

# Inteligencia Computacional 2025

## Desarrollo del cursado

- Clases de teoría: en estas clases se introducen los conceptos teóricos de cada unidad temática. La gran mayoría están grabadas en video y se encuentran disponibles online. Los videos que corresponden a cada semana deben verse **antes** de las clases de coloquio y prácticas. En todos los casos, estas introducciones deberán complementarse con las restantes actividades, las guías de estudio y la bibliografía recomendada. Este enfoque busca promover una participación más activa en la propia formación, aumentando la independencia y fortaleciendo las capacidades para el autoaprendizaje.
- Clases de coloquio: tienen como objetivo profundizar los temas desarrollados en las clases de teoría y trabajar sobre las dificultades e inquietudes conceptuales generadas a partir de las actividades prácticas. Se desarrollan con una modalidad de trabajo más flexible y participativa, de forma que el alumnado plantee abiertamente sus dudas y se discuta con la guía de la cátedra. Las temáticas están relacionadas con los objetivos específicos y sus aplicaciones, pero también están orientadas a tratar muchos de los aspectos mencionados en los objetivos generales.
- Clases de práctica: la práctica de laboratorio consiste en la formulación de soluciones e implementación de diferentes técnicas de inteligencia computacional. En estas clases se explican las características básicas del trabajo a realizar, asociado al tema desarrollado previamente en la clase teórica. Además, se trabaja con cada grupo individualmente, se realizan las evaluaciones correspondientes y se brinda una realimentación para ayudar a relacionar y fijar los conceptos teóricos. Se brinda orientación y apoyo durante la clase pero las actividades deben desarrollarse en forma autónoma.
- Consultas: se dispondrá de horarios semanales para evacuar todo tipo de dudas, tanto para cuestiones técnicas de la asignatura como administrativas del cursado. El día y la hora de cada clase de consulta serán acordados al comenzar el cursado.

Se recomienda prestar especial atención a los horarios de cursado y consultas, y asistir con puntualidad. Todas las evaluaciones se realizarán al comienzo del horario de clases: 12:00 hs.

Es necesario matricularse en la plataforma educativa [e-fich](#), desde la cual se podrán realizar consultas breves y recibir toda la información sobre el cursado. Los videos de las clases pueden verse en el [repositorio](#) y el [canal de la asignatura](#).

## Aprobación de la asignatura

### Requisitos regularización

#### Evaluación de práctica

Tiene por objetivo evaluar tanto los avances en la realización de trabajos prácticos como la incorporación de los conocimientos teóricos básicos relacionados. Esta instancia de evaluación es una de las más importantes del cursado ya que brinda la oportunidad de evaluar de cerca la evolución y sobre todo poder ajustar a tiempo los procesos de enseñanza y aprendizaje de acuerdo a las necesidades particulares individuales o del grupo de trabajo, recomendando lecturas adicionales, sugiriendo ampliaciones de algún ejercicio práctico,

explicando en detalle algún tema en particular, etc. La evaluación se extiende entre 15 a 30 minutos.

La evaluación de práctica será oral y grupal: defensa del trabajo práctico del tema correspondiente, mostrando todo el código fuente y todos los ejercicios resueltos. Durante esta defensa se evalúan tanto los conocimientos prácticos como los teóricos. El grupo completo debe estar presente al momento de comenzar la evaluación. Si bien la evaluación es grupal, cada integrante del grupo debe estar en condiciones de responder correctamente a las preguntas. Dependiendo de la cantidad de grupos, en esta instancia de evaluación/seguimiento suelen participar todos los integrantes de la cátedra.

Los grupos de trabajo deberán estar conformados por 2 o 3 integrantes. No se aceptarán trabajos individuales.

La codificación de todos los algoritmos desarrollados, sin la utilización de programas, bibliotecas o funciones previamente desarrolladas por terceros, es una característica importante del enfoque pedagógico que damos a la asignatura. Consideramos que el entendimiento de todos los algoritmos se completa al programarlos desde cero.

Vale aclarar que el aprendizaje de una forma de implementación particular, lenguaje o entorno de programación no es parte de los objetivos de la asignatura. Por lo tanto, los aspectos técnicos de la compilación y utilización de lenguajes y entornos de programación quedan bajo la responsabilidad de cada grupo de trabajo.

Al comenzar la clase en la que se debe entregar cada trabajo práctico, todos los programas deben funcionar correctamente y cada integrante del grupo debe ser capaz de explicar y justificar cada paso de la solución presentada.

## Evaluación parcial

Se proveen dos exámenes parciales, involucrando aproximadamente la mitad de los temas del programa en cada uno (según se detalla en el cronograma). Estas evaluaciones tienen especial énfasis en los aspectos teóricos.

En cada parcial se realizará una primera instancia de evaluación con preguntas sencillas en la modalidad de opciones múltiples. Si la calificación en esta instancia es de 40 o más, los puntos se pueden acumular junto a las demás evaluaciones para aspirar a la regularidad (ver sección Puntuación para regularizar). En el caso de la promoción, los puntos a acumular deben obtenerse en un examen oral. Para acceder al examen oral se requiere una calificación de al menos 80 en el de opciones múltiples. Vale aclarar que esta última calificación no se utiliza para la puntuación con la que se accede a la promoción (ver sección Requisitos promoción).

## Trabajo creativo integrador

Se debe realizar en grupos conformados de igual forma que para los trabajos prácticos. Este trabajo consiste en resolver un problema que debe proponer cada grupo. Deben ser problemas prácticos a resolver y desde la Cátedra se guiará al grupo para seleccionar y delimitar una de las propuestas y el restante desarrollo del trabajo. En base a una búsqueda bibliográfica de antecedentes relacionados se debe proponer e implementar una solución válida, aplicando las herramientas computacionales que se estudian durante el cursado. No se aceptarán trabajos individuales ni problemas que no hayan sido seleccionados junto a la cátedra.

Para facilitar el aprovechamiento de esta instancia se deberán cumplimentar 4 presentaciones en total (3 parciales y una final) y la calificación se definirá considerando todas las instancias de evaluación:

- 1ra presentación: cada grupo llevará al menos 3 propuestas de las cuales, con la guía de la cátedra, se elegirá una. Las propuestas se entregan por escrito (título y 200 palabras por cada idea).
- 2da presentación: entrega de una búsqueda bibliográfica y propuesta de solución (ambas también por escrito). La búsqueda bibliográfica deberá ser de una página, con las referencias en el formato correspondiente, y la propuesta de solución en aproximadamente 400 palabras. También se deberán adjuntar los artículos de las referencias para poder analizarlos juntos con la cátedra.
- 3ra presentación: implementación (código fuente) con todo el problema resuelto y funcionando correctamente al momento de ser entregado. Se deberá realizar una defensa oral de todo lo entregado.

La presentación final consiste en un informe escrito y una exposición oral de 15 minutos, con defensa de 5 minutos. Se puede descargar una guía de estilo y estructura para preparar el informe final desde el repositorio de la asignatura. Otros detalles del informe y la presentación se especificarán durante el cursado.

## Puntuación para regularizar

Para regularizar la asignatura se consideran todas las instancias de evaluación durante el cursado, con la siguiente distribución relativa de puntos:

- Evaluaciones de práctica: 40 puntos (distribuidos por tema).
- Evaluaciones parciales: 40 puntos (distribuidos por parcial).
- Trabajo creativo: 20 puntos

Para regularizar la asignatura (lo que implica la promoción de práctica) se requieren 60 puntos o más en la acumulación de todas las evaluaciones durante el cursado.

Aclaraciones importantes:

- Para regularizar no es válida la acumulación de puntos si en alguna de las instancias de evaluación se obtuviera menos del 40% de la puntuación.
- Los trabajos creativos deberán aprobarse con al menos 60% de la puntuación.

## Exámenes de recuperación

Se proveen dos instancias de recuperación de exámenes:

- Práctica: se podrán recuperar 2 evaluaciones en caso de no alcanzar el 40% en alguna de ellas.
- Parciales: se podrán recuperar los 2 exámenes para aumentar la calificación.

Los exámenes de recuperación serán individuales aunque la modalidad (escrito/oral) será dispuesta por el responsable de la asignatura independientemente de aquella con que se hubiese evaluado originalmente el tema. La calificación final es el máximo entre el examen original y el examen de recuperación.

## Requisitos promoción

Con las mismas consideraciones detalladas en los Requerimientos para Regularizar, para el caso de la Promoción (completa) se requieren 70 puntos o más en las evaluaciones durante el cursado. En este caso no es válida la acumulación de puntos si en alguna de las instancias de evaluación se obtuviera menos del 60% de la puntuación.

# Examen final

## Modalidad estudiantes regulares

Evaluación de teoría: el examen para alumnos regulares es oral, de entre 15 y 30 minutos por alumno. Se evalúan tres temas para los que el alumno puede realizar un desarrollo preliminar en pizarra o papel, y luego explicar oralmente, respondiendo a las preguntas y realizando las ampliaciones que se le soliciten.

Aclaración importante: si una vez finalizado el cursado no figura en las Actas de Cursado, su situación final es equivalente a no haber cursado la asignatura.

## Modalidad estudiantes libres

Trabajo creativo: equivalente a la que se realiza durante el cursado pero en este caso su realización no podrá ser grupal. Informar al responsable de la asignatura 15 días antes de la fecha en que desea rendir.

Práctica: a libro abierto y con computadora. Se formulan problemas que involucran varios temas de la asignatura. Se puede utilizar código fuente desarrollado previamente pero deberá defender adecuadamente cada parte de la implementación.

Teoría: igual que para regulares.

# Objetivos de la asignatura

## Generales

Que:

- adquiera una nueva visión de la computación, desde la perspectiva de la inteligencia computacional,
- entienda los principios en que se basan muchas de las tecnologías con las que tiene un contacto permanente,
- incremente sus capacidades para el trabajo en equipo y la distribución de tareas y responsabilidades,
- incremente sus destrezas para la transmisión oral y escrita de conocimientos científicos y tecnológicos,
- desarrolle su capacidad de análisis aplicando diversas estrategias para resolución de problemas,
- incremente sus destrezas para aprender de forma independiente,
- realice trabajos experimentales que reflejen situaciones reales típicas,
- establezca contacto con publicaciones de nivel científico, pudiendo analizarlas, reproducirlas parcialmente y criticarlas,
- desarrolle su creatividad en la propuesta de nuevas técnicas o aplicaciones y mejoras de las ya conocidas,
- utilice correctamente la terminología técnica del área,
- aplique e incremente sus conocimientos de inglés técnico.

Además, entre otros objetivos de formación general, se espera que:

- valore la discusión abierta como una fuente de generación y difusión de conocimientos,
- valore los medios que la Universidad pone a su disposición y desarrolle sentimientos positivos hacia ella,
- se involucre más intensamente con la vida universitaria,
- conozca los valores y principios que sustentan a las instituciones académicas,
- se introduzca al pensamiento científico y tecnológico,
- se interese por continuar su formación mediante estudios de postgrado.

## Específicos

Que:

- obtenga conocimientos generales acerca del área de la inteligencia computacional y una idea de su magnitud y diversidad,
- conozca las arquitecturas neuronales más utilizadas,
- comprenda las características dinámicas de redes recurrentes,
- aprenda los algoritmos de entrenamiento,
- implemente distintas arquitecturas neuronales y algoritmos básicos de entrenamiento,
- adquiera conocimientos sobre modelos para representación e inferencia de conocimiento,
- aprenda y utilice la teoría de conjuntos borrosos,
- aprenda los métodos de computación evolutiva y el diseño de la solución de problemas mediante éstos,
- implemente los algoritmos básicos de computación evolutiva y los aplique a problemas sencillos,
- aprenda e implemente métodos de inteligencia de poblaciones,
- adquiera independencia en el diseño de la solución de problemas mediante las técnicas tratadas,
- adquiera destrezas para aplicar con criterio y visión práctica las metodologías estudiadas,
- conozca diversas aplicaciones directas de la inteligencia computacional.

## Deshonestidad académica

En el caso de que se incurra en cualquier acto de deshonestidad académica, quedará automáticamente **LIBRE** sin importar la condición previa en la asignatura. Además se elevará un pedido a la Secretaría Académica para que sea sancionado de acuerdo al caso. Se considerarán actos de deshonestidad académica: copiar exámenes (de cualquier tipo y en cualquier forma), copiar informes, copiar programas o ideas originales para la resolución de problemas, etc. Esto no sólo se refiere al cursado actual sino que también a trabajos de cursados anteriores u otras fuentes. Todo trabajo presentado se considera que ha sido el resultado de su proceso de aprendizaje y fruto de su propio esfuerzo. Como regla general, en un caso de copia entre quienes están cursando son culpables ambas partes, por lo que se sugiere cuidar sus informes, códigos fuente o cualquier otro objeto de una evaluación. Se considera también un acto deshonesto utilizar material (texto, figuras, etc.) de otras fuentes (Internet, libros, revistas, etc.) sin realizar la cita correspondiente. No es aceptable la copia textual en ningún caso, desde ninguna fuente (libros, artículos, páginas web, etc.). Sólo se

puede transcribir una frase textual entre comillas y en itálica, y citando la fuente. Como es natural, no es posible enumerar todos los casos de deshonestidad académica por lo que la lista anterior no es exhaustiva y otros casos serán analizados oportunamente. En el caso de tener cualquier duda al respecto, siempre se debe consultar a la cátedra **antes de hacerlo**.

## Bibliografía

### Básica general

Título: Computational Intelligence: An Introduction

Autores: A. P. Engelbrecht

Título: Genetic Algorithms + Data Structures = Evolution Programs

Autores: Z. Michalewicz

Título: Genetic Algorithms in Search, Optimization and Machine Learning

Autores: D. Goldberg

Título: Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence

Autores: B. Kosko

Título: Neural Networks and Learning Machines

Autores: S. Haykin

Título: Swarm intelligence

Autores: J. Kennedy, R.C. Eberhart

### Básica por unidad temática

#### Introducción general. Perceptrón simple.

S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Ed.

1.1-1.4, 1.6 (opcional: 1.5, 1.7-1.9)

2.1-2.5, 2.8-2.10 (opcional: 2.6, 2.7, 2.8, 2.11-2.16)

3.1, 3.3, 3.5, 3.8

J. Freeman, D. Skapura, Neural Networks: Algorithms, Applications, and Programming Techniques

1.1, 1.2 (opcional: 1.3)

2.2 (opcional: 2.1, 2.4, 2.5)

A. Jain, J. Mao, K. Mohiuddin, Artificial Neural Networks: A Tutorial

pp 31-39 (backpropagation no)

R. Lippmann, An Introduction to Computing with Neural Nets

pp 4-7 (Hopfield no)

pp 13, 14

B. Widrow, M.A. Lehr, 30 years of adaptive neural networks: perceptron, Madaline, and backpropagation  
pp 1415-1420  
pp 1423-1432

## Reconocimiento de patrones

R. Duda, P. Hart, D. Stork, Pattern Classification, Second Edition, Wiley-Interscience, 2000  
1, 2.1-2.6, 3.1, 3.2

## Perceptrón multicapa

S. Haykin, Neural Networks: A Comprehensive Foundation  
4.1-4.6 (opcional: 6.7)  
4.12, 4.14 (opcional: 4.16)

J. Freeman, D. Skapura, Neural Networks: Algorithms, Applications, and Programming Techniques  
3.1-3.3

R. Lippmann, An Introduction to Computing with Neural Nets  
pp 16-18

B. Widrow, M.A. Lehr, 30 years of adaptive neural networks: perceptron, Madaline, and backpropagation  
pp 1420-1421  
pp 1432-1435

## Redes con funciones de base radial

C. Bishop, Neural Networks for Pattern Recognition  
5.1-5.3, 5.7-5.10  
S. Haykin, Neural Networks: A Comprehensive Foundation  
5.1, 5.7, 5.8, 5.11

## Mapas auto-organizativos

S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Ed.  
9.1-9.4 (opcional: 9.6)  
9.7

J. Freeman, D. Skapura, Neural Networks: Algorithms, Applications, and Programming Techniques  
7.1 (opcional: 7.3)

T. Kohonen, Self-organizing maps, 3rd. Ed.  
3.1, 3.2, 3.7-3.9, 3.13 (opcional: 3.4, 3.14)  
1.5.1, 6.1-6.3 (opcional: 6.9)

R. Lippmann, An Introduction to Computing with Neural Nets  
pp 19-21

## Redes dinámicas

S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Ed.

13.3, 13.4

14.7 (pp 687-696) (opcional: 14.8)

15.1, 15.2, 15.6, 15.7

D. R. Hush, B. G. Horne, Progress in Supervised Neural Networks  
pp 28-34 (sin versiones continuas)

R. Lippmann, An Introduction to Computing with Neural Nets  
pp 8-10

## Lógica borrosa. Memorias asociativas borrosas

B. Kosko, Neural networks and fuzzy systems

(opcional: 1)

7

Fuzzy expert systems and Fuzzy reasoning, Willam Siler y James Buckley  
(opcionales 3, 4 y 5)

Fuzzy logic systems for engineering: a tutorial, Jerry Mendel, Proceedings of the IEEE , vol 83  
no 3  
(opcional)

Soft computing and fuzzy logic, Lofti Zadeh, IEEE Software, vol 11, no 6, pp 48-56  
(opcional)

B. Kosko, Neural networks and fuzzy systems  
8

## Introducción a Inteligencia Colectiva

A. P. Engelbrecht, Computational Intelligence: An Introduction, 2nd Ed.

16.1, 16.2, 16.4 (opcional: 16.7)

17.1.1-17.1.4, 17.1.12 (opcional: 17.5)

M. Clerc, Particle swarm optimization,  
(opcional 3)

M. Dorigo, T. Stutzle, Ant Colony Optimization,  
(opcional 2)

J. Kennedy, R.C. Eberhart, Swarm intelligence,  
(opcionales 1,7)



E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: from natural to artificial intelligence,  
(opcionales 1,2)

## Computación evolutiva

Apunte de la Cátedra (material con introducción teórica y ejercicios)

D. E. Goldberg, Genetic Algorithms in Search, Optimization & Machine Learning  
1-4

J. R. Koza, Genetic Programming  
5, 6

Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs,  
(opcionales 1-5, 9)

M. Mitchell, An Introduction to Genetic Algorithms,  
(opcionales 1, 2)

T. Back, D. B. Fogel, Z. Michalewicz, Evolutionary Computation 1: Basic Algorithms and Operators,  
(opcionales 1-4, 8-10)

R. L. Haupt, S. E. Haupt, Practical Genetic Algorithms,  
(opcionales 1, 2)

T. Back, U. Hammel, H-F. Schewfel, Evolutionary Computation: Comments on History and Current State  
(opcional)

## Colonia de hormigas y enjambre de partículas

A. P. Engelbrecht, Computational Intelligence: An Introduction, 2nd Ed.  
16.1, 16.2, 16.4 (opcional: 16.7)  
17.1.1-17.1.4, 17.1.12 (opcional: 17.5)

M. Clerc, Particle swarm optimization,  
(opcional 3)

M. Dorigo, T. Stutzle, Ant Colony Optimization,  
(opcional 2)

J. Kennedy, R.C. Eberhart, Swarm intelligence,  
(opcionales 1,7)

E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: from natural to artificial intelligence,

(opcionales 1,2)

## Complementaria

(b) libros disponibles en la Biblioteca de la Facultad

(i) libros disponibles en el Instituto de Investigación en Señales e Inteligencia Computacional

## Redes neuronales

(i) S. Haykin, Neural Networks and Learning Machines, Prentice Hall, 2008.

M. M. Gupta, L. Jin, N. Homma, Static and Dynamic Neural Networks From Fundamentals to Advanced Theory, John Wiley & Sons, 2003.

(i) D. P. Mandic, J. Chambers, Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability, John Wiley & Sons, 2001.

(i) C. Bishop, Neural Networks for Pattern Recognition, Oxford, 1999. R. M. Hristev, The ANN Book, 1998.

(i) B. D. Ripley, Pattern Recognition and Neural Networks, Cambridge Univ. Press, 1996.

(i) T. Masters, Neural, Novel & Hybrid Algorithms for Time Series Prediction, J. Wiley & Sons, 1995.

(i) T. Kohonen, Self Organizing Maps, Springer Verlag, 1995.

(ib) S. Haykin, Neural Networks: A Comprehensive Foundation, Macmillan College Publishing Company, 1994.

(i) J. Freeman, Simulating Neural Networks, Addison Wesley, 1994.

(i) J. Freeman y D. Skapura, Redes neuronales: algoritmos, aplicaciones y técnicas de programación, Addison Wesley, 1993.

C. Lau, Neural Networks: Theoretical Foundations and Analysis, IEEE Press, 1992

## Lógica borrosa

(i) W. Siler, J. J. Buckley, Fuzzy Expert Systems and Fuzzy Reasoning, John Wiley & Sons, 2005.

(i) T. Ross, Fuzzy Logic with Engineering Applications, Wiley, 2004.

(i) K. Tanaka, H. O. Wang, Fuzzy Control Systems Design and Analysis, John Wiley & Sons, 2001.

(i) O. Wolkenhauer, Data Engineering: Fuzzy Mathematics in Systems Theory and Data Analysis, John Wiley & Sons, 2001.

(i) B. Kosko, Fuzzy Engineering, Prentice Hall, 1997.

(i) J-S. R. Jang, C-T. Sun, E. Mizutani, Neuro-Fuzzy And Soft Computing: A Computational Approach To Learning And Machine Intelligence, Prentice Hall, 1997.

(i) F. M. McNeill, E. Thro, Fuzzy Logic, A Practical Approach, Academic Press, 1994.

(i) B. Kosko, Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence, Prentice Hall, 1992.

L. A. Zadeh y J. Kacprzyk, Fuzzy Logic for the Management of Uncertainty, John Wiley, 1992.

J. C. Bezdek y S. K. Sankar, Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data, IEEE Press, 1992.

## Inteligencia Colectiva

S. Sumathi, T. Hamsapriya, P. Surekha, Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab, Springer, 2008.

C. A. Coello Coello, G. B. Lamont, A. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems Series: Genetic and Evolutionary Computation, (2da Edición), Springer, 2007.

M. Clerc, Particle swarm optimization, ISTE, 2006.

(i) R. L. Haupt, S. E. Haupt, Practical Genetic Algorithms, John Wiley & Sons, 2004.

M. Dorigo, T. Stützle, Ant Colony Optimization, MIT Press, 2004.

(i) A. Menon, Frontiers of Evolutionary Computation, Kluwer Academic Publishers, 2004.

J. Kennedy, R.C. Eberhart, Swarm intelligence, Academic Press, 2001

L. D. Chambers, The Practical Handbook of Genetic Algorithms, Vols. I, II & III, CRC Press, 2000.

(i) T. Back, D. B. Fogel, Z. Michalewicz, Evolutionary Computation 1: Basic Algorithms and Operators, IOP Publishing Ltd, 2000.

(i) T. Back, D. B. Fogel, Z. Michalewicz, Evolutionary Computation 2: Advanced Algorithms and Operators, IOP Publishing Ltd, 2000.

E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: from natural to artificial intelligence, Oxford University Press, 1999.

(i) M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, 1999.

(i) T. Back, D. B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, IOP Publishing Ltd and Oxford University Press, 1997.

S. Pal y P. Wang, Genetic Algorithm for Pattern Recognition, CRC Press, 1996

(i) Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer Verlag, 1992.

L. Davis (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, 1991.

(i) D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, 1989.

(i) S.G. de los Cobos Silva, J. Goddard Close, M.A. Gutiérrez Andrade, A.E. Martínez Licona, Búsqueda y exploración estocástica, Universidad Autónoma Metropolitana, 2010.