FACULTAD DE INGENIERÍA UDELAR

UNIVERSIDAD DE LA REPUBLICA URUGUAY

Despacho Hidrotérmico Óptimo con Técnicas de Aprendizaje por Refuerzos

Bruno Olivera

Programa de Posgrado en Ciencia de Datos
Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Abril de 2024

FACULTAD DE
INGENIERÍA
UDELAR

UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Despacho Hidrotérmico Óptimo con Técnicas de Aprendizaje por Refuerzos

Bruno Olivera

Tesis de Maestría presentada al Programa de Posgrado en Ciencia de Datos, Facultad de Ingeniería de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Magíster en Ciencia de Datos Aplicada.

Director:
Dr. Ing. Claudio Risso

Codirector:
Dr. Ing. Pablo Rodríguez-Bocca

Director académico:
Dr. Ing. Pablo Rodríguez-Bocca

Montevideo – Uruguay
Abril de 2024

Olivera, Bruno

Despacho Hidrotérmico Óptimo con Técnicas de Aprendizaje por Refuerzos / Bruno Olivera. - Montevideo: Universidad de la República, Facultad de Ingeniería, 2024.

XIII, 139 p.: il.; 29, 7cm.

Director:

Claudio Risso

Codirector:

Pablo Rodríguez-Bocca

Director académico:

Pablo Rodríguez-Bocca

Tesis de Maestría – Universidad de la República, Programa en Ciencia de Datos, 2024.

Referencias bibliográficas: p. 135 – 139.

1. Aprendizaje por Refuerzos, 2. Programación Dinámica, 3. Despacho Hidrotérmico, 4. Optimización Estocástica. I. Risso, Claudio, *et al.* II. Universidad de la República, Programa de Posgrado en Ciencia de Datos. III. Título.

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE TESIS

Dr. Ing. Alberto Pardo (UdelaR)

Msc. Ing. Alfredo Piria (URSEA)

Dr. Ing. Ignacio Ramírez (UdelaR)

Montevideo – Uruguay
Abril de 2024

Agradecimientos

En primer lugar, quisiera agradecer especialmente a mi familia, por su apoyo incondicional a lo largo de los años, siendo un pilar fundamental en mi formación como persona y profesional.

En segundo lugar, a mis directores de Tesis, Claudio Risso y Pablo Rodríguez-Bocca, por su excelente disposición y paciencia durante todo el largo proceso de desarrollo de esta Tesis, sin los cuales no me hubiera sido posible finalizarla.

En tercer lugar, quisiera agradecer a Graciela Ferreira y Rodrigo Porteiro, por sus comentarios y apoyo en la reconstrucción de los antecedentes históricos.

Por último, a los Sres. miembros del tribunal: Dr. Ing. Alberto Pardo, Msc. Ing. Alfredo Piria, y Dr. Ing. Ignacio Ramírez quienes se tomaron el tiempo de analizar mi trabajo y realizar la evaluación correspondiente. En especial al Dr. Ing. Ignacio Ramírez por las tareas realizadas en su rol como revisor.

RESUMEN

Toda empresa de energía eléctrica, que cuente con generadores térmicos y generadores hidroeléctricos, se enfrenta al problema del Despacho Hidrotérmico Óptimo. El mismo consiste en determinar una política de operación óptima de sus generadores de forma que se minimicen los costos a la hora de satisfacer la demanda energética en un determinado período de tiempo.

Históricamente se suele usar técnicas derivadas de formulaciones analíticas para resolver este problema, como por ejemplo la Programación Dinámica o Programación Dinámica Estocástica (debido a la naturaleza estocástica del problema) entre otras. Estas técnicas suelen enfrentarse a la denominada "maldición de la dimensionalidad", donde el espacio de estados sobre el que se trabaja tiende a crecer demasiado rápido, volviendo al problema intratable en términos computacionales. Para superar esa contrariedad y aplicar estas técnicas, por lo general, se opta por modificar o simplificar el problema original.

Este trabajo investiga la viabilidad de resolver el problema de Despacho Hidrotérmico Óptimo utilizando técnicas de *Reinforcement Learning*, una rama de *Machine Learning* que ha mostrado sobreponerse al problema de la "Maldición de la Dimensionalidad" en otros tipos de problemas.

El trabajo está estructurado en dos iteraciones, en cada iteración se detalla una instancia del problema (con alguna complejidad extra sobre la iteración anterior), y luego se procede a resolverlo por técnicas tradicionales, y por *Reinforcement Learning*, para luego analizar y comparar los resultados. *Reinforcement Learning* parece ser una alternativa prometedora para resolver el problema de Despacho Hidrotérmico Óptimo, ya que en muchos de los experimentos realizados se obtienen resultados similares, o hasta mejores, que las técnicas tradicionales, y ejecutándose de forma muy rápida.

Palabras claves:
Aprendizaje por Refuerzos, Programación Dinámica, Despacho Hidrotérmico, Optimización Estocástica.

ABSTRACT

Any power company with thermal generators and hydroelectric generators faces the Optimal Hydrothermal Dispatch problem. This problem consists of determining an optimal operational policy for its generators so that certain energy demands can be met over a certain period of time while minimizing costs.

Traditionally, techniques derived from analytical formulations are usually used to solve this problem, namely Dynamic Programming or Stochastic Dynamic Programming (due to the stochastic nature of the problem) among others. These techniques tend to stumble upon what is known as the "Curse of Dimensionality". In simple terms, the state space they work on tends to grow so much, that this approach quickly becomes computationally intractable.

This work explores the viability of solving the Optimal Hydrothermal Dispatch problem, using a branch of Machine Learning known as Reinforcement Learning, with the hopes of overcoming the "Curse of Dimensionality".

The work is structured as consecutive iterations, each of which defines an instance of the problem (every iteration adds some new complexity over the previous ones), and then proceeds to solve it using analytical techniques and using Reinforcement Learning, in order to analyze and compare results. Reinforcement Learning seems to be a promising alternative to solve the Optimal Hydrothermal Dispatch problem, seeing that many of the results obtained are similar or even better in some cases than the traditional techniques' results, all the while being very fast to execute.

Keywords:
Reinforcement Learning, Dynamic Programming, Hidrothermal Dispatch, Stochastic Optimization.

Lista de figuras

Lista de tablas

Tabla de contenidos

Capítulo 1

Introducción

En los sistemas de generación eléctrica se cuenta con múltiples fuentes de distintas características. Ejemplos son: generadores hidráulicos, térmicos, eólicos o solares. Esos recursos pueden clasificarse según algunos de sus atributos fundamentales. Una dimensión se orienta por su amigabilidad con el ambiente, donde las energías se distinguen entre renovables (e.g. eólica, solar, hidráulica o biomasa) y las no-renovables (térmicas a *fueloil* o gas). Esta clasificación no es importante a efectos de este estudio.

Otro ejemplo es la diferencia entre recursos despachables y no-despachables. En los no-despachables, no se cuenta con control sobre la generación, por ejemplo: solares/eólicos. Éstos generarán energía si están presentes los recursos necesarios (sol/viento) y no en caso contrario. En los despachables en cambio, se tiene control sobre la generación, esto es, sobre cuándo activarlos y a qué nivel de potencia. Tal es el caso de las represas hidroeléctricas y la mayoría del parque térmico, aunque en el segundo caso también hay excepciones, como los generadores de biomasa, que aun siendo térmicos no son controlados por el despacho central. Como veremos, esta clasificación es fundamental a efectos del presente estudio, donde precisamente se busca controlar la generación con el fin de minimizar los costos.

Finalmente, diferenciamos entre fuentes determinísticas y estocásticas. En el primer caso, los parámetros del generador se asumen conocidos durante el período de planificación. Un ejemplo de esto es el consumo por hora y el precio del combustible a usar para que una unidad térmica entregue cierto nivel de potencia. Ejemplos notorios de fuente estocástica lo constituyen las energías solar y eólica, ya que el recurso es altamente volátil en horizontes de planificación cortos, incluso de pocos días. Si bien esta incertidumbre es en la generación, al plantear el problema se traslada a la demanda como se explica en la próxima sección. Otro ejemplo de fuente estocástica lo constituyen los aportes hídricos que llegan a las centrales hidráulicas (precipitaciones, erogación de centrales aguas arriba, etc.). Mientras que los recursos eólicos y solares suelen ser altamente volátiles en el corto plazo, pero tienden a estabilizarse en el largo plazo, con los aportes hidráulicos ocurre exactamente lo contrario. En el corto plazo se suele tener buenas predicciones de cuánto va a llover por ejemplo, pero en el largo plazo es muy difícil de predecir y por tanto es altamente incierto. En el mediado/largo plazo, incluso el precio de los combustibles para los térmicos introduce incertidumbre, pero ese factor no se tiene en cuenta en el alcance de este trabajo.

Este estudio tiene por objetivo evaluar el uso de herramientas de aprendizaje automático como alternativa para el problema de planificación optimizada del despacho en un sistema eléctrico con incertidumbre.

1.1. Problema de Despacho Hidrotérmico Óptimo

Una forma simple de presentar este problema es reduciéndolo a la que era su forma previa a la introducción de las energías renovables no convencionales. Dado que las energías no-despachables deben ser recibidas por la red, podemos suponer que esa fuente de generación irá directamente a atender parte del consumo del sistema. Así, la fuente despachable será la responsable de atender la demanda residual, esto es: la diferencia entre la demanda real del sistema y la producción renovable no-convencional. De esta forma, la incertidumbre en la generación renovable no convencional (eólica y solar principalmente), se traslada a la demanda, la residual en este caso, que tiene alta varianza.

La simplificación anterior reduce el problema al control óptimo de las unidades de generación despachables, que en el sistema Uruguayo son las represas hidroeléctricas (a las que nos referiremos como fuentes hidráulicas de aquí en más) y las unidades térmicas. A efectos del despacho, quedan dos diferencias destacables entre ambas fuentes. El recurso necesario para los generadores térmicos es algún combustible (por ejemplo *fueloil*), lo cual tiene un costo directo asociado, mientras que los generadores hidráulicos utilizan como recurso el agua almacenada en su embalse, la cual en principio no tiene costo directo. Como simplificación adicional, se ignoran restricciones en el proceso de arranque de las unidades térmicas. Así, la segunda diferencia es que el uso de las unidades térmicas no afecta su estado, mientras que el uso del agua en los embalses reduce la altura del lago correspondiente, y por tanto, la cantidad de energía a recibir por m^3 turbinado. Adicionalmente, el agua usada en un período deja de estar disponible para uso futuro (i.e. tiene costo de oportunidad), lo que puede conducir a sobrecargar el parque térmico, y por tanto, aumentar el costo directo posterior. Por simplicidad, omitiremos de momento la dependencia entre erogados de una represa y aportes a otra ubicada aguas abajo en el mismo curso de agua. Esa situación se da en Uruguay entre las represas ubicadas sobre el Río Negro. Con el fin de tener una mejor perspectiva de la diversidad geográfica y tecnológica de las unidades de generación a planificar en el despacho real del sistema, se recomienda referirse a la Figura 1.1.

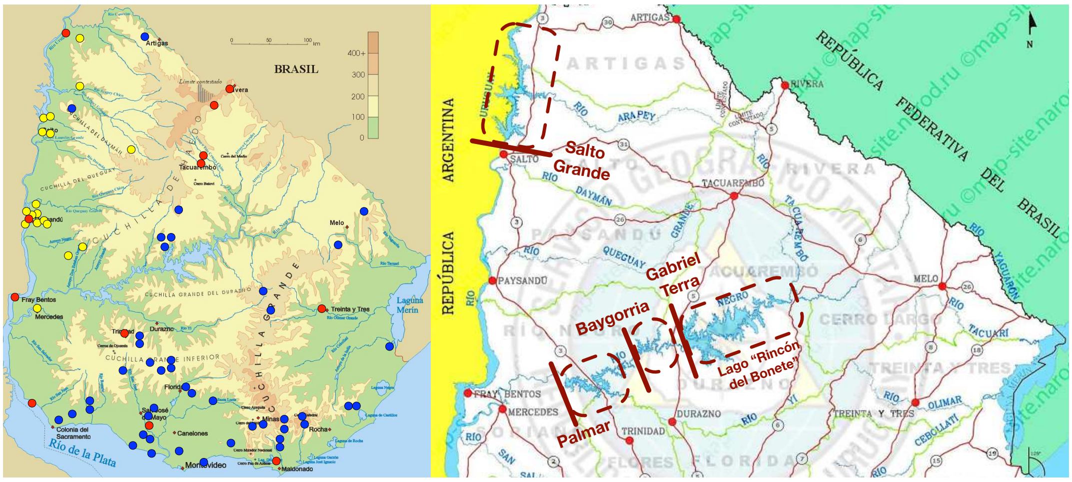


Figura 1.1: Unidades Eólicas (izquierda-azul), Fotovoltaicas (izquierda-amarillo), Biomasa (izquierda-rojo) e Hidroeléctrica (derecha). (Fuente UTE-ADME.)

Existe además un parque térmico de respaldo basado en tecnologías convencionales de combustibles fósiles, no destacado en las figuras. En este estudio, el modelo de referencia será de tiempo discreto. Los períodos de control son un conjunto finito de $N{+}1$ instantes t ($0 \leq t \leq T$) y N períodos sobre el horizonte de planificación $[0, T]$, digamos: $0 = t_0 < t_1 \ldots < t_N = T$. Las variables que intervienen en el funcionamiento de una unidad hidráulica h ($1 \leq h \leq H$) son:

S_{ht}: es la variable de estado asociada al volumen de agua almacenado en el embalse h al instante t.

A_{ht}: es la variable de control que representa el volumen de agua turbinado en el embalse h durante el período comprendido entre el instante t y el siguiente en la grilla. Los controles se asumen constantes en cada período.

J_{ht}: variable de control para el volumen de agua vertido en el embalse h entre el instante t y el siguiente. Los vertidos no generan energía, y de hecho, al elevar el nivel del agua posterior al salto del embalse, Sreducen la eficiencia del agua turbinada concurrentemente. Se consideran, por ejemplo, para no caer en situaciones donde la altura del lago exceda el límite de seguridad.

En general, la curva de producción de la unidad hidráulica h tiene una expresión como $EH_{ht} = f_h(S_{ht}, A_{ht}, J_{ht})$, siendo EH_{ht} la energía generada durante el período comprendido entre el instante t y el siguiente en la grilla, para los valores de las variables de esa unidad. La función f_h captura todas las características físicas, como ser: la geometría del lago, que determina su altura para S_{ht} dado; la de la represa, que determina el salto útil entre el lago y el nivel del río abajo, que depende a su vez de A_{ht} y J_{ht}; y la eficiencia de las turbinas.

Para capturar la dinámica de una represa hay que incorporar los aportes a los lagos, que afectan las variables de estado tanto como lo hacen las de control. Llamamos AP_{ht} a los aportes de agua al embalse h en el paso de tiempo que inicia en t. Al no considerar en este documento las aportes provenientes de los erogados de otras represas (e.g., Bonete a Baygorria y ésta a Palmar según Figura 1.1), AP_{ht} representa (de momento) los aportes por lluvias. El estado S_{ht} de una represa es afectado por el control y los aportes según una ecuación de balance de masa: $S_{h,t_{i+1}} = S_{h,t_i} + AP_{h,t_i} - A_{h,t_i} - J_{h,t_i}$. Por simplicidad, asumimos de aquí en más que los instantes están equi-espaciados e indexados, y usaremos $t+1$ para referirnos al instante posterior a t en la grilla de tiempo elegida.

Al no considerar sus *commitments* (i.e., los tiempos técnicos y costos asociados a la puesta en producción o apagado de las unidades), las unidades térmicas tienen un modelo de referencia simple, esto es, sin estado asociado. Podemos asumir entonces que la energía generada por una unidad fósil/térmica k ($1 \leq k \leq K$) en el intervalo $[t, t{+}1]$ es la variable de control, a saber ET_{kt}. La relación entre consumo y producción se asume conocida y fija. Representamos con $g_k(ET_{kt})$ al costo de combustible incurrido en la unidad k para generar la energía ET_{kt}, mediante una potencia sostenida a lo largo del período que inicia en t.

Optimalidad aparte, el problema se centra en atender la demanda d_t en todo momento t, que en este caso equivale a $d_t = \sum_{h=1}^{H} EH_{ht} + \sum_{k=1}^{K} ET_{kt}$. Una dificultad es que la demanda en sí misma es incierta. Se suma a esto que la demanda a atender en nuestro problema es la residual, la que es afectada a su vez por la incertidumbre en las generaciones eólica y solar. El planteo entonces pasa por expresar la demanda como una variable aleatoria d_t^λ, cuya distribución se asume conocida sobre un recorrido Λ_t. Asimismo, se cuenta con la incertidumbre en los aportes hídricos que llegan a las centrales hidráulicas; los aportes se expresan entonces como una variable aleatoria AP_t^{ϕ}, cuya distribución se asume conocida sobre un recorrido Φ_t.

El problema de despacho hidrotérmico óptimo (El-Hawary y Christensen, 1979) consiste en diseñar el control de las unidades térmicas e hidráulicas en un horizonte T a efectos de minimizar la suma de los costos en cada paso, sujeto a atender la demanda energética en todo momento. Al estar las variables vinculadas por la demanda y ser esta última aleatoria, las reformulamos según sus realizaciones. Nuestro problema de optimización de referencia busca alcanzar el menor valor esperado ($\mathbb{E}[\cdot]$) para el costo, al tiempo que asegura la demanda residual sobre el espacio de todas las realizaciones, esto es, para todo Λ_t , Φ_t, y t:

$$\begin{array}{ll} \min & \displaystyle\sum_{t=1}^{T} \mathbb{E}_{\Lambda_t}[\sum_{k=1}^{K} g_k(ET_{kt}^{\lambda})] \\ \text{s.t.} & \displaystyle\sum_{k=1}^{K} ET_{kt}^{\lambda} + \sum_{h=1}^{H} EH_{ht}^{\lambda} = d_t^{\lambda}, \quad \forall t, \lambda \in \Lambda_t, \\ & EH_{ht}^{\lambda} = f_h(S_{ht}^{\lambda}, A_{ht}^{\lambda}, J_{ht}^{\lambda}), \quad \forall h, t, \lambda \in \Lambda_t, \\ & S_{h,t+1}^{\lambda'} = S_{ht}^{\lambda} + AP_{ht}^{\phi} - A_{ht}^{\lambda} - J_{ht}^{\lambda}, \quad \forall h, t, \lambda \in \Lambda_t, \lambda' \in \Lambda_{t+1}, \phi \in \Phi_t, \\ & (S_{ht}^{\lambda}, A_{ht}^{\lambda}, J_{ht}^{\lambda}, ET_{kt}^{\lambda}) \in FL \end{array} \tag{1.1}$$

Existe un conjunto de límites técnicos para las unidades (puntos máximos y mínimos de funcionamiento) y consideraciones de seguridad (altura de lagos y límites de erogación en las represas) que son capturados por el conjunto FL, que corresponde al espacio de combinaciones de variables técnicamente factibles. Se remarca que éste es un problema de optimización estocástica, debido a la incertidumbre en la demanda residual y en los aportes.

Observar que en la formulación anterior, la decisión óptima tenderá a usar toda el agua posible para atender la demanda, ya que eso reduce los gastos de combustible, que son los únicos contabilizados en esa función objetivo. Esa tendencia a vaciar los lagos simplemente difiere costos hacia etapas posteriores del horizonte de planificación, lo que representa en general un problema financiero sobre horizontes extensos. Económicamente hablando, el agua es un activo rentable, ya que al mismo caudal de aportes, la altura del lago multiplica la eficiencia energética de los mismos. Por el contrario, si se es muy conservador con el agua y siempre se tiende a mantener los lagos a un nivel alto, podría llegar a ser mandatorio el vertido ante rachas de altas precipitaciones.

De entre las estrategias para paliar los defectos en la gestión del recurso hídrico destacamos dos. La primera pasa por tomar un horizonte de tiempo T muy largo, para que sucedan varios ciclos de carga/descarga en los embalses y el efecto del nivel final se diluya. La principal desventaja de esta aproximación pasa por su complejidad, ya que los pasos del control (i.e. Δt) no pueden ser muy extensos (e.g. se suele usar 1 hora). Si se planificara tan sólo un año en pasos de 1 hora, habrían 8760 pasos. Al multiplicar esto por los cuatro tipos de variables (S_{ht}, A_{ht}, J_{ht} y ET_{kt}), por la cantidad de unidades de cada tipo y por los escenarios de demanda y aportes en cada instante ($\lambda \in \Lambda_t$ y $\phi \in \Phi_t$), se alcanzaría un problema de dimensiones extravagantes para varias técnicas de optimización, aunque no necesariamente para las técnicas de Aprendizaje por Refuerzos (*Reinforcement Learning* en inglés) que utilizaremos en este trabajo.

Una segunda aproximación es usar un modelo distinto para mediano/largo plazo, donde se priorice la gestión del agua sobre el detalle técnico del control a corto plazo. El resultado de ese modelo sería cuantificar el costo de oportunidad asociado al volumen de agua en los embalses, que posteriormente podría sumarse a $g_k(ET_{kt}^{\lambda})$ como costo en la función objetivo de la formulación anterior, usando horizontes T cortos (de dos o tres días).

1.2. Antecedentes

En lo que refiere a los algoritmos utilizados para resolver el problema de valorar el agua en los embalses mediante una optimización al largo plazo se han propuesto varios métodos en la literatura, entre los que se destacan: *Programación No Lineal (NLP)* (Habibollahzadeh y Bubenko, 1986), *Dynamic Programming (DP)* (Jin-Shyr y Nanming, 1989), *Lagrangian Relaxation (LR)* (Ngundam et al. 2000), *Tabu Search (TS)* (Bai y Shahidehpour, 1996), *Genetic Algorithms (GA)* (Zoumas et al. 2004), *Programación Dinámica Estocástica (SDP)* (van der Wal, 1981) y *Programación Estocástica Dinámica Dual (SDDP)* (Rotting y Gjelsvik, 1992; Shapiro, 2011).

En particular, la técnica SDP es utilizada en este trabajo como marco de referencia para comparar los resultados obtenidos mediante *Reinforcement Learning*, y por eso se elabora en los capítulos 2, 3 y 4.

El objetivo de nuestro trabajo es explorar variantes en la formulación del *despacho hidrotérmico* y resolverlas haciendo uso de técnicas de *Reinforcement Learning*, con el fin de analizar la idoneidad de esos algoritmos para atacar un problema de creciente relevancia. Siempre que la complejidad del problema lo permita, se tomará como referencia de calidad la solución conseguida mediante alguna técnica tradicional.

Antes de comenzar, hacemos la aclaración de que no se hace mención a trabajos publicados posteriormente al comienzo de este trabajo, momento en que se realizó el relevamiento del estado del arte.

1.2.1. Ámbito Local

Este repaso de antecedentes enumera en orden cronológico los distintos tipos de estudios y/o aplicaciones relacionadas al objeto de estudio de la tesis. La lista incluye desde proyectos de investigación pura hasta paquetes de software comerciales, que son o han sido usados en producción en Uruguay en algún momento. Los del segundo grupo se elaboran brevemente en las subsecciones siguientes.

El primero de los modelos utilizados en Uruguay fue el modelo EDF: software desarrollado a medida por la empresa francesa "Electricité de France (EDF)", contratada por UTE en la segunda mitad de los 80s. Fue utilizado por UTE para resolver el problema de despacho óptimo a largo plazo entre los años 1988 y 2018 aproximadamente (Ferreira, 2008, EJECUTIVO, 2002). Se elaborará en la Sección 1.2.3.

Posteriormente, se han identificado una serie de proyectos de investigación en la temática enmarcados en diferentes instrumentos. A saber:

i) Convenio UTE-FING(IMERL/CeCal) *"Mejoras en los programas de optimización y simulación de la generación de energía eléctrica"* (Piria et al. 1992-1993). Se efectuaron ajustes en los programas de la empresa EDF (usados en la Gerencia de Planificación de UTE), lográndose reducciones en los tiempos de cálculo;

ii) Proyecto CONICYT-FING(IMERL/CeCal) *"Optimización de la coordinación hidrotérmica en el corto plazo (HIDROTER)"* (Piria et al. 1994-1997). Realizó la optimización del despacho de corto plazo para el problema de UTE. Incluyó la creación de los modelos y el desarrollo de los algoritmos, obteniéndose un paquete completo, probado en casos concretos suministrados por el Despacho de Carga de UTE. El software fue patentado en 1996;

iii) Convenio UTE-FCien, PEDECIBA-Matemáticas *"Cadenas de Markov gobernando algunos procesos aplicables a los ríos"* (1994 a 1995):

iv) Proyecto PARALIN con la Comisión Europea (Jofré et al. 1996-1997), para el desarrollo de algoritmos de computación de alta performance aplicados a problemas de energía (planificación de inversiones, problemas despacho de mediano y corto plazo), entre 1996 y 1997.

Otro de los modelos utilizados en producción es el modelo OPERGEN. El modelo OPERGEN es un software desarrollado a medida por la consultora IBERDROLA-PSRI, contratada por UTE a tales efectos, y que se utilizó entre el 2002 y el 2015 aproximadamente para la resolución del problema de despacho óptimo en el mediano y corto plazo (Ferreira, 2008, EJECUTIVO, 2002). Se elaborará en la Sección 1.2.4.

Para resolver el problema del despacho óptimo, la Administración del Mercado Eléctrico (ADME, 2024), ha usado durante varios años la herramienta SimSEE (SimSEE, 2024). UTE también hace uso de esta herramienta desde 2012. SimSEE fue desarrollado en el Instituto de Ingeniería Eléctrica (IIE) de la Facultad de Ingeniería (FING) en el marco del proyecto PDT 47/12 financiado por el BID durante los años 2006 y 2007, bajo la dirección del Dr. Ing. Gonzalo Casaravilla y con el apoyo de Facultad de Ingeniería, MIEM, URSEA, ADME y UTE (Casaravilla et. al., 2009, Chaer et. al., 2013, Chaer, 2008). Se elaborará en la Sección 1.2.5.

En la siguiente sección presentamos algunos de los conceptos y abstracciones en común más relevantes que usan estos modelos. Luego entramos en detalle en los modelos usados en producción (Sección 1.2.3 a Sección 1.2.5).

1.2.2. Abstracciones en común

En esta sección se explican algunos conceptos que utilizan las herramientas que se han usado en Uruguay, históricamente, para resolver el problema del despacho hidrotérmico óptimo.

1.2.2.1. Postes

Uno de los conceptos que vale la pena introducir es el concepto de los “postes” o “bandas horarias”. Algo que se asume en los modelos al optimizar es que la potencia de los generadores así como la demanda son constantes en cada paso de tiempo (se suele usar la potencia media). Esto es porque lo que se impone es un balance de energía en el paso, es decir que: en cada paso se tiene que generar la misma energía que se demanda. Por eso queremos que la potencia media de los generadores y la demanda sean los más representativos durante el paso del tiempo.

El mercado eléctrico uruguayo es marginalista: despacha a los generadores (i.e., produce energía) en orden creciente de costo hasta satisfacer la potencia en cada instante. En un sistema así, la función *costo vs potencia* es creciente y convexa. Como consecuencia de la última propiedad, atender una demanda variable en un período resulta más caro que atender una demanda sostenida e igual al promedio, aunque la energía consumida es la misma en ambos casos.

La demanda eléctrica del sistema tiene un perfil característico diario típico, con dos períodos de "pico" de demanda y dos "valles", como se puede apreciar en la Figura 1.2. Si se aplicara una discretización semanal del tiempo promediando el consumo, se perdería la información de esos efectos. Esto es muy importante porque es la potencia la que determina qué generadores debemos prender para satisfacerla, y el promedio esconde los picos.

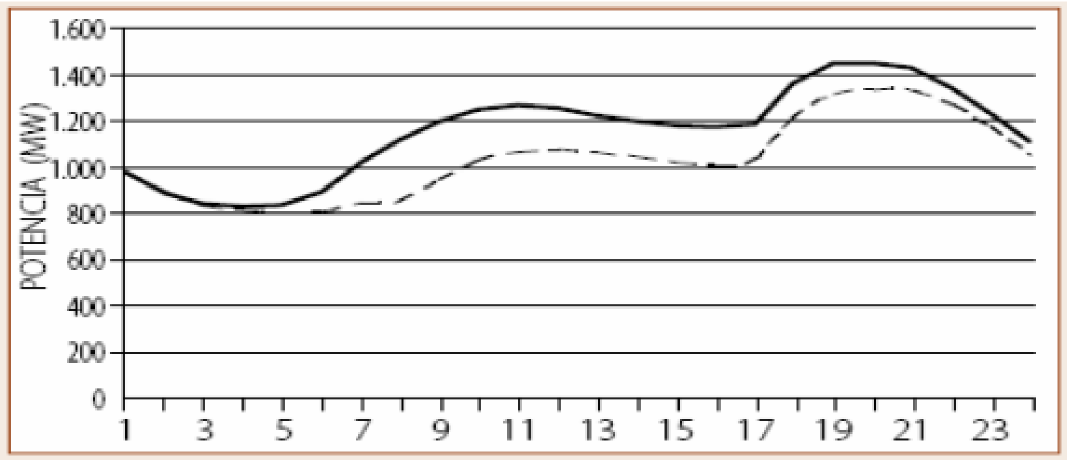


Figura 1.2: Curva de demanda semanal (La línea punteada es la demanda de los fines de semana). (Fuente: Ferreira, 2008)

Para evitar este inconveniente sin tener que disminuir el paso del tiempo, se introduce el concepto del "poste" o "banda horaria". Uno podría considerar una serie completa de consumos horarios semanal, una serie de 168 valores horarios consecutivos, y en cada transición semanal del proceso de optimización de largo plazo, calcular el costo incurrido usando la serie horaria completa. Los primeros antecedentes de modelado en nuestro país se remontan a los 80's, en el proyecto EDF (Ferreira, 2008) donde el poder de cómputo disponible era ridículamente bajo si se compara al actual. En una suerte de compromiso, la semana de descomponía en cuatro postes de distinta duración, ordenados en forma decreciente: uno de 5hs con el promedio del pico diario estacional, que se da entre lunes y viernes, otro de 30hs, con el promedio de las 2hs previas y posteriores al pico de lunes a viernes

(20hs), más las otras 10hs mayores, ya sean en fines de semana o días hábiles. Los otros dos postes capturaban los valles y las horas comunes.

Capturar el consumo en una sub-secuencia monótona decreciente de valores presenta al menos dos problemas. En primer lugar, se asume que el consumo es conocido (i.e., determinístico). Ese supuesto era realista hasta la introducción de las energías renovables no convencionales. Desde ese momento, la demanda a despachar es la residual: la diferencia entre la demanda de los usuarios del sistema (varianza muy baja) y la generación eólica más la solar, que tienen mucha dispersión estadística. La generalización de los postes a variables aleatorias mitiga este problema. El segundo problema identificado es que el reordenamiento rompe la correlación temporal. Así como es más caro atender una demanda variable que una constante, la existencia de unidades con procesos de arranque largos y/o costos (commitments) encarece el despacho cuando los picos se encuentran distribuidos en el tiempo, como en nuestro caso.

1.2.2.2. Estados Hidrológicos

Otro denominador común que suelen tener los sistemas de planificación a mediano/largo plazo son los estados hidrológicos. Como su nombre indica, la idea es capturar el nivel de hidrología del sistema hidroeléctrico, no sólo en el instante actual, sino también en algún contexto determinado. El contexto hidrológico es claramente importante, ya que no es lo mismo recibir un aporte determinado en medio de una sequía, que recibir el mismo aporte durante una inundación. La decisión respecto a si usar o guardar esa agua en esos dos contextos es radicalmente distinta.

Uno de los instrumentos más usados para asociar contexto y aportes son las *Cadenas de Markov*, una herramienta de la teoría de la probabilidad usada para representar un tipo especial de proceso estocástico discreto, en el que las probabilidades de emisión de otras variables aleatorias (los aportes en nuestro caso) quedan capturadas en un conjunto discreto de estados, a los que se puede llegar desde el resto de los estados con probabilidades fijas y conocidas para cada estado. Al igual que sucedió con los postes (ver Sección-1.2.2.1), los primeros antecedentes en nuestro país sobre modelado de aportes a través de Cadenas de Markov se remontan a los 80's, y se dieron en el marco del desarrollo de modelos cuantitativos para la toma de decisiones de despacho en UTE, en el proyecto EDF (Ferreira, 2008). La variable de estado hidrológica en ese caso se denominó $ESHY$: una variable que resume el estado hidrológico del sistema y se calcula para cada instante de tiempo t, como la media de los aportes registrados en las 12 semanas precedentes para cada una de las represas, ponderadas con los coeficientes energéticos medios de cada central.

A efectos ilustrativos y aunque en realidad se usan ventanas móviles de 12 semanas, supongamos que se cuenta con las crónicas de ternas de aportes semanales, esto es, $ternas = \{ap_t \in \mathbb{R}^3, t \in sem\}$ para un conjunto discreto de sem semanas consecutivas. Cada terna registra los aportes recibidos en los tres lagos con gestión: Bonete, Palmar y Salto. No hay registros de buena calidad para Baygorria, que además se usa como central de paso. Supongamos que también se cuenta con una función $pot : \mathbb{R}^3 \rightarrow \mathbb{R}$, que asocia a cada terna un valor de energía en agua, como si todos los aportes recibidos se hubieran usado para generación hidroeléctrica, turbinando a la cota media de cada uno de esos embalses. El resultado de $pot(ternas)$ es un conjunto de valores de energía hidráulica, que agrupados por quintiles, particionan el conjunto $ternas$ en cinco grupos, cada uno asociado a un estado hidrológico $ESHY = \{1, 2, 3, 4, 5\}$. La preimagen de un estado $i \in ESHY$ mediante pot es el conjunto de muestras de ternas de aportes asociadas a ese estado, y cualquiera de sus componentes sería una realización de aportes representativa para i. A su vez, el registro de la secuencia de estados en el recorrido de la variable $ESHY$ permite capturar las probabilidades de transición entre los estados.

Como referencia específica más próxima sobre el uso de Cadenas de Markov para estas aplicaciones, sugerimos remitirse a (Iribarren, 1999), las notas del matemático Gonzalo Pérez Iribarren editadas por el Centro de Matemática (FCIEN, UdelaR), en el marco de un proyecto Pedeciba de investigación aplicada desarrollado para UTE entre 1995 y 1996.

Otro ejemplo ilustrativo es el estado hidrológico elaborado en el presente trabajo. Si bien para simplificar los experimentos se terminó utilizando un solo estado hidrológico, al desarrollar el modelo de aportes se desarrolló un modelo para el estado hidrológico anual, el cual está detallado en la Sección 3.1.1.

1.2.2.3. Dinámica de los embalses

Otra propiedad subyacente explotada por los sistemas proviene de la dinámica de los lagos. La Tabla 1.1 presenta algunos parámetros de las centrales hidroeléctricas del Uruguay. En correspondencia con la imagen en Figura 1.1, se agrupan las unidades entre las del Río Negro (Bonete, Baygorria y Palmar) porque se encuentran en serie. La central binacional de Salto Grande (compartida con Argentina), ubicada sobre el Río Uruguay, funciona independientemente. La tabla muestra los tiempos de traslado entre los erogados del Río Negro, así como las potencias máximas y el tiempo de vaciado a potencia plena.

Nombre de la Unidad	**Número de Turbinas**	**Potencia Máxima**	**Días* para su vaciado**	**Origen de los aportes**
Bonete	4	149MW	182	Río Negro aguas arriba
Baygorria	3	111MW	3	Bonete [+8hs] más otros
Palmar	3	343MW	13	Baygorria [+16hs] más Yí
Salto Grande	14	1890MW	15	Río Uruguay aguas arriba

Tabla 1.1: Parámetros de unidades hidráulicas del Uruguay. NOTA (*) asumiendo que no hay vertidos ni aportes hidrológicos (Fuente UTE-ADME.)

De esos datos y a efectos de este estudio, se destacan dos hechos: i) la potencia combinada en el Río Negro corresponde solamente al 40 % del total hidroeléctrico, si se contabiliza sólo el 50 % de Salto Grande que le corresponde al Uruguay; y ii) el tiempo de vaciado del lago de Bonete supera en más de un orden de magnitud al de Salto Grande. En otras palabras, el lago de Bonete –cuyos erogados son returbinados en Baygorria y Palmar– es el gran acumulador de energía del país. A cota plena y en momentos de sequía generalizada, la energía acumulada en Bonete utilizada eficientemente en el tándem sobre el Río Negro, estaría próxima el 90 % del acumulado hidroeléctrico total.

La característica anterior apoya el hecho que en los primeros sistemas de planificación optimizada a largo plazo (uno o más años, ver Sección 1.2.3) solamente se calculaba el valor del agua de Bonete, y se usaba para esto un paso semanal (se necesitan más de 25 semanas para vaciar este lago). Mientras que Baygorria tiene una represa muy pequeña y se la modela como una represa sin embalse (central de paso), o sea con valor de agua cero y todo el volumen de agua que recibe en una semana es erogado en la misma semana. Ante el escaso poder de cómputo de los 80's, la dinámica intrínseca a nuestro parque hidroeléctrico facilitaba el modelado.

Respecto a cómo los parámetros influyen en algunos sistemas posteriores. Por ejemplo para el sistema EDF, y ya que Palmar tiene un tiempo de vaciado de entre 1 y 2 semanas, no se calcula el valor de agua para esta unidad, sino que tiene que ser ingresada por el usuario. Se suele usar un valor prácticamente nulo. Por otro lado, la represa de Salto Grande tiene un tiempo de vaciado similar al de Palmar, pero es tratada de forma distinta por tratarse de una represa aislada y además compartida con la República Argentina. El coeficiente energético correspondiente es multiplicado por el porcentaje correspondiente a la parte uruguaya de la represa (actualmente

0.5). En otras oportunidades también se ha utilizado un valor de agua prácticamente nulo para esta represa, y su despacho es decidido manualmente por el usuario.

1.2.3. Sistema EDF

El propósito primordial del sistema EDF es valorar el agua en el embalse de la central Rincón del Bonete. El *valor del agua* se computa de forma indirecta. Como se adelantara en la Sección 1.1, el valor deriva del costo de oportunidad para distintos niveles de stock inicial de agua en el embalse de esa represa. Los valores son el resultado de una optimización estocástica que minimiza la esperanza del costo total de gestión, esto es, de la suma del costo de explotación (consumo de combustible incurrido en las diferentes unidades) y el costo de no suministro de la energía eléctrica, o costo de falla. El costo se calcula sobre un período T de uno o más años, en pasos semanales. Las incertidumbres consideradas son: en la demanda, la disponibilidad de equipos (i.e., fallas que saquen unidades de generación de producción), el comportamiento hidrológico de los aportes (Sección 1.2.2.2) y los intercambios con países vecinos. El sistema consta de dos módulos:

MURVAGUA: Es el responsable de asignarle valor al agua para un conjunto discreto de niveles del lago en Bonete. Usa una variante de la Programación Dinámica Estocástica (SDP, ver Sección-2.2.5), en la que el estado surge de la combinación de: 10 volúmenes uniformes de reserva para el agua almacenada en Bonete; con los 5 estados hidrológicos identificados en la Sección 1.2.2.2. El foco en el lago de Bonete se deriva del hecho que es el que tiene la dinámica más lenta entre las represas (vaciado de 26 semanas a turbinado pleno sin aportes nuevo, ver 1.1). El tiempo de vaciado de las otras centrales está por debajo o próximo al paso en que se eligió discretizar el tiempo. Salto Grande tiene 213 % más potencia que la suma de las centrales sobre el Río Negro, pero sólo 10 % de su energía, porque es binacional, sus erogados no se reutilizan y por la diferencia en el tiempo de vaciado. EDF captura en el estado el hecho que el lago de Bonete es el acumulador de energía en nuestro país. En la Sección 1.2.3.1 se elabora en algunos detalles de este módulo.

MURDOC: La forma convencional en la que la SDP se presenta es como en Sección-2.2.5, donde los valores de Bellman en los distintos estados y los controles asociados se calculan conjuntamente.

Ésta es también la aproximación seguida en esta tesis cuando se usa SDP para tomar referencia de los resultados (Sección 3.5.2). La implementación en MURVAGUA sigue un aproximación distinta, que no incluye los controles como parte del resultado. El propósito de MURDOC es simular trayectorias óptimas para el control, que surgen de tomar realizaciones del procesos de aportes hidrológicos, para despacharlas óptimamente integrando el dato de valor de oportunidad del agua calculado previamente en MURVAGUA. La Sección 1.2.3.2 presenta otros detalles de este módulo.

1.2.3.1. Módulo de Optimización (MURVAGUA)

Dadas las constantes de tiempo de los embalses representados (Sección 1.2.2.3) y los regímenes hidrológicos (Sección 1.2.2.2), se eligió una discretización semanal del período de tiempo T de estudio (horizonte de optimización).

El módulo de optimización realiza el cálculo de valores esperados del agua en el embalse de Bonete (Palmar y Salto se representan con valor de agua nulo). La técnica de optimización utilizada es la programación dinámica estocástica (SDP, ver Sección-2.2.5). Se obtiene como resultado de este módulo la valorización económica del embalse de Bonete (derivada del costo futuro esperado respecto al stock en esa represa). El modelo SDP integra el nivel de Bonete y la Clase Hidrológica en cada uno de sus estados. En cada paso semanal, el cálculo de la función de Bellman agrega un nivel adicional de detalle, sumando variables de control y estado hasta cubrir: los volúmenes turbinados y vertidos en las centrales del Río Negro y Salto Grande; los niveles de los lagos al final del período; el nivel de generación de las centrales térmicas; la política de intercambio con otros mercados energéticos; el nivel de falla producido.

Se modelan los siguientes tipos de centrales de generación de energía eléctrica: Centrales térmicas a vapor; Turbinas de gas; Centrales hidráulicas: se dispone de funciones cota-volumen, coeficientes energéticos en función del erogado y el salto, para las tres centrales modeladas con embalse y para Bonete se incluye además una función de evaporación.

Las centrales hidráulicas consideradas son:

Bonete: Valor de agua en función del nivel de la represa, la clase hidrológica y la semana. Se modela con embalse

Palmar: Valor de agua nulo. Recibe aportes de agua propios más el agua erogada en Baygorria. Se modela con embalse.

Salto: Valor de agua nulo. Se modela con embalse.

Baygorria: Recibe el agua erogada en Bonete y no tiene aportes propios. Eroga la misma cantidad de agua que recibe (sin embalse).

Las centrales hidráulicas de Bonete, Baygorria y Palmar están ligadas por restricciones de pasaje del agua. El método de programación dinámica estocástica –utilizado para el cálculo del valor de agua– se basa en el principio de Bellman y en la identificación Markoviana de los procesos de aportes en los ríos (Sección 1.2.2.2), y en la ecuación de balance de masa, explicada en Sección 1.1. Este método opera de atrás hacia adelante en el tiempo calculando los valores de Bellman como se muestra en el ejemplo de la Figura 2.4. Los procesos estocásticos considerados son: demanda; aportes; disponibilidad de equipos; precios y disponibilidades de importación/exportación. Los valores de Bellman representan los valores esperados de los costos futuros a partir de ese estado hasta el final del periodo T considerado. Los valores finales de Bellman (los de la etapa T) se asignan a cero. Se debe tomar un horizonte de algunos años para diluir el error de asignar ese valor final.

En un instante de tiempo dado t, el estado del sistema se define por una combinación entre el nivel del embalse de Bonete (se dividió en 10 niveles) y el valor de la clase hidrológica (5 clases posibles), resultando en 50 estados (10×5) por etapa.

Para un estado dado (nivel de Bonete, clase hidrológica) en un instante t, el cálculo del ***costo semanal de operación semanal*** (COP) se realiza en forma aproximada, mediante una variante del Método Monte-Carlo (ver Sección 2.1), ya que hay cuatro fuentes de incertidumbre en la programación semanal, alcanzando: demanda, aportes, disponibilidad de equipos generadores y precio/disponibilidad de importación/exportación. El COP se calcula promediando los óptimos para una cantidad R de realizaciones conjuntas de las variables aleatorias, que se encuentran resolviendo problemas de programación lineal independientes.
Las realizaciones de los datos con incertidumbre se construyen:

Demanda: se usan valores medios estacionales para esa semana t, con los que se construye una realización de los postes (o bandas horarias, ver Sección 1.2.2.1).

Aportes: se toma una realización histórica en base a la preimagen del estado hidrológico $i \in ESHY = \{1, \ldots, 5\}$, según se ilustró en la Sección 1.2.2.2.

Importación de energía: Para los *costos de importación* se usan los valores promedios para el estado hidrológico $i \in ESHY$, en el entendido que los vecinos (Argentina y especialmente Brasil) también tienen alta dependencia de la generación hidráulica, y que ésta tiene fuerte correlación con la Uruguaya.

Disponibilidad de equipos: en algunos casos se reduce la disponibilidad máxima de acuerdo a la probabilidad de falla del equipo y en otros se trabaja con escenarios, el usuario elige las alternativas. De acuerdo al resultado, solamente se consideran las unidades disponibles para la programación semanal.

Los valores de Bellman representan los valores esperados de los costos futuros a partir de ese estado hasta el final del periodo T considerado. Los valores finales de Bellman (los de la etapa T) se asignan a cero. Se debe tomar un horizonte de algunos años para diluir el error de asignar ese valor final.

La derivada respecto al stock (i.e., el nivel del lago) del valor de Bellman (con signo cambiado) es el denominado como *valor del agua* y se mide en unidades monetarias por unidad de volumen de agua en el embalse considerado. Debido al bajo nivel de discretización usado (10 estados del lago), la derivada de la función de Bellman no se aproxima por cocientes incrementales. En cambio, se usa el hecho que la implementación de EDF estimó la función de Bellman en la etapa siguiente (computada en la iteración previa por ir hacia atrás) resolviendo un problema de Programación Lineal. En los problemas de Programación Matemática (ver Sección 2.2.2) en general, en los de Programación Lineal en particular, el costo marginal de una restricción –la variación del costo futuro respecto al nivel del lago en este caso– se corresponde con los ***Multiplicadores de Lagrange*** en el óptimo del problema para esas restricciones (ver *shadow prices* en Boyd y Vandenberghe, 2004).

Como anticipamos, en un instante de tiempo dado $t < T$, el estado del sistema se define por el valor del nivel $1 \leq s \leq 10$ del embalse de Bonete en conjunto con el valor de la clase hidrológica $1 \leq i \leq 5$. Con los valores concretos de la terna (t, s, i) y otros parámetros fijados por el operador, se generan realizaciones de las variables aleatorias: demanda/postes, aportes hidrológicos semanales por represa, disponibilidad de equipos generadores y precio/disponibilidad de importación/exportación.

Para la programación semanal se simula un despacho óptimo usando las realizaciones anteriores como insumo. Los costos de producción térmica y de falla son parámetros del sistema. Para las centrales Salto Grande y Palmar se usan las cotas medias de los lagos como altura referencia para la generación, y se asume valor del agua nulo. El único dato restante es el del valor del agua a usar en Bonete (el nivel es parte del estado).

Se recuerda que el tiempo de vaciado para Bonete a potencia máxima es de 26 semanas (Tabla 1.1), por lo que a turbinado máximo, en una semana no se alcanzaría la mitad del erogado necesario para mover el nivel del lago a un estado menor (la discretización es cada 10 % del volumen).

Además, para el costo semanal se utilizará el valor del agua calculado como se mencionó anteriormente, como la derivada respecto al stock de agua del valor de Bellman en la semana siguiente ($t + 1$), para el mismo nivel del lago s y el mismo estado hidrológico i. Así, el cálculo del costo de operación semanal se vuelve determinístico en cada una de las R realizaciones, y su promedio es un estimador del ***costo de operación semanal aproximado*** (COP).

Con esto tenemos un aproximado del costo de transición, al que falta ahora sumarle el costo futuro esperado. Para calcular el costo futuro esperado, hay que tener en cuenta si hubo una transición de ESHY o no. La referencia de estados hidrológicos es una cadena de markov en la que a lo sumo se puede pasar de un estado al anterior y/o al posterior, como en el ejemplo de aportes elaborado en la Sección 3.1.1 (con la salvedad de que aquella es anual).

Teniendo las 3 posibles transiciones de ESHY (i, $i - 1$ de ser posible, $i + 1$ de ser posible), y conociendo el nivel del lago ($1 \leq s \leq 10$), conocemos los 3 valores de Bellman correspondientes en el paso siguiente ($t + 1$). El costo esperado futuro (CF) es entonces la ponderación de estos 3 valores de Bellman según sus respectivas probabilidades de transición. El CF sumado al COP es entonces el valor de Bellman para el instante actual que queríamos hallar.

Para dar un ejemplo, supongamos que en el instante t nos encontramos en un ESHY = 2, y supongamos que las probabilidades de transición y los valores de Bellman en $t + 1$ sean los siguientes:

- ESHY 2 a ESHY 1 $\Longrightarrow$ 25 %, Valor de Bellman = 2000
- ESHY 2 a ESHY 2 $\Longrightarrow$ 60 %, Valor de Bellman = 1500
- ESHY 2 a ESHY 3 $\Longrightarrow$ 15 %, Valor de Bellman = 1000

Entonces podemos calcular el CF como:

$$CF = .25 * 2000 + .60 * 1500 + .15 * 1000 = 1550.$$

Supongamos además que la simulación de Monte Carlo resultó en un:

$$COP = 50.$$

El valor de Bellman que queremos calcular es entonces:

$$ValordeBellman = COP + CF = 1600.$$

Repitiendo este proceso siguiendo la metodología de programación dinámica, de atrás hacia adelante y para cada estado, podemos calcular todos los valores de Bellman que necesitamos. Cabe aclarar que como en el cálculo de cada valor de Bellman se necesitan los valores de Bellman/valores del agua de la semana siguiente, en la última semana los mismos se consideran nulos. Es decir que en el último paso, para obtener el valor de Bellman, basta con calcular el costo de operación de esa semana usando un valor de agua nulo.

1.2.3.2. Modelo de Simulación (MURDOC)

Como se mencionara al inicio de esta sección, la implementación en MURVAGUA no se ajusta estrictamente al esquema clásico de SDP presentado en Sección-2.2.5, e implementado en Algoritmo-8, que al igual que en Algoritmo-7, calculan los valores de Bellman en conjunto con el control a seguir. No obstante, tener los valores de Bellman permite derivar controles ante diversas situaciones. Una variante de esa idea se usa en esta tesis para ver cómo cambia la performance del despacho cuando el dato de aportes de la semana por venir se conoce a-priori (Algoritmo 9, Sección 3.5.3).

En el módulo MURDOC, las variantes del control se reconstruyen con los valores de agua calculados en el módulo MURVAGUA, y se usan para simular la operación del sistema. Con todos los costos de generación conocidos (incluye la valorización del agua embalsada), se determina aquella política que, para cada semana, sea la mejor operación del sistema. Al contar con los valores del agua, las unidades hidráulicas se pueden considerar análogas a una unidad térmica, siendo el valor del agua el costo de oportunidad correspondiente a la sustitución de máquinas térmicas. Además, al tener los valores del agua implícito dicho costo, se puede considerar como un problema de optimización *greedy*, en donde basta optimizar de forma local en cada paso. Se utiliza programación lineal para resolver la optimización de cada semana.

Considerando diferentes escenarios de los datos con incertidumbre, se efectúan las simulaciones, se recogen los resultados para cada escenario y se realizan cálculos estadísticos para las variables aleatorias relevantes (esperanzas, varianzas, percentiles, probabilidades para los diversos resultados de la operación). Para ello, las simulaciones usan múltiples realizaciones de los procesos estocásticos. En su parte experimental, esta tesis también incorpora ejemplos basados en esa idea, como los de Figura 3.4 y Figura 3.5 en Sección 3.5.

A efectos de la construcción de escenarios para la simulación, para los aportes de agua en las represas se trabaja con un conjunto de crónicas históricas y para la importación y exportación de energía con un conjunto de escenarios en precios y disponibilidades. Se utiliza generalmente la demanda media anual, calculada en el módulo de demanda. Se sortean la disponibilidad de los equipos de generación para cada escenario y semana.

1.2.4. OPERGEN

El modelo EDF se utiliza para valorizar el agua en Bonete (MURVAGUA) y para evaluar realizaciones del despacho –o cambios en él– mediante simulaciones (MURDOC) en horizontes de largo plazo (1 año o más). Entre las aplicaciones derivadas de una herramienta como la anterior está la cuantificación del retorno de inversiones. Por ejemplo, el retorno de inversión en la compra o arrendamiento de una unidad generadora, podría estimarse simulando el resultado del sistema en el período de repago operado con y sin esa unidad. Esta es una aplicación muy común en la planificación de un sistema eléctrico.

Otra aplicación, menos estratégica, pero mucho más notoria y crítica, hace a la planificación del despacho del sistema a corto plazo. El sistema eléctrico requiere de un control continuo, que tiene entre sus principales objetivos determinar cuáles son las unidades generadoras a usar en los días por venir, y cómo debe coordinarse su entrada/salida del sistema.

En un sistema intrínsecamente hidrotérmico (como era el uruguayo hasta el 2010) la optimización del despacho en un horizonte de corto plazo (una semana o menos) podía resolverse como un problema de programación matemática en la que, si el valor del agua es conocido, las unidades hidráulicas se asimilan a unidades térmicas. El sistema EDF encuentra ese valor para Bonete. Cuando nos movemos a la planificación del despacho a mediano plazo (algunos meses) es necesario calcular valores del agua para Salto Grande y Palmar, que requiere modelos/componentes de software con horizonte de planificación más corto que los vistos en la Sección 1.2.2.3.

Más aún, en la planificación a corto plazo (1 semana), el problema podía tratarse como determinístico hasta mediados del 2010, ya que pueden hacerse aproximaciones fiables de las incertidumbres como los aportes, y la demanda era altamente predecible en los días próximos por la estacionalidad y los pronósticos de temperatura. En el corto plazo incluso se puede modelar el almacenamiento en Baygorria, en vez ser solamente una central de paso como suele ser en los otros casos. Con un horizonte de optimización tan corto, y más aún siendo el problema determinístico, se podía agregar mucha más complejidad que ayudaba a mejorar el detalle de objetos integrados a los modelos, manteniéndose su resolución dentro de una eficiencia computacional razonable, utilizando técnicas como programación lineal.

Esta sección presenta el sistema OPERGEN: la implementación de un conjunto de modelos de optimización y simulación para mediano (Modelo MP) y corto plazo (Modelos CPC/CPS) que abordan los problemas antes descritos.

Los modelos OPERGEN –en conjunto con el modelo EDF– se utilizaron hasta 2015 de forma complementaria para resolver el despacho en diversos horizontes de tiempo. Por ejemplo, el modelo de largo plazo (EDF) calcula valores del agua de la central de Bonete, para determinada etapa del horizonte de optimización. Luego esos valores se utilizan, como datos de entrada, en la programación de mediano plazo (MP). A su vez, la etapa final de la programación de mediano plazo, coincide con la primera del largo plazo, para la cual se calcularon los valores del agua de Bonete. Adicionalmente, el modelo de mediano plazo calcula los valores de agua de las tres centrales (Bonete, Palmar y Salto Grande) en el comienzo de su horizonte de optimización, que coincide a su vez con el fin del horizonte de optimización del modelo de corto plazo (CPC/CPS). Finalmente, tal como se representa en el siguiente esquema de la Figura 1.3, el modelo de corto plazo utiliza esta valorización de los embalses, al final de la etapa que optimiza, para calcular el despacho óptimo.

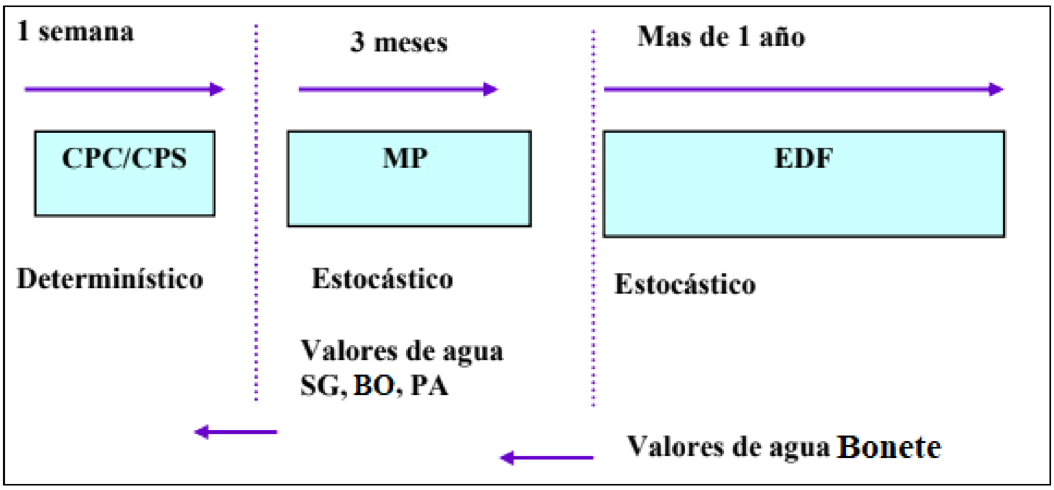


Figura 1.3: Funcionamiento complementario de los modelos EDF y OPERGEN. (Fuente: Ferreira, 2008)

Como se observa en la Figura 1.3, se trata de un esquema de optimización desde el futuro hacia el presente. En los modelos se puede trabajar con diferentes horizontes de tiempo: corto, mediano y largo plazo.

El problema de corto plazo se resuelve con un modelo de optimización determinístico mixto, con variables continuas y enteras, con algunas funciones no lineales.

Los problemas de mediano y largo plazo corresponden a modelos de optimización estocástica, que se resuelven aplicando variantes de la programación dinámica estocástica. En estos modelos se realiza una descomposición temporal del problema de optimización original en varias etapas, subdividiendo el período de optimización en subperíodos, resolviendo tantos problemas de optimización como etapas se consideren y bajo hipótesis de validez del principio de Bellman.

Modelos OPERGEN

Introducimos brevemente los modelos OPERGEN y luego se detallan más a fondo:

MP (modelo de Mediano Plazo): horizonte de optimización 3 meses. Es un modelo de optimización estocástico que calcula los valores de agua de Salto, Bonete y Palmar en forma conjunta. La obtención de la curva del valor del agua en función del volumen almacenado se realiza utilizando una metodología basada en SDDP (Shapiro, 2011). La técnica SDDP está desarrollada en la Sección 2.2.7.

CPC/CPS (modelo de Corto Plazo Completo/modelo de Corto Plazo Simple): modelos de corto plazo que optimizan la operación de una semana, calculando el despacho óptimo. Son modelos determinísticos. Utilizan un modelo de programación lineal sucesiva (Palacios-Gomez et al. 1982) para la optimización de una etapa.

A continuación se describen las diversas componentes de la función objetivo y un breve resumen de las restricciones consideradas en general en estos modelos.

Función objetivo:

- costo del combustible consumido,
- costo de operación de las centrales de generación hidráulica,
- costo de la compra de energía a otros sistemas,
- costo asignado a las interrupciones del suministro de energía, o sea, el costo de falla,

- costos futuros de operación (asociados a centrales de generación hidráulica),
- costo de arranque/parada de unidades térmicas,
- penalizaciones.

Restricciones consideradas:

Sistema térmico:

- Está constituido por unidades térmicas que pueden ser de tres tipos: unidades de Gas, unidades de Vapor y unidades de Ciclo Combinado.
- Las unidades térmicas se agrupan en centrales.
- Para centrales térmicas con combustible almacenable, se modela un almacenamiento de combustible con capacidad finita y posibilidades de aprovisionamiento en ciertas fechas.
- Posibilidad de mantenimiento programado.
- Posibilidad de modelar indisponibilidades fortuitas.

Sistema hidroeléctrico:

- Se modelan las centrales de Salto Grande, Bonete, Baygorria y Palmar.
- Salto Grande: en los modelos de Corto Plazo Simple y Mediano Plazo se considera que la energía producida en Salto Grande se reparte al 50 % entre Argentina y Uruguay, sin tener en cuenta el modelado de la diferencia embalsada.
- Baygorria: en el modelo de mediano plazo se considera como central de pasada (valor único de coeficiente energético en todas las etapas, sin embalse).

Restricciones de red:

- El sistema eléctrico está organizado por áreas geográficas, interconectadas con estructura de árbol o "radial".
- Las pérdidas en la red se reflejan a través de coeficientes unitariamente constantes.
- Se asume que no hay pérdidas en la transmisión de flujo de energía dentro de la misma área.
- Las áreas están interconectadas por líneas y cada línea tendrá asociado un coeficiente lineal de pérdidas y una capacidad máxima de transporte, pudiendo ser ambos distintos en función del sentido del flujo.

Demanda:

Se contemplan tres tipos de demanda:

- Con regulación tarifaria.
- Concertada mediante contratos bilaterales.
- A satisfacer en el mercado spot.

El segundo y tercer tipo permiten modelar intercambios con países vecinos, con diferentes precios y limitaciones. Se pueden introducir cualquier tipo de demandas por área. La demanda no satisfecha (excepto la de tipo 3) será penalizada mediante una función convexa escalonada por segmentos lineales.

Notar que los modelos de largo plazo como el EDF y el resto de modelos de largo plazo existentes, suelen modelar el sistema como uninodal (en el problema resuelto en este trabajo hacemos lo mismo). Eso significa que toda la energía generada entra a un único nodo, y hay que usarla para cubrir una única demanda. La realidad es que el sistema eléctrico uruguayo es multinodal, en donde se tiene que cubrir múltiples demandas para zonas distintas. Asimismo, la generación también es a nivel de áreas, y cubrir demandas en distintas áreas trae de la mano inconvenientes como las pérdidas de trasmisión, etc. En el corto plazo, como se puede ver en las restricciones de red y la demanda recién vistas, esto es muy importante, y los modelos de corto plazo pueden modelarlo debido a la mayor eficiencia computacional mencionada anteriormente.

1.2.4.1. Estructura temporal

Los modelos trabajan con una desagregación temporal en **etapas**, **períodos** y **bloques**. Todos los datos de entrada que varían con el tiempo (por ejemplo: aportes) dependen de la estructura temporal. El tiempo de cálculo depende fuertemente de la estructura temporal, que afecta la cantidad de variables y restricciones.

Etapas: intervalo de cálculo de las curvas de valores agua y costos futuros. Se toma normalmente de 1 semana.

Períodos: subdivisiones contiguas de una etapa, por ejemplo: un día. Un período corresponde al conjunto mínimo de bloques consecutivos para el que existen datos relativos a los aportes de agua entre otros atributos y, por otro lado, no interesa una agregación mayor. Influye en el árbol de escenarios (mencionados más adelante). Los balances de agua y combustible se hacen período a período

Bloques (equivalente a los postes de EDF)**:** subdivisiones contiguas de los períodos, generalmente se toman 3 bloques horarios consecutivos en el período (día) correspondientes a las horas de pico, valle y resto. Un bloque es la mínima unidad de tiempo que se considera en el modelo y al cual se le atribuye el mismo volumen de demanda, la misma utilización de agua en los embalses y otros atributos.

Dada una etapa, hay una restricción de demanda para cada bloque/período/área. Los modelos CPS/MP utilizan una estructura temporal con etapas/períodos/bloques, por ejemplo:

Duración de 1 etapa = 1 semana.

Duración de 1 período = 1 día.

Duración de 1 bloque = algunas horas.

El modelo CPC utiliza una estructura temporal con etapas/períodos, por ejemplo:

Duración de 1 etapa = 1 semana.

Duración de 1 período = 1 hora.

1.2.4.2. Modelo MP

Es un modelo de optimización estocástico que calcula los valores de agua de las centrales hidráulicas de Bonete, Salto Grande y Palmar en forma conjunta. El valor del agua de cada central depende así del estado o stock de agua propio y el de las otras dos centrales (nivel de referencia).

Para el tratamiento de la incertidumbre utiliza la técnica de análisis (simultáneo) de escenarios. Se consideran como variables aleatorias las siguientes magnitudes:

- Aportes en las represas
- Indisponibilidades en centrales hidráulicas y térmicas
- Disponibilidad de combustibles no almacenables
- Intercambios ocasionales:
  - Precio de compra/venta de energía
  - Potencia máxima de importación/exportación

Los datos inciertos se representan mediante árboles de escenarios. Un escenario es un conjunto de realizaciones posibles de una (o más) variable(s) aleatoria(s). Por ejemplo, si el horizonte de optimización es de 12 semanas, un escenario de aportes será una secuencia de 12 valores determinados de aportes.

Cálculo de la curva de valores de agua

La obtención de la curva del valor del agua en función del volumen almacenado se realiza utilizando una metodología basada en programación dinámica estocástica dual (SDDP, desarrollada en la Sección 2.2.7). Ello permite contemplar un gran número de variables de estado (en este caso los niveles de referencia de los embalses). Para su obtención se opera con un esquema de atrás hacia adelante en el árbol de escenarios considerado (como se explica en la Sección 2.2.7).

1.2.4.3. Modelos CPC/CPS

Son modelos determinísticos que optimizan la gestión de los recursos de generación en un horizonte correspondiente a la duración de una etapa, modelando más detalles que el MP. En la función objetivo puede aparecer el término correspondiente a la función de costos futuros que calcula el MP (función convexa lineal a tramos).

Las siguientes son algunas de las características particulares de estos modelos:

- Básicamente, estos modelos se corresponden con el MP para un escenario y etapa dados, pero tienen algunas diferencias en el modelado.
- Utilizan un modelo de programación lineal sucesiva para la optimización de una etapa.
- Permiten modelar el tiempo de tránsito en el volumen de agua fluyente.
- Se considera Baygorria como una central con salto variable y almacenamiento de agua.
- En la simulación operativa se pueden utilizar estos módulos para los cálculos de cada etapa

Diferencias entre CPS y CPC:

1. Modelado de procesos de arranque y parada de máquinas térmicas:
    - CPS: no modela duración de arranque, ni explícitamente su costo de mínimo técnico ni proceso de embotellado de las centrales térmicas a vapor. El costo de arranque de las unidades térmicas se tiene en cuenta mediante un sobrecosto por período de funcionamiento.
    - CPC: introduce variables binarias para el modelado en detalle de estos procesos, lo que introduce una mayor complejidad de cálculo y mayor tiempo de ejecución.

2. CPS: la operación de Salto Grande para Argentina se supone idéntica a la uruguaya.
3. CPC: usa una contabilidad de créditos de energía y optimiza el uso del embalse de Salto Grande según la cota vista.
4. CPS permite la agregación de varias horas en bloques para cada periodo de la etapa considerada (diferencia con CPC que utiliza solamente períodos y etapas).
5. CPS no permite la posibilidad de exigir la igualdad de erogado para ciertos periodos de tiempo.
6. CPS no permite el predespacho de unidades térmicas.

En la Figura 1.4 se comparan los tres modelos OPERGEN, con respecto a cómo optimizan una etapa en la cual no se consideran incertidumbres.

diferencias en modelado	Modelo		
	MP	CPS	CPC
optimización de 1 etapa	programación lineal	programación lineal y programación lineal sucesiva	a coef energ. fijo usa branch and bound con prog.lineal y luego programación lineal sucesiva
tiempo de tránsito en volumen de agua fluyente	no permite	permite	permite
Baygorria	de paso	central con salto variable y almacenamiento de agua	central con salto variable y almacenamiento de agua
proceso de arranque y parada de unidades térmicas	arranques con sobrecosto por periodo de funcionamiento	arranques con sobrecosto por periodo de funcionamiento	usa variables binarias para modelar proceso de arranque y parada
funcionamiento embotellado TV	no permite	no permite	si
coeficientes energéticos	constantes, según nivel de los lagos y operación hipotética	por aproximación lineal sucesiva	por aproximación lineal sucesiva
Operación de SG	Supone Argentina opera igual que Uruguay	Supone Argentina opera igual que Uruguay	hay contabilidad de créditos de energía, se optimiza el volumen visto
estructura temporal	etapa, periodo y bloque	etapa, periodo y bloque	etapa y periodo
predespacho de unidades térmicas	no se permite	no se permite	se permite
posibilidad de igualdad de erogado por periodos	no se permite	no se permite	se permite

Figura 1.4: Tabla comparativa de modelos OPERGEN. (Fuente: Ferreira, 2008)

1.2.5. SimSEE

El sistema “Simulación de Sistemas de Energía Eléctrica” (SimSEE, 2024) es la herramienta utilizada por la Administración del Mercado Eléctrico (ADME, 2024) para la gestión optimizada del mercado eléctrico uruguayo. Esta sección está basada en los trabajos: Casaravilla et. al., 2009, Chaer et. al., 2013 y Chaer, 2008.

SimSEE es una plataforma desarrollada con el propósito de facilitar la construcción de simuladores de sistemas de energía eléctrica. Los simuladores creados son herramientas de análisis de diferentes formas de operación del sistema mediante simulaciones de los diferentes escenarios, que permiten calcular los costos futuros de seguir por cada una de las trayectorias posibles y experimentar las consecuencias de las diferentes decisiones.

Una de las principales diferencias entre el SimSEE con las herramientas vistas hasta el momento (EDF, OPERGEN) es que, mientras que éstas fueron desarrolladas a medida para el sistema uruguayo, con los particulares de sus parque generador y demanda, la idea con el SimSEE es la de tener una herramienta más genérica, en donde el usuario pueda definir y configurar el sistema energético que quiera y poder trabajar dentro del mismo. Esto es útil, ya que el SimSEE fue en parte desarrollado con fines didácticos, y una herramienta de este estilo permite desarrollar varios ejercicios de variada complejidad para que los estudiantes puedan experimentar y aprender. Pero además, herramientas de carácter más genérico tienen sentido para que se puedan adaptar a un sistema cambiante.

En el momento en que se comenzó a desarrollar el SimSEE, la matriz energética uruguaya estaba cambiando drásticamente, con la introducción de una gran cantidad de generación renovable (eólica y solar). Este cambio en la matriz energética trajo una necesidad de agregar está nueva generación renovable a los modelos de despacho y optimización existentes (o crear nuevos), que habían sido creados a medida para un sistema que era en su gran mayoría hidro-térmico.

La forma en que funciona este modelo genérico, es que se considera al sistema compuesto por "actores" que intercambian energía. Los actores se clasifican en "generadores" a aquellos que entregan energía al sistema (generadores de energía, importaciones, etc.) y "demandas" a aquellos que consumen energía (demanda energética, exportaciones, etc.). Al igual que como se vio en el modelo OPERGEN, los generadores y las demandas pueden estar geográficamente distribuidas, por lo que es necesario considerar el sistema de transporte para representar adecuadamente las pérdidas de energía y los límites de la red eléctrica. El sistema de transporte se modela mediante "nodos" y "arcos". Los nodos son puntos del sistema a los cuales se conectan los generadores y las demandas. Los arcos son canales de conexión entre los nodos. En cada nodo se debe cumplir instantáneamente el balance de potencias y por tanto el de energía en el paso de tiempo considerado. Los arcos se utilizan para representar los límites de capacidad de transporte entre los nodos (un ejemplo de esto son las conexiones internacionales que tenemos con Argentina (mediante Salto

Grande [2000MW]) y Brasil (convertidoras de Rivera [70MW] y Melo [500MW]).

El esquema general de operación del SimSEE es el mismo que el visto para EDF y OPERGEN. Se cuenta con dos etapas: una etapa de optimización y una etapa de simulación.

En la etapa de optimización se calculan los valores del agua, función de costos futuros, y política de operación del sistema, al igual que lo visto en EDF, dividiendo el horizonte de tiempo en etapas, discretizando el lago en niveles, y utilizando la técnica SDP (más sobre esto en las secciones siguientes). La principal diferencia en esta etapa con EDF y OPERGEN, es que si bien EDF calculaba únicamente el valor del agua de Bonete mediante SDP, y OEPRGEN calculaba el valor del agua de Bonete, SG y Palmar mediante SDDP, en este caso se calculan los tres simultáneamente mediante SDP.

En la etapa de simulación, es análogo al modelo MURDOC del EDF. Aquí, se simula la operación del sistema utilizando los resultados obtenidos de la etapa de optimización, y se estudian varios escenarios de las variables aleatorias, ya sea mediante crónicas históricas (cuando están disponibles), o mediante series sintéticas y simulación de Monte Carlo (cuando se tiene un modelo para al distribución de la variable aleatoria). En cada paso de simulación se debe resolver un problema de despacho mediante la resolución de un problema de optimización con restricciones, que se plantea como un problema lineal y se resuelve mediante un algoritmo Simplex (Rutishauser y Gutknecht, 1991). En particular, el SimSEE adopta una implementación "colaborativa" del problema lineal, en el sentido de que cada actor del sistema tiene conocimiento de cuántas variables de optimización aporta al problema lineal (en la función objetivo), cuántas de ellas son variables enteras (lo que requiere un tratamiento especial en la resolución del Simplex) y cuántas restricciones quiere él imponer. Así, iterando sobre los actores aportando este conocimiento, se va construyendo el problema lineal completo (todos los postes a la vez, ver Sección 1.2.2.1) para el despacho a resolver.

Además de compartir técnicas y entidades con los sistemas anteriores, y aunque la parametrización de SimSEE generaliza los componentes del mercado a despachar, todos los sistemas mencionados (SimSEE incluido), así como los problemas explorados en esta tesis, apuntan a despachar óptimamente un mercado marginalista (algo usual en los mercados energéticos) en el cual los actores *abren sus parámetros técnicos de producción*. La última característica es particular de la regulación del mercado uruguayo y algunos pares en la región, que históricamente proyectan al despacho eléctrico como un problema de *ingeniería*, en lugar de como uno *financie-*

ro en el que se cierran ofertas comerciales. La última es la mecánica más frecuente entre los mercados eléctricos de las economías más desarrolladas (Vignolo y Monzón, 2002).

A continuación se entra un poco más en detalle algunas de las técnicas utilizadas por SimSEE. Los autores destacan una primera clasificación de los métodos de solución del problema del despacho hidrotérmico, en aquellos que plantean la función objetivo en forma recursiva en el tiempo y aquellas soluciones que formulan la función objetivo con variables de optimización para todos los pasos de tiempo y resuelven el problema de optimización no por etapas sino considerando todas las etapas a la vez.

En el planteo recursivo, la función objetivo se escribe como la suma del costo incurrido en la etapa actual más el valor del objetivo a partir de la etapa siguiente. Llamando Costo Futuro (CF) de la etapa k al costo de operar el sistema desde esa etapa hasta el final de los tiempos, el planteo recursivo implica escribir $CF(k) = CE(k) + CF(k+1)$, siendo $CE(k)$ el costo directo incurrido en la etapa k.

Este es el tipo de formulación que se utilizó en SimSEE. En el tipo de formulación no recursivo, el problema se plantea sobre todas las etapas en forma simultánea, no apareciendo por tanto el concepto de Costo Futuro como una función de cada etapa.

Las formulaciones recursivas facilitan la consideración de lo estocástico pues en cada etapa tiene sentido considerar conocida la realización de los procesos estocásticos hasta esa etapa y desconocida a partir de ahí. En los planteos no recursivos, dada una etapa, para lograr que todas las realizaciones de los procesos estocásticos coincidan en las etapas anteriores, se deben agregar restricciones adicionales al problema de optimización.

1.2.5.1. Maldición de la dimensionalidad de Bellman

Para la implementación de SimSEE se adoptó la formulación recursiva de la función CF y la solución del problema de optimización por el procedimiento clásico conocido como “Programación Dinámica Estocástica” (SDP, ver Sección 2.2.5), en el que la optimización se lleva a cabo en forma recursiva, asumiendo conocidos los valores de la función CF en la última etapa del horizonte de tiempo estudiado y resolviendo paso a paso desde el futuro hacia el presente. Para este planteo se realiza una discretización de las variables de estado del sistema. El producto cartesiano de las discretizaciones de las variables de estado define una malla de puntos del espacio de estado en los que se calculará CF, en cada paso de tiempo. La cantidad de puntos de esta malla crece exponencialmente con la cantidad de variables de estado. Esto lleva al problema conocido como "Maldición de la dimensionalidad de Bellman" que hace que el algoritmo de SDP no sea aplicable a sistemas con muchas variables de estado. Si bien presenta ese problema, el algoritmo SDP permite considerar adecuadamente los procesos estocásticos y también permite manejar no linealidades en la función de costo de cada etapa, sin la necesidad de asumir propiedades tales coma la convexidad de dicha función.

Utilizando el método conocido como Programación Dinámica Estocástica Dual (SDDP)(Shapiro, 2011)(técnica desarrollada en la Sección 2.2.7), se puede resolver el problema mediante aproximaciones sucesivas que evitan calcular CF sobre una discretización del espacio de estado.

Por esta razón, el método SDDP tiene una ventaja importante frente al SDP en el caso en que no es necesario considerar procesos estocásticos. La ventaja está en que por el planteo del problema, los valores de CF de cada etapa son funciones lineales a tramos y son entonces rápidamente aproximadas por el algoritmo SDDP. Esto elimina la necesidad de discretizar las variables de estado y de calcular el Costo Futuro para cada punto de la discretización evitando así la Maldición de la Dimensionalidad que sufre la SDP. Como contrapartida, en la SDDP no es posible hacer un tratamiento sencillo de los procesos estocásticos y la aproximación de CF en cada etapa que utiliza la SDDP, puede llevar a una diferencia con respecto al óptimo de CF. Esta diferencia se puede producir cuando CF no es convexa.

1.2.5.2. Tratamiento de Lo Estocástico

Por Lo Estocástico nos referimos a la consideración del conjunto de entradas al sistema que son valores que surgen de procesos estocásticos, es decir valores que no podemos asumir que conocemos sino que tenemos una descripción estadística de los mismos. Los ejemplos más importantes de estos procesos en el sistema uruguayo son los caudales de aportes hidráulicos a las represas y la rotura de las máquinas de generación térmica. En esta sección se hace referencia a Lo Estocástico en lo que respecta a su impacto sobre la solución del problema de despacho por SDP.

SDP y Lo Estocástico

En el algoritmo SDP, el tratamiento de lo estocástico es relativamente sencillo. Basta con considerar en cada etapa, en el cálculo CF en cada punto de la malla de cálculo, muchas realizaciones de los procesos estocásticos y calcular así el valor esperado de CF. La linealidad del valor esperado permite calcular el valor esperado de CF usando la fórmula recursiva y resolverlo paso a paso. Para la consideración de las realizaciones de los procesos estocásticos hay por lo menos dos enfoques posibles.

Sorteos de Monte Carlo

La implementación más simple es realizar sorteos (Monte Carlo, ver Sección 2.1) de las diferentes variables aleatorias (cada una con sus funciones de densidad de probabilidad). Este es el método implementado en SimSEE. Un aspecto importante de la implementación del método es que los sorteos que se consideren en una etapa de tiempo deben ser los mismos para el cálculo de todos los puntos del espacio de estado. Esto no es un requisito del método de Monte Carlo, sino un requisito impuesto (sin pérdida de generalidad) para lograr que sin importar el número de sorteos, la función $CF(k)$ (es decir los valores de Bellman de la etapa k) sean monótonos respecto de las variables de estados. Esto es importantísimo pues permite ir subiendo de a poco la cantidad de sorteos utilizados, obteniendo siempre políticas de operación razonables. La razonabilidad está en que al ser CF y sus derivadas monótonas respecto de las variables de estado, cada recurso es valorizado en forma creciente en la medida en que escasea. Si para una etapa del tiempo dada, no se impone el mismo juego de sorteos en el cálculo de cada punto del espacio de estado, podrá ocurrir por ejemplo, que al calcular en un punto correspondiente al "lago lleno" el sorteo que identifica el estado de disponible de una central térmica importante determine que la misma esté no disponible, significando un costo de la etapa elevado por falta de potencia, mientras que en el mismo paso de tiempo, al calcular en un estado con el lago vacío puede resultar de los sorteos que el parque térmico está todo disponible y que no falte potencia lográndose un costo de la etapa, inferior que al caso con más recursos. Claro está que, usando un número suficientemente elevado de sorteos, al hacer los valores esperados se volverá a recuperar la monotonía.

Se recalca este aspecto práctico de la implementación del método de Monte Carlo en la SDP. La no consideración de este detalle práctico tiene consecuencias nefastas sobre los resultados y es a veces una de las razones para intentar solucionar el tratamiento de lo estocástico mediante el producto cartesiano de probabilidades que se comenta a continuación.

Producto cartesiano de probabilidades

Este método consiste en discretizar el espacio de cada una de las variables aleatorias, asignando un peso a cada punto de la discretización y obteniendo el espacio de probabilidad conjunta sobre la malla de puntos definida por el producto cartesiano de las discretizaciones. A cada punto de la malla se le asigna el producto de los pesos de cada una de las variables, en ese punto. Como se puede intuir, este método sufre de una suerte de "Maldición de la Dimensionalidad" al tener que considerar el producto cartesiano de las discretizaciones. Como forma paliativa de este problema, para el cálculo se suele tener alguna heurística que permita clasificar cada punto entre aquellos para los que se espera un apartamiento importante de la solución respecto al promedio de los que no.

Por ejemplo, al considerar la disponibilidad de las máquinas, una heurística puede ser que si la suma de las potencias de las máquinas indisponibles es inferior al 5 % de la potencia de la demanda, es de esperar que se pueda cumplir con el suministro y que no se incurra en costos de falla y por lo tanto para esos puntos se pueda aceptar como representativo el resultado obtenido del cálculo con uno de ellos elegido como el representante del grupo.

1.2.5.3. Importancia de la función de Costo Futuro y sus derivadas

Otro aspecto importante que se quiere destacar del algoritmo SDP clásico es que como resultado se obtiene la función de Costo Futuro (también conocida como función de Bellman) para cada paso de tiempo y para cada punto del espacio de estado (resultado del producto cartesiano de las discretizaciones sobre las variables). Las derivadas de la función de Costo Futuro respecto de cada una de las variables de estado, corresponden a la valorización en el presente que se hace del recurso asociado a cada una de esas variables de estado en cuanto a su aporte a la función de Costo Futuro (los ya mencionados "valores del agua"). Esto permite plantear el problema de despacho en cada etapa por separado, dando valor a los recursos almacenados del sistema y permite por tanto tener una "Política de Operación" que es indispensable para la operación diaria de los embalses.

1.2.5.4. Optimización de la Operación

El operador del sistema debe tomar las decisiones de despacho en todo momento, con el objetivo de lograr satisfacer la demanda en condiciones de calidad pre-establecidas y al menor costo posible. Planteado de esta forma, el problema de obtener el conjunto de reglas que debe seguir el operador, es decir obtener “La Política de Operación”, es un problema de optimización. Como se mencionó, se utiliza la técnica SDP.

Ecuación de Estados y variables de Control del sistema

Supondremos que disponemos de una representación discreta de la evolución del estado del sistema que nos permite calcular el estado al inicio del paso siguiente conocido el estado al inicio del paso actual, y las entradas (las de control y las externas) en el paso actual.

La “ecuación de evolución” del estado, o “ecuación de transición” se podría escribir como:

$$x_{k+1} = f(x_k, u_k, r_k, k) \tag{1.2}$$

Dónde x_k es el vector de estado al inicio del paso k, r_k son las entradas al sistema (aportes hidráulicos, vientos, precios de combustibles, etc.). y u_k son las variables de control del sistema (potencias en las máquinas en cada poste dentro del paso y cualquier otra variable de decisión).

La ecuación 1.2 describe la evolución del vector de estado desde un estado al inicio del paso de tiempo k durante dicho paso de tiempo como consecuencia de la aplicación de las variables de control u_k y de las entradas r_k .

Calcular la política de operación “óptima” es determinar el valor de u_k en función de x_k, r_k y k que minimiza una función de costo objetivo en un horizonte de tiempo especificado.

Generalmente la función de costo a minimizar es el costo esperado de abastecer la demanda, en el horizonte de tiempo.

Optimización dinámica estocástica

En cada paso de tiempo, si conocemos las funciones de control u_k podremos calcular el costo de generación en el paso (si las u_k son las potencias de cada máquina, el costo es la sumatoria de las u_k por las horas del poste y por los costos variables de producción de cada central).

En forma general podremos escribir el costo del paso k, como una función del tipo: $CE(x, u_k, r_k, k)$. Esto es como una función cuyo valor se puede determinar conocido el estado inicial x, las entradas de control u_k, las entradas no controlables r_k y el paso de tiempo en el que estamos k.

Si podemos calcular la función $CE(x, u_k, r_k, k)$, estamos en condiciones de definir el Costo Futuro (CF), como el costo de operar en el sistema desde un estado e instante conocido hasta el fin de los tiempos.

$$CF(x, U_k, R_k, k) = \sum_{j=k}^{\infty} q^{j-k} \cdot CE(x_j, u_j, r_j, j)$$
$$U_k = u_k, u_{k+1}, ...$$
$$R_k = r_k, r_{k+1}, ...$$
(1.3)

Donde $U_k = u_k, u_{k+1}, ...$ es una realización de las entradas de control desde el paso k en adelante y $R_k = r_k, r_{k+1}, ...$ es una realización de las entradas no controladas desde el paso k en adelante. La Eq. 1.4 representa la transición de estados del sistema:

$$x_{j+1} = f(x_j, u_j, r_j, j).$$
(1.4)

El actualizador $q = (\frac{1}{1+\alpha})^{DurPaso/DurAño}$, es para tener en cuenta la tasa de oportunidad del capital.

La tasa de descuento anual es: α y el actualizador así calculado, lleva un valor calculado al fin de un paso a su valor al inicio del paso. Estamos dando prioridad al presente frente al futuro. Aparte de tener una aplicación para reflejar los costos financieros del dinero, el actualizador q tiene importancia en la convergencia del algoritmo que usaremos para el cálculo de la política de operación. Dado que $0 < q < 1$, observando la ecuación del costo futuro, vemos que si el costo de una etapa está acotado, por un valor M, el costo futuro está acotado a la sumatoria

$$CF(x, U_k, R_k, k) \leq \sum_{j=k}^{\infty} q^{j-k} \cdot M = \frac{1}{1-q} \cdot M,$$

dónde hemos utilizado que la serie exponencial del q converge si se cumple $0 < q < 1$.

Si consideramos una tasa de descuento nula, resulta $q = 1$ y no tenemos asegurada una cota de CF. Es más, tenemos asegurado que cualquier error numérico introducido durante el cálculo de CF permanecerá sin amortiguarse.

Observando la sumatoria con la que definimos el costo futuro, vemos que podemos realizar la definición en forma recursiva:

$$CF(x, U_k, R_k, k) = CE(x, u_k, r_k, k) + q \cdot CF(x', U_{k+1}, R_{k+1}, k+1)$$
$$x' = f(x, u_k, r_k, k) \tag{1.5}$$

Donde el valor x' es el resultado de la evolución del estado en la etapa k partiendo del estado x al inicio del paso k y para valores de las entradas u_k y r_k conocidos.

Ahora nos planteamos el problema de conseguir la mejor serie de control del sistema. Esto es, la U_k que haga mínimo el valor esperado de $CF(x, U_k, R_k, k)$ para todas las realizaciones posibles de la serie R_k.

Llamemos $CF(x, k)$ al valor del mínimo costo futuro que es posible obtener usando la mejor política de operación, cuando partimos en la etapa k desde el estado x. Observar que estamos usando dos funciones con el mismo nombre, pero las diferenciamos por los parámetros. Ambas son el costo futuro, pero una es el mínimo costo futuro esperable y la otra es el costo futuro que resultaría con unas series determinadas de las entradas.

Como lo que pase desde $k + 1$ en adelante no puede afectar a lo que pase en la etapa k, podemos separar el problema de la siguiente forma:

$$CF(x, k) = \mathbb{E}\left[\min_{u_k} \{CE(x, u_k, r_k, k) + q \cdot CF(x', k+1)\}\right]_{r_k}$$
$$x' = f(x, u_k, r_k, k)$$
$$U_k = u_k, u_{k+1}, ... = \{u_k, U_{k+1}\}$$
$$R_k = r_k, r_{k+1}, ... = \{r_k, R_{k+1}\} \tag{1.6}$$

Dicho en palabras, para calcular el valor esperado del costo futuro de operar en forma óptima el sistema $CF(x, k)$, tenemos en cuenta que conocemos las entradas de la etapa k, y que según el valor de dichas entradas, podemos calcular el estado al final de la etapa. Entonces, para cada valor de las entradas el costo desde el inicio de la etapa será el costo de la etapa $CE(x, u_k, r_k, k)$, más el valor esperado del costo futuro de operar en forma óptima desde la etapa $k+1$ partiendo del estado al que lleguemos x' multiplicado por el factor de actualización q. Tendremos entonces como solución del problema de minimización (dentro del valor esperado en la fórmula) un $CF^*(x, r_k, k)$ y un $u^*(x, r_k, k)$.

El valor esperado de $CF^*(x, r_k, k)$ será el valor esperado del costo futuro de la operación óptima desde el inicio de la etapa k partiendo del estado x, conocidas las entradas no controlables r_k. El valor esperado en el conjunto de entradas r_k posibles será:

$$CF(x,k) = \mathbb{E}\Bigg[CF^*(x, u_k, r_k, k)\Bigg]_{r_k} \tag{1.7}$$

Los valores $u^*(x, r_k, k)$ definen la política de operación de la etapa k. Conocido el estado de inicio de la etapa, el valor de las entradas no controladas y conocido el tiempo de inicio de la etapa, tenemos el valor para las entradas de control que nos permiten guiar al sistema por la trayectoria de mínimo costo esperado.

Lo que observamos es que conocido $CF(x, k+1)$, podemos resolver el problema de minimización y obtener el $u^*(x, r_k, k)$ y la función $CF(x, k)$. Esto muestra que si para algún paso k del futuro conocemos la función $CF(x, k)$ es posible construir, resolviendo los problemas de optimización de cada etapa, desde la etapa k (en el futuro), etapa por etapa hacia el presente (en sentido inverso del tiempo) y obtener las funciones $u^*(x, r_k, k)$ y $CF(x, j)$ para todas las etapas con $j < k$.

En la práctica lo que se hace es extender el fin del horizonte de tiempo de estudio más allá del tiempo final que realmente nos interesa analizar y comenzamos imponiendo en la última etapa del horizonte así extendido la función $CF(x, j_{\text{última}+1} = 0)$.

Esto es, extendemos el horizonte y nos imaginamos que la función de CF es cero al inicio de la etapa que comenzaría a continuación de la última etapa del horizonte extendido.

En la práctica, el sistema impone restricciones sobre los posibles valores de las variables de control. Esto se expresará como un conjunto de restricciones del tipo $g(x, u, r, k) \leq 0$.

El pseudo del código para el cálculo sería:

Algoritmo 1 Pseudo SDP SimSEE

procedure SDPSIMSEE
    for all $x \in GrillaEstados$ **do**
        $CF(x, k_{última+1} = 0)$
    end for
    for $k \leftarrow k_{última} \ldots 1$ **do**
        for all $x \in GrillaEstados$ **do**
            $CF(x,k) = \mathbb{E}\Big[\min_{u_k} \{CE(x,u_k,r_k,k) + q \cdot CF(x',k+1)\}\Big]_{r_k}$
            con $x' = f(x,u_k,r_k,k)$
            sujeto a $g(x,u,r,k) \leq 0$
        end for
    end for
end procedure

En la práctica, habrá que discretizar el espacio posible de x en un conjunto de puntos sobre los que se calculará $CF(x,k)$ y habrá que elegir la discretización lo suficientemente fina como para que sea posible interpolar los valores intermedios no calculados.

NOTA: En esta sección se respetó la nomenclatura utilizada por los autores de los trabajos relevados. Algunos de los mismos conceptos están representados con distinta nomenclatura en nuestro trabajo.

1.2.6. *Reinforcement Learning* en el Sistema Energético

El *Reinforcement Learning* (RL) se ha utilizado con gran éxito en diversas tareas como en el desarrollo de agentes que aprendan a jugar a juegos como el *Ajedrez*, *Go* y *Shogi* (Silver et al. 2018); videojuegos sencillos como los de la consola *Atari* (Mnih et al. 2013); e incluso videojuegos sumamente complejos como *DOTA2* (OpenAI et al. 2019) y *Starcraft 2* (Vinyals et al. 2019), en todos estos casos superando el desempeño humano. También se ha utilizado en diversas otras tareas de optimización y de toma de decisión multi-instancia. Comentaremos ahora el uso que se le ha dado al RL en el sistema energético.

El RL se ha utilizado para resolver varias tareas relacionadas al sistema energético. Glavic et al. 2017 y Vázquez-Canteli y Nagy, 2019 hacen resúmenes de varias de estas tareas y de varios de los trabajos que emplean RL para resolverlas. A continuación se mencionan algunos de los problemas relacionados, relevantes para nuestro trabajo, que fueron resueltos utilizando RL. Los mismos serán desarrollados a fondo en la Sección 2.4 luego de haber introducido la temática del RL. Estos trabajos son los siguientes:

- **RL para el Despacho Económico (Jasmin et al. 2011):** los autores utilizan técnicas de RL para encontrar el despacho energético óptimo. Similar al problema que se resuelve en este trabajo, pero con la diferencia de que ellos cuentan solamente con unidades térmicas, y además se está resolviendo el despacho óptimo de un solo paso de tiempo y no un despacho a largo plazo en varias etapas. Su complejidad radica en que tienen una gran cantidad de generadores térmicos (20 o más), y además los mismos cuentan con funciones de costos no lineales.
- **RL para el Almacenamiento Energético (Henze y Dodier, 2003):** los autores utilizan técnicas de RL para determinar el almacenamiento energético óptimo. Como se mencionó en la Sección 1.1 cuando se introdujo el problema del despacho óptimo que se intenta resolver en el presente trabajo, el mismo puede ser dicho en otras palabras como "almacenar agua en los embalses de manera óptima" sujeto a la aleatoriedad de los aportes hídricos futuros. Los autores de este trabajo relacionado hacen lo análogo, pero determinando el almacenamiento óptimo de energía dentro de baterías conectadas a paneles solares, sujeto a la aleatoriedad de la irradiación solar.
- ***Demand Response* - Vehículos Eléctricos (Dusparic et al. 2013):** los autores utilizan técnicas de RL para optimizar la carga de vehículos eléctricos bajo varias restricciones y minimizando costos. Este trabajo es un ejemplo de las técnicas denominadas *Demand Response*, cuyo objetivo es el de trasladar el uso de energía a períodos de baja demanda o a períodos de alta disponibilidad de energía renovable para abaratar costos.
- **Sistemas CVAA (Li et al. 2015):-** los autores utilizan técnicas de RL para optimizar el uso de sistemas CVAA (Calefacción, Ventilación y Aire Acondicionado). El uso de estos sistemas (en particular en edificios de oficinas) suele ser una gran parte de la demanda energética. Con el objetivo de llevar un área de un edificio a un determinado nivel de confort, los autores determinan una política de operación óptima del sistema CVAA mediante el control de la temperatura y la velocidad del aire, de manera de minimizar el consumo energético.

1.3. Motivación y Aproximación Metodológica

De acuerdo a lo presentado en Capítulo 1.1, el problema de fondo de este estudio es el Despacho Hidrotérmico Óptimo y Estocástico: un problema de planificación que busca la optimización matemática en un campo de naturaleza económica, a saber: el ajuste en todo momento de un período dado entre la demanda y la producción/oferta de energía. El problema de fondo es de notoria importancia práctica, especialmente en Uruguay, por la reciente diversificación de su matriz energética.

Históricamente, el problema de referencia se ha resuelto mediante una combinación de herramientas clásicas de optimización, entre las que destacamos Programación Matemática y Programación Dinámica. La programación matemática presenta muy buena escalabilidad ante la cantidad y diversidad de componentes a integrar en el modelo, aunque esa eficiencia decae notoriamente al buscar capturar la aleatoriedad de algunos parámetros. Por el contrario, la programación dinámica clásica captura eficientemente el modelado estocástico, pero escala pobremente ante la cantidad de estados a modelar. Lo último se debe principalmente al algoritmo de referencia de Bellman, un método recursivo, que requiere tantas operaciones como combinaciones de estados y que además recién logra una solución al final.

En este contexto, el *Reinforcement Learning* se muestra como una alternativa promisoria, en la medida que permite capturar el modelo del sistema y su aleatoriedad en un marco conceptual similar al de la programación dinámica, pero construye sus soluciones en base a algoritmos iterativos, i.e., mediante aproximaciones sucesivas, con lo cual, se puede tener una solución factible de buena calidad en un período de tiempo mucho menor.

El objetivo de este estudio es explorar el desempeño de las técnicas de *Reinforcement Learning* aplicadas al problema de la Planificación del Despacho Energético. Las instancias de referencia están inspiradas en particulares del parque generador nacional y los eventos exógenos que lo afectan, como la variación en los aportes por lluvias. La estrategia elegida para ponderar las bondades de esta técnica pasa por la evaluación experimental sobre un conjunto de instancias representativo de la realidad nacional y la comparación de sus resultados con los conseguidos mediante el uso de otras técnicas de referencia.

Como compromiso entre el esfuerzo esperado en una tesis y el cumplimiento del objetivo académico antes mencionado, el estudio no busca ser exhaustivo en la diversidad de los componentes ni en el nivel de detalle de sus modelos de referencia. El foco se pone en cambio en evaluar cómo escala la técnica con la cantidad de componentes/estados. Los resultados conseguidos son promisorios en general, mostrando una performance muy alta en algunas pruebas. Incluso cuando la técnica no ha superado a alternativas clásicas, se ha identificado la causa raíz de la diferencia.

1.4. Estructura de la Tesis

El presente trabajo comienza por presentar los antecedentes, y la definición del problema general que se abarca en esta tesis (Despacho Hidro-térmico Óptimo), al principio del Capítulo 2. El Capítulo 2 continúa haciendo una introducción teórica de los conceptos involucrados, empezando por las técnicas tradicionales de optimización, y siguiendo con una introducción a la técnica de *Reinforcement Learning* (RL). El Capítulo culmina haciendo un estudio del estado del arte y trabajos relacionados que aplican RL al sistema energético.

El trabajo está estructurado en iteraciones incrementales, en donde cada una resuelve una instancia del problema y agrega alguna complejidad sobre la anterior. Se realizaron dos iteraciones en el alcance de este trabajo, la primera presentada en el Capítulo 3 y la segunda en el Capítulo 4. La estructura de estos dos Capítulos es la misma y es la siguiente: comienzan por definir la instancia del problema a resolver en la iteración, luego se obtienen valores de interés para tener como referencia; ya sean cotas y/o soluciones por técnicas de optimización tradicionales, luego se procede a hacer la solución mediante RL, y por último se analizan y comparan los resultados.

El documento culmina en el Capítulo 5 con un resumen de las conclusiones obtenidas y un lineamiento de posibles trabajos futuros.

Capítulo 2

Marco Teórico

Este capítulo proporciona un marco teórico de las distintas técnicas utilizadas a lo largo del trabajo. Comienza enfocándose principalmente en las técnicas de optimización tradicionales, luego pone el foco en las técnicas de *Reinforcement Learning*, y por último, entra en detalle en algunos trabajos relacionados y aplicaciones de *Reinforcement Learning* al sistema energético.

2.1. Métodos de Monte Carlo

Esta sección realiza una introducción a conceptos básicos de los Métodos de Monte Carlo. Aplicaremos estos conceptos para validar que nuestra muestra de instancias de problema es representativa, algo importante para validar y/o entrenar los métodos de solución utilizados en este trabajo. Entre las fuentes de referencia para este contenido se ha usado el curso FING-UdelaR (Cancela, 2022), referencia base: (Fishman, 1996). Recomendamos referirse a ese material como complemento a la información aquí presentada.

El método Monte Carlo es una técnica estocástica que estima magnitudes complejas o numéricamente costosas a partir de una muestra de experimentos/simulaciones estadísticamente representativos, que se implementa haciendo uso de una computadora.

Los métodos de Monte Carlo son indisociables de la probabilidad y estadística, por lo que sus orígenes históricos se remontan a los de esa disciplina (siglos XVI y XVII), cuando se comienzan a crear abstracciones matemáticas para cuantificar el azar en ciertos juegos o fenómenos, para los que intuitivamente se identificaban tendencias sobre resultados de ciertos experimentos al repetirlos muchas veces.

La motivación práctica de la probabilidad es hacer uso de esas abstracciones (e.g., variables aleatorias, distribuciones de probabilidad, etc.) para inferir resultados de fenómenos inciertos, complejos y costosos de reproducir en la cantidad necesaria para estimarlos a través de estadísticos. Considerar que hasta hace pocas décadas, cualquier experimento o cálculo debía hacerse con intervención manual, lo que limitaba la escalabilidad y condicionaba el costo de cualquier proceso no-analítico. Los métodos de Monte Carlo por el contrario, se apoyan en el poder de cómputo existente para aproximar mediante promedios de experimentos automatizados lo que sería el resultado de derivaciones analíticas o incluso para estimar valores cuando no existen tales expresiones.

Las ideas básicas de la técnica tuvieron sus primeras aplicaciones prácticas en el marco del Proyecto Manhattan para la creación de las armas nucleares, concretamente para estimar la probabilidad de que los neutrones resultantes de la fisión de un átomo de Uranio-235 colisionaran con el núcleo de otro átomo dentro de la estructura cristalina del metal. El nombre del método y su generalización como marco para resolver problemas de múltiple tipo se atribuye a Stanislaw Ulam y a John von Neumann en 1946, dos integrantes del equipo en el referido proyecto.

Monte Carlo se soporta en la ley de los grandes números, para lo que en general se plantea el problema de forma tal que el resultado deseado se corresponda con el valor esperado de alguna variable aleatoria, que luego será aproximada por la media.

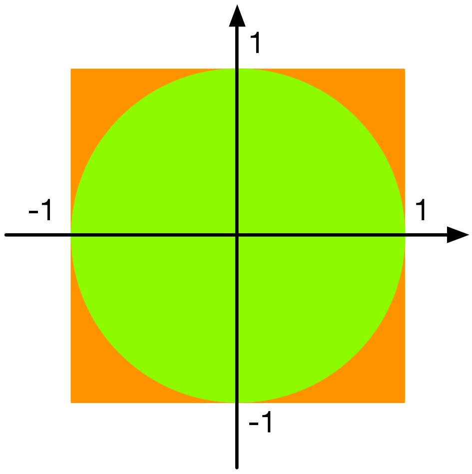
Teorema 1. *Ley Débil de los Grandes Números. Si $\{X_i\}$ es una sucesión de variables aleatorias (vv.aa.) independientes e idénticamente distribuidas (i.i.d.) de valor esperado $E[X_i] = \mu$, entonces el promedio $\overline{X_n} = \dfrac{X_1 + \cdots + X_n}{n}$ converge a μ: $\lim_{n\to\infty} \mathrm{P}\left(\left|\overline{X}_n - \mu\right| < \varepsilon\right) = 1$ para todo $\epsilon > 0$.*

Monte Carlo se concibió originalmente como estimador estadístico de magnitudes difíciles de calcular analíticamente, pero es muy común su aplicación en problemas determinísticos, en particular en el cálculo de áreas mediante integrales. Supongamos que queremos estimar el valor de π. Hay múltiples expresiones analíticas en forma de límites/series para calcular ese valor que pueden ser usadas para aproximarlo; algunas de convergencia muy rápida, como la serie de Ramanujan:

$$\frac{1}{\pi} = \sum_{n=0}^{\infty} \binom{2n}{n}^3 \frac{42n+5}{2^{12n+4}},$$

cuyo error relativo con cada nuevo término evoluciona como: 1.86e^{-2}, 2.28e^{-4},

$3.04e^{-6}$, $4.20e^{-8}$, esto es, se gana aproximadamente dos dígitos decimales con cada nuevo término. Llegar a una expresión como la anterior no siempre es posible analíticamente, y aun cuando sí lo fuera, requiere en general un esfuerzo de desarrollo teórico considerable. Como alternativa, podríamos usar simplemente Monte Carlo, diseñando un experimento cuyo valor esperado sea π.



Si (x_i, y_i) fueran muestras i.i.d. de pares de vv.aa. uniformes en [-1,1], cada par correspondería a un punto al azar en el cuadrado $[-1, 1]\times[-1, 1]$, que tiene área 4 (límites anaranjado en la figura a la izquierda).

El área del círculo inscripto (radio 1, destacado en verde en la figura) es π, así que la probabilidad de que un punto al azar del primer cuadrado esté además dentro del círculo es $\pi/4$.

Consideramos la variable aleatoria indicatriz del evento "estar en el círculo", que notamos por $\mathbb{1}_C$: la v.a. binaria que toma valor 1 cuando el evento se produce, i.e., cuando $x_i^2 + y_i^2 \leq 1$ y el valor 0 en otro caso. Observar que $\mathbb{1}_C$ es Bernoulli de parámetro (i.e. probabilidad de suceso) $p = \pi/4$. Como la esperanza de una v.a. Bernoulli coincide con su parámetro, el valor esperado $E[\mathbb{1}_C] = \pi/4$, así que –por la ley de los grandes números– la media $\overline{X_n}$ en una muestra de n vv.aa. $\mathbb{1}_C$ también tiende a $\pi/4$, de donde $4 \times \lim_{n\to\infty} \overline{X_n} \to \pi$. El pseudocódigo para implementar este estimador es el de Algoritmo-2.

Algoritmo 2 Estimador Monte Carlo para Π

procedure ESTIMAPI($nsamp$) ▷ Estima π sobre $nsamp$ muestras
    $suma \leftarrow 0$;
    for $i \leftarrow 1 \ldots nsamp$ **do**
        $x_i \leftarrow 2 \times (rand() - 0.5)$;
        $y_i \leftarrow 2 \times (rand() - 0.5)$;
        if $x_i^2 + y_i^2 \leq 1$ **then**
            $suma = suma + 1$;
        end if
    end for
    return $4 \times suma/nsamp$; ▷ Retorna el estimador $\pi \approx 4 \cdot \overline{X_n}$
end procedure

Las líneas Algoritmo-2.4 y Algoritmo-2.5 presumen la existencia de una función $rand()$ que genera números aleatorios e independientes de distribución uniforme en [0,1]. La generación de números aleatorios es un área de investigación en sí misma que recurre a multiplicidad de aproximaciones. Entre ellas se incluye: el uso de enormes bibliotecas/repositorios con resultados de experimentos reales, componentes de hardware especializado a tales efectos y el uso de algoritmos pseudoaleatorios, esto es, procedimientos exactos que parecen producir números al azar, como la resolución de sistemas dinámicos caóticos o alguna forma de relación de congruencia $r_{i+1} = (a \cdot r_i + b) \mod m$, que al poder asignar una semilla r_0 permite reproducir exactamente (i.e. en forma determinística) una secuencia de números a-priori percibida como aleatoria.

Origen aparte, lo mínimo necesario es alguna librería que provea números aleatorios con distribución uniforme continua, ya que a partir de ellos se pueden construir números aleatorios de cualquier otra distribución, siendo las técnicas más comunes *Roulette Wheel* para vv.aa. discretas y *Inverse Transformation Method* para las continuas. El primero de los métodos es el más simple. Supongamos que una v.a. discreta X toma tres valores posibles a, b y c, con probabilidades respectivas 30 %, 50 % y 20 %. Si U es uniforme en [0,1] podemos generar muestras de X asignándole valores según el valor de U de esta forma: $X = a$ cuando U <0.3, $X = b$ cuando 0.3≤ U ≤0.8 y $X = c$ cuando U >0.8.

Para generar muestras de un v.a. X continua con función de densidad $f(x)$ aplicamos *Inverse Transformation Method*. Comenzamos por calcular la función de distribución $F(z) = \int_{-\infty}^{x} f(t)dt$. Recordar que: $F : \mathbb{R} \to [0, 1]$, $\lim_{x\to-\infty} F(x) = 0$, $\lim_{x\to\infty} F(x) = 1$, $F'(x) = f(x)$ y asumamos por simplicidad que $f(x) > 0$, de donde F es creciente e invertible. Dada una muestra u de una v.a. U uniforme en [0,1] se resuelve la ecuación $F(x) = u$ y se toma x como muestra de X. Como ejemplo tomamos la distribución exponencial de parámetro $\lambda > 0$, que vale $f_\lambda(x) = \lambda e^{-\lambda x}$ cuando $x \geq 0$ y cero en otro caso. La distribución resulta:

$$F(x) = \begin{cases} 0 & x < 0 \\ 1 - e^{-\lambda x} & x \geq 0 \end{cases},$$

así que dado $u \in [0, 1]$ se resuelve $u = 1 - e^{-\lambda x}$, para llegar a $x = -\frac{1}{\lambda}\ln(1 - u)$. Al ser u una muestra de una uniforme en [0,1], $1 - u$ tiene esa misma distribución y la expresión puede simplificarse a $x = -\frac{1}{\lambda}\ln u$. Observar que aunque la distribución exponencial no cumple $f(x) > 0$ cuando $x < 0$, la técnica funciona de todos modos. Como fuente complementaria sobre la generación y transformación de números aleatorios así como sobre su uso en simulaciones recomendamos referirse a (Winston y Goldberg, 2004).

Cuando se usa como generador el algoritmo `rng(1,'v5uniform')` de `MATLAB R2015a (8.5.0)`, el error relativo de Algoritmo-2 cada cinco millones de nuevos puntos (hasta veinte) evoluciona como: 8.92e^{-5}, 2.41e^{-5}, 2.52e^{-6}, 1.43e^{-5}. Se observa una tendencia a la baja aunque no monótona, ya que el error para $nsamp$=15M es menor que para $nsamp$=20M. El resultado es bastante menos impresionante que el de la fórmula de Ramanujan. Existen técnicas para mejorar la performance del método que escapan al alcance y objetivos de este documento. Por el momento, adelantamos que cuando se cuenta con formulaciones derivadas de expresiones analíticas determinísticas, éstas suelen desempeñar mejor en *dimensiones bajas*. A medida que aumenta la dimensionalidad, Monte Carlo se vuelve una mejor opción. Para entender cómo evoluciona el error en estos métodos recurrimos a otro resultado fundamental de la probabilidad y estadística, el Teorema Central del Límite.

Teorema 2. *Límite Central. Sea $\{X_i\}$ una sucesión de vv.aa. i.i.d. de valor esperado $E[X_i] = \mu$ y varianza $Var[X_i] = \sigma^2 < \infty$. Entonces $Z_n = \dfrac{\sum_{i=1}^n X_i - n\mu}{\sigma\sqrt{n}}$ = $\dfrac{\overline{X_n} - \mu}{\sigma/\sqrt{n}}$ tiende a tener distribución normal $N(0,1)$, esto es: $\lim_{n\to\infty} \mathrm{P}\left(Z_n \leq z\right) = \Phi(z) = \dfrac{1}{\sqrt{2\pi}} \displaystyle\int_{-\infty}^{z} e^{-\frac{x^2}{2}} dx$.*

El Teorema Central no asume ninguna distribución en particular; solamente requiere varianza acotada. Para comprender qué tan potente es, imaginemos que las variables X_i son $U_{[-1,1]}$ (uniformes en [-1,1]). La densidad de X_1 es entonces $f_1(x) = \frac{1}{2} \cdot \mathbb{1}_{|x|\leq 1}$, esto es, $\frac{1}{2}$ cuando $|x| \leq 1$ y 0 en otro caso. Para calcular la de las sumas recurriremos a la siguiente propiedad de las distribuciones.

Definición 1. *Sean $f(t)$ y $g(t)$ dos funciones reales e integrables en $\mathbb{R}$. La convolución de f y g (denotada por $f * g$) es una nueva función $h(t) = (f * g)(t)$, cuyos valores corresponden al resultado de $(f * g)(t) = \displaystyle\int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau$.*

Propiedad 1. *Densidad de la Suma. Sean X e Y dos vv.aa. continuas independientes de densidades $f_X(t)$ y $f_Y(t)$ conocidas. Se cumple que la densidad de la v.a. suma de ambas, i.e., $Z = X + Y$ tiene densidad igual a $f_X * f_Y$.*

Por la propiedad anterior, la densidad de $X_1 + X_2$ es $f_2(x) = (f_1 * f_1)(x) = (\frac{1}{2} - \frac{|x|}{4}) \cdot \mathbb{1}_{|x| \leq 2}$. Asimismo, la densidad de $X_1 + X_2 + X_3 = (X_1 + X_2) + X_3$ es $f_3(x) = (f_2 * f_1)(x) = \frac{(3-x^2)}{8} \cdot \mathbb{1}_{|x| \leq 1} + \frac{(|x|-3)^2}{16} \cdot \mathbb{1}_{1<|x| \leq 3}$. Para cuantificar su evolución respecto a la Distribución Normal hay que igualar μ y σ^2.

Por la linealidad de la esperanza, el valor esperado de la suma de los X_n es cero, ya que $E[X_i] = (-1+1)/2 = 0$ por ser uniforme en [-1,1] y $E[X_1 + \cdots + X_n] = E[X_1] + \cdots + E[X_n] = 0$. En lo que respecta a la varianza de la suma $Var[X_1 + \cdots + X_n] = Var[X_1] + \cdots + Var[X_n]$ porque las X_i son independientes. Como además, $Var[X_i] = \sigma^2 = \frac{1}{12}(1-(-1))^2 = \frac{1}{3}$ (todas son uniformes en [-1,1]), $Var[X_1 + \cdots + X_n] = n\sigma^2 = \frac{n}{3}$. Por simplicidad, evaluamos cómo evoluciona el límite de la suma según Teorema-2 comparando con $N(0, \frac{n}{3})$.

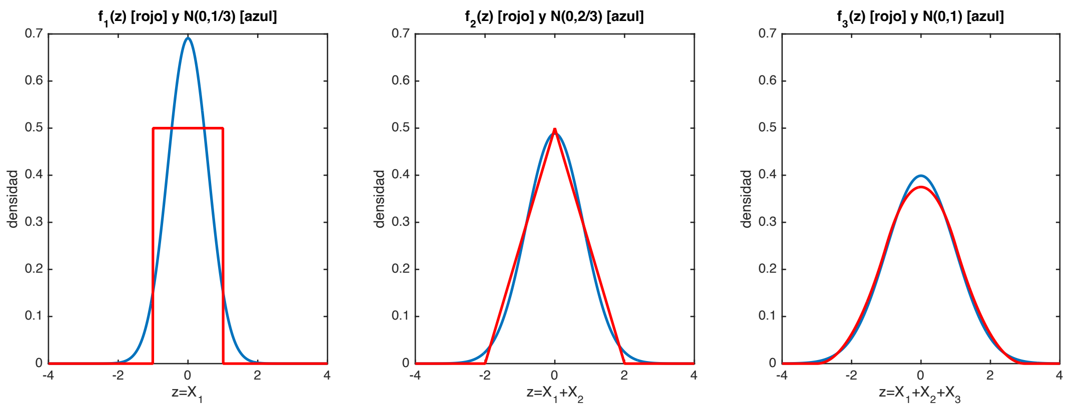


Figura 2.1: Comparación de las funciones de la densidad de sumas de variables $U_{[-1,1]}$ respecto a la normal de la misma media y varianza.

La Figura-2.1 presenta la evolución de la densidad de la suma de las variables uniformes para $n = 1, 2, 3$. En cada caso se compara con la Normal de los mismos parámetros, que es $N(0, \frac{n}{3})$. La gráfica para $n = 4$ se omitió porque la diferencia entre distribuciones era imperceptible. En general, la convergencia en el Teorema-2 es tan rápida que se prescinde del límite para usar directamente la expresión $P(Z_n \leq z) \approx \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{x^2}{2}} dx$, dado que el valor n necesario para alcanzar cierto intervalo de confianza suele ser mucho más alto que la cantidad de convoluciones necesarias para aproximar la $N(0, 1)$ con error razonable.

Si lo que estamos buscando es estimar el valor esperado de cierta v.a. con promedios, el Teorema Central del Límite nos permite calcular cómo evoluciona el error con la cantidad n de muestras (experimentos). Logramos que la diferencia entre promedio y valor esperado esté acotada por err con cierta probabilidad o nivel de confianza NC cuando $P(|\overline{X_n} - \mu| \leq err) = NC$, que se puede reescribir como $P(-err \leq \overline{X_n} - \mu \leq err) = P(-\frac{err}{\sigma/\sqrt{n}} \leq Z_n \leq \frac{err}{\sigma/\sqrt{n}}) \approx \Phi(\frac{err}{\sigma/\sqrt{n}}) - \Phi(-\frac{err}{\sigma/\sqrt{n}})$. Siendo $\{X_i\}$ vv.aa. i.i.d. de media μ y varianza σ^2, el error es:

$$err = \left|\frac{X_1 + \cdots + X_n}{n} - \mu\right| \approx \frac{Z_{NC} \cdot \sigma}{\sqrt{n}}, \tag{2.1}$$

donde dado un nivel de confianza objetivo NC (e.g. 90 %=0.9), $Z_{NC} = z$ tal que $\frac{1}{\sqrt{2\pi}} \int_{-z}^{z} e^{-x^2/2} dx = NC$. (e.g. $Z_{0.9}$ =1.645). La regla general es que hay que cuadruplicar la cantidad de muestras (experimentos) para bajar a la mitad el error a nivel de confianza dado.

Buscamos ahora la cantidad de muestras necesarias para estimar π mediante Algoritmo-2 con NC=90 % y error relativo $1e^{-4}$, que corresponde a err =3.1416e^{-4}. Recordamos que $\mathbb{1}_C$ es una v.a. Bernoulli de valor esperado $p = \frac{\pi}{4}$, así que su varianza es $\sigma^2 = p(1-p) = \frac{\pi}{4}(1 - \frac{\pi}{4})$, de donde σ ≈0.4105. El número de experimentos para alcanzar esos valores es 4,620,153. Se recuerda que el error registrado usando cinco millones de experimentos fue 8.92e^{-5}, ligeramente por debajo del $1e^{-4}$ buscado.

La idea implementada en Algoritmo-2 puede generalizarse para el cálculo de integrales. Cuando se tiene una dimensión (i.e. se integra en $\mathbb{R}^1$) los métodos de cuadratura alcanzan errores muy bajos en pocas iteraciones. Supongamos que se necesitan N intervalos para lograr cierto error al integrar una función $f(x)$ en [0,1]. La función puede extenderse a $\mathbb{R}^2$ usando $f(x+y)$ y podríamos integrarla en [0,1]×[0,1] con el mismo error usando un cuadriculado de N^2 celdas. Generalizando la idea concluimos que lo esperable usando esas técnicas en $\mathbb{R}^m$ es necesitar N^m hipercubos para sostener un nivel de error dado, esto es, la complejidad es exponencial en la dimensión. La integración usando Monte Carlo sigue teniendo la Ecuación (2.1) como referencia de las muestras necesarias para lograr cierto error. Por eso las variantes de la técnica son el estándar de facto para estimar soluciones en problemas de muchas dimensiones.

Para realizar simulaciones desarrollamos un generador de aportes hidrológicos sintéticos. Tanto para validar la Programación Dinámica y el *Reinforcement Learning* (RL), como para el entrenamiento del algoritmo de RL, se utilizan muestras de datos generadas por ese generador. Similar al ejemplo sencillo anterior de cálculo de π, utilizamos el Teorema Central del Límite para determinar el tamaño necesario para que las muestras sean representativas de la distribución de los datos, distribución que no es conocida, sino que es estimada a partir de medias utilizando la Ley de los Grandes Números. Veremos todos los detalles en el Capítulo 3.

2.2. Técnicas Tradicionales de Optimización

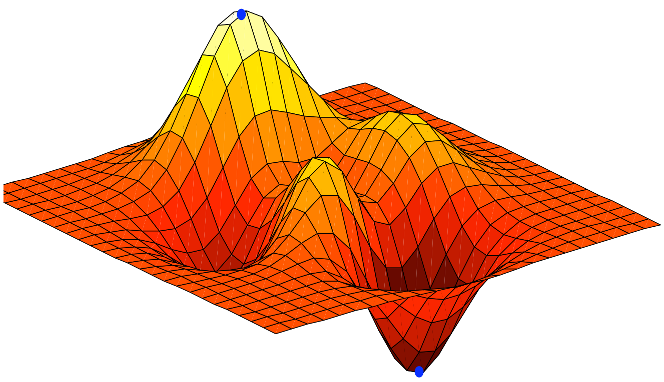
Esta sección realiza una introducción a conceptos básicos de optimización. Como fuentes de referencia para este contenido se han usado los cursos FING-UdelaR de la siguiente lista: Moscatelli, 2022, Risso et al. 2020, Iturriaga y Nesmachnow, 2022, Risso y Rodríguez-Bocca, 2021, Testuri, 2019 y Testuri, 2022, cuyas referencias base son respectivamente las siguientes: Hillier et al. 1991, Ribeiro y Hansen, 2001, Goldberg, 1989, Boyd y Vandenberghe, 2004, Wolsey y Nemhauser, 1988, y Birge y Louveaux, 2011. Recomendamos referirse al material de esos cursos como complemento a la información aquí presentada.

2.2.1. Problema de Optimización General

En términos generales, el objetivo de la Optimización puede definirse como *la gestión eficiente de recursos escasos mediante un enfoque cuantitativo*. Existe un objetivo que se busca maximizar o minimizar. El problema depende de un conjunto de variables de control y pueden existir restricciones a cumplir como condición para que una solución sea considerada tal.

A continuación definimos el problema general de optimización:

Definición 2. *Dado un dominio X para un conjunto de n variables (e.g. $X = \mathbb{R}^n$ o $X = \mathbb{N}^n$), una función objetivo $f : X \to \mathbb{R}$, y un conjunto $F \subseteq X$ de soluciones técnicamente viables para el problema, el planteo general de optimización (G) consiste en hallar $\bar{x} \in F$, tal que $f(\bar{x})$ es el mínimo (o máximo) valor posible de $f()$.*



$$(G) \begin{cases} \min f(x) \\ x \in F \end{cases}$$

Figura 2.2: Representación del problema general de optimización.

El formato anterior es el más general, y permite expresar el problema aun cuando no se conozcan expresiones analíticas para algunos de los objetos. Supongamos que nuestro problema fuera coordinar las frecuencias y fases de los semáforos de una avenida, con fin de conseguir el tránsito más fluido posible en ella y su vecindad. Podríamos elegir minimizar el tiempo medio entre que los vehículos ingresan y abandonan la avenida, y pedir además que los largos de las filas de ingreso a esa avenida desde calles conexas no superen ciertos umbrales con cierta probabilidad. El problema (G) está bien definido (si tenemos datos históricos de tráfico), pero no es posible usar una expresión analítica para calcular $f()$ y F. No obstante, podríamos simular sobre un conjunto grande de casos independientes (Monte Carlo) y usar los promedios resultantes como estimadores.

2.2.2. Programación Matemática

Definición 3. *Diremos que un problema de optimización (G) es de programación matemática (P), si la región factible F puede expresarse como la intersección de un conjunto de restricciones algebraicas en la variable x (combinado con el dominio X de las variables) y la función $f()$ también tiene expresión analítica.*

$$(P)\begin{cases} \min f(x) \\ g_i(x) \leq 0, \quad 1 \leq i \leq p, \\ h_j(x) = 0, \quad 1 \leq j \leq q, \\ x \in X \end{cases}$$

El problema (P) tiene n variables y $m = p+q$ restricciones. Cuando no se especifica X, se asume que $X \subseteq \mathbb{R}^n$.

2.2.3. Clasificación de Métodos de optimización

Clasificamos los métodos para resolver los problemas de optimización en las siguientes clases no excluyentes:

- **Directos:** No conseguimos una solución al problema hasta terminar. Llegan a la solución en un número finito de pasos para ciertos problemas (asumiendo precisión absoluta).
- **Reductivos:** Se construye una sucesión finita de instancias equivalentes hasta alcanzar una trivial (son directos).
- **Recursivos:** La solución se construye incrementalmente, a partir de soluciones de instancias más pequeñas del mismo problema (también son métodos directos).

- **Iterativos:** Se generan aproximaciones sucesivas para la solución empezando desde una estimación inicial. En general, el óptimo no se alcanza en una cantidad finita de pasos.
- **Estocásticos:** Algunos algoritmos iterativos no son determinísticos, y su convergencia es estadística.

Un ejemplo clásico de método reductivo lo constituye la Escalerización Gaussiana para resolver sistemas de ecuaciones lineales, que basándose en operaciones que no cambian la solución (i.e., multiplicar una ecuación por un escalar no nulo; intercambiar la posición de dos ecuaciones y sumar a una ecuación un múltiplo de otra), simplifican el problema hasta llevarlo a uno equivalente cuya solución es trivial. Como ejemplo de método recursivo veremos la Programación Dinámica (ver Sec. 2.2.5).

Los métodos directos son muy usados para resolver problemas de tamaño reducido y baja complejidad computacional. Esto se explica por el hecho que no arrojan solución hasta no terminar completamente. En problemas más complejos y/o muy grandes, se prefiere en general el uso de métodos iterativos, dado que ellos permiten conseguir soluciones de error acotado a un costo computacional menor.

Para finalizar, se destaca la eficiencia de algunos algoritmos estocásticos para resolver problemas computacionalmente complejos. Como ejemplos relevantes destacamos las metaheurísticas (Risso et al. 2020, Ribeiro y Hansen, 2001) y el Descenso por Gradiente Estocástico (ver Sec. 2.2.6).

2.2.4. Técnicas *Greedy*

Hacemos un breve recordatorio sobre las técnicas *greedy* (Black, 1999), ya que se utilizan a lo largo de este trabajo.

Un algoritmo *greedy* es aquel que siempre toma la mejor solución inmediata o local en cada paso para a la hora de buscar la solución a un problema. Los algoritmos *greedy* pueden encontrar la solución global óptima en algunos problemas, pero por lo general obtienen soluciones que no son óptimas. Uno de los usos que le damos a estas técnicas en el presente trabajo es para obtener soluciones de referencia para poder comparar resultados con los obtenidos por otras técnicas.

2.2.5. Método de Programación Dinámica

Muchos de los problemas tratables en forma directa cumplen el *Principio de Optimalidad* (Bellman), que dice: "Dada una secuencia óptima de decisiones, toda subsecuencia de ella es a su vez óptima para el subproblema correspondiente". En el problema del camino más corto en una red (costos positivos), el Principio de Optimalidad representa que si se conoce el camino más corto entre dos nodos (A y C en Figura 2.3), y el nodo B pertenece al mismo, no hay un camino más corto para conectar a A y B (o B y C), que el subcamino correspondiente en el original. Observar que de existir $\overline{AB}$ más corto que el AB original, podría construirse un camino $\overline{AC}$ de menor costo total concatenando $\overline{AB}$ y BC.

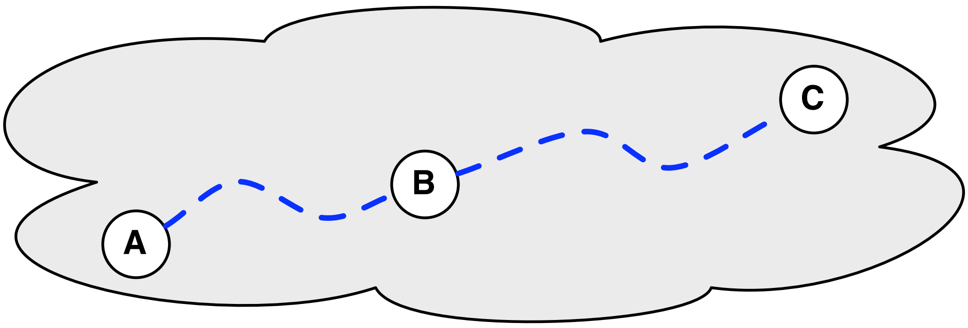


Figura 2.3: Principio de Optimalidad de Bellman.

El Método de Programación Dinámica (Bellman 1940-1953) (Bellman, 1957a) es un *método recursivo* para resolver problemas complejos mediante su descomposición en problemas más simples. Durante la recursión se almacenan los resultados encontrados al momento, y luego se usan en los problemas más complejos. Deben existir "pasos base", instancias de tamaño reducido para las que la solución es trivial o conocida. Se usa en problemas de: matemáticas, *management*, economía, *computer science*, *machine learning* y bioinformática, por nombrar algunos.

El uso de este método requiere una estructura particular del problema: con un conjunto discreto de etapas y estados en cada etapa. El problema debe cumplir el Principio de Optimalidad entre sus etapas, lo que permite un planteo *greedy*. La técnica puede extenderse a la resolución de problemas estocásticos cuando el proceso de referencia es Markoviano (Bellman, 1957b).

Suponemos que el problema consta de T etapas $(1, \ldots, T)$, que en cada etapa hay n estados $(1, \ldots, n$, los mismos por simplicidad) y que conocemos la función de costo final $f(s, T)$, con $s = 1, \ldots, n$, siendo $f(s, t)$ el costo en el estado s y tiempo t para $s = 1, \ldots, n$ y $t = 1, \ldots, T$.

Se conoce por último la función de costo $c(s, j, t+1)$, con el costo incurrido por pasar del estado s en t, al estado j en $t+1$.

La recursión queda: $f(s,t) = \text{mín}_{1 \le j \le n}\{c(s,j,t+1) + f(j,t+1)\}$. El valor óptimo es $f(s,1)$ para el(los) estado(s) inicial(es). La estructura de atrás hacia adelante antes presentada (i.e., *backwards*) es la más común, pero también hay ejemplos en los que la recursión se plantea desde el principio hacia al final. Algunos de estos métodos *forward* se mencionan más adelante en este trabajo como es el caso de la técnica SDDP (Shapiro, 2011) desarrollada en la Sección 2.2.7.

Suele ser bastante eficiente y constituye una de las pocas excepciones de interés dentro de los problemas combinatorios, en las que hay algoritmos eficientes para resolverlo. Enriqueciendo el número de estados se pueden modelar problemas más complejos (a costa de la eficiencia).

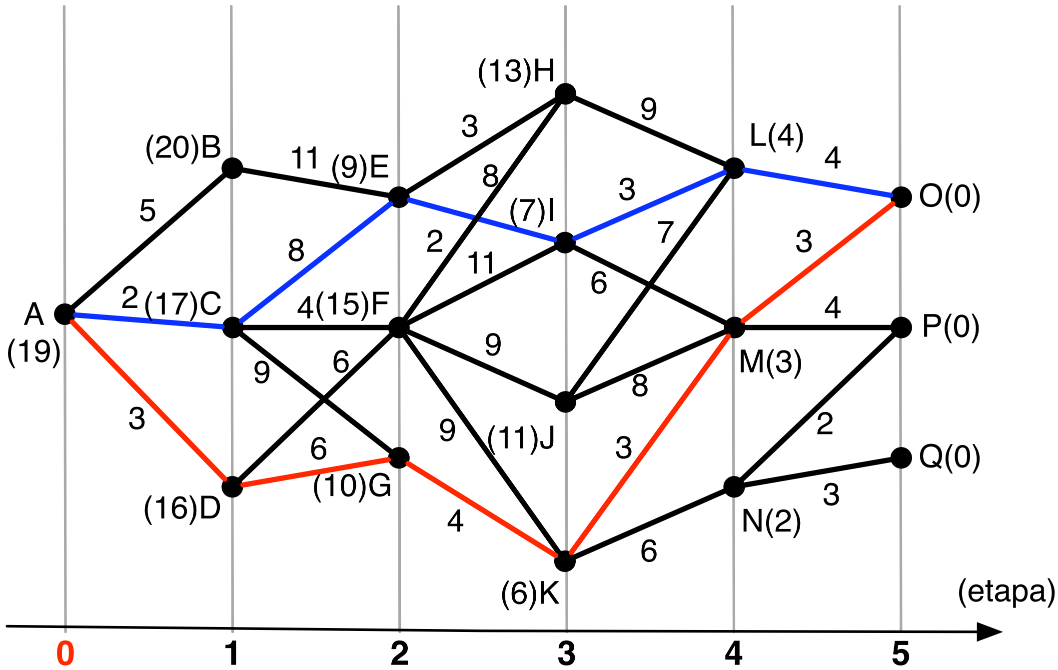


Figura 2.4: Ejemplo Programación Dinámica.

Como ejemplo de la técnica, supongamos que debemos encontrar cómo recorrer las etapas en la Figura 2.4 para llegar desde A hasta O, P, o Q (indistintamente) acumulando la menor distancia posible.

Para comenzar, nos paramos en el instante $t = 4$ y aplicamos la recursión para calcular f para los estados L, M y N. Como el valor de f para los estados terminales (O,P,Q) es 0, lo único que importa es el mínimo costo de transición entre L,M,N y O,P,Q. Es así que $f(L) = 4$, $f(M) = 3$, y $f(N) = 2$.

Ahora nos paramos en el instante $t = 3$ y repetimos la recursión para calcular $f(H), f(I), f(J), f(K)$. Para abreviar, veamos simplemente el caso del estado I. Podemos ver que desde I se puede ir hacia L con un costo de 3, y hacia M con un costo de 6, y ahora que sabemos que $f(L) = 4$ y $f(M) = 3$, podemos calcular entonces $f(I)$ como:

$$f(I) = \text{mín}\{3 + f(L), 6 + f(M)\} = \text{mín}\{7, 9\} = 7.$$

Esto quiere decir que el camino de menor costo para ir desde I hasta un nodo terminal tiene un costo de 7. Repetimos este cálculo para los demás nodos del instante $t = 3$, y luego hacemos los mismo para los nodos del instante $t = 2$, del instante $t = 1$, y por último para el nodo A del instante $t = 0$. Así llegamos a la conclusión de que el camino de menor costo para ir desde el estado inicial A hasta algún nodo terminal es 19, y que hay dos caminos posibles para hacerlo, que son los marcados en rojo y en azul en la Figura 2.4.

En la Sección 3.5 utilizaremos Programación Dinámica para resolver algunas instancias de nuestro problema.

2.2.6. Descenso por Gradiente Estocástico

Un *método iterativo* para la solución de problemas de optimización es el descenso por gradiente. Es muy utilizado por las técnicas de *Machine Learning* (ML) como por ejemplo el aprendizaje supervisado, o el *Reinforcement Learning* utilizado en este trabajo.

2.2.6.1. Descenso de gradiente en una dimensión

Veamos primero el caso de una dimensión, es decir buscamos:

$$x^* \longleftarrow \underset{x}{\operatorname{argmin}} f(x),$$

siendo $f : \mathbb{R} \to \mathbb{R}$ una función continua y diferenciable.

Como el nombre lo sugiere, lo que buscamos es descender en la dirección en que la función decrece para tratar de encontrar el mínimo, esto es, en la dirección opuesta al gradiente (derivada cuando se trata una dimensión). Para esto, recordamos el teorema de Taylor de primer orden:

$$f(x + \epsilon) = f(x) + \epsilon f'(x) + O(\epsilon^2).$$

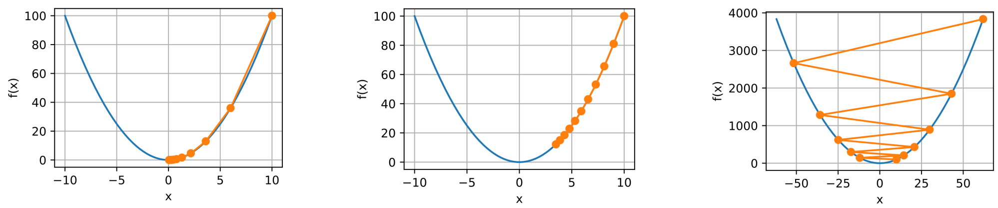
Elegimos dar un paso fijo de tamaño $\alpha > 0$ hacia donde decrece la función, eligiendo $\epsilon = -\alpha f'(x)$, esto resulta en:

$$f(x - \alpha f'(x)) = f(x) - \alpha f'^2(x) + O(\alpha^2 f'^2(x)).$$

Si la derivada no desaparece ($f'(x) \neq 0$), efectivamente el paso decrece ($\alpha f'^2(x) > 0$), y si el paso es pequeño ($O() \approx 0$):

$$f(x - \alpha f'(x)) \lessapprox f(x).$$

De esta forma podemos crear la iteración $x_{t+1} := x_t - \alpha f'(x_t)$, que irá reduciendo el valor de $f(x)$ y así poder acercarse a x^* (siempre y cuando se sigan cumpliendo las hipótesis de continuidad y diferenciabilidad).



(a) $\alpha = 0.2 \Rightarrow x_{10} = 0.0604$ **(b)** $\alpha = 0.05 \Rightarrow x_{10} = 3.4867$ muy lento y no se acerca a 0 **(c)** $\alpha = 1.1 \Rightarrow x_{10} = 61.9173$ muy rápido y diverge

Figura 2.5: Ejemplo Descenso por Gradiente en una dimensión. 10 iteraciones para la función $f(x) = x^2$ y distintos *learning rates*.

En ML, al tamaño del paso α se lo conoce como *learning rate*, y hay que tener particular cuidado en cómo elegirlo. La Figura 2.5 muestra un ejemplo de elecciones de *learning rates*, cómo afectan el descenso, y problemas asociados que pueden surgir. Incluso si se elige un buen *learning rate*, suele ser necesario una o varias condiciones de parada para terminar la iteración. Las más comunes son parar por cantidad de iteraciones, por tiempo de cómputo, o porque el valor de $|f'(x)|$ es suficientemente pequeño, entre otras.

2.2.6.2. Descenso por gradiente

El caso de una dimensión es muy útil para obtener la intuición de la técnica de descenso por gradiente. Veamos ahora la formulación análoga para el caso general.

Se tiene ahora un $\mathbf{x} \in \mathbb{R}^d$ y $f : \mathbb{R}^d \rightarrow \mathbb{R}$ una función continua y diferenciable. En lugar de la derivada usada en el caso de una dimensión, ahora usamos el gradiente:

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_d}\right]^\top.$$

Nuevamente, aplicando el teorema de Taylor, ahora con $\epsilon \in \mathbb{R}^d$:

$$f(\mathbf{x} + \epsilon) = f(\mathbf{x}) + \epsilon^\top \nabla f(\mathbf{x}) + O(\|\epsilon\|^2).$$

Análogamente, la iteración que reduce $f(\mathbf{x})$ es entonces:

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \alpha \nabla f(\mathbf{x}_t).$$

2.2.6.3. Descenso por gradiente *batch*

Como dijimos, los métodos de descenso por gradiente son comúnmente utilizados por las técnicas supervisadas de *Machine Learning* (ML). En el aprendizaje supervisado se cuenta con un conjunto de datos de entrada y de salida (x, y), y queremos aprender una función $f()$, tal que $y = f(x)$. Los métodos de aprendizaje utilizan un modelo paramétrico $\hat{y} = f(x; \theta)$ (por ejemplo una red neuronal), y el objetivo es encontrar los mejores parámetros θ que reduzcan el error del modelo. Para esto se introduce una función de pérdida o error entre el resultado del modelo y la realidad, $\mathcal{L}(\hat{y}, y)$. El aprendizaje se reduce entonces al siguiente problema de optimización:

$$\theta^* \longleftarrow \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(f(x; \theta), y).$$

La iteración de descenso por gradiente aplicada a este problema de aprendizaje supervisado es entonces:

$$\theta_{t+1} := \theta_t - \alpha \nabla_\theta \mathcal{L},$$

pero como no se suele conocer la distribución real de los datos, se suele aproximar el gradiente de $\mathcal{L}$ promediando el gradiente de la pérdida en cada elemento del conjunto de datos de entrenamiento de la siguiente forma:

$$\nabla_\theta \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \mathcal{L}(f(x_i; \theta), y_i),$$

siendo N el total de los datos de entrenamiento.
Cuando se utiliza descenso por gradiente en esta forma se lo conoce como *Batch Gradient Descent* (BGD).

2.2.6.4. Descenso por gradiente estocástico

Una de las desventajas que tiene BGD es que el conjunto de datos puede ser muy grande para calcular todo el gradiente, o puede ser muy lento. Una posible mejora es la técnica de *Stochastic Gradient Descent* (SGD). La técnica SGD puede verse como una aproximación de Monte Carlo para el gradiente empírico, y consiste en realizar varias iteraciones en donde se toma una porción aleatoria de los datos de entrenamiento (*mini-batch*) de tamaño m, y se utiliza para estimar el gradiente y realizar un paso de descenso por gradiente:

$$\hat{g} \longleftarrow \frac{1}{m}\nabla_\theta \textstyle\sum_{i=1}^{m} \mathcal{L}(f(x_i;\theta), y_i),$$

$$\theta_{t+1} := \theta_t - \alpha\hat{g}.$$

SGD tiene varias ventajas sobre BGD, como por ejemplo el ser más rápido y obtener mejores resultados para grandes conjuntos de datos. Además es más robusto a problemas comunes que sufren los algoritmos de ML, como por ejemplo el sobreajuste, en donde un algoritmo se ajusta mucho a los datos de entrenamiento, y no es capaz de generalizar a datos no vistos anteriormente.

2.2.7. Programación Dinámica Estocástica Dual

Otra de las técnicas de optimización que vamos a detallar es la Programación Dinámica Estocástica Dual(SDDP). Esta técnica ha sido usado para resolver el problema de despacho hidrotérmico óptimo, en particular en sistemas de gran tamaño como Brasil (Gorenstin et al. 1991), ya que a diferencia de la SDP, no sufre del problema de la “maldición de la dimensionalidad”. Esto se debe a que, como veremos, en esta técnica no es necesario discretizar el espacio de estados y calcular un costo futuro (o valores de Bellman) para cada punto, sino que la función de costo futuro se aproxima de forma analítica. La misma termina siendo una función lineal a tramos que el algoritmo SDDP puede aproximar bastante rápido. Como contrapartida, puede introducir errores, ya que asume convexidad de la función de costo futuro (si hay no convexidades el algoritmo puede no converger), cosa que sucede en el problema de despacho óptimo cuando las centrales de generación tienen mínimos técnicos.

A continuación detallamos la técnica.

Caso Determinista

Ilustraremos los conceptos de la programación dinámica dual con el siguiente problema lineal:

$$\begin{aligned} \min \quad & C_1X_1 + C_2X_2 \\ \text{s.t.} \quad & A_1X_1 \geq B_1, \\ & E_1X_1 + A_2X_2 \geq B_2 \end{aligned} \tag{2.2}$$

El problema (2.2) se puede interpretar como un proceso de decisión secuencial en dos etapas. En la primera etapa, elegimos un valor factible de prueba para X_1 (que cumpla $A_1X_1 \geq B_1$). Dado el valor de prueba, $\hat{X}_1$, encontramos la solución óptima de la función de la segunda etapa:

$$\begin{aligned} \min \quad & C_2X_2 \\ \text{s.t.} \quad & A_2X_2 \geq B_2 - E_1\hat{X}_1 \end{aligned} \tag{2.3}$$

Notar que el valor $\hat{X}_1$ es un valor conocido en el problema de la segunda etapa (2.3), y por eso va en el lado derecho de las restricciones. El objetivo es minimizar la suma de las funciones de costo de la primera y segunda etapa.

El algoritmo de programación dinámica (DP) se puede usar para resolver problemas de decisión secuenciales como el problema (2.2). En el enfoque DP tradicional, el problema de la primera etapa se define como:

$$\begin{aligned} \min \quad & C_1X_1 + CF_1(X_1) \\ \text{s.t.} \quad & A_1X_1 \geq B_1 \end{aligned} \tag{2.4}$$

En la terminología DP, C_1X_1 representa el "costo inmediato" y $CF_1(X_1)$ representa el "costo futuro" de tomar la decisión X_1, es decir, las consecuencias de esta decisión para el problema de la segunda etapa. La función de costo futuro $CF_1(X_1)$ se define como:

$$\begin{aligned} CF_1(X_1) = \ \min \quad & C_2X_2 \\ \text{s.t.} \quad & A_2X_2 \geq B_2 - E_1X_1 \end{aligned} \tag{2.5}$$

La función de costo futuro (2.5) se puede decir que "traduce" los costos de la segunda etapa como una función de las decisiones de la primera etapa X_1. Si esta función está disponible, el problema en dos etapas (2.2) se puede resolver como el problema en una etapa (2.4).

El algoritmo DP construye la función de costo futuro $CF_1(X_1)$ discretizando X_1 en un conjunto de valores de prueba $\{\hat{X}_{1i}, i = 1, ..., n\}$ y resolviendo el problema (2.5) para cada uno de esos valores. Valores intermedios de $CF_1(X_1)$ se obtienen por interpolación de los valores discretizados vecinos. Una vez que esta función está construida, el problema de la primera etapa (2.4) se puede resolver.

El enfoque DP tiene muchas cualidades atractivas: se puede extender fácilmente a problemas multietapa; se puede extender a casos estocásticos; puede atender valores discretos, no linealidades, etc. Como se mencionó previamente, su principal desventaja es el crecimiento exponencial del esfuerzo computacional con las dimensiones del sistema. Esta "maldición de la dimensionalidad" limita la aplicación de DP a problemas con un tamaño reducido de variables (más aún en el caso estocástico).

Una manera posible de evitar el problema de la dimensionalidad es aproximar la función de costo futuro por una función analítica en vez de un conjunto de valores discretos. Por ejemplo, uno podría calcular la función de costos futuros $CF_1(X_1)$ para una muestra de estados $\{\hat{X}_{1i}\}$, y luego ajustar un polinomio (por ejemplo una función cuadrática o cúbica de X_1) a estos valores. El polinomio luego sería usado en la etapa previa para suministrar valores de costo futuro para cualquier decisión de prueba X_1. Veremos que la función de costo futuro se puede representar de forma exacta como una función lineal a tramos, y que podemos usar una relajación de esta función a tramos como nuestra aproximación.

La estructura de la función de costos futuros se puede caracterizar tomando el ***Problema Dual*** del problema de la segunda etapa (2.5):

$$
\begin{aligned}
CF_1(X_1) = \ & \max \quad \psi(B_2 - E_1 X_1) \\
& \text{s.t.} \quad \psi A_2 \leq C_2
\end{aligned}
\tag{2.6}
$$

donde ψ es el vector de variables duales. Sabemos de la teoría LP que la solución óptima del problema dual (2.6) y la del problema primal (2.5) coinciden. Por ende, tanto (2.5) como (2.6) representan la función de costos futuros $CF_1(X_1)$. Notar, sin embargo, que la variable de decisión X_1 está en la función objetivo de (2.6), y no en el lado derecho de las restricciones como en el problema primal (2.5). Esto significa que el conjunto de soluciones posibles al problema (2.6), que corresponde con los vértices del conjunto de restricciones $\psi A2 \leq C2$, puede ser caracterizado ***antes*** de conocer la decisión X_1.

Sea $\Psi = \{\psi^1, ..., \psi^V\}$ los vértices del conjunto de restricciones. Como la solución óptima pertenece a este conjunto, el problema (2.6) se puede resolver en principio por enumeración:

$$
CF_1(X_1) = \max\{\psi^i(B_2 - E_1 X_1), \text{ para } i = 1, ..., V\}
\tag{2.7}
$$

El problema (2.7) se puede reescribir como un problema lineal:

$$
\begin{aligned}
CF_1(X_1) = \ & \min \quad CF \\
& \text{s.t.} \quad CF \geq \psi^1(B_2 - E_1 X_1), \\
& \qquad\quad ..., \\
& \qquad\quad ..., \\
& \qquad\quad CF \geq \psi^V(B_2 - E_1 X_1)
\end{aligned}
\tag{2.8}
$$

donde CF es una variable escalar. La equivalencia entre (2.7) y (2.8) se puede establecer fácilmente observando que $CF \geq \psi^i(B_2 - E_1 X_1), \text{ para } i = 1, ..., V$ en el problema (2.8) implica que $CF \geq \max\{\psi^i(B_2 - E_1 X_1)\}$. Como el objetivo es minimizar CF, se puede concluir que la restricción se cumplirá en la igualdad como es requerido por (2.7).

El problema (2.8) tiene una interpretación geométrica interesante. Indica que la función de costo futuro $CF_1(X_1)$ es una función lineal a tramos de la variable de decisión X_1. Los componentes de esta función a tramos son los hiperplanos soporte de la función de costos futuros definida por cada $\psi^i(B_2 - E_1X_1)$. Esto implica que la función de costo futuro se puede caracterizar sin discretizar X_1; es suficiente conocer los coeficientes $\{\psi^i\}$ de los hiperplanos soporte.

Naturalmente, el cálculo de todos los vértices $\{\psi^i\}$ en el conjunto Ψ puede ser una tarea muy difícil. El enfoque será calcular un subconjunto de estos vértices y construir una ***aproximación*** a la función de costo futuro. En principio podemos ver que estos vértices se pueden calcular como variables duales del problema de la segunda etapa (2.5):

$$
\begin{aligned}
CF_1(\hat{X}_{1i}) = \ \min \quad & C_2X_2 & \qquad \text{Variables Duales} \\
\text{s.t.} \quad & A_2X_2 \geq B_2 - E_1\hat{X}_{1i} & \psi^i
\end{aligned}
\tag{2.9}
$$

donde $\hat{X}_{1i}$ es un valor de prueba. Sea ψ^i el vector multiplicador simplex asociado a las restricciones del problema (2.9). Sabemos de la teoría LP que este vector es uno de los vértices del conjunto solución Ψ en el problema dual. Entonces, se puede usar para construir uno de los hiperplanos soporte de la función de costo futuro $CF_1(X_1)$.

En otras palabras, dado un conjunto de n decisiones de prueba $\{X_{1i}, i = 1, ..., n\}$, podemos calcular el conjunto de multiplicadores asociados $\{\psi^i, i = 1, ..., n\}$ resolviendo el problema (2.9) para cada uno de los valores de prueba. Una aproximación a la función de costo futuro se puede entonces construir como:

$$
\begin{aligned}
\hat{CF}_1(X_1) = \ \min \quad & CF \\
\text{s.t.} \quad & CF \geq \psi^i(B_2 - E_1X_1) \text{ para } i = 1, ..., n
\end{aligned}
$$

$$(2.10)$$

Es fácil ver que la función aproximada $\hat{CF}_1(X_1)$ es una cota inferior de la función de costo futuro $CF_1(X_1)$, porque el problema (2.10) tiene solo un subconjunto de las restricciones del problema (2.8). La función de costo futuro aproximada puede ser usada para resolver el problema de la primera etapa como en la formulación DP:

$$
\begin{aligned}
Z = \ \min \quad & C_1X_1 + \hat{CF}_1(X_1) \\
\text{s.t.} \quad & A_1X_1 \geq B_1
\end{aligned}
\tag{2.11}
$$

Notar que (2.11) es un problema LP. Sustituyendo (2.10) en (2.11), obtenemos:

$$
\begin{aligned}
Z = \ \min \quad & C_1X_1 + CF \\
\text{s.t.} \quad & A_1X_1 \geq B_1, \\
& CF - \psi^i(B_2 - E_1X_1) \geq 0 \text{ para } i = 1, ..., n
\end{aligned}
\tag{2.12}
$$

Como $\hat{CF}_1(X_1)$ es una aproximación de la función de costo futuro, no podemos garantizar que la solución del problema (2.11) sea la solución óptima del problema de dos etapas (2.2). Sin embargo, como $\hat{CF}_1(X_1)$ es una cota inferior de la función de costo futuro, sabemos que la solución óptima de (2.12) es una cota inferior $\underline{Z}$ del verdadero costo óptimo. En otras palabras,

$$\underline{Z} = C_1\hat{X}_1 + CF \tag{2.13}$$

donde $\hat{X}_1$ y CF son las soluciones óptimas del problema aproximado (2.12). Por otro lado, una cota superior $\overline{Z}$ se puede obtener sumando el valor de la solución del problema de la segunda etapa (2.9) a $C_1\hat{X}_1$:

$$\overline{Z} = C_1\hat{X}_1 + CF_1(\hat{X}_1).$$

(2.14)

La diferencia entre la cota superior e inferior $\overline{Z} - \underline{Z}$ se puede usar para verificar la precisión de la función de costo futuro aproximada. Notemos que el término $C_1\hat{X}_1$ se cancela, porque pertenece a ambas cotas. Entonces, podemos ver que $\overline{Z} - \underline{Z}$ mide la diferencia entre el costo futuro predicho (dado por CF) y el costo futuro verdadero (dado por $CF_1(\hat{X}_1)$) para la solución de prueba actual $\hat{X}_1$. Si esta diferencia es menor a una cierta tolerancia, el problema está resuelto. Sino, un nuevo conjunto de decisiones de prueba se debe usar para construir una nueva aproximación a la función de costo futuro.

El proceso hasta el momento se puede resumir con el siguiente algoritmo:

a) seleccionar un conjunto de n decisiones de prueba $\{\hat{X}_{1i}, i = 1, ..., n\}$
b) para cada decisión de prueba, resolver el problema de la segunda etapa y calcular los multiplicadores asociados ψ^i como en (2.9)
c) usar los multiplicadores $\{\psi^i\}$ para construir una aproximación a la función de costo futuro como en (2.10); resolver el problema aproximado de la primera etapa (2.11)
d) calcular cotas superior e inferior como en (2.13) y (2.14); si $\overline{Z} - \underline{Z}$ está dentro de una determinada tolerancia ϵ, termina; sino ir a (a).

Un punto importante que falta discutir es la selección de decisiones de prueba $\{\hat{X}_{1i}\}$ en el paso (a). En cada iteración del algoritmo, usaremos como decisión de prueba adicional a la solución óptima $\hat{X}_1$ del paso (c) (el problema aproximado de la primera etapa) en la iteración ***previa***. Esto asegura que construyamos aproximaciones a la función de costo futuro alrededor de puntos que son buenos candidatos de ser la solución óptima.

El algoritmo de programación dinámica dual en dos etapas (DDP) está compuesto entonces por los siguientes pasos:

a) inicializar: la función de costo futuro $\hat{CF}_1(X_1) = 0$; cota superior $\overline{Z} = \infty$; número de vértices $n = 0$
b) resolver el problema aproximado de la primera etapa (2.12); sea $\hat{X}_1$ la solución óptima
c) calcular la cota inferior $\underline{Z}$ como en (2.13); si $\overline{Z} - \underline{Z} \leq \epsilon$, parar; sino, ir a (d)
d) resolver el problema de la segunda etapa (2.9), es decir calcular $CF_1(\hat{X}_1)$; actualizar $\overline{Z}$ como en (2.14)

e) incrementar el número de vértices $n \leftarrow n+1$; sea el multiplicador asociado a la solución óptima del paso (d) ψ^n; actualizar el aproximado de la función de costo $\hat{CF}_1(X_1)$ como en (2.10)

f) ir al paso (b).

El algoritmo DDP descrito arriba tiene varias cualidades atractivas: no se requiere discretización del estado; cotas superior e inferior se obtienen en cada iteración; la solución óptima previa del problema aproximado de optimización (2.12) en el paso (b) se puede usar como una solución inicial en la próxima iteración (notar que la única diferencia entre dos problemas sucesivos es una restricción lineal adicional asociado al nuevo vértice ψ^n).

Luego de explicada la versión DDP en dos etapas, veamos la generalización del algoritmo DDP a múltiples etapas, el cual se compone por los siguientes pasos:

a) Sea T el horizonte de planificación; inicializar $\hat{CF}_t(X_t) = 0$ para $t = 1, .., T$; $\overline{Z} = \infty$;

b) resolver el problema aproximado de la primera etapa (2.12); sea $\hat{X}_1$ la solución óptima

c) calcular la cota inferior $\underline{Z}$ como en (2.13); si $\overline{Z} - \underline{Z} \leq \epsilon$, parar; sino, ir a (d)

d) repetir para $t = 2, ..., T$ (simulación hacia adelante)

resolver el problema de optimización para la decisión de prueba $\hat{X}_{t-1}$:

$$\begin{aligned} \min \quad & C_t X_t + \hat{CF}_t(X_t) \\ \text{s.t.} \quad & A_t X_t \geq B_t - E_{t-1}\hat{X}_{t-1} \end{aligned} \tag{2.15}$$

guardar la solución óptima como $\hat{X}_t$

e) calcular la cota superior

$$\overline{Z} = \sum_{t=1}^{T} C_t \hat{X}_t \tag{2.16}$$

f) repetir para $t = T, T-1, ..., 2$ (recursión hacia atrás)

resolver el problema de optimización para la decisión de prueba $\hat{X}_{t-1}$:

$$
\begin{aligned}
\min \quad & C_t X_t + \hat{CF}_t(X_t) \\
\text{s.t.} \quad & A_t X_t \geq B_t - E_{t-1}\hat{X}_{t-1}
\end{aligned}
\tag{2.17}
$$

Sea ψ_{t-1} el multiplicador asociado a las restricciones del problema (2.17) para la solución óptima; usar este multiplicador para construir un hiperplano soporte adicional para el aproximado de la función de costo futuro en la etapa anterior, $\hat{CF}_{t-1}(X_{t-1})$

g) ir al paso (b)

Caso Estocástico

El enfoque DDP visto en la sección anterior también se puede extender al caso estocástico. Ilustraremos esto con el siguiente problema en dos etapas:

$$
\begin{aligned}
\min \quad & C_1X_1 + P_1C_2X_{21} + P_2C_2X_{22} + ... + P_mC_2X_{2m} \\
\text{s.t.} \quad & A_1X_1 \geq B_1, \\
& E_1X_1 + A_2X_{21} \geq B_{21}, \\
& E_1X_1 \qquad\qquad + A_2X_{22} \geq B_{22}, \\
& ..., \\
& ..., \\
& E_1X_1 \qquad\qquad\qquad\qquad + A_2X_{2m} \geq B_{2m}
\end{aligned}
\tag{2.18}
$$

El problema (2.18) se puede interpretar de la siguiente forma: en la primera etapa, se toma una decisión X_1; dada la decisión X_1 de prueba, habrá m subproblemas de la segunda etapa:

$$
\begin{aligned}
CF_{1j}(X_1) = \min \quad & C_2X_{2j} \\
\text{s.t.} \quad & A_2X_{2j} \geq B_{2j} - E_1X_1
\end{aligned}
$$

(2.19)

para todo $j = 1, ..., m$. El objetivo es minimizar la suma de costos de la primera etapa C_1X_1 más el valor esperado de los costos de la segunda etapa ($\sum P_jC_2X_{2j}$), donde P_j representa la probabilidad de cada subproblema (naturalmente, $\sum P_j = 1$).

Análogamente al caso determinístico, el problema en dos etapas (2.18) se puede en principio resolver por una recursión DP estocástica. Dada una decisión de prueba X_1, se puede construir la función de costo futuro esperado $\overline{CF}_1(X_1)$ como:

$$\overline{CF}_1(X_1) = \sum_{j=1}^{m} P_jCF_{1j}(X_1) \tag{2.20}$$

donde $CF_{1j}(X_1)$ está definido en (2.19). El problema de la primera etapa en la recursión DP se convierte en:

$$\begin{aligned} \min \quad & C_1X_1 + \overline{CF}_1(X_1) \\ \text{s.t.} \quad & A_1X_1 \geq B_1 \end{aligned} \tag{2.21}$$

donde C_1X_1 representa el costo inmediato, y $\overline{CF}_1(X_1)$ representa las consecuencias futuras ***esperadas*** de la decisión X_1. Las derivaciones que conducen al algoritmo DDP son similares a las del caso determinístico. Para presentar el DDP estocástico multietapa asumimos, sin pérdida de generalidad, que los vectores del lado derecho $\{Bt, t = 1, ..., T\}$ son variables aleatorias independientes, y que cada B_t está discretizado en m valores o escenarios $\{B_{tj}, j = 1, ..., m\}$ con probabilidades $\{P_{tj}, j = 1, ..., m\}$. El algoritmo se implementa de la siguiente manera:

a) definir un conjunto de decisiones de prueba $\{\hat{X}_{ti}$, para $i = 1, ..., n; t = 1, ..., T\}$
b) repetir para $t = T, T-1, ..., 2$ (recursión hacia atrás)
    repetir para cada decisión de prueba $\hat{X}_{ti}, i = 1, ..., n$
        repetir para cada escenario $B_{tj}, j = 1, ..., m$
            resolver el problema de optimización para t, $\hat{X}_{t-1i}$ y B_{tj}:

$$\begin{aligned} \min \quad & C_t X_t + \hat{CF}_t(X_t) \\ \text{s.t.} \quad & A_t X_t \geq B_{tj} - E_{t-1}\hat{X}_{t-1i} \end{aligned} \tag{2.22}$$

Sea ψ_{t-1ij} el multiplicador asociado a las restricciones del problema (2.22) en la solución óptima

Calcular el valor esperado del vértice $\overline{\psi}_{t-1i} = \sum P_{tj}\psi_{t-1ij}$, y construir un hiperplano soporte del aproximado de la función de costo futuro esperado para la etapa $t-1$, $\hat{CF}_{t-1}(X_{t-1})$

c) ir al paso a).

Como en el caso determinístico, un aspecto importante del algoritmo es determinar las decisiones de prueba $\{\hat{X}_{ti}\}$. Idealmente, deberíamos realizar una simulación hacia adelante (como en el paso (d) del caso determinístico multietapa) para cada combinación de escenarios $\{B_{tj}\}$. Notar, sin embargo, que las combinaciones incrementan exponencialmente con el número de etapas. Una alternativa que se suele usar es hacer una simulación Monte Carlo hacia adelante para una muestra de los escenarios de la siguiente manera:

a) resolver el problema de la primera etapa (2.21) con $\overline{CF}_1(X_1) = 0$ en la primera iteración; sea $\hat{X}_1$ la solución óptima; inicializar $X_{1i} = \hat{X}_1$ para $i = 1, ..., n$

b) repetir para $t = 2, ..., T$

repetir para $i = 1, ..., n$

muestrear un vector B_{ti} del conjunto $\{B_{tj}, j = 1, ..., m\}$

resolver el problema para la etapa t, muestra i

$$\begin{aligned} \min \quad & C_t X_t + \hat{CF}_t(X_t) \\ \text{s.t.} \quad & A_t X_t \geq B_{ti} - E_{t-1}\hat{X}_{t-1i} \end{aligned} \tag{2.23}$$

guardar la solución óptima como $\hat{X}_{ti}$

Como en el caso determinístico, el objetivo de esta simulación es determinar "buenas" decisiones de prueba en cada etapa, es decir, alrededor cuáles se debería tratar de aproximar la función de costo futuro.

Un aspecto que queda por discutir es cuál es el cálculo de las cotas superior e inferior. Se puede ver que la cota inferior $\underline{Z}$ se obtiene de la solución del problema de la primera etapa (2.21), como en el caso determinístico. La cota superior $\overline{Z}$, en cambio, se ***estima*** de los resultados de la simulación Monte Carlo para todas las etapas y escenarios, esto es:

$$\overline{Z} = (1/n) \sum_{i=1}^{n} Z_i \tag{2.24}$$

donde Z_i es el costo total de una corrida Monte Carlo:

$$Z_i = \sum_{t=1}^{T} C_t \hat{X}_{ti}. \tag{2.25}$$

La incertidumbre alrededor del estimador de $\overline{Z}$ en la expresión (2.24) se puede estimar por la varianza del estimador:

$$\sigma_{\overline{Z}}^2 = (1/n^2) \sum_{i=1}^{n} (\overline{Z} - Z_i)^2. \tag{2.26}$$

Por ejemplo, el intervalo de confianza al $95\,\%$ del "verdadero" valor de $\overline{Z}$ está dado por:

$$[\overline{Z} - 1.96\sigma_{\overline{Z}}, \overline{Z} + 1.96\sigma_{\overline{Z}}]. \tag{2.27}$$

La incertidumbre alrededor del estimador de la cota superior se puede usar como un criterio de convergencia: por ejemplo, si la cota inferior $\underline{Z}$ está en el intervalo (2.27), el algoritmo se detiene. Este criterio introduce una relación entre la precisión aceptable de la simulación (dada por el tamaño de la muestra n) y la precisión de la política óptima calculada por el algoritmo SDDP.

2.2.8. Programación Dinámica y *Reinforcement Learning*

Esta breve reseña histórica está basada en lo dicho en Sutton y Barto, 2018.

La historia del *Reinforcement Learning* (RL) tiene dos hilos conductores principales; dos temáticas que fueron extensamente estudiadas por separado durante décadas antes de desembocar en lo que es el RL moderno. Uno de estos hilos es el aprendizaje a prueba y error que comenzó en la psicología del aprendizaje de los animales que estuvo presente desde los primeros trabajos en inteligencia artificial. El otro hilo concierne al problema del control óptimo, y a su solución mediante el uso de funciones de valor y programación dinámica. En su gran mayoría, este hilo no involucraba ningún tipo de aprendizaje. Ambos hilos se juntaron al final de los 80's para formar el campo de RL.

El término "control óptimo" se comenzó a utilizar para describir el problema de diseñar un controlador que minimice alguna medida del comportamiento de un sistema dinámico en el tiempo. Uno de los primeros enfoques para encarar estos problemas fue desarrollado a mediados de los años 50's por Richard Bellman. Este enfoque usa los conceptos del estado de un sistema dinámico y de una función de valor (o función de retorno óptimo), para definir una ecuación funcional hoy conocida como la "Ecuación de Bellman". A la clase de métodos para resolver problemas de control óptimo resolviendo esta ecuación se los denominó Programación Dinámica (Bellman, 1957a). Bellman también introdujo la versión discreta estocástica del problema de control óptimo conocido como los Procesos de Decisión de Markov (MDPs)(Bellman, 1957b).

Cuando es aplicable, la programación dinámica se considera como la técnica principal para resolver problemas de control óptimo estocástico generales. Sufre de lo que se conoce como la "maldición de la dimensionalidad". Dado que la complejidad computacional crece de forma exponencial con el número de variables de estado, la técnica se vuelve inviable en los problemas que requieren una cantidad enorme de estados para ser expresados en la formulación de Bellman. Recordar que el algoritmo de programación dinámica es *directo*. El no contar con solución o aproximación alguna antes de finalizar la ejecución combinado con la maldición de la dimensionalidad inviabilizan la técnica en muchas aplicaciones. Una de las motivaciones de esta tesis es examinar cómo las técnicas de aprendizaje por refuerzo (*Reinforcement Learning*) se desempeñan ante problemas en los que la programación dinámica caería en la maldición de la dimensionalidad.

Como mencionamos antes, las conexiones entre el control óptimo y la programación dinámica con el aprendizaje llevaron mucho tiempo en ser identificadas y reconocidas. Probablemente debido a la distancia entre las disciplinas involucradas y sus diferentes objetivos. Otra explicación surge del enfoque, ya que la programación dinámica es un ordenamiento de cálculos que requiere un modelo preciso del sistema y una solución analítica a la ecuación de Bellman. Se agrega que en las aplicaciones más comunes, la programación dinámica procede de atrás para adelante en el tiempo, lo que es poco intuitivo y difícil de relacionar con un proceso de aprendizaje, que naturalmente se imagina hacia adelante.

Identificadas las conexiones, el área ha crecido sensible y sostenidamente en diversidad y profundidad, generando paradigmas enmascarados en nuevos términos como: "programación dinámica heurística" (Werbos, 1977), "Neuro-dynamic programming" (Bertsekas y Tsitsiklis, 1996), o "programación dinámica aproximada" (Powell, 2007). Todos estos enfoques hacen énfasis en distintos aspectos del tema, pero todos comparten con RL el interés de circunnavegar las carencias de la programación dinámica clásica.

2.3. *Reinforcement Learning* (RL)

Las técnicas de Reinforcement Learning (RL) (Sutton y Barto, 2018, Bertsekas, 2019) o Aprendizaje por refuerzos son un conjunto de técnicas de aprendizaje automático en las que un **agente** aprende mediante interacción con un **entorno** para lograr un objetivo. El agente aprende de la consecuencia que tienen sus acciones mediante ensayo y error. Además del agente y del entorno, se pueden identificar cuatro componentes principales en un sistema de RL: una política, una recompensa, una función de valor, y, opcionalmente, un modelo del entorno. A continuación, explicaremos brevemente cada uno de estos conceptos, los cuales formalizaremos inmediatamente después.

Una **política** define el comportamiento del agente en cualquier instante dado. En términos simples, una política es un mapeo entre estados del entorno y acciones a ser tomadas en esos respectivos estados. La política es el elemento central del agente de RL, ya que ella es suficiente para determinar completamente su comportamiento.

Una **recompensa** define el objetivo en un problema de RL. En cada instante de tiempo, el entorno le envía al agente de RL una recompensa numérica. El objetivo del agente es maximizar el total de recompensas obtenidas a largo plazo. La recompensa enviada al agente depende únicamente de la acción actual del agente y del estado actual del entorno; pero ambos son a su vez consecuencia de la política elegida. Si la acción elegida por la política en un estado obtiene una recompensa baja, la política puede ser modificada para elegir una acción distinta en ese mismo estado en el futuro.

A diferencia de la recompensa que indica si algo es bueno o no de forma inmediata, la **función de valor** especifica si algo es bueno o no a largo plazo. En términos simples, el valor de un estado es la recompensa total que un agente puede esperar acumular en el futuro, desde ese estado. El valor de un estado determina qué tan deseable es a largo plazo, teniendo en cuenta los posibles estados que le seguirán, y las recompensas que se podrán obtener en esos estados. El valor de un estado es lo que más nos concierne a la hora de hacer o evaluar decisiones. La elección de acciones se hace en base a juicios de valor. Buscamos acciones que nos lleven a estados de alto valor, no de alta recompensa inmediata, ya que estas acciones obtendrán la mayor recompensa a largo plazo. El concepto de la función de valor se desarrolla a fondo en la Sección 2.3.3.

El cuarto elemento de algunos sistemas de RL es un **modelo del entorno**. Esto es, algo que simule el comportamiento del entorno y que nos permita hacer inferencias sobre cómo va a evolucionar: que capture la dinámica entre estados del sistema y acciones. Con un modelo se podría predecir, por ejemplo, dado un estado y una acción, el próximo estado y la próxima recompensa, resultante de aplicar esa acción en ese estado. Esto se puede usar para planificar, es decir, elegir un lineamiento de acciones considerando posibles situaciones futuras antes de haberlas experimentado. En la literatura, se distingue entre planificación y aprendizaje simplemente por el hecho de que uno aprende mediante la interacción con el entorno (aprendizaje, a veces llamado RL directo, aprende a través de experiencias reales), y el otro aprende mediante interacción con un modelo del entorno (planificación, a veces llamado RL indirecto, aprende a través de experiencias simuladas). Aprender interactuando con un modelo del entorno puede tener ventajas sobre aprender interactuando con el entorno real, como ser: menor costo computacional, el no desperdicio de recursos, o la posibilidad de resolver problemas sobre extensos horizontes temporales. Por otro lado, como siempre, el modelo puede tener asociado un error o sesgo sobre la realidad que modela. Cuando no se cuenta con un modelo, hay algoritmos de RL que van aprendiendo un modelo a la vez que van interactuando con el entorno, para sacarle el mayor provecho de aprender con experiencia real y con experiencia simulada a la par.

En este trabajo no entramos en esta distinción, ya que para la tarea de despacho hidrotérmico, lo único que tiene sentido es trabajar con un modelo y experiencias simuladas. Claro está que no podemos poner a un agente a operar los generadores energéticos del país y experimentar hasta que aprenda una política, menos aún cuando se trata de un despacho a largo plazo. Lo único razonable es trabajar con un modelo, que el agente experimente todo lo necesario mediante simulaciones, y esperar que la política óptima que se aprende al finalizar las simulaciones, se aplique y sea óptima también en la realidad. Por razones de alcance, en este trabajo no se profundiza en la validación de los modelos de entorno utilizados respecto a la realidad que modelan, algo que debe ser estudiado previo al uso práctico de la solución desarrollada.

A continuación formalizaremos los conceptos fundamentales introducidos previamente. Nos basaremos en la notación y definiciones utilizadas en el Capítulo 3, *Finite Markov Decision Processes*, de Sutton y Barto, 2018.

2.3.1. Agente, Entorno y Política

El agente siempre se encuentra en el entorno, y el entorno se determina completamente por su estado. El agente interactúa en el entorno en una secuencia discreta de pasos temporales, $t = 0, 1, 2, \ldots$[1]. En el instante t, el agente se encuentra en cierto estado $S_t \in \mathcal{S}$, y decide tomar la acción $A_t \in \mathcal{A}$[2], mediante la cuál pasa a un nuevo estado, S_{t+1}, y observa una recompensa, $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$. La Figura 2.6 representa el esquema básico de la interacción entre el entorno y el agente de RL.

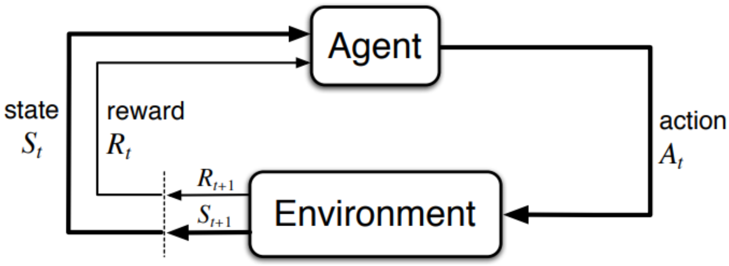


Figura 2.6: Esquema básico de la interacción entre el entorno y el agente de RL. Fuente (Sutton y Barto, 2018)

Este proceso se repite en cada paso del tiempo, generando una trayectoria del agente: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \ldots$. En su versión más simple, los conjuntos $\mathcal{S}$, $\mathcal{A}$, y $\mathcal{R}$ son discretos y finitos, y las variables S_t y R_t son aleatorias, de las cuales se conoce su distribución de probabilidad.

[1]Puede generalizarse a procesos continuos.

[2]Para simplificar la notación, asumimos que en todos los estados se pueden tomar las mismas acciones.

El agente va aprendiendo de las recompensas que obtiene y este proceso se repite hasta que se cumple cierta condición de parada. La política que el agente utiliza para determinar cada acción en cada estado es estocástica, y puede expresarse como una función $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$, donde $\pi(a|s)$ es la probabilidad de que el agente elija la acción a cuando se encuentra en el estado s, es decir $\pi(a|s) = \Pr(A_t = a|S_t = s)\ \forall t$. Observar que para remarcar el condicional utilizamos la notación $\pi(.|.)$ pero en realidad es una función ordinaria $\pi(., .)$. También observar que suponemos es un proceso markoviano, donde la acción a tomar no depende del instante de tiempo en que se encuentra el agente, sino solo del estado del entorno. Finalmente, para que sea una distribución de probabilidad se cumple:

$$\sum_{a \in \mathcal{A}} \pi(a|s) = 1\ \forall s \in \mathcal{S}.$$

2.3.2. Recompensa y Episodios

El objetivo es aprender una política de acciones para maximizar el valor esperado del retorno G_t en cada estado. El retorno es la suma de las recompensas descontadas esperadas a partir del tiempo t:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k}.$$

A $\gamma \in [0, 1]$ se lo conoce como el factor de descuento. El factor de descuento es utilizado para determinar qué tanta importancia le da el agente a las recompensas en el futuro lejano con respecto a las recompensas en el futuro cercano. Si $\gamma = 0$, la política será completamente miope, es decir, solo aprenderá de acciones que produzcan una recompensa inmediata. Si $\gamma = 1$, todas las recompensas tendrán el mismo valor sin importar en qué instante aparezcan. Según la tarea, y según qué se use como recompensa, puede ser deseable que las recompensas sean descontadas. Por ejemplo, cuando se trabaja con predicciones, las mismas pueden ser menos confiables mientras más nos alejamos en el futuro, este aumento de varianza se puede reflejar en el factor de descuento. Otro caso es cuando la magnitud utilizada para las recompensas pierde valor en el tiempo de forma inherente, como es el caso económico. No es lo mismo una cierta cantidad de dinero hoy que la misma cantidad de dinero dentro de 10 años. El factor de descuento γ se puede utilizar para que el agente aprenda esta diferencia.

En muchos procesos el agente no actúa por tiempo infinito, sino que hay un tiempo final, T. Por ejemplo, cuando el proceso es un juego como el ajedrez, los instantes de tiempo serían cada vez que se hace una jugada, y el tiempo final sería una partida completa. En este caso entonces,

$$G_t = \sum_{k=0}^{T} \gamma^k R_{t+k}.$$

En este ejemplo (y en muchos otros casos), el tiempo final varía en cada partida. Y el agente puede jugar varias partidas consecutivas. Se denomina **episodio** a una realización completa del proceso que resuelve el agente, en este caso cada partida. En el problema que se resuelve en esta tesis se trata de una optimización económica en un cierto horizonte de tiempo determinado y fijo, el episodio entonces está formado por todos los pasos comprendidos en el horizonte de planificación. En la práctica el problema de planificación se repite de forma continua, un episodio tras otro.

2.3.3. Funciones de Valor

La función de valor es una función del estado (o de un par estado-acción) que estima qué tan bueno es para el agente estar en un estado dado (o qué tan bueno es aplicar una acción dada en un estado dado). Por "bueno" estamos hablando en términos de las recompensas futuras que se pueden esperar. Obviamente, las recompensas futuras que se puedan esperar dependen de las acciones que se tomen. Por esto es que las funciones de valor suelen estar definidas con respecto a una política. Informalmente, el **valor de un estado** s para una política π, denotado $v_\pi(s)$, es el retorno esperado partiendo de s y siguiendo la política π. Se puede definir con siguiente fórmula:

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s\right],$$

donde $\mathbb{E}_\pi$ se refiere al valor esperado cuando el agente sigue la política estocástica π. Si el tiempo t es importante para caracterizar al problema, se asume que está incorporado al estado (como ejemplo de esto podemos ver la Figura 2.4, donde identificar un estado, digamos M, identifica también que nos encontramos en el tiempo $t = 4$).

Asimismo, se define el **valor de tomar una acción** a en el estado s bajo la política π, denotado $q_\pi(s, a)$, como el retorno esperado empezando desde s, tomando la acción a, y luego siguiendo la política π:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\middle|S_t = s, A_t = a\right].$$

Es claro ver que si se tuvieran los valores óptimos de cada estado, o de cada par estado-acción (comúnmente denotados como v_* y q_*), entonces sería trivial obtener una política óptima (comúnmente denotada π_*) simplemente mediante una política que siga v_* (si contamos con un modelo) o a q_* si es conocida. Claramente un agente que aprenda estos valores habrá hecho un muy buen trabajo, pero en la práctica esto rara vez sucede. Para los problemas en los que interesa aplicar RL, esto suele ser intratable desde el punto de vista computacional. Incluso si se cuenta con un modelo completo y acertado de la dinámica del entorno, puede ser imposible calcular la política óptima de esta manera, ya que por ejemplo, el espacio de estados puede ser demasiado grande. En estos casos, las funciones de valor deben ser aproximadas usando alguna representación de función parametrizada más compacta. Entramos en este tema más adelante.

Entonces, en el contexto de RL, el gran desafío que tiene el agente es aprender su política en base a la experiencia, buscando aproximarse a la política óptima. Esta búsqueda es un proceso de optimización donde se presenta el desafío de la explotación y exploración de todas las acciones posibles.

2.3.4. Exploración vs Explotación

La explotación y exploración del espacio de búsqueda son factores claves que deben considerarse para el buen desempeño de cualquier técnica de búsqueda y optimización global. Mientras la **explotación** (también conocida como búsqueda local o intensificación) guía la búsqueda teniendo en cuenta la información que se obtiene de las mejores soluciones encontradas hasta el momento, la **exploración** (también conocida como búsqueda global o diversificación) propicia descubrir regiones sin explorar y previene una convergencia prematura. Por ejemplo, en un problema de maximización puro, todo algoritmo que lo resuelva debe recorrer el espacio de soluciones buscando los máximos locales en procura de encontrar el máximo global. En este caso, explotar el conocimiento significa buscar el máximo local más cercano (por ejemplo con una técnica de descenso por gradiente), mientras que explorar el espacio significa escapar del óptimo local más cercano buscando en otra zonas otro óptimo quizás mejor (por ejemplo creando una solución distante con una heurística).

Este *trade-off* entre exploración y explotación del espacio también sucede en RL (a diferencia de otros tipos de aprendizaje de máquina), siendo uno de sus desafíos principales. En búsqueda de una mejor política, el agente debe explotar y explorar el espacio de estados/acciones. Donde, para obtener la mayor recompensa, el agente de RL debe preferir acciones que haya probado en el pasado y que hayan sido efectivas en obtener recompensas. Pero para descubrir estas acciones (o potencialmente otras mejores), tiene que probar nuevas acciones que no haya probado hasta el momento. Es decir, el agente tiene que **explotar** lo que ya sabe para obtener recompensas, pero también tiene que **explorar** para elegir mejores acciones en el futuro. Hay que balancear ambas para lograr el objetivo de maximizar el retorno. El agente tiene que probar una variedad de acciones, y progresivamente debe favorecer las que parezcan ser las mejores. En una tarea estocástica, cada acción debe ser probada varias veces para obtener una estimación fiable de la recompensa esperada. Hay algunas técnicas para atacar este desafío. En el Capítulo 3 describimos qué técnicas utilizamos para encarar este problema en nuestra solución.

Técnica $\epsilon - greedy$

Una de las técnicas de exploración/explotación más utilizadas es la técnica $\epsilon - greedy$. Esta técnica consiste en seguir una política el $(100 - \epsilon)\,\%$ de las veces (explotación), y tomar una acción al azar el restante $\epsilon\,\%$ de las veces (exploración) para un $\epsilon > 0$ pequeño. El ϵ puede ser fijo, pero también puede ser dinámico. El que sea dinámico es útil para variar el grado de exploración a lo largo del entrenamiento del algoritmo. Es común que al comienzo del entrenamiento se explore una mayor cantidad de veces, ya que aún no se tiene una política lo suficientemente buena para explotar, y a medida que la política va mejorando se va disminuyendo la exploración en favor de más explotación. Esta variante dinámica es usada en este trabajo como se detalla más adelante en la solución.

2.3.5. *General Policy Iteration*

La metodología general que sigue la mayoría de los algoritmos de RL para encontrar una política óptima se conoce como *General Policy Iteration* (GPI). GPI consiste en tomar una política inicial y una inicialización para la función de valor (podría ser al azar por ejemplo) y luego aplicar procesos de forma simultánea e interactiva, conocidos como *Policy Evaluation* (PE) y *Policy Improvement* (PI). PE se encarga de ajustar la función de valor a la política actual, y PI que se encarga de hacer la política *greedy* con respecto a la función de valor actual. Está probado que esta metodología converge a la política óptima y la función de valor óptima (para esta demostración se usa el *Policy Improvement Theorem*, detallado en los Capítulos 4 y 5 de Sutton y Barto, 2018).

La convergencia se da en el límite si hay exploración, es decir, cuando hay probabilidad efectiva de visitar todos los pares estado-acción. Para lograr esto se distinguen dos enfoques: *on-policy* y *off-policy*. Los métodos *on-policy* tratan de evaluar o mejorar la misma política que usan para tomar las decisiones, mientras que los métodos *off-policy* evalúan o mejoran una política distinta de la que usan para generar datos.

La forma más común de aproximar la función de valor cuando no se cuenta con la dinámica completa del sistema –a diferencia de lo supuesto en la Programación Dinámica por ejemplo– es a través de la experiencia. Ejemplos de esto son los métodos de Monte Carlo (MC) y los métodos *Temporal Difference* (TD). Estos métodos difieren en la forma en que hacen la tarea de PE. Mientras que los métodos MC toman datos de episodios completos y promedian los retornos para estimar la función de valor, los métodos TD no precisan esperar que se complete el episodio, sino que pueden actualizar paso a paso, o luego de alguna cantidad de pasos.

La regla general para la tarea de PE suele tener la forma:

$$NuevaEstimacin \longleftarrow ViejaEstimacin + Paso[Objetivo - ViejaEstimacin].$$

La expresión "$[Objetivo - ViejaEstimacin]$" es un error en la estimación. Se reduce tomando pasos hacia "$Objetivo$" de tamaño "$Paso$". El objetivo se asume que indica una dirección deseable en la que moverse. Los distintos métodos usan distintos objetivos como veremos a continuación.

Notaremos como V y Q a las estimaciones de las funciones de valor v_π y q_π respectivamente. Veamos las reglas de actualización para algunos métodos.

La regla de actualización para los métodos de MC es:

$$V(S_t) \longleftarrow V(S_t) + \alpha[G_t - V(S_t)],$$

donde G_t es el retorno total a partir del paso t (el "Objetivo" en este caso), por lo que es claro que es necesario que se complete el episodio para aplicar la regla, V es la estimación de la función de valor y α es el paso de aprendizaje.

El algoritmo TD más sencillo, conocido como TD(0), tiene la siguiente regla de actualización:

$$V(S_t) \longleftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)],$$

donde se puede aplicar esta regla al final de cada paso de tiempo, ya que usa otros valores estimados, sin necesidad de que termine el episodio. Además de esta ventaja, los métodos TD han demostrado converger más rápido que los de MC. Veamos dos de las variantes más populares de algoritmos que usan TD como PE (se incluyen por completitud, no utilizaremos estas variantes en nuestra solución):

1) Sarsa : *ON-POLICY TD*

Regla de Actualización:

$$Q(S_t, A_t) \longleftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)].$$

2) Q-Learning : *OFF-POLICY TD*

Regla de Actualización:

$$Q(S_t, A_t) \longleftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)].$$

En (Watkins, 1989) se presenta un popular algoritmo de control *OFF-POLICY TD*, donde la función que se aprende directamente aproxima la función de acción-valor óptima (q_*) independientemente de la política que se esté siguiendo. Este algoritmo es muy usado en la literatura. Más adelante mencionamos algunos trabajos que utilizan este algoritmo (o variantes del mismo) para resolver distintos problemas del sistema energético.

2.3.6. Aproximación de Funciones

Las técnicas presentadas hasta ahora se conocen como tabulares, ya que asumen que la función de valor es representada como una tabla con una entrada para cada estado ($V(S)$) o para cada par estado-valor ($Q(S, A)$), y donde se utiliza alguna de las reglas vistas para actualizar las entradas de esta tabla. Como mencionamos antes, en muchos de los casos en los que nos interesa aplicar RL, no es posible representar el espacio de estados de esta forma ya que el espacio de estados es muy grande o porque alguna de las variables que definen el estado es continua (como es el caso del problema que se resuelve en esta tesis). Entonces, se tiene un problema de generalización, ya que se podrán visitar solamente un subconjunto acotado de estados, y a partir de ellos se tiene que poder generalizar para estados que nunca hayan sido visitados.

En lugar de la representación tabular, se usan funciones parametrizadas para representar la función de valor:

$$\hat{v}(s, \mathbf{w}) \approx v_\pi(s),$$

en donde en vez de actualizar nuestra estimación de v como plantean las técnicas tabulares, lo que se va actualizando es el vector de parámetros $\mathbf{w}$. Por ejemplo, podríamos usar una red neuronal para representar $\hat{v}$ y entonces los pesos de la red neuronal sería el vector $\mathbf{w}$.

El problema de generalización a partir de ejemplos es usado en otras ramas de ML como el aprendizaje supervisado. Se le suele llamar aproximación de funciones porque toma muestras de la función deseada (función de valor por ejemplo), y trata de generalizar a partir de ellos para construir una aproximación de la función completa.

Una de las técnicas de aproximación de funciones más estudiada y utilizada es el descenso por gradiente, donde aprendemos el vector de parámetros $\mathbf{w}$ minimizando el error de aproximación cometido en los ejemplos observados, ajustando $\mathbf{w}$ dando un pequeño paso en la dirección que más reduce el error (o el cuadrado del error):

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{1}{2}\alpha\nabla[v_\pi(S_t) - \hat{v}(S_t, \mathbf{w}_t)]^2 = \mathbf{w}_t + \alpha[v_\pi(S_t) - \hat{v}(S_t, \mathbf{w}_t)]\nabla\hat{v}(S_t, \mathbf{w}_t),$$

siendo $\nabla\hat{v}(S_t, \mathbf{w}_t)$ el gradiente de la función $\hat{v}$ con respecto al vector de parámetros $\mathbf{w}$.

Por supuesto, como $v_\pi(S_t)$ es desconocido, se suele sustituir por el objetivo, como vimos en los métodos tabulares (por ejemplo G_t para los métodos de MC). Siendo V_t el objetivo, la regla general de los métodos de descenso por gradiente es:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[V_t - \hat{v}(S_t, \mathbf{w}_t)]\nabla\hat{v}(S_t, \mathbf{w}_t).$$

Para la función q, la regla general es:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[Q_t - \hat{q}(S_t, A_t, \mathbf{w}_t)]\nabla\hat{q}(S_t, A_t, \mathbf{w}_t).$$

Basándose en esta regla, existen variantes de aproximación de funciones mediante descenso por gradiente de los algoritmos Sarsa y Q-learning vistos. Dos familias de métodos de aproximación de funciones mediante descenso por gradiente se suelen utilizar en RL. La primera es usar redes neuronales y *back-propagation* (Rumelhart et al. 1986) para el cálculo de los gradientes, y la otra es usar funciones lineales. A continuación entramos en detalle en los métodos lineales, ya que serán utilizados para resolver el problema de esta tesis.

2.3.6.1. Métodos lineales

Los métodos lineales son un caso especial de aproximación de funciones por descenso por gradiente en los que la función $\hat{v}$ es una función lineal del parámetro $\mathbf{w}$. A cada estado s le corresponde un vector de *features:* $\mathbf{x}(s) = (x_1(s), x_2(s), ..., x_n(s))^\top$, con el mismo número de componentes que $\mathbf{w}$. En este caso, la función de estado aproximado tiene la forma:

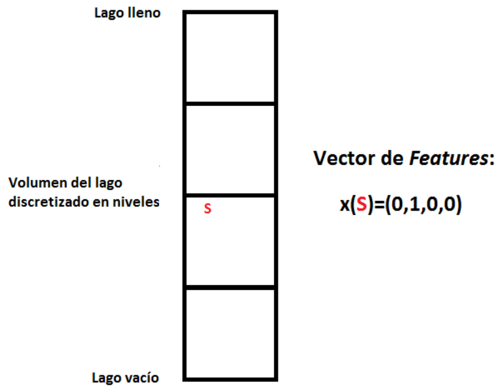
$$\hat{v}(s, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}(s) = \sum_{i=1}^{n} w_i x_i(s).$$

También es natural la forma del gradiente en este caso:

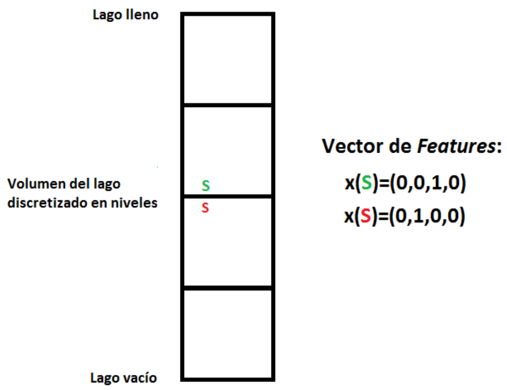
$$\nabla \hat{v}(s, \mathbf{w}) = \mathbf{x}(s).$$

2.3.6.2. Atributos (*Features*)

La selección de *features* es muy importante para agregar conocimiento del dominio a la tarea de RL. Los *features* deben representar características específicas de la tarea. Tomemos como ejemplo el problema que se resuelve en esta tesis, en donde las variables de estado más importantes son el volumen del lago en las represas. Una forma sencilla que pueden tomar los *features* (y la que se usó en los primeros experimentos en este trabajo), es simplemente discretizar el volumen del lago en niveles, y asignar a cada nivel un parámetro binario del vector $\mathbf{w}$. La Figura 2.7a muestra un esquema de esto con el vector de *features* asociado. Esta técnica tiene algunas desventajas, en la Figura 2.7b se puede ver el mismo ejemplo para otro estado del lago. Se puede apreciar que los vectores de parámetros son completamente distintos, y que el valor de ambos estados puede llegar a ser completamente distinto, a pesar de ser prácticamente el mismo estado. Incluso aunque el valor de ambos estados llegue a ser similar, llevará más tiempo de entrenamiento, ya que la visita a uno de los estados, no aporta ninguna información para el valor del otro.



(a) Un punto.



(b) Dos puntos.

Figura 2.7: Vector de *features* binarios para discretización del lago por niveles.

2.3.6.3. *Coarse Coding*

Para evitar estos problemas, y para generar *features* más potentes en general, se puede usar una familia de técnicas conocidas como *Coarse Coding*. La idea de estas técnicas se puede ver en la Figura 2.8. En la misma se ve un ejemplo de una variable de estado bidimensional, en donde se superponen varias *features* binarias circulares, y la idea es que cada punto del estado *activará* algunas de esas *features* (tomando valor 1) y otras no (tomando valor 0). Obviamente, estas *features* no tienen porqué ser circulares, y esto no solo se cumple en el caso de dos dimensiones. En nuestro caso, la altura del lago es unidimensional, y las *features* serían distintas porciones del nivel del lago que se superponen. Volviendo al ejemplo anterior, en este caso, los dos puntos vistos deberían activar varias *features*, lo cual ayudaría a resolver los problemas antes mencionados.

2.3.6.4. *Radial Basis Functions*

En este trabajo utilizamos para nuestras *features* una técnica conocida como *Radial Basis Functions* (RBFs). RBFs es una generalización de las técnicas de *Coarse Coding* a *features* continuas. En vez de ser *features* binarias que solamente toman valor 0 o 1, las *features* pueden tomar cualquier valor en el intervalo $[0, 1]$. Es decir, que en vez de activar o no las *features*, se representa el “grado de activación” de cada *feature*. La típica *feature* RBF es Gaussiana, y su grado de activación depende solamente de la distancia del estado al centro de la campana, y de la desviación estándar, ver la Figura 2.9 para el caso de aplicar RBF a una variable unidimensional.

La Figura 2.10 muestra el vector de *features* para el ejemplo original de la altura

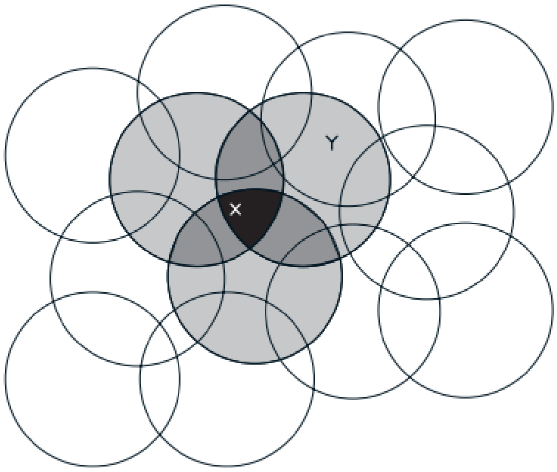
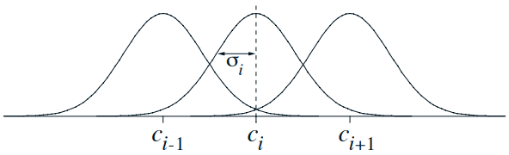


Figura 2.8: Ejemplo de *features* en *Coarse Coding*. Fuente (Sutton y Barto, 2018).



$$x_i(s) = exp\left(-\frac{||s - c_i||^2}{2\sigma_i^2}\right)$$

Figura 2.9: Vector de *features* usando técnica RBF para una variable unidimensional.

del lago, usando la técnica RBF.

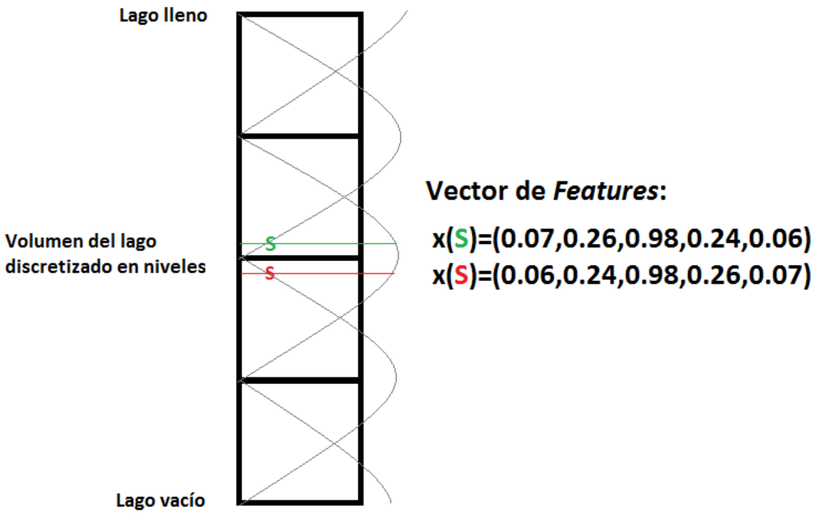


Figura 2.10: Vector de *features* usando técnica RBF para la altura del lago.

La cantidad de centros y la desviación estándar de las *features* RBF son hiper-parámetros a definir durante el entrenamiento. En el Capítulo 3 mencionamos las usadas para nuestro trabajo.

2.3.7. Métodos *Policy Gradient*

Todos los métodos vistos hasta ahora son lo que se conoce como métodos acción-valor. Este tipo de métodos aprenden la función de valor del estado o del par estado-acción (v, q) y luego usan estos valores directamente en la política (por ejemplo ϵ-*greedy*) para elegir las acciones.

En esta sección vamos a mencionar una nueva clase de métodos llamados *Policy Approximation Methods*. En lugar de aprender una función de valor, estos métodos aprenden directamente una política parametrizada que puede elegir acciones sin necesidad de consultar una función de valor (en realidad, algunos de estos métodos también aprenden una función de valor, hablamos de estos más adelante). En particular, los métodos que vamos a mencionar son los métodos *Policy Gradient*, ya que utilizan la técnica ascenso por gradiente para aprender los parámetros de la política.

Los métodos de aproximación de la política tienen algunas ventajas sobre los métodos acción-valor, como su capacidad de aprender una política estocástica óptima, cosa que los métodos acción-valor no tienen una forma natural de hacer. También, la política puede ser más sencilla de aprender que la función de valor (esto no siempre es así, depende del problema) y se puede acercar de forma asintótica a una política determinista. Además, elegir la parametrización de la política puede ser una buena forma de agregar conocimiento del dominio al sistema de RL. Y por último, estos métodos pueden manejar de forma natural espacios de acciones continuos ($a \in \mathbb{R}$), como es el caso del problema que se resuelve en esta tesis, en nuestro caso utilizamos una **política Gaussiana**, la cual puede definirse según:

$$\pi_\theta(a \mid s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-\left(a - \mu(s;\theta)\right)^2}{2\sigma^2}\right),$$

donde la media μ se encuentra parametrizada. y σ es la desviación estándar de la política (veremos más sobre esto en el Capítulo 3).

A continuación veremos tres métodos de aproximación de política: REINFORCE, REINFORCE con *baseline*, y *Actor-Critic*.

2.3.7.1. REINFORCE

El primero de estos algoritmos que vamos a mencionar se llama REINFORCE, y tiene la siguiente regla de actualización de los parámetros:

$$\theta_{i+1} = \theta_i + \alpha G_t \frac{\nabla \pi(A_t|S_t, \theta_t)}{\pi(A_t|S_t, \theta_t)}.$$

Como se puede ver es un método MC, ya que usa el retorno completo (G_t), lo que significa que tiene que terminar el episodio entero para aplicar la regla. Se puede notar que la actualización es directamente proporcional a G_t, lo cual tiene sentido ya que se quiere mover los parámetros en la dirección que más retorno generen, y que además es inversamente proporcional a la probabilidad de elegir esa acción ($\pi(A_t|S_t, \theta_t)$), lo cual tiene sentido, ya que no queremos que las acciones más probables tengan una ventaja injusta.

Para ver cómo se llega a la fórmula de REINFORCE y el algoritmo central del que deriva esta familia de métodos, *Policy Gradient Theorem*, referir al Capítulo 13 de Sutton y Barto, 2018.

2.3.7.2. REINFORCE con *baseline*

Está probado (Capítulo 13 de Sutton y Barto, 2018) que al *Policy Gradient Theorem* se puede generalizar para incluir una comparación de la función q_π con un *baseline* $b(s)$ arbitrario. Esta línea base de comparación ayuda a disminuir la varianza de estos métodos, sin cambiar el valor esperado, lo cual acelera el aprendizaje. En algunos estados, todas las acciones pueden tener valor alto, por lo que es útil tener una línea base alta para diferenciar los valores altos de los "un poco menos altos". A su vez, en algunos estados todas las acciones tendrán valores bajos, por lo que una línea base baja es de utilidad. El uso de un *baseline* en el algoritmo REINFORCE se traduce en la siguiente regla de actualización:

$$\theta_{i+1} = \theta_i + \alpha(G_t - b(S_t)) \frac{\nabla \pi(A_t|S_t, \theta_t)}{\pi(A_t|S_t, \theta_t)}.$$

Una de las opciones más naturales para el *baseline* es un estimado de la función de valor, $\hat{v}(S_t, \mathbf{w})$, siendo $\mathbf{w}$ un vector de parámetros que se aprenda por alguno de los métodos acción-valor vistos anteriormente. El algoritmo REINFORCE que use un *baseline* de esta forma sería entonces[1]:

$$\delta \longleftarrow G - \hat{v}(S_t, \mathbf{w}),$$

$$\mathbf{w} \longleftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t, \mathbf{w}),$$

$$\theta \longleftarrow \theta + \alpha^{\theta} \gamma^t \delta \nabla \ln \pi(A_t | S_t, \theta).$$

2.3.7.3. Métodos *Actor-Critic*

En el método REINFORCE con *baseline*, la función de valor estima el valor del primer estado en cada transición y se usa como *baseline* para el retorno que le sigue, pero esto pasa antes de seleccionar la acción, y por ende no se puede usar para evaluar la acción. En cambio, en los métodos conocidos como *Actor-Critic*, la función de valor se usa también para el segundo estado de la transición. Cuando al segundo valor se lo descuenta y se le agrega la recompensa, es un estimador útil del retorno (como se vio en los métodos TD), y por ende sí se usa para evaluar la acción. Cuando la función de valor se usa para evaluar acciones de esta manera se la conoce como **crítico**, a la política se le llama **actor**, y a los métodos en general se les llama ***Actor-Critic***. Si usamos TD(0) como método de aprendizaje para la función de valor, la regla de actualización queda:

$$\theta_{i+1} = \theta_i + \alpha(R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w})) \frac{\nabla \pi(A_t | S_t, \theta_t)}{\pi(A_t | S_t, \theta_t)}.$$

Así llegamos al algoritmo que utilizamos en este trabajo, el algoritmo *One-Step Actor-Critic* que se puede ver a continuación en el Algoritmo 3. En el Capítulo 3 entramos en detalle en qué representa cada parte del algoritmo para nuestro caso particular.

2.3.8. Pasar raya desde lo general a lo particular

Para dar cierre a la sección se repasa el conjunto de abstracciones que serán implementadas en esta tesis con el fin de resolver el problema de referencia. Como dijimos, el algoritmo de RL que se va a utilizar es el *One-Step Actor Critic*. Se usará una política Gaussiana; y para la función de valor se usará una función lineal en donde los *features* usarán la técnica RBF. Para la exploración se utilizará la técnica

[1]Observar que $\nabla \ln x = \frac{\nabla x}{x}$.

Algoritmo 3 One-Step Actor Critic

Input: parametrización diferenciable para la política $\pi(a|s,\theta)$
Input: parametrización diferenciable para la función de estado-valor $\hat{v}(s,\mathbf{w})$
Parámetros: learning rates $\alpha^{\theta} > 0$, $\alpha^{\mathbf{w}} > 0$

Inicializar los parámetros $\theta \in \mathbb{R}^{d'}$ y $\mathbf{w} \in \mathbb{R}^{d}$ (ej. a 0)
procedure ONESTEPAC($nEps$)
  for $i \leftarrow 1 \ldots nEps$ **do**
    Inicializar S (primer estado del episodio)
    $I \leftarrow 1$
    while S no es terminal **do**
      $A \sim \pi(.|S,\theta)$ ▷ Tomar acción A, observar S', R
      $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ ▷ si S' es terminal $\hat{v}(S',\mathbf{w}) \doteq 0$
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}}\delta\nabla\hat{v}(S,\mathbf{w})$
      $\theta \leftarrow \theta + \alpha^{\theta} I\delta\nabla \ln \pi(A|S,\theta)$
      $I \leftarrow \gamma I$
      $S \leftarrow S'$
    end while
  end for
end procedure

$\epsilon - greedy$ con ϵ dinámico. En el Capítulo 3 se entra en mayor detalle en todo lo utilizado.

2.4. RL en el Sistema Energético

A continuación entramos en detalle en las aplicaciones del RL en el sistema energético mencionados en la Sección 1.2.6.

2.4.1. RL para el Despacho Económico

Jasmin et al. 2011 utiliza RL para resolver el problema del despacho económico. El problema del despacho económico consiste en determinar la producción en cada generador para cubrir la demanda energética al menor costo. Este es un subproblema del resuelto en esta tesis de maestría. Mientras que esta tesis se enfoca en el despacho hidro-térmico a largo plazo durante varios pasos (semanas por ejemplo) y simplificado (pocos generadores, costos de generación lineales), en (Jasmin et al. 2011) se enfocan en el despacho económico de un solo paso, pero con una mayor cantidad de generadores, de costos no necesariamente lineales. Los autores prueban su solución en tres escenarios distintos, para mencionar un ejemplo, en uno de los escenarios se cuenta con 20 generadores con funciones de costos cuadráticos. El problema de despacho económico a resolver es el siguiente:

$$\begin{aligned} \min_{X} \quad & \sum_{i=1}^{N} C_i(D_i) \\ \text{s. a.} \quad & D_T - \sum_{i=1}^{N} D_i = 0 \end{aligned},$$

donde C_i es la función de costo de la unidad generadora i, D_i es la energía generada por esa unidad, y D_T es la demanda total a cubrir.

Los autores tratan el problema como un problema de decisión multifase. Para N generadores, hay $N-1$ fases, en donde en la fase i se decide cuánto generar con el generador i, quedando la generación del generador N como la demanda total a cubrir menos la energía generada por los primeros $N-1$ generadores. Para resolver esto, utilizan Q-learning, usando el costo de generación como recompensa, y utilizando la técnica ϵ-greedy para ayudar con la exploración. En términos de RL, el esquema general es el siguiente, en la fase i el sistema se encuentra en un estado en donde hay que cubrir una demanda D_i, siendo D_i la demanda total a cubrir menos lo generado por los $i-1$ generadores anteriores, y donde hay que decidir la generación para el generador i. Se toma la acción a_i (generación del generador i), se obtiene una recompensa r_{i+1} que es el costo de generar a_i en el generador i, y se transiciona a un estado en donde hay que cubrir $D_i - a_i$ y la próxima decisión a tomar es la generación del generador $i+1$. Las recompensas r_i son utilizadas en la regla de actualización del algoritmo de Q-learning, tal cual fue explicado en la Sección 2.3.5.

2.4.2. RL para el Almacenamiento Energético

Como se mencionó anteriormente, una fuente de energía es no-despachable cuando no hay control sobre cuándo o qué cantidad de energía se genera con ella. Las energías eólica y solar constituyen ejemplos notorios, ya que su producción es consecuencia inmediata de un recurso natural incontrolable. Aunque tampoco los aportes hídricos son manejables, las represas hidráulicas sí son despachables por disponer del lago como reservorio. Una posibilidad para ganar gestión en las fuentes no-convencionales es complementar un parque con baterías de gran porte, acumulando energía a conveniencia costo-beneficio para entregarla en otro momento. En las horas de alta generación no convencional su producción puede exceder la demanda o su entrega a la red puede ser económicamente inconveniente (i.e., valor *spot* muy bajo). La misma se puede almacenar en baterías para ser usada luego en momentos en que sea más conveniente.

La decisión de uso y/o almacenamiento en baterías lleva asociada una política de control óptima a determinar, un subproblema de aquel resuelto en esta tesis. Si bien no utilizamos baterías en los problemas que modelamos, se podría fácilmente extender nuestro trabajo en este sentido, debido a la analogía en la política de control de una batería y una represa; en ambos casos se decide cuándo almacenar energía y cuándo utilizarla. Existe también una analogía entre los aportes estocásticos a futuro que pueden llegar a la represa, y la radiación solar/viento que puede *llegar* a futuro a cargar la batería.

El RL es ampliamente usado en la literatura para la determinación de políticas de operación óptima de baterías. Un ejemplo es (Henze y Dodier, 2003), en donde los autores utilizan Q-learning para determinar la política óptima de utilización de un sistema fotovoltaico con batería. El sistema cuenta con una cierta demanda base (D_{base}) que se quiere cubrir siempre de ser posible (por ejemplo la de un sistema de refrigeración u otros sistemas críticos), y cualquier demanda por sobre la base de forma secundaria si es posible. Se tiene una generación fotovoltaica G_t horaria estocástica, la cual se puede distribuir entre ir directo a cubrir demanda, o almacenar en la batería. Se cuenta con una demanda D_t horaria estocástica siendo siempre $D_t >= D_{base}$, la energía almacenada en la batería B_t, y con la demanda cubierta en la hora t, $S_t = S_G + S_B$, siendo S_G la energía usada de G_t y S_B la energía usada de B_t. La generación horaria y el estado de la batería se usan para determinar el estado del sistema, mientras que las acciones son la cantidad de demanda cubierta por generación, y la cantidad de demanda cubierta por la batería (toda generación que no se usa para cubrir demanda es almacenada en la batería si es posible o descartada en caso contrario). En términos de RL, el esquema general es el siguiente, en el instante t el sistema está en el estado (t, G_t, B_t), se toma la acción (S_G, S_B) obteniendo una recompensa R_{t+1} y se transiciona a un estado ($t+1$, G_{t+1}, $B_{t+1} = B_t - S_B + \text{máx}\{0, G_t - S_G\}$). La recompensa es el opuesto de una función de costo que varía según si se cubrió D_{base} o no. Si se cubrió, el costo es el cuadrado de la demanda no cubierta, mientras que si no se cubrió D_{base}, el costo es el cuadrado de la demanda secundaria ($D_t - D_{base}$) más el cuadrado de la demanda base no cubierta por un factor de penalización.

Los autores comparan la política óptima obtenida mediante RL con una que llaman “Control Convencional de un Sistema Fotovoltaico” (CCSF), la cual consiste en cubrir la mayor cantidad de demanda posible con generación y almacenar el excedente en la batería; si la generación no es suficiente, entonces utilizar energía de la batería hasta cubrir la demanda horaria. Esta política puede funcionar si la batería es bastante grande en comparación con la demanda, pero es una política heurística, que no optimiza ningún índice de *performance*. Para los experimentos que realizaron, si bien la política CCSF cubre más demanda total ($77\,\%$) que la obtenida mediante RL ($66\,\%$), la política obtenida mediante RL logra cubrir la demanda base el $93\,\%$ del tiempo mientras que la política CCSF la cubre solamente un $79\,\%$ del tiempo.

2.4.3. *Demand Response* - Vehículos Eléctricos

Una forma de optimizar el uso de energía son las técnicas denominadas *Demand Response* (DR). El objetivo de estas técnicas es el de trasladar el uso de energía a períodos de baja demanda o a períodos de alta disponibilidad de energía renovable. Si bien ya vimos un ejemplo del uso de baterías para aprovechar la generación de fuentes renovables, trasladar demanda a períodos de alta disponibilidad de energía renovable puede reducir los costos y las pérdidas de energía asociadas al almacenamiento.

Dusparic et al. 2013 estudian un sistema compuesto por 9 casas que cuentan con un vehículo eléctrico en un barrio cubierto por un único transformador, utilizando datos de un experimento realizado con medidores inteligentes en Irlanda. Se cuenta con información de la carga actual del sistema, con el precio actual de la energía, y con un predictor de carga y costo de energía para las próximas 24 horas. El objetivo es determinar la política óptima de carga de los vehículos eléctricos que minimiza los costos, manteniéndose dentro de los límites del transformador, y asegurándose que los vehículos se cargan lo necesario. Para determinar cuánto es necesario cargarlos, los vehículos tienen asociada una distancia a recorrer en el día que varía entre 50 y 80 millas, lo cual hace que la carga necesaria sea entre $35\,\%$ y $50\,\%$. Cada vehículo cuenta con un agente de RL para su control, el cual decide cada 15 minutos si cargar el vehículo o no durante los próximos 15 minutos. Se cuenta además con la demanda base de las casas, la cual varía entre 0.8kW y 3kW según el momento del día. Los autores utilizan la técnica W-learning (Humphrys, 2000) para entrenar los agentes.

W-learning es una técnica basada en Q-learning, utilizada cuando el agente tiene varias políticas a seguir. En W-learning, cada política se implementa como un proceso de Q-learning separado con su propio espacio de estados. Usando W-learning, el agente aprende, para cada estado de cada política, qué sucede, en términos de las recompensas obtenidas, si la acción nominada por esa política no es aplicada. Esto es capturado en lo que se denomina el valor W; mientras más alto W, más importante es para esa política que se aplique la acción sugerida. El agente en consecuencia ejecuta la acción nominada por la política que vaya a sufrir la mayor pérdida, si no se sigue su acción nominada (es decir, la de mayor valor W).

En este problema, los autores proponen 3 políticas:

- Política 1: esta política tiene el objetivo de cumplir con la carga mínima necesaria de la batería del vehículo. La recompensa es: 500 puntos por alcanzar la mínima carga necesaria, 500 puntos si la batería está en un nivel mayor al que estaba en el instante de tiempo anterior, y -500 puntos si el nivel de la batería no es mayor al nivel del instante anterior.
- Política 2: esta política tiene el objetivo de asegurar que la carga total del transformador que suministra todo el sistema se mantenga dentro de los límites. El límite se establece para desalentar a que los vehículos carguen durante los picos de la demanda base. La recompensa es: 500 puntos si la carga está por debajo del límite establecido, 0 puntos si la carga está cercana al límite, y -500 puntos si se superó el límite.
- Política 3: esta política tiene como objetivo asegurar que los vehículos se carguen durante los períodos de menor demanda (y por ende menor precio de la energía). Tiene la información de la demanda actual y la predicción de la demanda de las próximas 24 horas. En cada instante se clasifica la demanda actual como "baja", "media", o "alta" relativamente a la demanda predicha para las próximas 24 horas. La recompensa es: 500 puntos si cargan en un instante de baja demanda, 250 si cargan en un instante de demanda media, y -50 si cargan en un instante de demanda alta.

Los autores crearon escenarios de evaluación combinando las 3 políticas, y además se tiene un escenario base de comparación que consiste en cargar los vehículos en el momento en que vuelven a la casa hasta que estén completamente cargados. En cada instante, cada política sugiere una acción, y luego, usando W-learning, el agente determina la acción a tomar.

Los autores muestran que el RL es una técnica efectiva para DR. Logrando bajar la carga durante los picos de demanda en un $\sim 33\,\%$, y aumentando el uso de los valles en un $\sim 50\,\%$, manteniéndose dentro de los límites del transformador, y alcanzando la carga mínima necesaria de los vehículos para cubrir su recorrido.

2.4.4. Sistemas CVAA

Otro uso que se le ha dado al RL en el sistema energético es para el control de sistemas de "Calefacción, Ventilación y Aire Acondicionado" (CVAA). Gran parte del consumo energético viene de edificios, en particular de edificios de oficinas. Dentro de estos, un $\sim 40\,\%$ del consumo suele provenir de los sistemas de CVAA, por lo que la optimización del uso de estos sistemas es de gran importancia, y es un tema extensamente estudiado en la literatura.

El objetivo principal de los sistemas de CVAA es el de mantener un confort de temperatura para los ocupantes del edificio. Si bien ese confort es algo subjetivo, se han desarrollado algunos estándares para evaluarlo, por ejemplo el *Predicted Mean Vote* (PMV). El PMV evalúa el confort de una zona en una escala estándar para un grupo grande de personas.

Li et al. 2015 utilizan Q-learning para determinar la política óptima de control del sistema de CVAA de un edificio para mantener el confort (evaluado por PMV) en sus distintas zonas. Si bien el PMV se calcula teniendo en cuenta varios factores, como la temperatura del aire, la humedad, la velocidad del aire, y hasta factores como qué tan abrigada está la persona, se suele caracterizar con una escala que va de -3 a 3, siendo 0 el valor de máximo confort, valores cada vez más negativos cuando la sensación es cada vez más fría, y valores cada vez más positivos cuando la sensación es cada vez más calurosa. Dentro de los factores que afectan el PMV, los dos que se pueden controlar en los sistemas de CVAA son la temperatura, y la velocidad del aire. El objetivo es pues, llevar (y mantener) a una zona de un edificio a un PMV lo más cercano a 0 de forma óptima.

En términos de RL, como estado se utiliza el PMV, en el rango $[-1.0, \ldots, 1.3]$ discretizado de a 0.1. El espacio de acciones es bidimensional, por un lado la temperatura que cuyos posibles valores son $\{21, 22, \ldots, 30\}$, y por otro lado la velocidad del aire cuyos posible valores son $\{0, 0.05, \ldots, 0.5\}$. La recompensa sigue la fórmula:

$$r = \begin{cases} 0 & \text{si} \quad |PVM| < 0.1 \\ -200(|PVM| - 0.1)^2 & \text{si no} \end{cases},$$

donde se puede ver claramente que el objetivo es mantener el PMV en el rango

$[-0.1, 0.1]$, y que desviaciones de este rango en cualquier sentido son penalizadas de forma cuadrática según cuánto se alejan.

El paso de tiempo del experimento es de 1 minuto, y el sistema de CVAA está prendido durante 11 horas, por lo que cada episodio tiene 660 pasos. Para realizar el experimento los autores utilizan una herramienta llamada EnergyPlus, que les permite modelar un edificio con su sistema de energía y sistema de CVAA. En particular en este caso es un edificio de dos pisos con dos zonas cada uno al que se le ingresan todas las características necesarias, como la superficie, la cantidad de ventanas, cantidad de ocupantes, horario en el que es ocupado, etc.

Además de Q-learning, los autores utilizan el algoritmo *Multi-samples in Each Cell* (MEC) que es parte de un grupo de algoritmos llamados *Probably Approximately Correct* (PAC). Como a veces el espacio de estados y acciones es muy grande, el entrenamiento de un agente usando Q-learning puede ser muy lento y por ende el aprendizaje puede ser ineficiente. Para resolver esto se desarrollaron los algoritmos PAC. La idea es que aseguran una política cercana a la óptima en tiempo polinomial, y por ende se les suele llamar algoritmos de aprendizaje eficiente.

Si bien los autores prueban que tanto Q-learning como MEC son métodos efectivos para resolver el problema, comparan los resultados de ambos métodos para demostrar las ventajas de MEC. Los resultados son que Q-learning converge en 57 episodios con un Q-value de -139.5, mientras que MEC converge en 33 episodios con un Q-value de -123.7, es decir, que MEC converge más rápido y a un mejor valor. Una vez entrenadas ambas políticas, se prueban ambas partiendo de un PMV de 0.75 y se ve que la política MEC llega a la zona de confort en 4 minutos, mientras que la política Q-learning demora 12 minutos en alcanzar la misma zona.

2.4.5. Resumen de los trabajos relacionados en el sector energético

Agregamos las Tablas 2.1 y 2.2 a modo de resumen de algunos de los aspectos principales de los trabajos relacionados, que utilizan RL en sistema energético, presentados en esta sección:

Trabajo	**Sistema Energético**	**Algoritmo RL**	**Selección de Acciones**	**Subproblema o Análogo del nuestro**
Jasmin et al. 2011	Generación Térmica	Q-learning	ϵ-greedy	SI
Henze y Dodier, 2003	Baterías Generación Fotovoltaica	Q-learning	Soft-Max	SI
Dusparic et al. 2013	Vehículos Eléctricos	W-learning	n/a	NO
Li et al. 2015	CVAA	Q-learning MEC	MEC	NO

Tabla 2.1: Resumen de trabajos que usan RL en el sistema energético

Trabajo	**Objetivos**	**Horizonte/ Paso de Tiempo**	**Dinámica del sistema**
Jasmin et al. 2011	Costo Energético	Paso Único	Determinista
Henze y Dodier, 2003	Costo Energético Satisfacer Demanda Base	1 año/1 hora	Estocástica
Dusparic et al. 2013	*Demand Response* Costo Energético Satisfacer Carga Satisfacer Transformador	24hrs/15min	Estocástica
Li et al. 2015	Confort *Demand Response*	11hrs/1min	Determinista

Tabla 2.2: Resumen de trabajos que usan RL en el sistema energético

Capítulo 3

Iteración 1 - Aportes Hídricos Estocásticos con Demanda Determinística

En este capítulo resolvemos un caso particular del problema general de despacho presentado en la Sección 1.1. El capítulo comienza con una descripción específica de nuestra instancia del problema en la Sección 3.1, en donde a su vez se describe un generador de aportes hidrológicos sintéticos que será usado a lo largo de esta tesis. En las Secciones 3.2 y 3.3 se resumen el conjunto de aportes de referencia obtenidos del generador, y los algoritmos utilizados para evaluar y comparar distintas políticas que resuelven el problema. Como primeros valores de interés, en la Sección 3.4, determinamos cotas superior e inferior a la solución de la instancia del problema que se resuelve en este capítulo. Luego pasamos a resolver el problema mediante Programación Dinámica en la Sección 3.5, lo cual es de interés para tener una referencia con que comparar la solución que se obtendrá mediante RL. En la Sección 3.6 hacemos dicha solución mediante RL, y el capítulo culmina en la Sección 3.7 en donde se presentan algunos detalles más técnicos de los experimentos realizados y se detallan los resultados obtenidos.

3.1. Descripción del Problema

El problema de referencia es un despacho hidro-térmico con tres unidades de generación: dos térmicas de arranque rápido y una represa con embalse. La condición *rápida* de las unidades térmicas (arranques antes de 15 mins.) las ajusta a una hipótesis de trabajo en esta tesis: el ignorar restricciones técnicas en el proceso de arranque, tiempo de operación o parada de esas unidades (los *commitments*).

Adicionalmente y por simplicidad, se asumirá en esta aproximación que la curva de producción (generación vs. consumo) es lineal y sin mínimo técnico. Cada unidad (denominadas `T1` y `T2`) queda entonces determinada solamente por dos parámetros, a saber: la potencia máxima (M^T) y el rendimiento (c^T). Los valores de referencia para estos parámetros se presentan en la Tabla 3.1.

Unidad	Potencia Máxima	Rendimiento
`T1`	250 MW	4000 USD/MWh
`T2`	250 MW	100 USD/MWh

Tabla 3.1: Parámetros de las unidades térmicas utilizadas.

La única variable de control para una unidad térmica es la potencia generada por esa unidad en cada instante. Al estar discretizado el horizonte temporal, se tomará como referencia la energía ET_{kt} entregada en el intervalo t por la unidad k. Se asume que la potencia instantánea se mantiene estable en términos relativos en torno a su valor medio.

Para la unidad hidráulica también se ha tomado un modelo de referencia simple, con curva de producción lineal y sin mínimo técnico. Los parámetros en este caso se muestran en la Tabla 3.2.

Unidad	Volumen Máximo	Turbinado Máximo	Coeficiente Energético
`H1`	8200 hm^3	680 m^3/s	0.19 MW/$\frac{m^3}{s}$

Tabla 3.2: Parámetros de la unidad hidráulica utilizada utilizadas.

Observar que la potencia máxima de la unidad hidráulica es 129.2MW y que partiendo de la cota máxima, el tiempo de vaciado de la central es 19.94 semanas a potencia plena y sin nuevos aportes incrementales.

Finalmente, la Tabla 3.3 muestra el resto de los parámetros elegidos para el problema a resolver en la Iteración 1.

Horizonte de Tiempo	Paso de Tiempo	Demanda	Volumen Inicial Lago
2 años	Semanal	350 MW	4100 hm^3

Tabla 3.3: Parámetros de la Iteración 1

El problema a resolver en esta aproximación es cómo confeccionar una política de despacho optimizado para que este parque generador atienda una demanda fija (determinística), cuando los aportes hidrológicos al embalse son inciertos. Se obtendrán políticas solución mediante distintas técnicas de optimización, y se evaluarán y compararán usando un conjunto de aportes de referencia descrito más adelante en la Sección 3.2.

3.1.1. Modelo de los aportes

Aun cuando la aproximación tiene un objetivo exploratorio: *tener una referencia cuantitativa para la eficacia del uso de Reinforcement Learning en un problema de despacho hidro-térmico simplificado*, se ha buscado que algunos componentes tengan proximidad con la realidad nacional, en particular, los que hacen a los parámetros de la unidad hidráulica. En este experimento se han tomando parámetros públicos de la Represa de Rincón del Bonete.

Aunque los valores desagregados son confidenciales y su distribución es desconocida, sí son abiertas ciertas métricas derivadas de ellos, que surgen de 110 años de mediciones en el cuenca del Río Negro. La Tabla 3.4 presenta algunos valores de referencia para la variable aleatoria *aportes*.

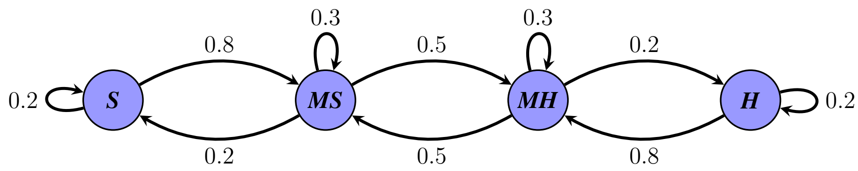
Aportes [m^3/s]	**Promedio**	**Recorrido**
Semanales, año más seco	53.42	–
Semanales, año más húmedo	1790.94	–
Semanales Est1 (ene-mar)	265.49	0-6063
Semanales Est2 (abr-jun)	693.79	0-14335
Semanales Est3 (jul-set)	924.87	0-6623
Semanales Est4 (oct-dic)	551.03	0-5226

Tabla 3.4: Datos históricos de aportes hidrológicos.

Entre los datos presentes en la Tabla 3.4, se destaca en primer lugar que el promedio semanal del año más húmedo de la muestra es aproximadamente 34 veces mayor al del año más seco. Esto indica claramente que no es recomendable asumir una distribución similar de un año a otro. Otro elemento conocido hace a las diferencias por la distribución estacional de las lluvias, siendo en general más secos los veranos y más húmedos los inviernos. Los datos a los que se pudo acceder en este experimento no están agregados por las clásicas estaciones, sino por trimestres a partir del 1ro de enero, según figura en la tabla. En lo que resta, nos referiremos como una estación a cada uno de estos cuatro trimestres.

Aunque valiosos, los datos anteriores son insuficientes para caracterizar los aportes hidrológicos. A efectos de confeccionar el generador de muestras, se han agregado hipótesis adicionales:

- Se asumen cuatro estados hidrológicos posibles para un año cualquiera, a saber: $E = \{S, MS, MH, H\}$, que corresponden respectivamente a: Seco, Medio Seco, Medio Húmedo y Húmedo. La dinámica de transiciones entre esos estados tiene paso anual y se ajusta a una Cadena de Markov de probabilidades conocidas.



- A estado hidrológico $h \in \{S, MS, MH, H\}$ conocido y estación $e \in \{1, 2, 3, 4\}$ dada (recordar que las estaciones son los trimestres), se asume que la distribución de la variable aleatoria aportes semanales $X \geq 0$ tiene densidad $f_{he}(x) = a_{he}^2 \cdot x \cdot e^{-a_{he}x}$, cuyo valor esperado es $E[X] = 2/a_{he}$. Esta distribución siempre permite algo próximo a 0 como valor probable, permite a su vez valores altos (aunque como eventos más raros), y queda caracterizada por un único parámetro, que se deriva simplemente a partir del valor esperado.
  El defecto de la distribución anterior es que a mayor valor esperado, mayor probabilidad de encontrar valores altos. Esto va en contra de los datos de la Tabla 3.4, donde hay registros parecidos para los aportes máximos (en el orden de 6000) en las estaciones 1, 3 y 4, pero se despega la 2 (más de 14000). Eso va a contrapelo de los aportes esperados por estación, donde la más alta es la 3. Nos faltan datos para hacer un análisis de eventos raros como son los picos. El ajuste en los valores esperados de la Tabla 3.4 busca capturar mejor estos eventos extremos.
  Surge de los datos en la Tabla 3.4 que el promedio semanal de aportes en un año cualquiera es aproximadamente 609m^3/s (omitimos unidades de acá en más). Tomando 53 como referencia del valor esperado para un año seco, 1791 para uno húmedo, y asumiendo que uno medio húmedo tiene el doble de aportes que uno medio seco, concluimos[1] que los aportes semanales esperados según el estado hidrológico son: 53, 354, 708 y 1791 respectivamente.
  A estado hidrológico dado, asumimos que el valor esperado de aportes entre estaciones sigue la proporción de la Tabla 3.4. Así, la matriz con la esperan-

[1] Se usó que la distribución estacionaria es [0.1, 0.4, 0.4, 0.1]

za de los aportes semanales para cada combinación de estado hidrológico y estación es:

	Est1	**Est2**	**Est3**	**Est4**	Prom.
S	23.1	60.4	80.5	47.9	53.0
MS	154.3	403.2	537.4	320.2	353.8
MH	308.6	806.3	1074.9	640.4	707.6
H	780.5	2039.7	2719.0	1620.0	1789.8
Esp.	265.5	693.8	924.9	551.0	

Tabla 3.5: Aportes semanales esperados según estado hidrológico y estación.

Cabe aclarar que para simplificar los experimentos (debido a que se reduce la dimensionalidad de los estados), a lo largo de todo este trabajo, se utilizaron datos solamente del estado hidrológico MS (Medio Seco). Además, se eligió el estado MS, ya que la historia necesaria para el ajuste (42 años; ver la Sub-sección 3.1.2-Tamaño de una Muestra Representativa) es compatible con la disponibilidad de registros históricos (110 años), no así para los estados S y H.

3.1.2. Simulaciones de instancias para entrenamiento y validación

Tanto para validar las políticas obtenidas mediante programación dinámica (Sección 3.5.3) y RL (Sección 3.6), como para entrenar el algoritmo de RL (Sección 3.6), es necesario recurrir a la realización de simulaciones. Para esto es necesario contar con un generador de aportes sintéticos. Se detalla a continuación el generador que se desarrolló.

Generador de Aportes Sintéticos

Recordemos que el modelo de aportes antes mencionado tiene la siguiente función de densidad: $f_{he}(x) = a_{he}^2 \cdot x \cdot e^{-a_{he}x}$, cuyo valor esperado es $E[X] = 2/a_{he}$.

La función generadora surge de sortear una uniforme [0,1], y luego aplicarle la inversa de la distribución de probabilidad (ver Sec-2.1). Dado que la distribución no tiene inversa analítica se estima mediante el método de Newton-Raphson. Las cuentas serían: la densidad es $f_{he}(x) = a_{he}^2 \cdot x \cdot e^{-a_{he}x}$, entonces la distribución es $F_{he}(x) = 1 - (a_{he} \cdot x + 1)e^{-a_{he}x}$. Dado una variable $r \sim unif(0, 1)$, buscamos x tal que $r = F_{he}(x)$, esto lo hacemos buscando el cero de $g(x) = F_{he}(x) - r$

mediante Newton-Raphson, es decir, con la iteración $x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)} = x_n + \frac{(a_{he} \cdot x_n + 1)e^{-a_{he}x_n} - r + 1}{a_{he}^2 \cdot x_n \cdot e^{-a_{he}x_n}}$. Con 10 iteraciones es suficiente para aproximar el valor.

Tamaño de una Muestra Representativa

En esta sección analizamos la cantidad de datos históricos necesaria para estimar los parámetros de una distribución como la anterior. Como se verá más adelante, en las soluciones por PD-estocástica (SDP) y RL, nos aseguramos de crear muestras de tamaño mayor al representativo a la hora de validar estos métodos.

En primer lugar, buscamos el tamaño de una muestra para caracterizar la distribución de aportes semanales en una estación cualquiera de un estado hidrológico cualquiera. Dado un estado/estación, la densidad de referencia es $f(x) = a^2 \cdot x \cdot e^{-ax}$, cuyo valor esperado es $\mu = E[X] = 2/a$, con varianza $\sigma^2 = VAR[X] = 2/a^2$. Usamos el promedio $A_n = \frac{X_1 + X_2 + \cdots + X_n}{n}$ como estimador de μ, asumiendo n muestras semanales de aportes (i.e. n muestras de la v.a. X), independientes e idénticamente distribuidas (i.i.d.).

Del teorema central del límite se deriva que se debe cumplir

$$n \geq Z_{NC}^2 \cdot \frac{\sigma^2}{tol^2},$$

donde: tol es el nivel de error máximo aceptado, σ^2 es la varianza de la v.a. X y Z_{NC} es el valor z tal que:

$$NC = \frac{1}{\sqrt{2\pi}} \int_{-z}^{z} e^{-x^2/2} dx.$$

Así, si buscamos un error máximo de 5 % (i.e. $tol = 0.05\mu$), y un nivel de confianza de 90 % ($NC = 0.9$), debe ser $Z_{NC} = 1.645$, de donde:

$$n \geq Z_{NC}^2 \cdot \frac{\sigma^2}{tol^2} = Z_{NC}^2 \cdot \frac{2/a^2}{(0.05 \cdot 2/a)^2} = \frac{1.645^2}{2 \cdot 0.05^2} = 541.205,$$

de donde $n = 542$ semanas, independientemente de cuál sea el estado hidrológico y la estación. Como en un año hay 13 semanas de cada estación, se necesitan unos 42 años para caracterizar razonablemente las distribuciones de un estado hidrológico cualquiera. Finalmente, como los estados S y H aparecen en media un 10 % de las veces cada uno, se necesitaría un histórico de más de 420 años para caracterizar estadísticamente el proceso. Hay que tener en cuenta que el uso de técnicas estadísticas para caracterizar los estados hidrológicos en sí mismos eleva ese número.

3.2. Conjunto de aportes de referencia

Con el Generador de Aportes Sintéticos descrito en la sección anterior (Sec-3.1.2), se genera un conjunto de aportes de referencia que será usado para evaluar y comparar las distintas políticas solución del problema que se obtengan mediante las distintas técnicas de optimización.

Sin pérdida de generalidad y a los solos efectos de mantener la complejidad de los modelos contra los que se compara bajo control (i.e., los clásicos), se ha decidido generar la serie de aportes en base a un único estado hidrológico, el Medio Seco (MS). Así, los valores estacionales esperados para los aportes se toman como en la segunda fila de Tabla-3.5. Las estaciones Seca y Húmeda son poco representativas de la realidad y no se cuenta con históricos suficientes para ajustarlas. Entre las dos más frecuentes (i.e., MS y MH), se entendió que la Media Seca ponía más estrés a la optimización del sistema hidrotérmico, lo que resulta más interesante.

El conjunto de referencia cuenta con 2000 años de aportes los cuales representan 1000 escenarios de nuestro experimento, ya que el horizonte de tiempo que se quiere optimizar es de 2 años. Cuando se quiere evaluar una política, se hace la simulación de los 1000 escenarios, y se promedia el costo de operación obtenido. Este valor es el que se utiliza para comparar dos políticas entre sí. En la Sección-3.1.2 se había concluido que –a estado hidrológico dado– un histórico de 42 años permitía estimar el valor esperado de los aportes estacionales con un error de 5 % para un nivel de confianza de 90 %. El tamaño de la muestra en esta simulación es notoriamente mayor y sería suficiente para alcanzar un 0.5 % de error para el mismo nivel de confianza, o estaría próximo a 1 % para un nivel de 95 %.

3.3. Método de evaluación y comparación de soluciones

Lo que se quiere hallar en este capítulo es la solución al problema de despacho presentado en la Sección 3.1. Recordar que esta solución es una política π que determina el control (turbinado) según el estado en el que se encuentra el sistema. En esta sección presentamos los algoritmos utilizados para evaluar y comparar distintas políticas solución del problema los cuales son independientes del método utilizado para obtener la política. Más adelante en el capítulo se aplican distintos métodos (PD, RL) para obtener políticas solución.

Los algoritmos están presentados con parámetros genéricos, ya que si bien tenemos parámetros fijos para la Iteración 1 (por ejemplo $T = 2$ años y $D = 350MW$), los algoritmos funcionan para cualquier parámetro que queramos.

Comenzamos por la función $cgen()$ que calcula el costo de operación del sistema en una semana (nuestro paso de tiempo). La misma se puede ver en el Algoritmo 4. Cabe destacar que este mismo algoritmo funciona para el caso en que la demanda no sea constante, sino estocástica y por ende tomando valores distintos cada semana. Simplemente habría que transformar la demanda constante a un parámetro del algoritmo:

Algoritmo 4 Pseudocódigo Costo de Generación

```
Input: coeficiente energético unidad hidráulica CE
Input: potencia máxima unidades térmicas MT1, MT2
Input: costo unidades térmicas CT1, CT2
Input: turbinado en m^3/s Tur
Input: demanda en MW D

procedure CGEN(Tur)
    PH = CE * Tur                         ▷ Potencia hidráulica [MW]
    DTer = D − PH                         ▷ Demanda a cubrir con Térmica
    if DTer < MT2 then
        costo = DTer * 24 * 7 * CT2
    else
        costo = MT2 * 24 * 7 * CT2 + (DTer − MT2) * 24 * 7 * CT1
    end if
end procedure
```

Como se puede ver, el algoritmo recibe el turbinado de la semana, el cual se usa junto con el coeficiente energético para calcular la potencia hidráulica. Teniendo esto, se puede calcular cuánto de la demanda falta por cubrir con térmica. Sabiendo este valor, se calcula y retorna el costo térmico, usando primero la térmica barata, y recurriendo a la cara si la barata no es suficiente (recordar que en este caso tenemos dos unidades térmicas, pero este algoritmo es trivialmente generalizable a cualquier cantidad de unidades térmicas).

Teniendo la función $cgen()$, podemos calcular ahora el costo de un escenario entero. Por escenario nos referimos a una realización de aportes para nuestro horizonte de tiempo (en este caso particular un vector de 104 aportes, uno por cada semana de los dos años). La función $costoEsc()$ se puede ver en el Algoritmo 5.
El algoritmo recibe la política y el vector de aportes, y calcula el costo del escenario

Algoritmo 5 Pseudocódigo Costo Escenario

Input: volumen inicial del lago V_{ini}
Input: horizonte de tiempo en años T
Input: vector de aportes semanales AP
Input: política de operación π
Input: cantidad de segundos en la semana $SegXSem$
Input: función que calcula el turbinado máximo posible según el volumen de agua $maxTur$

procedure COSTOESC(π, AP)
    $V = V_{ini}$
    $costo = 0$
    for $t \leftarrow 52 * T \ldots 1$ **do**
        $V = V + (AP(t) * SegXSem)$
        $Tur \leftarrow \pi$ ▷ Obtener Turbinado de la política
        $costo = costo + CGen(Tur)$
        $V = V - (Tur * SegXSem)$
    end for
end procedure

entero. La forma en que lo hace es partir de un volumen inicial para el lago, y operar el sistema según dicte la política semana a semana, calculando el costo de cada semana con la función $cgen()$, y actualizando el volumen del lago según lo turbinado y el aportes para la semana. El costo total es la suma de todos los costos semanales. Cabe aclarar que cuando se calcula el costo en este tipo de problemas, es común usar un factor de descuento semanal (γ) entre 0 y 1, de forma de amortizar el mismo. En los experimentos realizados en este trabajo se utilizó $\gamma = 1$, y es por esto que no aparece en el algoritmo anterior (Análogamente, en RL tenemos un factor de descuento γ para las recompensas, como se vio en la Sección 2.3.3 y como se ve presente en el Algoritmo 3 utilizado en nuestra solución RL).

Por último llegamos al Algoritmo que utilizamos para evaluar y comparar políticas. El mismo se puede ver a continuación en el Algoritmo 6.

El algoritmo recibe la política y hace uso del conjunto de escenarios de aportes de referencia, y simplemente calcula el costo para cada escenario del conjunto de referencia utilizando la función $costoEsc()$ y retorna el promedio de estos costos. Este es el valor que se usa para evaluar la política.

Algoritmo 6 Pseudocódigo Evaluar Política

Input: política de operación π
Input: conjunto de escenarios de aportes de referencia Ref

procedure EVAL(π)
    $costo = 0$
    for all $AP \in \mathcal{R}ef$ **do**
        $costo = costo + costoEsc(\pi, AP)$
    end for
    $costo = costo/size(Ref)$
end procedure

3.4. Cotas del problema

El contar con valores óptimos de referencias para instancias específicas de un problema de optimización, así como características de las soluciones correspondientes, constituye un elemento muy importante a la hora de diseñar un algoritmo general para optimizar un problema complejo, como es el problema desarrollado en este trabajo. Es por lo anterior que la aproximación metodológica/científica elegida incluye como primer paso la generación de multiplicidad de resultados de referencia basados en el uso de técnicas de optimización tradicionales.

La elección del problema base en esta sección se realizó buscando cercanía con el problema general, al tiempo que se identificaban algoritmos conocidos para su resolución. Se atacó el problema en etapas de complejidad creciente. Los primeros resultados buscan cotas de referencia para una versión determinística del problema. Se parte de la observación que si en lugar de ser una v.a., los aportes hidrológicos fueran valores constantes en cada estación e iguales al valor esperado correspondiente, el problema sería determinístico y su valor óptimo sería mejor que el valor esperado del problema completo, donde los aportes son inciertos. Pensar simplemente que si el problema fuera un juego de azar, pueden conseguirse mejores resultados conociendo los valores del sorteo de antemano.

La aproximación determinística propuesta tiene solución exacta y extremadamente eficiente cuando se usan valores continuos para el estado y el control. Sin embargo, no es adecuada para integrar la incertidumbre propia del problema real, hecho que había sido observado en la Sección-1.3 y que influyó en la estrategia elegida.

El problema puede formularse para ser resuelto por Programación Dinámica Estocástica, a costo de la discretización necesaria en los estados y el control, lo que introduce inevitablemente un error de modelado. Para mantener ese error bajo control, se ajustaron los estados y el control en primer lugar para un algoritmo de Programación Dinámica (determinístico), usando como referencia la solución exacta del primer paso. Ajustado ya el error de discretización por debajo de un valor aceptable, se extendió el algoritmo a uno de Programación Dinámica Estocástica para integrar la aleatoriedad. Todo este proceso está desarrollado en detalle en las siguientes secciones.

Finalmente, se utiliza el conjunto de aportes de referencia descrito en la Sección 3.2 para evaluar y comparar los resultados obtenidos mediante Programación Dinámica Estocástica y *Reinforcement Learning*.

Los detalles del proceso se elaboran en las siguientes secciones. Se deja nota que el proceso elegido no sólo fue de suma importancia para ajustar las variantes en la implementación del algoritmo de RL mismo y sus hiper-parámetros, sino que fue vital para identificar *bugs* en los programas, que al no generar *crashes* hubieran pasado inadvertidos de no ser por la disparidad en los resultados.

3.4.1. Cota Superior usando técnica *Greedy*

De acuerdo a la Definición 2 de un problema general de minimización, el óptimo es el valor más bajo posible entre los puntos factibles de la instancia a resolver. Es inmediato entonces que el valor de un punto factible cualquiera fija una referencia o **cota superior** para el óptimo del problema.

Se propone que el cálculo de esa cota superior sea el resultado de resolver el problema mediante una heurística que denominamos *técnica Greedy*. La idea detrás de la heurística es usar en cada etapa toda la generación hidráulica posible, recurriendo a la generación térmica solamente cuando es inevitable. Al ser la demanda fija y menor que la capacidad térmica instalada, ella puede cubrirse incluso cuando no haya recurso hidráulico explotable. Así, la construcción lograda con la técnica *Greedy* es claramente una solución factible y se cumple que el resultado obtenido con este algoritmo es mayor o igual al óptimo.

Como se adelantó en la introducción al problema de despacho (Sección 1.1), se asume que la solución óptima pasa por administrar inteligentemente el agua para poder usarla en el futuro y evitar costos mayores. El cómo hacerlo es de momento incierto. De ahí el valor práctico de contar con una referencia o cota para un valor desconocido al momento.

La técnica *Greedy* es entonces aquella cuya política es: $\pi = maxTur$, es decir turbinar siempre al máximo posible en cada instante. Para obtener el resultado simplemente aplicamos el Algoritmo 6 a esta política y nuestro conjunto de referencia.

El resultado obtenido mediante la *técnica Greedy* es:

COTA SUPERIOR: 2586.79 MUSD (millones de dólares americanos)

3.4.2. Cota Inferior usando Programación Lineal (LP)

Como se mencionó anteriormente, nuestro problema es estocástico en los aportes hidrológicos, sin embargo, si se toma el problema como determinístico utilizando las medias de los aportes en las distintas estaciones, se puede tratar el problema como uno de Programación Lineal (LP). Los aportes son estacionalmente fijos (recordar que estamos siempre en el estado hidrológico MS) y son 154.3 para el trimestre 1, 403.2 para el trimestre 2, 573.4 para el trimestre 3, y 320.2 para el trimestre 4 según la Tabla 3.5.

La **cota inferior** es el resultado de resolver una versión determinística del problema mediante Programación Lineal (LP) como se muestra en las siguientes secciones. Podemos tener certeza que el valor es una cota inferior porque el resultado es exacto (i.e. se usa un algoritmo exacto para resolver un LP) para una versión del problema que usa los valores esperados de los aportes como si fueran valores conocidos a-priori (i.e., antes de la optimización).

Adelantamos que el resultado en este caso es: **COTA INFERIOR: 1862.37** MUSD, para luego elaborar los detalles del modelo que permiten llegar a ese valor.

La resolución del problema mediante LP la vamos a desarrollar en tres partes, tomando como referencia el problema general de despacho presentado en la Eq.1.1 e instanciándolo para nuestro caso determinista. La primera parte entra en todo lo relacionado con la gestión del lago, mientras que la segunda parte entra en todo lo relacionado a los costos de producción. Estas dos partes mencionan todos los componentes necesarios que constituyen nuestra formulación LP (variables de control, función objetivo, restricciones, etc.). Por último, la tercera parte contiene la formulación LP en sí misma, y algunos detalles sobre la solución.

Gestión del Lago

Recordemos la formulación general del problema de despacho presentado en la Eq.1.1 y mencionemos nuevamente las variables relacionadas a la gestión del lago.

En primer lugar, tenemos la variable T que representa el período de planificación, mientras que el paso de tiempo es semanal y se lo denota como t, con $0 < t < T$.

Llamamos s_t al volumen del lago en m^3, con $0 \leq t \leq T$, donde el volumen inicial s_0 es un dato del problema. Ésta es la variable que representa nuestro estado. Los límites de esta variable son $0 \leq s_t \leq VM$, siendo $VM = 8200e^6$[m^3] (volumen máximo del lago como se puede apreciar en la Tabla 3.2).

Por otro lado tenemos el turbinado semanal a_t [m^3/s]. Ésta es nuestra única variable efectiva de control y representa el turbinado medio en la semana t. Veremos que la variable correspondiente al vertido puede reducirse y aquellas correspondientes a la generación térmica quedarán intrínsecamente integradas a la función de costo. Luego tenemos los aportes medios semanales ap_t $[m^3/s]$, que son parámetros del problema y representan la cantidad de agua que llega al embalse en la semana t (precipitaciones, etc.). Cabe remarcar que el volumen se registra en instantes puntuales, mientras que tanto el turbinado como los aportes son valores promedios en los intervalos de tiempo.

A aportes medios semanales ap_t y nivel de turbinado medio a_t dados en la semana t, el volumen del lado sigue la dinámica $s_t = s_{t-1} + (ap_t - a_t - j_t) \cdot scXwk$, siendo $scXwk$=604800 la cantidad de segundos por semana.

La variable j_t corresponde al vertido: flujo liberado desde el embalse aguas abajo sin ser turbinado, lo que sucede en general por motivos de seguridad, o para elevar la disponibilidad de agua hacia una posterior en una disposición en serie (.e.g., como sucede en el Río Negro). El vertido aparece como una necesidad estructural, económicamente negativa, así que simplemente lo asimilamos a una variable de holgura que transforma la ecuación de balance de masa en $s_t \leq s_{t-1} + (ap_t - a_t) \cdot scXwk$.

Notar que todas estas variables aparecen con un subíndice h en la Eq.1.1, que especifica la unidad hidráulica. Como en nuestro caso se cuenta con un solo generador hidráulico, podemos obviar este subíndice. Además, estando en el caso determinístico, y con demanda fija y conocida, desaparece el λ que representaba realizaciones de demanda.

Costo de Producción

Centrémonos ahora en lo relacionado al costo de producción del problema, capturada en Eq.1.1 en la forma de la función $g_k(ET_{kt}^{\lambda})$. En el caso determinístico, y con demanda fija y conocida, desaparece el λ, y en particular no aplica el valor esperado que aparece en la función objetivo. Además, recordemos que a producción hidráulica dada, el óptimo de producción térmica es trivial y consiste en utilizar la unidades térmicas en orden creciente de costo y dentro de su límite técnico, hasta cubrir la demanda.

Sea w_t la potencia hidráulica generada en un instante t cualquiera. La demanda horaria es fija con valor $D = 350MWh$. Al contar con dos unidades térmicas (i.e., T_1 y T_2) de distinta eficiencia para cubrir la demanda residual, se recurre en la medida de lo posible a la más barata (T_2). Por tanto, y tomando los parámetros de las unidades térmicas de la Tabla 3.1, si $w_t \geq 100$MW la diferencia se atiende con T_2 a un costo horario de 100USD$\cdot(D - w_t)$/MW de sostenerse constante w_t en esa hora. Si $w_t \leq 100$MW, el control óptimo es usar T_2 a máxima potencia (i.e., 250MW) y cubrir el resto con T_1, de donde el costo por hora sería 100USD·250 + 4000USD$\cdot(100 - w_t)$/MW. Reescribimos las expresiones anteriores como: $c_1(w) = p_1 - q_1 \cdot w$ y $c_2(w) = p_2 - q_2 \cdot w$, siendo: $p_1 = 35000$USD, $q_1 = 100$USD/MW, $p_2 = 425000$USD y $q_2 = 4000$USD/MW.

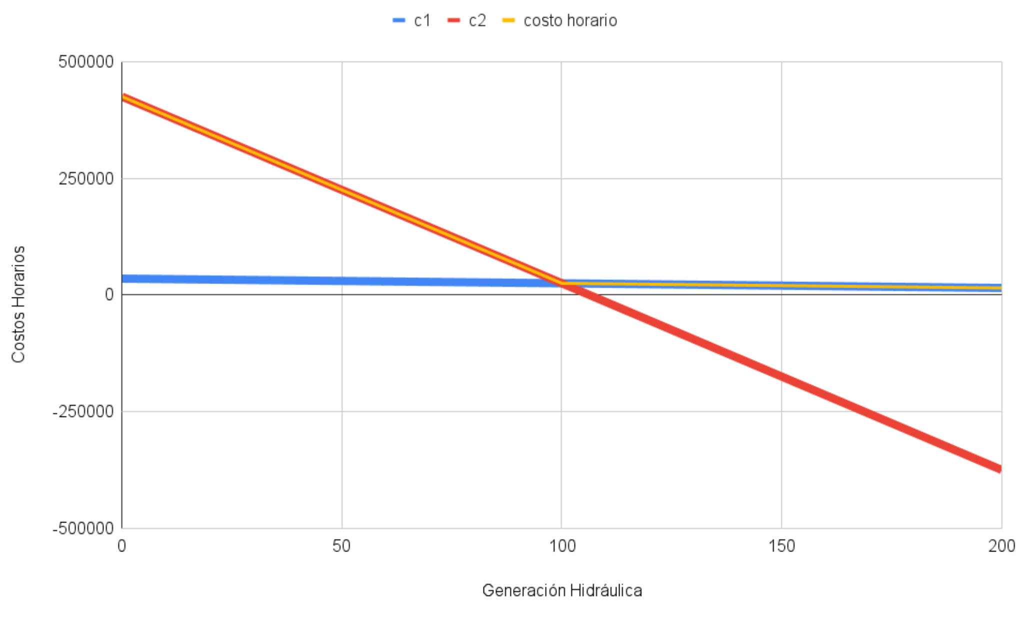


Figura 3.1: Costo horario con respecto a la generación hidráulica.

La Figura 3.1 muestra una gráfica de las funciones c_1 y c_2 junto con el costo horario de generación. Observar que el costo horario de generación es $c_t = \max\{c_1(w_t), c_2(w_t)\}$, donde se ha asumido que w_t es el valor sostenido de la generación hidráulica. Por ser afines, las funciones c_1 y c_2 son convexas, y al ser el costo térmico el máximo de ambas, él también ha de ser convexo.

Al ser convexa la función *costo de producción térmica vs generación hidráulica* y lineal la eficiencia de esta última, el problema de despacho hidrotérmico determinístico es equivalente a uno de Programación Lineal (LP). Incluso el estocástico lo es si se acepta el error de capturar la distribución mediante un conjunto de muestras de aportes, que puede ser enorme dada la eficiencia de los *solvers*.

LP para Despacho Hidrotérmico

Usando que el coeficiente energético CE de la represa es fijo (i.e., no depende de la altura del lago) e igual a 0.19MW/$\frac{m^3}{s}$ (según la Tabla 3.2), se llega a que $w_t = CE \cdot a_t$, y el problema queda expresado en términos de a_t y s_t íntegramente (recordar que a_t es el turbinado y s_t es el volumen de agua en el lago). Recordar además que s_t es nuestra variable de estado, a_t es nuestra variable de control, y se agrega c_t como variable auxiliar. La variable auxiliar c_t es para capturar el máximo de $c_1(t)$ y $c_2(t)$, que como vimos, corresponde al costo.

$$(DHT) \begin{cases} \min\limits_{a_t, s_t, c_t} hsXwk \cdot \sum\limits_{t=1}^{T} c_t \\ s_t \leq s_{t-1} + (ap_t - a_t) \cdot scXwk, & 1 \leq t \leq T, \\ c_t \geq p_1 - CE \cdot q_1 \cdot a_t, & 1 \leq t \leq T, \\ c_t \geq p_2 - CE \cdot q_2 \cdot a_t, & 1 \leq t \leq T, \\ 0 \leq a_t \leq TM,\ 0 \leq s_t \leq VM. \end{cases} \tag{3.1}$$

Además de los parámetros ya mencionados, el problema de la Eq.(3.1) requiere los valores ap_t y s_0 que, como mencionamos anteriormente, son conocidos. El resultado del modelo determinístico para $s_0 = VM/2$ es 1862.37 MUSD como adelantamos al comienzo, y constituye una cota inferior al problema. Es de interés analizar algunos detalles de la solución, para lo que presentamos la Figura 3.2.

Como principal característica del nivel del lago de la represa, se destaca que la tendencia general es a su uso, ya que no tiene costo directo, siendo el lago vacío el estado final. Esto se explica por el hecho que no se ha fijado valor residual para el agua que quede. Hay dos períodos en los que el nivel del lago aumenta, que coinciden con los picos previstos de aportes hidrológicos. El nivel de turbinado es regular con tendencia decreciente en la primera mitad, para alcanzar un mínimo en la semana 60, que coincide aproximadamente con el segundo período de aportes escasos (el primero inicia en $t = 0$). Posteriormente, el erogado hidráulico crece sostenidamente haciendo uso de los aportes más abundantes y buscando vaciar el embalse al final.

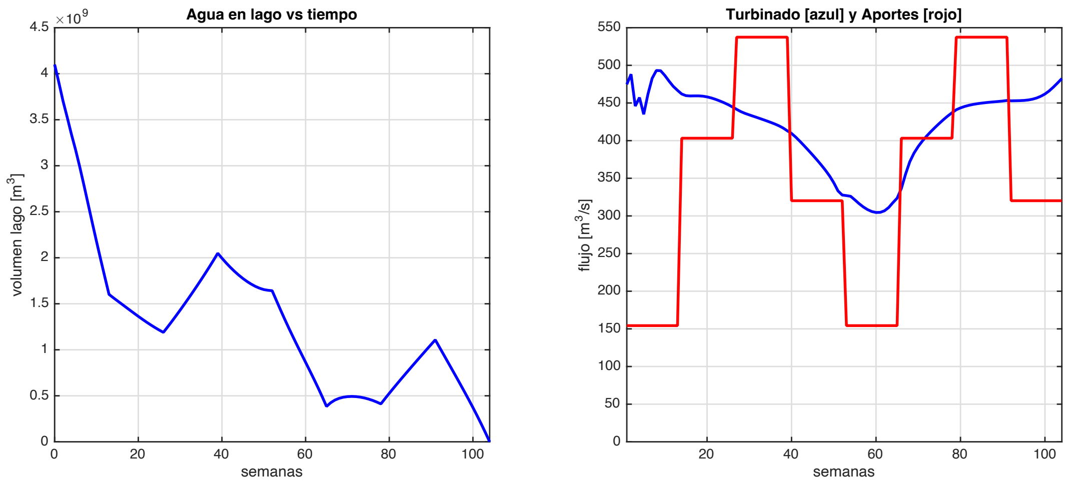


Figura 3.2: [IZQ] Evolución del nivel del lago en el período y [DER] relación entre turbinado (azul) y aportes (rojo).

3.5. Solución por Programación Dinámica

Una solución basada en la Eq.(3.1) no escala bien cuando queremos integrar la incertidumbre de los aportes (y eventualmente una demanda estocástica), por lo que para atacar este problema, vamos a resolverlo mediante Programación Dinámica Estocástica. Antes de pasar a esto, comenzamos por resolver el mismo problema determinístico mediante Programación Dinámica; esto es útil para ajustar la discretización de los estados y el control, y mostrar un error pequeño con respecto a la solución mediante LP.

3.5.1. Versión Determinista - Definición de discretización

Proponemos a continuación una implementación alternativa del problema basada en Programación Dinámica. La implementación más frecuente para este tipo de problema se describe por las transiciones entre estados, sobre un universo finito de ellos. Para mantener la línea del modelo anterior, se ha elegido una implementación que combina estados y control, donde el control corresponde al turbinado hidráulico, como en la Eq.(3.1).

El problema es muy sensible a la discretización debido a la notoria diferencia entre el rendimiento de ambas unidades térmicas, que es de 40 veces.

En la Figura 3.3 se presenta a la izquierda una aproximación usando 13 puntos. Son tres los puntos que caracterizan la función de costo: i) el inicial ($a = 0$), el final ($a = 680m^3/s$, que es TM; turbinado máximo según Tabla 3.2) y el punto de infle-

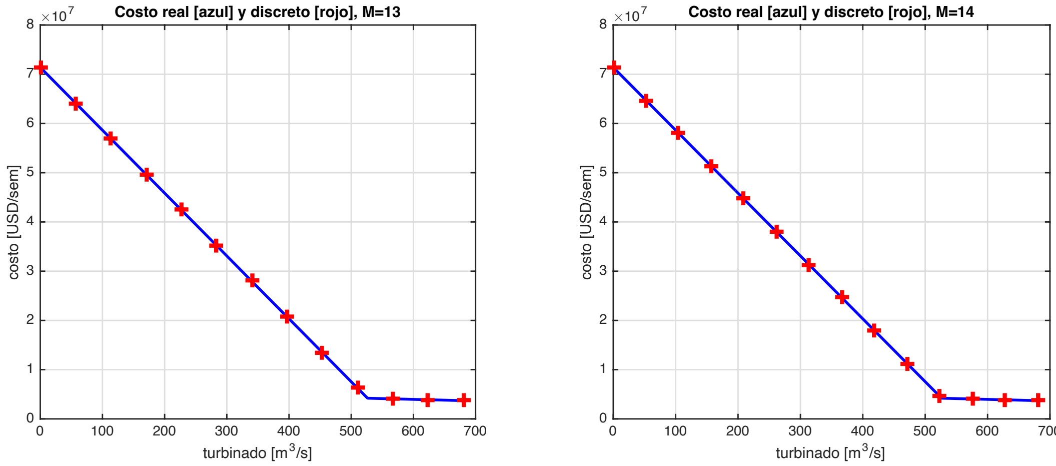


Figura 3.3: Aproximación discreta a la función de costo [IZQ con 13 puntos, DER con 14 puntos].

xión, en el que la demanda residual puede abastecerse sin recurrir a T_1 (la térmica cara). Aunque parece menor, el error en la discretización de la izquierda es muy alto cuando se acumula a lo largo de 104 semanas. El ajuste con 14 puntos (derecha de Figura 3.3) es mucho mejor y es mejor además que el de 15, mostrando la sensible dependencia entre la precisión y los valores puntuales usados para discretizar. Fijamos entonces en $M = 14$ el número de puntos a discretizar el control, de donde los pasos entre un control y el otro son $Da = TM/(M-1) = 52.3077m^3/s$. El control varía entre $a = 1$ (turbinado 0) y $a = 14$ (turbinado máximo), siendo $Da(a-1)$ el valor del turbinado.

Para la discretización de estados en el embalse, se asumen N estados uniformemente espaciados en el embalse, donde $s = 1$ es el lago vacío y $s = N$ corresponde al lago lleno (i.e., VM). El volumen del lago para s dado es $Ds(s-1)$, donde $Ds = VM/(N-1)$. Buscamos N para minimizar $|(Da \cdot SegXsem)/Ds - 1|$, en el entendido que un valor menor nos expone a redondear hacia un estado distante, mientras que uno mayor incrementa los cálculos sin conseguir una diferencia notoria en los resultados. El mejor valor sería $N = 260$, pero se eligió 261 al ser el impar que mejor aproxima el error. Se busca un valor impar para que el error en la condición inicial $VM/2$ sea nulo.

El Algoritmo 7 muestra el pseudocódigo para esta solución. En la línea Algoritmo 7.10 se puede apreciar claramente la recursión de Bellman como fue descrita en la Sección 2.2.5 en la que se introdujo la técnica de Programación Dinámica. El resultado obtenido para esta solución es `A((N-1)/2,1)`$\approx$ **1875.64** MUSD,

Algoritmo 7 Programación Dinámica - Versión Determinista

Input: cantidad de estados del lago N
Input: largo del período a optimizar en años T
Input: vector de aportes AP
Parámetros: turbinado a_t, matriz de valores de Bellman $A_{N\text{x}T+1}$, matriz que representa la política $\pi_{N\text{x}T}$

Inicializar la matriz $A = 0$
procedure PDDTERMINISTA(T, N, AP, A, π)
    for $t \leftarrow 52 * T \ldots 1$ **do**
        for $s \leftarrow 1 \ldots N$ **do**
            $A(s,t) = \text{mín}_{a\in(1.M)}\{cgen(a) + A(s', t+1)\}$ ▷ $s' = s - a + AP(t)$
            $\pi(s,t) = \text{argmin}_{a\in(1..M)}\{cgen(a) + A(s', t+1)\}$
        end for
    end for
end procedure

que está a 0.71 % del valor óptimo para el modelo continuo, lo que se considera adecuado.

3.5.2. Versión Estocástica - Impacto de la aleatoriedad

Usando los parámetros de la discretización anterior, resolvemos ahora el problema de planificación bajo incertidumbre usando la variante de SDP para las distribuciones de aportes conocidas. El pseudocódigo de esta versión es el de Algoritmo 8.

Nuevamente se puede apreciar la recursión de Bellman en la línea Algoritmo 8.10, solo que en este caso es la versión estocástica en donde $\Pr_t[s' \mid s, a]$ representa la probabilidad de que el siguiente estado sea s', dado que el estado actual es s y se toma la acción a. Recordar que $s' = s - a + AP(t)$ y por ende esta probabilidad de transición depende de la distribución de AP.

El resultado del problema anterior para esta versión es `A((N-1)/2,1)`$\approx$ **1937.90** MUSD y corresponde ahora al valor esperado futuro del costo de generación. La diferencia de unos 62.26 MUSD (un 3.34 %) es el costo del azar, que fuerza una planificación más conservadora ante la incertidumbre futura.

La Figura 3.4 muestra el volumen final del lago y la distribución de costos para esta versión del problema como resultado de simular la política de control óptima sobre el universo de 1000 realizaciones a dos años del conjunto de aportes de referencia detallado en la Sección-3.2.

Cabe destacar que en esta versión del problema, los aportes de la semana a

Algoritmo 8 Programación Dinámica - Versión Estocástica

Input: cantidad de estados del lago N
Input: largo del período a optimizar en años T
Input: distribución de aportes AP
Parámetros: turbinado a_t, matriz de valores de Bellman $A_{N\mathbf{x}T+1}$, matriz que representa la política $\pi_{N\mathbf{x}T}$

Inicializar la matriz $A = 0$
procedure PDESTOCÁSTICA(T, N, AP, A, π)
    for $t \leftarrow 52 * T \dots 1$ **do**
        for $s \leftarrow 1 \dots N$ **do**
            $A(s,t) = \text{mín}_{a \in (1..M)}\{cgen(a) + \sum_{s'} \Pr_t[s' \mid s,a] A(s', t+1)\}$
            $\pi(s,t) = \text{argmin}_{a \in (1..M)}\{cgen(a) + \sum_{s'} \Pr_t[s' \mid s,a] A(s', t+1)\}$
        end for
    end for
end procedure

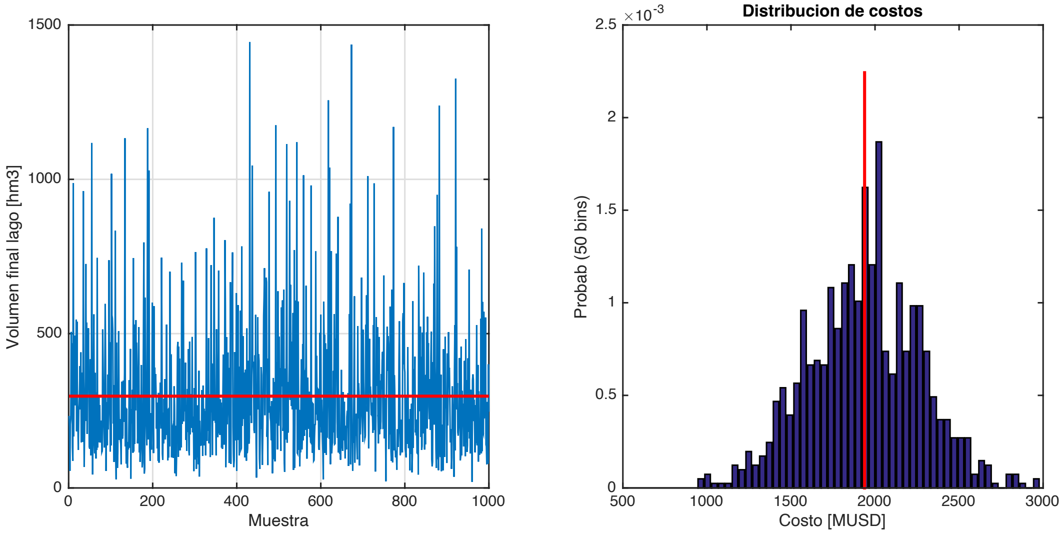


Figura 3.4: Nivel final del lago en cada muestra más promedio [izquierda] y distribución de costos efectivos y valor esperado [derecha].

planificar se asumen desconocidos, es decir que no se usa esta información a la hora de decidir el turbinado de esa semana. En la siguiente sección hacemos una versión en la que se cuenta con esta información, lo cual es útil para comparar con la solución de RL que también hace uso de esta información.

3.5.3. Versión Estocástica con datos de Aportes A-Priori y Simulación

Aunque aleatorios en el mediano plazo, en el problema real de despacho, el aporte de la semana próxima se conoce con alta precisión (datos de lluvia en la cuenca de los embalses, que alcanzan varios países).

En este punto se hace el ejercicio de simular el resultado promedio de simular el despacho usando los valores esperados futuros antes encontrados, pero usando en el control el dato del aporte en la semana a planificar. El pseudocódigo para esta simulación se puede ver en el Algoritmo 9.

Al tener el aporte de la semana a planificar, hay que volver a explorar los posibles turbinados en la semana actual, para determinar el adecuado. No se puede usar directamente lo que diga la matriz de Bellman hallada en la sección anterior, ya que esa matriz fue creada sin esa información y por ende puede ser incorrecta. Sí usamos el costo futuro de la semana siguiente, como se puede apreciar en la línea Algoritmo 9.17, ya que de la semana siguiente en adelante se sostiene la incertidumbre de los aportes.

El resultado al aplicar el Algoritmo 9 es un costo promedio de $\approx$ **1882.86** MUSD, que está a tan solo un 1.1 % de la cota inferior encontrada en la Sección-3.4.2. La Figura 3.5 muestra nuevamente el volumen final y los costos en el conjunto de aportes de referencia de 2000 años (i.e., 2añosx1000 simulaciones).

La calidad de esta combinación, SDP + Aportes A-Priori, refleja muy bien la operativa y es extraordinariamente buena en la calidad esperada como se puede apreciar por su cercanía con la cota inferior (no debería haber otra solución que la supere en más de un 1 %).

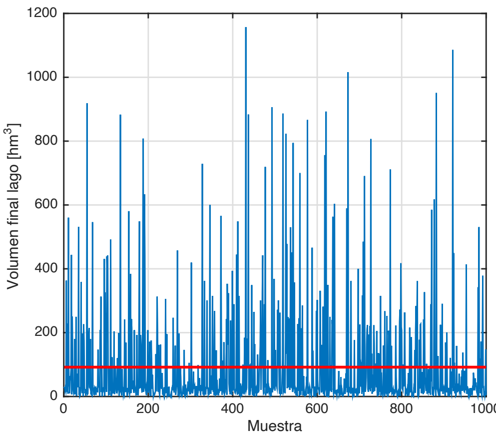
3.6. Solución por *Reinforcement Learning*

Como mencionamos en el Capítulo 2, el algoritmo de RL utilizado es el *One-Step Actor-Critic* mostrado en el Algoritmo 3. A continuación listamos qué representa cada elemento:

Algoritmo 9 Programación Dinámica Estocástica - Simulación

Input: conjunto de escenarios de aportes de referencia Ref
Input: cantidad de niveles del turbinado M
Input: largo del período a optimizar en años T
Input: distribución de aportes AP
Parámetros: matriz de valores de Bellman $A_{N\mathbf{x}T+1}$

procedure SIMULACIONPDE(T, M, Ref, A)
  for all $AP \in \mathcal{R}ef$ **do**
    $V = VM/2$
    $costo = 0$
    for $t \leftarrow 1 \ldots 52 * T$ **do**
      $fBllmn = zeros(M, 1)$
      for $a \leftarrow 1 \ldots M$ **do**
        if $AP(t) * SegXSem + V \geq Da * (a-1) * SegXSem$ **then**
          $V_{next} = V + (AP(t) - Da * (a-1)) * SegXSem$
          $s' \leftarrow V_{next}$ ▷ Estado siguiente según el volumen
          $fBllmn(a) = cgen(a) + A(s', t+1)$
        else
          $fBllmn(a) = inf$
        end if
      end for
      $control = \text{argmin}(fBllmn)$
      $costo = costo + cgen(control)$
      $V = V + (AP(t) - Da * (control - 1)) * SegXsem$
    end for
  end for
  $costo = costo/size(Ref)$
end procedure



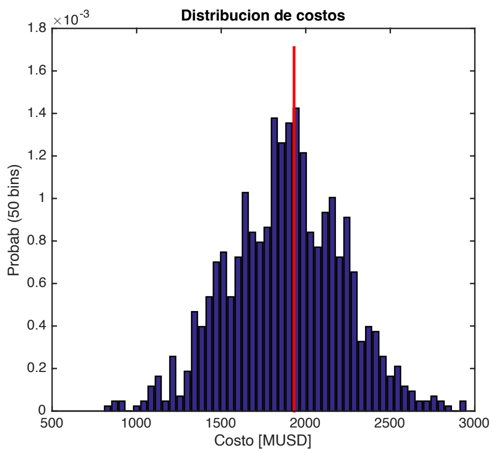


Figura 3.5: Nivel final del lago en cada muestra del conjunto de referencia más promedio [izquierda] y distribución de costos efectivos y valor esperado [derecha].

- S, S' - Representan el estado en dos instantes de tiempo consecutivos. En esta iteración del problema, el estado representa el nivel del lago en la represa junto con el instante de tiempo.
- A - Representa la acción que se toma. En esta iteración del problema, la acción es el turbinado realizado por la unidad hidráulica.
- R - Representa la recompensa observada. En esta iteración del problema, la recompensa es el opuesto del costo de operación que se incurre en ese instante, luego de cubrir la demanda. Se utiliza el opuesto, ya que el algoritmo busca la política que maximice las recompensas obtenidas y por ende minimice los costos.
- $\mathbf{w}$ - Representa el vector de parámetros de la función de valor. En esta iteración del problema, un parámetro para cada centroide RBF como fue mencionado en el Capítulo 2.
- $\hat{v}$ - Representa la función de valor. En esta iteración del problema, se utilizó una función lineal como la descrita en el Capítulo 2:

  $$\hat{v}(s, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}(s),$$

  siendo $x(s)$ el vector de *features* RBF.
- θ - Representa el vector de parámetros de la política. En esta iteración del problema, se tiene, nuevamente, un parámetro para cada centroide RBF.
- π - Representa la política. En nuestro caso utilizamos una política *Gaussiana*:

  $$\pi(a \mid s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-\left(a - \mu(s;\theta)\right)^2}{2\sigma^2}\right).$$

  Como se puede apreciar, lo que está parametrizado es la media μ.
- α^θ - Representa el *learning rate* para la política.
- $\alpha^\mathbf{w}$ - Representa el *learning rate* para la función de valor.

El algoritmo Actor-Critic aprende tanto la función de valor como la política a la vez, es decir, que ajusta los vectores de parámetros $\mathbf{w}$ y θ respectivamente.

El agente se entrenará por un número grande de episodios. Como mencionamos en la descripción del problema, cada episodio tiene una duración de 2 años y paso de tiempo semanal.

Se utiliza la técnica de *Exploring Starts*, en donde se elige aleatoriamente el estado inicial de cada episodio; es decir, se elige aleatoriamente la semana en la que empieza el episodio y el volumen del lago. Esta técnica ayuda en la exploración del espacio de estados lo cual es muy importante en los sistemas de RL como se mencionó en el Capítulo 2.

Con los mismos fines y para ayudar en la exploración del espacio de acciones, se utilizó la técnica $\epsilon - greedy$, en donde con cierta probabilidad se toma una acción al azar en vez de la que dicta la política.
Quedan como hiper-parámetros del algoritmo los siguientes:

- e - cantidad de episodios de entrenamiento,
- α^{θ} - *learning rate* para la política,
- $\alpha^{\mathbf{w}}$ - *learning rate* para la función de valor,
- γ - factor de descuento,
- σ - desvío estándar de la política *Gaussiana*,
- ϵ - porcentaje de épsilon *greedy*,
- σ_{RBF} - desvío estándar en RBF,
- C_{RBF} - cantidad centroides RBF.

Los aportes a la represa se obtienen del generador descrito en la Sección 3.1.2. Para el entrenamiento se genera una muestra reducida de aportes con este generador para que el agente aprenda la política óptima. A su vez, se usa el conjunto de aportes de referencia mencionado en la Sección 3.2 para evaluar la política obtenida y compararla con políticas que resulten de otros algoritmos, como el de Programación Dinámica, mediante el Algoritmo 6.

Se utiliza el paradigma azar-decisión, en donde se sortea la aleatoriedad antes de tomar la decisión en cada paso, es decir, se conoce el aporte de la semana a priori (al igual que en la Sección 3.5.3). Para la iteración 1, esto se traduce en que se sortea el aporte, y se agrega al volumen del lago, antes de turbinar en cada paso.

El procedimiento del experimento es entonces el siguiente:

a) Se genera un subconjunto de datos de aportes de entrenamiento (se probaron varios valores, con que la muestra sea mayor a 42 años de aportes, se logra que sea representativa del generador; como se determinó en la Sección 3.1.2).
b) Se sortea un estado inicial del sistema, tanto el volumen del lago como la semana dentro de las $T = 104$ posibles (*Exploring Starts*).

c) Se avanza siguiendo el algoritmo Actor Critic hasta llegar al paso $T = 104$. Esto concluye un episodio.

d) Se repiten los pasos b) y c) la cantidad de episodios que se quiere entrenar.

e) Se evalúa la política aprendida por el agente contra los 2000 años del conjunto de referencia (1000 corridas de 2 años). Para evaluar la política, se parte desde la semana 1 con el lago por la mitad (4100hm3), se opera el sistema siguiendo la política y se obtiene el costo de operación del período. Se repite este procedimiento para las 1000 corridas y se promedia el costo obtenido en esas 1000 corridas. Este paso no es otra cosa que aplicar el Algoritmo 6.

3.7. Experimentos y Resultados

3.7.1. Programación Dinámica

La solución por Programación Dinámica fue realizada en Matlab [1]. Los experimentos y resultados para esta técnica fueron mayormente explicados en la sección anterior. Recordamos que el resultado final es: 1.882.860.000, y el tiempo de ejecución es ~ 6 minutos en un computador personal de escritorio.

3.7.2. *Reinforcement Learning*

La solución por RL fue desarrollada en el lenguaje Python. El entrenamiento del algoritmo de RL fue paralelizado utilizando la herramienta **MPI** que es una de pasaje de mensajes entre procesos y el algoritmo fue corrido en el Centro Nacional de Supercomputación (ClusterUY) [2]. La solución final cuenta con 13 procesos, 12 realizando entrenamiento paralelo (entrenadores), y 1 que se encarga de la mediación entre los 12 entrenadores. Se probó un amplio rango de valores para los hiper-parámetros del algoritmo mencionados en la Sección 3.6. La Tabla 3.6 muestra el rango de valores que se probaron para los hiper-parámetros, junto con los mejores valores encontrados.

Para los mejores valores encontrados de los hiper-parámetros, se obtiene un resultado de 1.883.973.999 en unos ~ 2.3 minutos. Las Figuras 3.6 y 3.7 comparan los resultados obtenidos por Programación Dinámica y *Reinforcement Learning* junto con las cotas superior e inferior. Recordar que los valores que se muestran

[1]Matlab. The MathWorks Inc. https://www.mathworks.com/products/matlab.html. Accedido en 2023-06-09

[2]ClusterUY. Centro Nacional de Supercomputación, https://cluster.uy/. Accedido: 2023-06-09

Hiper-parámetro	Rango/Paso	Mejor Valor
e - cantidad de episodios de entrenamiento	1 Millón - 10 Millones	-
α^{θ} - *learning rate* para la política	1e-9 - 1e-2 / 1e-9	2.4e-9
$\alpha^{\mathbf{w}}$ - *learning rate* para la función de valor	0.005 - 0.8 / 0.005	0.2
γ - factor de descuento	1	-
σ - desvío estándar de la política *Gaussiana*	1 - 340 / 1	5
ϵ - porcentaje de épsilon *greedy*	Dinámico	-
σ_{RBF} - desvío estándar en RBF	340 - 4100 / 10	2520
C_{RBF} - cantidad centroides RBF	2 - 101 / 1	3

Tabla 3.6: Rango de valores usados y mejor valor para los hiperparámetros para el entrenamiento del agente de RL

son el resultado obtenido de promediar los resultados en nuestro conjunto de referencia de 2000 años de aportes. Es necesario comparar las metodologías de esta manera (y no comparar las políticas de operación obtenidas), ya que puede haber potencialmente infinitas políticas óptimas de operación.

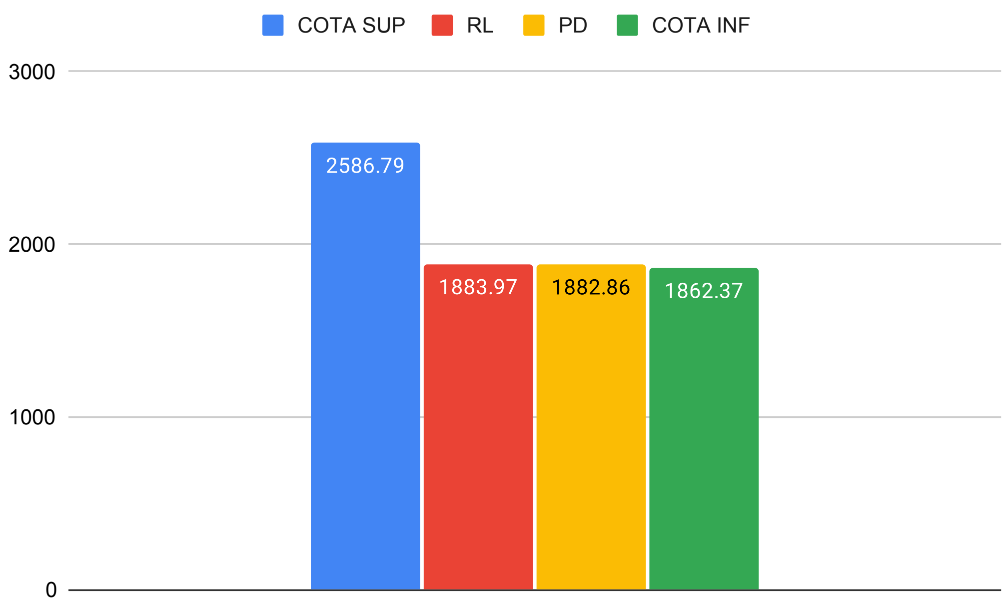


Figura 3.6: Comparación entre resultados de las soluciones por PD y RL junto con las cotas.

La Figura 3.8 muestra la evolución del resultado a medida que avanza la cantidad de episodios de entrenamiento del algoritmo de RL.

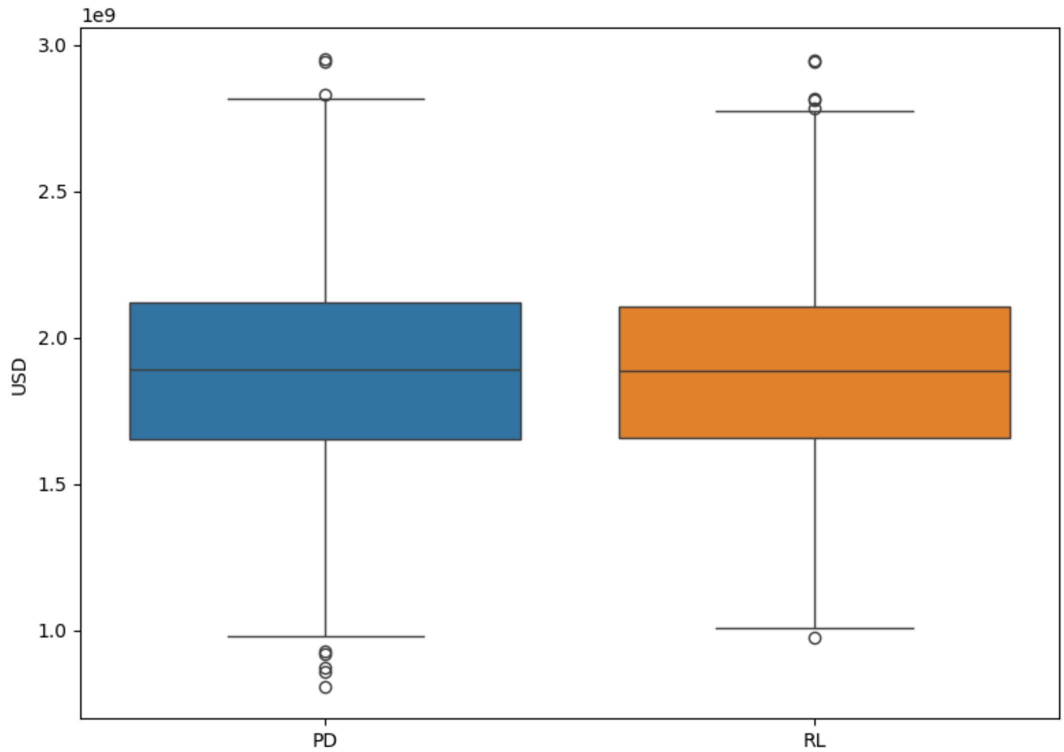


Figura 3.7: Comparación boxplot de los resultados obtenidos mediante PD y RL.

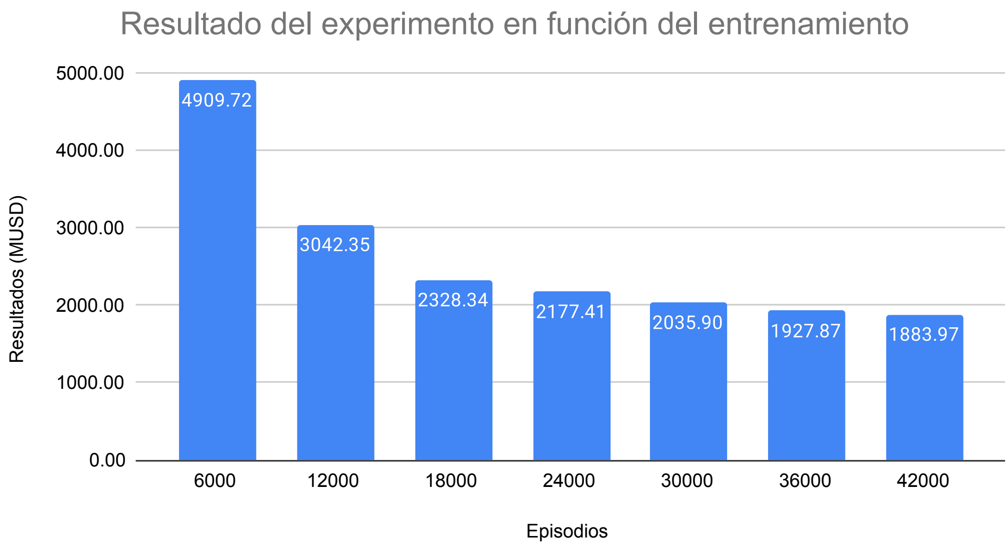


Figura 3.8: Resultado del experimento de RL obtenido durante el entrenamiento. Notar que el rango 6000-42000 es para los 12 procesos juntos, el rango para cada proceso es 500-3500 episodios.

3.7.3. Análisis de resultados

El primer resultado a destacar es la diferencia entre las soluciones por PD y RL, y la cota superior (técnica *greedy*), que es del entorno de los ~700 MUSD. Esto deja en evidencia lo mencionado anteriormente de cómo el uso inteligente del agua hace una diferencia tremendamente significativa (a diferencia de la técnica *greedy* que simplemente usa toda el agua en cada paso, siendo éste el óptimo local).

En segundo lugar, tanto el tamaño de la muestra de aportes como la discretización del lago para la técnica PD, resultan muy satisfactorios lo cual es evidenciado por la diferencia de los resultados con respecto a la cota inferior de las distintas versiones: $0.71\,\%$ (versión determinística) y $1.1\,\%$ (versión estocástica).

Por otro lado, la técnica de RL también resulta muy satisfactoria, estando el resultado obtenido a tan solo un $\sim\ 0.06\,\%$ del resultado obtenido mediante PD. Recordar que uno de los objetivos de este trabajo es evaluar el desempeño de estas técnicas para el problema del despacho hidrotérmico, comparándolas con técnicas de optimización tradicional. También resulta efectivo desde el punto de eficiencia computacional, lo cual no es muy relevante en esta Iteración, ya que las técnicas tradicionales siguen siendo viables en este caso, pero que es muy relevante cuando no lo son, como es el caso de la Iteración 2 presentada en el siguiente capítulo.

Analizando los resultados obtenidos mediante PD y RL a nivel de los 1000 escenarios de prueba, podemos ver que si bien el resultado promedio obtenido por PD es un $\sim\ 0.06\,\%$ mejor, en la gran mayoría de los escenarios, RL obtiene un mejor resultado. Los detalles de esto se pueden ver en la Tabla 3.7. También en la Tabla 3.7 y en la Figura 3.9 y Figura 3.10 se puede ver que RL tiene un peor desempeño en los escenarios más húmedos, mientras que en los escenarios más secos es constantemente superior a PD. Un estudio exhaustivo de éste fenómeno escapa del alcance del trabajo, pero la intuición es que se debe a la falta de representación de la función de valor lineal utilizada. Esto es interesante, ya que los estudios del estilo “estudiar los X escenarios más secos/húmedos” son comunes en la industria. Un análisis de éste fenómeno sobre los 1000 escenarios de prueba en tramos de a 200 se puede ver en la Figura 3.11. Notar que si bien la Figura 3.11 muestra un mejor desempeño de RL en los 800 escenarios más secos con respecto a PD, el mismo no alcanza para sobreponerse a la diferencia en los 200 escenarios más húmedos. No obstante esto, basta con ignorar los 9 primeros escenarios más húmedos para que el RL sea superior en los 991 escenarios restantes.

Este fenómeno se verá también presente en la Iteración 2 presentada en el siguiente capítulo.

	Mejor RL	Mejor PD
1000 Escenarios	**638**	362
50 Escenarios más húmedos	2	**48**
50 Escenarios más secos	**49**	1

Tabla 3.7: Cantidad de escenarios en los que cada técnica es mejor.

Figura 3.9: Comparación de los resultados obtenidos mediante PD y RL en los 50 escenarios más húmedos.

Figura 3.10: Comparación de los resultados obtenidos mediante PD y RL en los 50 escenarios más secos.

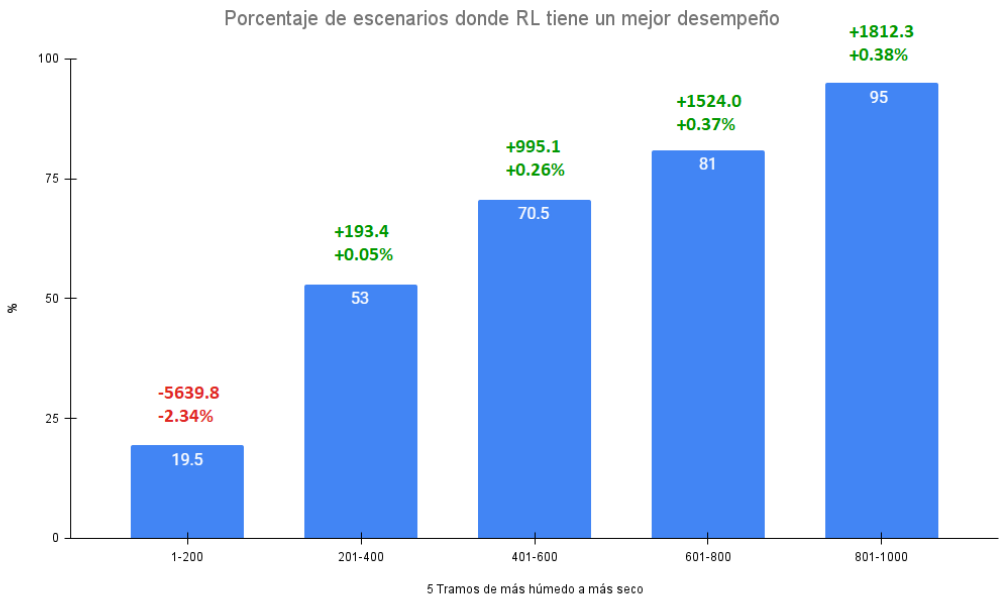


Figura 3.11: En la barra se muestra el porcentaje de escenarios del tramo en donde RL obtiene un mejor resultado que PD. Los números sobre las barras muestran la mejora/desmejora de RL con respecto a PD en el tramo, en error medio y en costo total (en MUSD).

Capítulo 4

Iteración 2 - Múltiples Generadores Hidráulicos

En este capítulo resolvemos un problema de despacho similar al resuelto en el Capítulo 3, pero con la principal diferencia que en este caso se cuenta con tres generadores hidráulicos (en lugar de uno solo), cada uno con su lago y embalse. Este caso es de especial interés, ya que como se mencionó en la Sección 2.2.8, al crecer la cantidad de lagos, crece también el espacio de estados del problema, se cae en la maldición de la dimensionalidad, y la planificación se vuelve cada vez más difícil de resolver computacionalmente mediante métodos tradicionales de optimización.

El capítulo comienza con una descripción específica de nuestra instancia del problema en la Sección 4.1 y en particular se detallan las características de los tres generadores hidráulicos. Al igual que para la Iteración 1, comenzamos por hallar cotas superiores e inferiores en la Sección 4.2, muy útiles a la hora de evaluar los resultados. Luego, en la Sección 4.3, obtenemos una aproximación a la solución mediante Programación Dinámica para tener como referencia. Luego resolvemos el problema presentado en esta iteración mediante RL en la Sección 4.4, y por último, en la Sección 4.5 se reportan y analizan los resultados obtenidos. En general, en cada sección se repite la metodología aplicada en la primer iteración.

4.1. Descripción del Problema

El problema de referencia ahora cuenta con 5 unidades generadoras, tres represas con embalse, y dos unidades térmicas. Las dos unidades térmicas, T1 y T2, tienen las mismas características que las usadas en el problema de la Iteración 1.

En cuanto a las unidades hidráulicas, H1 coincide con la utilizada en el problema de la Iteración 1, mientras que H2 y H3 coinciden en todas sus características con H1, con la excepción de que sus lagos tienen la mitad de capacidad. Las características se pueden ver detalladas a continuación en la Tabla 4.1.

Unidad	Volumen Máximo	Turbinado Máximo	Coeficiente Energético
`H1`	8200 hm^3	680 m^3/s	0.19 MW/$\frac{m^3}{s}$
`H2`	4100 hm^3	680 m^3/s	0.19 MW/$\frac{m^3}{s}$
`H3`	4100 hm^3	680 m^3/s	0.19 MW/$\frac{m^3}{s}$

Tabla 4.1: Unidades Hidráulicas Iteración 2

En lo que respecta a la línea de tiempo, se mantiene igual a la Iteración 1, horizonte de tiempo $T = 2$ años y paso semanal.

En lo que a la demanda refiere, se sigue asumiendo, sin pérdida de generalidad, que ella es constante en el tiempo. Esa premisa busca simplificar el cálculo de la función de costo sin afectar el objetivo del trabajo. Sin embargo, dado que el parque hidráulico es mayor en la realidad aquí simulada, se ha aumentado el valor de referencia de la demanda a 500 MW, para que no sea posible atenderla enteramente con el parque hidráulico. El volumen inicial en cada lago también se toma a la mitad del máximo técnico, a saber: `H1` = 4100 hm^3 y `H2` = `H3` = 2050 hm^3.

En cuanto a los aportes a los lagos, se asume que los aportes son los mismos para las tres unidades hidráulicas, y se utiliza el mismo modelo de aportes presentado en la Sección 3.1.1.

4.2. Cotas del Problema

Al igual que en la Iteración 1, la metodología utilizada comienza por determinar cotas a las solución del problema, las cuales son de vital importancia para tener como referencia a la hora de evaluar la solución obtenida mediante RL.

4.2.1. Cota Superior usando técnica *Greedy*

Como primer valor de referencia para una cota superior, y análogo a lo hecho en la Sección 3.4.1, recurrimos nuevamente a la técnica *Greedy*. Recordar que esta técnica es aquella que utiliza las unidades hidráulicas al máximo posible en cada instante, antes de recurrir a las unidades térmicas para cubrir la demanda restante. La política sería entonces:

$$\pi = (maxTur_{\mathtt{H1}}, maxTur_{\mathtt{H2}}, maxTur_{\mathtt{H3}}).$$

Al igual que en la Iteración 1, las unidades térmicas son capaces de cubrir la demanda en su totalidad (ver Tabla 3.1), por lo que la política *Greedy* es una solución factible, y por ende, una cota superior del óptimo del problema.

El valor de referencia obtenido para esta técnica es: **COTA SUPERIOR: 5226.6** MUSD.

Como se verá más adelante, esta cota resulta insatisfactoria debido a su alta holgura con respecto a la cota inferior. Es por esta razón que en la Sección 4.3 se desarrolla una heurística basada en PD para hilar un poco más fino, y obtener un mejor valor de referencia superior.

4.2.2. Cota Inferior usando Programación Lineal (LP)

Al igual que en la Iteración 1 (Sección 3), nuestro problema de referencia es estocástico en los aportes hidrológicos. Siguiendo un procedimiento análogo al realizado en la Sección 3.4.2, utilizaremos los aportes medios en las distintas estaciones para reducir el problema a uno determinístico, que resolveremos mediante Programación Lineal (LP). Este resultado constituye una cota inferior del problema por los mismos argumentos esgrimidos en la Sección 3.4.2.

La referencia LP se usó en la Iteración 1 para ajustar la discretización de un algortimo determinístico de Programación Dinámica (ver Sección 3.5.1), ya que su generalización al caso estocástico era eficiente en esa instancia, según consta en la Sección 3.5.2. En esta versión del problema es inviable seguir la misma metodología. Recordamos de la Sección 2.2.5 que la expresión para el tiempo de ejecución de un algoritmo de programación dinámica es proporcional a Tn^2, siendo T el número de etapas y n el de estados. En la Iteración 1 habíamos identificado que $n = 261$ estados permitía errores de discretización menores al 1 % para los parámetros de `H1`. Al tener `H2` y `H3` la mitad de capacidad, podríamos usar 131 estados en cada caso. El problema es que la cantidad de estados del sistema surge de la combinación de estados en cada represa. Si el algoritmo de SDP demorara un minuto en resolver el problema de la Iteración 1, hay que pensar que demorará $131^2 \times 131^2$ veces más en esta Iteración, lo que serían más de 550 años. Por escapar al objeto de este estudio, no nos extenderemos en alternativas para paliar ese problema, como interpolaciones en la función de valor para usar menos estados en la discretización o la posibilidad de paralelizar parcialmente los cálculos.

La estrategia a seguir en esta sección se soporta en la extraordinaria eficiencia de la versión LP determinística, que permite resolver miles de problemas en poco tiempo. Recordemos que para evaluar soluciones del problema sobre realizaciones del proceso de aportes, contamos con un conjunto de 2000 años de aportes de referencia (Sección 3.2). La idea en esta sección es evaluar la política en los 1000 escenarios formados de a 2 años del conjunto de referencia, y estimar el valor esperado de producción a través de la media de las 1000 soluciones determinísticas.

El hecho que el resultado de ese proceso sea una cota inferior de lo que sería la solución al problema estocástico completo se soporta principalmente en dos propiedades: i) la función de costos es convexa y decreciente respecto al turbinado, por lo que la disponibilidad sostenida de agua resulta más eficiente que oscilar en torno a una media del mismo valor, y ii) el procedimiento solamente consigue un promedio de costos independientes, no busca ni construye en general soluciones factibles y tiene, por tanto, un espacio de configuraciones más amplio.

Esta cota debería ser siempre una mejor cota inferior (i.e., más elevada) que la que usa solamente los aportes medios por lo que será el valor que usaremos como cota inferior. Adelantamos que el resultado en este caso es: **COTA INFERIOR: 2376.8** MUSD, para luego elaborar los detalles del modelo que permiten llegar a ese valor. Adelantamos también que la versión que usa los valores esperados de los aportes obtiene como resultado: **2041.1** MUSD, confirmando que es una peor cota inferior.

Al igual que en la Iteración 1, presentamos la resolución del problema LP tomando como referencia el problema general de despacho presentado en la Eq. 1.1 e instanciándolo para nuestro caso determinista. Inmediatamente a continuación, elaboramos en lo relacionado con las gestión de los lagos, para agregar posteriormente lo relacionado al modelado de los costos de producción. Se concluye con el modelo completo de programación lineal.

Gestión del Lago

Llamamos $v_t^{(1)}$ al volumen del lago en `H1` expresado en m^3, con $0 \leq t \leq T = 2$ años, donde el volumen inicial $v_0^{(1)}$ es un dato del problema. Los límites de esta variable son $0 \leq v_t^{(1)} \leq VM_1$, siendo $VM_1 = 8200e^6$[m^3]. Al estado en los lagos de `H2` y `H3` le corresponden las variables $v_t^{(2)}$ y $v_t^{(3)}$ respectivamente, que cumplen:

$0 \leq v_t^{(2)}, v_t^{(3)} \leq VM_1/2$. Las variables de control (expresadas en m^3/s) son $x_t^{(1)}$, $x_t^{(2)}$ y $x_t^{(3)}$ respectivamente. El balance de masa en los lagos determina que $v_t^{(i)} \leq v_{t-1}^{(i)} + (a_t^{(i)} - x_t^{(i)}) \cdot scXwk$, con $i = 1, 2, 3$. Recordar que $scXwk$=604800 es la cantidad de segundos por semana. Se asume que los erogados son independientes de los aportes.

Costo de Producción y Modelo Lineal

Sea w_t la potencia hidráulica generada en un instante t cualquiera. Se cumple que $w_t = w_t^{(1)} + w_t^{(2)} + w_t^{(3)} = CE \cdot (x_t^{(1)} + x_t^{(2)} + x_t^{(3)})$, debido a que todas las hidráulicas tiene la misma eficiencia. La demanda horaria es fija con valor $D = 500MWh$.

La capacidad combinada de las térmicas `T1` y `T2` permite alimentar toda la demanda. La diferente eficiencia lleva a recurrir en la medida de lo posible a la barata (`T2`). Por tanto, si $w_t \geq 250$MW la diferencia se atiende con `T2` a un costo horario de 100USD$\cdot(D - w_t)$/MW de sostenerse constante w_t en esa hora. Si $w_t \leq 250$MW, el control óptimo es usar `T2` a máxima potencia (i.e., 250MW) y cubrir el resto con T_1, de donde el costo por hora sería 100USD$\cdot$250 + 4000USD$\cdot(250 - w_t)$/MW. Reescribimos las expresiones anteriores como: $c_1(w) = p_1 - q_1 \cdot w$ y $c_2(w) = p_2 - q_2 \cdot w$, siendo: $p_1 = 50000$USD, $q_1 = 100$USD/MW, $p_2 = 1025000$USD y $q_2 = 4000$USD/MW. El costo horario de generación es $c_t = \max\{c_1(w_t), c_2(w_t)\}$. Usando que el coeficiente energético CE de las represas es fijo e igual a 0.19MW/$\frac{\mathrm{m}^3}{s}$, se llega a que $w_t = CE \cdot (x_t^{(1)} + x_t^{(2)} + x_t^{(3)})$, y el problema queda:

$$(DHT) \begin{cases} \min\limits_{x_t^{(i)}, v_t^{(i)}, c_t} hsXwk \cdot \sum\limits_{t=1}^{T} c_t \\ v_t^{(1)} \leq v_{t-1}^{(1)} + (a_t^{(1)} - x_t^{(1)}) \cdot scXwk, & 1 \leq t \leq T, \\ v_t^{(2)} \leq v_{t-1}^{(2)} + (a_t^{(2)} - x_t^{(2)}) \cdot scXwk, & 1 \leq t \leq T, \\ v_t^{(3)} \leq v_{t-1}^{(3)} + (a_t^{(3)} - x_t^{(3)}) \cdot scXwk, & 1 \leq t \leq T, \\ c_t \geq p_1 - CE \cdot q_1 \cdot (x_t^{(1)} + x_t^{(2)} + x_t^{(3)}), & 1 \leq t \leq T, \\ c_t \geq p_2 - CE \cdot q_2 \cdot (x_t^{(1)} + x_t^{(2)} + x_t^{(3)}), & 1 \leq t \leq T, \\ 0 \leq x_t^{(i)} \leq TM,\ 0 \leq v_t^{(1)} \leq VM, \\ 0 \leq v_t^{(2)} \leq \frac{VM}{2}, 0 \leq v_t^{(3)} \leq \frac{VM}{2}. \end{cases} \tag{4.1}$$

Además de los parámetros ya mencionados, el problema (4.1) requiere los valores $a_t^{(i)}$ y $v_0^{(i)}$.

Usando los valores estacionales esperados para los aportes en todos los lagos, llegamos a una solución interesante, que casi siempre debe usar la térmica cara, salvo al final. Para los niveles iniciales en los lagos tomamos la mitad del valor máximo en cada caso. El resultado del modelo determinístico es 2041.1 MUSD, lo que constituye una cota inferior a la solución con *Reinforcement Learning*, que resuelve una versión estocástica del mismo problema.

La Figura 4.1 esquematiza algunas características de la solución. La gestión en las represas `H2` y `H3` coincide exactamente (i.e. mismo turbinado y volumen) en el período. A la izquierda de la figura se presenta la evolución en lago principal (el de `H1`, en azul) y en rojo los otros dos. A la derecha se muestra la generación hidráulica combinada en celeste, así como el umbral sobre el cual no es necesario usar la térmica cara.

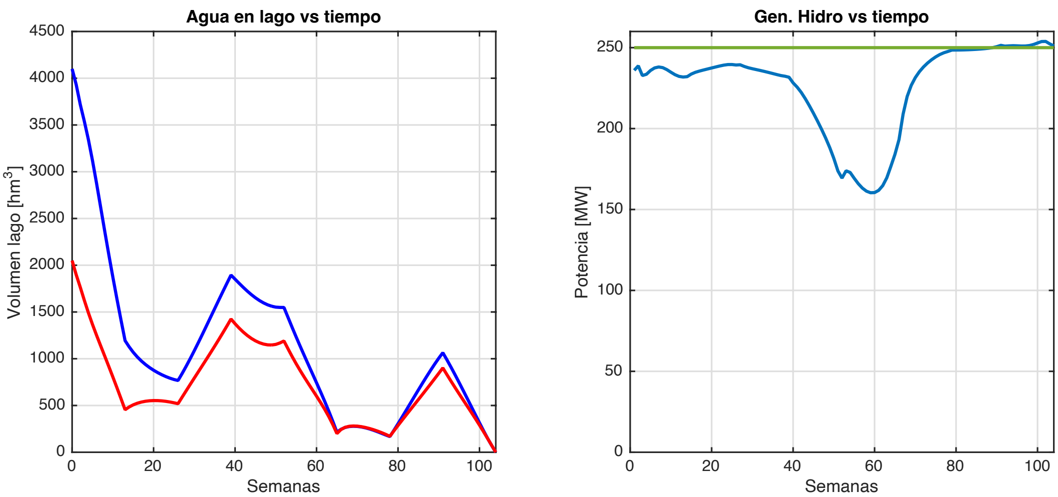


Figura 4.1: [IZQ] Evolución de niveles en lagos en el período y [DER] generación hidráulica total.

Promedio de soluciones determinísticas en el conjunto de referencia

Como se adelantó al comienzo, este mismo LP, que se detalló en la sección anterior para los aportes medios, se puede resolver pero usando los aportes de los 1000 escenarios que se forman con el conjunto de aportes de referencia. Promediando el resultado en los 1000 escenarios se obtiene una cota inferior de 2376.8 MUSD, que es una mejor cota que la obtenida usando los aportes medios (2041.1 MUSD). Es claro que este promedio es una cota inferior, ya que obtener una política óptima para cada uno de los 1000 escenarios y promediar sus resultados, tiene que ser necesariamente menor o igual al resultado obtenido por una única política que intenta atender a los 1000 escenarios lo mejor posible.

4.3. Cota superior Heurística mediante PD (HPD)

Como mencionamos previamente, resolver el problema completo de esta sección mediante PD –con la implementación de referencia– sería computacionalmente intratable. Es precisamente ese hecho el que nos motivó en la Sección 4.2.2 a buscar una cota/referencia inferior mediante un promedio de soluciones deterministas. Como veremos posteriormente, esa referencia se muestra satisfactoria.

En oposición a lo anterior, la cota superior calculada en la Sección 4.2.1 presenta una holgura muy alta con la inferior. Dicha holgura también era alta en la Iteración 1, pero en ella se contaba con la referencia para la solución exacta calculada con SDP en la Sección 3.5. En esta iteración, proponemos entonces como referencia otra heurística más precisa, también basada en SDP.

La idea consiste en resolver 3 subproblemas independientes (computacionalmente tratables) mediante la misma técnica de PD detallada en Algoritmo 8, en donde cada uno cuenta únicamente con una de las unidades hidráulicas. Complementariamente, el modelo de cada subproblema asume la existencia de dos térmicas de igual rendimiento que las reales, pero con un tercio de la potencia máxima correspondiente. Finalmente, el requerimiento de demanda a ser atendido en cada subproblema también es un tercio de la demanda total.

La Tabla 4.2 detalla las características de estos subproblemas. Para resolver estos subproblemas se utiliza la misma metodología que la descrita en la Sección 3.5 (en particular la versión vista en la Subsección 3.5.3). Luego de resolver estos 3 subproblemas, se suman los costos obtenidos para obtener una aproximación del resultado al problema entero mediante PD.

Sub-problema	Unidad Hidráulica	Térmico 1	Térmico 2	Demanda
1	`H1`	84 MW | 4000 USD/MWh	84 MW | 100 USD/MWh	167 MW
2	`H2`	83 MW | 4000 USD/MWh	83 MW | 100 USD/MWh	166 MW
3	`H3`	83 MW | 4000 USD/MWh	83 MW | 100 USD/MWh	166 MW

Tabla 4.2: Detalles de los subproblemas para la Heurística PD (HPD).

Es claro que en general, la solución resultante de ensamblar las tres no sería óptima en sí, ya que la política resultante no puede atender casos en los que una unidad hidráulica "cubre" a otra en instantes en los que la segunda no tenga agua y la primera sí. En otras palabras, la región factible se ve reducida al descomponer el problema. Asimismo, un subproblema podría tener que recurrir a la térmica cara en un instante en el que otro de los subproblemas tiene holgura para generar con la térmica barata. De ambas observaciones concluimos que el problema completo es una relajación respecto a la unión de los problemas desagregados, y por tanto, puede tener una mejor solución. Sin embargo, en el ejemplo simplificado que se resuelve en esta iteración –en donde las tres unidades hidráulicas tienen los mismos aportes– esta aproximación termina siendo muy competitiva realmente.

En la Sección 4.5 se presentan los resultados obtenidos mediante este método junto con los obtenidos por RL y las cotas inferiores.

4.4. Solución por RL

La solución mediante RL es esencialmente la misma que la utilizada en la Iteración 1, presentada en la Sección 3.6. Se utiliza el mismo algoritmo One-Step Actor-Critic mostrado en el Algoritmo 3, con un par de diferencias que se describen a continuación:

- S - En esta iteración del problema, el estado representa el nivel de los tres lagos en las represas, junto con el instante de tiempo.
- π - La política π pasa a ser una multinormal de dimensión 3:

$$\pi(a|s) = (2\pi)^{-k/2} \det(\Sigma)^{-1/2} \exp(-1/2(a-\mu)^{\intercal}\Sigma^{-1}(a-\mu))$$

  siendo Σ la matriz de covarianza, y μ el vector de medias.

Cabe destacar que el cambio en el estado hace que lo que era un vector de 5 elementos en la Iteración 1 (asumiendo 5 centroides RBF), ahora significa que el estado es un tensor de dimensiones: (5,5,5), mientras que las acciones se representan con un tensor de dimensiones (3,5,5,5), ya que se tiene el turbinado de las tres unidades hidráulicas. Esto a su vez se cumple para el horizonte de tiempo de 104 semanas en nuestro caso, por lo que el espacio completo de estados es de dimensiones (5,5,5,104), mientras que el espacio completo de acciones es de dimensiones (3,5,5,5,104). En la Figura 4.2 se puede ver un ejemplo del estado para un instante de tiempo asumiendo $3{*}3{*}3 = 27$ centroides RBF. Esta Figura es análoga a la Figura 2.10, solo que ahora se tienen 3 lagos y 27 centroides. Notar que las Gaussianas RBF no aparecen en el dibujo ya que en este caso son en 4 dimensiones.

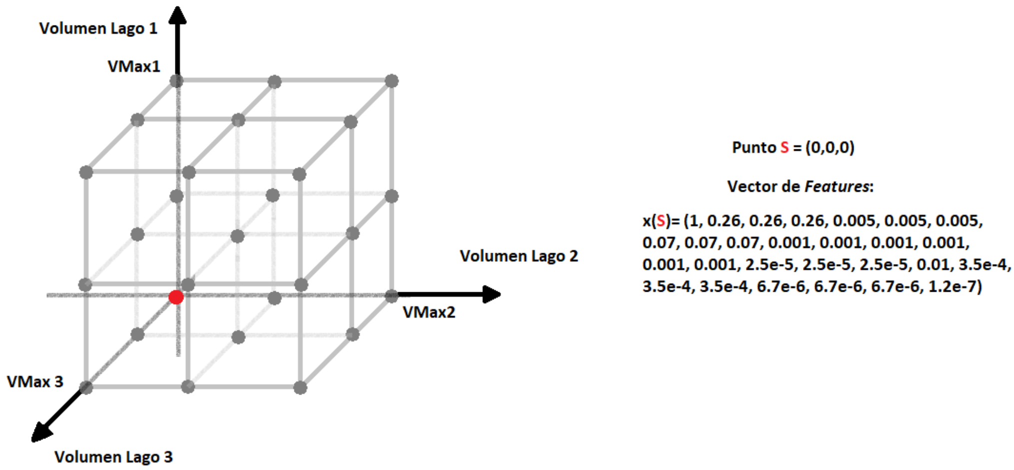


Figura 4.2: Ejemplo del estado en un instante de tiempo para 27 centroides RBF y vector de features de ejemplo para el punto (0,0,0).

El procedimiento del experimento es análogo al presentado en la Sección 3.6 con la diferencia de que ahora se tienen 3 lagos, por lo que al sortear el estado inicial del sistema en un episodio, se sortea el volumen de los 3 lagos junto con el instante de tiempo. Además, el conjunto de 2000 años de aportes de referencia, ahora contiene aportes para los 3 lagos, y a la hora de evaluar, los 3 lagos comienzan con la mitad de su volumen total; es decir: `H1` = 4100 hm^3, `H2` = `H3` = 2050 hm^3.

La siguiente sección muestra los resultados obtenidos mediante RL, y su comparación con los otros métodos.

4.5. Experimentos y Resultados

4.5.1. Casos para Experimentación

Se realizaron experimentos para varios casos distintos de la instancia del problema que encara esta iteración. Los casos se crearon variando la cantidad de aportes a alguna o todas las unidades hidráulicas. La Tabla 4.3 describe los casos en sus tres primeras columnas, mostrando la cantidad de aportes que tiene cada unidad hidráulica (por ejemplo 100 %,50 %,50 % para el caso F). Además muestra, en las dos últimas columnas, el resultado obtenido al resolver el caso como un problema determinístico usando LP, y resolviendo los 1000 escenarios de referencia mediante LP y promediando.

En la Tabla 4.3 se puede apreciar lo mencionado en la Sección 4.2.2 de que

CASO	Aportes H1	Aportes H2	Aportes H3	DET MUSD	PROM DET MUSD
A	100 %	100 %	100 %	2041.1	2376.8
B	95 %	95 %	95 %	2747.9	2952.1
C	90 %	90 %	90 %	3454.7	3573.1
D	85 %	85 %	85 %	4161.6	4227.1
E	80 %	80 %	80 %	4868.4	4902.7
F	100 %	50 %	50 %	6753.3	6760.8

Tabla 4.3: Descripción de los casos para el experimento (tres primeras columnas) junto a las cotas inferiores (últimas dos columnas).

la variante que promedia los LP para cada escenario del conjunto de referencia es siempre una mejor cota inferior que la versión que simplemente usa los aportes medios.

4.5.2. Análisis de Resultados

La Tabla 4.4 y la Figura 4.3 resumen los resultados obtenidos para los distintos casos en los que se experimentó. El tiempo de ejecución de los experimentos (versión RL) varía entre ~ 0.7 y ~ 1.9 minutos. Los valores presentados son las cotas inferiores, los resultados obtenidos por HPD (usando la metodología explicada en la Sección 4.3), y los resultados obtenidos mediante RL (con la metodología descrita en la Sección 4.4). A su vez, se presentan los errores absolutos y relativos de comparar los valores obtenidos mediante RL con los obtenidos mediante HPD. También se presentan el error en términos de potencia, y como porcentaje de la demanda. Estos últimos dos valores reportados resultan de interés para contextualizar los valores de errores en costo, que pueden parecer muy grandes. Recordar que la unidad térmica cara tiene un costo 40 veces superior a la barata, por lo que un pequeño error puede resultar en una diferencia grande de costo.

No obstante lo anterior, se puede ver que en algunos casos el resultado obtenido mediante RL es muy similar o hasta mejor que el obtenido mediante HPD, mientras que en otros casos se obtienen resultados peores de los deseables. Se nota nuevamente el fenómeno visto en la Sección 3.7.3, en donde a mayor nivel de aportes RL obtiene un desempeño cada vez peor, mientras que a menores aportes RL obtiene un resultado cada vez mejor. Estimamos que estos errores se atribuyen a la capacidad limitada de representación de la función de valor utilizada. Recordar que se utilizó una función lineal en el estado (en la representación del estado mediante

RBF para ser exactos) como función de valor. Se utilizó esta representación por su simplicidad para los experimentos, pero además mostró que tanto en la Iteración 1, como en algunos de los casos de la Iteración 2, se obtienen muy buenos resultados con esta representación sencilla. Originalmente se pensaba realizar una tercera iteración del problema, en donde se cambiara esta función lineal por una con mayor poder de representación, en particular podría ser una red neuronal por ejemplo, pero lamentablemente esta iteración quedó por fuera del alcance de este trabajo y queda entonces como trabajo futuro.

CASO	COTA INF MUSD	HPD MUSD	RL MUSD	Error Absoluto MUSD	Error Relativo	Error Potencia MW	Porcentaje Demanda
A	2376.7	2584.0	2996.9	412.9	15.98 %	6.3	1.25 %
B	2952.1	3136.9	3429.8	292.9	9.34 %	4.3	0.86 %
C	3573.1	3736.2	3913.8	177.6	4.75 %	2.6	0.52 %
D	4227.1	4372.5	4462.6	90.1	2.06 %	1.3	0.26 %
E	4902.7	5036.3	5056.6	20.3	0.40 %	0.3	0.06 %
F	6760.8	6992.6	6781.5	-211.1	-3.02 %	-3.1	-0.62 %

Tabla 4.4: Resultados de los casos.

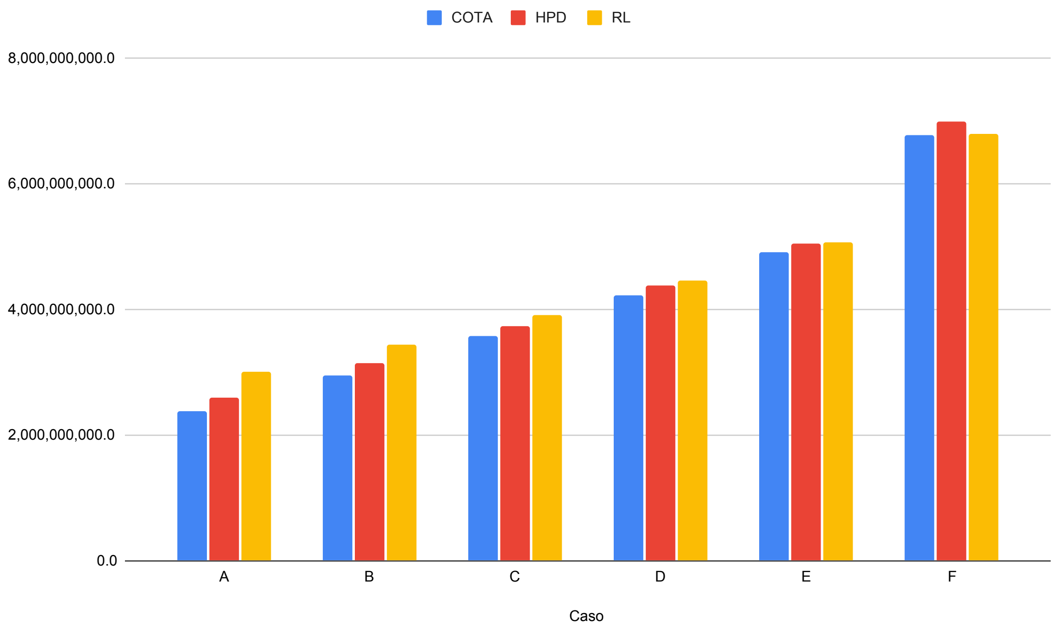


Figura 4.3: Cotas y resultados de los distintos métodos para cada caso.

Capítulo 5

Conclusiones

La solución desarrollada mediante RL obtiene resultados competitivos a los de PD en varios de los experimentos realizados, mejores en algunos casos y peores en otros. Del análisis detallado de las soluciones, estimamos que los casos en que los resultados no son muy buenos se explican por el limitado poder de representación de la función lineal que se usó para la función de valor. Como se menciona en la siguiente sección, una de las líneas de trabajo futuro sería reemplazar esta función lineal con una de mayor poder representativo como una red neuronal. Esto podría mejorar los casos en los que los resultados fueron por debajo de lo deseado.

Por otro lado, la técnica de RL resulta muy eficaz en términos de sobreponerse al problema de la “Maldición de la Dimensionalidad”. Esto queda evidenciado por los bajos tiempos de ejecución, en donde incluso en los casos más complejos estuvo por debajo de los 2 minutos.

Por estos motivos sostenemos que el RL es una técnica prometedora para la resolución del problema del Despacho Hidrotérmico Óptimo.

El trabajo culmina detallando los posibles trabajos futuros en la siguiente sección.

Trabajos Futuros

Este trabajo está estructurado de forma iterativa incremental, en donde cada iteración agrega alguna complejidad sobre la anterior. A lo largo del trabajo se mencionaron varios elementos a agregar y/o modificar en futuras iteraciones que quedaron fuera del alcance del trabajo.

Hay dos líneas principales de trabajos futuros para realizar nuevas iteraciones.

La primera consiste en agregar complejidad al problema, mientras que la segunda consiste en agregar complejidad a la solución.

La línea de agregar complejidad al problema tiene por objetivo hacer que el modelo de la realidad sobre el cual se trabaja sea lo más parecido a ésta. Recordar que en las iteraciones que se trabajó hasta ahora, se modela una versión de la realidad muy simplificada. Hay una cantidad de elementos a agregar para que el modelo sea un mejor reflejo de la realidad, por nombrar algunos, integrar las energías renovables, las cuales no son despachables, pero que nos fuerzan a usar una demanda residual estocástica, a diferencia de la demanda fija en las iteraciones en este trabajo. Otro elemento a agregar es el hecho de que las unidades hidráulicas en la realidad muchas veces no son independientes, sino que están encadenadas por lo que habría que hacer las modificaciones acordes. Otro elemento que fue simplificado es el coeficiente energético en las unidades hidráulicas, el cual se tomó como constante, mientras que en la realidad es dinámico y depende del salto de agua en la represa. En fin, como los mencionados, hay una cantidad de elementos que se pueden agregar para que el modelo se acerque más a la realidad en iteraciones futuras.

Por otro lado, está la línea que agrega complejidad a la solución. Ésta se basa en modificar la solución presentada con el objetivo de obtener mejores resultados. Uno de los claros ejemplos de esto es pasar de una función de valor lineal, como la que se usó, a usar una representación más compleja y con mayor poder representativo, como por ejemplo una red neuronal. Los cambios en esta línea de trabajos futuros abarcan un amplio rango de cambios, desde cambios pequeños; como por ejemplo modificar la heurística de inicialización del algoritmo, hasta cambios más drásticos; como lo sería, por ejemplo, usar un algoritmo de RL distinto. Como los mencionados, hay una cantidad de cambios posibles que se pueden investigar para tratar de mejorar los resultados obtenidos.

Referencias bibliográficas

ADME. (2024). *Administración del Mercado Eléctrico Uruguayo*. https://adme.com.uy

Bai, X., y Shahidehpour, S. (1996). Hydro-thermal, scheduling by tabu search and decomposition method. *IEEE Transactions on Power Systems*, *11*(2), 968-974. https://doi.org/10.1109/59.496182

Bellman, R. (1957a). Dynamic programming. *Princeton, USA: Princeton University Press*, *1*(2), 3.

Bellman, R. (1957b). A Markovian decision process. *Journal of mathematics and mechanics*, 679-684.

Bertsekas, D. (2019). *Reinforcement learning and optimal control*. Athena Scientific.

Bertsekas, D., y Tsitsiklis, J. (1996). Neuro-dynamic programming. 1996. *Athena Scientific*.

Birge, J., y Louveaux, F. (2011). *Introduction to Stochastic Programming*. Springer New York. https://books.google.com.uy/books?id=Vp0Bp8kjPxUC

Black, P. E. (1999). *Algorithms and Theory of Computation Handbook*. CRC Press LLC. https://www.nist.gov/dads/HTML/greedyHeuristic.html

Boyd, S., y Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Cancela, H. (2022). *Métodos de Monte-Carlo*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Casaravilla et. al. (2009). SIMSEE - MEMORIA FINAL DE EJECUCIóN PROYECTO PDT 47/12. https://simsee.org/db-docs/Docs_secciones/nid_10222/pdt_47_12.pdf

Chaer, R. (2008). Simulación de sistemas de energía eléctrica. *Universidad de la Republica (Uruguay). Facultad de Ingenieria*. https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/2877

Chaer et. al. (2013). Memoria Final Proyecto ANII-FSE2009-18. https://www.colibri.udelar.edu.uy/jspui/bitstream/20.500.12008/36745/1/CBTMFCGAPCC10.pdf

Dusparic, I., Harris, C., Marinescu, A., Cahill, V., y Clarke, S. (2013). Multi-agent residential demand response based on load forecasting. *2013 1st IEEE Conference on Technologies for Sustainability (SusTech)*, 90-96. https://doi.org/10.1109/SusTech.2013.6617303

EJECUTIVO, P. (2002). REGLAMENTO DEL MERCADO MAYORISTA DE ENERGíA ELECTRICA. *DIARIO OFICIAL, DOCUMENTOS PODER EJECUTIVO*, (Nro. 26.097).

El-Hawary, M., y Christensen, G. (1979). *Optimal Economic Operation of Electric Power Systems*. Academic Press. https://books.google.com.uy/books?id=lA1tAAAAIAAJ

Ferreira, G. (2008). Modelos utilizados para el despacho energético óptimo. *UTE*. https://www.bcu.gub.uy/Comunicaciones/Jornadas%20de%20Economa/iees03j3681009.pdf

Fishman, G. S. (1996). *Monte Carlo*. Springer New York. https://doi.org/10.1007/978-1-4757-2553-7

Glavic, M., Fonteneau, R., y Ernst, D. (2017). Reinforcement Learning for Electric Power System Decision and Control: Past Considerations and Perspectives [20th IFAC World Congress]. *IFAC-PapersOnLine*, *50*(1), 6918-6927. https://doi.org/https://doi.org/10.1016/j.ifacol.2017.08.1217

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley. https://books.google.com.uy/books?id=3_RQAAAAMAAJ

Gorenstin, B., Campodonico, N., Costa, J., y Pereira, M. (1991). Stochastic optimization of a hydro-thermal system including network constraints. *IEEE Transactions on Power Systems - IEEE TRANS POWER SYST*, *7*, 127-133. https://doi.org/10.1109/PICA.1991.160617

Habibollahzadeh, H., y Bubenko, J. A. (1986). Application of Decomposition Techniques to Short-Term Operation Planning of Hydrothermal Power System. *IEEE Transactions on Power Systems*, *1*(1), 41-47. https://doi.org/10.1109/TPWRS.1986.4334842

Henze, G. P., y Dodier, R. H. (2003). Adaptive Optimal Control of a Grid-Independent Photovoltaic System. *Journal of Solar Energy Engineering*, *125*(1), 34-42. https://doi.org/10.1115/1.1532005

Hillier, F., Lieberman, G., y Osuna, M. (1991). *Introducción a la investigación de operaciones*. McGraw-Hill. https://books.google.com.uy/books?id=Q_BybwAACAAJ

Humphrys, M. (2000). W-learning: Competition among selfish Q-learners.

Iribarren, G. (1999). *Cadenas de Markov gobernando algunos procesos aplicables a los ríos: Aplicaciones estadísticas a algunos ríos de la región*. Publicaciones Matemáticas del Uruguay. https://books.google.com.uy/books?id=xh3vAAAAMAAJ

Iturriaga, S., y Nesmachnow, S. (2022). *Algoritmos Evolutivos*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Jasmin, E., Imthias Ahamed, T., y Jagathy Raj, V. (2011). Reinforcement Learning approaches to Economic Dispatch problem. *International Journal of Electrical Power & Energy Systems*, *33*(4), 836-845. https://doi.org/https://doi.org/10.1016/j.ijepes.2010.12.008

Jin-Shyr, Y., y Nanming, C. (1989). Short term hydrothermal coordination using multi-pass dynamic programming. *IEEE Transactions on Power Systems*, *4*(3), 1050-1056. https://doi.org/10.1109/59.32598

Jofré, A., Ayguade, E., Cela, J. M., Bonnans, F., Piria, A., y Risso, C. (1996-1997). Proyecto de Colaboración Internacional PARALIN: Universidad Politécnica de Catalunya (ES), École Normale Supérieure de Lyon (FR), Institut National de Recherche en Informatique et en Automatique (FR), Universidad de Chile (CH), Universidad de la República (UY).

Li, D., Zhao, D., Zhu, Y., y Xia, Z. (2015). Thermal comfort control based on MEC algorithm for HVAC systems. *2015 International Joint Conference on Neural Networks (IJCNN)*, 1-6. https://doi.org/10.1109/IJCNN.2015.7280436

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., y Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. https://doi.org/10.48550/ARXIV.1312.5602

Moscatelli, S. (2022). *Introducción a la Investigación de Operaciones*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Ngundam, J., Kenfack, F., y Tatietse, T. (2000). Optimal scheduling of large-scale hydrothermal power systems using the Lagrangian relaxation technique. *International Journal of Electrical Power & Energy Systems*, *22*(4), 237-245. https://doi.org/https://doi.org/10.1016/S0142-0615(99)00054-X

OpenAI, : Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H. P. d. O., Raiman, J., … Zhang, S. (2019). Dota 2 with Large Scale Deep Reinforcement Learning. https://doi.org/10.48550/ARXIV.1912.06680

Palacios-Gomez, F., Lasdon, L., y Engquist, M. (1982). Nonlinear optimization by successive linear programming. *Management science*, *28*(10), 1106-1120.

Piria, A., Tasende, D., Ferreira, G., Tempone, R., y Fernández, E. (1994-1997). Proyecto CONICYT-BID N^{o}173: “Optimización de la coordinación hidrotérmica en el corto plazo” (HIDROTER).

Piria, A., Tasende, D., Gianoni, S., Oliveira, R., López, A., y Dretz, G. (1992-1993). Convenio UTE-FING: “Mejoras en los programas de optimización y simulación de la generación de energía eléctrica”.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality* (Vol. 703). John Wiley & Sons.

Ribeiro, C., y Hansen, P. (2001). *Essays and Surveys in Metaheuristics*. Springer My Copy UK. https://books.google.com.uy/books?id=-qbmoAEACAAJ

Risso, C., y Rodríguez-Bocca, P. (2021). *Optimización Continua y Aplicaciones*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Risso, C., Rodríguez-Bocca, P., Mauttone, A., Piñeyro, P., Romero, P., y Nesmachnow, S. (2020). *Metaheurísticas y Optimización sobre Redes*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Rotting, T., y Gjelsvik, A. (1992). Stochastic dual dynamic programming for seasonal scheduling in the Norwegian power system. *IEEE Transactions on Power Systems*, *7*(1), 273-279. https://doi.org/10.1109/59.141714

Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533-536.

Rutishauser, H., y Gutknecht, M. (1991). *Lectures on Numerical Mathematics*. Birkhäuser Boston. https://books.google.com.uy/books?id=hhrvAAAAMAAJ

Shapiro, A. (2011). Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, *209*(1), 63-72. https://doi.org/https://doi.org/10.1016/j.ejor.2010.08.007

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., y Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, *362*(6419), 1140-1144. https://doi.org/10.1126/science.aar6404

SimSEE. (2024). *Simulación de Sistemas de Energía Eléctrica*. https://simsee.org

Sutton, R. S., y Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Testuri, C. (2019). *Fundamentos de Programación Entera*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

Testuri, C. (2022). *Optimización bajo Incertidumbre*. Instituto de Computación, Facultad de Ingeniería, UdelaR.

van der Wal, J. (1981). *Stochastic Dynamic Programming: Successive Approximations and Nearly Optimal Strategies for Markov Decision Processes and Markov Games*. Mathematisch Centrum. https://books.google.com.uy/books?id=GMoGPwAACAAJ

Vázquez-Canteli, J. R., y Nagy, Z. (2019). Reinforcement learning for demand response: A review of algorithms and modeling techniques. *Applied Energy*, *235*, 1072-1089. https://doi.org/https://doi.org/10.1016/j.apenergy.2018.11.002

Vignolo, M., y Monzón, P. (2002). Deregulating the electricity sector. *Proceedings of the Second International Conference on Power and Energy Systems, IASTED. Greece*. https://iie.fing.edu.uy/publicaciones/2002/VM02

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., ... Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 1-5.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Werbos, P. (1977). Advanced forecasting methods for global crisis warning and models of intelligence. *General System Yearbook*, 25-38.

Winston, W., y Goldberg, J. (2004). *Operations Research: Applications and Algorithms*. Thomson Brooks/Cole, Belmont CA.

Wolsey, L., y Nemhauser, G. (1988). *Integer and Combinatorial Optimization*. Wiley. https://books.google.com.uy/books?id=uG4PAQAAMAAJ

Zoumas, C., Bakirtzis, A., Theocharis, J., y Petridis, V. (2004). A genetic algorithm solution approach to the hydrothermal coordination problem. *IEEE Transactions on Power Systems*, *19*(3), 1356-1364. https://doi.org/10.1109/TPWRS.2004.825896