ISSN 1688-2806

Universidad de la República
Facultad de Ingeniería

Programación Dinámica Estocástica Dual aplicada a la optimización del sistema eléctrico uruguayo

Tesis presentada a la Facultad de Ingeniería de la Universidad de la República por

Rodrigo Porteiro

en cumplimiento parcial de los requerimientos para la obtención del título de Magister en Ingeniería Eléctrica.

Director de Tesis

Dr. Andrés Ferragut ..... Universidad Ort

Tribunal

Dr. Álvaro Giusto ..... IIE-FING-UDELAR
Dr. Antonio Mauttone ..... INCO-FING-UDELAR
Dr. Pablo Belzarena ..... IIE-FING-UDELAR

Director Académico

Dr. Pablo Monzón ..... IIE-FING-UDELAR

Montevideo
jueves 13 abril, 2023

Programación Dinámica Estocástica Dual aplicada a la optimización del sistema eléctrico uruguayo, Rodrigo Porteiro.

ISSN 1688-2806

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 101 páginas.
Compilada el jueves 13 abril, 2023.
http://iie.fing.edu.uy/

If people do not believe that mathematics is simple, it is only because they do not realize how complicated life is.

John Von Neumann

Esta página ha sido intencionalmente dejada en blanco.

Agradecimientos

A Andrés y a Pablo, por haberme acompañado en este camino.

A Fernando, ya que junto con Andrés compartimos muchas jornadas de discusión relacionadas a este trabajo y de él aprendí muchísimo.

A mis compañeros de UTE: Daniel, Alfonso y Manolo, que gracias a su enorme solidaridad para compartir sus conocimientos y experiencia fue que comencé este viaje y desde que trabajo con ellos me hicieron sentir como uno más.

A mi esposa Gaby, que siempre me apoyó y sin ella no hubiera sido posible ni siquiera empezar este trabajo.

A mis hijas Clara y Juana, por su inocencia y amor incondicional a pesar de algún que otro abrazo que no les pude dar por estar embarcado en este desafío.

Finalmente, a mis padres. A mamá por haberme enseñado la importancia de estar toda la vida aprendiendo. Y muy especialmente a papá, que hace poco nos dejó físicamente y sin embargo sigo aprendiendo día a día simplemente recordando cada uno de los consejos que me dio. ¡Cómo te extraño viejo!, gracias.

Esta página ha sido intencionalmente dejada en blanco.

Dedicada a papá.

Esta página ha sido intencionalmente dejada en blanco.

Abstract

The operation of the electric grid in systems with a large hydroelectric component or other type of energy storage is often cast as a dynamic programming problem, in which state variables represent the stored energy. To avoid the curse of dimensionality in discretizing such states, the SDDP technique has been successfully applied. However, new demands are being placed on the optimization, with the penetration of renewable sources of faster variability, and the possible incorporation of shorter term energy storage. These new components expose two difficulties. The high variability of renewable resources and short-term storage requires representing faster time scales, making it more complex to represent the dispatch problem of a stage. Also, new sources of uncertainty arise wich lead to mitigating the curse of stochastic dimensionality, an aspect not addressed by the standard SDDP technique.

A preliminary extension of the SDDP framework is presented to consider two different time scales and a markovian model to represent the hydrological situation. The method is applied to a stylized model of the Uruguayan electricity system, relying on new open source implementations of SDDP to carry out the computations. The analysis of the results focuses on the execution times and indicates that the method remains treatable despite the increased dimension of the problem.

Then, the problem of mitigating the curse of stochastic dimensionality is addressed. For this, a specific software library was developed to be able to incorporate technical aspects not considered in existing libraries. With this tool, methods were analyzed to represent uncertainty, seeking to reduce the number of optimization problems to be solved. The results show a considerable reduction in execution times, but not enough to address problems of a size similar to those required by the industry.

Finally, various cutting pruning techniques are implemented seeking to reduce the complexity of each of the dispatch problems. Good results are obtained in terms of algorithmic convergence with a reduction in execution times. However, this reduction is still insufficient.

It is concluded that the SDDP technique applied to the long-term planning of small systems with a high component of renewable resources should be considered only when the dimension of the state space is high. When the state space dimension is small, the curse of stochastic dimensionality is the one that appears as a limitation, forcing an increase in the number of linear problems to be solved. It is also observed that the impact on execution times associated with the complexity of the dispatch problem, such as the number of constraints or the number of integer variables, is not significant compared to the impact produced by the two curses of dimesionality studied.

Esta página ha sido intencionalmente dejada en blanco.

Resumen

La operación de la red eléctrica en sistemas con un gran componente hidroeléctrico u otro tipo de almacenamiento de energía a menudo se presenta como un problema de programación dinámica, en el que las variables de estado representan la energía almacenada. Para evitar la maldición de la dimensionalidad al discretizar tales estados, se ha aplicado con éxito la técnica SDDP. Sin embargo, están surgiendo nuevos desafíos al abordar la optimización, con la penetración de fuentes renovables de mayor variabilidad, y la posible incorporación de almacenamiento de energía de plazos cortos. Estos nuevos componentes exponen dos dificultades. La alta variabilidad de los recursos renovables y el almacenamiento a corto plazo requieren representar escalas de tiempo más rápidas, lo que hace más complejo plantear el problema de despacho de un paso de tiempo. Además, surgen nuevas fuentes de incertidumbre y esto conduce a mitigar la maldición de la dimensionalidad estocástica, un aspecto no abordado por la técnica SDDP estándar.

En primer lugar, se presenta una extensión preliminar de la técnica SDDP para considerar dos escalas de tiempo diferentes y un modelo markoviano para representar la situación hidrológica. El método se aplica a un modelo estilizado del sistema eléctrico uruguayo, apoyándose en nuevas implementaciones de código abierto de SDDP para realizar los cálculos. El análisis de los resultados se centra en los tiempos de ejecución e indica que el método sigue siendo tratable a pesar de la mayor dimensión del problema.

Luego, se aborda el problema de mitigar la maldición de la dimensionalidad estocástica. Para esto, se desarrolló una biblioteca de software específica para poder incorporar aspectos técnicos no considerados en las bibliotecas existentes. Con esta herramienta se analizaron métodos para representar la incertidumbre, buscando reducir el número de problemas de optimización a resolver. Los resultados muestran una reducción considerable en los tiempos de ejecución, pero no lo suficiente como para abordar problemas de un tamaño similar a los que requiere la industria.

Finalmente, se implementan diversas técnicas de poda de cortes buscando reducir la complejidad de cada uno de los problemas de despacho. Se obtienen buenos resultados en términos de convergencia algorítmica con una reducción en los tiempos de ejecución. Sin embargo, esta reducción sigue siendo insuficiente.

Se concluye que la técnica SDDP aplicada a la planificación de largo plazo de sistemas pequeños con un alto componente de recursos renovables, debe ser considerada solamente cuando la dimensión del espacio de estados es alta. Cuando la dimensión del espacio de estados es pequeña, la maldición de la dimensionalidad estocástica es la que aparece como una limitante, obligando a aumentar el número de problemas lineales a resolver. Se evidencia también que el impacto en los tiempos de ejecución asociado a la complejidad del problema de despacho, como el número de restricciones o el número de variables enteras, no es significativo frente al impacto producido por las dos maldiciones de dimensionalidad estudiadas.

Esta página ha sido intencionalmente dejada en blanco.

Prefacio

La realización del presente trabajo es la conjunción de una serie de hechos que se fueron dando a lo largo de mi vida. Desde mi etapa escolar tuve un gran interés por las matemáticas, en particular por la resolución de problemas. En mis años de liceal comencé a entender que utilizando a las matemáticas como herramienta, es posible contribuir a la solución de problemas de gran relevancia para el desarrollo de la humanidad. Más adelante ya en mis años universitarios, decidí formarme para poder aplicar la matemática en problemas relevantes. Fue así que descubrí que a medida que sucedían los avances tecnológicos, la combinación de la matemática con la computación resultaban fundamentales para poder contribuir a la comunidad y me formé en ingeniería informática. Sin embargo, luego de mucho aprendizaje, yo sentía que no era sencillo encontrar problemas importantes en donde aplicar lo aprendido de forma de resolverlos y agregar valor de una u otra forma.

Todo cambió cuando comencé a trabajar en UTE. Mis primeros años en la empresa, trabajando en el área de distribución, me permitieron explorar decenas de problemas que se presentaban de muy alta complejidad, para los cuales las herramientas en las que me había especializado podían ser útiles. Esos años de exploración sirvieron para que comprenda la verdadera complejidad detrás de la realidad, complejidad que muchas veces no se presenta en etapas de aprendizaje académico con toda claridad.

Luego de esos años de mucho aprendizaje (a pesar de no haber aplicado prácticamente nada de lo aprendido), me cambié de unidad dentro de la empresa y pasé a formar parte del área de planificación. En ese momento fue que por primera vez tuve contacto con el problema que se aborda en esta tesis. Para mí tuvo un gran impacto observar que el conocimiento obtenido a lo largo de los años me permitía contribuir en infinidad de mejoras en el modelado del sistema eléctrico. Muchas de estas contribuciones pueden traducirse en el ahorro de millones de dólares para UTE y por lo tanto para el Uruguay. Esto resulta extremadamente motivante y me ha llevado a transitar diversos caminos muy desafiantes, uno de los cuáles se inicia con el trabajo que presento en esta Tesis.

Rodrigo Porteiro

Esta página ha sido intencionalmente dejada en blanco.

Tabla de contenidos

Capítulo 1

Introducción

En este capítulo se presenta, en líneas generales, el problema que se aborda en el presente trabajo y cuáles son las motivaciones para estudiarlo. Además, se describe la estructura general del documento.

1.1. Motivación y descripción del problema a resolver

El despacho económico de generación eléctrica para abastecer la demanda de los consumidores ha sido a lo largo de los años un problema de gran interés en empresas del sector eléctrico verticalmente integradas. Más recientemente, la solución a este problema se ha abordado por los operadores del sistema a cargo de la administración eléctrica y energética. El problema presenta muchas facetas que se resumirán en este trabajo. Para un tratamiento más profundo sobre el abordaje de este problema se pueden ver los siguientes trabajos [12, 24, 25, 30, 31].

Un problema estándar que se ha ha resuelto a través de diversas técnicas de optimización es el del despacho hidro-térmico. En este problema se dispone de un parque de generación con centrales térmicas e hidráulicas con embalse. Los costos de generación de las máquinas térmicas resultan conocidos, aunque pueden variar en el tiempo (generalmente están relacionados al precio del petróleo). Los costos de generación hidráulica son nulos, pero utilizan la energía almacenada en forma de agua, energía que se repone a través de un proceso no controlado de aportes hidrológicos. Es por eso que la decisión de utilizar en el presente o diferir al futuro el uso del agua, tiene un costo implícito que no es otra cosa que el costo de oportunidad por sustituir máquinas térmicas a futuro. O sea, si se utiliza el agua en el presente y no se repone a través de aporte, es posible que en el futuro no se disponga de agua y haya que incurrir en grandes costos térmicos. A su vez, si no se utiliza el agua y se incurre en un costo térmico en el presente y luego los aportes hidráulicos futuros obligan a verter agua sin ser utilizada para generar, se incurre en una ineficiencia que tiene un costo para el operador. Es por eso que habitualmente se utilizan técnicas de optimización buscando minimizar el costo de abastecimiento de la demanda eléctrica sujeto a las restricciones técnicas que presenta el parque y a las incertidumbres presentes en los aportes hidráulicos, roturas de las máquinas, demanda eléctrica, entre otras variables. El resultado que se obtiene al optimizar un sistema como el descripto es una política de operación que depende del estado del sistema eléctrico (en este caso del estado de los embalses hidráulicos) y decide como operar el sistema en cada paso de tiempo. Dependiendo de las características del parque y si el propósito es modelar el corto, mediano o largo plazo, los

pasos a considerar pueden ser horarios, diarios o semanales; y los horizontes temporales pueden ir desde unos pocos días a decenas de años.

En sistemas con amplia capacidad de almacenamiento de energía, en particular en aquellos con represas hidroeléctricas con grandes lagos, las decisiones de despacho quedan acopladas a lo largo de escalas de tiempo muy largas (semanas, meses o incluso años). Este acoplamiento se debe a que la disponibilidad de energía en forma de agua embalsada depende fuertemente de patrones de aportes hidráulicos con dinámicas muy lentas (por ejemplo, en Uruguay se han registrado en la historia sequías de duración mayor a un año). Es por esta razón que se requiere optimizar la decisión de utilizar la energía en el presente o diferir su uso a lo largo del tiempo, en particular a través del almacenamiento de agua (o cualquier otro tipo de acumulación de energía, p.e. baterías). Resulta natural abordar el problema utilizando técnicas de Programación Dinámica Estocástica (SDP) [9, 38], en donde el estado del sistema se puede representar con al menos una variable por cada recurso de almacenamiento de energía (por ejemplo, volumen de un lago o capacidad de una batería). Computar la función de valor para cada una de las discretizaciones de estas variables de estado sufre del conocido problema de la maldición de la dimensionalidad de Bellman (o de estados). En [32], Pereira y Pinto introducen la Programación Dinámica Estocástica Dual (SDDP), una estrategia de aproximación que permitió el desarrollo de aplicaciones de uso industrial [2] que consiguen abordar el problema de operación y despacho económico con una cantidad significativa de variables de estado. Más recientemente, este método recibió una importante atención desde la academia [18,20,21,33], y nuevas implementaciones de esta técnica encuentran disponibles en repositorios públicos de software [16, 37]. Esta técnica, que resulta ser el centro de este trabajo, se explica en detalle en la Sección 2.3.

Las herramientas que resuelven el despacho económico de un sistema eléctrico deben ser capaces de incorporar más características y detalles ante el dinamismo del sector. En particular, en los últimos años se han producido cambios sustanciales debido a la gran penetración de generación mediante fuentes renovables como la energía eólica y la energía solar, que tienen la característica de ser no despachables y presentar una gran variabilidad en escalas temporales muy pequeñas. Su incorporación directa en el algoritmo de programación dinámica estocástica implicaría reducir el paso de tiempo para considerar una escala horaria y a su vez mantener una representación adecuada de los recursos que operan con escalas temporales largas (principalmente el recurso hidráulico), lo que se traduce en un problema de altísima complejidad. Además, si se incorporan recursos de almacenamiento de corto plazo (por ejemplo, baterías) o dispositivos de gestión de la demanda eléctrica para mitigar la variabilidad mencionada, se requieren variables de estado adicionales que modelen estos recursos, lo que agrava el problema de la maldición de la dimensionalidad de estados. En el trabajo de Costa [13] se aborda el problema de la incorporación de almacenamiento y en el trabajo de Zhang [41] la incorporación de dispositivos de gestión de la demanda, en ambos utilizando variantes de SDP.

Otra de las dificultades que se presentan para resolver de manera adecuada el problema planteado es la naturaleza estocástica del mismo. Para capturar la incertidumbre de las variables que afectan el despacho económico de forma adecuada, estas se deben considerar como variables aleatorias modeladas en procesos estocásticos que capturen su distribución conjunta cuando las mismas están correlacionadas. Por ejemplo, las variables aleatorias que representan los aportes hidráulicos de diferentes cuencas se encuentran correlacionadas, por lo que resulta apropiado construir un único proceso estocástico que capture dicha correlación. Un aspecto importante a tener en cuenta que resulta crucial en este trabajo es que la Programación Dinámica Estocástica y sus variantes requieren conocer las distribuciones probabilísticas de todas las variables

aleatorias. Habitualmente las distribuciones no se conocen a priori, por lo que deben ser estimadas mediante métodos de Montecarlo realizando un muestreo de los procesos estocásticos apropiados. Esta forma de estimar las distribuciones puede conducir a lo que se conoce como la segunda maldición de la dimensionalidad [35]. Básicamente, esta maldición consiste en que la cantidad de muestreos que hay que realizar para capturar la distribución conjunta crece en forma exponencial dependiendo de la cantidad de variables aleatorias y la estructura estadística de las mismas.

Por lo anteriormente expuesto, es necesario alcanzar un adecuado equilibrio entre la precisión del modelo y el costo computacional, lo que no resulta trivial. Un factor importante es que no siempre los problemas requieren una solución simplificada por restricciones algorítmicas, ya que diversas técnicas matemáticas frecuentemente no pueden ser implementadas por restricciones computacionales (capacidad de cómputo, capacidad de almacenamiento). Esto extiende el problema al ámbito de la computación y hace que sea imprescindible, ante los vertiginosos avances tecnológicos, replantearse problemas que en el pasado no eran resolubles sin hacer profundas simplificaciones. Finalmente, el enorme potencial de reducción de costos o aumentos de beneficios ante una resolución más detallada del problema, resalta la importancia económica con fuerte incidencia a nivel país. Esto convierte al problema abordado en esta tesis en un desafío muy motivante.

1.2. Contribuciones de esta tesis

Este trabajo apunta a analizar el potencial que tiene el algoritmo de SDDP de ser aplicado a un sistema eléctrico pequeño, pero en constante crecimiento y en constante cambio de estructura, como el de Uruguay. No sólo los recientes cambios en el sistema uruguayo que incorporaron gran cantidad de generación eólica y solar hacen que sea más complejo el modelado, sino también los cambios que se esperan para el mediano plazo. En este aspecto, la necesidad de acumulación de energía, ya sea en forma de baterías o de otro tipo de acumuladores, debido a que no hay más capacidad de acumulación hidráulica en los ríos de Uruguay, hacen muy necesario el análisis de acumuladores desde el punto de vista del modelado. A su vez, muchos proyectos vinculados con gestión de la demanda, hidrógeno verde, contratos interrumpibles, aparecen en el horizonte.

Las principales contribuciones de esta Tesis en línea con lo mencionado en el párrafo anterior tienen que ver con:

1. Análisis de la técnica SDDP utilizando varias escalas de tiempo y evaluando el impacto de la incorporación de variables de estado. Esto contribuye fuertemente a la mejora en el modelado de recursos de gran variabilidad en ventanas de tiempo muy pequeñas y a la representación de acumuladores u otros componentes para los que su modelado requiere variables de estado (ciclos combinados, contratos interrumpibles, centrales de bombeo, baterías, etc.). Todos estos aspectos no podrían ser modelados utilizando técnicas de programación dinámica estocástica con discretización del espacio de estados.

2. Desarrollo de una biblioteca SDDP específica para la resolución del problema de operación de un sistema eléctrico. Esta biblioteca resulta muy útil como punto de partida para el desarrollo de una herramienta industrial que aplique SDDP, pero haciendo hincapié en sistemas relativamente pequeños. Este desarrollo se realizó en Python, un lenguaje que está en continuo crecimiento y que permitiría a futuro incorporar diversas técnicas de inteligencia artificial y computación paralela con facilidad.

3. Por último, este trabajo contribuye al análisis de técnicas de reducción del espacio aleatorio considerado en el algoritmo SDDP. Este aspecto está en continuo desarrollo en la comunidad científica y en este trabajo se le dio un enfoque muy simple intentando plantear una solución adecuada a un sistema pequeño, en donde la maldición de la dimensionalidad estocástica se hace presente.

En resumen, este trabajo introduce técnicas de modelado que se adaptan al gran dinamismo que se presenta en los sistemas eléctricos actuales. Además, estas técnicas planteadas permiten aprovechar la gran evolución que ha habido en los últimos años de la industria informática, no sólo en términos de capacidad de cálculo sino que también en lo que refiere a capacidad de almacenamiento y manejo de grandes volúmenes de datos.

1.3. Estructura del documento

La Tesis se organiza en seis capítulos y un apéndice.

El Capítulo 1 es la introducción al trabajo realizado y se presentan las principales contribuciones de la Tesis.

En el Capítulo 2, se presenta el algoritmo Programación Dinámica Estocástica Dual (SDDP) en detalle. Además, se describe un modelo markoviano para incorporar aportes hidrológicos en un esquema SDDP para la resolución del problema de operación en un sistema eléctrico genérico con presencia hidroeléctrica. Por último, se presenta una representación del tiempo que considera dos escalas, una semanal asociada al paso de tiempo del algoritmo SDDP y otra horaria que permite detallar cada paso.

En el Capítulo 3 se presenta el diseño y la metodología de estimación de un modelo Markoviano para representar la situación hidrológica del sistema y aplicarlo en la resolución del problema de operación con despacho económico de energía eléctrica. Este modelo se presenta para el sistema uruguayo y se basa en la información de caudales de aportes históricos. Adicionalmente, se proponen modificaciones en las ecuaciones del problema de despacho de forma de contemplar dos escalas de tiempo. La utilidad de la metodología propuesta, radica en poder incorporar en el modelado, recursos que operan a escalas muy distintas de tiempo. En particular los recursos solar y eólico presentan gran variabilidad en pocas horas, mientras que el recurso hidráulico tiene una dinámica que puede operar en meses o incluso años.

En el Capítulo 4 se presenta la aplicación de SDDP utilizando el modelo markoviano y la representación de dos escalas de tiempo analizados en los capítulos previos para la resolución del problema de operación con despacho económico de energía eléctrica, aplicado al sistema uruguayo. Utilizando la biblioteca que resuelve SDDP [16] desarrollada en el lenguaje Julia [10], se implementa la solución planteada. Luego se incorpora una batería al problema inicial ampliando el espacio de estados, para analizar el impacto que produce esta nueva variable tanto en el óptimo del problema como en el tiempo de ejecución de la resolución. Se publicó una versión de este capítulo en [34].

Debido a las limitaciones que posee la biblioteca SDDP genérica en Julia empleada en el Capítulo 4, se implementó una biblioteca en Python específica para resolver el problema de operación de un sistema eléctrico aplicando SDDP. En el Capítulo 5 se describen la estructura y las funcionalidades de la biblioteca.

En el Capítulo 6 se presentan una serie de problemas a abordar, identificados en el Capítulo 4, con el objetivo de reducir el tiempo de cómputo del algoritmo SDDP utilizando la biblioteca desarrollada. Al ser este paquete de desarrollo propio, fue posible incorporar fácilmente modificaciones al algoritmo SDDP estándar. Con esta nueva herramienta, se realizaron varios estudios experimentales que permitieron realizar un

análisis de las distintas limitaciones que posee el algoritmo SDDP aplicado a un sistema de características similares al uruguayo.

Finalmente, en el Capítulo 7 se presentan las conclusiones del trabajo realizado en el marco de la tesis y se identifican con precisión las posibles ventajas, pero sobre todo las grandes limitaciones que posee SDDP aplicada a un sistema como el uruguayo. Se establecen varias líneas de investigación a futuro que tienen como objetivo reducir las limitaciones de las metodologías analizadas en esta Tesis para la resolución del problema de despacho económico de energía eléctrica.

El Apéndice A presenta una especificación con mayor detalle del uso de la biblioteca presentada en el Capítulo 5, haciendo hincapié en los aspectos informáticos.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 2

Programación dinámica estocástica dual

En el presente capítulo, se presenta en general la técnica de programación dinámica estocástica dual, que se aplicará a lo largo de este trabajo en los diversos estudios. En el presente capítulo, no se consideran modelos de Markov ni se contemplan distintas escalas de tiempo. Estos dos aspectos se analizan en el Capítulo 3, con el objetivo de representar adecuadamente la aleatoriedad y las diferentes dinámicas de los recursos asociados a un sistema eléctrico.

2.1. Introducción

En muchas situaciones en las que se deben tomar decisiones a lo largo del tiempo, sucede que algún dato (generalmente del futuro) es incierto. La programación estocástica, en términos generales, aborda este tipo de problemas a través de una estructura básica de dos etapas. En la primera etapa se toman las decisiones, luego se revela la información que posee incertidumbre y en una segunda etapa conociendo la información revelada se toma otra decisión. Planteado este esquema en dos etapas, el problema consiste en determinar las decisiones óptimas de manera de minimizar la esperanza del costo futuro, o maximizar la esperanza del beneficio futuro capturando de esta manera la incertidumbre.

Un ejemplo clásico, que naturalmente posee este tipo de estructura, es el problema del vendedor de diarios (*newsvendor*, en inglés) [17, 19]. Este problema consiste en decidir la cantidad de periódicos óptima a encargar para maximizar los beneficios siendo la demanda de diarios incierta. Una vez revelada la demanda, se incurre en un costo en caso de haberla subestimado (se podría haber vendido más) pero también se incurre en un costo en caso de haberla sobrestimado (quedan periódicos sin vender).

En muchas ocasiones se presentan problemas de múltiples etapas que se pueden interpretar cómo una serie de problemas de dos etapas encadenados. En estos casos, la estructura a partir de la etapa inicial consiste en: tomar una decisión, esperar que se revele la información previa a la segunda etapa, tomar la decisión de la segunda etapa, esperar que se revele la información de la tercera etapa, y así sucesivamente en un horizonte dado de tiempo. En la Sección 1, se observó que el problema de operación del sistema eléctrico se puede modelar mediante esta estructura, por lo que su resolución se aborda habitualmente utilizando técnicas de Programación Dinámica Estocástica (SDP). Este abordaje se ha desarrollado a lo largo de los años en la industria energética

reflejándose este desarrollo en diversos trabajos [24, 30, 31, 36].

Teniendo en cuenta que la resolución de este tipo de estructuras mediante la discretización de la función de valor, resulta en la maldición de la dimensionalidad del espacio de estados [6], se propone abordar el problema mediante SDDP. Se describe en este capítulo en detalle el algoritmo SDDP, ya que es la técnica base tratada en este trabajo a partir de la cuál se exploran variantes que permitan mejorar la calidad de la solución del problema de operación de un sistema eléctrico; no sólo en cuanto a resultados sino también en términos de tiempos de ejecución.

En el Capítulo 4 se aplicará al sistema eléctrico uruguayo el algoritmo SDDP presentado. Dada la gran importancia de los aportes hidrológicos y su variabilidad de inercia larga, es necesario tener especial cuidado en el modelado de su incertidumbre. A su vez, la incorporación al sistema eléctrico de recursos renovables como ser el recurso eólico y solar, que poseen una gran variabilidad en escalas rápidas de tiempo llevan a considerar la descomposición en dos escalas en el marco del algoritmo SDDP. En el Capítulo 3 se describe el modelo markoviano que permite considerar una variable de estado hidrológica, y se describe la descomposición en dos escalas de tiempo que permitirá considerar recursos con inercia corta (como ser eólico, solar y baterías) en conjunto con recursos de inercia larga (centrales hidráulicas con embalse).

2.2. Estructura general del problema

Se considera la resolución de sobre un horizonte T, con pasos de tiempo indexados por $k = 0, \dots T-1$. Este modelo incluye un vector de estado $x_k \in \mathbb{R}^n$ que representa el estado del sistema; así como perturbaciones $w_k \in \mathbb{R}^p$ que son aleatorias y pueden tener patrones de evolución complejos, donde n es la dimensión del espacio de estados, p la del espacio aleatorio y m del espacio de control. El vector de control $u_k \in \mathbb{R}^m$ modela las acciones.

El costo por paso en este modelo se representa como una función $g_k(x_k, u_k, w_k)$ El objetivo es obtener una *política* $\pi = \{\mu_k\}_{k=0,\dots,T-1}$ tal que se alcance el siguiente óptimo:

$$\min_{\pi} E_w \left[\sum_{k=0}^{T-1} g_k(x_k, \mu_k(x_k, w_k), w_k) \right]. \tag{2.1}$$

El problema anterior se combina con las restricciones dinámicas del sistema, estas son:

$$x_{k+1} = f_k(x_k, \mu_k(x_k, w_k), w_k),$$

donde $f_k(x_k, u_k, w_k)$ es una función que asigna el estado actual al estado futuro en función de las acciones tomadas y las perturbaciones estocásticas realizadas. Es importante destacar que se define la ecuación de manera que el control μ_k dependa del estado actual y de las perturbaciones estocásticas, quedando planteado así un esquema azar-decisión [35].

El algoritmo de Programación Dinámica [9] permite desacoplar el problema de optimización sobre el el eje temporal, computando de forma recursiva la *función de costo futuro* o *función de valor*:

$$V_k(x_k) = \min_{\pi} E_w \left[\sum_{j=k}^{T-1} g_j(x_j, \mu_k(x_j, w_j), w_j) \right]. \tag{2.2}$$

Esta función, computa para un estado, el costo mínimo según la política de decisión óptima que resulta de sumar los costos de cada uno de los pasos de la trayectoria

óptima hasta el final del horizonte temporal. Dado que esta es una función que toma un valor real (vector de estados), se acostumbra computarla a través de una discretización del espacio de estados. La maldición de la dimensionalidad de Bellman [35], establece que el número de evaluaciones crece exponencialmente con la dimensión de este espacio. Generalmente, esto restringe el alcance de las técnicas clásicas de programación dinámica a unas pocas variables de estado y estructuras de discretización de paso grueso. Es por eso que considerando la misma estructura se pone foco en el algoritmo SDDP.

2.3. SDDP

Una poderosa alternativa a la técnica que discretiza el espacio de estados, que ofrece garantías de convergencia, es la programación dinámica dual estocástica, propuesta por primera vez en [32], pero cuyas raíces se remontan al trabajo de Kelley sobre métodos de planos de corte [29] y las técnicas de descomposición de Benders [8]. SDDP se aplica en el caso donde los costos g_k son funciones convexas y las restricciones dinámicas f_k son funciones afines; en este caso, la función de valor (2.2) es convexa, lo que garantiza que cualquier hiperplano soporte de la función de valor constituya una cota inferior global del costo total.

Computar un nuevo hiperplano

Teniendo en cuenta esta observación, el algoritmo SDDP realiza una serie de iteraciones o *pasadas* donde en cada etapa k se agrega un nuevo hiperplano de soporte para V_k. Por lo tanto, en la pasada l del algoritmo, se tiene un límite inferior global V_k^l del costo. Este límite es calculado como el máximo sobre todos los hiperplanos agregados, es decir, una función convexa lineal a tramos. Para calcular un nuevo hiperplano soporte se resuelve:

$$\begin{aligned}\hat{\beta}_k^{(l+1)}(w) = \min_{x,u} \left\{ g_k(x,u,w) + V_{k+1}^{(l+1)}(f_k(x,u,w)) \right\}, \\ s.t. \quad x = x_k^{(l)} \quad [\hat{\lambda}_k^{(l+1)}(w)].\end{aligned}$$

es decir, un paso de la iteración de Bellman con la entrada $V_{k+1}^{(l+1)}$, que es la estimación actual del valor de Bellman. La restricción añadida genera un valor objetivo $\hat{\beta}_k^{(l+1)}(w)$ y una variable dual $\hat{\lambda}_k^{(l+1)}(w)$ por cada realización w. Promediando sobre las realizaciones de w se obtiene:

$$\begin{aligned}\beta_k^{(l+1)} &= E[\hat{\beta}_k^{(l+1)}(w)] \\ \lambda_k^{(l+1)} &= E[\hat{\lambda}_k^{(l+1)}(w)]\end{aligned}$$

De la optimización se sigue que:

$$\beta_k^{(l+1)} + (\lambda_k^{(l+1)})^T (x_k - x_k^{(l)})$$

es un hiperplano soporte para V_k en $x = x_k^{(l)}$. Esto agrega un corte que mejora la estimación de la función de valor en cada pasada. Los detalles de implementación se discutirán al aplicar el método en la Sección 4.2. Las propiedades de convergencia y una implementación detallada se dan en [20].

Para que tenga sentido la aproximación por hiperplanos de la función de costo futuro, se requiere que dicha función sea convexa, es por eso que el problema a resolver en cada estado que computa el algoritmo se puede expresar como:

$$V_k(x_k, w) = \min_{x,u} \{g_k(x, u, w) + \theta_k\},$$
$$s.t. \quad x = x_k^{(l)} \quad [\hat{\lambda}_k^{(l+1)}(w)]$$
$$\theta_k \geq \beta_k^{(l+1)} + (\lambda_k^{(l+1)})^T (x_k - x_k^{(l)})$$

En la expresión anterior, se puede ver que $\hat{\lambda}_k^{(l+1)}(w)$ es un subgradiente de la función $V_k(x_k, w)$ respecto de la variable x_k. Al tener el problema expresado de esta forma, es posible definir dos fases dentro de cada iteración de la solución. En la primera fase se realiza una pasada hacia adelante, utilizando la política óptima dada por la aproximación actual de costo futuro (en la primera iteración esta función es idénticamente nula ya que el algoritmo no ha incorporado ningún hiperplano). La segunda fase de una iteración consiste en la pasada para atrás, en la que se agrega para cada paso desde el último al primero, un nuevo hiperplano de acuerdo a lo definido, para cada uno de los estados que fueron visitados en la pasada hacia adelante. De esta forma, se enriquece la aproximación de la función de valor actual, en cada uno de los pasos de tiempo.

El algoritmo completo consiste en la aplicación sucesiva de iteraciones hasta que se cumpla un criterio de parada predeterminado. En el siguiente esquema algorítmico 1 se presenta un seudocódigo del algoritmo SDDP.

2.4. Ejemplo de aplicación al problema del vendedor de periódicos

El problema de *newsvendor* [17], suele plantearse en una sola etapa. En primer lugar, se toma la decisión de cuántos periódicos comprar para abastecer una demanda futura (eso se suma a lo que hay en stock), luego se revela la información de la demanda. Existen tres situaciones: puede que se haya comprado en exceso, que la cantidad sea justa, o que falten periódicos para abastecer la demanda. Si se compró en exceso, se deben descartar periódicos y si los periódicos no son suficientes, se pueden comprar periódicos de forma urgente, pero a mayor precio. Cómo hay incertidumbre en la demanda, se debe resolver un problema de optimización con incertidumbre. Dada la simpleza del problema, el mismo puede ser resuelto utilizando técnicas estándar de programación estocástica.

En esta sección, se presenta un ejemplo en el cual el problema es en tres etapas. Se puede interpretar como que son dos días de venta de periódicos en el que antes del primer día se decide una compra. Si el primer día hubo escasez, se compran periódicos a un precio mayor que el precio inicial para cubrir el faltante y si sobran periódicos, se dispone de ellos para el segundo día. Luego de revelada la demanda del segundo día, si hay escasez se puede comprar a un precio más elevado para cubrir el faltante del segundo día.

El problema de optimización queda expresado en el esquema 2.3. En donde w_1 y w_2 son variables aleatorias que representan la demanda de periódicos que se revela en el día 1 y en el día 2. La variable x_i es el stock en el paso i, p_1 representa el precio inicial, p_2 y $p3$ son el precio del periódico si hay que comprarlo de urgencia el primer día o el segundo día, respectivamente. Finalmente, u_i con $i = 0.,2$ representa la cantidad de periódicos que se decide comprar en cada etapa, esto es, antes del primer día, luego de conocida la demanda w_1 y luego de conocer la demanda w_2.

Este problema se puede descomponer en tres etapas, de forma de aplicar el algoritmo SDDP y validar el óptimo.

Algorithm 1 Algoritmo SDDP

$K = 0$
$L = []$
$i = 0$ ▷ Paso de tiempo, el último es T
while $K \leq Max_{iter}$ **do** ▷ Pasada hacia adelante
  $x = x_i$
  while $i < T$ **do**
    obtener realización $w_i \in W$
    resolverProblemaPaso(x_i, w_i) ▷ se obtiene x_i' según la dinámica
    agregar$((x_i), L)$ ▷ se agrega el estado a la trayectoria
    $x = x_i'$
    $i = i + 1$
  end while
  $L_{inv} = invertirLista(L)$ ▷ Comienza pasada hacia atrás
  $i = T - 1$
  for $(x) \in L_{inv}$ **do**
    for $w_i \in W$ **do**
      $\beta_k^{(l+1)}, \lambda_k^{(l+1)} = resolverProblemaPaso(x_i, w_i)$
      ▷ el óptimo y variables duales asociadas al estado
    end for
  end for
  $\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w_k)]$
  $\lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w_k)]$
  $actualizar(V, i, \beta_k^{(l+1)}, \lambda_k^{(l+1)})$
    ▷ Se agrega el nuevo corte a la aproximación de Bellman del paso i.
  $K = K + 1$
end while

$$\begin{aligned} &\min E[p_1 u_1 + p_2 u_2 + p_3 u_3] \\ &s.t. \\ &x_1 = x_0 + u_1, \\ &x_1 + u_2 \geqslant w_1, \\ &x_2 = x_1 + u_2 - w_1, \\ &x_2 + u_3 \geqslant w_2. \end{aligned} \tag{2.3}$$

En primer lugar, se debe considerar la decisión para el paso inicial, asumiendo que la información futura (en este caso la del segundo paso y el final), está dada por la función de costo futuro. Por lo que el el paso inicial tiene la siguiente estructura definida en el esquema 2.4.

$$\begin{aligned} &\min E[p_1 u_1 + \theta_1] \\ &s.t. \\ &x_1 = x_0 + u_1, \\ &\theta_1 \geqslant \alpha_1^{(l)} x_1 + \beta_1^{(l)}. \end{aligned} \tag{2.4}$$

Puede verse que la variable θ_1 permite modelar la función de costo futuro en el objetivo, ya que existe un conjunto de expresiones lineales que imponen restricciones

sobre θ_1 que son los hiperplanos que se recolectan en cada iteración del algoritmo SDDP.

Por otra parte, la estructura del problema del segundo paso se define en el el esquema 2.5.

$$\begin{aligned} &\text{mín } E[p_2 u_2 + \theta_2] \\ &s.t. \\ &x_2 = x_1 - w1 + u_2, \\ &\theta_2 \geqslant \alpha_2^{(l)} x_2 + \beta_2^{(l)}. \end{aligned} \tag{2.5}$$

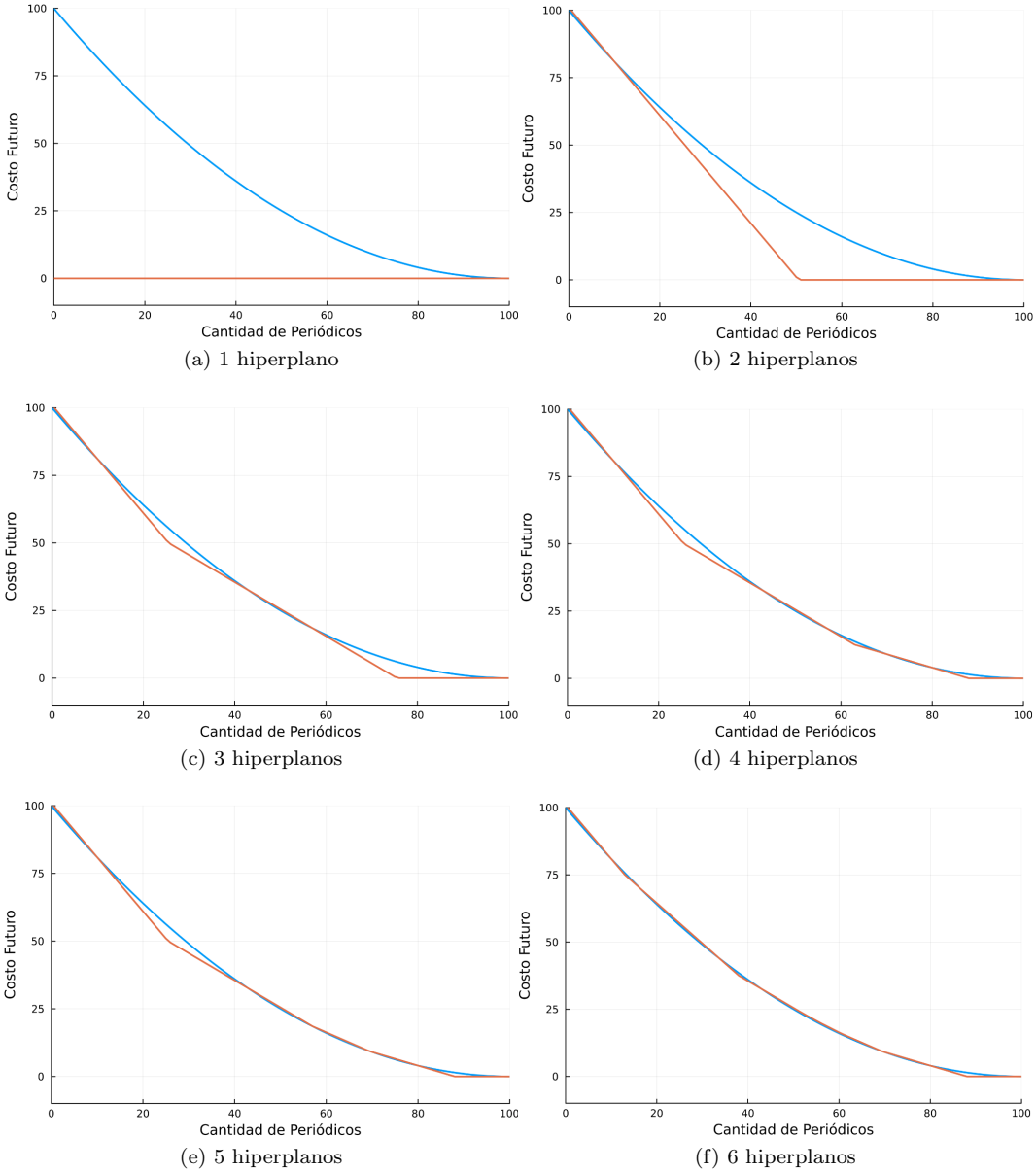
La etapa final, considera que el futuro no tiene costo y minimiza la esperanza sobre la variable aleatoria w_2, asegurando el abastecimiento de los periódicos.

$$\begin{aligned} &\text{mín } E[p_3 u_3] \\ &s.t. \\ &x_2 + u_3 \geqslant w_2. \end{aligned} \tag{2.6}$$

De esta manera, queda definido el esquema de resolución a través de tres etapas del problema 2.3 mediante SDDP.

Este problema se implementó para validar el algoritmo SDDP, utilizando el lenguaje *Julia*. El Algoritmo 1 se aplica a este problema teniendo en cuenta que cuando se resuelve el problema propio de cada etapa. Se aplica el esquema 2.4 al resolver la etapa 1 (previo al primer día), luego el esquema 2.5 al resolver la etapa 2 (previo al segundo día) y el esquema 2.6 como etapa final. En la pasada hacia atrás se incorporan hiperplanos que enriquecen las aproximaciones futuras θ_1 y θ_2, y en la pasada hacia adelante se recorre una nueva trayectoria.

En la Figura 2.1, se observan para un problema de *newsvendor* determinado una serie de gráficos que muestran en celeste la función de costo futuro para la etapa 1 y en naranja la aproximación por hiperplanos incremental a través de las iteraciones del algoritmo SDDP. Aunque estas figuras son sólo con fines ilustrativos, se ve claramente que luego de la aparición del sexto hiperplano activo en la función de costo futuro, la aproximación obtenida es muy buena.



(a) 1 hiperplano

(b) 2 hiperplanos

(c) 3 hiperplanos

(d) 4 hiperplanos

(e) 5 hiperplanos

(f) 6 hiperplanos

Figura 2.1: Mejora en la aproximación de la función de valor a través de las iteraciones del algoritmo

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Incorporación de un modelo Markoviano de aportes y de múltiples escalas de tiempo.

En este capítulo se presenta un modelo Markoviano para considerar los aportes hidrológicos al sistema uruguayo. Se describe la forma de estimación del modelo y su posterior incorporación en el algoritmo SDDP. A su vez, se presenta un abordaje para la resolución del problema de despacho económico de energía eléctrica que considera múltiples escalas de tiempo y utiliza SDDP. Este enfoque preserva variables de estado para tener en cuenta la dinámica de paso de tiempo lento (más allá de un día) y considera balances horarios dentro del paso de tiempo del algoritmo SDDP mediante restricciones lineales adicionales. Esto implica la resolución de problemas lineales de mayor complejidad, pero dentro de un ámbito que sigue siendo computacionalmente tratable.

3.1. Modelo Markoviano

En la discusión del algoritmo SDDP no se analizó el hecho de que las variables aleatorias también pueden estar correlacionadas entre diferentes pasos de tiempo, hecho que resulta claro en el caso de los aportes hidrológicos. De hecho, desde que se tiene registro de los aportes del Río Negro han ocurridos varias sequías largas [1], incluso alguna de una duración mayor a dos años. Considerando que a la hora de resolver el problema de operación del sistema eléctrico es necesario trabajar con pasos de tiempo cortos (semanas, días o incluso horas), está claro que de alguna manera hay que capturar la información del estado hidrológico del sistema a través de algún modelo adecuado para manejar los aportes. Una forma estándar es considerar que existe una variable hidrológica que se puede modelar a través de una cadena de Markov [5]. Por lo tanto, una representación útil para modelar esta dependencia temporal a través de una cadena de Markov es considerar un estado de Markov h_k que resuma la situación hidrológica actual. La dinámica de este estado está gobernada por una cadena de Markov no homogénea con probabilidades de transición:

$$P_{hh'}^{(k)} = P(h_{k+1} = h' \mid h_k = h)$$

Las probabilidades de transición pueden estimarse a partir de datos históricos.

En [33] se introduce un modelo con dos estados (seco y húmedo); la práctica local en Uruguay [11], es usar un modelo de 5 niveles del estado hidrológico como se describe en la Sección 3.1.1. Dado que el estado hidrológico es $h_k = h$, los posibles aportes hidráulicos deben ser muestreados de acuerdo con alguna distribución condicional $w_k \mid h$.

3.1.1. Estimación del modelo a partir de datos históricos

En Uruguay existen datos históricos de caudal de aportes semanal para los dos ríos en donde se construyeron represas hidroeléctricas, el Río Uruguay y el Río Negro. En la Figura 3.1 se observa la localización geográfica de las 4 centrales hidráulicas que existen en Uruguay. Tres se encuentran en el Río Negro: Rincón del Bonete, Baygorria y Palmar y en el Río Uruguay la central de Salto Grande.

El registro relevante a tener en cuenta para la estimación del modelo de Markov es el aporte incremental en cada una de las centrales. Debido a que no hay cauces hídricos significativos que generen aportes incrementales en la central de Baygorria, los registros semanales existentes son ternas de caudales: un valor para Rincón del Bonete, un valor para Palmar y un valor para Salto Grande. La correlación entre los aportes de Rincón del Bonete y Palmar resulta obvia porque pertenecen al mismo Río. Al ser Uruguay un país pequeño, es claro que los aportes de Salto Grande no son independientes de los del Río Negro. Dadas estas consideraciones resulta necesario contemplar esta correlación en el modelo a estimar.

La idea es estimar una cadena de Markov en la que una sola variable de estado represente la hidrología. Sin embargo, esta cadena no es homogénea si se va a estimar en pasos de tiempo relativamente cortos. La práctica local utiliza un paso de tiempo semanal, coherente con las medidas históricas. Considerando que se tienen T años de datos y cada año tiene 52 semanas, para estimar el modelo se procede de la siguiente forma.

- Para mantener la variable hidrológica simple, se reduce la serie histórica multivariada a una serie univariada. Para ello se ponderan los 3 aportes de cada una de las $T \times 52$ semanas en proporción a los coeficientes energéticos de las centrales correspondientes. Sea $W_{y,l}$ este escalar correspondiente a la semana l del año y, con $l \in 1 \dots 52$.
- Se toman todos los valores escalares $W_{y,l}$ y se particiona en 5 cuantiles, definiendo el quintil más bajo como condición *muy seca* y el quintil más alto como *muy húmeda*, esto se hace para una semana l fija.
- Se repite el procedimiento anterior para todas las semanas del año, por lo que se obtiene una serie histórica semanal con un valor que se toma como estado discreto $h_l = 0 \dots 4$.
- Se procede a estimar una matriz de transición para cada uno de las 52 semanas del año. Para estimar la matriz de transición de la semana l, se procede, como es usual, a contar cada una de las transiciones de estado: la entrada $P^{(l)}_{hh'}$ de la matriz se calcula contando dentro de la serie histórica cuántas transiciones se produjeron entre el estado h en la semana l y el estado h' en la semana $l+1$, para luego dividir ese valor entre el total de visitas al estado h en la semana l.

Con este procedimiento se obtiene una matriz de transición $P^{(l)}_{hh'}$ para cada semana del año. En caso de ser necesario un paso de tiempo del algoritmo más corto que una semana, se considera la matriz $P^{(k)}_{hh'}$ de forma que si k coincide con el inicio de la semana l, la matriz $P^{(k)}_{hh'} = P^{(l)}_{hh'}$, de lo contrario $P^{(k)}_{hh'}$ es la matriz identidad.

Figura 3.1: Localización de las centrales hidroeléctricas en Uruguay.

3.1.2. Incorporación del modelo Markoviano de aportes en el esquema SDDP

La incorporación de las transiciones de Markov en la programación dinámica conduce a la ampliación del estado existente propio del problema a resolver (x_k) para obtener un nuevo estado $\tilde{x}_k = (x_k, h_k)$ y una iteración de Bellman generalizando la definida en el Capítulo 2 que se puede descomponer de la siguiente manera:

$$\hat{V}_k((x,h),w) = \min_u \left\{ g_k(x,u,w) + \sum_{h'} P_{hh'}^{(k)} V_{k+1}(x',h') \right\}$$

$$V_k(x,h) = E\left[\hat{V}_k((x,h),w) \mid h_k = h \right],$$

donde $x' = f_k(x, u, w)$ representa el siguiente estado (estado al final del paso actual). La primera ecuación calcula una estimación del costo final para cada realización de las variables aleatorias (w debe ser coherente con h) y la segunda ecuación corresponde al promedio de estas realizaciones, que se calcula a partir de la distribución condicional para el estado hidrológico actual h.

Es importante tener en cuenta a la hora de resolver el problema, que, para reproducir la distribución condicional mencionada en el párrafo anterior, se toma una serie de ternas de aportes dada la semana del año l y el valor del estado $h = e$ para esa semana. Luego, se toma el conjunto de ternas de la historia de todas las semanas l que fueron clasificadas con el estado $h = e$ y se sortea de manera uniforme y con reposición entre dichas ternas.

De la misma forma que en el caso base, la misma técnica de SDDP se puede aplicar sustituyendo la aproximación corriente de la función de valor (lineal a tramos) y resolviendo:

$$\hat{\beta}_k^{(l+1)}(w) = \min_{x,u} \left\{ g_k(x, u, w) + \sum_{h'} P_{hh'}^{(k)} V_{k+1}^{(l+1)}(x', h') \right\}, \\ s.t. \quad x = x_k^{(l)} \quad [\hat{\lambda}_k^{(l+1)}(w)]. \tag{3.1}$$

Una cota inferior mejorada para la función de valor $V_k(x, h)$ se puede obtener promediando como antes, pero utilizando la distribución condicional dado h:

$$\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w_k) \mid h_k = h], \\ \lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w_k) \mid h_k = h]. \tag{3.2}$$

Los coeficientes definen, al igual que en el caso anterior, un hiperplano soporte para la función de valor. En la práctica, esta distribución condicional no es conocida, pero puede estimarse empíricamente a partir de los datos históricos: la parte condicionante se convierte simplemente en un promedio de los escenarios que pertenecen al estado hidrológico actual h (p. ej., solo realizaciones húmedas asociadas a ese paso del año), en lugar de ser un promedio sobre conjunto completo.

3.2. Estructura del algoritmo SDDP, considerando el modelo markoviano

La estructura exterior del algoritmo, consiste en la realización de iteraciones hasta lograr la convergencia según un criterio de parada. En el algoritmo estándar, cada iteración consta de dos etapas: la pasada hacia adelante y la pasada hacia atrás.

Una pasada adelante del algoritmo se puede esquematizar en los siguientes pasos:

1. Se toma el estado inicial x_0, h_0 del sistema incluyendo el estado markoviano, se establece $i = 0$.
2. Se sortea una transición de Markov utilizando la matriz $P_{h_i h'}$ de transición definida en la sección anterior.
3. Se sortea una realización aleatoria $w_i \in W$ condicionada a el estado markoviano actual h_i.
4. Se resuelve el problema lineal asociado al paso i, utilizando el estado actual x_i y y la realización w_i. Este problema es el que se expresa en la Ec. 3.2, utilizando la función de valor V_{k+1} del paso siguiente aplicada a h' y a x' obtenido según la Ec. 2.2. x' es el estado no markoviano siguiente de acuerdo a la dinámica.

5. Se coloca al punto (x_i, h_i) en una lista L (trayectoria de la pasada hacia adelante) y se incrementa $i = i + 1$, y el nuevo $x_i = x'$.

6. Si $i = T$ se termina, de lo contrario se vuelve a 2).

En la pasada hacia atrás asociada a la pasada hacia adelante descrita, se toma la trayectoria almacenada en L comenzando por el último elemento (estado al final de la trayectoria). Se toma el elemento (x, h) y se calcula un hiperplano de acuerdo al procedimiento establecido en la Sección 3.1.2. Dicho hiperplano, se agrega como aproximación de la función de valor $V^{(l+1)}(x, h)$. De esta manera será considerado en la próxima resolución del problema lineal asociado al paso l, como muestra la Ecuación 3.2.

Se analiza a continuación la interpretación de cada componente del algoritmo descripto, aplicado al problema que se tratará en esta Tesis.

Si se supone un sistema hidrotérmico, con una sola central hidráulica y un modelo markoviano como el analizado para la representación de la situación hidrológica, el espacio de estados consiste en una única variable no markoviana x, que representa el volumen del embalse y una única variable markoviana discreta h que representa la hidrología.

En este caso, la pasada hacia adelante consiste en:

1. Comenzar en el estado inicial del embalse y la situación hidrológica actual.

2. Sortear una situación hidrológica para el paso siguiente h'.

3. A partir de h', sortear un aporte hidráulico a la central hidroeléctrica.

4. Resolver el problema de despacho óptimo utilizando la aproximación de la función de Bellman actual (al inicio es la función idénticamente nula).

5. Agregar x y h a la trayectoria actual.

6. Avanzar al siguiente paso de tiempo actualizando x de acuerdo a la dinámica del embalse y el resultado óptimo de 4). Si no hay más pasos de tiempo, terminar. De lo contrario volver a 1).

Una vez obtenida una trayectoria para el volumen del embalse y para la variable hidrológica, se procede a realizar la pasada hacia atrás:

1. Se toma el último estado de la trayectoria (x, h).

2. Para cada posible estado hidrológico h', se resuelven todos los problemas de despacho posibles asociados a cada realización de aportes w condicionada a h'.

3. Se realiza el promedio de los valores óptimos de todos los problemas de despacho $\beta_k^{(l+1)}$ y el promedio de todos los vectores duales asociados a la variables de estado x, $\lambda_k^{(l+1)}$. De esa forma se obtiene un nuevo hiperplano que se incorpora a la aproximación futura actual $V_k^{(l+1)}$.

4. Se remueve el estado visitado (x, h) de la trayectoria y se vuelve a 1).

Se completa así una iteración del algoritmo SDDP. Se realizan tantas iteraciones como sea necesario para lograr convergencia. Un posible criterio de parada es el de verificar que la aproximación actual del costo total no varía más que un valor ϵ a lo largo de una cantidad m de iteraciones. Este criterio, conocido en inglés como *Bound Stalling*, es el que se utiliza a lo largo de este trabajo.

3.3. Descomposición en varias escalas de tiempo

En la actualidad, una cantidad considerable de energía despachada en los sistemas eléctricos proviene de fuentes de energía renovables como la eólica y la solar, que poseen una alta variabilidad en una escala de tiempo *más rápida* que el recurso hidráulico (aportes a las cuencas). Por lo tanto, es necesario incluir en la programación hidrotérmica general esta escala de tiempo refinada para modelar variaciones rápidas y almacenamiento a corto plazo (baterías). A continuación, se explica cómo introducir de forma genérica esta descomposición de múltiples escalas de tiempo en la SDDP en un modelo simple para transmitir las ideas principales. En el Capítulo 4 se describe el detalle de la implementación realizada de cada recurso de escala rápida para un sistema real.

Para modelar la escala de tiempo rápida, se considera el estado x_k en el paso de tiempo k compuesto. La ecuación básica de la dinámica en caso de los embalses es:

$$x_{k+1} = x_k - u_k - s_k + w_k, \tag{3.3}$$

donde u_k es la cantidad de agua turbinada, s_k es la cantidad de agua vertida y w_k es la cantidad de agua recibida en el embalse (aleatoria) por los aportes hidrológicos.

Se propone entonces descomponer u_k en una escala de tiempo más fina, esto es:

$$u_k = \sum_{i=0}^{t-1} u_{ik},$$

Por otra parte, se tienen una serie de recursos que condensaremos genéricamente en la variable p_k, para considerar múltiples ecuaciones de balance de potencia dentro de un mismo paso, es necesario considerar un paso de tiempo en la escala más fina, por lo que las ecuaciones resultantes están dadas para cada k por:

$$d_{ik} = \nu u_{ik} + p_{ik} \quad i = 0, \dots, t-1. \tag{3.4}$$

Aquí ν es un coeficiente de rendimiento del generador hidroeléctrico (por simplicidad se supone independiente del nivel del embalse) y p_{ik} representa genéricamente la potencia de cualquier recurso no considerado. Asumiendo que el costo de operación es c_k asociado a la generación no hidráulica, la función de costo presente en el paso k es:

$$g_k(x_k, (u_{ik}, s_k, p_{ik})) = \sum_{i=0}^{t-1} c_k p_{ik}.$$

Si bien esta función no depende explícitamente del nivel del embalse (o almacenamiento) ni de la energía turbinada o vertida, lo hace indirectamente a través de la restricción (4.2) ya que se debe satisfacer la demanda eléctrica.

Este modelo más detallado aumenta el costo computacional: en lugar de tener una sola variable de control $u_k \in \mathbb{R}$ para el uso del agua, se tiene: $(u_{ik}, i = 1, \dots, t) \in \mathbb{R}^t$. Lo mismo ocurre con las variables de potencia. En el Capítulo 4 se analiza el impacto en los tiempos de cómputo al incorporar el modelado detallado descrito en la presente sección, así como también una batería con escala rápida que se describe en el mismo.

Se presenta de forma genérica un seudocódigo del Algoritmo 2, estándar SDDP, incorporando el modelo de Markov y las múltiples escalas de tiempo. Este algoritmo será utilizado en el Capítulo 4 para la evaluación realizada.

Existen diversas variantes estructurales, en las que se realizan dos o más pasadas hacia adelante o hacia atrás en cada iteración. Muchas de estas variantes resultan más adecuadas para esquemas de computación paralela, y está probada su equivalencia con la forma estándar presentada [18].

Algorithm 2 Algoritmo SDDP completo

$K = 0$
$L = []$
$i = 0$ ▷ Paso de tiempo, el último es T
while $K \leq Max_{iter}$ **do** ▷ Pasada hacia adelante
  $x = x_i$
  $h = h_i$ ▷ utilizando la matriz de transición
  while $i < T$ **do**
    obtener realización $w_i \in W$
    resolverProblemaPaso(x_i, h_i, w_i) ▷ se obtiene x'_i según la dinámica
    agregar$((x_i, h_i), L)$ ▷ se agrega el estado a la trayectoria
    $x = x'_i$
    $h = h'_i$ ▷ utilizando la matriz de transición
    $i = i + 1$
  end while
  $L_{inv} = invertirLista(L)$ ▷ Comienza pasada hacia atrás
  $i = T - 1$
  for $(x, h) \in L_{inv}$ **do**
    for $h' \in H$ **do** ▷ estado markoviano siguiente
      for $w_i \in W$ **do** ▷ realización condicionada el estado h'
        $\beta_k^{(l+1)}, \lambda_k^{(l+1)} = resolverProblemaPaso(x_i, h_i, w_i)$
        ▷ el óptimo y variables duales asociadas al estado
      end for
    end for
    $\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w_k) \mid h_k = h]$
    $\lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w_k) \mid h_k = h]$
    $actualizar(V, i, \beta_k^{(l+1)}, \lambda_k^{(l+1)})$
  ▷ Se agrega el nuevo corte a la aproximación de Bellman del paso i, para el estado h.
  end for
  $K = K + 1$
end while

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Modelo del despacho uruguayo mediante SDDP

Se aplican en este capítulo las ideas descritas en los Capítulos 2 y 3, en el sistema Uruguayo. Como el factor climático es de gran relevancia en el sistema eléctrico uruguayo, se estima un proceso estocástico de Markov con una variable de estado utilizando los datos históricos disponibles de aportes hidrológicos en Uruguay tal como se presentó en el Capítulo 3. La variable de estado markoviana, en conjunto con las que representan los volúmenes de los lagos son consideradas en el algoritmo SDDP. Por otra parte, debido a la existencia de recursos que operan a escalas rápidas de tiempo, en conjunto con el recurso hidráulico que opera a escala lenta, se consideran las ideas expuestas en el Capítulo 3 respecto de las múltiples escalas de tiempo. Inicialmente se implementa un modelo básico que no utiliza dos escalas de tiempo y su estructura es muy simple. Luego se mejora dicho modelo incorporando dos escalas de tiempo de acuerdo a lo descrito en el Capítulo 2 y agregando diversos recursos. Finalmente, se considera el modelo completo con dos escalas de tiempo incluyendo un almacenamiento de corto plazo (batería). Además, se incorpora una variable de estado que representa la energía almacenada en la batería para enriquecer el espacio de estados y analizar el impacto tanto en el costo de operación como en los tiempos de cómputo. En la sección 4.2 se presentan los resultados experimentales.

4.1. Modelos considerados

Modelo básico Se considera el modelo más simple de operación de un sistema hidrotérmico sobre un horizonte T, con pasos de tiempo indexados por $k = 0, \dots T-1$. El vector de estados $x_k \in \mathbb{R}^n$ del sistema representa el volumen actual de los n lagos de las represas hidroeléctricas; se consideran los aportes hidrológicos de acuerdo al modelo markoviano presentado en el Capítulo 3. El vector de control $u_k \in \mathbb{R}^m$ modela las acciones que puede tomar el operador del sistema: la generación eléctrica de las centrales hidroeléctricas (a partir del agua turbinada), la cantidad de agua vertida y las decisiones económicas de despacho de los generadores térmicos para abastecer la demanda en cada etapa. En este modelo no se utilizan múltiples escalas de tiempo. A su vez, no se consideran inicialmente otros aspectos del problema, como las energías renovables no gestionables, almacenamientos de corto plazo o los intercambios de energía con los sistemas vecinos.

Modelo incorporando varias escalas de tiempo En la actualidad una cantidad considerable de energía despachada en los sistemas eléctricos proviene de fuentes de energía renovables como la eólica y la solar, que poseen una alta variabilidad en una escala de tiempo *más rápida* que el recurso hidráulico (aportes a las cuencas). Por lo tanto, es necesario incluir en la programación hidrotérmica general esta escala de tiempo refinada para modelar variaciones rápidas y almacenamiento a corto plazo (baterías). A continuación, se explica cómo introducir esta descomposición de múltiples escalas de tiempo en la SDDP al modelo básico. Por razones de claridad se vuelven a presentar algunos resultados del Capítulo 3, aplicados al caso específico del sistema uruguayo.

Para modelar la escala de tiempo rápida, se considera el estado x_k en el paso de tiempo k compuesto por el nivel corriente de los embalses y los recursos de almacenamiento del sistema (p.ej. baterías, centrales de bombeo, etc.). La ecuación básica de la dinámica para cada uno de los embalses es:

$$x_{k+1} = x_k - u_k - s_k + w_k, \tag{4.1}$$

donde u_k es la cantidad de agua turbinada, s_k es la cantidad de agua vertida y w_k es la cantidad de agua recibida en el embalse (aleatoria) por los aportes hidrológicos.

Se propone entonces descomponer u_k en una escala de tiempo más fina, esto es:

$$u_k = \sum_{i=0}^{t-1} u_{ik},$$

donde u_{ik} es el agua utilizada para la generación en cada paso de la escala de tiempo detallada (por ejemplo, horas en lugar de días o semanas).

Si d_{ik} es la demanda en la escala de tiempo más fina y e_{ik} es la energía disponible de las energías renovables, las ecuaciones de balance de potencia están dadas para cada k por:

$$d_{ik} = e_{ik} + \nu u_{ik} + p_{ik} \quad i = 0, \dots, t-1. \tag{4.2}$$

Para incorporar un almacenamiento de corto plazo, que incluya una variable de estado y además el balance de energía en la escalar rápida se modifica la ecuación (4.2), obteniéndose la ecuación 4.3:

$$d_{ik} + p_{ik}^{exp} + q_{ik}^{c} = e_{ik} + \sum_{j} \nu_j u_{ik}^{j} + p_{ik} + p_{ik}^{imp} + q_{ik}^{d}, \tag{4.3}$$

para $i = 0, \dots, t-1$, $k = 0, \dots, T-1$. La dinámica de la batería por hora es capturada por la siguiente Ecuación (4.1).

$$b_{i+1,k} = b_{ik} + \eta_c q_{ik}^{c} - \frac{1}{\eta_d} q_{ik}^{d}, \quad i = 0, \dots, t-1,$$

identificando $b_{0k} = b_k$ and $b_{tk} = b_{k+1}$. En las ecuaciones se consideran para la batería rendimientos de carga y de descarga (η_c y η_d).

Esto completa la definición del modelo, en la siguiente sección se discuten los resultados de la optimización.

4.2. Implementación y simulaciones

Se resumen a continuación los resultados obtenidos al aplicar la técnica SDDP al modelo estilizado del sistema uruguayo que se desarrolló. El análisis presentado en esta sección se enfoca las consideraciones de *tiempos de cómputo* y no explícitamente en las consideraciones de costos y energías halladas en la optimización. El objetivo

central es cuantificar la complejidad y el tiempo de cálculo requerido al incorporar la escala de tiempo detallada, así como un mayor número de variables de estado. Los costos presentados a continuación por lo tanto no pretenden reflejar los costos reales de operación del sistema uruguayo.

Todos los modelos descriptos en 4.1 se implementaron en Julia [10], un lenguaje de modelado matemático muy eficiente de código abierto. En particular, para modelar el problema propuesto se utilizó la biblioteca de SDDP implementado por el paquete SDDP.jl [16].

A continuación, se analizan tres variantes de modelado: la primera (M1) constituye un modelo de referencia para la evaluación comparativa con las otras dos variantes. Incorpora solo la escala de tiempo más lenta (diaria) y no incluye el almacenamiento a corto plazo, por lo que su estado incluye solamente los niveles de los embalses, y el balance energético se plantea a escala diaria. Este modelo por tanto supone que la energía renovable se puede utilizar por completo en el paso, una suposición que puede no ser válida en la realidad debido a la incapacidad de despachar estas fuentes en una hora determinada (excedentes renovables).

En la segunda variante (M2) se incorpora la escala de tiempo más rápida (horaria) al modelo base, pero no se incluye el almacenamiento de corto plazo. Esta segunda variante incluye todas las limitaciones de potencia del corto plazo y, en particular, refleja que las fuentes renovables no pueden ser despachadas a voluntad. Estas dos variantes sin almacenamiento comparten un espacio de estado de cuatro estados continuos (los embalses) y un estado hidrológico de Markov discreto. Sin embargo, el modelo M2 incorpora un mayor espacio de variables de control dentro de la escala de tiempo detallada. Esto nos permite evaluar el impacto en el tiempo de ejecución de incorporar el despacho detallado sin ampliar el espacio de estados.

La tercera variante (M3) que se consideró agrega un almacenamiento a corto plazo (batería) al modelo M2. En este caso, se agrega una variable de estado para representar el nivel de almacenamiento en la escala de tiempo lenta. Por lo tanto, también se está ampliando el espacio de estados con respecto a M2. La idea principal en esta variante es comparar con las técnicas clásicas de programación dinámica estocástica (SDP) (por ejemplo, [11]). En SDP, agregar una variable de estado requiere de la discretización de ese estado adicional. La recursión SDP requiere la enumeración de todas las combinaciones de valores de estado inicial de cada paso para todos los estados. Si la variable añadida se cuantifica en l niveles, entonces el esfuerzo computacional se incrementa al menos por un factor de l. En SDDP, el impacto de agregar una variable de estado no produce tal explosión en el tiempo de cómputo con el aumento de la dimensionalidad, porque los estados no se tratan a través de una discretización, sino que se utilizan hiperplanos para una aproximación global de la función de costo futuro. El impacto real en el tiempo de ejecución está determinado por el número total de iteraciones requeridas por SDDP para alcanzar la convergencia debido al espacio de estados más rico que se debe explorar. Sin embargo, este incremento en el número de iteraciones sigue siendo menos costoso que el enfoque de discretización. Los casos de estudio planteados que se acaban de describir tienen como objetivo validar este comportamiento.

4.2.1. Datos de entrada de los modelos

Todos los modelos se alimentan con los siguientes parámetros de entrada: la demanda agregada del país, así como las generaciones de energía solar y eólica son las tomadas de las correspondientes al año 2017 [3]. Los costos de generación, así como el precio de los intercambios de energía se eligen de manera que sean similares a los costos medios del sistema uruguayo actual a fin de lograr una aproximación razonable

Tabla 4.1: Estadísticas de demanda y energía renovable y costos de generación

Potencias medias y energías anuales de demanda renovables	
Potencia media de demanda	1334 MW
Potencia media renovable	570.4 MW
Energía anual demandada	11654 GWh
Energía anual renovable	4984 GWh
Costos	
Térmico	200 USD/MWh
Exportación	15 USD/MWh
Importación	360 USD/MWh

Tabla 4.2: Parámetros para el modelo M3 del almacenamiento de corto plazo

Parámetros del almacenamiento	
$\underline{B}$	0 MWh
$\overline{B}$	500 MWh
$\bar{Q}^d$	250 MW
$\bar{Q}^c$	250 MW
$\eta_c = \eta_d$	0.98

de los costos totales de operación. Sin embargo, no se debe esperar que estos costos reflejen el costo de operación del sistema real. En la Tabla 4.1 se presenta un resumen de algunos de estos datos de entrada.

Adicionalmente, para el modelo M3 utilizamos los parámetros para el almacenamiento a corto plazo resumidos en la Tabla 4.2.

Como criterio de convergencia para el algoritmo SDDP, se eligió la regla de parada *Bound Stalling* en los tres modelos. Según esta regla, el algoritmo SDDP terminará después de que el límite inferior del costo no haya mejorado durante un número determinado de iteraciones (establecido en 5 en el caso de este trabajo). Después de la convergencia, la política obtenida se simula mediante el método de Monte Carlo para 1000 escenarios de datos de entrada aleatorios. El costo de operación medio simulado se calcula promediando el costo de todos los escenarios. Todos los cálculos que se presentan a continuación se realizaron en una computadora Intel Core i7 de 3,40 GHz y 32 GB de RAM.

4.2.2. Resultados experimentales

Los resultados para cada una de las tres variantes descritas se presentan en la tabla 4.3. La columna de iteraciones muestra cuántos pasos toma el algoritmo SDDP para lograr convergencia, y el tiempo de ejecución se informa en la columna correspondiente. Finalmente, se presenta la cota inferior del costo alcanzado, así como el costo medio obtenido luego de simular la política hallada de acuerdo a lo descrito.

Interpretando los resultados, viendo la Tabla 4.1 se sabe que la energía anual que se debe inyectar al sistema, suponiendo que se pueda usar toda la energía renovable,

Tabla 4.3: Resultados Experimentales

Modelo	**Iteraciones**	**Tiempo de ejecución (s)**	**Cota inferior del costo (MUSD)**	**Costo simulado promedio (MUSD)**
M1	29	318	52.25	54.91
M2	71	1611	79.33	85.79
M3	99	3127	73.73	78.58

es de aproximadamente $6670GWh$. Del costo obtenido podemos inferir que la política está cubriendo esta energía con aproximadamente 96 % de generación hidráulica y 4 % de generación térmica, similar a la situación actual de Uruguay.

Comparando los tres modelos, el primero obtiene un menor costo, lo cual es esperable ya que solo considera el balance energético diario y esto implica que se puede aprovechar toda la energía renovable disponible. En términos matemáticos, el modelo M1 es una relajación de M2, por lo que siempre alcanzará un costo menor. El modelo M2 tiene en cuenta la escala de tiempo detallada y, por lo tanto, el hecho de que la demanda y la generación renovable pueden no estar alineadas durante el día. El aumento en el costo se debe al hecho de que ahora no se puede utilizar la energía renovable de forma indistinta en el día, por lo tanto, se debe introducir energía térmica adicional en el sistema en determinadas horas. Parte de este costo lo recupera el modelo M3, que introduce la batería de almacenamiento a corto plazo, lo que ahora permite que el sistema aproveche mejor la energía renovable.

Más importante aún, se discute a continuación el tiempo de convergencia y el rendimiento de nuestra implementación. El modelo de referencia M1 converge en solo 29 iteraciones y 318 segundos, a pesar de incluir los cuatro embalses del estado, así como la variable markoviana que representa el estado hidrológico. Esto ilustra que el modelo SDDP es adecuado para el problema abordado, ya que nos permite modelar el sistema completo con una escala de tiempo diaria y obtener resultados rápidos. El segundo modelo incorpora el paso de tiempo detallado, multiplicando por 24 el número de variables de control. Sin embargo, esto no conduce a un gran aumento en el tiempo de cálculo. El verdadero poder de la técnica SDDP se hace evidente en el tercer modelo, donde se introduce un nuevo estado y sus variables de control asociadas, pero el tiempo de cálculo sólo es aproximadamente el doble. Hay que tener en cuenta que si se abordara este problema utilizando SDP, el tiempo de ejecución del modelo M3 sería aproximadamente el del modelo M2 multiplicado por la cantidad de discretizaciones de la variable de estado que representa la batería. Suponiendo que esta discretización fuera sólo de 5 puntos, el tiempo de ejecución se multiplicaría por 5 en SDP mientras que en SDDP apenas se duplicó. El rendimiento logrado muestra que SDDP constituye un enfoque prometedor para modelar el sistema utilizando la descomposición de múltiples escalas de tiempo que se desarrolló.

4.3. Conclusiones de este análisis

En este capítulo se propuso una adaptación de la técnica de programación dinámica estocástica dual para resolver el problema de despacho óptimo en un sistema hidrotérmico. La principal contribución fue incorporar en el marco de SDDP una escala

de tiempo detallada, lo que permite modelar los efectos a corto plazo, lo que se hace necesario para capturar la variabilidad de las fuentes renovables y medir mejor el impacto de incorporar el almacenamiento de energía a corto plazo en el sistema. Los resultados obtenidos muestran tiempos de cómputo prometedores, particularmente cuando se amplía el espacio de estados, lo que abre la puerta a incorporar más detalles al modelo. Sin embargo, en un sistema con una cantidad de variables de estado acotada, el tiempo de cómputo puede que se vea afectado por la maldición de la dimensionalidad estocástica. Esto implica buscar algún mecanismo para resolver menor cantidad de problemas lineales y la biblioteca de Julia utilizada no tiene la suficiente flexibilidad para abordar este desafío. Por esta razón, en el Capítulo 5 se plantea el desarrollo de una biblioteca en Python específica para la aplicación de SDDP sobre el sistema eléctrico que por ser de desarrollo propio brinda la posibilidad de explorar soluciones que mitiguen la maldición de la dimensionalidad. Estas soluciones son abordadas en el Capítulo 6.

Capítulo 5

Desarrollo de una biblioteca SDDP en Python adaptada al sistema energético

Dentro de los objetivos del presente trabajo de tesis está la necesidad de mantener tiempos de cómputo razonables al incorporar el modelado de diversos recursos energéticos a un sistema energético como el uruguayo. En el capítulo 4 se analizó la aplicación del algoritmo SDDP con el objetivo de mitigar la maldición de la dimensionalidad de Bellman (maldición del espacio de estados). Sin embargo, no es suficiente para mantener los tiempos de cómputo dentro de lo razonable mitigar la maldición de la dimensionalidad de estados, sino que también se hace necesario mitigar la maldición de la dimensionalidad estocástica. Esto se debe a la incertidumbre que incorporan las nuevas tecnologías en el sistema. En este capítulo se especifican las principales características de la biblioteca desarrollada.

5.1. Introducción

La biblioteca utilizada para el análisis realizado en el Capítulo 4 no permite abordar con facilidad ciertas técnicas a explorar en el marco del algoritmo SDDP aplicado al sistema eléctrico. Esto se debe a que está construida sobre una estructura poco flexible, lo que no permite modificar con facilidad el código fuente vinculado con el tratamiento estocástico que se realiza en el algoritmo estándar. A pesar de que hay disponible alguna biblioteca alternativa a la utilizada [37], se desarrolló en Python una biblioteca específica para aplicar el algoritmo SDDP en el problema de operación de un sistema eléctrico similar al uruguayo. Se decidió realizar este desarrollo por si alguna biblioteca alternativa no presentaba la flexibilidad deseada. La biblioteca desarrollada permite tener la suficiente flexibilidad para incorporar mejoras del algoritmo SDDP estándar. En el presente capítulo se describen cualitativamente las posibilidades que brinda la biblioteca desarrollada para implementar SDDP estándar. Además, se describen las funcionalidades específicas incorporadas propias del problema energético a resolver. En el Apéndice A, se especifican desde el punto de vista informático las funcionalidades con las que cuenta la biblioteca, y se presenta un ejemplo de su uso.

En el Capítulo 6, se realiza el análisis de las alternativas investigadas para mitigar la maldición de la dimensionalidad estocástica utilizando la biblioteca desarrollada y

sus funcionalidades adicionales.

5.2. Componentes de la biblioteca

La biblioteca desarrollada permite definir un problema de operación del sistema teniendo en cuenta cuatro aspectos:

- Parámetros relevantes del algoritmo SDDP.
- Parámetros generales vinculados al problema de operación.
- Definición de los componentes del parque de generación.
- Definición de los componentes que definen los procesos estocásticos que modelan la incertidumbre.

En las siguientes secciones del presente capítulo se describen cada uno de los puntos anteriores. Para ello se pone énfasis en lo cualitativo, describiendo en cada caso qué simplificaciones se asumen en el modelo propuesto. La biblioteca resuelve el problema asociado a un despacho en un paso de tiempo utilizando programación lineal, para lo que utiliza un resolvedor lineal externo. La biblioteca está estructurada basada en el paradigma de orientación a objetos. Es por esto que, cualquier incorporación que sea necesaria, es posible desarrollarla con un esfuerzo acotado en la biblioteca siempre que se haga en el marco de la estructura planteada (ver Apéndice A).

5.2.1. Algoritmo base

Tal como se estableció en el Capítulo 2, el algoritmo estándar cuenta con una serie de iteraciones. En cada iteración se obtiene una mejor aproximación de la función de costo futuro en cada paso. A través de un criterio de parada se decide en qué momento se deja de iterar tomando el conjunto de funciones de costo futuro calculados hasta ese momento, lo que resulta equivalente a tener una política de operación para ese criterio de parada. Una vez obtenida dicha política a través de las funciones de costo futuro, es posible aplicar el Método de Montecarlo y simular un conjunto de escenarios de realizaciones estocásticas utilizando esta política. Con este conjunto de simulaciones se obtiene una distribución de los costos del sistema y de las energías despachadas para cada uno de los recursos del parque.

Respecto del algoritmo base es posible definir a través de la biblioteca los siguientes parámetros:

- Parque de generación con el que se realiza la optimización
- Procesos estocásticos que utiliza para modelar la incertidumbre
- Definición del horizonte de tiempo (tiempo inicial y tiempo final)
- Definición de la duración de un paso de tiempo
- Criterio de parada del algoritmo

En la Sección 5.3 se agregan funcionalidades que requieren la configuración de más parámetros de entrada.

5.2.2. Componentes del parque del sistema eléctrico

Se describen a continuación los componentes del sistema eléctrico modelados, así como las simplificaciones que se hicieron en estos componentes. Es importante aclarar que no se considera la red eléctrica en el modelo, o lo que es equivalente, todo generador y demanda están conectados a un único nodo.

Generador hidráulico Para el generador hidráulico se utiliza un modelo considerando embalse. El embalse de cada generador hidráulico se modela mediante una variable de estado representando su volumen, que es incorporada a las variables de estado del algoritmo SDDP. Los parámetros que definen el generador hidráulico son:

- *Volumen inicial del lago*: representa el volumen del lago al inicio del horizonte de resolución del problema.
- *Volumen mínimo y volumen máximo del lago*: representan la cota mínima y la cota máxima de la variable de estado asociada al volumen del lago.
- *Caudal turbinado máximo*: es la capacidad máxima en caudal que tiene la central para turbinar.
- *Caudal vertido máximo*: es la capacidad máxima en caudal que tiene la central para verter.
- *Coeficiente energético de la central*: representa un coeficiente energético constante a lo largo de toda la corrida
- *Centrales aguas arriba*: son las centrales que se encuentran conectadas por un curso de agua, aguas arriba de la central que se está definiendo.
- *Centrales aguas abajo*: son las centrales que se encuentran conectadas por un curso de agua, aguas abajo de la central que se está definiendo.
- *Aportes*: representa la variable aleatoria asociada al proceso estocástico encargado de sortear dicha variable contemplando la distribución de probabilidad apropiada.

Cabe aclarar que este modelo de generador hidráulico es extremadamente simple ya que considera el coeficiente energético constante. Por ejemplo, la relación que existe entre la potencia eléctrica que entrega la central el caudal erogado por la misma, es una relación no lineal que depende del salto de agua de la central. A su vez, el salto depende de la cota aguas arriba de la central y de la cota aguas abajo (cota del río). Sin embargo, la cota del río también depende de la siguiente central aguas abajo, si es que la hay. Otro ejemplo de complejidad de la central hidráulica tiene que ver con la amplitud del vertedero. Habitualmente no es posible verter de la misma manera dependiendo de la altura del lago. A su vez, en la operación de las centrales hidráulicas existen otro tipo de restricciones. Un ejemplo son las restricciones para el control de cotas, generalmente por temas de seguridad de la presa. Otro ejemplo es la incorporación de restricciones de índole ecológico, como ser el caso de forzar un erogado mínimo en la central para mantener el río fluyendo. En el trabajo de Diniz [15], se modela de forma muy detallada la producción de las centrales hidroeléctricas dependiendo del erogado y las características del río.

Generador térmico El generador térmico implementado en la biblioteca es muy simple. No posee un mínimo técnico y su costo de generación es proporcional a un precio que se incluye como parámetro. No aporta estado al algoritmo SDDP. Los parámetros que definen el generador térmico son:

- *Potencia máxima*: es la potencia máxima que puede despachar el generador térmico.
- *Precio de la energía*: es el precio por MWh de la energía generada

En este modelo de generador térmico no hay un aporte a las variables de estado del algoritmo. Sin embargo, hay un aporte a la función objetivo en términos de costo, que consiste en el costo asociado a la generación térmica. Básicamente, es el producto entre el precio de la energía y la energía generada.

Al igual que en el hidráulico, se podría considerar un modelo detallado de un generador térmico más complejo [27]. Por ejemplo, podría considerar diferentes turbinas, cada una con su disponibilidad asociada y un modelo de roturas. A su vez, se podrían considerar los mínimos técnicos utilizando variables enteras, ya que en la realidad una máquina térmica puede estar apagada o despachando por encima del mínimo técnico, pero no puede despachar una potencia menor que el mínimo técnico. Incluso si se tratara de una máquina térmica de tamaño considerable, es posible que su estado de disponibilidad sea necesario considerarlo como estado del algoritmo. Por ejemplo, si se tratara de una central atómica, que habitualmente posee una potencia muy grande, sería prudente considerar en el estado la disponibilidad de la misma, así como los tiempos de reparación en caso de indisponibilidad, a través de un conjunto de variables de estado.

Generador eólico y generador fotovoltaico Tanto el generador fotovoltaico como el generador eólico, se modelan a través de su factor de planta. Para ello se tienen una serie de escenarios de factor solar y eólico que están correlacionados, que surgen de los datos históricos. Para cada generador se define la potencia instalada y para obtener la potencia despachada simplemente se multiplica por el factor actual. Los parámetros que definen este tipo de generador son:

- *Potencia instalada*: es la potencia instalada del recurso en cuestión.
- *Factor*: es la variable aleatoria (que se encuentra en un proceso estocástico) que permite sortear factores de planta en cada instante manteniendo la distribución de probabilidad adecuada.

Nuevamente cabe aclarar que es posible definir modelos más complejos para el recurso eólico y el recurso solar fotovoltaico, como en el trabajo de Abid [4]. Por ejemplo, para el recurso eólico sería posible modelar todo un parque con sus diferentes aerogeneradores y considerar la velocidad del viento, el efecto de estelas entre generadores, entre otras cosas. Otro ejemplo para el caso del recurso solar, sería el de modelar la generación a partir de la radiación, que depende entre otros factores de la cobertura de nubes del cielo.

Intercambios internacionales de energía Los intercambios internacionales implementados en la biblioteca son muy simples. No posee un mínimo técnico y su costo (o beneficio) de intercambio es proporcional a un precio de la energía que se incluye como parámetro. Cabe aclarar que en el caso de la importación se considera un costo y en el caso de la exportación se considera un beneficio. No aportan estado al algoritmo SDDP. Los parámetros que definen a los intercambios de energía son:

- *Potencia máxima intercambiable*: es la potencia máxima a la que se puede intercambiar con el país vecino.
- *Tipo de transacción*: indica si se trata de una exportación o una importación.
- *Precio de la energía*: es el precio por MWh de la energía importada o exportada.

En el caso de los intercambios internacionales, sería posible incorporar un modelado más complejo de los mismos que a través de un proceso estocástico que permita considerar la potencia y los precios dependiendo de las situaciones de los países que intercambian. Por ejemplo, considerando los precios dependientes del costo marginal de la energía en cada uno de los países.

Demanda Cada demanda considerada en el parque, es modelada a través de un proceso estocástico que determina la potencia demandada.

- *Potencia*: es la variable aleatoria (que se encuentra en un proceso estocástico) que permite sortear realizaciones en cada instante de potencia demandada, manteniendo la distribución de probabilidad adecuada.

Todas las consideraciones descriptas como mejoras en el modelado de cada uno de los componentes del parque, no fueron incorporadas en este estudio ya que el objetivo principal es realizar un análisis de impactos en los tiempos de cómputo al aumentar el espacio de estados (maldición de la dimensionalidad de estado) o la complejidad estocástica (maldición de la dimensionalidad estocástica). Sin embargo, es importante tener en cuenta que de ser necesario incorporar este tipo de detalles, la biblioteca está estructurada cumpliendo los criterios básicos de escalabilidad de software, de manera que se puedan agregar modelos de alta complejidad sin mayor dificultad.

5.2.3. Procesos Estocásticos

La incertidumbre del problema de operación, se contempla a través de la construcción de procesos estocásticos. Cada proceso estocástico tiene un conjunto de variables aleatorias y es posible muestrear dichas variables con la distribución apropiada. A su vez, un proceso estocástico puede aportar estado al algoritmo SDDP. Por ejemplo, el proceso estocástico markoviano para los aportes hidrológicos (ver Capítulo 2) posee tres variables aleatorias, que representan los aportes a las centrales de Bonete, Palmar y Salto, respectivamente. Además, como fue descripto, el mismo posee una variable de estado hidrológica que es utilizada por el algoritmo SDDP.

Existe una clara relación de colaboración en términos de software entre el algoritmo base (SDDP en este caso), los componentes del parque y los procesos estocásticos. El algoritmo de SDDP se basa, como vimos en el Capítulo 2, en aproximar las funciones de Bellman para cada paso de tiempo mediante hiperplanos. Estas funciones tienen como dominio el espacio de estados, que lo aportan los componentes del parque y los procesos estocásticos. A su vez, cada componente del parque puede tener un dato con incertidumbre que se modela mediante una variable aleatoria, que debe sortearse a partir de un proceso estocástico. Eventualmente, existen variables aleatorias correlacionadas que pertenecen a diferentes componentes del parque pero que están incluidas en un único proceso estocástico, de forma de contemplar dicha correlación.

Un ejemplo que ilustra estas relaciones se diagrama en la Fig. 5.1. En la misma se puede ver un sistema con dos centrales hidráulicas en dos ríos diferentes, un generador eólico y un generador solar completando el parque. En este caso, existe un proceso estocástico que permite realizar aportes hidráulicos para ambas centrales, y a su vez aporta una variable de estado hidrológica (que se define a partir de la propia situación de aportes) al algoritmo base SDDP. Además, una de las centrales posee embalse, por lo que el volumen de dicho embalse es considerado como otra variable de estado para el algoritmo base. Por otra parte, tanto el generador eólico, como el generador solar tienen una variable aleatoria que representa su factor de planta. Estas variables aleatorias pertenecen a un mismo proceso estocástico que preserva la correlación temporal y geográfica entre los recursos solar y eólico. Ambos procesos estocásticos conforman el Azar. El algoritmo SDDP recibe, por lo tanto: al Azar, al Parque y el estado. Esta información en conjunto con la información paramétrica del algoritmo, permite resolver el problema de operación.

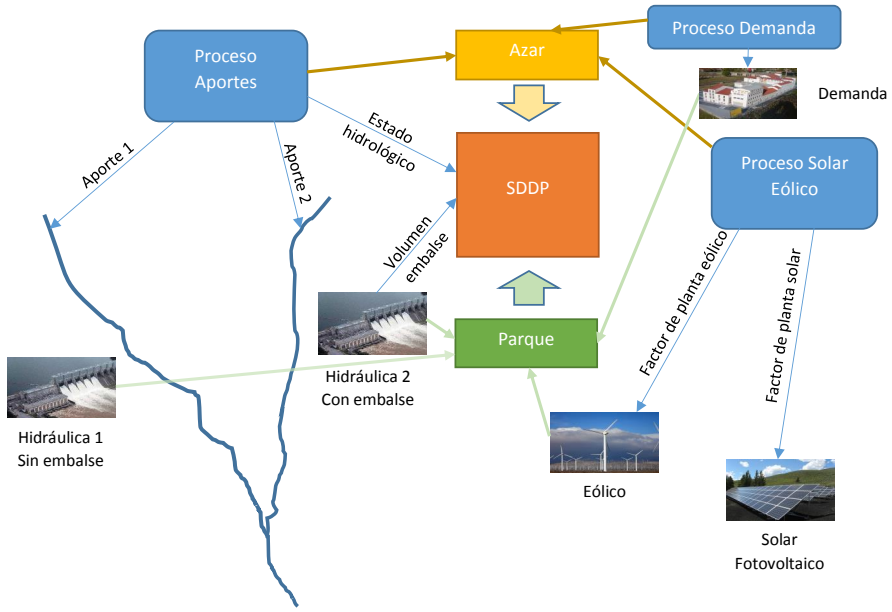


Figura 5.1: Relación entre los componentes que definen el problema de operación.

5.3. Funcionalidades incorporadas

La principal motivación para el desarrollo de esta biblioteca, fue que la biblioteca utilizada para el análisis realizado en el Capítulo 4 no permitía incorporar con facilidad ajustes al algoritmo SDDP buscando mejorar los tiempos de cómputo del algoritmo. A su vez, tampoco resultaba sencillo agregar funcionalidades generales. Se describen en esta sección una serie de ideas que surgieron para mejorar el algoritmo SDDP y su implementación.

5.3.1. Disminuir la cantidad de problemas lineales que se resuelven en la pasada hacia atrás

En la pasada hacia atrás del algoritmo de SDDP, ocurre que para cada paso de resolución se resuelve un problema lineal con cada realización existente de la aleatoriedad. Una condición necesaria para la convergencia del algoritmo SDDP, es que en cada pasada hacia atrás la distribución conjunta de las variables aleatorias involucradas debe estar dada por una distribución discreta estática durante todo el proceso algorítmico. Esto es equivalente a tener un conjunto finito de realizaciones de las variables aleatorias, y el algoritmo en cada paso que da hacia atrás resuelve absolutamente todas las realizaciones, a través de un problema de despacho diferente para cada una de ellas.

La cantidad de problemas de despacho total que se resuelve en el algoritmo, es crucial para disminuir los tiempos de cómputo. En el siguiente capítulo, se analiza la posibilidad de atacar la maldición de la dimensionalidad estocástica utilizando la funcionalidad que se incorpora en la biblioteca de seleccionar solamente algunas de las realizaciones de la aleatoriedad, en lugar de todas. Es por ello que se incorporó en la biblioteca la posibilidad de seleccionar un número menor de realizaciones. En el

Apéndice A, se detalla cómo hacer uso de esta funcionalidad.

Otra de las posibilidades para disminuir la cantidad total de problemas de despacho a resolver, es la de reducir la cantidad de realizaciones de aleatoriedad, a medida que el paso de tiempo se acerca al fin. Esto intuitivamente resulta del hecho de que representar efectos del futuro con menor detalle podría no afectar la calidad de la solución. Con este propósito se implementó la funcionalidad que permite reducir la cantidad de realizaciones a medida que avanza el tiempo.

5.3.2. Otras funcionalidades

Además de las funcionalidades descriptas, se incorporaron otras funcionalidades que facilitan la realización de pruebas y casos de estudio, así como también la salida del sistema.

Entre estas funcionalidades se encuentran:

Diferentes formas de simular la hidrología La simulación de la hidrología en un sistema hidrotérmico, resulta fundamental. En la biblioteca utilizada en el Capítulo 4, solamente era posible simular con el mismo proceso estocástico que se había utilizado para encontrar la política óptima. Se incorporaron en este caso entonces las siguientes posibilidades:

- *Utilización del mismo proceso de la optimización*: Se permite simular los aportes hidrológicos utilizando el proceso con el cuál se halló la política óptima
- *Sorteos independientes*: En cada paso de simulación se sortean aportes de un paso equivalente dentro del año tomado de la historia, pero sin mantener la correlación temporal.
- *Crónicas históricas*: Se simula cada uno de los escenarios que sucedieron en la historia.
- *Política miope*: Se simula asumiendo que no hay costo futuro, para ello se establece en cero el valor de la función de Bellman.

Cada una de estas formas de simular puede tener su utilidad dependiendo el estudio que se esté realizando.

Interfaz con resolvedor lineal Se incorporó una interfaz de software que permite la utilización de diferentes resolvedores lineales para colaborar con la resolución del algoritmo SDDP. En particular se implementó la interfaz para el resolvedor Gurobi [23]. El detalle de esta implementación se presenta en el Apéndice A.

Salidas útiles para el análisis La biblioteca utilizada en el Capítulo 4, posee una serie de salidas de datos muy buena. Sin embargo, algunas salidas fundamentales no estaban presentes, lo que limitaba el análisis de los estudios. Se implementaron por lo tanto diversas salidas en la biblioteca desarrollada:

- Un resumen con todas las variables relevantes y el control óptimo para cada uno de los escenarios simulados.
- Un archivo con las aproximaciones mediante hiperplanos de las funciones de Bellman de cada uno de los pasos.
- La posibilidad de imprimir a archivo cada uno de los problemas lineales resueltos en el ámbito del algoritmo SDDP en un formato estándar para poderlo analizar en cualquier resolvedor lineal externo.

Los detalles se presentan en el Apéndice A.

Poda de hiperplanos Resultó necesario, además, implementar varios algoritmos de poda de hiperplanos, procurando que un determinado problema lineal no crezca excesivamente en cuanto a número de restricciones. Para ello se implementaron los siguientes métodos.

- *K-últimos*: Se quitan del problema lineal los hiperplanos que tienen una vida mayor a K iteraciones. Este método es muy fácil de implementar, pero se pierden hiperplanos útiles y a partir de cierta cantidad de iteraciones no se reduce el gap entre el costo simulado y la cota inferior dada por los hiperplanos.
- *K-últimos activos*: Se quitan del problema lineal los hiperplanos que tienen una vida mayor a K iteraciones, tomando en cuenta sólo los que se encuentran activos al evaluar la función de valor. Resulta fácil de implementar, pero tiene el mismo problema que el anterior, aunque se logran mejores resultados con valores de K más pequeños.
- *L - Nivel de dominio [14]*: Es más complejo de implementar y el consumo de memoria es mayor. Mantiene un nivel menor de hiperplanos a considerar manteniendo una buena aproximación.

Los detalles informáticos se presentan en el Apéndice A.

En el siguiente capítulo se explora la posibilidad de mitigar la maldición de la dimensionalidad estocástica. Para ello, muchas de las funcionalidades incorporadas a la biblioteca se utilizan.

5.4. Conclusiones

En este capítulo se presentó el desarrollo de una biblioteca que implementa el algoritmo SDDP, específicamente al problema de operación de un sistema eléctrico. Para ello se implementó en Python, utilizando un paradigma orientado a objetos, el algoritmo estándar SDDP [32]. Luego se incorporaron funcionalidades habituales, necesarias para el buen desempeño en términos de tiempos de ejecución. Además, se incorporaron funcionalidades que permiten abordar en el siguiente capítulo la reducción de los tiempos de ejecución provocada por la maldición de la dimensionalidad estocástica. En el Apéndice A se describen las funcionalidades y el uso de la biblioteca desde una óptica informática. Una línea de trabajo a futuro es la de estructurar la biblioteca de forma que pueda ser utilizada de manera genérica por la comunidad científica interesada en el modelado de la operación de un sistema eléctrico. Para lograr este objetivo, es necesario cumplir con una serie de estándares de documentación técnica y de usuario que exceden el alcance de esta Tesis.

Capítulo 6

Análisis: Maldición de la dimensionalidad estocástica en SDDP

En este capítulo se aborda el problema de mitigar la maldición de la dimensionalidad estocástica [35] presente en SDDP. En el análisis presentado en el Capítulo 4, se estudió el impacto que produce la incorporación de variables de estado en el tiempo de ejecución del algoritmo SDDP, manteniendo constante el resto de los parámetros que afectan el tiempo de cómputo.

En el presente análisis se pretenden atacar dos de los puntos débiles que tiene el algoritmo SDDP. En primer lugar, reducir la cantidad de problemas lineales totales que se plantean en una corrida debido a la necesidad de contemplar de forma exhaustiva las realizaciones de las variables aleatorias. En segundo lugar, mantener cada problema lineal acotado en términos de cantidad de restricciones, lo que implica analizar técnicas de poda de hiperplanos.

En el trabajo presentado en el Capítulo 4, se utilizó una biblioteca de SDDP desarrollada en el lenguaje Julia. A pesar de que esta biblioteca es de código libre, realizar modificaciones o agregados a la misma que permitan implementar variantes sobre el algoritmo SDDP estándar, no resulta trivial. Dada la necesidad de incorporar diversas funcionalidades adicionales al algoritmo SDDP para investigar técnicas que permitan mitigar la maldición de la dimensionalidad estocástica, se optó por el desarrollo de una biblioteca en Python que implementa SDDP. En esta biblioteca, al ser de desarrollo propio, resulta muy sencillo incorporar técnicas específicas al algoritmo estándar.

Se analizan en este trabajo técnicas de reducción del número de realizaciones de variables aleatorias con el objetivo de disminuir la cantidad total de problemas lineales. Además, se analizan diferentes técnicas de poda de hiperplanos con el objetivo de disminuir el tiempo de resolución de cada problema lineal. Asimismo, se presenta un caso de estudio que permite analizar en conjunto estas técnicas y el modelo estocástico propuesto.

6.1. Motivación

El algoritmo más utilizado para la resolución del problema de operación, es el de la programación dinámica estocástica con discretización del espacio de estados (SDP).

Como se mencionó en la Capítulo 2, este algoritmo tiene el inconveniente que incurre en la maldición de la dimensionalidad del espacio de estados.

Cómo ejemplo, si se modela el sistema uruguayo con tres centrales con capacidad de almacenamiento (Bonete, Palmar y Salto) cada una con una variable de estado asociada y una variable de estado adicional para representar la situación hidrológica, se tienen cuatro variables de estado $V_{bon}, V_{pal}, V_{sal}$ y H, respectivamente. Si se toma la decisión de discretizar cada variable asociada a un lago en diez pasos y la variable hidrológica en cinco pasos, el espacio de estados discreto resultante tiene $\#V_{bon} \times \#V_{pal} \times \#V_{sal} \times \#H = 10^3 \times 5 = 5000$ puntos.

Dadas las características del algoritmo SDP, en cada paso de tiempo de optimización se calcula el valor de Bellman en un punto del espacio de estados mediante el método de Montecarlo aplicado a una muestra representativa de las realizaciones de las variables aleatorias. Si se supone una población finita representando a la distribución conjunta de las variables aleatorias de cardinal W y se toma una cantidad de muestras 5 %, entonces la cantidad total de resoluciones de problemas lineales en un paso de optimización sería de $\frac{W}{20} * \#V_{bon} \times \#V_{pal} \times \#V_{sal} \times H$. Para el ejemplo del sistema uruguayo simplificado descripto en el párrafo anterior la aleatoriedad está dada por los aportes hidrológicos. Dada una semana del año, se toma como población base el conjunto de aportes observados en las crónicas hidrológicas desde 1909 hasta la fecha en la semana correspondiente del año. En este caso entonces $\frac{W}{20} \approx 5$, por lo que en el caso de SDP para cada paso de optimización se estarían resolviendo $5 \times 10^3 \times 5 = 25000$ problemas lineales.

En el caso de SDDP, el algoritmo exige que para cada punto en donde se construye un hiperplano para agregar a la función de costo futuro, se utilice toda la población que representa la distribución empírica. En el ejemplo citado anteriormente del sistema uruguayo, en SDP por cada punto del espacio de estados se resuelven aproximadamente 5 problemas lineales, mientras que en SDDP por cada punto deberían resolverse en el entorno de 100, o sea, 20 veces más cantidad de problemas lineales por cada punto optimizado. En contraste, SDDP no requiere recorrer todos los puntos del espacio de estado para lograr una aproximación buena de la función de costo futuro. En definitiva, la cantidad de problemas lineales a resolver depende fundamentalmente de la cantidad de muestras a tomar por cada punto del espacio de estados en las pasadas hacia atrás del algoritmo. En caso de que no sea posible modelar mediante un conjunto acotado de muestras las distribuciones de las variables aleatorias involucradas, se manifiesta la denominada maldición de la dimensionalidad estocástica. En particular, cuando se requiere generar muestras multivariadas para representar distribuciones conjuntas.

En la Figura 6.1, se puede observar una trayectoria de la pasada hacia adelante. A partir del estado inicial (x_0, h_0), se sortea w_0 condicionado a h_0 y se resuelve el problema asociado al paso utilizando la aproximación actual de la función de Bellman para el paso actual y el estado discreto h_0. Una vez resuelto el problema del paso, la dinámica permite avanzar hasta el paso siguiente $t = 1$ y determinar el estado futuro x_1. A través de la matriz de transición asociada al estado h, se obtiene el nuevo estado h_1 y se repite el proceso hasta cubrir el horizonte completo de la corrida ($t = T$). Al finalizar la pasada hacia adelante, se preserva la trayectoria ejecutada, por la cual se realizará la pasada hacia atrás.

En la pasada hacia atrás, se recorre en orden inverso cada estado de la trayectoria visitada en la pasada hacia adelante. En la Figura 6.2, se presenta un esquema de la resolución de un estado x_k, h_k. Para cada uno de los estados markovianos posibles del paso $k+1$, se resuelve una serie de problemas lineales utilizando la aproximación futura actual, asociados a cada una de las realizaciones de W condicionadas al estado h_{k+1}. El conjunto de soluciones primales y duales de dichos problemas, permiten construir un nuevo hiperplano de acuerdo a la Ecuación 6.1 que se agregará a la aproximación de

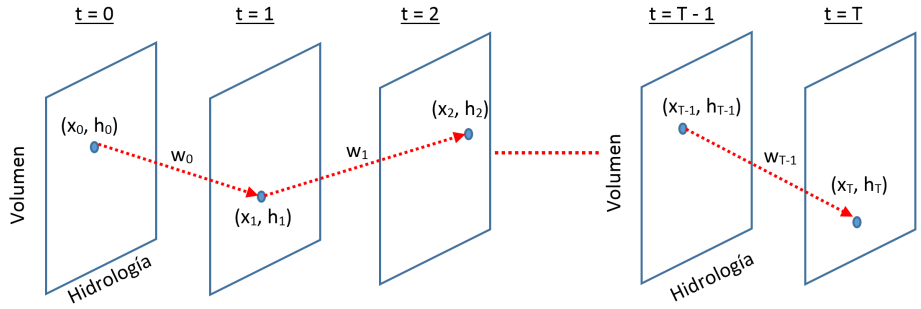


Figura 6.1: Esquema de ejecución de una pasada hacia adelante.

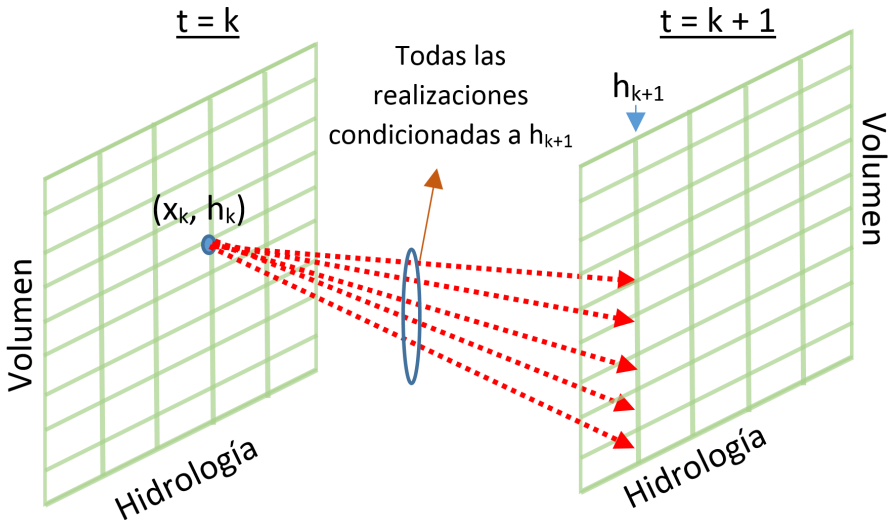


Figura 6.2: Esquema de resolución de un conjunto de transiciones de estado fijada la transición de Markov.

Bellman del paso $k+1$ y el estado h_{k+1}. Este proceso se repite hasta llegar a $t = 0$. Una vez finalizada la vuelta atrás, se tiene para cada paso y cada estado markoviano, una aproximación mejorada respecto a la iteración anterior (con un hiperplano adicional).

En sistemas eléctricos extremadamente grandes y con decenas de componentes de almacenamiento (principalmente embalses hidráulicos), como por ejemplo el sistema brasileño, es de suma importancia mitigar la maldición de la dimensionalidad del espacio de estados para tener un modelo que compute en tiempos razonables. En el presente trabajo se analizó en el Capítulo 4 el impacto de incorporar variables de estado en un sistema eléctrico utilizando SDDP. Sin embargo, el sistema uruguayo es un sistema pequeño y en la actualidad puede ser representado adecuadamente con un conjunto pequeño de variables de estado. Esto hace que sea de vital importancia buscar variantes algorítmicas dentro de SDDP que mitiguen la maldición de la dimensionalidad estocástica, ya que con pocas variables de estado el beneficio que brinda esta técnica se compensa con la necesidad de utilizar mayor cantidad de muestras. En consecuencia, para un sistema chico, el algoritmo SDDP puede ser incluso más costoso que el algoritmo SDP.

Es posible que en unos años se incremente en el sistema uruguayo la presencia

de dispositivos de almacenamiento o de recursos para gestionar la demanda que impliquen aumentar la cantidad de variables de estado a considerar. En esas hipótesis el algoritmo SDDP se presentaría con muchas ventajas respecto al algoritmo SDP. Sin embargo, en la situación actual, resulta imprescindible abordar el problema de disminuir la cantidad de muestras que se toman por cada punto del espacio de estados en la resolución de SDDP. En el trabajo desarrollado por Higle [26], se aborda esta problemática para el problema de dos pasos. Más de 20 años después, Zen [40] presenta una solución equivalente a la propuesta por Higle pero para el problema de múltiples pasos de tiempo. Estos trabajos se analizaron en profundidad en el marco de esta tesis, llegándose a la conclusión de que no logran mitigar la maldición de la dimensionalidad. Básicamente, los trabajos citados plantean sortear un subconjunto de la muestra completa disponible para reducir la cantidad de problemas lineales a resolver en cada paso visitado en la vuelta hacia atrás. El problema que presentan es que la cantidad de iteraciones necesarias para lograr convergencia al reducir las muestras sorteadas, aumenta considerablemente. Finalmente, la cantidad total de problemas lineales resueltos luego de la convergencia resulta similar al método estándar, por lo que no existe una reducción en los tiempos de cómputo en la práctica.

Además de la cantidad de problemas lineales a resolver, otro de los factores que resulta imprescindible considerar para disminuir los tiempos de cómputo, es el de reducir la complejidad de cada uno de los problemas lineales. En el caso de SDP, las restricciones que se presentan en el problema lineal son las relacionadas al modelado de cada componente de la realidad, y se considera el costo futuro de los recursos que poseen estado a través de un término lineal en el objetivo. Sin embargo, en SDDP la función de costo futuro se aproxima mediante un conjunto de hiperplanos como se describió en 2, lo que implica una restricción en el problema lineal por cada hiperplano considerado en la función. Esto hace que en SDDP se incremente el tiempo de cómputo del problema a resolver a medida que se tiene una aproximación más rica (mayor cantidad de hiperplanos).

Por lo tanto, los objetivos del trabajo presentado en este capítulo son:

- Analizar estrategias para mitigar la maldición de la dimensionalidad estocástica.
- Analizar técnicas de poda de hiperplanos que permitan acotar la complejidad computacional del problema lineal a resolver.
- Comparar la técnica SDDP con las mejoras analizadas y respecto de la técnica estándar SDP.

6.2. Modelo estocástico reducido y poda de hiperplanos

En cualquier variante de la programación dinámica estocástica aplicada en un horizonte de tiempo, es necesario lograr obtener una aproximación de la función de costo futuro en todos los pasos de tiempo t para cualquier valor de estado. En la técnica SDP, en la que discretiza el espacio de estados para cada paso de tiempo, debe obtenerse para cada estado una estimación de la esperanza del valor de Bellman. Para ello es habitual generar varias realizaciones del azar condicionado al estado y resolver el problema de despacho para cada realización, obteniendo de esa manera la esperanza empírica. Por la ley de los grandes números, al incrementar la cantidad de muestras la aproximación del costo en el estado considerado es de mejor calidad. Si se tiene una población discreta y finita de la aleatoriedad que respeta la distribución probabilística real, es suficiente tomar una muestra representativa para obtener una buena aproximación, por lo que se reduce significativamente la cantidad de problemas a resolver en ese estado tomando dicha muestra. A continuación, se verá que en SDDP la propia estructura de la aproximación de la función de valor impide abordar el modelo

de aleatoriedad fijado un estado en un paso de tiempo de la misma forma que en SDP. También se presenta una posible solución al respecto. Por otra parte, se presentan las técnicas de poda de hiperplanos consideradas para el caso de estudio presentado.

6.2.1. Sorteo de muestra aleatoria

De acuerdo a la Ec. 6.1 y de las definidas en la Sección 2.3, para incorporar un nuevo hiperplano en la pasada para atrás se debe resolver la esperanza en el conjunto de variables aleatorias del sistema del valor de costo $\hat{\beta}$ y de cada una de las variables duales en ese punto, $\hat{\lambda}$.

$$\beta_k^{(l+1)} = E[\hat{\beta}_k^{(l+1)}(w)], \lambda_k^{(l+1)} = E[\hat{\lambda}_k^{(l+1)}(w)] \tag{6.1}$$

El algoritmo numérico SDDP, exige calcular dicha esperanza de forma empírica utilizando, durante todas las iteraciones del algoritmo, una población de realizaciones del azar fija dado un estado y un paso de tiempo. Si la población es grande, surge la inquietud de proceder de la misma manera que en SDP. Esto es, sortear una muestra pequeña de la población y utilizarla para calcular las esperanzas. Sin embargo, al analizar la estructura de aproximación por hiperplanos que presenta SDDP, se observa que, si se utiliza una muestra sorteada cada vez que se va a agregar un nuevo hiperplano, el mismo puede tener un sesgo hacia abajo o hacia arriba fijado el estado, respecto del hiperplano que resultaría de utilizar toda la población. La aproximación por hiperplanos permite obtener la aproximación $V_k(x)$, en un paso t fijo, luego de k iteraciones del algoritmo en el punto x del espacio de estados. Se tiene que $V_k(x) = max_x\{h_1(x), \ldots, h_k(x)\}$ donde h_i son los hiperplanos presentes en la aproximación luego de k iteraciones. Teniendo en cuenta el carácter incremental del conjunto de hiperplanos (en el algoritmo estándar no se quitan ni se modifican), es claro que si $b > a$ entonces $V_b(x) \geq V_a(x)$. Esto se debe a que el $max\{A\} \geq max\{B\}$ si $A \supset B$. En consecuencia, a medida que ocurren las iteraciones, la aproximación de la función de valor en un punto siempre se incrementa. Una vez que la aproximación se acerca a la función real, puede existir un sesgo que sobrestime un hiperplano de manera que supere en un punto a dicha función, provocado por no haber tomado toda la población aleatoria. Esto incumple la premisa de que los hiperplanos siempre son una cota inferior de la función real y en la práctica no garantiza convergencia. En la Figura 6.3 se puede observar un ejemplo con una sola variable de estado continua x. Se observa la función de valor real en el paso de tiempo t en color negro, V_t, la aproximación actual de la función de valor $\hat{V}_t^k$ en color verde y la incorporación de un nuevo hiperplano h_{k+1} (color rojo). A su vez, es claro que la aproximación $\hat{V}_t^{k+1}$ no es una cota inferior en todo el espacio de V_t.

6.2.2. Selección de una muestra fija por paso de tiempo

Debido al inconveniente presentado en la sección anterior, si se quiere reducir la cantidad de problemas lineales a resolver en cada punto de la trayectoria hacia atrás, es necesario fijar una muestra representativa de toda la población y mantenerla hasta la convergencia. En la práctica, es habitual utilizar datos históricos como distribución conjunta empírica condicionada al estado, tal como se presentó con los aportes hidráulicos en el Capítulo 2. En el caso de Uruguay, existen datos de aportes hidráulicos a las principales cuencas desde el año 1909 [3]. De acuerdo al modelo presentado, que contempla el paso semanal, existen por cada semana del año más de cien medidas históricas. Si se quieren utilizar, dada una semana del año, todos los aportes que sucedieron en esa semana en la historia, deberían resolverse más de cien problemas

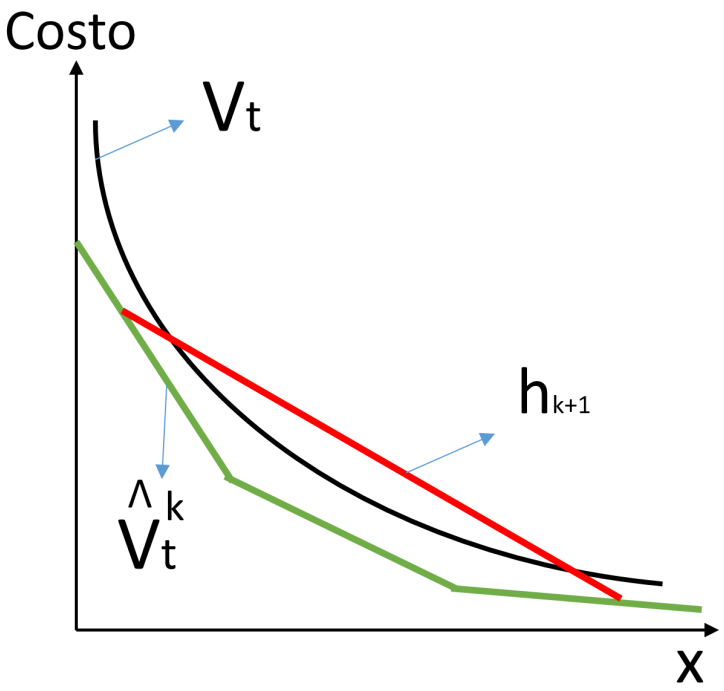


Figura 6.3: Función de Bellman y su aproximación sucesiva por hiperplanos.

lineales por cada paso de la vuelta atrás. En caso de que haya más variables aleatorias involucradas, capturar la distribución conjunta de todas ellas puede resultar extremadamente complejo. Este impacto en el modelado de la distribución de probabilidad conjunta de las variables aleatorias condicionada al estado actual es lo que se conoce como la maldición de la dimensionalidad estocástica.

Una de las razones principales para modelar el problema de operación de un sistema eléctrico utilizando SDDP radica en poder mitigar la maldición de la dimensionalidad del espacio de estados. La necesidad de ampliar el espacio de estados en sistemas modernos cobra una gran importancia. Muchas son las razones para incorporar variables de estado al modelado. Por ejemplo, cualquier almacenamiento de energía relevante implica ampliar el espacio de estados, ya sea un embalse hidráulico, una batería o incluso almacenar combustibles para usar en el parque térmico. Por otro lado, cualquier generador vital para el sistema debería contribuir a ampliar el espacio de estados, en particular para modelar su disponibilidad. Es sabido que la probabilidad de rotura de una máquina es altamente dependiente del tiempo que transcurrió desde su última reparación, por lo que dicha variable debería pertenecer al espacio de estados.

Otra razón para ampliar el espacio de estados, es el modelado de la falla del sistema. Cuando no es posible abastecer la demanda, es necesario tomar recaudos para evitar un colapso. Una de las primeras medidas que se toman antes de optar por cortes de energía, consiste en solicitar a la población que ahorre. Este tipo de medidas, una vez que se toman, deben estar desplegadas por un lapso de tiempo prudente. Por ejemplo, no tendría sentido solicitar a la población que ahorre un día y al otro día pedirle que deje de ahorrar. Al modelar este tipo de instrumentos de falla, se debe garantizar entonces que la medida tomada permanezca activa a lo largo de varios pasos, lo que implica utilizar una variable de estado.

Por otra parte, el modelado de procesos estocástico requiere en muchas ocasiones la incorporación de estado. Así como sucede en el caso del modelo markoviano presentado para los aportes, podría surgir la necesidad de mantener estado para otro tipo de recursos, como ser, por ejemplo, el recurso eólico o el recurso solar.

En el algoritmo SDP, resulta imposible contemplar siquiera más de una decena de variables de estado. Esta es la ventaja comparativa que posee el algoritmo SDDP respecto del SDP, ya que permite incorporar decenas de variables de estado. A cambio de esta ventaja, la hipótesis de convexidad en la función de costo futuro resulta más restrictiva. Sin embargo, en sistemas con complejidad creciente resulta mucho más beneficioso modelar un espacio de estados rico a pesar de tener que cumplir con la hipótesis de convexidad. Es por eso que, a pesar de que el algoritmo SDDP logra mitigar la maldición de la dimensionalidad de estados, no logra hacerlo con la maldición de la dimensionalidad estocástica. Por esta razón, se exploran en uno de los casos de estudio presentado en este capítulo, técnicas de reducción de muestras para mitigar la maldición de la dimensionalidad estocástica.

6.2.3. Poda de hiperplanos

Se utilizaron diversas técnicas de poda de hiperplanos, todas estudiadas ampliamente en la bibliografía [7, 14, 22]. Se enumeran a continuación las que implementaron en la biblioteca y se utilizaron en el caso de estudio de este capítulo. En el propio caso de estudio se analizan ventajas y desventajas de cada una. Se utilizaron diversas técnicas de poda de hiperplanos, todas estudiadas ampliamente en la bibliografía [7, 14, 22]. Se enumeran a continuación las que implementaron en la biblioteca y se utilizaron en el caso de estudio de este capítulo. En el propio caso de estudio se analizan ventajas y desventajas de cada una.

1. k-últimos: Estrategia que elimina de la aproximación los que tienen una vida mayor a K iteraciones.
2. k-últimos activos: Estrategia que elimina de la aproximación los que tienen una vida mayor a K veces estando inactivos a lo largo de las iteraciones.
3. l-niveles de dominancia: Estrategia más compleja que trabaja sobre hiperplanos dominados.De Matos describe en [14] esta estrategia de forma muy detallada.

6.3. Casos de estudio

Se plantea en esta sección una serie de casos de estudio con el objetivo de analizar la incorporación de estrategias para mitigar la dimensión de la dimensionalidad estocástica (caso de estudio 1) y las técnicas de poda de hiperplanos para acotar la complejidad computacional del problema a resolver (caso de estudio 2). No se hace hincapié en la mitigación de la maldición de la dimensionalidad del espacio de estados, tema ya analizado en el Capítulo 4. A su vez, en el caso de estudio 3, se compara el desempeño en términos de tiempo de ejecución de las técnicas SDP y SDDP. Todos los casos de estudio se realizan sobre un sistema de generación hidrotérmica que incorpora recursos solares y eólicos, además de modelar un intercambio internacional de importación exportación y una falla para la demanda. El modelo de referencia que se toma como base de comparación en término de resultados y tiempos de cómputo aplica programación dinámica estocástica (SDP). Es un modelo muy simplificado del sistema uruguayo, en términos generales.

	Volumen inicial (hm^3)	**Volumen máximo** (hm^3)	**Turbinado máximo** $(\frac{m^3}{s})$	**Coeficiente energético** $(MW/\frac{m^3}{s})$
Bonete	4000	8200	680	0.2
Baygorria	186	—	828	0.12
Palmar	440	1300	1374	0.25
Salto	1280	3000	4200	0.11

Tabla 6.1: Hipótesis para los parámetros de los generadores hidráulicos.

6.3.1. Sistema de referencia y parámetros del estudio

A continuación, se describen las características del sistema de referencia utilizado para analizar los casos de estudio. Se debe tener en cuenta que los resultados obtenidos en términos de costos y energías, **no pretenden representar en detalle un estudio real para Uruguay, sino que simplemente captar la complejidad del problema en términos de dimensión para analizar mejoras en los tiempos de ejecución**.

Parámetros generales. El estudio tiene un horizonte de 3 años en la optimización (2022-2024), con paso diario. La simulación para la obtención de los resultados se realiza con las crónicas históricas de aportes. Tiene un horizonte de 2 años (2022-2023), de manera de tener un año final de guardia que elimine los efectos de fin de juego. Se utiliza una tasa de descuento del 10 % anual.

Parque. El parque considerado consta de los siguientes componentes: 4 centrales hidráulicas siendo una simplificación de las existentes en Uruguay; 1 central térmica con una potencia instalada equivalente a todo el parque térmico; 1 parque eólico equivalente a toda la potencia eólica instalada en Uruguay; 1 parque solar fotovoltaico equivalente a toda la potencia fotovoltaica instalada en Uruguay; una demanda equivalente a la de Uruguay proyectada con un 2 % de crecimiento anual a partir de la de 2021; una falla asociada a la demanda; un intercambio internacional de exportación y otro intercambio internacional de importación representando en conjunto el intercambio en ambas direcciones con Argentina y Brasil. En la Tabla 6.1, se encuentran todos los detalles de cada una de las centrales hidráulicas presentes en el sistema. En el caso de la central de Baygorria, se considera un volumen constante, ya que el embalse es tan pequeño que no se considera para el estudio. Todas las centrales hidráulicas se modelan sin límite de vertimiento. Por otra parte, cada central se modela con un coeficiente energético constante igual al promedio con el que operan las mismas.

En la Tabla 6.2, se presentan los parámetros correspondientes a los componentes térmicos, eólicos, fotovoltaicos, importaciones, exportaciones y fallas. Se considera una sola máquina térmica con el total de la potencia instalada del sistema y con un costo variable por unidad de energía equivalente al costo medio de todas las centrales. A su vez, se considera una sola máquina eólica con una potencia instalada equivalente a la potencia total de todos los parques eólicos y con un factor de planta medio acorde a las medidas históricas. De la misma manera que el recurso eólico, se modela el fotovoltaico. En la misma tabla se presentan las hipótesis respecto de la importación, la exportación y la falla. Se modela el costo térmico menor a la importación y el costo de importación, menor a la falla.

	Potencia Instalada (MW)	Costo (USD/MWh)	Factor de planta
Térmico	1400	150	—
Eólico	1445	0	0.39
Fotovoltaico	240	0	0.22
Importación	1000	200	—
Exportación	1000	40	—
Falla	2000	250	—

Tabla 6.2: Hipótesis para los parámetros de los componentes térmico, eólico, solar, importación, exportación y falla.

Azar. La única aleatoriedad que presenta este sistema simplificado, corresponde a los aportes hidráulicos. Se representa a través de un modelo de Markov para aportes tal como el descripto en el Capítulo 2. Los procesos que definen el factor de planta eólico, el factor de planta solar y la demanda, son determinísticos. Para la demanda se toma la de 2019 como base y se ajusta por un factor de crecimiento de energía anual, manteniendo el perfil de la curva de carga horaria. Los factores de planta del eólico y del solar horarios considerados, son los de 2019. El hecho de que estos recursos sean determinísticos, no afecta el análisis comparativo que se pretende realizar en este capítulo.

6.3.2. Caso de estudio 1: Selección de conjunto reducido de realizaciones del azar

En este caso de estudio se procede a reducir la cantidad de realizaciones de aportes hidráulicos para cada estado markoviano. Para ello se toma como resultado de referencia el costo esperado simulado con crónicas históricas obtenido a partir de la resolución de la programación dinámica estocástica, con el sistema definido en la Sección 6.3.1. Luego se resuelve utilizando SDDP con la totalidad de las realizaciones. Y finalmente se toman cuatro casos de reducción de la cantidad de realizaciones: 80 %, 60 %, 40 % y 20 %.

De acuerdo a lo expuesto en el Capítulo 2, para cada semana del año se tienen 100 valores de aportes históricos. Esto implica que por cada uno de los valores de la variable de estado hidrológica que condiciona al proceso estocástico de aportes, se tienen 20 valores, ya que la partición es por quintiles dentro de las 100. Dependiendo del porcentaje de realizaciones que se utilice, la cantidad que se preserva por estado es: 16 para el 80 %, 12 para el 60 %, 8 para el 40 % y 4 para el 20 %. Como el objetivo es analizar la repercusión en el costo esperado simulado, se considera una diferencia máxima admisible el 3 % respecto del modelo de referencia para considerar que la reducción llevada a cabo es aceptable.

Para cumplir con este objetivo de reducción se procede de la siguiente forma. Según el Capítulo 2, $W_{y,l}$ es un escalar para el año y de la historia y la semana l dentro del año, que se obtiene al ponderar los aportes de las centrales correspondientes por su coeficiente energético. Cada $W_{y,l}$ tiene asociado por tanto una terna (la terna a partir de la cual se obtuvo el escalar). Sobre esta variable se reduce la cantidad de realizaciones tomando un subconjunto de la distribución empírica dada por la historia. Para un paso dado y un valor de la variable de estado hidrológica (h), se obtienen los representantes históricos W que se clasifican con ese h. Dado que se trabajó con 100 años de historia y que un valor de h representa un quintil, por cada valor de h fijada la

semana, existen entonces 20 valores de W asociados a ese h. Por lo que se debe reducir la representación de la distribución $P(W_l|h)$, seleccionando de acuerdo a cada caso un subconjunto de estos 20 valores que mantenga el mínimo y el máximo valor de W_l. Para realizar la selección, se procede a tomar cuantiles sobre los 20 valores de acuerdo a la cantidad de muestras que se quieran preservar. Como no existe consenso respecto del cálculo de cuantiles en una distribución discreta, se explica a continuación cómo fueron tomados los representantes.

Se supone que la muestra ordenada es $W_l^{(1)} \leq W_l^{(2)} \leq \dots W_l^{(n)}$ y que son equiprobables. Si la cantidad de cuantiles a considerar es k, para elegir el representante de la muestra W_p que divide dos cuantiles adyacentes, se calcula $p = i * (n+1)/k$ con $i \in 1...k-1$. Si p es entero, entonces se toma W_p como elemento de partición y si p no es entero se elige el valor límite $\frac{W_{\lfloor p \rfloor} + W_{\lceil p \rceil}}{2}$. Para cada subconjunto de la muestra inicial clasificado como cuantil, se toma la mediana como representante de dicho cuantil. El conjunto de todos los representantes es $\hat{W}$. Cómo cada valor de $\hat{W}$ tiene asociada una terna de aportes, se tiene entonces un conjunto reducido de ternas de aportes a utilizar en el algoritmo SDDP.

Para fijar ideas se supone aplicar el procedimiento para reducir el conjunto $W = \{50, 80, 200, 310, 350, 500, 650, 750, 800, 1000, 1200, 1400\}$ según cuartiles ($k = 4$) se obtiene que los valores de partición son $\{255, 575, 900\}$. De esta manera quedan los conjuntos asociados a cada cuartil de la siguiente forma:

- Cuartil 1, $P(W \leq 0{,}25)$: $C_1 = \{50, 80, 200\}$
- Cuartil 2, $P(0{,}25 \leq W \leq 0{,}5)$: $C_2 = \{310, 350, 500\}$
- Cuartil 3, $P(0{,}5 \leq W \leq 0{,}75)$: $C_1 = \{650, 750, 800\}$
- Cuartil 4, $P(W \geq 0{,}75)$: $C_1 = \{1000, 1200, 1400\}$

Es así que el conjunto reducido de representantes obtenido (mediana de cada subconjunto) es: $\hat{W} = \{80, 350, 750, 1200\}$.

En la Figura 6.4 se observan las distribuciones discretas para la muestra completa y para la muestra reducida para el ejemplo presentado. Es importante tener en cuenta que para validar si la reducción obtenida resulta en una simplificación adecuada se debe observar el resultado empírico del modelo de operación, comparando los diferentes resultados del modelo según las reducciones realizadas. Este análisis se presenta en la Sección 6.4.

Dentro de este caso de estudio, se analiza la posibilidad de reducir la cantidad total de problemas de despacho a resolver utilizando menos cantidad de realizaciones del azar a medida que el paso de tiempo se acerca al final del horizonte. Esto intuitivamente resulta del hecho de que representar efectos del futuro con menor detalle podría no afectar la calidad de la solución. Con este propósito se implementó la funcionalidad que permite reducir la cantidad de realizaciones a medida que avanza el tiempo. Se exponen los resultados en la Sección 6.4.

6.3.3. Caso de estudio 2: Poda de hiperplanos

El objetivo del presente capítulo es el de reducir los tiempos de cómputo no vinculados a la dimensión del espacio de estados. Uno de los factores que impactan en el tiempo de ejecución es la dimensión del espacio estocástico, ya que esto repercute directamente en la cantidad de problemas lineales que hay que resolver. Esto es tratado en el Caso 1. Otro de los aspectos a considerar desde el punto de vista estructural del algoritmo es el de disminuir el tiempo de ejecución de cada uno de los problemas lineales a resolver. Considerando el trabajo de De Matos [14] como referencia dentro de la vasta literatura que existe respecto de la poda de hiperplanos, se entendió que

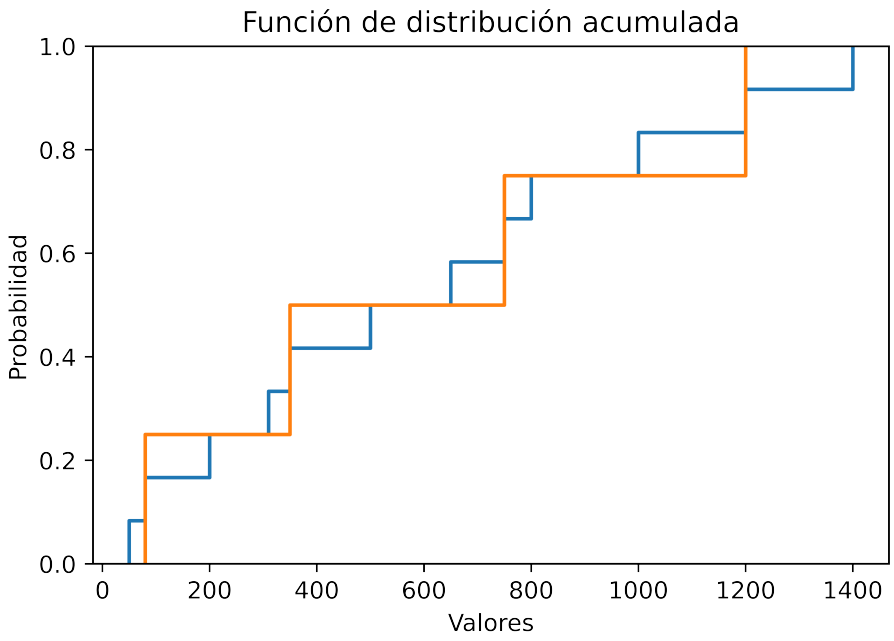


Figura 6.4: Comparación entre las funciones de distribución acumulada antes y después de la reducción.

se debían implementar los métodos más extendidos en los casos de estudio. La razón para incorporar este tipo de técnicas es que producen una reducción significativa en los tiempos de cómputo. Por esta razón, se incorporó a la biblioteca desarrollada (Capítulo 5) las funcionalidades de poda descriptas.

En este caso de estudio se analiza en forma desacoplada el impacto de las diferentes técnicas de poda en los tiempos de cómputo, en los despachos de energía esperados y en los costos esperados analizados sobre el sistema de referencia presentado.

6.3.4. Caso de estudio 3: Comparación con SDP

En este caso de estudio, se plantea como objetivo comparar la técnica SDDP mejorada según lo analizado en los casos de estudio anteriores respecto de la técnica SDP. La resolución SDP que se utiliza, discretiza el lago de la central de Rincón de Bonete en 10 valores, el de la central de Palmar en 5 valores y el de la central de Salto Grande en 5 valores. A su vez, en cada punto del espacio de estados discretizado, se realizan 5 sorteos para reproducir la distribución condicional. Para realizar esta comparación se analiza la operación del embalse hidráulico más grande del sistema (central de Bonete) en ambas corridas, en particular para una crónica hidráulica extrema, la del año 1917 (la más seca de la historia). También se comparan de forma general los costos y las energías esperadas anuales para cada uno de los componentes del parque de generación.

6.4. Resultados y conclusiones

En esta sección se presentan los resultados obtenidos en los casos de estudio y las conclusiones a las que se llegó. El modelo de referencia que se considera en los casos

Porcentaje de realizaciones (%)	Tiempo de ejecución (h)	Valor en función de valor aproximada (MUSD)	Costo simulado histórico (MUSD)	Error relativo (%)
20	6.4	159.4	181.3	13.7
40	12.5	159.0	178.7	12.4
60	18.9	161.3	176.1	2.8
80	24.3	163.1	173.4	1.2
100	31.1	164.7	171.3	0.0

Tabla 6.3: Resultados del análisis de porcentaje de realizaciones.

de estudio 1 y 2, es el que aplica SDDP utilizando el conjunto total de realizaciones y sin realizar poda de hiperplanos, ya que en ese caso se está en las hipótesis de la teoría SDDP, por lo que los resultados son exactos de acuerdo a un criterio de parada.

6.5. Resultados: Caso de estudio 1

Para realizar este estudio se realizaron cinco corridas con las hipótesis descriptas en la Sección 6.3.1 de acuerdo a lo definido en la Sección 6.3.2. Para este análisis se tomó como criterio de parada la realización de 2000 iteraciones, ya que empíricamente se verificó que para el caso de referencia es un número que garantiza la calidad de la solución.

En la Tabla 6.3, se presentan los resultados obtenidos dependiendo del porcentaje elegido sobre el total de la muestra.

Una primera hipótesis que confirma la tabla, es que el tiempo de ejecución total del estudio es proporcional a la cantidad de muestras utilizadas. Esto se debe a que cada una de las muestras implica la resolución de un problema lineal. Por otro lado, se puede observar que, al disminuir la cantidad de muestras, la calidad de la solución al realizar simulaciones con crónicas históricas se ve afectada. Esto se debe a que al reducir la cantidad de muestras de hidrología a representar en el algoritmo SDDP, la cantidad de eventos de hidrología extrema disminuye. En caso de que no se capturen ciertos eventos de hidrología extrema, la política obtenida en términos de las funciones de Bellman es menos conservadora. Esto hace que el costo esperado de las simulaciones históricas (en donde aparecen todos los eventos extremos de la historia) sea mayor cuánto menos muestras se tomaron para hallar la política óptima, ya que al simular se utilizan incluso los eventos extremos. Teniendo en cuenta que se considera aceptable una diferencia relativa menor al 3 % respecto de la corrida de referencia en términos de simulación histórica, se puede concluir que, de las reducciones ensayadas, la primera que cumple el criterio es la de 60 %. Cabe aclarar que en el caso de referencia (color verde en la Tabla 6.3), la diferencia entre el costo simulado histórico y el valor en la función de valor aproximada se debe a que el proceso de aportes con el cual se obtiene la política óptima (proceso markoviano), no captura completamente ciertos eventos extremos. El costo simulado con el propio proceso markoviano en todos los casos es casi coincidente con el valor de la función de valor (convergencia del algoritmo SDDP),

Corrida	Tiempo de ejecución (h)	Valor en función de valor aproximada (MUSD)	Costo simulado histórico (MUSD)	Error relativo (%)
Caso base	31.1	164.7	171.3	0.0
Caso 60	18.9	161.3	176.1	2.8
Caso 60-40-20	13.8	161.1	176.2	2.9

Tabla 6.4: Resultados del análisis de disminución de realizaciones a través de los años.

por eso no se presenta.

Otra de las posibilidades para disminuir la cantidad total de problemas de despacho a resolver, que se exploró en este caso de estudio es la de reducir la cantidad de realizaciones de aleatoriedad, a medida que el paso de tiempo se acerca al fin. La Tabla 6.4 presenta los resultados en términos de costos anuales comparando: 1) el caso base (resaltado en verde), utilizando el 100 % de las muestras, 2) el caso en donde se toman todos los años con un 60 % de muestras y 3) un caso en el que el primer año se toma 60 %, el segundo año 40 % y el tercer año un 20 %. Se puede ver claramente que hay una reducción sensible en los tiempos de cómputo, proporcional a la cantidad de problemas lineales que se resuelven. A su vez, no existe un impacto significativo en los costos esperados (diferencias menores a 3 %).

6.6. Resultados: Caso de estudio 2

Para realizar este estudio se realizaron diversas corridas del algoritmo SDDP estándar (utilizando todas las muestras), modificando en cada una la técnica de poda de hiperplanos. Debido a que la poda de hiperplanos puede repercutir en la cantidad de iteraciones necesarias para obtener una solución buena, se utilizó como criterio de parada el criterio estándar definido por Pereira y Pinto [32] (solo en este caso de estudio). El mismo consiste en dar por terminada la corrida, cuando el gap existente entre el costo simulado y el costo dado por la aproximación de hiperplanos es menor a un umbral (se utilizó en este caso 3 %).

En la Tabla 6.5, se presenta para cada tipo de poda: la cantidad de iteraciones hasta convergencia, el tiempo de ejecución de un año de optimización, el tiempo máximo de resolución de un problema lineal y la cantidad máxima de hiperplanos presente en las aproximaciones de la función de valor al inicio de la corrida.

La principal conclusión que se puede obtener a partir de este caso de estudio, es que las técnicas de poda que se basan en mantener los últimos k hiperplanos agregados pueden no converger. Incluso en caso de convergencia para valores de k apropiados, la reducción que se obtiene en los tiempos de ejecución no resultan las mejores. Para el caso de las técnicas que mantienen los últimos k hiperplanos activos, se presentan mejores reducciones en los tiempos de cómputo. Sin embargo, es notoria la mejoría que se obtiene cuando se utilizan técnicas de nivel de dominancia, en particular la técnica que utiliza 1 nivel de dominancia (resaltada en verde en la Tabla 6.5), es la que presenta mejor desempeño y una reducción de casi 3 veces el tiempo de ejecución del modelo de referencia.

Tipo de poda	Iteraciones hasta convergencia (3%)	Tiempo de ejecución por año de corrida (h)	Tiempo máximo por problema (ms)	Cantidad máxima de hiperplanos por problema
Sin poda	2000	18.9	6.54	2000
10-últimos	No converge	---	0.00	10
20-últimos	No converge	---	0.00	20
30-últimos	2500	12.4	3.42	30
10-activos	2000	9.3	3.21	10
20-activos	2000	9.4	3.24	20
Nivel 1	1500	7.8	3.60	31
Nivel 3	1500	7.9	3.66	33

Tabla 6.5: Resultados del análisis de tiempos de ejecución según técnica de poda de hiperplanos.

	SDP	SDDP	Diferencias
Bonete	647.0	644.3	-2.7
Baygorria	424.6	420.7	-3.9
Palmar	1331.3	1330.3	-1.0
Salto	3831.8	3831.8	0.0
Térmica	2298.4	2300.1	1.7
Eólica	5020.1	5020.1	0.0
Fotovoltaica	463.0	463.0	0.0
Importación	1.2	1.3	0.1
Exportación	-2265.1	-2260.8	4.3
Falla	1.3	1.5	0.2
Demanda	11752.3	11752.3	0.0

Tabla 6.6: Energías esperadas (GWh) para cada componente del parque en el año 2023 del estudio para SDP y SDDP.

6.7. Resultados: Caso de estudio 3

Las Tablas 6.6 y 6.7 presentan el resultado de la comparación entre SDP y SDDP de energías esperadas y costos esperados, respectivamente. Se toma como referencia de comparación el año 2023 del estudio realizado, ya que no está afectado por los valores de fin de juego ni por las condiciones iniciales (la corrida abarca tres años completos, 2022, 2023 y 2024).

Por otra parte, en la Tabla 6.8 se puede observar que los resultados obtenidos resul-

	SDP	SDDP	Diferencias
Bonete	0.0	0.0	0.0
Baygorria	0.0	0.0	0.0
Palmar	0.0	0.0	0.0
Salto	0.0	0.0	0.0
Térmica	344.8	345.0	0.4
Eólica	0.0	0.0	0.0
Fotovoltaica	0.0	0.0	0.0
Importación	0.2	0.3	0.0
Exportación	-90.6	-90.4	0.2
Falla	0.3	0.4	0.0
Demanda	0.0	0.0	0.0
Costo Total	254.7	255.2	0.7

Tabla 6.7: Costos esperadas para el año 2023 en millones de USD para SDP y SDDP.

Algoritmo	**Tiempo de ejecución (horas)**
SDP	1.5
SDDP	23.4

Tabla 6.8: Tiempos de ejecución para SDP y SDDP en el caso de estudio.

tan prácticamente equivalentes en términos de costos y energías (diferencias inferiores al 1 %). Sin embargo, se observa que a pesar de los esfuerzos de reducción del tiempo de ejecución del algoritmo SDDP, el mismo es 15,6 veces mayor que el algoritmo SDP.

Para validar la operación del embalse principal del sistema, se presenta en la Fig. 6.5 la evolución de la cota del embalse de la central hidráulica Bonete en el caso promedio para 2023 y en la Fig. 6.6 la evolución de la cota del mismo embalse pero en la crónica de 1917 simulada para el año 2023, tanto para SDP como para SDDP. La crónica de 1917 fue la más seca de la historia, por lo tanto, es importante que el modelo reproduzca bien ese caso extremo. Se puede observar que el modelo SDDP opera el lago de Bonete casi de manera idéntica que el modelo SDP de referencia, tanto en el caso promedio como en la crónica extrema de 1917 simulada en 2023.

La similitud de la operación del embalse principal tanto en media como en la crónica extrema, en conjunto con la validación realizada de las energías y los costos esperados, permite concluir que el modelo implementado SDDP, resulta equivalente al SDP en cuanto a resultados.

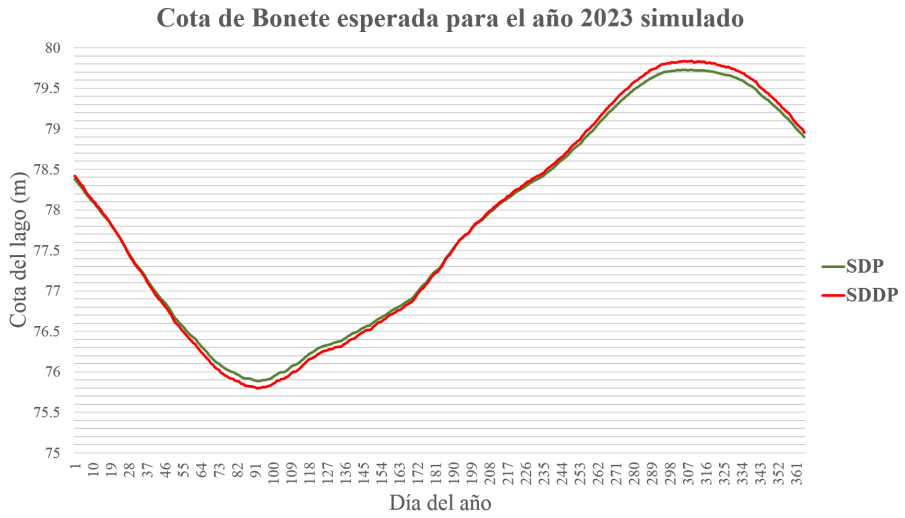


Figura 6.5: Comparación entre la operación del embalse de Bonete en valor esperado de la cota para el año 2023.

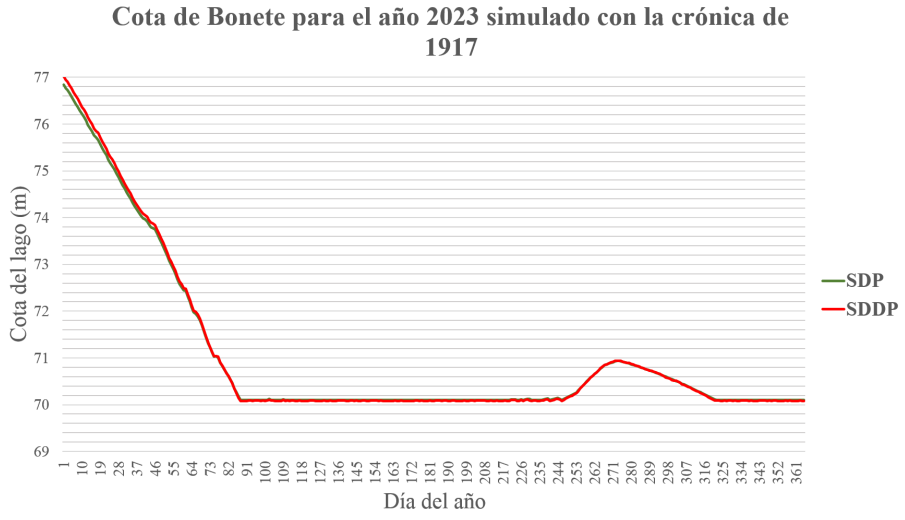


Figura 6.6: Comparación entre la operación de la cota del embalse de Bonete para el año 2023 simulado con la crónica hidráulica de 1917 (más seca de la historia).

El análisis del embalse de Bonete muestra que de la operación del mismo utilizando SDDP es muy similar que la obtenida en SDP. Esto constituye un resultado importante ya que SDP se utiliza en Uruguay desde hace años [11].

6.8. Conclusiones

En el caso de estudio 1 se logró obtener mediante la selección del 60 % de las muestras un resultado equivalente al que se obtiene utilizando el total. Esto permite lograr una reducción aproximada de un 40 % en los tiempos de cómputo.

Por otra parte, en el caso de estudio 2, se logra obtener una reducción significativa (casi la tercera parte) en los tiempos de cómputo, aplicando poda de hiperplanos. En particular, la de nivel de dominancia uno.

Al combinar ambas técnicas y compararlas con la SDP (Caso de estudio 3), se observa que los resultados obtenidos en términos energéticos y de costo son muy buenos. Sin embargo, se observa que el tiempo de ejecución es sensiblemente mayor en SDDP que en SDP.

Para explicar esta situación, hay que tener en cuenta dos factores. Por un lado, la cantidad de problemas lineales que se resuelve en cada uno de los algoritmos, y por otro lado el tiempo de ejecución de cada problema lineal resuelto.

Para la técnica SDP en cada paso se resuelve, para cada punto del espacio de estados, 5 problemas lineales. Considerando que la cantidad de estados se calcula como el producto cartesiano de las discretizaciones de las 4 variables de estado (Bonete, Palmar, Salto y variable hidrológica), se tiene que el total de estados es: $10 \times 5 \times 5 \times 5 = 1250$. Por lo tanto, en cada paso se resuelven $1250 \times 5 = 6250$ problemas lineales.

La resolución SDDP utiliza un 60 % de realizaciones para cada Markov para cada paso, o sea 12 realizaciones. En total son 5 estados markovianos y se realizan 2000 iteraciones para obtener un resultado preciso. Por lo tanto, se resuelven $12 \times 5 \times 1500 = 90000$ problemas lineales por paso.

De esta manera, la cantidad de problemas lineales que se resuelven en SDDP por paso es casi 20 veces mayor en SDP (14,4 veces mayor). A su vez, el tiempo de ejecución de SDDP es 15,6 veces mayor que en SDP. Esto permite concluir que hay un factor adicional de incremento en los tiempos de ejecución en SDDP, que obedece a la complejidad del propio problema lineal. Esto se explica por el hecho de que en SDDP se representa la función de costo futuro mediante un conjunto de restricciones (una adicional por cada hiperplano), mientras que en SDP el costo futuro se considera sin adicionar restricciones.

De lo anterior expresado, se puede concluir que la mayor fortaleza del algoritmo SDDP, respecto del algoritmo SDP, se presenta al resolver problemas en los que la dimensionalidad del espacio de estados es grande, como se vio en el Capítulo 4. Sin embargo, los resultados respecto a mitigar la maldición de la dimensionalidad estocástica resultan limitados en problemas con pocas variables de estado. El potencial del algoritmo SDDP se manifiesta en sistema con decenas o cientos de variables de estado. Es posible que, en Uruguay, a medida que se incorporen nuevos recursos al sistema y se manifieste la maldición de la dimensionalidad de estados (o de Bellman) en el algoritmo SDP, el algoritmo SDDP se presente como una clara alternativa al algoritmo SDP.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 7

Conclusiones y trabajo a futuro

El objetivo del presente trabajo es el de analizar el algoritmo SDDP aplicado a la solución del problema de operación de un sistema eléctrico. La necesidad creciente en los sistemas eléctricos de modelar tecnologías más avanzadas, requiere abordar el compromiso entre lograr un modelo lo suficientemente detallado y mantener tiempos de cómputo razonables. Una de las limitaciones más grandes de las técnicas habituales como la SDP para resolver el problema de operación, es que incurren en la llamada maldición de la dimensionalidad del espacio de estados. La técnica de SDDP tiene la fortaleza de mitigar esta maldición, pero a cambio de esto presenta también algunas limitaciones. Una de estas limitaciones es que requiere que la función de costo futuro sea convexa. Esto no representa un compromiso mayor, ya que estructurar el costo futuro mediante una función convexa a la hora del modelado del problema de operación de un sistema eléctrico resulta conveniente y no requiere grandes simplificaciones. Otra de las limitaciones es que, a pesar de mitigar la maldición de la dimensionalidad de estados, incurre en la maldición de la dimensionalidad estocástica. Por último, la aproximación de la función de valor mediante hiperplanos, implica aumentar el tamaño de los problemas de programación lineal a resolver. Estas limitaciones repercuten en los tiempos de ejecución del algoritmo. Lo deseable es lograr una mejora en el modelado a través del aumento del espacio de estados sin comprometer los tiempos de cómputo y la calidad de la solución.

En el Capítulo 2 se presenta en términos generales el algoritmo SDDP. A la hora de modelar adecuadamente un sistema hidrotérmico, se requiere un modelo de aportes hidráulicos a las cuencas que contemple la situación climatológica. A su vez, se requieren algunos ajustes al algoritmo general para considerar recursos que operan en diferentes escalas de tiempo. Por lo tanto, en el Capítulo 3 se especifica un modelo de Markov para la representación de aportes y se presentan una serie de agregados al modelo base para considerar múltiples escalas. Luego, utilizando lo presentado, se analiza en el Capítulo 4 el uso de la técnica SDDP en un modelo estilizado del sistema uruguayo y el impacto de incorporar variables de estado para representar nuevos componentes en el sistema. Para esto se utilizó una biblioteca SDDP existente, desarrollada en el lenguaje Julia. Luego, ante la necesidad de explorar simplificaciones o mejoras en el algoritmo, que la rigidez de la biblioteca utilizada no permite incorporar, se desarrolló una biblioteca de SDDP en Python específica para la aplicación a sistemas eléctricos. Esta biblioteca se describe en el Capítulo 5 y se especifica en el Apéndice A. Finalmente, utilizando la biblioteca desarrollada, se aborda en el Capítulo 6 el problema de reducción de tiempos de cómputo a través de la mitigación de la maldición de la dimensionalidad estocástica. Para ello se propone un abordaje que se analiza a través

de un caso de estudio que incorpora técnicas estándar de poda de hiperplanos.

En el análisis realizado en el modelo uruguayo presentado en el Capítulo 4 incorporando un modelo markoviano de aportes, la principal contribución fue incorporar en el marco de SDDP una escala de tiempo detallada. Esta escala permite modelar los efectos de corto plazo, lo que se hace necesario para capturar la variabilidad de las fuentes renovables y medir adecuadamente el impacto de incorporar el almacenamiento de energía a corto plazo en el sistema. Los resultados obtenidos muestran tiempos de cómputo prometedores, particularmente cuando se amplía el espacio de estados, lo que abre la puerta a incorporar más detalles al modelo. Sin embargo, para un sistema con un espacio de estados relativamente pequeño, la maldición de la dimensionalidad estocástica se convierte en limitante. Como consecuencia de esto, se decide realizar una serie de modificaciones al algoritmo base. Ante la poca flexibilidad para realizarlas en la biblioteca utilizada, se desarrolla en Python, una biblioteca específica de SDDP aplicada al sistema eléctrico.

El Capítulo 5 presenta la biblioteca desarrollada. La principal contribución de la biblioteca consiste en la posibilidad de incorporar de forma sencilla los componentes habituales de un sistema eléctrico: generadores hidráulicos encadenados, generadores térmicos, generadores eólicos, generadores fotovoltaicos, demandas eléctricas, intercambios internacionales de energía, entre otros. Asimismo, contempla diversas configuraciones para la ejecución del algoritmo SDDP, incorporando la posibilidad de simular con crónicas históricas de aportes hidrológicos. Por otra parte, al ser desarrollada utilizando un paradigma orientado a objetos, resulta fácilmente escalable, lo que brinda un gran potencial en el caso de ser necesario incorporar nuevos componentes al parque eléctrico o nuevos modelos para contemplar la incertidumbre. Queda pendiente como trabajo a futuro, poner a disposición la biblioteca a través de un repositorio con la documentación adecuada para el uso público. Esto quedó fuera del alcance de este trabajo debido a que se hizo hincapié en el uso de la misma específicamente para los estudios realizados en la tesis.

El Capítulo 6 aborda el problema de reducción de los tiempos de ejecución, atacando dos puntos fundamentales. La maldición de la dimensionalidad estocástica y el empleo de técnicas estándar de poda de hiperplanos. Se exploran diferentes formas de reducción de la cantidad de muestras a considerar a través de un caso de estudio. A pesar de que se logró una reducción en los tiempos de cómputo no despreciable manteniendo la calidad de la solución hallada, los resultados obtenidos para el sistema uruguayo en términos de tiempos de ejecución no resultaron satisfactorios, confirmando que el potencial del algoritmo SDDP se manifiesta en sistemas con decenas o cientos de variables de estado.

Quedan abiertas ciertas líneas de trabajo futuro dentro del marco del algoritmo SDDP: una importante es explotar el hecho de que la técnica es altamente paralelizable, una característica que no se explotó en este trabajo. Esto mejoraría drásticamente los tiempos de cálculo de la optimización. Una segunda línea de trabajo consiste en incluir consideraciones de red eléctrica, agregando sus restricciones, que generalmente se omite debido a la complejidad computacional, pero con estas técnicas ahora parece estar al alcance. Sin embargo, la principal línea de trabajo futuro es la de explorar algoritmos enmarcados en lo que se denomina Programación Dinámica Aproximada (ADP) o alguna variante de aprendizaje por refuerzos [28, 35, 39, 39, 40], en particular las que solamente realizan pasadas hacia adelante (simulaciones) y utilizan diversos tipos de estructura de aproximación global de la función de costo futuro. Este tipo de técnicas, a pesar de no tener un fuerte sustento matemático, reúnen todas las cualidades deseables para resolver el problema de operación del sistema eléctrico. En primer lugar, ADP es muy apropiada para mitigar la maldición de la dimensionalidad del espacio de estados, ya que utilizando algún aproximador global (como ser redes neuro-

nales, funciones de base radial, etc.) no requiere discretización del espacio de estados. En segundo lugar, no presenta el problema de la maldición de la dimensionalidad estocástica, ya que la distribución de probabilidad conjunta de las variables aleatorias, se aprende a medida que se van simulando diferentes trayectorias, evitando el problema de divergencia que sucede en SDDP debido a la estructura de hiperplanos de la aproximación de la función de Bellman. En tercer lugar, no requiere hipótesis de convexidad, ya que la estructura de los aproximadores a utilizar no lo exige. Por último, la comunidad de desarrollo asociada a este tipo de técnicas está muy avanzada, por lo que existen infinidad de herramientas de muy buena calidad que pueden ser empleadas para este tipo de desarrollos. El inconveniente que surge al aplicar técnicas de ADP, es el de introducir conocimiento a priori de forma de lograr una convergencia más rápida. A su vez, presenta dificultades a la hora de explorar un espacio de control continuo. Abordar estas limitantes representa un gran desafío, en particular para la solución del problema de operación de un sistema eléctrico, por lo que es una línea sólida de trabajo futuro que ataca todas las limitantes de las técnicas analizadas en esta tesis. Dentro de las extensiones o incorporaciones a realizar a la biblioteca desarrollada, se presenta una fuerte línea de trabajo vinculada con el desarrollo de otros tipos de algoritmos de programación dinámica estocástica, como por ejemplo ADP.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice A

Biblioteca SDDP para la resolución del problema de operación de un sistema eléctrico.

En el presente apéndice se especifica el paquete desarrollado en Python que permite resolver el problema de operación de un sistema eléctrico. Dentro de esta especificación se hace especial hincapié en los módulos autocontenidos que pueden ser utilizados de manera independiente. De ninguna manera pretende ser una especificación completa de la biblioteca o un manual de usuario, sino que simplemente describe lo fundamental de cada componente de software, para finalizar con un ejemplo simple de uso. El desarrollo de esta biblioteca se realizó pensando en el problema que se analizó a lo largo de la tesis, por lo que no constituye en sí un producto independiente y totalmente portable. Sin embargo, este desarrollo respeta las líneas esenciales del desarrollo de software, por lo que se puede adaptar su uso a otro tipo de problemas con cierto esfuerzo de desarrollo.

A.1. Modelo del parque generador y componentes de incertidumbre

Dentro del modelo del sistema eléctrico se deben considerar tanto los componentes asociados al sistema (*participantes*) como los procesos estocásticos que permiten manejar la incertidumbre en el marco del problema a resolver. La solución informática diseñada desacopla la construcción del parque generador y sus componentes, de la construcción del modelo que contempla la incertidumbre. Este desacople se implementa a través de variables aleatorias. Cada valor asociado a un *participante* del parque generador que posea incertidumbre se representa como una variable aleatoria. A su vez, cada variable aleatoria puede ser realizada respetando su distribución. Para cumplir este cometido las variables aleatorias pertenecen a procesos estocásticos. Naturalmente, a la hora de diseñar el modelo estocástico se debe tener en cuenta que las variables aleatorias que tengan una correlación relevante entre sí, deben ser consideradas de forma conjunta dentro de un proceso estocástico.

Por otra parte, tanto los *participantes* como los procesos estocásticos pueden contener variables que representen estado en el problema dinámico de operación que se va a resolver. Un claro ejemplo de una variable de estado asociada a un participante,

es el volumen del embalse de una central hidráulica. Para un proceso estocástico, un ejemplo de variable de estado es la variable hidrológica de un proceso markoviano de aportes, tal como se definió en el Capítulo 3, que representa al estado hidrológico de la naturaleza.

A.1.1. Parque generador y componentes del sistema eléctrico

Para modelar el sistema eléctrico se debe considerar el parque generador y otros componentes necesarios. Entendemos como *participantes* a la unión de estos dos conjuntos. Los *participantes* que pueden incorporarse al sistema con la biblioteca desarrollada son: generadores, intercambios internacionales, acumuladores y demandas.

Dentro de los generadores implementados se encuentran:

- *Generador Hidráulico:* Representa un generador hidráulico que puede ser modelado con o sin embalse (de pasada). En el caso de que el modelo tiene embalse, el volumen del mismo es una variable de estado del sistema. La biblioteca permite encadenar centrales hidráulicas de manera de representar las cuencas hidrográficas.
- *Generador Térmico:* Representa un generador térmico, el modelo implementado no considera los mínimos técnicos.
- *Generador Eólico:* Representa un generador eólico. A partir de una variable aleatoria que representa el factor de planta y la potencia instalada se obtienen realizaciones de la curva de generación.
- *Generador Fotovoltaico:* Representa un generador fotovoltaico. A partir de una variable aleatoria que representa el factor de planta y la potencia instalada se obtienen realizaciones de la curva de generación.

Por otra parte, se modelaron intercambios internacionales de forma muy simple. Tanto para la importación como para la exportación se define una potencia máxima y un precio de compra o venta de energía, respectivamente.

Un caso especial de *participante* implementado en la biblioteca es el del acumulador. Puede actuar como un generador si tiene energía almacenada o como demanda para cargarse. Se modeló con diferentes rendimientos para la carga y la descarga. Por último, se desarrolló el *participante* que representa una demanda. El mismo se modela a través de una variable aleatoria que representa la demanda horaria, por lo que al igual que los generadores eólicos y solares deben estar asociados a un proceso estocástico que produzca realizaciones de las variables aleatorias.

A.1.2. Incertidumbre: Modelo de procesos estocásticos

Las variables aleatorias presentes en el sistema son parte de un proceso estocástico. Cada proceso estocástico tiene la capacidad de producir realizaciones de las variables aleatorias que contiene. Si el proceso estocástico tiene estado, el mismo puede utilizar la información de estado para condicionar las realizaciones de las variables aleatorias.

La biblioteca posee un módulo llamado *Azar*, que contiene la colección de procesos estocásticos del sistema. Cada proceso estocástico implementa una interfaz que posee una operación que recibe un instante de tiempo y produce una realización de las variables aleatorias.

A.1.3. Modelo completo

El modelo completo está compuesto por dos grandes componentes: el parque y el azar. Dentro del parque se encuentran todos los *participantes* incorporados por el

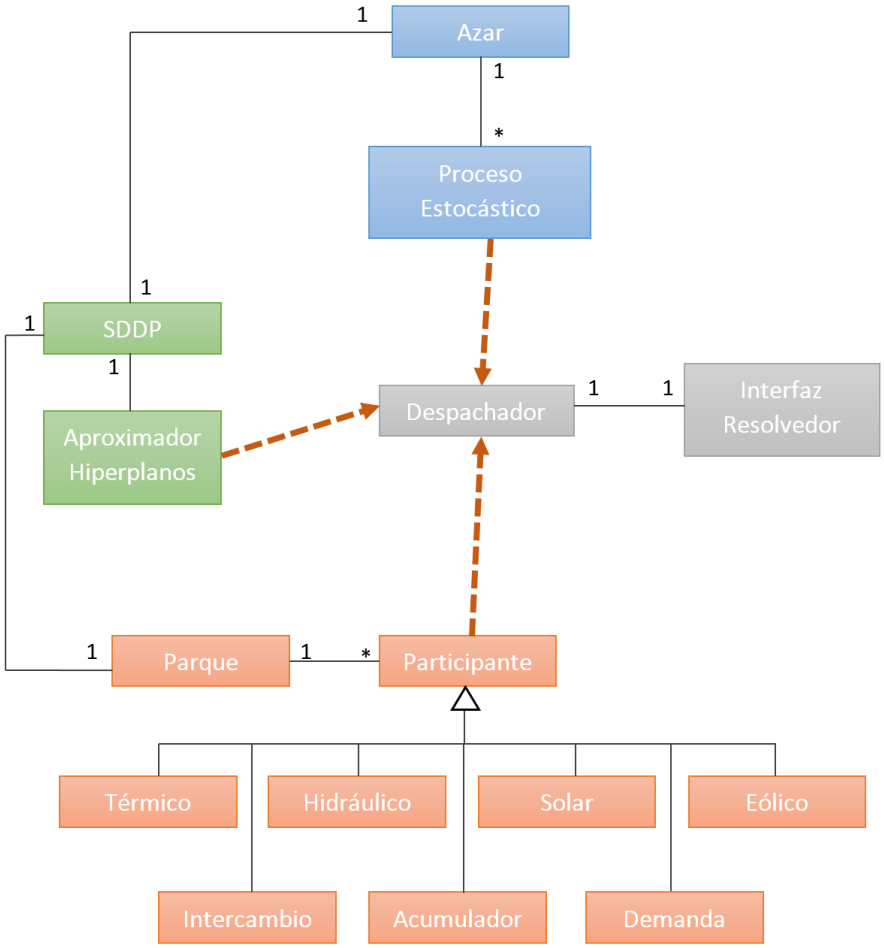


Figura A.1: Modelo conceptual de la estructura de resolución del algoritmo SDDP.

usuario al modelo definidos en la Sección A.1.1. El componente azar corresponde a la colección de procesos estocásticos del sistema. Ambos componentes se vinculan a través de las variables aleatorias que pertenecen tanto a los *participantes* como a los procesos estocásticos. La Figura A.1 representa de forma genérica la estructura del modelo completo. Se puede observar que los componentes del parque, los procesos estocásticos y la aproximación de la función de valor por hiperplanos aportan al despachador (flechas punteadas) la información necesaria para la construcción de cada problema lineal. El despachador utiliza un resolvedor particular a través de la interfaz genérica para la resolución del problema.

A.2. Resolución SDDP

A la hora de resolver el problema de operación asociado a determinado sistema eléctrico definido como fue descrito en la Sección A.1, se deben incorporar los parámetros asociados a la propia resolución del problema. Para ello se definió un módulo

llamado *SDDP*, que permite especificar los parámetros necesarios para implementar la resolución. Estos son:

- *Parque:* Una vez que se crean cada uno de los *participantes* que forman parte del modelo, se agregan a un componente que representa el parque. Dicho componente es uno de los parámetros del resolvedor *SDDP*.
- *Azar:* Cada uno de los procesos estocásticos a utilizar en el sistema, se deben incorporar a la colección contenida en el módulo *Azar*. Este módulo es parámetro del resolvedor *SDDP* .
- *Horizonte de tiempo:* Representa la cantidad de pasos de tiempo que se quiere resolver.
- *Despachador:* Es el módulo que se encarga de resolver los problemas lineales asociados a un paso de tiempo, en la Sección A.2.2 se describe el módulo.
- *Cantidad de realizaciones a producir en cada punto de la trayectoria:* Representa la cantidad de realizaciones aleatorias que se producen en cada punto del espacio de estados en la pasada para atrás.
- *Otros parámetros:* Se introducen una serie de parámetros correspondientes a la estrategia de muestreo, el criterio de parada y los parámetros relativos a la poda de hiperplanos.

Se describen en las siguientes secciones los componentes auxiliares del resolvedor SDDP más relevantes.

A.2.1. Aproximador de la función de Bellman

La biblioteca utiliza un módulo denominado *AproximacionHiperplanos* de forma de encapsular la aproximación de la función de Bellman de un paso en particular a través de hiperplanos. Este módulo posee una colección de hiperplanos que representan la aproximación de la función. Además, maneja las funcionalidades que permiten realizar la poda de estos hiperplanos con el objetivo de minimizar las restricciones al problema lineal. Por otra parte, maneja la generación de las restricciones y el aporte al objetivo del problema lineal para comunicar al componente despachador colaborando con la construcción del problema de despacho de un paso.

A.2.2. Despachador, mecanismo de construcción del problema lineal

De acuerdo al algoritmo *SDDP* presentado en el Capítulo 2, tanto en la pasada hacia atrás como en la pasada hacia adelante se requiere construir y resolver lo que se conoce como el problema de despacho. En *SDDP* este problema de optimización se aborda mediante programación lineal. En la biblioteca desarrollada, el módulo *Despachador* es el encargado de construir el problema de despacho con estructura lineal cada vez que se requiere.

El módulo *Despachador* contiene una variable que refiere a un resolvedor lineal genérico (a través de la interfaz definida en la Sección A.2.3. Para construir el problema, este módulo recorre todos los *participantes* del parque y les solicita una serie de tareas. En primer lugar, les solicita que devuelvan las variables de control que van a utilizar. En segundo lugar, les solicita que devuelvan las ecuaciones de las restricciones a agregar el problema lineal. Y en tercer lugar les solicita que aporten términos al objetivo del problema. A su vez, como en el problema lineal interviene la aproximación futura, el despachador le solicita al aproximador de la función de Bellman que aporte sus variables de control, sus restricciones y su contribución al objetivo.

Suponiendo un sistema simple con una central hidráulica y una central térmica cuando se le solicita al objeto generador hidráulico la información que aporta al problema lineal, el mismo devuelve como variables de control su turbinado y su vertido con las cotas de caja correspondientes. Devuelve como restricción el balance hidráulico de la central, y no aporta costo al objetivo. Sin embargo, cuando se le solicita al objeto generador térmico la información, el mismo devuelve como variable de control la potencia inyectada y sus cotas de caja, no aporta restricciones y su contribución al objetivo resulta del costo en $\frac{USD}{MWh}$ asociado a la generación. Por otra parte, como el espacio de estados solamente está compuesto por el volumen del embalse de la central, el aproximador de la función de Bellman aporta una serie de hiperplanos en ese espacio como restricciones y una variable de control para representar la función como contribución al objetivo. Por último, el despachador genera la restricción de balance de demanda eléctrica, para la que utiliza las variables de control que ya fueron aportadas por los *participantes*.

A.2.3. Interfaz con el resolvedor lineal

Uno de los objetivos de la biblioteca desarrollada, es que sea fácilmente escalable. Un punto extremadamente importante en este sentido radica en tener la posibilidad de modificar el resolvedor lineal que se utiliza, o incluso utilizar varios en simultáneo. A pesar de que en este trabajo se implementó solamente la interfaz con el resolvedor Gurobi, se diseñó una interfaz genérica para poder contemplar la implementación de instancias con cualquier resolvedor. En la Sección A.3.6 se detallan los métodos de la interfaz.

A.2.4. Simulador

El componente *Simulador*, es el encargado de simular el sistema utilizando las aproximaciones de las funciones de Bellman halladas (que definen una política), de acuerdo a las directivas del usuario. Fueron implementadas cuatro formas de simular las variables aleatorias del sistema: 1) con crónicas históricas, 2) utilizando los procesos estocásticos del azar con los que se obtuvo la política, 3) sorteando de manera independiente las variables aleatorias en cada paso y 4) utilizando una política miope (la función de costo futuro es idénticamente nula en todos los pasos).

Por otra parte, dentro de las funcionalidades que ofrece el componente simulador, se encuentran:

- *Guardar los escenarios simulados:* Se almacenan los resultados de todos los escenarios, esto es, los valores del control óptimo para cada una de las variables de control y la evolución de la dinámica de estados.
- *Obtención de resultados relevantes:* Se obtiene el costo medio de simulación total, las energías medias por paso y por generador, los costos medios por paso, entre otros resultados.
- *Visualización de resultados:* Despliega gráficos asociados a los datos relevantes.

Es importante tener en cuenta que, a este módulo, al tener encapsulada toda la información relativa a la salida de las simulaciones ejecutadas, es sencillo incorporarle funcionalidades de post-procesamiento de resultados.

A.3. Especificación de software

En esta sección se especifican las funciones asociadas a cada componente de la biblioteca, describiendo sus parámetros y el retorno.

A.3.1. Clase Parque

```
def init(hydros, termicos, eolicos, solares, intercambios, demandas)
```

Descripción: Constructor de la clase Parque, recibe como parámetro:

Parámetros:

- hydros: una lista de generadores hidráulicos
- termicos: una lista de generadores térmicos
- eolicos: una lista de generadores eólicos
- solares: una lista de generadores solares
- intercambios: una lista de intercambios internacionales (importación o exportación)
- demandas: una lista de demandas

Cada uno de los objetos pertenecientes a la lista que recibe el parque, debe construirse previamente utilizando las clases descriptas asociadas a los componentes del parque que se describen más adelante.

```
def construirProblemaDespacho()
```

Descripción: Construye un problema de despacho de acuerdo a la situación actual del paso en cuestión, para ello le solicita a cada componente del parque sus variables, sus restricciones y su aporte al objetivo. Utiliza al despachador para construir el problema y aporta como restricción fundamental del parque, la de balance de demanda.

Retorno: Un objeto que contiene la descripción del problema asociada al resolvedor lineal que se encuentre presente.

```
def actualizarEstado(descripcionEstado)
```

Descripción: Actualiza el estado de todos los componentes del parque de acuerdo al parámetro que representa el nuevo estado.

Parámetros:

- descripcionEstado: diccionario que tiene como clave el nombre de una variable de estado y como valor el nuevo estado que debe impactarse en dicha variable.

```
def dameVarsEstado()
```

Descripción: Devuelve el conjunto de variables de estado del sistema.

Retorno: Lista con las variables de estado de todos los componentes del parque.

Clase Hidráulico

```
def init(nombre, eIni, vaAporte, volMax, turMax, verMax, coefEnergetico)
```

Descripción: Constructor de la clase Hidráulico.

Parámetros:

- nombre: es el nombre de la central.
- eIni: el estado inicial, volumen del embalse.
- vaAporte: variable aleatoria que modela el aporte de la central.
- volMax: volumen máximo del embalse en hm^3.
- turMax: turbinado máximo que soporta la central en m^3/s.
- verMax: vertimiento máximo que soporta la central m^3/s.
- coefEnergetico: coeficiente energético de la central en $\frac{MW}{m^3/s}$, se considera constante.

```
def agregarAguasArriba(hidro)
```

Descripción: Agrega un generador hidráulico aguas arriba de la central.

Parámetros:

- hidro: Generador hidráulico que se encadena aguas arriba.

```
def totalAguasArriba()
```

Descripción: Devuelve el total de agua que llega a la central, producto del erogado de las centrales que están aguas arriba.

Retorno: Suma de los erogados de todas las centrales que están inmediatamente aguas arriba de la central.

```
def dameVarsEstado()
```

Descripción: Devuelve la colección de variables de estado, en este caso el volumen del embalse.

Retorno: Colección de variables de estado del generador hidráulico.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de estado y de control

Retorno: Diccionario con las variables de estado y de control del generador hidráulico.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el generador hidráulico al problema de despacho.

Retorno: Diccionario con las restricciones del generador hidráulico en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del generador hidráulico al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Térmico

```
def init(nombre,potMax,precio)
```

Descripción: Constructor de la clase Térmico.

Parámetros:

- nombre: es el nombre de la central.
- potMax: es la potencia máxima en MW que puede inyectar al sistema el generador térmico.
- precio: precio de la energía generada en $\frac{USD}{MWh}$.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control del generador térmico.

Retorno: Diccionario con las variables de control del generador térmico.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el generador térmico al problema de despacho.

Retorno: Diccionario con las restricciones del generador térmico en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del generador térmico al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Eólico

```
def init(nombre,potInstalada,precio,factor)
```

Descripción: Constructor de la clase Eólico.

Parámetros:

- nombre: es el nombre de la central.
- potInstalada: es la potencia instalada en MW que puede inyectar al sistema el generador eólico.
- precio: precio de la energía generada en $\frac{USD}{MWh}$.
- factor: variable aleatoria que es el factor de planta.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control del generador eólico.

Retorno: Diccionario con las variables de control del generador eólico.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el generador eólico al problema de despacho.

Retorno: Diccionario con las restricciones del generador eólico en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del generador eólico al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Solar

```
def init(nombre, potInstalada, precio, factor)
```

Descripción: Constructor de la clase Solar.

Parámetros:

- nombre: es el nombre de la central.
- potInstalada: es la potencia instalada en MW que puede inyectar al sistema el generador solar.
- precio: precio de la energía generada en $\frac{USD}{MWh}$.
- factor: variable aleatoria que es el factor de planta.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control del generador solar.

Retorno: Diccionario con las variables de control del generador solar.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el generador solar al problema de despacho.

Retorno: Diccionario con las restricciones del generador solar en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del generador solar al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Intercambio

```
def init(nombre,potMax,precio)
```

Descripción: Constructor de la clase Intercambio.

Parámetros:

- nombre: es el nombre de la central.
- potMax: es la potencia máxima en valor absoluto en MW que puede intercambiar con el sistema el componente intercambio. Potencia positiva es importación y negativa exportación.
- precio: precio de la energía intercambiada en $\frac{USD}{MWh}$

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control del componente intercambio.

Retorno: Diccionario con las variables de control del componente intercambio.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el componente intercambio al problema de despacho.

Retorno: Diccionario con las restricciones del componente intercambio en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del componente intercambio al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Acumulador

```
def init(nombre, potMax, almacMax, rendCarga, rendDescarga, almacIni)
```

Descripción: Constructor de la clase Acumulador.

Parámetros:

- nombre: es el nombre de la central de acumulación.
- potMax: es la potencia máxima en valor absoluto en MW que puede inyectar o tomar del sistema. Potencia positiva es inyección y negativa es almacenamiento.
- almacMax: es la energía máxima en MWh que puede almacenar el acumulador.
- rendCarga: es el rendimiento (adimensionado) en la carga del acumulador.
- rendDescarga: es el rendimiento (adimensionado) en la inyección del acumulador.
- almacIni: es la energía incial en MWh presente en el acumulador.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control del componente acumulador.

Retorno: Diccionario con las variables de control del componente acumulador.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el componente acumulador al problema de despacho.

Retorno: Diccionario con las restricciones del componente acumulador en el problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del componente acumulador al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

Clase Demanda

```
def init(nombre, vaDemanda, costoFalla)
```

Descripción: Constructor de la clase Demanda.

Parámetros:

- nombre: es el nombre de la demanda.
- vaDemanda: variable aleatoria que permite modelar la demanda.
- costoFalla: es el costo en $\frac{USD}{MWh}$ al que se incurre por no abastecer la demanda.

```
def dameVariables()
```

Descripción: Devuelve el conjunto de todas las variables de control de la demanda.

Retorno: Diccionario con las variables de control del demanda.

```
def dameRestricciones()
```

Descripción: Devuelve el conjunto de restricciones que aporta el demanda al problema de despacho.

Retorno: Diccionario con las restricciones del demanda al problema de despacho.

```
def aporteObjetivo()
```

Descripción: Devuelve la contribución al objetivo del demanda al problema de despacho.

Retorno: Expresión lineal a agregar al objetivo del problema de despacho.

A.3.2. Clase Azar

La clase azar es la encargada de manejar la incertidumbre. En esencia cuenta con una colección de procesos estocásticos y un método que los realiza en cada paso de tiempo.

```
def init(procesos)
```

Descripción: Constructor de la clase Azar.

Parámetros:

- procesos: es la de procesos estocásticos del sistema

Clase Proceso Estocástico

La clase proceso estocástico implementa una interfaz con el método *producirRealizacion*, tiene una colección de variables aleatorias y eventualmente una colección de variables de estado. Es responsabilidad del usuario construir los procesos estocásticos con sus variables aleatorias de forma de ser utilizadas por componentes del parque.

```
def producirRealizacion(paso)
```

Descripción: Produce una realización de las variables de aleatorias del sistema, dado el paso actual.

Parámetros:

- paso: es el paso de tiempo para el que se quiere producir una realización.

A.3.3. Clase SDDP

La clase SDDP es la fundamental a la hora de resolver el problema de operación del sistema eléctrico.

```
def init(parque, azar, despachador, descripcionHorizonte, maxIter, tipoPoda)
```

Descripción: Constructor de la clase SDDP.

Parámetros:

- parque: un objeto de la clase Parque, con el que se resolverá el problema de operación.
- azar: objeto de la clase Azar conteniendo todos los procesos estocásticos a utilizar en la resolución del problema de operación.
- descripcionHorizonte: objeto que define la fecha de inicio y fin del problema a resolver, así como la duración del paso.
- maxIter: cantidad máxima de iteraciones en las que se detiene el algoritmo SDDP.
- tipoPoda: estrategia de poda de hiperplanos a elegir

```
def optimizar()
```

Descripción: Optimiza el objeto SDDP.

Retorno: Devuelve una lista de objetos AproximacionHiperplanos representando la función de valor hallada óptima para cada paso de tiempo.

```
def simular(cantSimulaciones, tipoSimulacion)
```

Descripción: Simula una cantidad de escenarios utilizando definida por el usuario y devuelve la información de la operación del parque generador para cada escenario.

Retorno: Devuelve una lista de objetos AproximacionHiperplanos representando la función de valor hallada óptima para cada paso de tiempo.

A.3.4. Clase AproximacionHiperplanos

```
def init(dimension, tipoPoda)
```

Descripción: Constructor de la clase SDDP.

Parámetros:

- dimension: es la dimensión del espacio de estados.
- tipoPoda: indica que tipo de técnica de poda se utilizará.

```
def dameVariables()
```

Descripción: Devuelve las variables a aportar al problema lineal, solamente una variable auxiliar para acotar con los hiperplanos.

Retorno: Variable de estado auxiliar para acotar con los hiperplanos.

```
def dameRestricciones()
```

Descripción: Devuelve las restricciones a incorporar al problema lineal.

Retorno: Colección de restricciones asociadas a los hiperplanos.

```
def aportarObjetivo()
```

Descripción: Devuelve la expresión lineal que se adiciona al objetivo del problema de despacho.

Retorno: Expresión lineal a aportar al objetivo.

```
def evaluar(x)
```

Descripción: Devuelve la evaluación de la función de Bellman en el punto x.

Parámetros:

- x: es el valor de estado a evaluar según la aproximación por hiperplanos.

Retorno: Número real resultado de la evaluación.

```
def agregarCorte(hiperplano)
```

Descripción: Agregar un nuevo hiperplano a la aproximación de Bellman actual.

Parámetros:

- hiperplano: vector que representa el hiperplano a agregar.

```
def graficarCortes()
```

Descripción: Grafica la aproximación por hiperplanos.

A.3.5. Clase Despachador

```
def init(resolvedor)
```

Descripción: Constructor de la clase Despachador.

Parámetros:

- resolvedor: Es el objeto resolvedor lineal al que invocará el despachador para resolver los problemas de despacho.

```
def construirProblema(nombre, variables, restricciones, objetivo)
```

Descripción: Construye un problema dadas las variables, las restricciones y el objetivo, lo mantiene como un atributo del objeto invocado.

Parámetros:

- nombre: es la dimensión del espacio de estados.
- variables: variables a utilizar en la construcción del problema.
- restricciones: conjunto de restricciones lineales a utilizar en la construcción del problema.
- objetivo: expresión lineal que representa el objetivo del problema.

```
def despachar()
```

Descripción: Resuelve el problema actual del despachador utilizando el resolvedor lineal de la clase.

```
def escribirProblema(ruta)
```

Descripción: Escribe en disco el problema de despacho actual de la clase.

Parámetros:

- ruta: ruta en donde se quiere escribir el problema.

```
def obtenerSolucion(variables)
```

Descripción: Dadas las variables a través de un parámetro, se les asigna el valor óptimo hallado por el problema ya resuelto.

Parámetros:

- variables: lista de variables a las que se le quiere asignar la solución óptima.

A.3.6. Clase InterfazResolvedor

```
def construirProblema(nombre, variables, restricciones, objetivo)
```

Descripción: Construye un problema dadas las variables, las restricciones y el objetivo, lo mantiene como un atributo del objeto invocado.

Parámetros:

- nombre: es la dimensión del espacio de estados.
- variables: variables a utilizar en la construcción del problema.
- restricciones: conjunto de restricciones lineales a utilizar en la construcción del problema.
- objetivo: expresión lineal que representa el objetivo del problema.

```
def agregarObjetivo(modelo, objetivo)
```

Descripción: Agrega el objetivo al modelo.

Parámetros:

- modelo: problema lineal.
- objetivo: expresión lineal que representa el objetivo del problema.

```
def agregarRestricciones(modelo, restricciones)
```

Descripción: Agrega la colección de restricciones al modelo.

Parámetros:

- modelo: problema lineal.
- objetivo: conjunto de expresiones lineales que son las restricciones a arreglar al problema.

```
def optimizar(modelo)
```

Descripción: Optimiza el modelo que recibe como parámetro. La solución se asigna en el mismo objeto modelo.

Parámetros:

- modelo: problema lineal.

```
def escribirProblema(problema, ruta)
```

Descripción: Escribe en disco el problema de despacho actual de la clase.

Parámetros:

- problema: es el problema que se quiere escribir a disco.
- ruta: ruta en donde se quiere escribir el problema.

A.4. Ejemplo de resolución

En este capítulo se describieron superficialmente los componentes esenciales del desarrollo realizado en el marco de la biblioteca SDDP. En esta sección se presenta la resolución de un problema de operación extremadamente sencillo, de forma de cubrir en todos los aspectos el abordaje en software de esta solución.

Se definirá un parque conformado por una central hidráulica, una central térmica y una demanda. Los aportes a la central hidráulica y la demanda del sistema son muestreados a partir de procesos estocásticos. Se asume para simplificar, que los procesos estocásticos están ya estimados y se levantan desde un archivo, pero no se conoce su estructura (caja negra). A continuación, se presenta el código y luego se describe cada fragmento.

```
1   procesoAportes = ProcesoEstocasticoAportes(rutaAportes)
2   procesoDemanda = ProcesoEstocasticoDemanda(rutaDemanda)

3   azar = Azar()
4   azar.agregarProcesoEstocastico("aportes", procesoAportes)
5   azar.agregarProcesoEstocastico("demanda", procesoDemanda)

6   miCentralHidraulica = Hidraulico("hidroJuguete",0,vaAporte,100,10,20,0.5)
7   miCentralTermica = Termico("termicaJuguete",100, 150)
8   miDemanda = Demanda("nombre",vaDemanda,200)

9   parque = Parque([miCentralHidraulica], [miCentralTermica],
                    [],[],[],[miDemanda])

10  resolucionSDDP = SDDP(parque,azar,Despachador("gurobi"),2,100,"no")

11  resolucionSDDP.optimizar()
12  simulacion = resolucionSDDP.simular(1000)
```

Las líneas 1 y 2, construyen los procesos estocásticos involucrados en la resolución del problema de despacho. Se asume que la información viene dada en las rutas que se pasan por parámetro. En la línea 3 se construye el Azar y en las líneas 4 y 5 se agregan los procesos estocásticos al Azar. Se cuenta entonces con uno de los componentes fundamentales, el Azar, que se encarga de manejar la incertidumbre.

En la línea 6 se construye un objeto Hidráulico. Se especifica su nombre, el volumen inicial del embalse, la variable aleatoria que modela el aporte (presente en el proceso construido anteriormente), el turbinado máximo, el vertimiento máximo y el coeficiente energético, tal como se detalló en secciones previas. En la línea 7 se construye un objeto Térmico. Se especifica su nombre, su potencia máxima y su costo variable. En la línea 8 se construye el objeto Demanda. Se especifica un nombre para la demanda, la variable aleatoria que modela su incertidumbre (presente en el proceso estocástico definido para la demanda y el costo variable asociado a la demanda que no es posible suministrar (costo de falla). Ya se cuenta con todos los componentes del parque, por lo que se agregan en la línea 9, construyéndose así un objeto Parque.

Al contar con el Parque y el Azar completo, se está en condiciones de construir el problema de operación a resolver mediante SDDP. Para eso se construye en la línea 10, un objeto SDDP. Su construcción toma como parámetros el parque y el azar construidos previamente, un objeto despachador, que en este caso utiliza el resolvedor Gurobi. Como horizonte de tiempo se toman 2 años (por defecto el paso es semanal) y se realizan 100 iteraciones del algoritmo SDDP (criterio de parada). Finalmente se establece que no se utilizará poda de hiperplanos.

En la línea 10, se invoca la función optimizar del objeto SDDP. Esto resuelve el problema de operación luego de realizar 100 iteraciones. El resultado de las correspondientes funciones de Bellman, queda alojado en el objeto, por lo que en la línea 12 se le solicita simular 1000 escenarios. El resultado de la simulación se almacena en la variable *simulacion*. En esa variable se encuentra toda la información de cada uno de los escenarios simulados que permite hacer cualquier tipo de postprocesamiento. De esta forma se trabajó en el Capítulo 6, pero con un sistema eléctrico mucho más complejo.

Esta página ha sido intencionalmente dejada en blanco.

Referencias

[1] Energy consulting and analytics.

[2] PSR. `https://www.psr-inc.com/softwares-en/`, 2018. [Online; accessed Aug-2018].

[3] Administración del mercado eléctrico. `http://adme.com.uy`, 2022. [Online; accessed Aug-2022].

[4] Anum Abid, Tahir Nadeem Malik, Farah Abid, and Intisar Ali Sajjad. Dynamic economic dispatch incorporating photovoltaic and wind generation using hybrid fpa with sqp. *IETE Journal of Research*, 66(2):204–213, 2020.

[5] Dilek Eren Akyuz, Mehmetcik Bayazit, and Bihrat Onoz. Markov chain models for hydrological drought characteristics. *Journal of Hydrometeorology*, 13(1):298–309, 2012.

[6] Richard Bellman. Control theory. *Scientific American*, 211(3):186–201, 1964.

[7] Felipe Beltrán, Erlon C Finardi, Guilherme M Fredo, and Welington de Oliveira. Improving the performance of the stochastic dual dynamic programming algorithm using chebyshev centers. *Optimization and Engineering*, pages 1–22, 2020.

[8] Jacques F Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252, 1962.

[9] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 2005.

[10] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[11] Gonzalo Casaravilla, Ruben Chaer, and Pablo Alfaro. Simsee: Simulador de sistemas de energía eléctrica. Technical report, Proyecto PDT 47/12. Technical Report 7, Universidad de la República (Uruguay). Facultad de Ingeniería. Instituto de Ingeniería Elétrica, Number 7-Dec, 2008.

[12] Antonio J Conejo, Miguel Carrión, Juan M Morales, et al. *Decision making under uncertainty in electricity markets*, volume 1. Springer, 2010.

[13] Luis M Costa and George Kariniotakis. A stochastic dynamic programming model for optimal use of local energy resources in a market environment. In *2007 IEEE Lausanne Power Tech*, pages 449–454. IEEE, 2007.

[14] Vitor L De Matos, Andy B Philpott, and Erlon C Finardi. Improving the performance of stochastic dual dynamic programming. *Journal of Computational and Applied Mathematics*, 290:196–208, 2015.

[15] André Luiz Diniz, Pedro Paulo Ielo Esteves, and Claudia Alejandra Sagastizábal. A mathematical model for the efficiency curves of hydroelectric units. In *2007 IEEE Power Engineering Society General Meeting*, pages 1–7. IEEE, 2007.

[16] O. Dowson and L. Kapelevich. SDDP.jl: a Julia package for stochastic dual dynamic programming. *INFORMS Journal on Computing*, 33:27–33, 2021.

[17] Francis Y Edgeworth. The mathematical theory of banking. *Journal of the Royal Statistical Society*, 51(1):113–127, 1888.

[18] Christian Füllner and Steffen Rebennack. Stochastic dual dynamic programming and its variants. *preprint, Karlsruhe Institute of Technology*, 2021.

[19] Guillermo Gallego. Newsvendor problem. *IEOR course of Columbia University*, 142:143, 1995.

[20] Pierre Girardeau, Vincent Leclere, and Andrew B Philpott. On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40(1):130–145, 2014.

[21] Vincent Guigues. Sddp for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57(1):167–203, 2014.

[22] Vincent Guigues. Dual dynamic programing with cut selection: Convergence proof and numerical experiments. *European Journal of Operational Research*, 258(1):47–57, 2017.

[23] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

[24] TS Halliburton and HR Sirisena. Stochastic optimization for hydro-thermal power systems. *Optimal Control Applications and Methods*, 6(2):91–103, 1985.

[25] Chris Harris. *Electricity markets: pricing, structures and economics*, volume 565. John Wiley & Sons, 2011.

[26] Julia L Higle, Suvrajeet Sen, and S Sen. *Stochastic decomposition: a statistical method for large scale stochastic linear programming*, volume 8. Springer Science & Business Media, 1996.

[27] Masoud Javadi and Turaj Amraee. Economic dispatch: A mixed-integer linear model for thermal generating units. In *2018 IEEE International Conference on Environment and Electrical Engineering and 2018 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*, pages 1–5. IEEE, 2018.

[28] Daniel R Jiang, Thuy V Pham, Warren B Powell, Daniel F Salas, and Warren R Scott. A comparison of approximate dynamic programming techniques on benchmark energy storage problems: Does anything work? In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–8. IEEE, 2014.

[29] James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[30] C. Li, R. Yan, and J. Zhou. Stochastic optimization of interconnected multireservoir power systems. *IEEE Transactions on Power Systems*, 5(4):1487–1496, 1990.

[31] Tristao Araripe Neto, Mo. F. Pereira, and Jerson Kelman. A risk-constrained stochastic dynamic programming approach to the operation planning of hydrothermal systems. *IEEE Transactions on Power Apparatus and Systems*, PAS-104(2):273–279, 1985.

[32] Mario VF Pereira and Leontina MVG Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical programming*, 52(1-3):359–375, 1991.

[33] Andrew B Philpott and Vitor L De Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470–483, 2012.

[34] Rodrigo Porteiro, Andres Ferragut, and Fernando Paganini. Towards multi-timescale energy provisioning using stochastic dual dynamic programming. In *2018 IEEE 9th Power, Instrumentation and Measurement Meeting (EPIM)*, pages 1–6. IEEE, 2018.

[35] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[36] VH Quintana and AY Chikhani. A stochastic model for mid-term operation planning of hydro-thermal systems with random reservoir inflows. *IEEE Transactions on Power Apparatus and Systems*, (3):1119–1127, 1981.

[37] Luciano Raso, David Dorchies, Jan Kwakkel, and Pierre Olivier Malaterre. Optimist: A python library for water system optimal operation and analysis using sddp. 2016.

[38] Johannes Van Der Wal. *Stochastic dynamic programming*. PhD thesis, Methematisch Centrum Amsterdam, The Netherlands, 1980.

[39] Ting Yang, Liyuan Zhao, Wei Li, and Albert Y Zomaya. Dynamic energy dispatch strategy for integrated energy system based on improved deep reinforcement learning. *Energy*, 235:121377, 2021.

[40] Peng Zeng, Hepeng Li, Haibo He, and Shuhui Li. Dynamic energy management of a microgrid using approximate dynamic programming and deep recurrent neural network learning. *IEEE Transactions on Smart Grid*, 10(4):4435–4445, 2018.

[41] Lizhi Zhang, Jiyuan Kuang, Bo Sun, Fan Li, and Chenghui Zhang. A two-stage operation optimization method of integrated energy systems with demand response and energy storage. *Energy*, 208:118423, 2020.

Esta página ha sido intencionalmente dejada en blanco.

Índice de tablas

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

Esta es la última página.
Compilado el jueves 13 abril, 2023.
http://iie.fing.edu.uy/