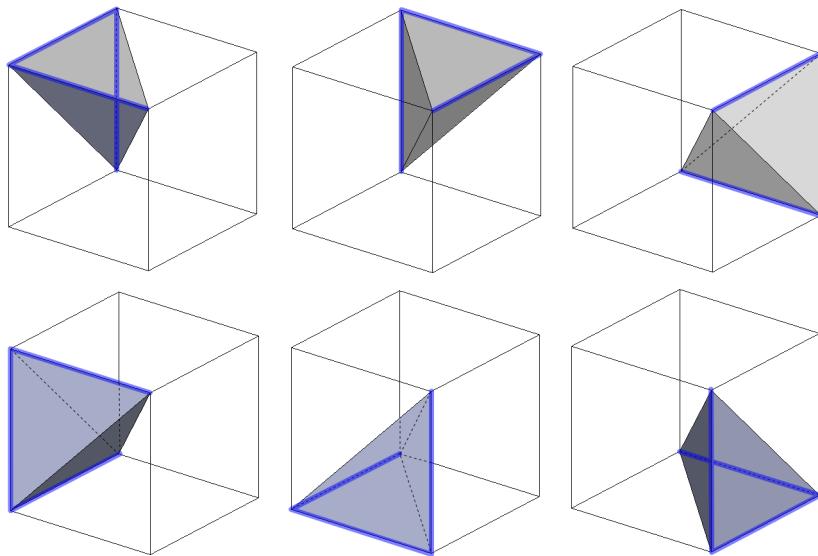


Figura 6.3 Cada tetraedro se define como la envolvente convexa de los cuatro vértices que se encuentran en cada recorrido mínimo, que conecta a ambos vértices de la diagonal principal.

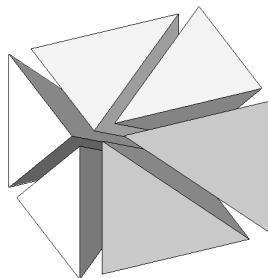


Figura 6.4 La triangulación de Kuhn del cubo es una malla tetraédrica, la primera etapa para la construcción de la solución inicial para nuestro problema de optimización.

Hasta este momento el cubo está particionado en seis tetraedros que tienen en común ciertos vértices y aristas. La triangulación de Kuhn es una malla tetrahédrica con ocho vértices y diecinueve aristas, así que si $n \leq 8$ se escogerán los puntos de la solución inicial de manera aleatoria de entre los ocho vértices de esta malla, es decir, de entre los ocho vértices del cubo. El siguiente paso en la construcción de nuestra solución inicial,

cuando $n > 8$, es subdividir recursivamente la triangulación de Kuhn, donde cada nuevo vértice que se genere será un punto más de la solución inicial hasta que se alcancen n vértices.

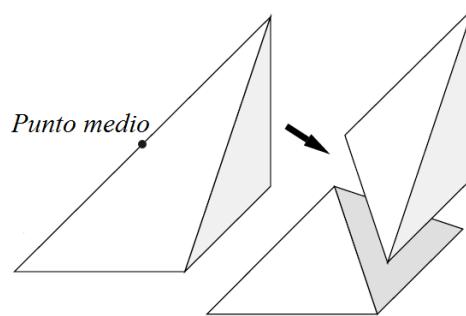


Figura 6.5 El proceso de subdivisión de un tetraedro por el punto medio de su arista más grande. Gráfico por [Moo92].

El algoritmo consiste en subdividir los tetraedros que comparten la arista más grande de la malla por el punto medio de dicha arista, lo que Brandts denomina *refinamiento verde*[BKK11]. Así pues, se va generando un nuevo vértice en la malla y los vértices vienen a ser los puntos de la solución inicial. El algoritmo se describe en el seudocódigo de la figura 6.6. Ejemplos de particiones tetraédricas para algunos valores de n se encuentran en la figura 6.7.

```

Inicio :
 $T \leftarrow$ Lista de tetraedros iniciales;
 $V \leftarrow$ Lista de los vértices de T;
 $k \leftarrow n - 8;$ 
    for  $i$  in range  $[1, k]$  :
         $AM \leftarrow$ Arista máxima de los tetraedros en  $T$ ;
         $T \leftarrow P(T, AM);$ 
         $mp \leftarrow$ Punto medio de  $AM$ ;
         $V \leftarrow V \cup mp;$ 
         $i++;$ 
Fin;
 $P(T, A)$  es una función que subdivide en dos tetraedros a todos los elementos de  $T$  que comparten la arista  $A$ , por el punto medio de  $A$ . Se devuelve una lista actualizada de los tetraedros.

```

Figura 6.6 Seudocódigo del proceso de partición tetraédrica propuesto para encontrar una solución inicial.

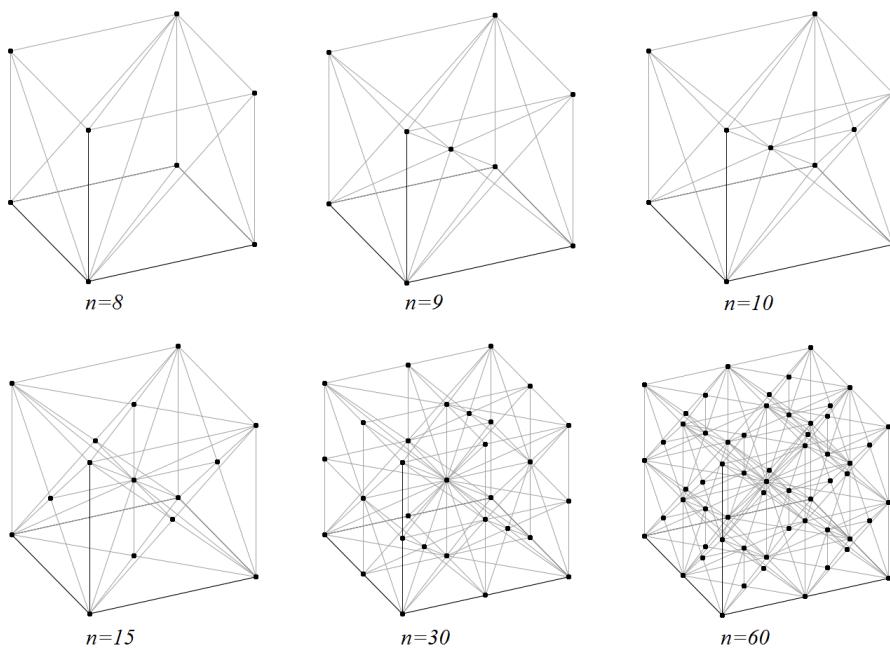


Figura 6.7 Particiones tetraédricas para distintos valores de n . Las líneas entre los puntos son las aristas de la malla.

Es importante hacer la observación que un buen lugar para comenzar a colocar puntos en un poliedro, de manera que su distancia mínima sea lo más grande posible, es en su frontera [Ste09, SP09]. Esto puede apreciarse claramente ya que entre más dispersos estén los puntos, su distancia mínima se verá incrementada. Así pues la intuición nos lleva a predecir que una buena cantidad de los puntos se ubicarán en la frontera del cubo. En este sentido, el algoritmo que hemos propuesto para la solución inicial funciona adecuadamente puesto que distribuye los puntos de manera más o menos uniforme en el cubo y una buena cantidad de vértices se encuentran justo en la frontera del cubo. Proponemos que el iniciar con una solución factible de buena calidad, es decir, con una buena distribución de los puntos de la solución inicial a bajo costo influirá al reducir el tiempo de cómputo cuando se implemente la técnica heurística para buscar óptimos globales. En la parte experimental se hará un análisis para demostrar la validez de este argumento.

6.4. Metaheurística con búsqueda local

Una vez que se tiene completada la primera etapa de la construcción de una solución inicial proseguimos con la búsqueda local. Podemos distinguir básicamente dos enfoques distintos para la búsqueda local. El primero de ellos es el propuesto por [Gho96] y

por [Erk90], el BLS (Búsqueda Local exhaustiva de Mejor intercambio) donde en cada iteración se examina exhaustivamente la vecindad de la solución en curso para encontrar el mejor intercambio posible con una solución vecina. Este enfoque viene siendo el mismo que el del algoritmo de mejora sucesiva, por lo cual queda descartado. El otro enfoque es el FLS (Búsqueda Local de Primer mejor intercambio) [RR07], donde se examinan los elementos de la solución en curso cuya separación es la distancia mínima, luego se realiza una búsqueda local en su vecindad y se realiza un intercambio con la primera solución vecina que mejore la función objetivo en curso.

En nuestra implementación del algoritmo utilizaremos una variante de la segunda estrategia. Una solución vecina se propone al escoger aleatoriamente un punto en la δ -caja vecindad, de alguno de los dos vértices de la arista de menor distancia en la malla tetraédrica, es decir, de aquellos dos elementos de la solución inicial cuya distancia sea igual a la función objetivo en curso. La selección de uno de los dos puntos también es al azar.

Sea $V = \{v_1, v_2, \dots, v_n\}$ una solución inicial obtenida por el método de partición tetraédrica recursiva. Sea $f(V) = D$ la función objetivo inicial, existen entonces $v_i, v_j \in V$ tal que $d(v_i, v_j) = D$, se selecciona al azar uno de los dos puntos, sin pérdida de generalidad elegimos a v_j . Sea $\delta > 0$, entonces

$$\mathcal{N}_\delta(V) = V \cup C(v_j, \delta).$$

Vamos a guardar en cada iteración la mejor solución hasta el momento como V_{Max} , en la primera iteración asignamos por defecto $V_{Max} := V$. El siguiente paso es encontrar una solución vecina $V' \in \mathcal{N}_\delta(V)$, para lo cual se genera un punto v'_j aleatoriamente en la δ -caja vecindad de v_j y se obtiene así $V' = \{v'_1, \dots, v'_j, \dots, v'_n\}$. Es importante revisar que v'_j se encuentre en \mathcal{C}_{RGB} , así que se descartan las soluciones vecinas aleatorias que no satisfagan esta condición, véase figura 6.8.

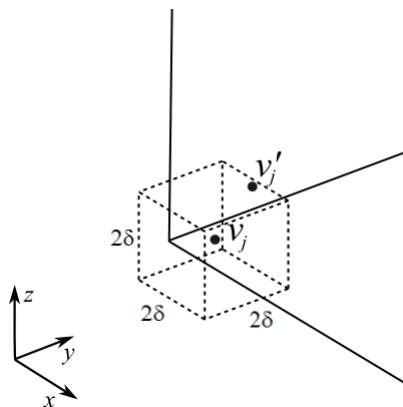


Figura 6.8 En el gráfico, v_j es un punto cercano a la frontera del cubo RGB, el punto aleatorio v'_j debe estar dentro de la δ -caja vecindad de v_j , así como dentro del cubo.

Ahora comparamos las funciones objetivo. Si $f(V') > f(V)$ entonces se acepta la solución vecina y se reescribe $V := V'$, la nueva solución para la siguiente iteración. En este caso, al haberse mejorado la función objetivo en curso, comparamos el valor de $f(V')$ con $f(V_{Max})$ y si resulta que $f(V') > f(V_{Max})$ entonces se reescribe también $V_{Max} = V'$. Ahora bien, si $f(V') < f(V)$ se tiene que revisar si $f(V') - f(V)$ pasa el criterio de aceptación, tanto para el caso del algoritmo de recocido simulado como el de umbral de aceptación. Veamos un ejemplo para ilustrar lo anterior, en las figuras 6.9, 6.10, 6.11, 6.12 y 6.13.

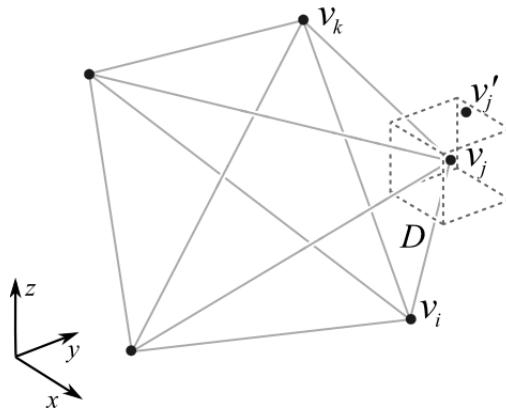


Figura 6.9 Sea V solución inicial. Por ser la primera iteración asignamos por defecto $V_{Max} := V$. Existen v_i, v_j en V cuya distancia es D . Se genera aleatoriamente un punto v'_j en la δ -caja vecindad de v_j para obtener una solución vecina V' . En este ejemplo $f(V') > f(V)$ así que se acepta el movimiento y se reescribe $V := V'$.

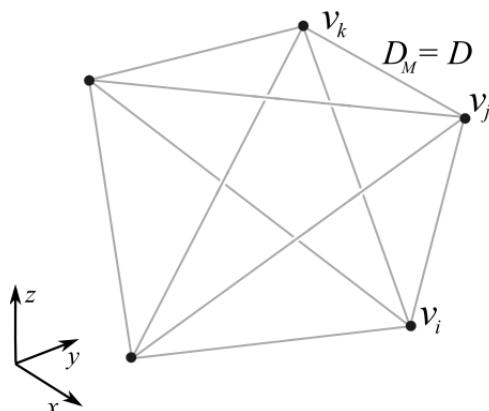


Figura 6.10 Por haberse mejorado la función objetivo (Figura 6.9), se compara $f(V)$ con $f(V_{Max})$ (donde V es la nueva solución en curso), y como $D = f(V) > f(V_{Max}) = D_M$ entonces se reescribe $V_{Max} := V$ y $D_M := D$.

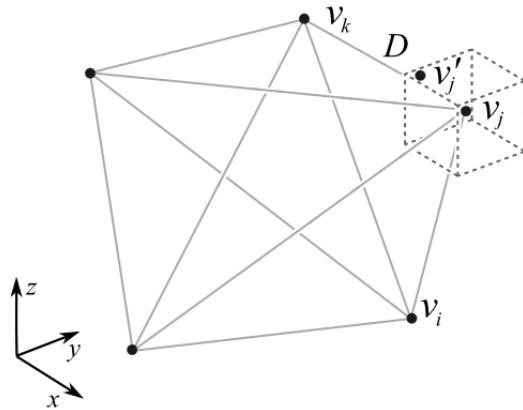


Figura 6.11 En la siguiente iteración se propone un movimiento. Al comparar las funciones objetivo se tiene que $f(V') < f(V)$, es decir, se trata de un empeoramiento. Supongamos además que Δf no pasa el criterio de aceptación, entonces se rechaza el movimiento.

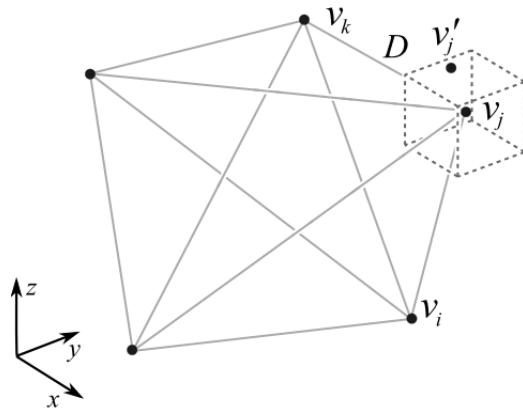


Figura 6.12 Se propone un nuevo movimiento. Otra vez $f(V') < f(V)$, pero supongamos ahora que Δf sí pasa el criterio de aceptación. Se reescribe $V := V'$ como la nueva solución en curso. No hay necesidad de comparar $f(V')$ con $f(V_{Max})$ en este caso.

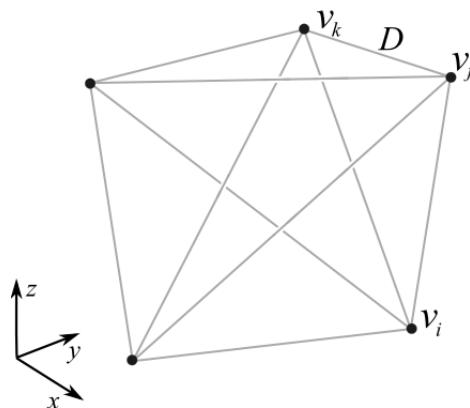


Figura 6.13 Después de dos iteraciones completas.

Veamos por último, figura 6.14, el seudocódigo que describe en términos generales la implementación del algoritmo de recocido simulado y el de umbral de aceptación, siendo la única diferencia la manera en que se aceptan soluciones vecinas que empeoran a la función objetivo.

Inicio :

$V \leftarrow$ Se genera una solución inicial;

$t_0 \leftarrow$ Se define la temperatura inicial;

$\varepsilon \leftarrow$ Se define el cero virtual;

$V_{MAX} \leftarrow V$;

While $t_0 > \varepsilon$:

While No se satisface condición para reducir la temperatura;

$V' \leftarrow$ Vecino aleatorio de V ;

$\Delta \leftarrow f(V') - f(V)$;

if $\Delta \geq 0$:

$V \leftarrow V'$;

if $f(V) > f(V_{MAX})$:

$V_{MAX} \leftarrow V$;

if $\Delta < 0$:

if Δ pasa el criterio de aceptación:

$V \leftarrow V'$;

End While;

$t_0 \leftarrow$ Se reduce la temperatura

End While;

Figura 6.14 Seudocódigo del algoritmo de recocido simulado y umbral de aceptación.

Capítulo 7

Aplicación del método de optimización en el espacio de color CIE L*u*v*

Habiendo sentado las bases de los algoritmos de umbral, así como su implementación en una metaheurística que combina una etapa constructiva seguida de una etapa de búsqueda local adaptado para la resolución del problema de n -dispersión en el cubo, trasladaremos ahora el cúmulo de ideas generado al espacio CIE L*u*v*. Como se explicó en el primer capítulo, el espacio de color CIE RGB no es un espacio perceptualmente uniforme, por lo tanto, la misma distancia no representará la misma disimilitud perceptual para distintas parejas de colores, por lo que es necesario trabajar en un espacio de color donde la percepción visual sea uniforme, en nuestro caso, el espacio de color que utilizaremos es CIE L*u*v* definido en el primer capítulo. Ahora, puesto que los resultados serán interpretados en un monitor de computadora, necesitamos trabajar con los colores que pueden ser representados en pantalla, es decir, nos limitaremos a los colores del cubo RGB. Utilizaremos las transformaciones descritas en las secciones 1.2, 1.3, 1.4 y 1.5, como lo ilustra la figura 7.1, para obtener la imagen de \mathcal{C}_{RGB} en el espacio CIE L*u*v*. A este conjunto lo denominaremos $\mathcal{C}_{L^*u^*v^*}$ (figura 7.2), es decir

$$\mathcal{C}_{L^*u^*v^*} = T_2(T_1(\mathcal{C}_{RGB})).$$



Figura 7.1 Transformaciones entre los espacios de color descritas en el capítulo 1.

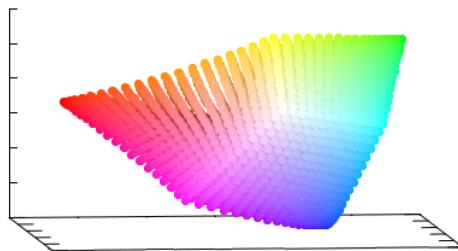


Figura 7.2 Vista de una aproximación de $\mathcal{C}_{L^*u^*v^*}$.

7.1. Formulación del problema

La formulación del problema de optimización combinatoria en el espacio CIE L*u*v* queda de la siguiente manera. Sea $\mathcal{Y} = \mathcal{Y}(n, \mathcal{C}_{L^*u^*v^*})$ el conjunto de todas las combinaciones posibles de n elementos en $\mathcal{C}_{L^*u^*v^*}$. Sea f la función definida como

$$f(V) = \min_{x,y \in V} d(x,y),$$

donde d es la distancia euclídea y $V \in \mathcal{Y}$. La función de vencidad $\mathcal{N}_\delta(V)$ está dada por la definición 6.3. Sean $v_{i_{min}}$ y $v_{j_{min}}$ los vértices de una arista cuya longitud representa la distancia mínima en V , es decir, $f(V) = d(v_{i_{min}}, v_{j_{min}})$. Entonces el objetivo es encontrar $V^* \in \mathcal{Y}$ tal que

$$f(V^*) = \max_{V \in \mathcal{Y}} f(V).$$

7.2. Solución inicial

Para la generación de la solución inicial utilizaremos el método de partición tetraédrica recursiva discutido en la sección 6.3, adaptándolo convenientemente. De la misma manera que en el caso del cubo, utilizaremos ocho vértices iniciales para comenzar la subdivisión tetraédrica de $\mathcal{C}_{L^*u^*v^*}$, cuando $n > 8$. Estos ocho vértices son las imágenes correspondientes en $\mathcal{C}_{L^*u^*v^*}$ de los vértices del cubo RGB. La tabla 7.1 indica tales correspondencias

Color	\mathcal{C}_{RGB}	$\mathcal{C}_{L^*u^*v^*}$
Negro	(0,0,0)	(0,0,0)
Rojo	(255,0,0)	(53.24,175.02,37.76)
Verde	(0,255,0)	(87.73,-83.06,107.41)
Azul	(0,0,255)	(32.29,-9.40,-130.33)
Amarillo	(255,255,0)	(97.13,7.71,106.80)
Magenta	(255,0,255)	(60.32,84.08,-108.66)
Cyan	(0,255,255)	(91.11,-70.46,-15.17)
Blanco	(255,255,255)	(100,0.01,0.02)

Tabla 7.1 Correspondencia entre vértices de \mathcal{C}_{RGB} y $\mathcal{C}_{L^*u^*v^*}$.

La figura 7.3 muestra los vértices iniciales para la partición tetraédrica tanto en \mathcal{C}_{RGB} como en $\mathcal{C}_{L^*u^*v^*}$.

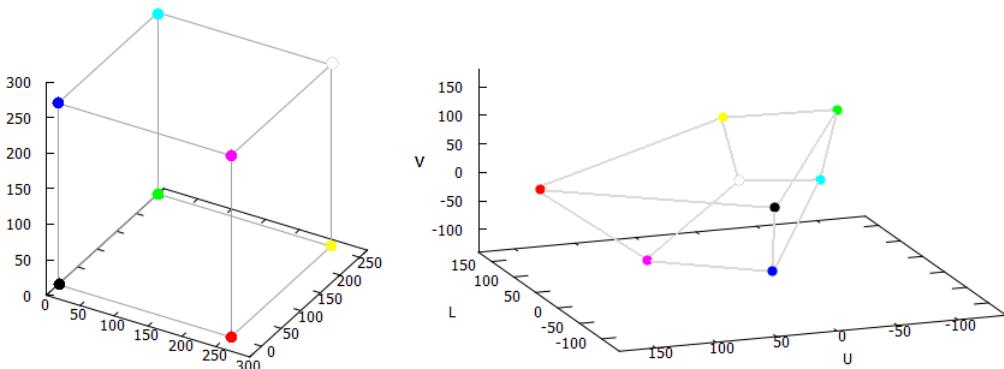


Figura 7.3 En este gráfico se muestran los ocho vértices de \mathcal{C}_{RGB} y sus imágenes correspondientes en $\mathcal{C}_{L^*u^*v^*}$.

Al generar la solución inicial con la partición tetraédrica recursiva, puede ser que algunos puntos queden fuera de $\mathcal{C}_{L^*u^*v^*}$, esto lo corregimos haciendo una inspección de la solución inicial. Se revisa la imagen inversa de los puntos y se verifica que estén dentro de \mathcal{C}_{RGB} . Si la imagen inversa de un punto de la solución inicial no está en \mathcal{C}_{RGB} , ésta se intercambia por el punto más cercano en \mathcal{C}_{RGB} y hacemos también el intercambio correspondiente de las imágenes en $\mathcal{C}_{L^*u^*v^*}$, así ajustamos la solución inicial para que sea factible. En la figura 7.4 se muestran las mallas que representan la solución inicial en $\mathcal{C}_{L^*u^*v^*}$, por el método de partición tetraédrica recursiva, para $n = 8$ y $n = 15$.

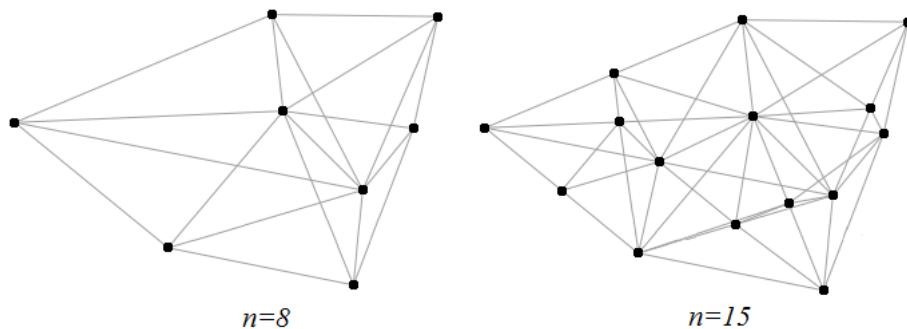


Figura 7.4 Soluciones iniciales en $\mathcal{C}_{L^*u^*v^*}$ generadas con el método de partición tetraédrica recursiva.

Una vez que tenemos la solución inicial procedemos a utilizar el algoritmo descrito en la sección 6.4 para mejorar la ubicación de los puntos iniciales. Cada vez que se proponga una solución vecina en la etapa de búsqueda local, se verificará que $T_1^{-1}(T_2^{-1}(x)) \in \mathcal{C}_{RGB}$, de lo contrario se desecha dicha solución vecina.

En la siguiente parte de este trabajo se explicará la manera en que se eligieron los parámetros iniciales, así como los experimentos llevados a cabo para tal fin. En esta parte experimental se implementaron ocho variantes del algoritmo de recocido simulado y se hizo una comparación de desempeño entre ellos para distintos valores de n . Se utilizaron pruebas estadísticas no paramétricas para el análisis de los datos recolectados. Por último, se utilizó la mejor variante del algoritmo para calcular soluciones para algunos valores de n y mostrar las paletas de colores gráficamente.

Parte II

Experimentos computacionales

Capítulo 8

Selección de parámetros

El algoritmo de recocido simulado y su variante umbral de aceptación han mostrado ser efectivos para resolver distintos problemas de optimización del tipo *NP-duros* [Lin93, Cer85, PPR⁺03, Kin92]. Sin embargo, una correcta aplicación de este algoritmo requiere un manejo adecuado y efectivo de los distintos parámetros que lo gobiernan. En nuestra implementación los parámetros son:

- δ , el tamaño de la vecindad. Este parámetro se refiere a las dimensiones geométricas de la vecindad de la solución donde se buscarán vecinos que la reemplacen. Vecindades muy pequeñas harán que el algoritmo se desempeñe más lento para explorar el espacio de soluciones factibles, mientras que, vecindades muy grandes pueden propiciar que las soluciones vecinas propuestas no se acepten porque no pasan el criterio de aceptación, debido a su lejanía.
- t_0 , la temperatura inicial. La selección de este parámetro es muy importante ya que nos dice qué tanto puede empeorar una solución en un principio. La selección adecuada de una estrategia de enfriamiento es vital para lograr el efecto metaheurístico deseado: escapar de óptimos locales. Un valor inicial muy grande de t_0 aceptará cualquier tipo de empeoramiento y no redundará en una mejora de la función objetivo. Con un valor inicial muy pequeño se corre el riesgo de que el algoritmo no sea capaz de escapar de algún óptimo local.
- λ , el tamaño del lote. Este parámetro está relacionado con la longitud de las cadenas finitas de Markov homogéneas, su función se describió en la sección 4.3.
- φ , factor de enfriamiento. Como su nombre lo indica, está relacionado con la función de decremento g , encargada de reducir la temperatura al finalizar cada cadena de longitud finita de Markov homogénea.

- θ , cardinalidad de la δ -caja vecindad. Al trabajar en \mathbb{R}^3 las vecindades euclidianas tienen cardinalidad infinita pero, por razones obvias, la cantidad de vecinos a explorar será finita.

8.1. Tamaño de la δ -caja-vecindad.

Dada la naturaleza geométrica del problema que tratamos en este trabajo, al referirnos al tamaño de la δ -caja vecindad podemos pensar en dos cosas: las dimensiones geométricas de la δ -caja vecindad y la cantidad de puntos θ contenidos en ella. Ambos parámetros se determinaron experimentalmente como se explica más adelante.

Para la elección adecuada del sistema de vecindades, definido en la sección 3.1, se tomaron en cuenta algunas propiedades, como lo menciona [GP97]:

“Diferentes sistemas de vecindades tienen diferentes propiedades, lo que las puede hacer más o menos idóneas dependiendo del problema en cuestión. Vale la pena mencionar dos en particular:

- *Cardinalidad de la vecindad, es decir, la cantidad de vecinos para una solución. Vecindades pequeñas son preferibles desde el punto de vista de la velocidad de la búsqueda local. Sin embargo, si la vecindad es muy pequeña la solución final puede ser de poca calidad.*
- *Facilidad para calcular valores de nuevas soluciones. Es muy favorable que al proponerse una nueva solución adyacente, el sistema de vecindades permita una actualización rápida de la función objetivo, en lugar de tener que hacer una recalculación completa.”*

En concordancia con [GW88], la preocupación primordial para seleccionar un *sistema de vecindades* adecuado es ¿qué tan rápido convergerá el algoritmo? Tomando en cuenta esta situación, hemos planteado un sistema de vecindades variable, donde el valor de δ será lo suficientemente grande al principio para permitir que el algoritmo encuentre mejoras con suficiente rapidez, pero conforme el algoritmo vaya avanzando, el tamaño de δ se irá reduciendo paulatinamente para encontrar soluciones más refinadas que mejoren la función objetivo. El valor inicial se denomina δ_0 . El valor de θ lo mantendremos constante durante la ejecución del algoritmo, la justificación para ello la daremos en la sección 8.3.

La regla que seguimos es sencilla: si al examinar 2θ soluciones vecinas de la solución en curso, ninguna de ellas pasa el criterio de aceptación para reemplazarla, entonces se procede a reducir el tamaño de la vecindad por mitad y se continúa el algoritmo con vecindades de tamaño $\frac{\delta}{2}$; si se vuelve a repetir la situación donde no se aceptaron 2θ soluciones vecinas, se reduce el tamaño de las vecindades a la mitad; y así sucesivamente. Las soluciones vecinas se construyen al seleccionar aleatoriamente un punto en la δ -caja-vecindad, de uno de los dos vértices de la arista que representa una distancia

mínima de la solución en curso, y puesto que acordamos explorar hasta θ puntos en cada δ -caja-vecindad, se permitirán un máximo de 2θ propuestas en total antes de reducir el valor de δ , en caso de no haber una aceptación.

Ahora bien, ¿cuál es el valor que escogimos para δ_0 ? Intuitivamente éste parámetro debe tener cierta dependencia del número de puntos n , es decir, para un valor bajo de n las distancias entre los puntos de la solución inicial, generada por el método de partición tetraédrica recursiva, serán más grandes que para un valor alto de n . Así, entre menos puntos se coloquen en el espacio en cuestión, se esperaría que el valor de δ_0 fuese mayor para poder explorar con mayor rapidez el espacio de soluciones factibles. Por otro lado, se determinaron experimentalmente cotas mínimas y máximas para las coordenadas L^*, u^*, v^* de los puntos de $\mathcal{C}_{L^*u^*v^*}$, quedando los puntos delimitados por una caja de $100 \times 258 \times 241.5$ unidades. Entonces, tenemos que $\delta_0 \leq 50$, pues la δ_0 -caja vecindad más grande posible, dentro de los límites encontrados, tendría como dimensiones $100 \times 100 \times 100$. Dado n , era de interés encontrar un valor adecuado para δ_0 , por lo que se propuso un sencillo mecanismo empírico para determinar este parámetro inicial, pero antes, veamos una consideración que se hizo para determinar los valores de n que se utilizaron en los experimentos.

En [CC81] se menciona que para que dos colores en el espacio CIE $L^*u^*v^*$ se puedan distinguir sin problema, éstos deben estar a una distancia no menor de 40 unidades. Los resultados experimentales obtenidos indican que, para valores de n mayores que 50, la condición del límite de las 40 unidades no se satisface. Por lo anterior, sólo haremos experimentos para $n \leq 50$. Por otro lado, debido a la gran cantidad de cálculos a realizar, tomaremos una lista reducida de valores de n , $N = \{10, 15, \dots, 50\}$.

Dado $n \in N$, se utilizó el algoritmo de mejora sucesiva descrito en la sección 3.3 con distintos valores de δ_0 tomados del intervalo $[0, 50]$. Debido a la gran cantidad de cálculos a realizar, se discretizó el intervalo $[0, 50]$ del siguiente modo: $\delta_0 = 5, 10, \dots, 50$. La razón para haber utilizado el algoritmo de mejora sucesiva para determinar los valores iniciales de δ_0 , es porque no se requiere de otros parámetros para que el algoritmo funcione, excepto por δ_0 y θ . Se utilizó este algoritmo para determinar valores de δ_0 , para cada n , que permitieran al algoritmo llegar a una solución cuya función objetivo tuviera el valor más alto, en el menor tiempo posible. Cabe señalar que en este experimento el tamaño de δ_0 no se redujo durante la ejecución del algoritmo.

Ahora bien, ¿cuál es el valor de θ utilizado? No se tiene en principio ninguna referencia para este parámetro, así que se procedió a tomar los valores $\theta = 10, 20, \dots, 100$ para cada pareja (n, δ) . También es importante considerar que, debido a la aleatoriedad involucrada en el algoritmo, fue conveniente hacer varias ejecuciones por cada terna (n, δ_0, θ) y se tomó el promedio de las funciones objetivo. Por razones de ahorro en el tiempo de cómputo se escogió hacer sólo cinco ejecuciones del algoritmo en cada caso. También es importante mencionar que por cada ejecución se realizó un número fijo de iteraciones (este fue el criterio de paro utilizado) por tratarse del algoritmo de mejora sucesiva. Se realizaron 500 iteraciones por ejecución. A partir de la información reca-

bada se hizo un análisis estadístico para determinar los valores para δ_0 y θ . Las tablas correspondientes a los experimentos se encuentran en los apéndices B.1 y B.2. La tabla 8.1 contiene un resumen de la información recabada.

n	10	15	20	25	30	35	40	45	50
δ_0	25	15	20	25	20	20	10	10	10
θ	20	20	20	20	20	20	20	20	20

Tabla 8.1 Valores obtenidos para los parámetros δ_0 y θ .

8.2. Valor para la temperatura inicial t_0

De acuerdo con [Kin92] y [LM86] “el valor de t_0 debe ser mucho más grande que la diferencia entre el peor y el mejor valor posible de la función objetivo”, para el algoritmo de recocido simulado, y en su análisis se propone que $t_0 = \max\{d(A, B) \mid A, B \in \mathcal{S}\}$, donde \mathcal{S} es un conjunto de n lugares en el problema de *localización óptima de instalaciones perjudiciales*. Al no conocer en principio el mejor valor posible de la función objetivo para cada n , tuvimos que tomar un enfoque distinto para determinar t_0 , mediante una *búsqueda binaria*, descrita por el seudocódigo de la figura 8.1.

Inicio :

Se fijan valores para n, δ, It ;

Se define rango de porcentaje de aceptación $[r_{min}, r_{max}]$;

Se escoge temperatura inicial $T_0 = 1$;

Se genera solución inicial V_0 ;

While Porcentaje de aceptación fuera del rango $[r_{min}, r_{max}]$:

for i in range $[1, It]$:

 Aplicar algoritmo de umbral de aceptación a V_0 ;

if movimiento es aceptado:

$acep \leftarrow acep + 1$;

$pa \leftarrow 100 * acep / It$;

if $pa < r_{min}$:

$T_0 \leftarrow 2T_0$ (si no hubo antes otro porcentaje de aceptación $pa' > r_{min}$);

$T_0 \leftarrow (T_0 + T'_0) / 2$ (si hubo antes un porcentaje de aceptación $pa' > r_{min}$);

if $pa > r_{max}$:

$T_0 \leftarrow T_0 / 2$ (si no hubo antes otro porcentaje de aceptación $pa' < r_{max}$);

$T_0 \leftarrow T_0 / 2$ (si hubo antes un porcentaje de aceptación $pa' < r_{max}$);

End While;

Figura 8.1 Seudocódigo del algoritmo de búsqueda binaria para determinar t_0 .

Como se mencionó en la sección 4.3, el porcentaje de aceptación inicial se fijó en $95 \pm 0.1\%$. Se realizaron cinco ejecuciones del algoritmo de búsqueda binaria para cada valor de n , tomando el promedio de las temperaturas finales, y realizando 500 iteraciones en cada paso al aplicar el algoritmo umbral de aceptación a temperatura constante. La temperatura la iniciamos en $t = 1$. A continuación, en la tabla 8.2 se muestra una de las ejecuciones realizadas para determinar t_0 en el caso $n = 20$, $\delta_0 = 20$.

Paso	Temperatura	Aceptaciones (%)
1	1.0	9.6
2	2.0	12.4
3	4.0	31.4
4	8.0	65.4
5	16.0	95.8
6	12.0	89
7	14.0	93.2
8	15.0	95.4
9	14.5	93.4
10	14.75	95.0

Tabla 8.2 Ejemplo de la ejecución del algoritmo de búsqueda binaria para determinar t_0 .

Las temperaturas iniciales calculadas a partir de este método se encuentran en la tabla 8.3.

n	10	15	20	25	30	35	40	45	50
t_0	20.13	13.09	15.27	15.72	13.78	11.57	8.51	8.18	8.14

Tabla 8.3 Temperaturas iniciales determinadas con el algoritmo de búsqueda binaria.

8.3. Número de lote λ y factor de enfriamiento φ

Para el parámetro λ no se cuenta con información previa que nos diga la manera de determinarlo. Por otro lado, en la literatura se encuentran valores sugeridos para el factor de enfriamiento $\varphi \in [0.89, 0.90]$, [GvdHTG06, MGDP13, LM86]. De manera similar a como se determinaron los valores de θ y δ_0 , realizamos una serie de experimentos con parejas de valores (λ, φ) . Como resultado se obtuvo, para cada n , una matriz con los valores de las funciones objetivo para cada pareja (λ, φ) . Para λ se hicieron pruebas con los valores 20, 40, 60, 80 y 100, mostrando mejores resultados en cuanto mayor era el valor de λ , sin embargo, los tiempos de cómputo se elevaban de manera considerable sin que hubiera una mejora importante en la función objetivo. Se optó por tanto trabajar con el valor intermedio $\lambda = 60$. Asimismo, para definir el valor de φ se encontró que

las mejores soluciones se obtenían con un factor de enfriamiento alto, cercano a 1, pero por razones de tiempo de cálculo y eficiencia se decidió optar por el valor $\varphi = 0.90$, que arroja valores cercanos a los correspondientes a $\varphi = 0.95$, pero con un ahorro considerable de tiempo. Las tablas con la información y los detalles al respecto se encuentran en los apéndices B.3 y B.4.

Capítulo 9

Resultados experimentales

En este último capítulo describiremos los experimentos computacionales llevados a cabo, utilizando los parámetros que se calcularon en el capítulo 8 y el algoritmo de umbral propiamente descrito en la sección 6.4. Se tomaron en cuenta algunas variantes del algoritmo para su comparación, mediante el empleo de dos pruebas estadísticas no paramétricas, descritas en el apéndice B, para sacar conclusiones importantes. Para los experimentos se utilizó una computadora HP Pavilion g4 Notebook PC con procesador AMD E2-3000M a 1.8 Ghz, 4 GB de memoria RAM y sistema operativo Windows 7. El lenguaje de programación utilizado es el de Python.

El objetivo de este capítulo será analizar el desempeño de ocho variantes del algoritmo de recocido simulado para determinar aquél con el mejor desempeño. Esto lo haremos al comparar los resultados obtenidos en los experimentos computacionales, y mediante la aplicación de pruebas estadísticas no paramétricas para sacar conclusiones importantes. La manera de hacer el análisis para comparar las variantes del algoritmo está en acuerdo con [MGDP13], donde se hace una comparación exhaustiva de 10 heurísticas y 20 metaheurísticas para la resolución del Problema de Máxima Dispersión.

Las variantes están determinadas por las distintas combinaciones que se pueden generar a partir de los siguientes elementos: tipo de algoritmo, tipo de solución inicial y tipo de función de decremento. La nomenclatura para las variantes surge de la siguiente convención:

- **RS:** Recocido simulado.
- **UA:** Umbral de aceptación.
- **0:** Solución inicial por el método de partición tetraédrica recursiva.
- **1:** Solución inicial aleatoria.
- **g:** función de decremento para la temperatura $g(\varphi, m) = \varphi^m t_0$.

- **h:** función de decremento para la temperatura $h(\varphi, m) = \frac{1}{2}\varphi^m t_0 \cdot (\operatorname{sen}(m) + 2)$.

Por ejemplo, la variante **UA-g-0** es algoritmo de umbral de aceptación con función de decremento g y solución inicial dada por la partición tetraédrica recursiva. Los parámetros finales utilizados están dados en la tabla 9.1.

<i>n</i>	δ_0	t_0	θ	λ	φ
10	25	20.129	20	60	0.9
15	15	13.088	20	60	0.9
20	20	15.265	20	60	0.9
25	25	15.724	20	60	0.9
30	20	13.777	20	60	0.9
35	20	11.567	20	60	0.9
40	10	8.511	20	60	0.9
45	10	8.176	20	60	0.9
50	10	8.140	20	60	0.9

Tabla 9.1 Parámetros utilizados para cada una de las ocho variantes del algoritmo de umbral.

Para estos experimentos se escogió $\varepsilon = 0.5$ (la elección de este valor se discutió en la sección 4.3). Al final de esta sección se mostrarán resultados con valores más pequeños para ε , utilizando la mejor variante del algoritmo, y se comprobará que no hay mejoras importantes en las soluciones, a expensas de un incremento sustancial en el tiempo de cómputo. Se utilizaron dos cifras decimales en los valores de la función objetivo para reducir considerablemente la cantidad de empates que surgían al no utilizar cifra decimal, o al utilizar una sola cifra decimal.

Así como en el capítulo anterior, se escogió una lista reducida de valores para n , $N = \{10, 15, \dots, 50\}$, debido a la enorme cantidad de cálculos a realizar.

Dada la naturaleza aleatoria del algoritmo, se optó por realizar más de una ejecución para después tomar los promedios. Para cada variante del algoritmo, y para cada $n \in N$, se realizaron ocho ejecuciones con los parámetros correspondientes descritos en la tabla 8.1. Los valores recabados de la función objetivo y los tiempos de ejecución (medidos en segundos) se organizaron en tablas, una tabla por cada n . La tabla 9.2 corresponde al caso $n = 25$. El resto de las tablas se encuentran en los apéndices B.5 y B.6.

Observaciones	Valores de la función objetivo para $n = 25$							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	55.35	53.54	53.34	53.98	57.16	55.74	53.76	54.20
2	55.37	55.85	53.51	55.18	54.72	57.13	55.10	52.54
3	55.64	56.03	53.89	53.81	55.23	56.34	54.76	55.64
4	54.67	55.76	53.07	54.19	56.35	55.94	55.61	56.57
5	54.24	54.39	54.77	54.47	56.61	56.97	53.06	55.20
6	54.83	54.51	53.10	54.53	57.31	55.77	53.78	52.81
7	54.50	55.03	53.42	54.14	55.53	55.92	53.80	52.60
8	56.84	55.08	53.86	53.73	56.73	55.86	54.40	54.89
Media:	55.18	55.02	53.62	54.25	56.20	56.21	54.28	54.30

Tabla 9.2 En cada columna se registró el valor de la función objetivo obtenido en cada ejecución del algoritmo. Cada columna corresponde a una variante distinta del algoritmo. En cada observación se resaltó el valor más alto obtenido.

La tabla 9.3 muestra los promedios de los valores de la función objetivo obtenidos en ocho ejecuciones, que se obtuvieron para cada variante del algoritmo y para cada valor de n . Se realizó la prueba de Friedman para analizar si las diferencias entre las variantes del algoritmo son estadísticamente significativas o no. Se utilizó el programa MedCalc versión 13.1.2.0 para realizar el cálculo de las pruebas estadísticas.

Tanto para la tabla con los promedios de la función objetivo como para la tabla de los tiempos se obtuvo un valor de $P < 0.00001$, donde P es la probabilidad de que los datos en la tabla hayan sido generados aleatoriamente. Esto indica claramente que las diferencias entre las variantes sí son significativas en términos estadísticos. El valor más alto para la clasificación media (**C.M.**) lo obtuvo la variante **UA-g-0**, seguido de la variante **UA-g-1**, lo cual las coloca como las variantes del algoritmo con mejor desempeño, en términos del valor de la función objetivo. En la tabla 9.3 el valor de P aparece truncado a cuatro decimales pero no es idéntico a 0. Las clasificaciones medias **C.M.** son el promedio de valores enteros que van del 1 al 8, por lo que solo resulta necesario utilizar una cifra decimal. Para mayores detalles sobre la prueba de Friedman consultese el apéndice A.

	Valor promedio de la función objetivo							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
N=10	94.71	94.11	92.99	91.73	95.19	94.45	92.09	92.54
N=15	73.27	73.69	72.51	72.90	74.28	74.83	74.08	73.56
N=20	63.24	62.41	60.35	61.03	62.94	63.23	62.10	61.60
N=25	55.18	55.02	53.62	54.25	56.20	56.21	54.28	54.30
N=30	50.44	49.79	48.66	48.37	50.79	50.50	50.11	49.37
N=35	45.93	45.76	45.51	45.95	46.98	46.86	46.40	45.83
N=40	42.83	42.48	41.65	40.39	43.76	43.71	42.19	41.90
N=45	40.69	39.96	39.02	38.77	41.48	40.84	40.42	39.28
N=50	38.12	38.02	36.79	36.58	39.52	39.20	37.69	37.24
C. M.	5.8	4.4	1.8	1.8	7.6	7.1	4.3	3.2
P=	0.0000							

Tabla 9.3 Promedios de los valores de la función objetivo en los experimentos para cada n y cada variante del algoritmo.

En la tabla 9.4 se muestra un resumen del análisis que se realizó con los resultados de los experimentos computacionales, aquéllos recabados en las tablas de los apéndices B.5 y B.6. En cada renglón se despliegan tres cantidades por cada variante del algoritmo. A continuación daremos una descripción de su significado.

La cantidad **Dev %** se refiere a la desviación relativa del mejor valor de la función objetivo, para esa variante del algoritmo, con respecto al mejor valor de la función objetivo entre todas las variantes del algoritmo.

La cantidad **Mejor** se refiere al número de ocasiones, de las ocho ejecuciones, en que esa variante del algoritmo obtuvo el mejor valor de la función objetivo, con respecto a los valores de las demás variantes del algoritmo.

La cantidad **Tiempo** es el promedio de los tiempos de cálculo de las ocho ejecuciones.

Por ejemplo, si examinamos la tabla 9.2, el valor más alto en la tabla es 57.31 y le corresponde a la variante **UA-g-0**. Ahora, para la variante **RS-g-0** el valor más alto que obtuvo es 56.84, entonces la desviación relativa de este valor con el más alto se calcula como

$$\text{Dev \%} = \frac{|56.84 - 57.31|}{57.31} = 0.82 \%$$

En la misma tabla podemos observar que la variante **UA - g - 1** obtuvo en cuatro ejecuciones el valor más alto (resaltado en negritas), por lo cual ésta variante tiene un 4 donde corresponde la cantidad para **Mejor**, para $n = 25$.

Al final de la tabla 9.4 se muestra un resumen de las mediciones realizadas, donde **Dev %** es el promedio de las desviaciones relativas por columna; **Mejor** es la suma de éstos valores por columna; y **Tiempo** es el promedio de los tiempos por columna.

		RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
N=10	Dev %	0.47%	0.59%	1.05%	2.01%	0.00%	0.00%	1.81%	1.43%
	Mejor	1	0	0	0	4	3	0	0
	Tiempo	6.31	9.43	6.99	5.87	9.13	8.74	6.36	6.46
N=15	Dev %	2.50%	1.80%	0.85%	1.19%	0.00%	0.39%	0.04%	2.37%
	Mejor	1	1	0	0	1	3	2	0
	Tiempo	15.52	10.36	10.28	11.99	13.20	14.82	12.14	13.28
N=20	Dev %	1.01%	1.39%	3.00%	3.74%	0.50%	0.00%	1.15%	2.99%
	Mejor	2	0	0	0	3	3	0	0
	Tiempo	23.06	21.97	17.78	17.40	21.85	22.73	17.94	17.14
N=25	Dev %	0.82%	2.23%	4.43%	3.72%	0.00%	0.31%	2.97%	1.29%
	Mejor	0	0	0	0	3	4	0	1
	Tiempo	41.23	35.79	30.22	32.96	38.37	41.03	30.72	29.82
N=30	Dev %	0.00%	1.95%	4.25%	5.67%	0.85%	0.09%	2.37%	1.70%
	Mejor	1	0	0	0	2	2	2	1
	Tiempo	58.91	48.35	37.51	47.67	50.27	48.70	36.35	45.67
N=35	Dev %	0.95%	0.59%	2.46%	1.24%	0.00%	0.46%	1.69%	2.37%
	Mejor	1	1	0	0	4	1	1	0
	Tiempo	68.34	74.98	50.14	65.62	61.22	63.57	60.75	66.35
N=40	Dev %	2.10%	1.45%	4.54%	7.98%	0.00%	0.27%	3.21%	4.62%
	Mejor	0	1	0	0	4	3	0	0
	Tiempo	74.85	83.22	52.39	66.36	66.53	70.84	48.34	66.49
N=45	Dev %	2.14%	2.73%	6.03%	5.00%	0.00%	1.80%	2.58%	4.83%
	Mejor	1	0	0	0	6	1	0	0
	Tiempo	94.25	103.86	64.36	88.61	88.16	88.21	68.89	78.48
N=50	Dev %	4.50%	3.11%	5.55%	7.46%	0.00%	0.17%	4.20%	5.49%
	Mejor	0	0	0	0	4	4	0	0
	Tiempo	98.53	132.95	67.95	100.61	110.83	111.93	84.54	96.11
Resumen	Dev %	1.61%	1.76%	3.58%	4.22%	0.15%	0.39%	2.22%	3.01%
	Mejor	8	3	0	0	30	24	5	2
	Tiempo	53.44	57.88	37.51	48.56	51.06	52.29	40.67	46.64

Tabla 9.4 Concentrado de los resultados obtenidos por cada una de las ocho variantes del algoritmo para cada valor de n .

La variante del algoritmo que consistentemente obtuvo los mejores valores de la función objetivo fue **UA-g-0**, seguido de **UA-g-1**. Consecuentemente, los valores más bajos para **Dev %** los obtuvieron dichas variantes del algoritmo. Podemos observar también que las variantes que utilizaron una solución inicial construida, con el algoritmo de partición tetraédrica recursiva, obtuvieron en general mejores resultados que sus contrapartes que utilizaron una solución inicial aleatoria, esto se ve reflejado tanto en **Dev %** como en **Mejor**. Otra observación importante es que las variantes con umbral de aceptación (**UA**) obtuvieron mejores resultados que las variantes correspondientes de recocido simulado (**RS**). También podemos notar que los métodos que iniciaron con solución inicial aleatoria obtuvieron, en promedio, tiempos ligeramente mayores que aquellos que iniciaron

con solución inicial construida por el método de partición tetraédrica recursiva. Por último, cabe señalar que las conclusiones obtenidas a partir de la tabla 9.4 son consistentes con las clasificaciones medias obtenidas con la prueba de Friedman, tabla 9.3.

Podemos concluir, para la resolución de nuestro problema de optimización combinatoria, que:

- El algoritmo de umbral de aceptación tiene un mejor desempeño que el algoritmo de recocido simulado para implementaciones de tiempo finito.
- Iniciar con una solución construida, con el método de partición tetraédrica recursiva, permite llegar a mejores soluciones en tiempos menores que una solución inicial aleatoria.
- La función de decremento g de la estrategia geométrica provee de mejores resultados que la función oscilante amortiguada h .

Hemos comprobado a partir del análisis estadístico y los índices de la tabla anterior que la mejor variante del algoritmo para resolver nuestro problema de optimización es **UA-g-0**. Por último, se utilizó esta variante para calcular soluciones aproximadas para $n = 10, 20, 30, 40$ y 50 . El valor utilizado para el cero virtual es $\varepsilon = 0.1$, para que el algoritmo realizara más iteraciones antes de detenerse, con la expectativa de obtener soluciones ligeramente mejores que para $\varepsilon = 0.5$. Al final de este capítulo se muestran las paletas de colores obtenidas en cada caso.

Los parámetros utilizados para este último grupo de experimentos computacionales son aquellos dados por la tabla 9.1. Ahora bien, para demostrar experimentalmente que si refinamos aún más el valor de ε no se obtendrán mejoras sustanciales en la solución, pero sí se incrementará de manera significativa el tiempo de cálculo, haremos también los experimentos para $\varepsilon = 0.01$.

Antes de mostrar los resultados de este último grupo de experimentos computacionales, veremos con más detalle el ejemplo cuando $n = 30$. Utilizaremos la variante del algoritmo **UA-g-0**.

En primer lugar, la figura 9.1 muestra un resumen de la ejecución del algoritmo donde se obtuvo la solución aproximada. En él podemos apreciar el valor de los parámetros introducidos, los cambios en el tamaño de la δ -caja vecindad, el valor de la función objetivo inicial y la final, así como el tiempo de cálculo y número de iteraciones.

Número de vértices (n):	30
¿Solución inicial determinista (0) o aleatoria (1)?:	0
Valor de λ :	60
Factor de enfriamiento (φ):	0.9
Cero virtual (ε):	0.1
Valor de δ_0 :	20
Valor de t_0 :	13.77
 “Se reduce delta: 10.0”	
 “Se reduce delta: 5.0”	
 “Se reduce delta: 2.5”	
 “Se reduce delta: 1.25”	
 “Se reduce delta: 0.625”	
 “Se reduce delta: 0.3125”	
Valor inicial de la función objetivo:	24.68
Valor final de la función objetivo:	51.73
Temperatura final:	0.097
Iteraciones totales:	21105
Aceptaciones totales:	7440
Porcentaje de aceptación:	35 %
Tiempo de cálculo (s):	78.77

Figura 9.1 Resumen de una ejecución de la variante del algoritmo UA-g-0 para $n = 30$.

En la figura 9.2 se muestra el historial de aceptaciones del ejemplo anterior, que consiste en graficar puntos cuyas coordenadas representan el número de solución aceptada (abscisa) y el valor de la función objetivo de dicha solución (ordenada). Los puntos rojos en la gráfica representan los momentos en que se realizó una disminución en la temperatura. La estrella representa el número de aceptación y el valor de la función objetivo de la mejor aproximación a la solución óptima. En el gráfico podemos notar claramente como el valor de la función objetivo oscila bastante al principio. Conforme avanza el algoritmo, los valores de la función objetivo se vuelven progresivamente mayores y su rango de oscilación es menor puesto que la temperatura ha ido descendiendo paulatinamente. El algoritmo se detiene cuando la temperatura ha alcanzado un valor por debajo de $\varepsilon = 0.1$.

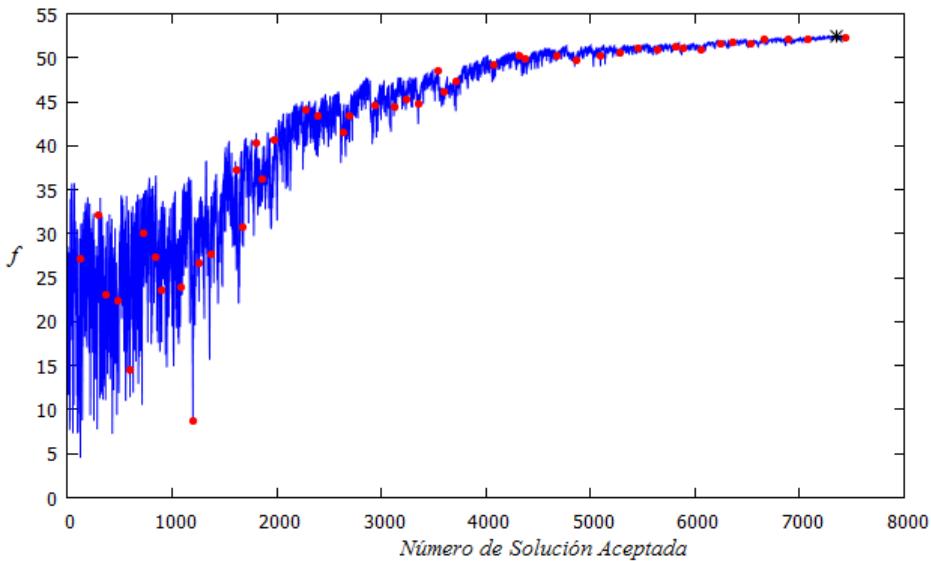


Figura 9.2 Historial de aceptación para $n=30$ con la variante **UA-g-0**.

En la figura 8.3 se muestra gráficamente la solución de nuestro ejemplo para $n = 30$, en el espacio CIE $L^*u^*v^*$. Los vértices en forma de cruz representan las ubicaciones de los puntos de la solución inicial, y junto con las aristas forman la malla de la partición tetraédrica. Los puntos coloreados son propiamente los elementos de la solución aproximada.

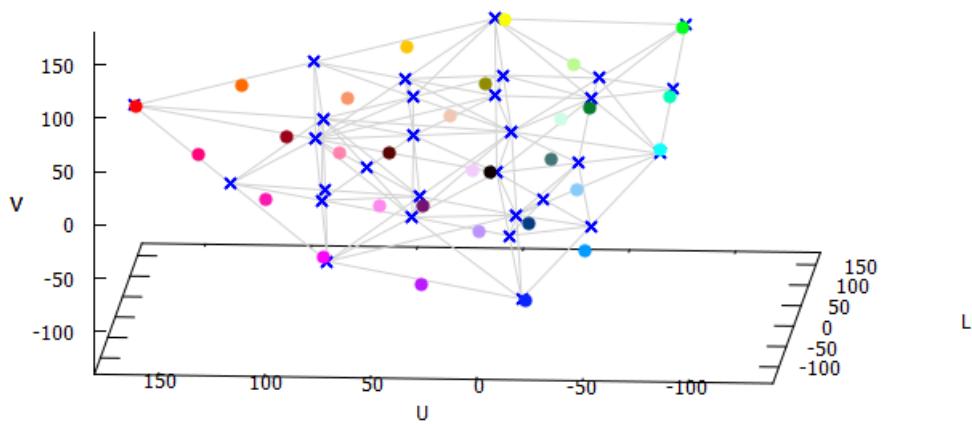


Figura 9.3 Representación de las soluciones, inicial y cuasi-óptima, para $n = 30$ en el espacio CIE $L^*u^*v^*$.

En la figura 9.4 se muestra la solución cuasi-óptima de nuestro ejemplo para $n = 30$, desde otro ángulo visual. En esta figura, si dos puntos en la solución se encuentran a

una distancia equivalente al valor de la función objetivo 51.73, o mayor hasta en un 5%, entonces se dibuja una arista. Véase que la mayoría de los puntos están conectados, lo cual quiere decir que la distribución de los puntos en el espacio $\mathcal{C}_{L^*u^*v^*}$ es buena.

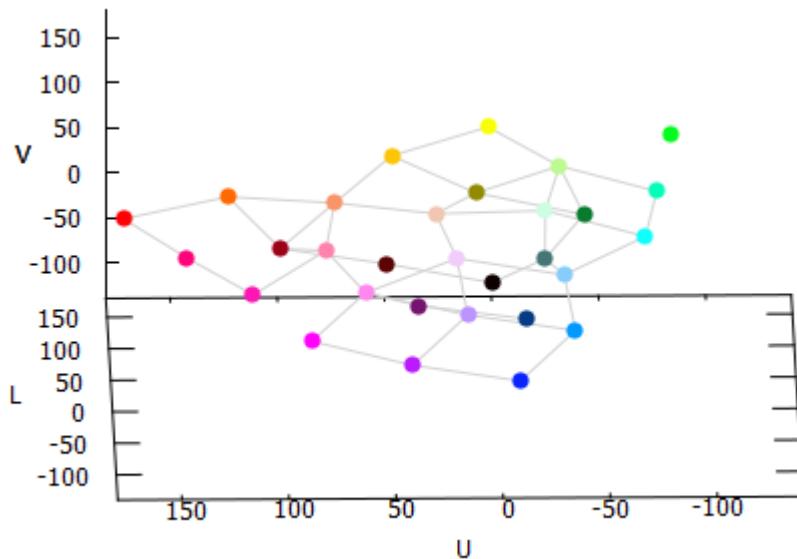


Figura 9.4 Los puntos conectados por una arista se encuentran a una distancia entre sí de 51.73 y hasta un 5% más.

A continuación, mostraremos los resultados experimentales para algunos valores de n , con $\epsilon = 0.1$ y $\epsilon = 0.01$, véase tabla 9.5. Al comparar los resultados se puede apreciar que la variación en el valor de la función objetivo no es tan significativa como la variación en los tiempos de cálculo, al refinar el valor de ϵ . Incluso, para $n = 20, 40$ y 50 , ni siquiera hubo mejoría en el valor de la función objetivo, sino que el valor de la función objetivo fue menor para $\epsilon = 0.01$. Así pues, queda demostrado experimentalmente que seguir reduciendo el valor de ϵ no necesariamente redundó en obtener mejoras sustanciales en las soluciones, pero si hay un incremento importante en los tiempos de cálculo.

	n	10	20	30	40	50
$\epsilon = 0.1$	f	95.89	64.84	51.73	46.24	41.17
	tiempo	15.11	44.34	78.76	132.27	225.60
$\epsilon = 0.01$	f	97.39	64.00	52.51	45.46	40.62
	tiempo	24.19	56.97	102.61	197.21	272.25

Tabla 9.5 Resultados del experimento final con la variante UA-g-0.

Veamos por último la paleta de colores obtenida para $n = 10, 20, 30, 40$ y 50 , con $\varepsilon = 0.1$. Los colores se acomodaron más o menos de acuerdo a su tonalidad para una mejor apreciación. Es importante señalar que las paletas no son únicas, pues debido a que los métodos utilizados tienen una fuerte componente aleatoria, en cada ejecución se obtendrán soluciones ligeramente distintas para un mismo valor de n .

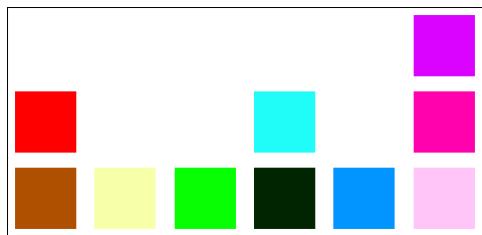


Figura 9.5 Paleta de colores para $n = 10$

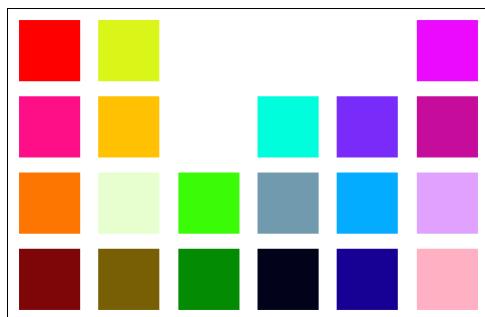


Figura 9.6 Paleta de colores para $n = 20$

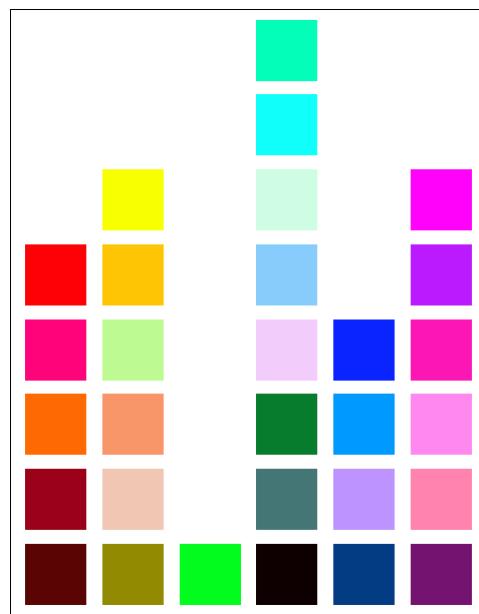


Figura 9.7 Paleta de colores para $n = 30$

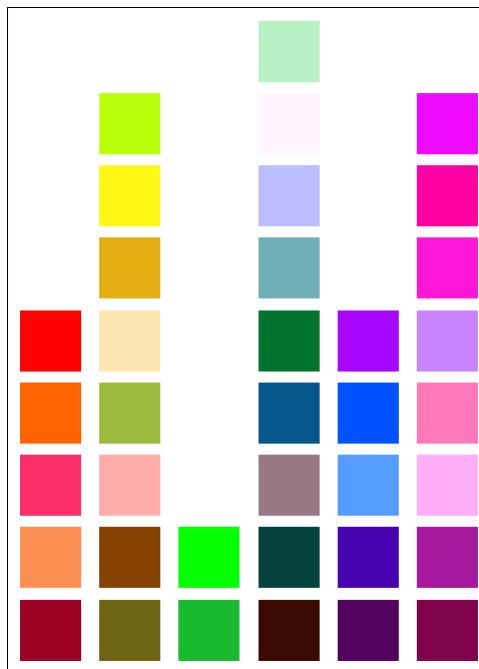


Figura 9.8 Paleta de colores para $n = 40$

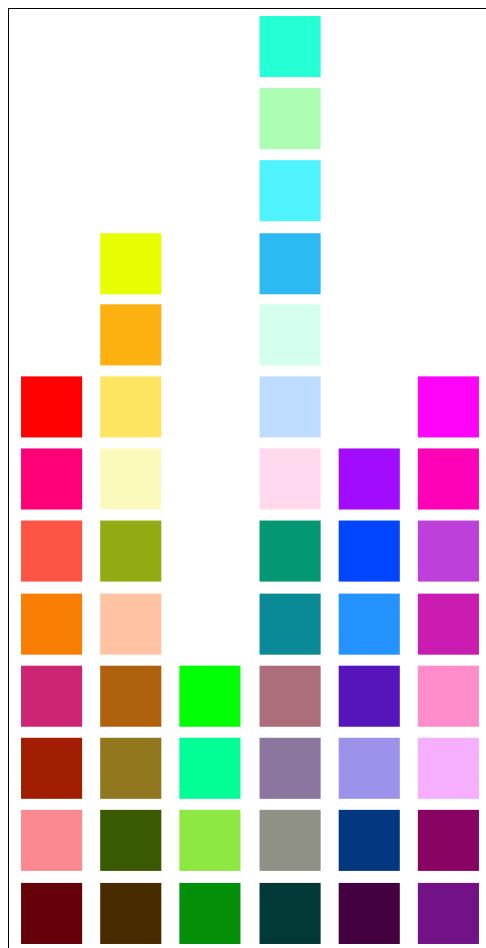


Figura 9.9 Paleta de colores para $n = 50$

Capítulo 10

Conclusiones

A lo largo de este trabajo, hemos visto que el diseño de una paleta de n colores bien diferenciados no es una tarea sencilla. En primer lugar, es necesario contar con un modelo matemático que describa la manera en que el ser humano percibe el color. Así pues, en 1931 la Comisión Internacional de la Iluminación propuso el primer modelo matemático para este fin. En segundo lugar, es necesario establecer una manera para diferenciar los colores de manera objetiva. Debido a que los espacios de color se modelan en tres dimensiones, como subconjuntos de \mathbb{R}^3 , la manera natural para medir la diferencia entre dos colores es utilizando la métrica euclíadiana.

La primer dificultad que surge, al tratar de medir la diferencia entre colores con la métrica euclíadiana, es que el espacio de color CIE RGB de 1931 no es perceptualmente uniforme. Para tratar de corregir esta situación, el espacio CIE RGB fue modificado mediante la aplicación de un conjunto de transformaciones, para obtener un espacio con una percepción visual más uniforme, el espacio CIE $L^*u^*v^*$.

Antes de formular el problema de optimización en el espacio CIE $L^*u^*v^*$, se hizo el planteamiento en el cubo unitario como un problema de optimización convexa. El problema consiste en encontrar un conjunto de n puntos dentro del dominio, tal que la distancia mínima entre las parejas de puntos sea la mayor posible. Así pues, $f(X) = \min_{x,y \in X} d(x,y)$, con X un conjunto de n puntos en el cubo unitario, y por lo tanto el modelo de optimización es

$$\max_{X \subset \mathcal{C}} f(X),$$

donde \mathcal{C} es el cubo unitario.

En el caso del cubo, la demostración que se dió para encontrar la solución analítica, para $n = 3$, nos sugiere que resolver este problema para cualquier n puede ser una tarea bastante compleja. En el caso discreto, no se conocen métodos exactos para resolver el problema de n -dispersión (o problema de diversificación Máx-Min) de manera eficiente, así que los métodos heurísticos surgen como una alternativa eficaz para dar soluciones

aproximadas a la solución óptima de este problema.

Debido a que los colores se desplegaron en un monitor de computadora, el espacio de color se limitó al gamut que puede ser generado por una pantalla a base de pixeles compuestos por tres LED, rojo, verde y azul. La computadora es capaz de reproducir 256^3 colores, que se representan como puntos con coordenadas enteras de un cubo $\mathcal{C}_{RGB} = \{(x, y, z) \mid 0 \leq x, y, z \leq 255\}$. Aplicando las transformaciones necesarias a \mathcal{C}_{RGB} , se obtiene su imagen en el espacio CIE L^{*}u^{*}v^{*}, a la cual denominamos $\mathcal{C}_{L^*u^*v^*}$.

La metaheurística utilizada en este trabajo es el algoritmo de recocido simulado, que surgió por analogía del proceso de recocido en la mecánica estadística. Debido a que este algoritmo goza de una fuerte componente estocástica, éste se puede modelar utilizando la teoría de las cadenas de Markov. Esto permite su análisis asintótico para llegar a resultados importantes de convergencia. El resultado principal nos dice que, si se permite un número infinito de iteraciones y que la temperatura descienda lo suficientemente despacio, el algoritmo es capaz de encontrar una solución óptima global con probabilidad uno. Claro que en la práctica estamos limitados a un número finito de iteraciones, por lo que solo podemos encontrar soluciones aproximadas.

En la implementación del algoritmo se utilizaron dos etapas, la primera constructiva y la segunda de búsqueda local. En la primer etapa se utilizó una técnica que denominamos partición tetraédrica recursiva, para la colocación inicial de n puntos en el espacio de color. Una vez que se tiene la solución inicial, se procede a mejorar las ubicaciones de los puntos mediante la etapa de búsqueda local. El algoritmo se detiene al satisfacer alguna condición de paro.

Un aspecto importante para el desempeño del algoritmo es la selección adecuada de los parámetros que lo gobiernan, parte de lo que se conoce como la estrategia de enfriamiento: δ_0 , tamaño de la vecindad; θ , valor máximo de soluciones vecinas a explorar; t_0 , temperatura inicial; λ , tamaño del lote, que se relaciona con las longitudes de las cadenas homogéneas de Markov; φ , factor de enfriamiento, responsable de disminuir el valor de la temperatura; y ϵ , el cero virtual, para definir un criterio de paro. La determinación de estos parámetros fue experimental. Básicamente, se procedió a realizar experimentos computacionales con distintos valores de los parámetros y se recolectaron los datos en matrices, a las que después se les aplicó un análisis de varianza mediante pruebas estadísticas no paramétricas (Friedman y Wilcoxon) para determinar las combinaciones de parámetros adecuadas. El valor de la temperatura inicial t_0 se calculó mediante el empleo de un algoritmo de búsqueda binaria.

Una vez determinados los parámetros, se procedió a la realización de los experimentos computacionales. Vale la pena mencionar que el algoritmo se ejecutó en $\mathcal{C}_{L^*u^*v^*}$ como si se tratase de un espacio continuo, y al obtener la solución aproximada, las coordenadas de los puntos correspondientes en \mathcal{C}_{RGB} se redondearon a valores enteros.

En la parte experimental se realizaron pruebas con ocho variantes del algoritmo, que resultan de considerar todas las posibles combinaciones de tres elementos: tipo de algoritmo, recocido simulado o umbral de aceptación; tipo de solución inicial, aleatoria

o construída; tipo de función de decremento, función de decrecimiento de la estrategia geométrica o una función oscilante amortiguada propuesta en este trabajo.

Debido a la gran cantidad de cálculos que se requerían, se escogió una lista reducida de valores para n , $N = \{10, 15, \dots, 50\}$. La razón por la que no se escogieron valores más allá de 50 resulta de las observaciones de Carter&Carter, respecto a la distancia mínima para que dos colores sean bien distinguibles en el espacio $\mathcal{C}_{L^*u^*v^*}$. Esta distancia es de 40 unidades, límite que se alcanza para $n = 50$.

Debido a la naturaleza aleatoria del algoritmo, se consideró necesario realizar varias ejecuciones del algoritmo para cada n y cada variante del algoritmo. Los datos se recolectaron en matrices y se realizó un análisis de varianza mediante el uso de pruebas estadísticas no paramétricas, la prueba de Friedman y la de Wilcoxon, para determinar si las diferencias entre las variantes del algoritmo eran estadísticamente significativas. Se concluyó que la variante **UA-g-0** es la que tiene el mejor desempeño, entregando soluciones de buena calidad con tiempos de cómputo razonables.

En general, las variantes que utilizaron una solución inicial construida tuvieron mejor desempeño que aquéllas que utilizaron una solución inicial aleatoria. La función de decremento que probó dar los mejores resultados fue la de la estrategia geométrica. Asimismo, los resultados arrojados por las variantes que emplearon el algoritmo de umbral de aceptación fueron significativamente mejores, en términos estadísticos, a aquéllos arrojados por las variantes que emplearon el algoritmo de recocido simulado.

Las soluciones que se obtienen con la implementación que se desarrolló en este trabajo no son únicas. Esto se debe a la variación que genera la etapa de búsqueda local. El programa que se desarrolló permite al usuario escoger valores distintos para los parámetros encargados de gobernar al algoritmo, pues los parámetros encontrados no son necesariamente óptimos, sino que se determinaron a manera de que el algoritmo se comporte eficientemente y entregue soluciones aproximadas de buena calidad. Asimismo, el programa no está limitado a calcular soluciones para los valores de n que se escogieron para los experimentos, sino que es capaz de encontrar soluciones para cualquier n , no obstante, utilizando los parámetros del valor más cercano en $N = \{10, 15, \dots, 50\}$, que se determinaron en este trabajo.

Parte III

Apéndice

Apéndice A

Pruebas estadísticas no paramétricas

En este apéndice se hará una revisión breve de las pruebas estadísticas no paramétricas que se utilizaron para sacar conclusiones en la parte experimental de este trabajo. Comenzaremos retomando dos conceptos básicos en la probabilidad y estadística.

Definición A.1. Una *distribución de probabilidad* f es una función (discreta o continua) que asigna una probabilidad a cada posible conjunto de observaciones (o mediciones) de los posibles resultados de un experimento aleatorio.

Definición A.2. Una *variable aleatoria* X es aquélla que toma valores al azar dentro de un conjunto de posibles resultados, los cuales tienen una probabilidad asociada, que se denomina *espacio muestral*.

Existen básicamente dos tipos de distribuciones de probabilidad: discretas y continuas. En el primer caso la probabilidad es un valor bien definido en el intervalo $[0, 1]$ para cada uno de los posibles resultados del espacio muestral discreto, siendo la suma de todas las probabilidades igual a 1. En el segundo caso, la probabilidad se calcula para un intervalo de mediciones como el área bajo la curva de la distribución de probabilidad continua asociada. En tales distribuciones, el área bajo la curva de la gráfica de f en $(-\infty, \infty)$ es igual a 1. La figura A.1 es una ilustración de una de las distribuciones de probabilidad continuas de mayor uso, la distribución normal

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

donde μ es la media y σ la desviación estándar. Para la distribución normal estándar se fijan los valores $\mu = 0$ y $\sigma = 1$. En la figura A.1 se aprecia una zona sombreada que es el área bajo la curva en el intervalo $[1, 1.5]$. El valor 9.2% quiere decir que el área sombreada es de 0.092, lo que es igual a decir que la probabilidad de que la variable

aleatoria X en cuestión tome valores en el rango $[1, 1.5]$ es de 9.2%, esto se expresa como $\mathbf{P}(1 < X < 1.5) = 0.092$.

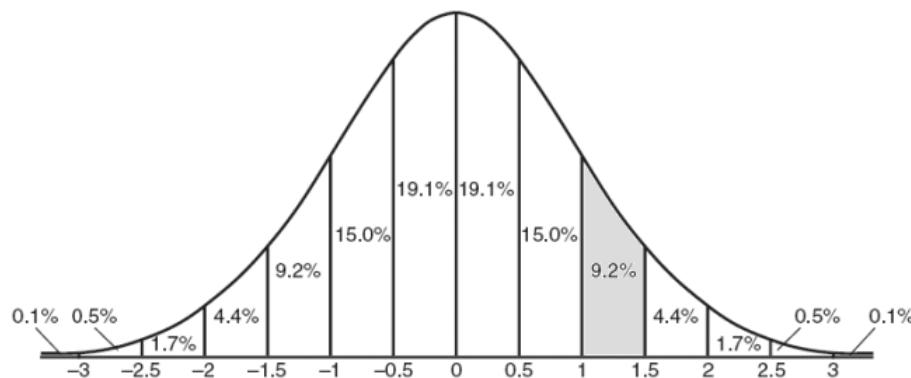


Figura A.1 La gráfica de la distribución normal estándar. La probabilidad de que la variable aleatoria X tome valores en el rango $[1, 1.5]$ es de 9.2%

Las pruebas que a continuación revisaremos, la prueba de Friedman y la prueba de Wilcoxon, se utilizan en lo que se conoce como *análisis de varianza*, la cual es una herramienta estadística utilizada para encontrar los factores responsables de causar variación en alguna característica específica de una población dada. La diferencia de estas dos pruebas con respecto a los métodos tradicionales del análisis de varianza es que éstas no requieren la suposición de alguna distribución de probabilidad en particular para las observaciones, razón por la cual reciben el nombre de pruebas *no paramétricas*. Esto implica que tales pruebas tienen la posibilidad de ser aplicadas en un espectro más amplio de situaciones, así como el hecho de que tienen una mayor economía y simplicidad en los cálculos. La siguiente información está basada en [Fri37], [Wik14b], [Wil45] y [Low13], donde pueden encontrarse mayores detalles al respecto.

A.1. Prueba de Friedman

Es una prueba estadística para comparar tres o más muestras relacionadas donde la distribución de probabilidad de las observaciones se desconoce. El objetivo primario que se quiere alcanzar es demostrar que las medias de las muestras son significativamente distintas. El procedimiento involucra clasificar a las observaciones en una escala de 1 a k , donde k es el número de *tratamientos*, asignando la clasificación más alta (k) a la observación más favorable, en términos de la característica estudiada, y la clasificación más baja (1) a la observación menos favorable. En caso de existir empate entre dos o más observaciones se les asigna el promedio de las clasificaciones que abarcan, es decir, de aquéllas que se les hubieran asignado sin empate.

La información de las muestras viene dada en forma de una matriz de n renglones (bloques) y k columnas (tratamientos), que denotaremos como $\{x_{ij}\}_{n \times k}$, donde cada entrada de la matriz corresponde a una observación. Ahora bien, a partir de esta información se elabora la matriz de las clasificaciones $\{c_{ij}\}_{n \times k}$, donde c_{ij} es la clasificación de la observación x_{ij} con respecto a las demás observaciones del mismo bloque i . El siguiente paso es la realización de los cálculos descritos a continuación

- La media aritmética de las clasificaciones en cada tratamiento, $\bar{c}_j = \frac{1}{n} \sum c_{ij}$
- La media aritmética de todas las clasificaciones, $\bar{c} = \frac{1}{nk} \sum_{i=1}^n \sum_{j=1}^k c_{ij}$
- $SS_t = n \sum_{j=1}^k (\bar{c}_j - \bar{c})^2$
- $SS_e = \frac{1}{n(k-1)} \sum_{i=1}^n \sum_{j=1}^k (c_{ij} - \bar{c})^2$
- La prueba estadística está dada por $Q = \frac{SS_t}{SS_e}$

La prueba estadística se considera un valor numérico que resume la información de un conjunto de datos para realizar una prueba de hipótesis, que consiste básicamente en tratar de refutar la *hipótesis nula* H_0 y aceptar la *hipótesis alternativa* H_1 , basándose en la probabilidad de que la prueba estadística se encuentre en cierto rango de valores.

- H_0 : no hay diferencias estadísticamente significativas entre las muestras,
- H_1 : las muestras presentan diferencias estadísticamente significativas.

En el caso de la prueba de Friedman, la distribución de probabilidad de Q puede ser aproximada por la distribución de probabilidad chi cuadrada con $k - 1$ grados de libertad. Ahora bien, si Q resulta ser un valor poco probable, de acuerdo con su distribución de probabilidad, entonces podemos pensar que este valor no se obtuvo de manera aleatoria, sino que existe un factor de varianza en las muestras que está originando una diferencia significativa, y como resultado, produciendo un valor de Q poco probable. Al valor $\mathbf{P}(\chi_{k-1}^2 \geq Q)$ se le denomina *p-valor*. Si este valor es menor o igual al *nivel de confianza*, tradicionalmente 1 % ó 5 %, entonces se rechaza la hipótesis nula H_0 y se acepta la hipótesis alternativa H_1 .

Un ejemplo práctico para entender esta prueba estadística es el siguiente. Supongamos que se desea comparar la calidad de cuatro vinos distintos y se han seleccionado 9 jueces para evaluar a los vinos. Los resultados de las puntuaciones otorgadas por los jueces se encuentran en la tabla A1.1.

Juez	Vino			
	V1	V2	V3	V4
J1	2.8	3.6	1.4	2
J2	5.9	1.7	0.9	2.2
J3	3.3	5.1	1.1	0.9
J4	4.4	2.2	3.2	1.1
J5	1.7	2.1	0.8	0.5
J6	3.8	4.1	1.5	1.5
J7	6.6	4.7	2.8	1.4
J8	3.1	2.7	1.4	3.5
J9	0	1.3	0.5	1.2

Tabla A1.1 Puntuaciones otorgadas por los jueces.

El primer paso en la aplicación de la prueba de Friedman es determinar la matriz de las clasificaciones. En este caso las clasificaciones varían del 1 al 4. Nótese que J6 otorgó la misma puntuación a V3 y V4 en la tabla anterior, así que se les asigna la clasificación que corresponde al promedio de las clasificaciones 1 y 2, véase la tabla A1.2.

Juez	Vino			
	V1	V2	V3	V4
J1	3	4	1	2
J2	4	2	1	3
J3	3	4	2	1
J4	4	2	3	1
J5	3	4	2	1
J6	3	4	1.5	1.5
J7	4	3	2	1
J8	3	2	1	4
J9	1	4	2	3
C.M.	3.11	3.22	1.72	1.94

Tabla A1.2 Tabla de las clasificaciones. Al final de la tabla está la clasificación media (C.M.) por cada columna.

Una vez hechos los cálculos se obtiene $Q = 9.8764$. Consultando la distribución chi cuadrada con 3 grados de libertad se obtiene $\mathbf{P}(\chi_3^2 \geq 9.8764) = 0.0196$, por lo tanto se procede a rechazar la hipótesis nula (habiendo predefinido el nivel de confianza en 0.05), lo cual nos lleva a concluir que las diferencias en las muestras son estadísticamente significativas. Examinando los promedios de las clasificaciones encontramos que V1 y V2 obtuvieron los promedios más altos, por lo cual el mejor vino será uno de estos dos, pero todavía hace falta verificar si las diferencias entre las clasificaciones obtenidas por

V1 y V2 son estadísticamente significativas o no, ya que resultaron ser bastante similares. Para ello, es necesario a utilizar una prueba estadística para dos muestras, la prueba de Wilcoxon que describiremos a continuación.

A.2. Prueba de Wilcoxon

La prueba de Wilcoxon se utiliza para comparar dos muestras relacionadas donde la distribución de probabilidad de las observaciones se desconoce. La hipótesis nula y la hipótesis alternativa son

- H_0 : la media de las diferencias entre parejas de datos es cero,
- H_1 : la media de las diferencias entre parejas de datos es distinta de cero.

El objetivo sigue siendo el mismo: refutar la hipótesis nula para así aceptar la hipótesis alternativa. Las hipótesis de la prueba de Wilcoxon vienen a ser equivalentes, en cierto sentido, a las hipótesis de la prueba de Friedman, ya que en ambos casos se desea comprobar que las diferencias entre las muestras no se deben meramente al azar, sólo que en el caso de Wilcoxon tratamos únicamente con dos muestras.

El procedimiento consiste en comparar las dos muestras haciendo parejas con observaciones de cada una de ellas, sea entonces m el número de parejas. Denotamos a las parejas como $(x_{1,i}, x_{2,i})$, donde $x_{1,i}$ es una observación de la primera muestra y $x_{2,i}$ es una observación de la segunda muestra. A continuación hacemos los siguientes cálculos

- Obtener $|x_{1,i} - x_{2,i}|$ y $sgn(x_{1,i} - x_{2,i})$, donde $sgn(x) := \begin{cases} -1 & \text{si } x < 0, \\ 0 & \text{si } x = 0, \\ 1 & \text{si } x > 0. \end{cases}$
- Excluir aquellas parejas tales que $|x_{1,i} - x_{2,i}| = 0$. Sea m_r la cantidad reducida de parejas.
- Se ordenan las parejas ascendentemente con respecto a las diferencias absolutas $|x_{1i} - x_{2i}|$.
- Se clasifican las parejas otorgándole 1 a aquélla con la diferencia absoluta más pequeña. En caso de empate, se asigna el promedio de las clasificaciones abarcadas, como en la prueba de Friedman. Denotamos las clasificaciones como c_i .
- La prueba estadística está dada por $W = |\sum_{i=1}^{m_r} [sgn(x_{1,i} - x_{2,i})c_i]|$, el valor absoluto de la suma de las clasificaciones con signo.

Para valores $m_r < 10$, la distribución de probabilidad de W se puede calcular mediante la simple enumeración de todas las posibles combinaciones de signos en las m_r clasificaciones, que son 2^{m_r} en total, o bien se consulta en tablas de referencia para la prueba de Wilcoxon.

Si la probabilidad de obtener un valor mayor o igual a W es menor que el nivel de confianza, entonces se rechaza H_0 y se concluye que las diferencias son estadísticamente significativas en las muestras. Por ejemplo, si $m_r = 3$ existe un total de ocho combinaciones posibles para asignar signos a las clasificaciones 1,2 y 3, y cada una de estas combinaciones posee la misma probabilidad $\frac{1}{8}$. Véase la tabla A2.1.

1	2	3	W
+	+	+	+6
-	+	+	+4
+	-	+	+2
+	+	-	0
-	-	+	0
-	+	-	-2
+	-	-	-4
-	-	-	-6

Tabla A2.1 Las posibles combinaciones de signos para las clasificaciones 1,2 y 3.

La figura A2.1 muestra las gráficas de las distribuciones de probabilidad de W para valores pequeños de m_r .

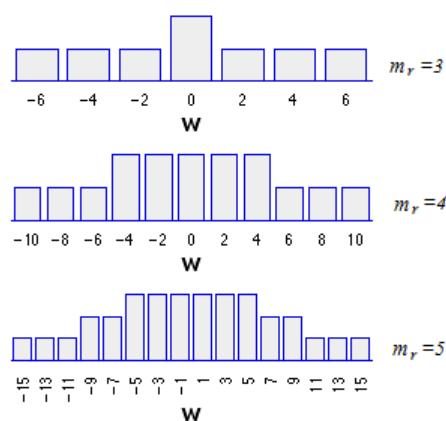


Figura A2.1 Distribuciones de probabilidad de W para valores de $m_r = 3, 4$ y 5 , [Low13].

Para valores cada vez más grandes de m_r la distribución de probabilidad de W converge a la distribución normal, como lo sugiere la figura anterior, así que para $m_r > 10$

$$z = \frac{(W - \mu_W) \pm 0.5}{\sigma_W},$$

donde $\sigma_W = \sqrt{\frac{m_r(m_r+1)(2m_r+1)}{6}}$ es la desviación estándar y μ_W es la media, que para W suponemos que es cero. El valor ± 0.5 es una corrección que toma el signo “-” cuando $W > \mu_W$ y el signo “+” cuando $W < \mu_W$, en este caso $\mu_W = 0$ así que la fórmula queda simplificada como

$$z = \frac{W - 0.5}{\sigma_W}.$$

A z se le denomina *z-valor* y es el valor que le corresponde a W en una distribución normal estándar mediante el cambio de variable descrito. Lo único que restaría por hacer es comprobar en tablas de la distribución normal estándar cuál es la probabilidad de obtener un valor mayor o igual al *z-valor*. Si la probabilidad es menor que el nivel de confianza entonces se rechaza H_0 y se concluye que las diferencias entre las muestras son significativas.

Si aplicamos la prueba de Wilcoxon al ejemplo anterior para comparar V1 con V2 obtenemos que $W = 8$, véase la tabla A2.3, y al consultar una tabla de valores críticos para W se tiene que si $m_r = 9$ entonces $\mathbf{P}(W \geq 35) = 0.05$, lo cual implica que $\mathbf{P}(W \geq 8) > 0.05$ por lo cual no podemos rechazar H_0 , es decir, tenemos que concluir que las diferencias entre ambas muestras no son significativas, o bien, que ambos vinos son de la misma calidad.

Juez	Vino				
	V1	V2	abs	sgn	cal
J1	2.8	3.6	0.8	-1	4
J2	5.9	1.7	4.2	1	9
J3	3.3	5.1	1.8	-1	6
J4	4.4	2.2	2.2	1	8
J5	1.7	2.1	0.4	-1	2.5
J6	3.8	4.1	0.3	-1	1
J7	6.6	4.7	1.9	1	7
J8	3.1	2.7	0.4	1	2.5
J9	0	1.3	1.3	-1	5

Figura A2.3 Comparación de V1 con V2 mediante la prueba de Wilcoxon.

Apéndice B

Tablas

B.1. Matrices δ_0 contra θ para valores de la función objetivo

En las siguientes tablas se muestran los resultados experimentales para determinar un valor apropiado de θ para cada valor de $n = 10, 15, \dots, 50$. Con la información recabada de los experimentos se construyó una matriz $\{a(n)_{ij}\}$ (para cada n) con los promedios de los valores de la función objetivo, calculados mediante la ejecución del algoritmo de mejora sucesiva para nuestro problema de optimización. La entrada $a(n)_{ij}$ es el promedio de los valores de la función objetivo en cinco ejecuciones, para un valor de n dado, con los parámetros: δ_0 correspondiente al valor asignado al renglón i , y θ correspondiente al valor asignado a la columna j . Por ejemplo, la entrada $a(10)_{11} = 85.76$ corresponde al promedio de cinco ejecuciones del algoritmo de mejora sucesiva para $n = 10$, con $\delta_0=5$ y $\theta=10$.

Como criterio de paro en cada ejecución del algoritmo se optó por determinar un número fijo de iteraciones, en este caso 500 iteraciones. Como el objetivo en este experimento era el observar el comportamiento del algoritmo en etapas iniciales, con distintas combinaciones de valores para δ_0 y θ , no era necesario dar un número muy grande de iteraciones.

Una vez recabados los datos, se utilizó una prueba estadística no paramétrica para determinar si existen o no diferencias significativas entre columnas de datos, para entonces escoger el valor de θ que en promedio ofreciera la mejor solución. En todos los casos la prueba de Friedman dió una probabilidad (**P**) menor que un nivel de confianza del 2%, por lo cual se determinó que sí existen diferencias estadísticamente significativas con respecto a la variación de θ .

Se optó por escoger para cada n el valor de θ cuya columna correspondiente obtuviera el valor promedio más alto (**Media**), que en la mayoría de los casos coincidió también con la clasificación media más alta (**C.M.**). En algunos casos, cuando dos columnas

tenían promedios muy similares, se utilizó la prueba de Wilcoxon para determinar si existían o no diferencias significativas entre ellas y poder entonces hacer una elección conveniente. Se resalta en gris la(s) columna(s) con el mejor resultado.

Las siguientes nueve tablas corresponden, cada una, a los valores de $n = 10, 15, 20, 25, 30, 35, 40, 45$ y 50 .

En la tabla correspondiente a $n = 30$, las medias para $\theta = 10$ y $\theta = 20$ son muy similares, así que se aplicó la prueba de Wilcoxon entre las columnas correspondientes y se obtuvo una probabilidad de 0.5566, por lo que se descartó que las diferencias entre los datos fueran estadísticamente significativas. Escogemos entonces el valor $\theta = 20$ que está en acorde con la tendencia de las demás tablas.

En la tabla correspondiente a $n = 50$, la media para $\theta = 10$ y su clasificación media son las mejores, sin embargo, debido a que la tendencia para los demás valores de n es que $\theta = 20$, se aplicó la prueba de Wilcoxon entre las columnas de $\theta = 10$ y $\theta = 20$ y se obtuvo una probabilidad de 0.16, por lo que se descartó que las diferencias entre los datos fueran estadísticamente significativas. Escogemos entonces el valor $\theta = 20$ para uniformizar este parámetro en el algoritmo de optimización para los distintos valores de n empleados.

Valores δ_0	Valores θ									
	10	20	30	40	50	60	70	80	90	100
5	85.76	86.34	85.14	86.38	86.71	87.11	89.09	88.17	86.17	86.23
10	87.70	86.30	86.81	86.86	88.16	87.44	84.83	85.84	87.72	85.63
15	88.60	88.29	87.54	86.24	87.28	86.50	88.11	84.39	85.77	85.29
20	88.32	89.94	89.55	87.03	88.68	86.49	89.14	90.08	86.33	88.97
25	88.28	89.29	89.10	90.18	89.83	88.06	89.65	87.99	88.29	86.96
30	87.03	88.25	88.69	89.17	89.49	87.03	87.32	87.55	86.25	87.09
35	88.28	86.75	87.78	87.61	87.32	87.76	88.17	87.14	87.73	87.12
40	86.14	89.77	90.85	88.21	87.87	84.04	87.11	86.75	86.26	88.18
45	87.12	89.42	89.65	89.05	87.91	86.39	85.33	86.04	84.31	86.61
50	88.13	91.36	87.92	89.16	86.30	88.75	87.27	87.77	84.76	84.39
Media	87.54	88.57	88.30	87.99	87.96	86.96	87.60	87.17	86.36	86.65
C.M.	5.5	7.0	6.9	6.8	6.7	4.8	5.9	4.6	3.4	3.4

$n=15$	Valores θ									
	10	20	30	40	50	60	70	80	90	100
5	63.43	66.67	67.71	66.96	68.54	67.66	66.37	67.53	67.12	66.46
10	65.94	68.14	68.98	68.66	67.88	67.39	68.61	66.72	68.28	67.20
15	67.15	69.25	66.75	68.62	68.65	68.20	67.75	67.10	68.54	67.23
20	68.21	68.26	68.92	67.81	66.91	67.24	67.95	67.26	66.48	67.64
25	68.16	67.86	69.68	68.10	67.75	68.10	68.03	66.59	65.02	66.30
30	67.30	70.12	68.37	67.61	66.95	67.55	67.45	67.07	66.75	64.24
35	68.77	68.84	67.97	67.77	66.84	66.94	66.61	66.52	65.08	64.52
40	66.27	69.42	68.23	67.79	67.60	67.35	66.99	64.88	65.77	65.47
45	67.44	67.77	67.48	67.89	67.13	65.69	64.99	64.16	65.56	63.78
50	68.86	70.12	68.08	68.55	67.28	65.96	63.49	65.84	66.21	63.51
Media	67.15	68.65	68.22	67.98	67.55	67.21	66.82	66.37	66.48	65.64
C.M.	5.6	8.3	8.1	7.7	5.7	5.6	4.7	3.1	3.8	2.4

$n=20$	Valores θ									
	10	20	30	40	50	60	70	80	90	100
5	49.36	51.20	53.11	52.20	53.37	52.57	53.24	52.55	53.11	53.15
10	49.08	53.82	55.08	55.11	54.63	54.57	55.15	55.02	54.17	53.68
15	53.28	56.06	53.80	54.54	53.98	53.10	54.83	53.71	53.71	54.63
20	54.79	55.72	55.65	54.05	54.29	52.23	52.86	55.29	55.59	53.53
25	54.56	54.33	54.52	54.17	52.95	53.10	53.19	53.50	53.24	52.76
30	52.86	53.03	53.76	53.71	53.69	52.60	52.60	52.85	54.39	54.07
35	53.39	54.29	53.11	53.43	52.60	53.29	52.11	51.30	51.29	52.47
40	53.08	55.73	53.41	53.23	52.58	51.32	53.38	51.48	51.57	50.86
45	53.17	55.17	54.70	53.00	52.83	51.10	52.93	52.37	51.40	50.66
50	53.01	56.07	52.69	53.22	52.91	51.43	52.35	52.23	51.37	50.47
Media	52.66	54.54	53.98	53.67	53.38	52.53	53.26	53.03	52.98	52.63
C.M.	5.4	7.8	7.6	6.9	5.7	3.0	5.8	4.3	4.6	3.8

$n=25$	Valores θ									
	10	20	30	40	50	60	70	80	90	100
5	44.36	46.15	45.80	46.69	46.55	46.91	45.96	45.62	47.39	46.35
10	45.58	47.53	47.55	47.53	46.66	46.07	47.16	46.11	47.70	47.73
15	47.79	48.59	48.60	47.07	47.75	47.82	47.96	47.69	47.80	47.29
20	48.78	48.95	48.24	48.33	47.75	47.94	47.54	49.12	46.52	46.89
25	50.52	49.87	48.05	48.50	47.92	47.82	48.68	47.18	46.67	48.16
30	48.34	49.98	48.64	47.08	46.92	48.02	46.57	47.04	46.47	47.43
35	48.51	49.54	48.07	47.40	46.62	47.79	47.87	45.07	45.82	45.54
40	47.70	48.81	47.92	48.40	47.12	46.71	47.84	46.33	46.23	45.69
45	47.17	49.14	47.52	46.55	47.03	46.47	46.99	45.03	46.16	46.41
50	48.20	48.83	47.21	45.74	46.79	46.96	45.77	45.99	44.62	44.94
Media	47.69	48.74	47.76	47.33	47.11	47.25	47.23	46.52	46.54	46.64
C.M.	6.5	8.8	7.4	5.7	4.8	5.4	5.4	3.4	3.6	4.0

Valores δ_0	$n=30$	Valores θ									
		10	20	30	40	50	60	70	80	90	100
	5	36.93	40.10	38.88	40.22	39.54	40.95	41.02	40.16	41.17	41.45
	10	41.21	40.91	42.71	41.62	42.67	42.41	41.93	42.05	42.81	41.98
	15	43.05	42.27	42.93	43.28	42.68	43.29	42.80	43.40	42.30	42.88
	20	43.58	44.20	43.45	43.05	42.98	42.53	43.71	42.74	43.56	42.82
	25	44.58	43.56	43.41	43.31	43.03	43.59	42.14	43.20	42.23	41.68
	30	44.73	43.75	43.09	42.59	42.29	42.31	42.99	42.22	41.71	41.17
	35	43.23	44.35	43.06	42.75	41.81	41.70	42.86	40.87	40.87	41.88
	40	45.26	44.00	43.14	43.33	41.09	41.86	40.64	39.79	41.48	41.03
	45	44.81	43.73	43.61	42.57	40.97	42.05	40.92	40.54	39.97	40.63
	50	44.23	43.74	42.31	41.69	41.16	41.32	40.16	39.43	40.60	40.17
	Media	43.16	43.06	42.66	42.44	41.82	42.20	41.92	41.44	41.67	41.57
	C.M.	7.7	7.0	6.9	6.2	4.4	5.9	4.9	3.7	4.5	3.9

Valores δ_0	$n=35$	Valores θ									
		10	20	30	40	50	60	70	80	90	100
	5	35.68	38.02	38.26	38.11	38.81	38.21	36.80	38.75	38.23	37.83
	10	38.48	39.91	39.17	39.08	38.74	38.87	38.23	39.23	37.73	38.43
	15	39.75	39.64	39.39	39.33	38.20	38.33	38.93	38.62	39.10	38.86
	20	39.81	39.93	40.04	38.26	38.45	38.57	39.07	38.75	38.73	38.84
	25	39.17	40.08	39.27	38.81	38.23	38.83	38.72	38.28	37.73	37.97
	30	38.50	40.23	39.11	39.34	38.35	38.30	37.42	37.83	38.02	37.65
	35	40.26	40.23	38.13	39.24	38.00	38.80	36.77	37.12	38.02	37.34
	40	39.27	39.52	37.85	37.31	37.10	37.62	37.20	36.91	38.06	36.59
	45	40.58	39.27	39.18	38.99	37.63	37.44	36.93	37.34	36.78	37.35
	50	39.94	39.83	39.10	37.99	38.31	37.70	37.21	37.10	37.07	37.01
	Media	39.14	39.67	38.95	38.64	38.18	38.27	37.73	37.99	37.95	37.79
	C.M.	7.7	8.9	8.0	6.1	4.7	5.2	3.3	4.3	3.9	2.9

Valores δ_0	$n=40$	Valores θ									
		10	20	30	40	50	60	70	80	90	100
	5	35.57	36.58	36.85	36.50	36.50	36.67	36.63	36.31	36.00	36.29
	10	36.46	37.19	36.86	36.52	36.71	36.69	36.48	36.27	35.92	36.30
	15	37.04	37.10	36.37	37.06	36.50	36.37	35.92	35.89	36.25	35.69
	20	37.15	37.26	36.50	36.19	36.21	36.06	36.15	35.94	36.08	35.98
	25	36.88	37.28	36.35	36.26	35.50	35.79	36.04	35.85	36.30	35.89
	30	36.98	36.98	36.20	36.54	35.97	36.18	35.82	35.76	35.76	35.72
	35	36.45	37.33	36.10	36.76	35.84	36.00	35.46	35.65	35.71	35.69
	40	37.21	37.28	36.33	36.24	35.78	36.09	35.73	35.31	35.64	35.63
	45	37.15	36.57	36.50	35.87	35.73	35.79	35.64	35.75	35.59	35.66
	50	36.49	36.93	36.45	36.39	35.46	35.93	35.37	35.43	35.58	35.57
	Media	36.74	37.05	36.45	36.43	36.02	36.16	35.92	35.82	35.88	35.84
	C.M.	7.6	9.6	7.9	7.1	5.0	5.6	3.8	2.5	3.4	2.6

Valores δ_0	$n=45$	Valores θ									
		10	20	30	40	50	60	70	80	90	100
Valores δ_0	5	34.50	35.52	35.27	35.28	35.13	34.99	35.49	35.40	34.67	35.34
	10	36.02	35.42	35.48	35.62	35.55	34.95	35.75	35.61	35.46	34.96
	15	36.11	36.24	35.71	34.95	35.03	35.40	35.01	34.84	35.05	35.91
	20	35.58	35.97	36.18	35.46	34.16	34.98	35.54	34.71	34.91	35.06
	25	35.88	35.92	35.72	35.39	34.62	35.01	34.01	34.47	34.22	34.67
	30	36.25	35.67	35.79	35.08	34.87	34.33	34.59	34.67	34.20	34.73
	35	35.84	36.38	35.35	34.63	33.99	34.82	34.14	33.84	34.69	34.01
	40	34.94	35.57	35.09	35.31	33.89	33.89	34.59	35.18	34.73	34.37
	45	36.06	36.02	35.50	34.59	34.02	33.41	33.95	33.62	33.95	34.20
	50	35.80	35.99	35.18	34.64	34.20	34.43	34.64	33.70	33.75	33.94
Media		35.70	35.87	35.53	35.09	34.55	34.62	34.77	34.61	34.56	34.72
C.M.		8.1	8.9	7.5	6.4	3.7	3.7	4.9	3.7	3.4	4.7

Valores δ_0	$n=50$	Valores θ									
		10	20	30	40	50	60	70	80	90	100
Valores δ_0	5	32.88	33.93	34.30	34.45	33.46	34.28	34.02	33.61	33.39	33.30
	10	34.36	34.64	33.85	34.12	34.39	34.20	34.14	33.57	33.94	34.16
	15	34.30	34.17	33.71	34.00	33.70	32.69	33.06	33.14	33.02	33.43
	20	34.58	34.12	34.18	33.71	32.90	33.59	33.02	33.14	32.97	32.15
	25	34.31	34.30	34.29	32.69	33.22	32.36	32.85	33.16	32.74	32.16
	30	34.51	32.95	33.26	32.56	32.92	31.63	32.17	32.19	31.97	31.94
	35	33.71	33.63	32.32	32.66	32.02	31.77	31.83	31.68	31.25	31.24
	40	34.26	33.99	33.02	32.69	32.22	31.42	31.46	32.08	31.48	31.50
	45	34.07	33.39	32.97	32.45	31.64	31.82	31.31	31.11	31.48	31.19
	50	35.08	33.62	32.98	31.94	30.90	31.76	31.84	31.31	31.82	31.36
Media		34.21	33.87	33.49	33.13	32.74	32.55	32.57	32.50	32.41	32.24
C.M.		8.9	8.6	7.5	6.7	5.3	4.0	4.4	3.7	3.2	2.7

La figura B1.1 resume de manera gráfica los resultados descritos por las tablas de esta sección. Se utilizaron las medias de los valores de la función objetivo para cada valor de θ para hacer una gráfica por cada n .

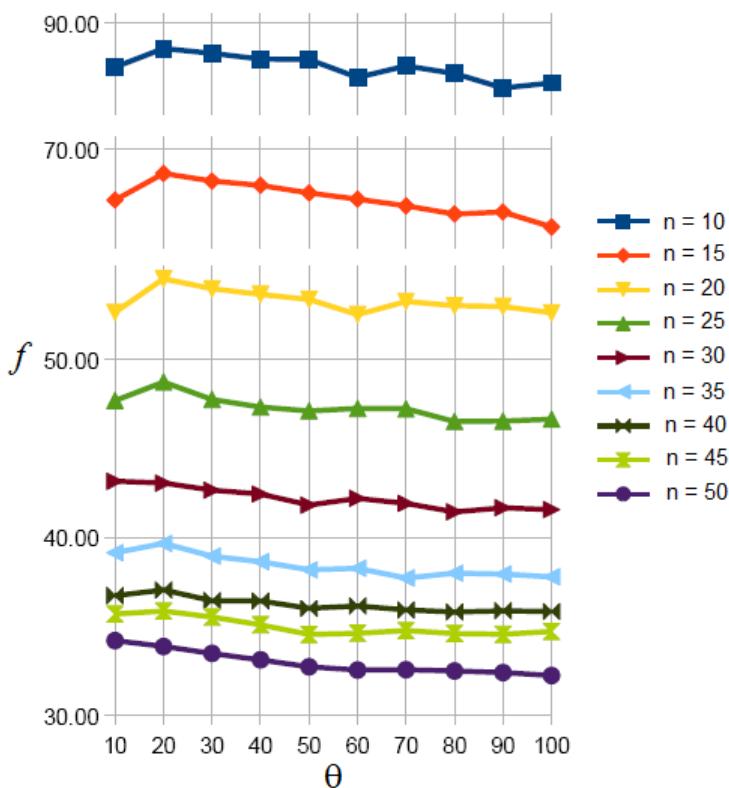


Figura B1.1 Gráficas de los promedios de los valores de la función objetivo por cada valor de θ y para cada n .

B.2. Matrices θ contra δ_0 para valores de la función objetivo

Las siguientes tablas son en realidad las matrices transpuestas del grupo de tablas anteriores. En este grupo de tablas se analizó el comportamiento del algoritmo con respecto a la variación de δ_0 . En todos los casos la prueba de Friedman dió una probabilidad (P) menor que un nivel de confianza del 2 %, por lo cual se determinó que sí existen diferencias estadísticamente significativas con respecto a la variación de δ_0 . En general, se optó por escoger para cada n el valor de δ_0 cuya columna correspondiente obtuviera el valor promedio más alto (**Media**), y que en la mayoría de los casos, coincidió también con la clasificación media más alta (**C.M.**). En algunos casos se utilizó la prueba de Wilcoxon para determinar si existían o no diferencias significativas entre dos columnas con medias muy similares, para poder hacer una elección conveniente. En las tablas se resalta en gris la(s) columna(s) con el mejor resultado.

En la tabla correspondiente a $n = 20$, la media para $\delta_0 = 20$ es mayor que la media para $\delta_0 = 15$ por una pequeña diferencia, pero la clasificación media favorece más a

$\delta_0 = 15$, así que se aplicó una prueba de Wilcoxon entre las columnas correspondientes y se obtuvo una probabilidad de 0.7695, por lo que se descartó que las diferencias entre los datos fueran estadísticamente significativas. Escogemos entonces el valor $\delta_0 = 20$ que tiene la media más alta.

Valores θ	$n=10$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	85.76	87.70	88.60	88.32	88.28	87.03	88.28	86.14	87.12	88.13
	20	86.34	86.30	88.29	89.94	89.29	88.25	86.75	89.77	89.42	91.36
	30	85.14	86.81	87.54	89.55	89.10	88.69	87.78	90.85	89.65	87.92
	40	86.38	86.86	86.24	87.03	90.18	89.17	87.61	88.21	89.05	89.16
	50	86.71	88.16	87.28	88.68	89.83	89.49	87.32	87.87	87.91	86.30
	60	87.11	87.44	86.50	86.49	88.06	87.03	87.76	84.04	86.39	88.75
	70	89.09	84.83	88.11	89.14	89.65	87.32	88.17	87.11	85.33	87.27
	80	88.17	85.84	84.39	90.08	87.99	87.55	87.14	86.75	86.04	87.77
	90	86.17	87.72	85.77	86.33	88.29	86.25	87.73	86.26	84.31	84.76
	100	86.23	85.63	85.29	88.97	86.96	87.09	87.12	88.18	86.61	84.39
	Media	86.71	86.73	86.80	88.45	88.76	87.79	87.57	87.52	87.18	87.58
	C.M.	3.9	3.9	3.8	7.7	8.4	5.9	6.0	5.4	4.6	5.4

Valores θ	$n=15$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	63.43	65.94	67.15	68.21	68.16	67.30	68.77	66.27	67.44	68.86
	20	66.67	68.14	69.25	68.26	67.86	70.12	68.84	69.42	67.77	70.12
	30	67.71	68.98	66.75	68.92	69.68	68.37	67.97	68.23	67.48	68.08
	40	66.96	68.66	68.62	67.81	68.10	67.61	67.77	67.79	67.89	68.55
	50	68.54	67.88	68.65	66.91	67.75	66.95	66.84	67.60	67.13	67.28
	60	67.66	67.39	68.20	67.24	68.10	67.55	66.94	67.35	65.69	65.96
	70	66.37	68.61	67.75	67.95	68.03	67.45	66.61	66.99	64.99	63.49
	80	67.53	66.72	67.10	67.26	66.59	67.07	66.52	64.88	64.16	65.84
	90	67.12	68.28	68.54	66.48	65.02	66.75	65.08	65.77	65.56	66.21
	100	66.46	67.20	67.23	67.64	66.30	64.24	64.52	65.47	63.78	63.51
	Media	66.84	67.78	67.92	67.67	67.56	67.34	66.99	66.98	66.19	66.79
	C.M.	5.1	7.2	7.5	6.5	6.4	5.6	4.0	4.8	2.9	5.0

Valores θ	$n=20$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	49.36	49.08	53.28	54.79	54.56	52.86	53.39	53.08	53.17	53.01
	20	51.20	53.82	56.06	55.72	54.33	53.03	54.29	55.73	55.17	56.07
	30	53.11	55.08	53.80	55.65	54.52	53.76	53.11	53.41	54.70	52.69
	40	52.20	55.11	54.54	54.05	54.17	53.71	53.43	53.23	53.00	53.22
	50	53.37	54.63	53.98	54.29	52.95	53.69	52.60	52.58	52.83	52.91
	60	52.57	54.57	53.10	52.23	53.10	52.60	53.29	51.32	51.10	51.43
	70	53.24	55.15	54.83	52.86	53.19	52.60	52.11	53.38	52.93	52.35
	80	52.55	55.02	53.71	55.29	53.50	52.85	51.30	51.48	52.37	52.23
	90	53.11	54.17	53.71	55.59	53.24	54.39	51.29	51.57	51.40	51.37
	100	53.15	53.68	54.63	53.53	52.76	54.07	52.47	50.86	50.66	50.47
	Media	52.39	54.03	54.16	54.40	53.63	53.36	52.73	52.66	52.73	52.58
	C.M.	4.0	7.8	8.0	7.8	6.6	5.6	3.8	4.1	4.0	3.3

Valores θ	$n=25$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	44.36	45.58	47.79	48.78	50.52	48.34	48.51	47.70	47.17	48.20
	20	46.15	47.53	48.59	48.95	49.87	49.98	49.54	48.81	49.14	48.83
	30	45.80	47.55	48.60	48.24	48.05	48.64	48.07	47.92	47.52	47.21
	40	46.69	47.53	47.07	48.33	48.50	47.08	47.40	48.40	46.55	45.74
	50	46.55	46.66	47.75	47.75	47.92	46.92	46.62	47.12	47.03	46.79
	60	46.91	46.07	47.82	47.94	47.82	48.02	47.79	46.71	46.47	46.96
	70	45.96	47.16	47.96	47.54	48.68	46.57	47.87	47.84	46.99	45.77
	80	45.62	46.11	47.69	49.12	47.18	47.04	45.07	46.33	45.03	45.99
	90	47.39	47.70	47.80	46.52	46.67	46.47	45.82	46.23	46.16	44.62
	100	46.35	47.73	47.29	46.89	48.16	47.43	45.54	45.69	46.41	44.94
	Media	46.18	46.96	47.84	48.01	48.34	47.65	47.22	47.28	46.85	46.50
	C.M.	2.8	4.7	7.2	7.7	8.7	7.0	5.1	5.2	3.6	3.0

Valores θ	$n=30$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	36.93	41.21	43.05	43.58	44.58	44.73	43.23	45.26	44.81	44.23
	20	40.10	40.91	42.27	44.20	43.56	43.75	44.35	44.00	43.73	43.74
	30	38.88	42.71	42.93	43.45	43.41	43.09	43.06	43.14	43.61	42.31
	40	40.22	41.62	43.28	43.05	43.31	42.59	42.75	43.33	42.57	41.69
	50	39.54	42.67	42.68	42.98	43.03	42.29	41.81	41.09	40.97	41.16
	60	40.95	42.41	43.29	42.53	43.59	42.31	41.70	41.86	42.05	41.32
	70	41.02	41.93	42.80	43.71	42.14	42.99	42.86	40.64	40.92	40.16
	80	40.16	42.05	43.40	42.74	43.20	42.22	40.87	39.79	40.54	39.43
	90	41.17	42.81	42.30	43.56	42.23	41.71	40.87	41.48	39.97	40.60
	100	41.45	41.98	42.88	42.82	41.68	41.17	41.88	41.03	40.63	40.17
	Media	40.04	42.03	42.89	43.26	43.07	42.69	42.34	42.16	41.98	41.48
	C.M.	2.2	5.1	7.0	8.4	7.6	6.4	5.6	5.4	4.5	2.8

Valores θ	$n=35$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	35.68	38.48	39.75	39.81	39.17	38.50	40.26	39.27	40.58	39.94
	20	38.02	39.91	39.64	39.93	40.08	40.23	40.23	39.52	39.27	39.83
	30	38.26	39.17	39.39	40.04	39.27	39.11	38.13	37.85	39.18	39.10
	40	38.11	39.08	39.33	38.26	38.81	39.34	39.24	37.31	38.99	37.99
	50	38.81	38.74	38.20	38.45	38.23	38.35	38.00	37.10	37.63	38.31
	60	38.21	38.87	38.33	38.57	38.83	38.30	38.80	37.62	37.44	37.70
	70	36.80	38.23	38.93	39.07	38.72	37.42	36.77	37.20	36.93	37.21
	80	38.75	39.23	38.62	38.75	38.28	37.83	37.12	36.91	37.34	37.10
	90	38.23	37.73	39.10	38.73	37.73	38.02	38.02	38.06	36.78	37.07
	100	37.83	38.43	38.86	38.84	37.97	37.65	37.34	36.59	37.35	37.01
	Media	37.87	38.79	39.01	39.05	38.71	38.48	38.39	37.74	38.15	38.13
	C.M.	4.7	6.8	7.4	8.0	6.4	6.0	5.3	2.6	4.0	3.9

Valores θ	$n=40$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	35.57	36.46	37.04	37.15	36.88	36.98	36.45	37.21	37.15	36.49
	20	36.58	37.19	37.10	37.26	37.28	36.98	37.33	37.28	36.57	36.93
	30	36.85	36.86	36.37	36.50	36.35	36.20	36.10	36.33	36.50	36.45
	40	36.50	36.52	37.06	36.19	36.26	36.54	36.76	36.24	35.87	36.39
	50	36.50	36.71	36.50	36.21	35.50	35.97	35.84	35.78	35.73	35.46
	60	36.67	36.69	36.37	36.06	35.79	36.18	36.00	36.09	35.79	35.93
	70	36.63	36.48	35.92	36.15	36.04	35.82	35.46	35.73	35.64	35.37
	80	36.31	36.27	35.89	35.94	35.85	35.76	35.65	35.31	35.75	35.43
	90	36.00	35.92	36.25	36.08	36.30	35.76	35.71	35.64	35.59	35.58
	100	36.29	36.30	35.69	35.98	35.89	35.72	35.69	35.63	35.66	35.57
	Media	36.39	36.54	36.42	36.35	36.21	36.19	36.10	36.12	36.03	35.96
	C.M.	7.1	8.0	7.1	6.8	5.6	5.4	4.4	4.4	3.6	2.7

Valores θ	$n=45$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
	10	34.50	36.02	36.11	35.58	35.88	36.25	35.84	34.94	36.06	35.80
	20	35.52	35.42	36.24	35.97	35.92	35.67	36.38	35.57	36.02	35.99
	30	35.27	35.48	35.71	36.18	35.72	35.79	35.35	35.09	35.50	35.18
	40	35.28	35.62	34.95	35.46	35.39	35.08	34.63	35.31	34.59	34.64
	50	35.13	35.55	35.03	34.16	34.62	34.87	33.99	33.89	34.02	34.20
	60	34.99	34.95	35.40	34.98	35.01	34.33	34.82	33.89	33.41	34.43
	70	35.49	35.75	35.01	35.54	34.01	34.59	34.14	34.59	33.95	34.64
	80	35.40	35.61	34.84	34.71	34.47	34.67	33.84	35.18	33.62	33.70
	90	34.67	35.46	35.05	34.91	34.22	34.20	34.69	34.73	33.95	33.75
	100	35.34	34.96	35.91	35.06	34.67	34.73	34.01	34.37	34.20	33.94
	Media	35.16	35.48	35.42	35.25	34.99	35.02	34.77	34.75	34.53	34.63
	C.M.	6.0	7.6	8.0	7.0	5.7	5.7	4.2	3.9	3.4	3.5

Valores δ_0	$n=50$	Valores δ_0									
		5	10	15	20	25	30	35	40	45	50
10	32.88	34.36	34.30	34.58	34.31	34.51	33.71	34.26	34.07	35.08	
20	33.93	34.64	34.17	34.12	34.30	32.95	33.63	33.99	33.39	33.62	
30	34.30	33.85	33.71	34.18	34.29	33.26	32.32	33.02	32.97	32.98	
40	34.45	34.12	34.00	33.71	32.69	32.56	32.66	32.69	32.45	31.94	
50	33.46	34.39	33.70	32.90	33.22	32.92	32.02	32.22	31.64	30.90	
60	34.28	34.20	32.69	33.59	32.36	31.63	31.77	31.42	31.82	31.76	
70	34.02	34.14	33.06	33.02	32.85	32.17	31.83	31.46	31.31	31.84	
80	33.61	33.57	33.14	33.14	33.16	32.19	31.68	32.08	31.11	31.31	
90	33.39	33.94	33.02	32.97	32.74	31.97	31.25	31.48	31.48	31.82	
100	33.30	34.16	33.43	32.15	32.16	31.94	31.24	31.50	31.19	31.36	
Media	33.76	34.14	33.52	33.44	33.21	32.61	32.21	32.41	32.14	32.26	
C.M.	8.0	9.1	7.5	7.0	7.0	4.5	2.7	3.6	2.2	3.4	

En la figura B2.1 se resume de manera gráfica los resultados descritos por las tablas de esta sección. Se utilizaron las medias de los valores de la función objetivo para cada valor de δ_0 para hacer una gráfica por cada n .

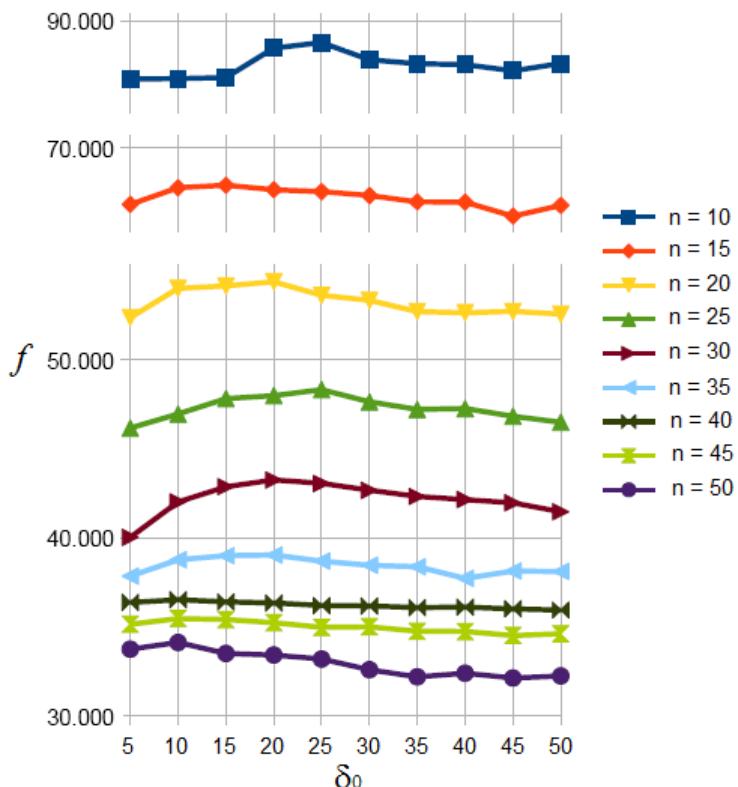


Figura B2.1 Gráficas de los promedios de los valores de la función objetivo por cada δ_0 y para cada valor de n .

B.3. Matrices λ contra φ para valores de la función objetivo y tiempo de cálculo

En las siguientes tablas se muestran los resultados experimentales para determinar un valor apropiado de θ para cada valor de $n = 10, 15, \dots, 50$. Con la información recabada de los experimentos se construyó una matriz $\{b(n)_{ij}\}$ (para cada n) con los promedios de los valores de la función objetivo, calculados mediante la ejecución del algoritmo de mejora sucesiva para nuestro problema de optimización. La entrada $b(n)_{ij}$ es el promedio de los valores de la función objetivo en cinco ejecuciones, para un valor de n dado, con los parámetros: λ correspondiente al valor asignado al renglón i , y φ correspondiente al valor asignado a la columna j . Por ejemplo, la entrada $b(10)_{11} = 89.98$ corresponde al promedio de cinco ejecuciones del algoritmo de mejora sucesiva para $n = 10$, con $\lambda=20$ y $\varphi=0.75$. Asimismo, se construyeron las matrices $\{c(n)_{ij}\}$ de los tiempos de cálculo (medido en segundos). La entrada $c(n)_{ij}$ es el promedio de los tiempos que se utilizaron para calcular $b(n)_{ij}$.

Como criterio de paro se optó por fijar un cero virtual $\varepsilon = 0.5$ (la elección de este valor se discutió en la sección 4.3). En este grupo de tablas se analizó el comportamiento del algoritmo con respecto a la variación de φ . En todos los casos la prueba de Friedman arrojó una probabilidad (**P**) menor que un nivel de confianza del 0.01 %, por lo cual se determinó que sí existen diferencias estadísticamente significativas con respecto a la variación de φ .

En las tablas se puede apreciar de manera consistente que a mayor valor de φ , mayor valor de la función objetivo. Para cada valor de n se muestran las tablas de la función objetivo y de los tiempos de cálculo correspondientes, dieciocho en total. El tiempo también se incrementa de manera consistente al incrementarse el valor de φ . La elección del parámetro φ depende de qué tanta calidad requiera el usuario en la solución y qué tanto tiempo quiera invertir en ello. Se sugiere utilizar el valor $\varphi = 0.9$, pues entrega soluciones cercanas a aquéllas otorgadas por $\varphi = 0.95$, pero a un costo computacional menor de casi un 50 % en tiempo.

		Función Objetivo, $n=30$							Tiempo (s), $n=30$				
Valores λ	Valores φ	Valores φ					Valores λ	Valores φ	Valores φ				
		0.75	0.80	0.85	0.90	0.95			0.75	0.80	0.85	0.90	0.95
	20	47.77	48.20	48.06	49.24	50.61			10.61	11.30	12.94	18.14	32.98
	40	48.67	49.58	49.84	50.71	51.26			17.91	21.55	24.75	33.96	59.13
	60	49.92	49.89	50.70	51.07	51.77			23.14	27.45	40.42	50.49	88.50
	80	50.43	50.42	50.53	50.90	51.49			29.62	34.69	47.41	63.67	108.74
	100	50.34	50.71	51.07	51.41	52.05			38.40	45.57	55.96	81.83	143.68
Media		49.43	49.76	50.04	50.66	51.43							
C.M.		1.4	1.8	2.8	4	5							
P=		0.0000											

		Función Objetivo, $n=35$							Tiempo (s), $n=35$				
Valores λ	Valores φ	Valores φ					Valores λ	Valores φ	Valores φ				
		0.75	0.80	0.85	0.90	0.95			0.75	0.80	0.85	0.90	0.95
	20	44.37	44.11	44.92	45.56	47.08			11.91	15.36	18.03	22.92	47.92
	40	44.30	45.72	46.41	46.57	47.23			22.81	30.53	40.34	51.46	89.15
	60	45.89	46.86	47.15	47.18	47.84			34.16	37.51	46.81	59.61	120.16
	80	46.36	46.99	47.29	47.87	47.77			38.94	58.40	70.20	88.13	160.42
	100	46.56	47.41	47.46	47.65	48.57			53.89	73.69	91.61	106.16	179.37
Media		45.50	46.22	46.64	46.97	47.70							
C.M.		1.2	1.8	3	4.2	4.8							
P=		0.0000											

		Función Objetivo, $n=40$							Tiempo (s), $n=40$				
Valores λ	Valores φ	Valores φ					Valores λ	Valores φ	Valores φ				
		0.75	0.80	0.85	0.90	0.95			0.75	0.80	0.85	0.90	0.95
	20	40.11	40.89	41.45	42.78	43.19			12.13	18.64	19.99	24.05	42.29
	40	42.05	43.13	42.65	43.43	44.47			22.14	30.12	34.04	45.67	76.28
	60	43.01	42.84	43.49	43.86	44.41			36.88	39.85	48.36	69.04	120.23
	80	43.46	44.13	44.08	44.37	45.21			42.69	52.51	62.66	85.40	157.28
	100	42.78	43.53	44.21	44.38	45.19			45.69	59.20	82.99	108.30	184.56
Media		42.28	42.90	43.17	43.76	44.49							
C.M.		1.2	2.2	2.6	4	5							
P=		0.0000											

		Función Objetivo, $n=45$							Tiempo (s), $n=45$				
Valores λ	Valores λ	Valores φ					Valores λ	Valores λ	Valores φ				
		0.75	0.80	0.85	0.90	0.95			0.75	0.80	0.85	0.90	0.95
	20	38.27	39.11	40.11	39.48	40.81		20	14.51	15.78	22.25	24.97	40.26
	40	38.95	40.13	40.43	40.77	41.52		40	20.58	30.67	46.50	73.35	104.29
	60	40.81	40.58	40.72	41.39	41.79		60	36.89	44.91	79.22	101.55	178.75
	80	40.90	41.40	41.73	41.84	41.86		80	46.81	58.74	91.87	135.20	226.07
	100	40.73	41.19	41.58	41.75	42.48		100	67.69	69.58	93.76	168.24	222.39
Media		39.93	40.48	40.92	41.05	41.69							
C.M.		1.4	1.8	3	3.8	5							
P=		0.0000											

		Función Objetivo, $n=50$							Tiempo (s), $n=50$				
Valores λ	Valores λ	Valores φ					Valores λ	Valores λ	Valores φ				
		0.75	0.80	0.85	0.90	0.95			0.75	0.80	0.85	0.90	0.95
	20	36.46	37.23	37.92	37.76	38.58		20	14.59	19.98	29.31	29.77	51.40
	40	37.87	38.47	38.98	38.58	39.74		40	25.66	32.73	41.93	55.71	95.73
	60	37.98	38.66	38.85	39.48	39.74		60	40.40	39.94	53.08	78.52	128.47
	80	38.18	39.01	39.23	39.57	40.08		80	46.63	60.10	69.66	92.65	182.99
	100	38.40	39.06	39.70	39.61	40.23		100	56.77	61.57	82.78	103.96	200.63
Media		37.78	38.49	38.94	39.00	39.67							
C.M.		1	2	3.6	3.4	5							
P=		0.0000											

B.4. Matrices φ contra λ para valores de la función objetivo

El siguiente grupo de tablas son en realidad las matrices transpuestas de la sección anterior. Análogamente, se analizó el comportamiento del algoritmo al variar el valor de λ , tomando valores en $\{20, 40, 60, 80, 100\}$. La prueba de Friedman arrojó una probabilidad $P < 0.003$ en todos los casos, por lo que podemos afirmar que sí existen diferencias estadísticas significativas en el conjunto de datos con respecto a la variación de λ .

En general, se puede apreciar en las tablas que, al incrementar el valor de λ se incrementa también el valor de la función objetivo. Esto lo refleja claramente tanto la media como la clasificación media. Si revisamos las tablas de los tiempos correspondientes, en la sección anterior, notaremos también que el tiempo se incrementa a la par de elevar el valor de λ . Para reducir los tiempos de cómputo, se eligió el valor intermedio de $\lambda = 60$ para los cálculos de los experimentos finales, pero queda a voluntad del usuario escoger el valor que más le convenga en términos de calidad / tiempo.

Función Objetivo, $n=10$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	89.98	93.67	91.65	93.48	94.78
	0.80	90.96	93.82	94.45	94.79	95.31
	0.85	91.05	93.95	91.99	94.99	93.80
	0.90	92.93	94.70	95.67	94.04	95.50
	0.95	94.25	95.22	95.45	95.46	95.53
	Media	91.83	94.27	93.84	94.55	94.98
	C.M.	1	3	3	3.6	4.4
P=		0.0020				

Función Objetivo, $n=15$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	72.24	73.09	74.23	73.64	72.96
	0.80	72.45	73.48	74.43	74.77	74.70
	0.85	72.71	73.44	74.78	74.69	75.82
	0.90	72.37	74.18	74.78	73.91	75.36
	0.95	74.48	75.52	75.15	75.02	76.33
	Media	72.85	73.94	74.67	74.41	75.04
	C.M.	1	2.8	3.8	3.2	4.2
P=		0.0028				

Función Objetivo, $n=20$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	60.95	61.34	62.40	62.58	62.86
	0.80	60.71	62.35	63.44	62.68	63.38
	0.85	61.49	62.73	62.63	63.34	62.94
	0.90	61.18	63.16	63.60	63.97	63.66
	0.95	63.32	63.01	64.29	63.75	64.06
	Media	61.53	62.52	63.27	63.26	63.38
	C.M.	1.2	2	3.6	4	4.2
P=		0.0004				

Función Objetivo, $n=25$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	53.76	54.57	54.48	54.24	55.15
	0.80	53.04	54.54	55.50	55.16	55.76
	0.85	53.53	55.33	55.48	55.45	56.78
	0.90	55.03	55.34	56.10	56.58	56.74
	0.95	55.21	55.70	56.61	57.09	57.40
	Media	54.11	55.09	55.63	55.70	56.36
	C.M.	1	2.4	3.4	3.2	5
P=		0.0000				

Función Objetivo, $n=30$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	47.77	48.67	49.92	50.43	50.34
	0.80	48.20	49.58	49.89	50.42	50.71
	0.85	48.06	49.84	50.70	50.53	51.07
	0.90	49.24	50.71	51.07	50.90	51.41
	0.95	50.61	51.26	51.77	51.49	52.05
	Media	48.78	50.01	50.67	50.75	51.11
C.M.		1	2	3.6	3.6	4.8
P=		0.0000				

Función Objetivo, $n=35$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	44.37	44.30	45.89	46.36	46.56
	0.80	44.11	45.72	46.86	46.99	47.41
	0.85	44.92	46.41	47.15	47.29	47.46
	0.90	45.56	46.57	47.18	47.87	47.65
	0.95	47.08	47.23	47.84	47.77	48.57
	Media	45.21	46.05	46.98	47.25	47.53
C.M.		1.2	1.8	3.2	4	4.8
P=		0.0000				

Función Objetivo, $n=40$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	40.11	42.05	43.01	43.46	42.78
	0.80	40.89	43.13	42.84	44.13	43.53
	0.85	41.45	42.65	43.49	44.08	44.21
	0.90	42.78	43.43	43.86	44.37	44.38
	0.95	43.19	44.47	44.41	45.21	45.19
	Media	41.69	43.15	43.52	44.25	44.02
C.M.		1	2.4	2.8	4.6	4.2
P=		0.0000				

Función Objetivo, $n=45$

		Valores λ				
		20	40	60	80	100
Valores φ	0.75	38.27	38.95	40.81	40.90	40.73
	0.80	39.11	40.13	40.58	41.40	41.19
	0.85	40.11	40.43	40.72	41.73	41.58
	0.90	39.48	40.77	41.39	41.84	41.75
	0.95	40.81	41.52	41.79	41.86	42.48
	Media	39.56	40.36	41.06	41.55	41.55
C.M.		1	2	3.2	4.8	4
P=		0.0000				

		Función Objetivo, $n=50$				
		Valores λ				
Valores φ	0.75	36.46	37.87	37.98	38.18	38.40
	0.80	37.23	38.47	38.66	39.01	39.06
	0.85	37.92	38.98	38.85	39.23	39.70
	0.90	37.76	38.58	39.48	39.57	39.61
	0.95	38.58	39.74	39.74	40.08	40.23
	Media	37.59	38.73	38.94	39.21	39.40
	C.M.	1	2.4	2.6	4	5
	P=	0.0000				

B.5. Matrices de observaciones de las ocho variantes del algoritmo para valores de la función objetivo

El siguiente grupo de tablas representan los datos recolectados en el experimento final, el cual consistió en probar ocho variantes del algoritmo de umbral, descritas en la sección 8.1, para $n = 10, 15, \dots, 50$. En este caso, los renglones de las matrices representan distintas observaciones para las ocho variantes del algoritmo, mientras que las columnas corresponden a las mediciones recabadas por cada variante del algoritmo. Para cada valor de n se obtuvo una matriz $\{d(n)_{ij}\}$.

La entrada $d(n)_{ij}$ es el valor de la función objetivo que se obtuvo en la ejecución i con la variante del algoritmo correspondiente a la columna j . Los parámetros utilizados fueron aquellos que se describieron en secciones anteriores. El factor de enfriamiento $\varphi = 0.9$, valor del lote $\lambda = 60$ y cero virtual $\varepsilon = 0.5$. En cada renglón se resaltó en negritas el valor más alto de la función objetivo. El último renglón de cada tabla registra el promedio de las observaciones.

Observaciones	Valores de la función objetivo para $n = 10$							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	95.11	94.98	94.53	90.42	95.91	93.26	92.25	93.87
2	94.89	91.88	95.19	87.99	95.33	96.07	92.33	94.82
3	95.41	95.63	94.99	93.00	96.20	89.92	90.54	93.50
4	93.84	95.37	92.31	94.27	95.24	96.16	94.46	94.57
5	94.96	92.55	92.97	93.01	93.83	94.16	89.48	88.32
6	92.97	95.34	92.16	91.08	95.76	95.69	93.63	91.19
7	95.75	95.42	89.85	91.76	93.32	96.20	92.39	93.27
8	94.73	91.69	91.88	92.33	95.90	94.12	91.59	90.81
Media:	94.71	94.11	92.99	91.73	95.19	94.45	92.09	92.54

Valores de la función objetivo para $n = 15$

Variantes del método

	Variables del método								
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	73.71	71.50	73.50	75.41	74.28	76.02	75.49	74.51
	2	71.22	74.94	70.40	73.93	74.15	74.35	73.26	73.67
	3	74.41	72.20	71.51	72.46	72.48	74.02	71.75	73.84
	4	74.22	73.19	69.74	73.92	76.32	73.03	72.76	71.68
	5	70.88	73.91	72.14	73.20	75.25	75.83	73.51	74.16
	6	74.33	74.95	75.67	72.59	74.40	75.02	76.29	74.46
	7	74.02	74.34	73.34	70.64	73.87	74.88	75.38	73.15
	8	73.39	74.52	73.81	71.06	73.46	75.50	74.20	73.04
Media:	73.27	73.69	72.51	72.90	74.28	74.83	74.08	73.56	

Valores de la función objetivo para $n = 20$

Variantes del método

	Variables del método								
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	63.68	61.25	59.81	61.93	61.65	60.58	61.27	61.95
	2	62.75	61.20	58.60	58.65	64.34	64.15	61.63	60.40
	3	62.07	62.85	60.61	61.15	63.52	63.20	63.92	61.74
	4	62.80	62.95	62.72	60.12	61.65	63.43	63.15	61.96
	5	64.01	63.76	59.55	60.65	62.68	64.66	61.29	62.73
	6	63.97	63.13	59.10	61.83	64.18	63.06	61.06	60.24
	7	63.61	61.85	61.68	62.25	63.61	63.59	61.74	61.38
	8	63.04	62.25	60.69	61.66	61.91	63.16	62.75	62.41
Media:	63.24	62.41	60.35	61.03	62.94	63.23	62.10	61.60	

Valores de la función objetivo para $n = 25$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	55.35	53.54	53.34	53.98	57.16	55.74	53.76	54.20
	2	55.37	55.85	53.51	55.18	54.72	57.13	55.10	52.54
	3	55.64	56.03	53.89	53.81	55.23	56.34	54.76	55.64
	4	54.67	55.76	53.07	54.19	56.35	55.94	55.61	56.57
	5	54.24	54.39	54.77	54.47	56.61	56.97	53.06	55.20
	6	54.83	54.51	53.10	54.53	57.31	55.77	53.78	52.81
	7	54.50	55.03	53.42	54.14	55.53	55.92	53.80	52.60
	8	56.84	55.08	53.86	53.73	56.73	55.86	54.40	54.89
Media:	55.18	55.02	53.62	54.25	56.20	56.21	54.28	54.30	

Valores de la función objetivo para $n = 30$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	48.73	49.06	49.08	48.56	50.45	50.39	50.72	48.49
	2	51.11	50.24	48.16	48.49	50.70	51.12	50.40	51.15
	3	49.63	48.50	49.68	48.04	51.13	51.15	50.56	49.89
	4	49.76	49.60	49.82	49.09	50.55	49.35	50.81	49.62
	5	50.66	50.38	47.82	47.87	51.49	49.44	48.66	48.67
	6	51.40	51.02	48.49	48.63	51.59	49.81	50.11	48.96
	7	50.20	49.10	48.26	47.25	49.87	50.75	49.70	49.27
	8	52.04	50.42	47.98	49.00	50.53	51.99	49.93	48.88
Media:	50.44	49.79	48.66	48.37	50.79	50.50	50.11	49.37	

Valores de la función objetivo para $n = 35$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	44.61	47.60	44.59	46.44	46.12	47.50	46.37	46.74
	2	46.42	45.70	46.15	46.26	47.61	45.06	46.69	46.41
	3	45.21	45.66	44.11	46.70	47.88	47.44	46.29	46.52
	4	46.53	45.32	45.30	45.91	46.86	47.66	46.56	45.37
	5	45.74	45.81	45.96	43.29	45.41	46.07	47.07	45.04
	6	47.43	44.69	45.54	47.28	47.39	47.26	45.68	46.70
	7	46.20	46.11	46.70	45.46	46.88	46.77	46.59	45.14
	8	45.29	45.15	45.73	46.24	47.67	47.12	45.95	44.73
Media:	45.93	45.76	45.51	45.95	46.98	46.86	46.40	45.83	

Valores de la función objetivo para $n = 40$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	43.33	42.92	40.80	40.77	44.41	42.73	42.32	41.50
	2	42.18	43.08	42.59	40.84	44.12	43.34	41.93	41.51
	3	43.18	44.00	42.21	41.02	42.97	43.74	42.52	42.05
	4	42.86	42.50	42.62	40.68	44.13	44.28	43.22	40.79
	5	42.66	41.23	41.14	39.75	43.23	44.53	41.87	42.59
	6	41.55	41.35	42.09	39.32	42.98	43.14	42.01	42.01
	7	43.71	42.54	40.70	41.09	44.65	44.44	42.42	42.30
	8	43.18	42.23	41.05	39.64	43.56	43.44	41.23	42.45
Media:	42.83	42.48	41.65	40.39	43.76	43.71	42.19	41.90	

Valores de la función objetivo para $n = 45$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	40.32	41.16	38.28	39.33	41.67	40.18	39.94	39.00
	2	41.37	39.54	38.92	38.75	41.53	41.30	39.95	38.63
	3	40.66	39.37	39.10	37.72	42.32	40.91	40.64	38.99
	4	40.74	40.12	38.57	39.31	41.13	40.94	40.36	39.65
	5	41.18	39.72	39.72	39.35	40.96	40.22	40.38	38.66
	6	39.68	40.14	39.76	38.35	41.04	40.13	40.61	39.31
	7	41.41	39.88	38.79	40.20	41.12	41.55	41.23	39.74
	8	40.16	39.74	39.02	37.14	42.07	41.52	40.26	40.27
Media:	40.69	39.96	39.02	38.77	41.48	40.84	40.42	39.28	

Valores de la función objetivo para $n = 50$

Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	38.09	37.98	36.42	36.93	38.91	39.15	37.19	37.52
	2	38.51	39.06	36.72	36.40	38.82	39.31	37.23	37.16
	3	38.40	37.94	36.58	37.14	39.87	39.72	38.63	37.35
	4	37.92	36.84	38.08	36.20	39.27	39.67	37.55	38.11
	5	38.41	37.46	36.58	36.60	39.62	38.36	37.65	36.51
	6	37.81	38.29	36.84	37.31	39.54	37.21	38.25	36.83
	7	38.28	38.63	36.39	35.28	39.79	39.95	37.15	37.46
	8	37.51	37.95	36.72	36.74	40.32	40.25	37.88	36.96
Media:	38.12	38.02	36.79	36.58	39.52	39.20	37.69	37.24	

B.6. Matrices de observaciones de las ocho variantes del algoritmo para valores de tiempo de cálculo

El siguiente grupo de tablas representa los tiempos de cálculo correspondientes de las tablas de la sección anterior. El último renglón en cada tabla registra el promedio de los tiempos de cada columna.

Tiempo (s), $n=10$ Variantes del método

	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	4.54	10.01	7.51	3.30	7.73	8.40	6.46	7.57
	2	4.74	10.43	7.06	5.05	9.63	9.55	6.88	4.62
	3	4.52	9.72	7.14	5.65	10.24	7.60	6.38	7.58
	4	7.68	8.97	6.49	4.33	8.71	9.47	5.18	5.66
	5	5.31	10.15	7.05	6.93	11.74	9.59	4.99	6.45
	6	5.03	7.96	6.81	6.79	10.41	9.87	6.70	5.75
	7	9.36	7.98	6.42	9.26	5.81	6.85	8.18	7.59
	8	9.32	10.21	7.46	5.65	8.79	8.59	6.11	6.45
Media:	6.31	9.43	6.99	5.87	9.13	8.74	6.36	6.46	

Observaciones	Tiempo (s), <i>n</i> =15							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	15.63	7.02	6.10	12.75	10.66	13.77	12.45	10.39
2	11.80	6.68	7.39	13.26	10.43	13.28	13.80	13.71
3	14.85	10.44	6.17	11.43	14.42	12.74	11.18	13.53
4	19.54	7.41	12.88	12.93	15.70	16.44	10.92	14.90
5	13.01	9.71	13.87	11.14	13.60	17.99	11.45	13.14
6	12.85	15.24	12.96	12.72	14.48	15.70	12.22	12.57
7	18.41	12.95	11.58	9.36	14.60	14.15	14.12	13.06
8	18.05	13.44	11.33	12.31	11.73	14.51	10.95	14.95
Media:	15.52	10.36	10.28	11.99	13.20	14.82	12.14	13.28

Observaciones	Tiempo (s), <i>n</i> =20							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	15.96	24.48	19.09	17.42	23.62	21.47	19.38	18.97
2	27.01	25.78	15.14	18.62	24.21	25.99	16.06	18.09
3	24.13	17.31	17.32	16.88	24.89	27.60	21.47	13.64
4	24.08	17.79	17.61	12.75	12.90	24.58	15.76	19.14
5	26.61	17.16	14.92	11.52	23.75	19.85	22.07	16.67
6	18.06	23.41	19.67	22.04	24.26	18.19	18.90	15.81
7	24.08	26.48	21.62	22.95	22.34	25.37	15.65	11.50
8	24.57	23.39	16.89	17.02	18.82	18.76	14.23	23.28
Media:	23.06	21.97	17.78	17.40	21.85	22.73	17.94	17.14

Observaciones	Tiempo (s), <i>n</i> =25							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	44.73	41.49	29.19	29.94	38.94	39.72	29.75	31.51
2	41.57	28.39	22.30	29.81	42.57	42.76	31.68	28.52
3	42.40	45.93	33.42	33.90	32.34	36.01	32.05	28.14
4	35.77	31.38	31.01	33.53	36.99	39.33	30.72	31.72
5	36.27	35.82	33.08	32.48	34.88	38.59	31.09	30.37
6	42.84	31.30	30.56	33.87	47.13	47.91	28.90	25.91
7	42.57	29.89	30.99	36.35	33.42	44.86	30.25	31.16
8	43.73	42.13	31.22	33.83	40.69	39.03	31.36	31.19
Media:	41.23	35.79	30.22	32.96	38.37	41.03	30.72	29.82

	Tiempo (s), <i>n</i> =30								
	Variantes del método								
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	55.15	47.06	40.85	47.08	56.11	46.13	38.02	46.68
	2	62.04	50.04	36.57	43.53	52.49	57.58	41.57	44.18
	3	58.41	47.28	34.54	45.37	49.65	52.15	40.02	46.20
	4	55.83	63.13	43.33	46.42	46.13	37.84	37.54	42.92
	5	64.26	52.66	34.58	54.12	56.91	44.09	32.63	39.68
	6	62.21	44.91	36.80	47.31	45.69	42.72	30.24	46.77
	7	54.35	39.09	33.33	48.53	45.29	53.84	28.64	54.40
	8	59.03	42.63	40.05	48.97	49.87	55.26	42.17	44.51
Media:		58.91	48.35	37.51	47.67	50.27	48.70	36.35	45.67

	Tiempo (s), <i>n</i> =35								
	Variantes del método								
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	64.63	81.69	47.70	66.49	55.90	63.99	62.27	66.01
	2	62.61	74.71	45.22	66.93	59.51	44.18	65.02	71.51
	3	62.53	68.85	45.37	68.51	67.42	72.28	64.65	67.38
	4	75.53	76.04	48.66	75.55	55.60	66.55	66.05	57.53
	5	73.48	73.07	56.84	52.14	66.29	64.68	60.53	66.65
	6	75.43	75.05	54.98	71.05	61.75	67.60	54.70	71.38
	7	64.83	79.78	43.10	56.57	61.29	69.60	60.67	64.21
	8	67.64	70.66	59.26	67.70	62.04	59.68	52.11	66.15
Media:		68.34	74.98	50.14	65.62	61.22	63.57	60.75	66.35

	Tiempo (s), <i>n</i> =40								
	Variantes del método								
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1	
Observaciones	1	79.90	89.54	62.01	60.10	69.17	62.20	49.62	67.48
	2	79.25	89.92	36.08	65.62	63.69	64.68	49.14	68.48
	3	82.90	84.71	49.31	58.18	51.64	86.57	40.17	67.98
	4	87.56	76.03	46.42	83.90	63.50	72.34	39.81	52.85
	5	77.04	83.05	58.17	70.61	69.67	75.81	45.70	70.17
	6	60.15	76.39	63.48	72.87	62.91	62.49	48.12	69.77
	7	70.85	86.11	52.16	64.44	72.69	71.85	66.16	61.50
	8	61.14	80.04	51.48	55.15	78.97	70.76	48.02	73.67
Media:		74.85	83.22	52.39	66.36	66.53	70.84	48.34	66.49

Observaciones	Tiempo (s), n=45							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	83.78	119.51	59.22	96.45	84.38	91.38	71.17	71.53
2	96.55	121.36	53.93	78.39	85.64	102.93	67.51	66.91
3	86.39	108.81	63.05	81.79	88.99	88.53	70.90	73.82
4	82.74	102.35	70.52	99.95	90.89	79.02	66.29	72.68
5	109.53	110.42	63.81	91.07	91.59	78.13	70.67	75.09
6	99.13	102.73	67.38	82.19	89.79	76.87	57.36	84.85
7	99.37	71.17	69.68	93.89	84.98	92.98	74.54	92.15
8	96.51	94.51	67.25	85.16	88.99	95.81	72.68	90.83
Media:	94.25	103.86	64.35	88.61	88.16	88.21	68.89	78.48

Observaciones	Tiempo (s), n=50							
	Variantes del método							
	RS-g-0	RS-g-1	RS-h-0	RS-h-1	UA-g-0	UA-g-1	UA-h-0	UA-h-1
1	133.10	123.67	59.65	95.96	100.37	110.21	79.57	93.93
2	77.59	141.27	65.19	102.06	110.67	112.72	75.45	96.49
3	94.10	120.80	60.02	97.58	117.27	116.57	69.61	100.81
4	105.02	135.40	67.27	97.80	112.33	128.61	83.40	123.11
5	94.99	141.83	68.03	92.10	114.97	111.64	87.09	84.46
6	82.38	147.73	80.72	114.35	101.00	75.59	108.83	95.34
7	100.51	130.94	69.69	100.37	133.93	119.88	82.81	86.81
8	100.55	121.95	73.05	104.62	96.07	120.24	89.57	87.95
Media:	98.53	132.95	67.95	100.61	110.83	111.93	84.54	96.11

Bibliografía

- [Ada42] David L. Mac Adam, *Visual sensitivities to color differences in daylight*, Journal of the Optical Society of America **32** (1942), no. 5, 247–274.
- [Ado00a] Adobe, *The cie color models*, 2000, Technical Guides.
- [Ado00b] ———, *Color models cieluv*, 2000, Technical Guides.
- [Ado00c] ———, *Color models ciexyz*, 2000, Technical Guides.
- [Ado08] Adoniscik, *Spectral locus in xyz and xy*, <http://commons.wikimedia.org/wiki/File:Spectral-locus.png>, April 2008.
- [AK91] Ingo Althöfer and Klaus-Uwe Koschnick, *On the convergence of "threshold accepting"*, Applied Mathematics and Optimization **24** (1991), 183–195.
- [AKvL97] Emile H. L. Aarts, H. M. Korst, and Peter J. M. van Laarhoven, *Local search in combinatorial optimization. chapter 4: Simulated annealing*, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, 1997.
- [AL85] Emile H. L. Aarts and P. J. M. Van Laarhoven, *Statistical cooling: a general approach to combinatorial optimization problems*, Philips Journal of Research **40** (1985), 193–226.
- [AL97] Emile Aarts and Jan Karel Lenstra, *Local search in combinatorial optimization. chapter 1: Introduction*, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, 1997.
- [AM93] Walter Alt and Kazimierz Malanowski, *The lagrange-newton method for nonlinear optimal control problems*, Computational Optimization and Applications **2** (1993), 77–100.

- [BK94] Kenneth D. Bose and Andrew B. Kahng, *Best-so-far vs. where-you-are: Implications for optimal finite-time annealing*, Systems & Control Letters **22** (1994), no. 1, 71–78, UCLA Computer Science Dept., Los Angeles, CA 90024-1596 USA.
- [BKK11] Jan Brandts, Sergey Korotov, and Michael Krizek, *Efficient preconditioned solution methods for elliptic partial differential equations. chapter 6: A geometric toolbox for tetrahedral finite elements partitions*, Bentham Science Publishers Ltd., 2011.
- [Bla08] Craig Blackwell, *Color vision. 2. color matching*, www.youtube.com/watch?v=82ItpxqPP4I, January 2008.
- [BR03] Christiam Blum and Andrea Roli, *Metaheuristics in combinatorial optimization: Overview and conceptual comparison*, ACM Computing Surveys **35** (2003), no. 3, 268–308.
- [CC81] E. C. Carter and R.C. Carter, *Color and conspicuousness*, Journal of the Optical Society of America **71** (1981), no. 6, 723–729.
- [CC82] Robert C. Carter and Ellen C. Carter, *High-contrast sets of colors*, Applied Optics **21** (1982), no. 16, 2936–2939.
- [CCPJ13] Mónica Cardona, M. Angels Colomer, and Mario J. Pérez-Jiménez, *Characterizing the aperiodicity of irreducible markov chains by using p systems*, Dpt. of Mathematics, University of Lleida and Dpt. of Computer Science & Artificial Intelligence, University of Sevilla, 2013.
- [CD81] R. Chandrasekaran and A. Daughety, *Location on tree networks: P-centre and n-dispersion problems*, Mathematics of Operations Research **6** (1981), no. 1, 50–57.
- [Cer85] Vlado Cerny, *Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm*, Journal of Optimization Theory and Applications **45** (1985), no. 1, 41–51.
- [CPS99] Paola Campadelli, Roberto Posenato, and Raimondo Schettini, *An algorithm for the selection of high-contrast color sets*, COLOR research and application **24** (1999), no. 2, 132–138.
- [Dat05] Jon Dattorro, *Convex optimization & euclidean distance geometry*, Meboo Publishing USA, 2005.
- [DS90] G. Dueck and T. Scheuer, *Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing*, Journal of Computational Physics **90** (1990), 161–175.

- [Epp01] David Eppstein, *Global optimization of mesh quality*, Meshing Roundtable, 2001.
- [Erk90] E. Erkut, *The discrete p -dispersion problem*, European Journal of Operational Research **46** (1990), no. 1, 48–60.
- [Eze07] Vanessa Ezekowitz, *File:cones smj2 e.svg*, October 2007.
- [FR98] Adrian Ford and Alan Roberts, *Colour space conversions*, August 1998.
- [Fri37] Milton Friedman, *The use of ranks to avoid the assumption of normality implicit in the analysis ov variance*, Journal of American Statistical Association **32** (1937), no. 200, 675–701.
- [Geo99] Paul Louis George, *Tet mesh: construction, optimization and adaptation*, Proceedings, 8th International Meshing Roundtable **8** (1999), 133–141.
- [Gho96] Jay B. Ghosh, *Computational aspects of the maximum diversity problem*, Operations Research Letters **19** (1996), no. 4, 175–181.
- [GK03] Fred Glover and Gary A. Kochenberger (eds.), *Handbook of metaheuristics*, International Series in Operations Research & Management Science, Kluwer Academic Publishers, 2003.
- [GKD98] Fred Glover, Ching-Chung Kuo, and Krishna S. Dhir, *Heuristic algorithms for the maximum diversity problem*, Journal of Information and Optimization Sciences **19** (1998), no. 1, 109–132.
- [Glo86] Fred Glover, *Future paths for integer programming and links to artificial intelligence*, Compt. & Ops. Res. **13** (1986), no. 5, 533–549.
- [GP97] Edward J. Anderson & Celia A. Glass and Chris N. Potts, *Local search in combinatorial optimization. chapter 11: Machine scheduling.*, Wiley, 1997.
- [Gui32] John Guild, *The colorimetric properties of the spectrum*, Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character **230** (1932), 149–187.
- [GvdHTG06] C.A. Glasbey, G.W.A.M. van der Heijden, V. Toh, and A. J. Gray, *Colour displays for categorical images*, Biomathematics and Statistics Scotland, Biometris, Department of Statistics and Modelling Science, October 2006.

- [GW88] Larry Goldstein and Michael Waterman, *Neighborhood size in the simulated annealing algorithm*, American Journal of Mathematical and Management Sciences **8** (1988), 409–423.
- [HL88] K Ho-Le, *Finite element mesh generation methods: a review and classification*, Computer Aided Design **20** (1988), no. 1, 27–38.
- [IM76] Dean L. Isaacson and Richard W. Madsen, *Markov chains: Theory and applications*, Wiley, 1976.
- [Inc09] Graxx Inc, *Le système de repérage cie xyz*, <http://pages.videotron.com/graxx/CIE2.html>, 2009.
- [JSW07] Gerold Jäger, Anand Srivastav, and Katja Wolf, *Solving generalized maximum dispersion with linear programming*, AAIM Third International Conference Proceedings **4508** (2007), 1–10.
- [Ker10] Douglas A. Kerr, *The cie xyz and xyy color spaces*, Issue 1, March 2010.
- [KGV83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, *Optimization by simulated annealing*, Science **220** (1983), no. 4598, 671–680.
- [Kin92] Rex K. Kincaid, *Good solutions to discrete noxious location problems via metaheuristics*, Annals of Operations Research **40** (1992), 265–281.
- [KKD11] Simon Kulovec, Leon Kos, and Jozef Duhovnik, *Mesh smoothing with global optimization under constraints*, Journal of Mechanical Engineering **57** (2011), 555–567.
- [Kuh60] H. W. Kuhn, *Some combinatorial lemmas in topology*, IBM Journal of Research and Development **45** (1960), 518–524.
- [Lin93] Feng-Tse Lin, *Applying the genetic approach to simulated annealing in solving some np-hard problems*, IEEE Transactions on Systems Man and Cybernetics **23** (1993), no. 6, 1752–1767.
- [LM86] M. Lundy and A. Mees, *Convergence of an annealing algorithm*, Mathematical Programming **34** (1986), no. 1, 111–124.
- [Low13] Richard Lowry, *Concepts & applications of inferential statistics. chapter 12*, <http://vassarstats.net/textbook/>, 2013.
- [MGDP13] Rafael Martí, Micael Gallego, Abraham Duarte, and Eduardo G. Pardo, *Heuristic and metaheuristics for the maximum diversity problem*, Journal of Heuristics **19** (2013), no. 4, 591–615.

- [Moo92] Douglas William Moore, *Simplicial mesh generation with applications*, Ph.D. thesis, Cornell University, 1992.
- [Net] Netdisseny, *Nociones básicas de diseño. teoría del color*, <http://repositori.cuaed.unam.mx:8080/jspui/bitstream/123456789/1901/1/teoria-del-color.pdf>.
- [Nic11] André Nicolas, *Largest triangle with vertices in the unit cube*, <http://math.stackexchange.com/questions/44396/largest-triangle-with-vertices-in-the-unit-cube>, June 2011.
- [OAS⁺00] Okabe, Atsuyuki, Sugihara, Kokichi, and Nok Chiu, *Spatial tessellations: Concepts and applications of voronoi diagrams*, Chichester: John Wiley & Sons Ltd., 2000.
- [OL96] I.H. Osman and G. Laporte, *Metaheuristics: A bibliography*, Ann. Oper. Res. **63** (1996), 513–623.
- [Poy95] Charles Poynton, *A guided tour of color space*, Proceedings of the SMPTE Advanced Television an Electronic Imaging Conference (1995), 167–180.
- [Poy06] ———, *Color faq - frequently asked questions color*, November 2006.
- [PPR⁺03] Joaquín Pérez, Rodolfo A. Pazos, David Romero, René Santaolaya, Guillermo Rodriguez, and Victor Sosa, *Adaptive and scalable allocation of data objects in the web*, Lecture Notes in Computer Science **2667** (2003), 134–143.
- [PS82] C.H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*, Prentice-Hall, New York, 1982.
- [RMGD10] Mauricio G.C. Resende, Rafael Martí, Micael Gallego, and Abraham Duarte, *Grasp and path relinking for the max-min diversity problem*, Computers & Operations Research **37** (2010), no. 3, 498–508.
- [RR07] Mauricio G.C. Resende and Celso C. Ribeiro, *Grasp: Greedy randomized adaptive search procedures*, Tech. report, AT&T Labs, 2007.
- [Sel13] Jeremy Selan, *Visualizing the cie xyz color space*, <http://www.pixelsham.com/2013/04/06/visualizing-the-cie-xyz-color-space-by-jeremy-selan/>, April 2013.
- [SMJ93] Andrew Stockman, Donald I. A. McLeod, and Nancy E. Johnson, *Spectral sensitivities of the human cones*, Journal of the Optical Society of America A, **10** (1993), no. 12, 2491–2521.

- [SP06] Volker Schmidt and Ursa Pantle, *Markov chains and monte-carlo simulation. chapter: Irreducible and aperiodic markov chains*, July 2006, Lecture Notes.
- [SP09] Joerg Steinruecken and Lutz Pluemer, *A web service to personalise map colouring*, Institute for Geodesy and Geoinformation University of Bonn, 2009.
- [Sta95] Frank L. Stasa, *Applied finite element analysis for engineers*, Hrw Series in Mechanical Engineering, Oxford University Press, 1995.
- [Ste09] Jörg Steinrücken, *Automatisierte erzeugung personalisierter ad-hoc-karten in einem service-basierten gis (mapping on demand)*, Ph.D. thesis, Rheinischen Friedrich-Wilhelms-Universität, 2009.
- [VMOR99] S. VoB, S. Martello, I.H. Osman, and C. Roucairol (eds.), *Meta-heuristics. advances and trends in local search paradigms for optimization*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.
- [Wes12] Stephen Westland, *Colour physics frequently asked questions*, Magus Publishing, 2012.
- [Wik14a] Wikipedia, *Cie 193 color space*, 2014.
- [Wik14b] _____, *Friedman test*, March 2014.
- [Wik14c] _____, *Wilcoxon signed-rank test*, April 2014.
- [Wil45] Frank Wilcoxon, *Individual comparison by ranking methods*, Biometrics Bulletin **1** (1945), no. 6, 80–83.
- [Win05] Peter Winker, *The stochastics of threshold accepting: Analysis of an application to the uniform design problem*, Discussion paper / Universität Erfurt Staatswissenschaftliche Fakultät **003E** (2005), 1–17.
- [Wri29] William David Wright, *A re-determination of the trichromatic coefficients of the spectral colours*, Transactions of the Optical Society **30** (1929), no. 4, 141–164.
- [YWLL11] Dong-Ming Yan, Wenping Wang, Bruno Levy, and Yang Liu, *Efficient computation of clipped voronoi diagram for mesh generation*, Preprint submitted to Computer-Aided Design, October 2011.