UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE CIENCIAS

Escuela Profesional de Matemática

Programación Dinámica Dual Estocástica aplicado a modelos de despacho hidrotérmico

TRABAJO SEMINARIO DE TESIS 1

Janice Katherine Escobedo Vásquez

ASESOR:
Ernesto Oré Albornoz

Lima - Perú

25 de agosto de 2023

Índice general

Introducción

Motivación

El uso de energía ha crecido enormemente el último siglo y crece continuamente. La disponibilidad de recursos y combustibles fósiles combinado con las consecuencias ambientales resultan en la producción de energía barata y la explotación de reservas disponibles.
Hoy en día, con la oferta al límite y la creciente preocupación medioambiental, la energía necesita ser extraida más eficientemente y con mejor gestionamiento. Podemos distinguir los siguientes 3 tipos de generación eléctrica:

Generación Térmica

La generación térmica utiliza combustible para calentar agua a medianas (30°C) y altas temperaturas (160°C) y aplicarlo a una gran variedad de usos principalmente para procesos industriales alimenticios, químicos, mineros etc.

En las centrales térmicas, el combustible se quema en una caldera provocando la energía térmica que se utiliza para calentar agua la cual se forma en vapor a una presión muy elevada. Luego, el vapor hace girar una turbina generando energía mecánica que posteriormente se transforma en energía eléctrica.

Generación Hidráulica

La energía hidráulica aprovecha los recursos naturales, como el agua, para generar electricidad, además de ser ecológica ya que no produce emisiones tóxicas.
A grandes rasgos el funcionamiento de una central de energía hidráulica comienza pasando la corriente de agua, almacenada inicialmente en embalses, por compuertas para posteriormente hacer que el agua caiga sobre aspas de turbina haciéndolas girar. Estas turbinas a la vez están conectadas con un alternador o generador eléctrico que permite que se genere así la electricidad.

Generación Hidroeléctrica

La energía hidroeléctrica es aquella que se genera al transformar la fuerza del agua en energía eléctrica. Para transformarla se hace circular un caudal de agua por un circuito hidráulico y así el agua va adquiriendo una velocidad a medida que la energía potencial se va transformando

parcialmente en energía cinética. Posteriormente con la turbina esta energía se transforma en energía mecánica para que finalmente el generador lo transforme en energía eléctrica.

El presente trabajo busca poder solucionar Sistemas Hidrotérmicos formados por reservorios o embalses, plantas térmicas y plantas hidrotérmicas.
Para dar solución a problemas de Despachos hidrotérmicos haremos uso de métodos de descomposición como el Método de Benders y el algoritmo de Programación Dinámica Dual Estocástica, SDDP por sus siglas en inglés.

Antecedentes SDDP

La Programación Dinámica tiene el potencial para resolver problemas de larga planificación de manera dinámica; sin embargo, debido a la discretización del espacio de estados sufre la maldición de la dimensionalidad. Este problema no permite que problemas de la vida real puedan tratarse computacionalmente, incluso hasta hoy en día. Por ello se quiere evitar la discretización usando técnicas de aproximación.

La primera descripción del algoritmo SDDP lo hizo Pereira en 1991 [1], desarrolló una técnica dual dinámica donde la aproximación de la función de costo futuro producía una solución local aproximada. Este algoritmo se acercaba con las iteraciones a la función de costo futuro pero solo en partes donde el espacio de estado era más probable de ocurrir en el problema de optimización. Esto fue una manera de lidiar con la dimensionalidad. Posteriormente extendió su desarrollo al caso estocástico añadiendo incertidumbre en las variables y extendiendo su rango de aplicaciones.

Inicialmente el algoritmo se aplicó a problemas de programación hidroeléctrica para despachar las centrales del vasto sistema hidroeléctrico brasileño. Para la aplicación se determinó una estrategia de operación donde, para cada etapa de periodo de planeación, dado el sistema de estado al inicio de la etapa se produjeran objetivos de generación para cada planta. La estrategia debía minimizar el valor esperado del costo de operación durante el periodo, que es compuesto por costos de combustible más penalizaciones por fallo en el suministro de cargas.

Posteriormente, Alexander Shapiro, en 2011, dió una descripción actualizada del algoritmo SDDP [2], analizó su convergencia y propiedades estadísticas del SDDP aplicado a problemas de Programación lineal estocástica dinámica y, bajo algunas hipótesis, a problemas de Aproximación a la Media Muestral (SAA).

SDDP es un algoritmo que usa los resultados del análisis convexo que establece que cualquier función convexa puede aproximarse por el supremo de funciones afines. SDDP se aplica a problemas de multietapa convexos.

El algoritmo construye políticas factibles de programación dinámica usando aproximaciones eternas de una función de costo futuro que es calculada usando los cortes de Benders. Las políticas definidas por estos cortes pueden ser evaluadas usando simulación y su cambio puede ser medido con respecto a su límite inferior de su coste previsto. Esto nos da un criterio de convergencia que puede ser aplicado para detener el algoritmo cuando el costo estimado sea lo suficientemente cercano a su cota inferior.

Organización del trabajo

El trabajo presente busca detallar la Programación Dinámica Dual Estocástica y aplicarlo a despachos hidrotérmicos. Para esto debemos analizar modelos, resultados y estudio de algunos temas previos.
En el Capítulo 1 mencionamos conceptos que serán útiles para el desarrollo de los siguientes capítulos incluyendo resultados y definiciones de Análisis Convexo, Programación Lineal y Dualidad. Posteriormente en el Capítulo 2 se describen modelos a manera de motivación, sobre Despachos Térmicos determinísticos. Luego consideramos problemas donde existe dinámica en un periodo de tiempo y su reescritura en función de las Ecuaciones de Bellman se describen en el Capítulo 3. Como uno de los problemas de la Programación Dinámica es la maldición de la dimensionalidad se opta por resolver los problemas dinámicos por métodos de aproximación como el Método de Benders, en caso de dos etapas, y el método de Benders multietapa los cuales son detallados en el Capítulo 4, en estos métodos se aprovecha la presencia de variables complicantes para descomponer el problema a la vez que aprovecha la dualidad para aproximar funciones de costo futuro mediante cartas. Hasta ahora hemos tocado problemas deterministicos; sin embargo, cuando las variables tengan incertidumbre se formulan problemas de Programación Estocástica, en el Capítulo 5 se detallan estos problemas incluyendo el de multietapa donde se menciona además la Programación Dinámica Estocástica. Finalmente los capítulos mencionados hasta ahora nos ayudan a poder tratar problemas de Programación Dinámica Dual Estocástica, basado en la aproximación a la media Muestral (SAA), cuyo algoritmo, aplicacion al despacho hidrotérmico y ejemplo se presentará al final del mismo capítulo.

En cada capítulo se encontrará teoría para poder entender las aplicaciones a modelos de despachos térmicos, hidráulicos o hidrotérmicos. Al final del trabajo se presentarán las conclusiones sobre el trabajo desarrollado y adicionalmente se tendrá las implementaciones de algoritmos en los Lenguajes GAMS y Julia, los cuales seràn adjuntados al final del trabajo.

Capítulo 1

Conceptos previos

A lo largo del trabajo necesitaremos conceptos relacionados al Análisis convexo, Programación Lineal tomando en consideración problemas Primales y Duales.

1.1. Análisis convexo

Definición 1.1.1 *(Cono) En un cono $K \subset \mathbb{R}^n$ se cumple que $\forall\, d \in K$ se tiene que $\lambda d \in K$ con $\lambda \geq 0$.*

Definición 1.1.2 *(Cono polar) El cono polar del cono K es definido como el siguiente conjunto*

$$K^* = \{v : \langle v, w\rangle \leq 0, \forall w \in K\}$$

Definición 1.1.3 *(Poliedro) Un conjunto $F \in \mathbb{R}^n$ es llamado poliedro si es la intersección finita de hiperplanos.*

Definición 1.1.4 *(Cono poliedral) Al conjunto C se le dice cono poliedral si tiene la forma*

$$C = \{x : Ax \leq b\}$$

Si C es un conjunto poliedral no vacío se cumple que cada punto de C puede se escrito como sigue

$$C = conv(v_1, v_2, .., v_s) + cone(r_1, r_2..., r_q)$$
$$C = \left\{ \sum_{i=1}^{s} \lambda_i v_i + \sum_{j=1}^{q} \mu_j r_j : \sum_{i=1}^{s} \lambda_i = 1, \lambda_i \geq 0, \mu_j \geq 0 \right\}$$

Definición 1.1.5 *(Función lineal por partes) [3] Un mapeo $F : D \to \mathbb{R}^m$ para el conjunto $D \subset \mathbb{R}^n$ es lineal por partes sobre D, si D puede representarse como la unión de conjuntos poliedrales finitos, con respecto a cada uno de los cuales $F(x)$ es dado por una expresión de la forma $Ax + a$ para alguna matriz $A \in \mathbb{R}^{m\times n}$ y un vector $a \in \mathbb{R}^m$*

Teorema 1.1.1 *(Funciones convexas lineales por partes) [3] Una función propia f es convexa y lineal por partes sí y solo sí se puede expresar de la forma*

$$f(x) = \begin{cases} max\{l_1(x), ..., l_p(x))\} & cuando\ x \in D \\ \infty & cuando\ x \notin D \end{cases}$$

donde D es un conjunto poliedral en $\mathbb{R}^n$ y las funciones l_i con $i = 1, ..., p$, son afines sobre $\mathbb{R}^n$.

Dado que las funciones afines se pueden expresar como $l_i(x) = \langle a, x\rangle + \beta$, entonces $f(x)$ toma la siguiente expresión

$$f(x) = \begin{cases} \max\{\langle a_i, x\rangle + \beta_i, i = 1, ..., p\} & \text{cuando } x \in D \\ \infty & \text{cuando } x \notin D \end{cases}$$

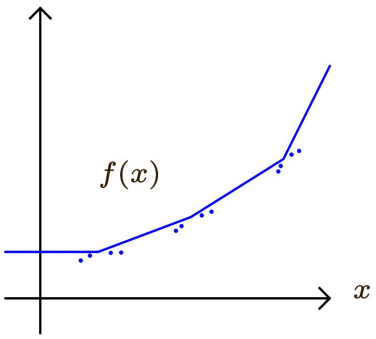


Figura 1.1: Función lineal por partes

Definición 1.1.6 *(Subgradiente de funciones convexas) Sea $\overline{x} \in domf$ donde f es una función $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ propia y convexa, se tiene*

$$\partial f(\overline{x}) = \{v, f(x) \geq f(\overline{x}) + \langle v, x - \overline{x}\rangle,\ para\ todo\ x\}$$

Proposición 1.1.1 *Sea $f_i : \mathbb{R}^n \to (-\infty, +\infty]$ con $i = 1, ..., m$ una colección de funciones convexas. Definimos $f(x) = max\{f_i(x); i = 1, ..., m\}$. Dado $x \in \mathbb{R}^n$ definimos el conjunto de índices activos de $x \in \mathbb{R}^n$ por $I(x) = \{i = 1, ..., m; f_i(x) = f(x)\}$. Si $x \in \cap_{i=1}^m dom(f_i)$ tal que f_i es continua en ese punto, entonces se tiene que*

$$\partial f(x) = conv\left(\bigcup_{i \in I(x)} \partial f_i(x)\right)$$

Si las funciones f_i fueran dferenciables se tendría $\partial f(x) = conv\,(\nabla f_i(x) : i \in I(x))$

Ejemplo 1.1.1 *(Subdiferencial de una función convexa y lineal por partes)*

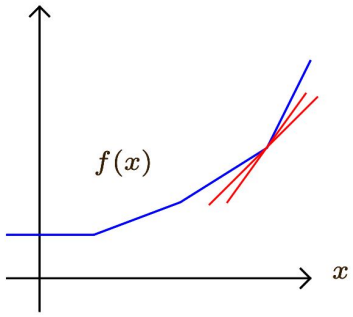


Figura 1.2: Subgradiente de función lineal por partes

Para una función convexa y lineal por partes, que se puede escribir como $\max_{i=1,\dots,m}\{a_i^t x + b_i\}$ se tiene que el subdiferencial es un poliedro

Por lo que el subdiferencial se expresa como

$$\partial f(x) = conv\{a_i \ : i \in I(x)\}$$

donde $I(x) = \{i : a_i x + b_i = f(x)\}$

Definición 1.1.7 *(Carta) A H se le denomina carta de una función φ cuando cumpla que H siempre está por debajo de φ y además H toque en un punto a φ*

1.2. Programación Lineal

Un modelo de Programación Lineal primal y dual se pueden escribir de la forma

$$P_P = \min_x \ \langle c, x\rangle \qquad (1.1) \qquad\qquad P_D = \max_x \ \langle b, y\rangle \qquad (1.2)$$
$$b - Ax \in \mathcal{Y}^* \qquad\qquad A^t y - c \in \mathcal{X}^*$$
$$x \in \mathcal{X} \qquad\qquad y \in \mathcal{Y}$$

donde $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n, x \in \mathbb{R}^n, \mathcal{X} \subset \mathbb{R}^n$ y $\mathcal{Y} \subset \mathbb{R}^m$ son conos poliedrales no vacíos y $\mathcal{X}^* \subset \mathbb{R}^n$ y $\mathcal{Y}^* \subset \mathbb{R}^m$ sus respectivos conos polares.

Los tipos de regiones factibles de los problemas de Programación Lineal se pueden distinguir en 3:

- **Región factible acotada y no vacía:** Si todas las variables de decisión son acotadas sobre la región factible y por tanto la función objetivo lo es, así existe el óptimo.
- **Región factible no acotada:** Una región no vacía se le llama no acotada si al menos una variable de decisión puede tomar valores suficientemente largos sobre la región factible. El óptimo puede o no existir.

- **Región factible vacía:** Al modelo se le llamará infactible entonces no tiene solución optimal. Si es no vacía, la región se le llama factible.

En caso tengamos funciones objetivos no lineales, existen maneras de reescribirlas usando solo funciones lineales, es el caso de algunas funciones que pueden reescribirse como funciones lineales por partes

Ejemplo 1.2.1 *Si consideramos el problema*

$$\begin{aligned} z = \min_{g_1,g_2} \; & f(g_1) + 20g_2 \\ & g_1 + g_2 \geq 150 \\ & 0 \leq g_2 \leq 110 \end{aligned} \tag{1.3}$$

$$\text{donde } f(g_1) = \begin{cases} g_1 & si\ 0 \leq g_1 \leq 50 \\ 50 + 50(g_1 - 50) & si\ 50 \leq g_1 \leq 80 \end{cases}$$

Consideramos las variables

$$u_1 = \begin{cases} g_1 & \text{si } 0 \leq g_1 \leq 50 \\ 50 & \text{si } 50 \leq g_1 \leq 80 \end{cases} \quad \text{y} \quad u_2 = \begin{cases} 0 & \text{si } 0 \leq g_1 \leq 50 \\ g_1 - 50 & \text{si } 50 \leq g_1 \leq 80 \end{cases}$$

Y así tenemos que se cumple

$$u_1 + u_2 = g_1 \wedge u_1 + 50 \cdot u_2 = f(g_1)$$

Y se puede ver que $0 \leq u_1 \leq 50$ y $0 \leq u_2 \leq 30$. Además de la definición de u_1 y u_2 se desprende que $u_2(50 - u_1) = 0$ quedando el problema final, como sigue

$$\begin{aligned} z = \min_{u_1,u_2,g_2} \; & u_1 + 50 \cdot u_2 + 20g_2 \\ & u_1 + u_2 + g_2 \geq 150 \\ & 0 \leq u_1 \leq 50, 0 \leq u_2 \leq 30 \\ & u_2(50 - u_1) = 0 \\ & 0 \leq g_2 \leq 110 \end{aligned} \tag{1.4}$$

1.2.1. Optimalidad

Proposición 1.2.1 *Consideramos el problema P_P y P_D. Si los dos modelos son factibles entonces*

$$min\{c^T x | b - Ax \in \mathcal{Y}^*, x \in \mathcal{X}\} \geq max\{b^T y | A^t y - c \in \mathcal{X}^*, y \in \mathcal{Y}\}$$

Teorema 1.2.1 *La región factible de un modelo de Programación lineal en su forma estandar tiene al menos un vértice.*

Teorema 1.2.2 *(Teorema del vértice optimal) Un modelo lineal en su forma estandar tiene solución óptima sí y solo sí su región factivle contiene un vértice optimal.*

Proposición 1.2.2 *Si P_P o P_D es finito, entonces los problemas primal y dual poseen solución y mismo valor óptimo.*

Además, si x_0 pertenece al conjunto admisible de P_P; y y_0 al de P_D etonces

$$\langle c, x_0\rangle \geq \langle A^t y_0, x_0\rangle = \langle y_0, Ax_0\rangle \geq \langle y_0, b\rangle$$

Teorema 1.2.3 *(Condición de Dualidad fuerte) Si un modelo en su forma estandar tiene solución optimal, entonces su modelo dual tiene solución optimal también, y viceversa; los valores objetivos de los dos problemas coinciden.*

Infactibilidad y no acotamiento

Teorema 1.2.4 *Si un modelo primal (dual) es no acotado entonces su correspondiente modelo dual (primal) es infactible*

Podemos resumir los posibilidades en la siguiente tabla tomando en cuenta que Óptimo significa "existe una solución optimal" e Infactible, "infactibilidad en el modelo".

	Modelo Dual		
Modelo Primal	Optimal	Infactible	No acotado
Optimal	Posible	Imposible	Imposible
Infactible	Imposible	Posible	Posible
No acotado	Imposible	Posible	Imposible

1.2.2. Algoritmo Simplex

Consideremos a el problema y su reformulación, respectivamente.

$$\left.\begin{array}{ll} max & c^t x \\ s.a. & Ax \leq b \\ & x \geq 0 \end{array}\right\} \equiv \left\{\begin{array}{ll} max & c^t x \\ s.a. & Ax + x_s = b \\ & x \geq 0, x_s \geq 0 \end{array}\right.$$

donde x_s se denominan variables de holgura

Definición 1.2.1 *(Matriz básica) B será una matriz básica si es la (m, m) submatriz determinada por columnas linealmente independientes extraídas de la matriz $\begin{bmatrix} A & I_m \end{bmatrix}$. Más aún, los índices a sus correspondientes columnas se coleccionan en un conjunto denotado por BI y su complemento, los índices que corresponden a las columnas no básicas se denotan por NI.*

Observación: Debido a que una matriz básica tiene columnas l.i. es invertible.

Definición 1.2.2 *(Matriz básica factible) Una matriz básica factible es una matriz básica que cumple $B^{-1}b \geq 0$.*

Al tener nuestra reformulación, la matriz tecnológica es $\begin{bmatrix} A & I_m \end{bmatrix}$ y por tanto nuestra variable pasa a ser $\begin{bmatrix} x \\ x_s \end{bmatrix}$

Definición 1.2.3 *(Solucion básica y no básica) Una solución básica esta confirmada por las variables de los índices básicos, BI; y una solución no básica, por los índices no básicos*

$$x_{BI} \equiv \begin{bmatrix} x \\ x_s \end{bmatrix}_{BI} \text{ y } x_{NI} \equiv \begin{bmatrix} x \\ x_s \end{bmatrix}_{NI}$$

Ahora consideremos una solución básica factible. Sea $BI \subseteq \{1, ..., m+n\}$ el conjunto de índices de las correspondientes variables básicas y sea $NI = \{1, ..., m+n\} \setminus BI$ el conjunto de índices de las variables no básicas.
Definimos

$$B = \begin{bmatrix} A & I_m \end{bmatrix}_{*,BI} \text{ y } N = \begin{bmatrix} A & I_m \end{bmatrix}_{*,NI}$$

El algoritmo Simplex busca de ir de solución básica factible en solución básica factible y sigue los siguientes pasos

Entrada: Valores de A, b y c en el modelo estandar $\max\{c^t x | Ax \leq b, x \geq 0\}$ y en una solución inicial básica factible para los modelos con variables.

Salida: O bien la solución optimal del modelo o el mensaje "el modelo es no acotado"

Paso 1: *Construir una matriz básica* Sea BI y NI a los conjuntos de índices de las variables básicas y no básicas respectivamente, correspondiente a la actual solución básica factible. Sea B la matriz que consta de las columnas $\begin{bmatrix} A & I_m \end{bmatrix}$ con indices BI, y sea N consiste de las columnas de $\begin{bmatrix} A & I_m \end{bmatrix}$ con índices en NI. La solución actual es $x_{BI} = B^{-1}b, x_{NI} = 0$, con valor objetivo actual correspondiente a $z = c^t_{BI} x_{BI}$. Ir al **Paso 2.**

Paso 2: *Test de optimalidad; escoger una variable entrante* El vector (fila) de coeficientes objetivos para las variables no básicas es $c^t_{NI} - c^t_{BI} B^{-1} N$. Si para cada entrada es este vector es no positivo entonces una solución optimal es encontrado y **PARAR.** En otro caso, se selecciona una entrada positiva en $c^t_{NI} - c^t_{BI} B^{-1} N$; sea $\alpha \in \{1, ..., n\}$ es el índice de el coefficiente objetivo (es decir, corresponde a la variable x_{NI_α}). Entonces x_{NI_α} será una variable básica. Ir a **Paso 3.**

Paso 3: *Test de Mínimo ratio; escogiendo la variable saliente.* Si $(B^{-1}N)_{j,\alpha} \leq 0$ para cada $j \in \{1, ..., m\}$, entonces se imprime el mensaje "el modelo no es acotado ", **PARAR**. En otro caso, determine un índice $\beta \in \{1, ..., m\}$ tal que:

$$\frac{(B^{-1}N)_\beta}{(B^{-1}N)_{\beta,\alpha}} = \min \left\{ \frac{(B^{-1}N)_j}{(B^{-1}N)_{j,\alpha}} \mid (B^{-1}N)_j > 0, j- = 1, ..., m \right\}$$

Ir al **Paso 4**

Paso 4: *Intercambiando.* Fijamos $BI := (BI \ \{BI_\beta\}) \cup \{NI_\alpha\}$ y $NI := (NI\{NI_\alpha\}) \cup \{BI_\beta\}$, retornar al **Paso 1**.

1.2.3. Tabla simplex

En cada paso iterativo del algoritmo simplex descrito en la subsection anterior, se necesita determinar la formulación actual del modelo. Esta formulación se calcula construyendo la matriz básica y no básica B y N respectivamente y derivando la formulación usando la expresión

$$\begin{aligned} \max \ & c_{BI}^T B^{-1} b + (c_{NI}^T - c_{BI}^T B^{-1} N) x_{NI} \\ & I_m x_{BI} + B^{-1} N x_{NI} = B^{-1} b \\ & x_{BI} \geq 0, x_{NI} \geq 0 \end{aligned}$$

La expresión involucra que en cada iteración se tenga que calcular la inversa de la matriz B de orden m. Lo que conlleva un costo computacional alto para cada iteración; sin embargo existe una técnica para llevar a cabo el cálculo requerido sin la necesidad de invertir la matriz B. Esta técnica hace uso de las llamadas *tablas-simplex*. Una tabla simplex puede ser esquemáticamente representado de forma particionada y permutada como sigue.

m columnas (BI)	n columnas (NI)	una columna
0^T	$c_{NI}^T - c_{BI}^T B^{-1} N$	$-c_{BI}^T B^{-1} b$
I_m	$B^{-1} N$	$B^{-1} b$

Esta tabla tiene $m + 1$ filas y $m + n + 1$ columnas.

Ejemplo 1.2.2 *Se desea minimizar los costos de 3 plantas térmicas cuyos costos de generación son de* $\$10, \$15$ *y* $\$8$ *y cuya demanda del sistema sea de* 40

El problema se formula como sigue

$$\begin{aligned} \min \ & 10g_1 + 15g_2 + 8g_3 \\ & g + 1 + g_2 + g_3 \geq 40 \\ & g_1 \leq 20, g_2 \leq 15, g_3 \leq 8 \end{aligned}$$

Como $\underset{x \in D}{min} f(x) \equiv -\underset{x \in D}{max}(-f(x))$, el problema es equivalente a

$$\begin{aligned} -\max \ & -10g_1 - 15g_2 - 8g_3 \\ & -g_1 - g_2 - g_3 \leq -40 \\ & g_1 \leq 20, g_2 \leq 15, g_3 \leq 8 \end{aligned}$$

La tabla del problema completando con variables de holgura es como la Tabla 1.1a. Para pasarlo a tabla simplex debemos tener el vector 0 arriba de las variables básicas. Tomemos a g_1, g_2, g_3, s_1 como variables básicas ya que $B^{-1} b \geq 0$.

-10	-15	-8	0	0	0	0	0
-1	-1	-1	1	0	0	0	-40
1	0	0	0	1	0	0	20
0	1	0	0	0	1	0	15
0	0	1	0	0	0	1	8

(a) Tabla inicial del problema

0	0	0	0	10	↓ 15	8	169
0	0	0	1	1	[1]	1	13
1	0	0	0	1	0	0	20
0	1	0	0	0	1	0	15
0	0	1	0	0	0	1	8

(b) Primera Tabla Simplex

0	0	0	-15	-5	0	-7	374
0	0	0	1	1	1	1	13
1	0	0	0	1	0	0	20
0	1	0	-1	-1	0	-1	2
0	0	1	0	0	0	1	8

(c) Segunda Tabla Simplex

Tabla 1.1: Tablas Simplex

Pivoteando obtenemos la primera Tabla Simplex vista en la Tabla 1.1b.
Por el test de mínimo ratio entra como variable básica s_2 y sale g_1 y pasamos a la Segunda Tabla Simplex, Tabla 1.1c. Como los costos reducidos de la tabla son negativos hemos llegado al óptimo.

Se alcanza el óptimo con un valor de -374. Así el valor óptimo al minimizar los costos es el opuesto es decir, 374.

1.2.4. Interpretación de variables duales

Suponiendo que la solución óptima del problema primal es

$$x^* = \begin{pmatrix} B^{-1}b \\ 0_{n-m} \end{pmatrix}$$

con B matriz básica factible y sea $w^{*\,t} = c_B^t B^{-1}$ la solución usando su dual complementaria. Podemos expresar la función objetivo en función de las variables no básicas

$$z = c_B^{-1}b - \sum_{j \in N}(z_j - c_j)x_j = b^t w^* - \sum_{j \in N}(z_j - c_j)x_j$$

Observamos que un pequeño cambio para los valores de b_i no provocan un cambio de base óptima (las variables no básicas no dejan de ser nulo). Así podemos suponer a z diferenciable en un entorno de b, quedando

$$\frac{\partial z}{\partial b_i} = w_i^*$$

Entonces podemos interpretar a la $i-$ésima variable dual como la razón de cambio de la función objetivo cuando la $i-$ésima restricción del problema primal es sometido a pequeños cambios.

Capítulo 2

Problemas Energéticos

El problema de manejo de embalses para un sistema de energía hidrotermal es decidir cuánta agua debe liberarse en cada periodo para minimizar la esperanza del costo operacional, incluyendo el costo de combustible de las plantas térmicas y el costo de las escaceses. Ya que los reservorios son limitados con capacidades de almacenamiento y los flujos de entrada son inciertos, tienen incertidumbre, entonces la optimización de la gestión de los embalses se vuelve compleja.
Resolver el problema usando Programación Lineal se vuelve aún más complejo por la gran escala que posee. Además, el problema deja de ser determinístico y al tomar la esperanza se requiere usar la utilización futura óptima del agua bajo un número posibles secuencias de entradas futuras. [4]

2.1. Problema de Despacho Térmico

Un problema de Despacho térmico que toma en cuenta suplir la demanda sin sobreexigir a su generador se formula como sigue

$$\begin{aligned} z = \min & \sum_{i-1}^{I} c_i g_i \\ s.a. & \sum_{i=1}^{I} g_i \geq d \\ & 0 \leq g_i \leq \overline{g}_i \;\; i = 1, ..., I \end{aligned} \tag{2.1}$$

donde

Parámetros:

i	indexa los generadores
I	cantidad de generadores
c_i	costo unitario de operación del $i-$ésimo generador
$\overline{g}_i$	capacidad del $i-$ésimo generador

d demanda del sistema

Variables:

g_i energía producida por el $i-$ésimo generador

Proposición 2.1.1 *Considerando el problema 2.1. Se cumple que*

$$\sum_{i=1}^{I} \overline{g}_i \geq d \tag{2.2}$$

sí y solo sí el problema es factible.

Prueba:

($\Rightarrow$) Si el conjunto factible fuese vacío significa que no existe g en el admisible del problema, entonces se cumple que para todo g se cumple

$$\sum_{i=1}^{I} g_i < d \quad \wedge \quad (0 > g_i \wedge g_i > \overline{g}_i \;\; i = 1, ..., I)$$

Entonces se tendría

$$\sum_{i=1}^{I} \overline{g}_i < \sum_{i=1}^{I} g_i < d \;\; (\Rightarrow\Leftarrow)$$

($\Leftarrow$) Si el problema es factible existe $x \in \mathbb{R}^n$ en el conjunto admisible de 2.1; es decir,

$$\sum_{i=1}^{I} g_i \geq d \quad \wedge \quad 0 \leq g_i \leq \overline{g}_i \;\; i = 1, ..., I$$

De ahí se tiene

$$d \leq \sum_{i-1}^{I} g_i \leq \sum_{i-1}^{I} \overline{g}_i$$

□

Suposicion: *En lo que continua se considerará de antemano la condición (2.2) y* $c_i > 0 \; \forall i \in \{1, ..., m\}$

Como se cumple la condición (2.2), por hipotesis, usando la Proposición 1.2.1 y 1.2.2 la solución se alcanza en un vértice. Para resolver el modelo podemos usar el algoritmo Simplex o el simplex Dual.

- Usando el Algoritmo Simplex: De antemano tenemos factibilidad en el problema; sin embargo, la solución básica factible no es explícita y portanto podemos usar el método de inicialiación Big-M.
- Usando el Algoritmo Simplex Dual: La solución básica factible es la trivial, el origen.

Añadiendole una perturbación $\delta \in \mathbb{R}$ al modelo tenemos

$$\begin{aligned} z = \min & \sum_{i=1}^{I} c_i g_i + M \cdot \delta \\ s.a. & \sum_{i=1}^{I} g_i + \delta = d \\ & 0 \leq g_i \leq \overline{g}_i \;\; i = 1, ..., I \\ & \delta \geq 0 \end{aligned} \tag{2.3}$$

donde

Parámetros:

i	indexa los generadores
I	cantidad de generadores
c_i	costo unitario de operación del $i-$ésimo generador
d	demanda del sistema
$\overline{g}_i$	capacidad del $i-$ésimo generador
c_δ	costo por no satisfacer la cantidad de energía δ

Variables:

g_i	energía producida por el $i-$ésimo generador
δ	energía no satisfecha

El problema 2.3 siempre es factible, considerando $g_i = 0$ para todo $i \in I$ y $\delta = d$. Además el modelo sigue teniendo la estructura de un modelo de Despacho térmico con generadoras $g_1, g_2, ..., g_I, \delta$; es decir, se le puede considerar a δ como una generadora artificial con costo de generación c_δ y en tal caso $M = c_\delta$. Resta hallar alguna cota para c_δ para poder obtener el mismo valor óptimo que el problema 2.1.

Observación 2.1.1 *Aunque ya podemos inciar el algoritmo Simplex con el factible dado, vale la pena mencionar el Metodo de Baleriaux que permite hallar las generaciones térmicas bajo ciertas condiciones y ver que relación tiene con el Algoritmo Simplex.*

2.2. Método de Baleriaux.

El método de Baleriaux se basa en la posibilidad de descomponer el problema en $K \times J$ subproblemas pero permite resolver los subproblemas analíticamente.
El problema 2.1 puede ser resuelto colocando los costos de los generadores de manera ascendente para alcanzar su máxima capacidad y satisfacer la demanda. Así la energía óptima producida por cada generador esta dada por

$$g_i^* = min\{\overline{g}_i, w_{i-1}\} \tag{2.4}$$

donde $w_{i-1} = max\{0, d - \sum_{n=1}^{i-1} \overline{g_n}\}$ y puede ser interpretado como la energía no suplida de un sistema en el que solo están disponibles los primeros $i-1$ generadores

$$\begin{aligned} w_{i-1} = \min_{g_1,...,g_{i-1},\delta} \quad & \delta \\ s.a. \quad & \delta + \sum_{n=1}^{i-1} g_n = d \\ & g_n \leq \overline{g}_n \quad n = 1, ..., i-1 \\ & \delta \geq 0 \end{aligned}$$

Si hallando el valor óptimo de w_{i-1}, el valor de δ es positivo puede ser visto como una variable escalar que representa el corte de carga del sistema debido a una capacidad insuficiente de generación. Por otro lado cuando δ asuma el valor cero, las generadoras posteriores $g_i, g_{i+1}, ..., g_I$ no generan energía.
Afirmamos que $g_i^* = w_{i-1} - w_i$; es decir, g_i^* es dado por la diferencia entre la energía no suplida antes y después de que el generador entre en funcionamiento.

a) Si (2.4) tiene solución en $g_i = \overline{g}_i$. En este caso, w_{i-1} y w_i son dados por $w_{i-1} = d - \sum_{n=1}^{i-1} \overline{g}_n$ y $w_i = d - \sum_{n=1}^{i} \overline{g}_n$, restando se tiene que $g_i = \overline{g}_i = w_{i-1} - w_i$.

b) Si (2.4) tiene solución en $g_i < \overline{g}_i$ entonces $g_i = max\{0, d - \sum_{n=1}^{i-1} \overline{g}_n\}$, que es por construcción w_{i-1}. Además nos queda que $w_i = 0$. Por tanto se cumple

$$g_i^* = max\{0, d - \sum_{n=1}^{i-1} \overline{g}_n\} - 0 = w_{i-1} - w_i$$

Concluyendo así que g_i^* se puede representar de otra manera como $g_i^* = w_{i-1} - w_i$.
En resumen el problema operativo puede ser resuelto con el siguiente algoritmo.

1. Ordenar los costos de manera ascendente, además ordenar las generadoras por los índices ascendentes.

2. Calcular la energía no suplida considerando las primeras i generadoras

$$w_i = max\{0, d - \sum_{n=1}^{i} \overline{g}_n\} \quad , i = 1, ..., I$$

3. Calcular la energía producida por cada generador

$$g_i = w_{i-1} - w_i \quad , i = 1, ..., I$$

4. Calcular el costo operativo

$$z = \sum_{i=1}^{I} c_i g_i$$

Proposición 2.2.1 *El Método de Baleriaux es equivalente a aplicar el Algoritmo Simplex del Dual para el modelo (2.1)*

Prueba: Consideremos los costos de generación ordenados ascendentemente.

$$c_1 < c_2 < ... < c_I$$

El problema y su dual se escriben como

$$\left.\begin{array}{cc} \min & \sum_{i=1}^{I} c_i g_i \\ s.a. & \sum_{i=1}^{I} g_i \geq d,\ g_i \leq \overline{g}_i\ i = 1, ..., I \\ & 0 \leq g_i, i = 1, ..., I \end{array}\right\} (P) \qquad \left.\begin{array}{cc} \max\limits_{y \in \mathbb{R}^{I+1}} & \begin{pmatrix} d & -\overline{g}_1 & ... & -\overline{g}_I \end{pmatrix}^t\ y \\ s.a. & \begin{bmatrix} \mathbb{1}^t & -Id \end{bmatrix}\ y \leq c \\ & 0 \leq y \end{array}\right\} (D)$$

Aplicamos el algoritmo simplex al Dual del problema; es decir, al problema (D)

$\overset{\downarrow}{d}$	$-\overline{g}_1$	$-\overline{g}_2$	$\cdots$	$-\overline{g}_I$	0	0	0	$\cdots$	0	0
$\boxed{1}$	-1	0	$\cdots$	0	1	0	0	$\cdots$	0	c_1
1	0	-1	$\cdots$	0	0	1	0	$\cdots$	0	c_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
1	0	0	$\cdots$	-1	0	0	0	$\cdots$	1	c_I

Tabla 2.1: Primera tabla simplex

Comenzamos la Primera Tabla Simplex, Tabla 2.1, la variable y_1 entra como variable básica ya que los costos reducidos restantes son negativos, mientras que la variable que sale dado por el Test de mínimo Ratio es la primera variable de holgura debido al ordenamiento de los costos. Así continuamos con el algoritmo.

0	$\overset{\downarrow}{d - \overline{g}_1}$	$-\overline{g}_2$	$\cdots$	$-\overline{g}_I$	-d	0	0	$\cdots$	0	$-d\ c_1$
1	-1	0	$\cdots$	0	1	0	0	$\cdots$	0	c_1
0	$\boxed{1}$	-1	$\cdots$	0	-1	1	0	$\cdots$	0	$c_2 - c_1$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
0	1	0	$\cdots$	-1	-1	0	0	$\cdots$	1	$c_I - c_1$

Tabla 2.2: Segunda tabla simplex

Definimos $w_i = max\{0, d - \sum_{n=1}^{i} \overline{g}_n\}, i = 1, ..., I.$

Continuando con la Segunda tabla Simplex, si $d - \overline{g}_1 \leq 0$, en la Tabla 2.2, entonces se alcanza el óptimo con un valor óptimo de $dc_1 = \overline{g}_1 c_1$ y por definición $w_1 = 0$, como las generaciones siguientes son nulas $w_i = 0\, \forall i = 2, ..., I$ y así $g_1 = d = d - w_1 = w_0 - w_1$, en otro caso el algoritmo Simplex continúa y $w_1 = d - \overline{g}_1$.

		$\overset{\downarrow}{d-\overline{g}_1-\overline{g}_2}$	$\cdots$	$-\overline{g}_I$	$-\overline{g}_1$	$\overline{g}_1-d$	0	$\cdots$	0	$-(c_1\overline{g}_1+c_2(d-\overline{g}_1))$
1	0	-1	$\cdots$	0	0	1	0	$\cdots$	0	c_2
0	1	-1	$\cdots$	0	-1	1	0	$\cdots$	0	c_2-c_1
0	0	$\boxed{1}$	$\cdots$	0	0	-1	1	$\cdots$	0	c_3-c_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
0	0	1	$\cdots$	-1	0	-1	0	$\cdots$	1	c_I-c_2

Tabla 2.3: Tercera tabla simplex

Si $d-\overline{g}_1-\overline{g}_2\leq 0$, en la Tabla 2.3 entonces se alcanza el óptimo con un valor óptimo de $c_1\overline{g}_1+c_2(d-\overline{g}_1)$ y por definición $w_2=0$ y así $g_2=d-\overline{g}_1=w_1=w_1-w_2$, en otro caso el algoritmo Simplex continúa y $w_2=d-\overline{g}_1-\overline{g}_2$

Para la Tabla $i-$ésima, con $i<I$, si $d-\sum_{n=1}^{i}\overline{g}_n\leq 0$ se alcanza el óptimo con valor $c_1\overline{g}_1+c_2(d-\overline{g}_1)+...+c_I(d-\sum_{n=1}^{i}\overline{g}_i)$ y por definición $w_i=0$ y así $g_i=d-\sum_{n=1}^{i}\overline{g}_i=w_i=w_i-w_{i+1}$, en otro caso el algoritmo continúa.

Si se llegará a dar el caso que $d-\sum_{n=1}^{I-1}\overline{g}_n\leq 0$, la Tabla sería la que se muestra en

$0_{1\times I}$	$d-\sum_{i=1}^{I}\overline{g}_i$	$-\overline{g}_1$	$-\overline{g}_2$	$-\overline{g}_3$	$\cdots$	$-\overline{g}_{I-2}$	$-\overline{g}_{I-1}$	$\sum_{i=1}^{I-1}\overline{g}_i-d$	z
	-1	0	0	0	$\cdots$	0	0	0	c_I
	-1	-1	0	0	$\cdots$	0	0	0	c_I-c_1
	-1	0	-1	0	$\cdots$	0	0	0	c_I-c_2
$Id_{I\times I}$	-1	0	0	-1	$\cdots$	0	0	0	c_I-c_3
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	-1	0	0	0	$\cdots$	-1	0	0	c_I-c_{I-2}
	-1	0	0	0	$\cdots$	0	-1	0	c_I-c_{I-1}

Tabla 2.4: Tabla simplex final con $z=c_1\overline{g}_1+c_2(d-\overline{g}_1)+...+c_I(d-\sum_{n=1}^{I}\overline{g}_n)$

la Tabla 2.4. Si $d-\sum_{n=1}^{i}\overline{g}_n\leq 0$ entonces se alcanza el óptimo con un valor óptimo de $c_1\overline{g}_1+c_2(d-\overline{g}_1)+...+c_I(d-\sum_{n=1}^{I}\overline{g}_n)$.

Como se cumple que el valor de cada generadora g_i se representa como $w_{i-1}-w_i$ con w_i definido como $w_i=max\{0,d-\sum_{n=1}^{i}\overline{g}_n\}, i=1,...,I$ entonces resolver el problema por el algoritmo Simplez Dual y por el Método de Baleriaux obtendrá los mismos valores optimales, así los dos métodos son equivalentes. □

Proposición 2.2.2 *El problema no tiene solución única sí y solo sí existen al menos dos costos c_i y c_j iguales con $i\neq j$*

(⇒) Probemos que si todos los costos son distintos entonces la solución es única. En este caso el algoritmo de Baleriaux admite los costos distintos y se da la recursión como se vio antes. Si llegamos a la tabla $i-$ésima el algoritmo tiene solución única.

($\Leftarrow$) Si la solución es única entonces todos los costos son distintos.
Si se diera $c_i = c_j$, con $i < j$, la variable saliente de la $i-$ésima tabla Simplex cambiaría y así cambiaría el valor de y_i siendo las soluciones duales $(y_1^*, y_2^*, ..., y_i^*, ..., y_j^*, ..., y_{I+1}^*)$, cuya solución en el primal se asocia a $(g_1^*, ..., g_i^*, ..., g_j^*, ..., g_I^*)$ y $(y_1^*, y_2^*, ..., y_j^*, ..., y_i^*, ..., y_{I+1}^*)$, cuya solución en el primal se asocia a $(g_1^*, ..., g_j^*, ..., g_i^*, ..., g_I^*)$. Por lo tanto la solución no es única cuando existan al menos 2 costos de generación térmica iguales.

Proposición 2.2.3 *Cuando* $max\{c_i, i = 1, ..., I\} < c_\delta$ *los problemas 2.1 y 2.3 tienen mismo valor optimal.*

Prueba: Según proposición anterior el problema 2.1 tiene solución y su solución se alcanza en la restricción activa pudiendo aplicar el Método de Baleriaux. Sean los costos ordenados de manera ascendente

$$c_1, c_2, c_3, ..., c_I$$

Supongamos el método se acaba incurriendo un costo en c_j, con $j < I$. Por otro lado, dada la hipótesis tenemos

$$c_i < \max\{c_i, i = 1, ..., I\} \leq c_\delta \ \ , i = 1, ..., I$$

Entonces los costos para el problema 2.3 se ordenan ascendentemente como sigue

$$c_1, c_2, c_3, ..., c_I, c_\delta$$

Si aplicamos el Método de Baleriaux seguiría deteniéndose incurriendo en el costo c_j y no generando energ[ia en las siguientes plantas térmicas, obteniendo el mismo valor óptimo que el problema.2.1 □

Hasta ahora hemos considerado los problemas de despacho determinísticos y de una etapa. Sin embargo, los problemas aplicativos toman en cuenta sistemas más complejos con más variables, tales como tiempos máximos y mínimos de operación o el costo adicional de arranque de las unidades térmicas, tiempo de enfriamiento de la planta de energía, entre otros. Además como ya mencionamos los sistemas eléctricos pueden ser caracterizados como térmicos, hidráulicos o hidrotérmicos segun el grado de utilización de las plantas térmicas o hidráulicas. Las plantas térmicas usan combustibles fósiles tales como gas, petroleo y carbón para producir calor y vapor adquirido por las altas preciones que mueven las aspas de turbina al generador. Las hidráulicas usan energía potencial del agua almacenada en un reservorio para mover aspas de turbina directamente. Además algunos despachos incluyen a los despachos térmicos como los despachos Hidráulicos e Hidrotérmicos serán vistos en los capítulos siguientes.

Capítulo 3

Programación Dinámica

La programación dinámica busca una solución que pueda ser representado por una serie de decisiones secuenciales que se toman con el tiempo, más aún, estas decisiones son tomadas de los subproblemas los cuales son problemas más simples que el general. Algunas definiciones importantes son:

- Horizonte de tiempo: Se refiere al número de etapas en el problema
- Etapas: Es una característica esencial el cual aprovecha la estructura del problema de optimización en múltiples etapas.
- Variables de estado: Viven en el espacio de estados S y describe el sistema de estado de la etapa actual sin requerir el conocimiento del comportamiento del sistema histórico.
- Función de transición: Nos muestra como cambian las variables de estado con el tiempo

El modelo dinámico a optimizar tiene la estructura siguiente

$$\begin{aligned} \min_{d_1,\ldots,d_{T-1}} & \sum_{i=1}^{T-1} f_i(s_i, d_i) + f_T(s_T) \\ s.a.\ & s_{i+1} = t_i(s_i, d_i) \ \ \forall i \in \{1, ..., T-1\} \\ & d_i \in D_i(s_i), s_{i+1} \in S_{i+1} \ \ \forall i \in \{1, ..., T-1\} \end{aligned}$$

donde s_i son las variables de estado, S_i es el espacio de estados para la etapa i, t_i función de transición. Para que el problema esté bien definido se debe tener

$$\{z | z = f_k(s, u), s \in S, u \in D_k(s)\} \subset S, k = 0, 1, ..., T-1$$

El espacio de estados puede ser discreto o continuo, en caso continuo consideramos una partición uniforme para que conjunto S_i sea finito o tomamos algunos de sus valores que nos sean representativos.
Teniendo en cuenta las definiciones, un problema de Programación Dinámica busca encontrar una política optimal para la decisión hecha sobre el periodo de tiempo de interés. El término

política se refiere a una regla para tomar decisiones que conducen a una secuencia de decisiones admisibles. Así, una *política optimal* se refiere a una regla para tomar decisiones que conducen a una secuencia de decisiones admisibles donde estas decisiones optimizan una función de las variables de estados ya predefinida. La Programación Dinámica produce una política optimal para cada periodo de tiempo del problema de planeamiento utilizando el *Principio de Optimalidad*.

Principio de Optimalidad: *Una política óptima tiene la propiedad de que cualquiera que sea el estado inicial y la decisión inicial, las decisiones restantes deben constituir una política óptima con respecto al estado resultante de la primera decisión.* [5]

El principio de optimalidad sugiere que una política optimal puede ser construida de manera fragmentada, primero construimos una política optimal para un subproblema que involucra la última etapa, luego extendemos la política optimal a otro subproblema que involucra las últimas 2 etapas y continuamos de esa manera hasta una política optimal para todo el problema.

3.1. Programación Dinámica Deterministica

3.1.1. Ecuación de Bellman

Segun Bellman podemos describir el trabajo de un proceso discreto de Programación Dinámica. Primero, sean los estados s_k definido en el espacio de estados denotado por S_k y t_k transformaciónes con propiedad de clausura, es decir $s_{k+1} = t_k(s_k, d) \in S_{k+1}$ siempre que $d \in D_k$, llamado el espacio de decisiones o espacio de control. Consideremos una nueva secuencia de decisiones $d_1, ..., d_{T-1}$ el cual genera una secuencia de puntos en S

$$\begin{aligned} s_2 &= t_1(s_1, d_1) \\ s_3 &= t_2(s_2, d_2) \\ \vdots &\qquad \vdots \\ s_T &= t_{T-1}(s_{T-1}, d_{T-1}) \end{aligned}$$

Y pedimos que esas variables de decisión sean escogidas de modo que minimicen una función escalar prescrita como sigue tomando en cuenta que s_n depende de s_{n-1} y d_{n-1}.

$$R = f_1(d_1, s_1) + f_2(d_2, s_2) + ... + f_{T-1}(d_{T-1}, s_{T-1}) + f_T(d_T, s_T)$$

Observamos que el mínimo valor depende de s_T, el estado inicial, siendo T el periodo o el número de etapas. Ahora introduzcamos una secuencia de funciones $\{v_n(s_n)\}$, $s_n \in S_n$ con $n = 1, 2, ...N$, definidos para el problema

$$\min_{\{d_0,...,d_N\}} R$$

Podemos obtener una relación útil conectando diferentes miembros de una familia [6]. De este modo, construimos un puente entre los problemas simples y los difíciles.

$$
\begin{aligned}
v^* = & \min_{d_1,...,d_T} [f_1(s_1,d_1) + f_2(s_2,d_2) + ... + f_T(s_T)] \\
& s.a.\ s_{i+1} = t_i(s_i,d_i)\ \ \forall i \in \{1,...,T-1\} \\
& \quad d_i \in D_i(s_i),\ s_{i+1} \in S_{i+1}\ \ \forall i \in \{1,...,T-1\} \\
= & \min_{d_1}[f_1(s_1,d_1) \quad + \quad \min_{d_2,...,d_{T-1}} [f_2(s_2,d_2) + ... + f_T(s_T)]] \\
& s.a.\ s_2 = t_1(s_1,d_1) \qquad s.a.\ \ s_{i+1} = t_i(s_i,d_i)\ \ \forall i \in \{1,...,T-1\} \\
& \quad d_1 \in D_1(s_1) \qquad\qquad d_i \in D_i(s_i), s_{i+1} \in S_{i+1}\ \ \forall i \in \{2,...,T-1\} \\
= & \min_{d_1}[f_1(s_1,d_1) \quad + \quad \min_{d_2}[f_2(s_2,d_2) \quad + \quad ... \quad + \quad \min_{d_{T-1}}[f_T(s_T)]]] \\
& s.a.\ s_2 = t_1(s_1,d_1) \quad s.a.\ \ s_3 = t_2(s_2,d_2) \qquad\qquad s.a.\ \ d_T \in D_T \\
& \quad d_1 \in D_1(s_1) \qquad\quad d_2 \in D_2(s_2) \\
& \quad s_1 \in S_1; \qquad\qquad s_2 \in S_2
\end{aligned}
$$

Entonces se pueden definir las funciones de Bellman $v_n(s_n)$ con $n \in \{1,...,T\}$. Dado $k_0 \in \{1,...,T-1\}$

$$
\begin{aligned}
v_{k_0}(s) = \min_{(u_{k_0},u_{k_0+1},...,u_{T-1})} & \sum_{k=k_0}^{T-1} f_n(d_n,s_n) + f_T(s_T) \\
& s_{k_0} = s,\ s_{k+1} = t_k(x_k,d_k),\ \ \forall k \in \{k_0,...,T-1\} \\
& d_k \in D_k(s_k),\ \ \forall k \in \{k_0,...,T-1\}
\end{aligned}
$$

Pero a la vez se desprende una técnica inductiva hacia atrás (tambien llamado "backward induction") para deducir las propiedades de v_1 a partir de la función más simple v_T. El problema v_T no se define recursivamente ya que es la última etapa.

$$
\begin{aligned}
v_T(s_T) = & \ \min\ f_T(d_T,s_T) \\
& s.a.\ d_T \in D_T
\end{aligned}
$$

Ya habiendo definido el proceso inductivo, a través del trabajo del autor Richard Bellman, el Principio de Optimalidad se formalizó en la Ecuación de Optimalidad de Bellman que representa los problemas "cola" del problema y es dado como sigue:

$$
\begin{aligned}
v_n(s_n) = \min_{d_n}\ & \{f_n(d_n,s_n) + v_{n+1}(s_{n+1})\} \\
& s.a.\ s_{n+1} = t_n(s_n,d_n) \\
& \quad d_n \in D_n(s_n),\ s_{n+1} \in S_{n+1}
\end{aligned} \tag{3.1}
$$

donde

- $n \in \{1,...,T-1\}$.

- $v_n(s_n)$ = Es el valor optimal de todas las decisiones subsiguientes, dado que estamos en el estado s_n con n etapas por recorrer.

Observación 3.1.1 *En caso S_k sea continuo, $v_k(s_k)$ retorna puntos óptimos para cada s_k; sin embargo, por la continuidad de los valores de S_k, v_k debe ser una función continua por lo que se opta por interpolar esos puntos óptimos mencionados para así obtener la función continua v_n. Por otro lado en caso los valores sean discretos, la definición de S_k asegura que las variables de estado en la siguiente etapa estén en S_{k+1}.*

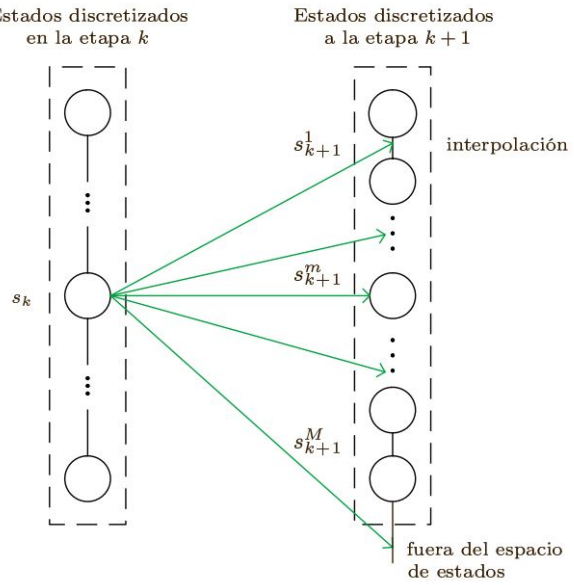


Figura 3.1: Cambio de variables de estado con el tiempo

En un problema de programación dinámica, si el sistema está en el estado s_n con n etapas por recorrer y la decisión d_n es escogida desde el conjunto de decisiones permisibles para esa etapa y estado, entonces la etapa retorna $f_n(d_n, s_n)$ y el estado del sistema en la etapa anterior $s_{n+1} = t_n(d_n, s_n)$, donde las dos son conocidas por estar en un problema determinístico.
Al final del problema, la política óptima se puede representar como:

$$\pi^* = \{d_0^*, d_1^*, ..., d_{N-1}^*\}$$

donde d_k^* es el valor de d_k que optimiza el problema 3.1 en la etapa k para cada s_k y para todo k; es decir,

$$d_k \in \underset{d_k \in D_k}{argmin}\{f_k(d_k, s_k) + v_{k+1}(s_{k+1})\}$$

De las variables de estado se desprende un árbol de decisión. En un árbol de decisión cada nodo representa un estado, y los arcos, decisiones. Para cada posible estado en cada etapa debemos crear un arco por cada posible decisión.

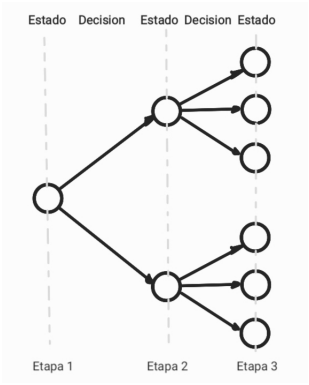


Figura 3.2: Árbol de decisión de un problema con 3 etapas

3.1.2. Maldición de la dimensionalidad

La Programación Dinámica a menudo está destinada a sufrir "La maldición de la dimesionalidad". La dimensionalidad se refiere al vasto crecimiento de los cálculos requeridos cuando el número de variables de estado incrementan. Si consideramos un problema con n variables de estado que se discretiza en m valores, entonces los cálculos por etapa en cada subproblema m^n.

Nro. de variables de estado	Nro. de operaciones
2	625
4	390625
6	24414.625
8	$1.526 \cdot 10^{11}$
10	$9.537 \cdot 10^{13}$

Tabla 3.1: Número de operaciones de 2 variables de estado, donde cada variable de estado toma 25 valores distintos.

3.2. Aplicación al Despacho Hidrotérmico

3.2.1. Planeamiento y arquitectura hidrotérmica

Planear la generación en sistemas hidrotérmicos con base hidráulica es complejo y amplio en dimensión, que van desde la planificación anual hasta la programación diaria de los embalses. Debido a la decisión operativa del sistema se pueden obtener consecuencias mostradas en la Figura 3.3 El operador tiene la opción de usar energía hidroeléctrica hoy, y en tal caso reducir los costos de la energía térmica a complementar, o almacenarla para usarla en la siguiente etapa. Si la decisión hoy es usar la energía de base hidroeléctrica y en el futuro los caudales son altos (lo que permite llenar los embalses) la operación se dice eficiente. Sin embargo, si ocurre una sequía en el futuro, los embalses no se recuperarán, y será necesario utilizar generación térmica

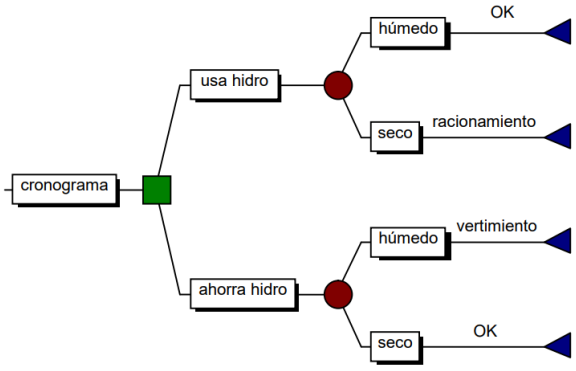


Figura 3.3: Toma de decisiones para el despacho hidrotérmico

más cara, o hasta interrumpir el suministro de la demanda.
Por otro lado, si la decisión de hoy es almacenar el agua para uso futuro a través del uso de más generación térmica, y los caudales futuros son altos sera necesario verter el agua, lo que significa un desperdicio de energia. Sin embargo, se ocurre una sequía en el futuro, el almacenamiento se usará para evitar la generación más cara o un racionamiento de energía.

3.2.2. Función de costo futuro

Se le denomina Función de costo futuro a la función α que representa el costo de generación térmica de la etapa $t+1$ hasta el final del periodo de estudio. Se observa que la función de costo futuro aumenta con el aumento del volumen turbinado, pues menos energía hidroeléctrica estará disponible en el futuro para desplazar la generación térmica.

El costo inmediato se refiere al costo de generación térmica mientras que el costo futuro se le conoce como Valor de agua. El valor del agua es el resultado del cálculo de la política de explotación óptima y representa el valor de la generación térmica más el déficit de suministro. A partir de esta parametrización, es posible representar una central hidráulica como una "central térmica" cuyo coste de explotación es el valor del agua (coste que posteriormente daremos nombre a costo de complementación térmica).

A modo de ejemplo, el impacto en los costos inmediatos y futuros según cuánta agua se turbina, se puede apreciar en la Figura 3.4.

Usando la Programación Dinámica se evita el inconveniente de desconocer esta función de costo futuro y así poder llegar al objetivo.

Como el objetivo es minimizar la suma de los costos operativos inmediatos y futuros, nos encontramos ante una función objetivo convexa. Ademas, el mínimo es obtenido cuando la derivada parcial con respecto al volumen turbinado es 0; es decir, se cumple

$$\frac{\partial FCT}{\partial V} = \frac{\partial FCI + FCF}{\partial V} \Rightarrow \frac{\partial FCI}{\partial V} = -\frac{\partial FCF}{\partial V}$$

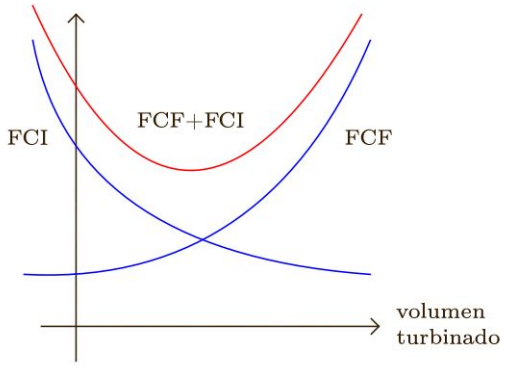


Figura 3.4: Función de Costo Inmediato y Futuro

3.2.3. Modelo

El despacho hidrotérmico con solo 1 reservorio, J plantas térmicas y con T etapas busca minimizar los costos de operación, como para cada etapa se turbina u_t entonces el costo total a minimizar es la suma de los costos de complementacion térmica considerando al volumen turbinado como una planta generadora más; es decir, el costo a minimizar es $c_1(u_1)+c_2(u_2)+...+c_T(u_T)$, donde el costo de complementacion termica se define como sigue

$$\begin{aligned} c_t(u_t) = \min_{g_t(1),...,g_t(J)} \quad & \sum_{j=1}^{J} c_t(j) \cdot g_t(j) \\ s.a. \quad & \sum_{j=1}^{J} g_t(j) + \rho u_t = d_t, \quad t = 1, ..., T \\ & g_t(j) \leq \overline{g}(j), \quad j = 1, ..., J, \quad t = 1, ..., T \end{aligned}$$

donde

Parámetros:

j	indexa las generadoras térmicas
J	cantidad de generadoras térmicas
$c_t(j)$	costo unitario de operacion del $j-$ésimo generador en la etapa t
d_t	demanda del sistema en la etapa t
$\overline{g}(j)$	capacidad del $j-$ésimo generador
ρ	coeficiente de producción de la planta hidroeléctrica
u_t	volumen turbinado en la etapa t

Variables:

$g_t(j)$	energía producida por el $j-$ésimo generador en la etapa t

El volumen turbinado se obtiene por la dinámica del reservorio. El modelo hidrotérmico se

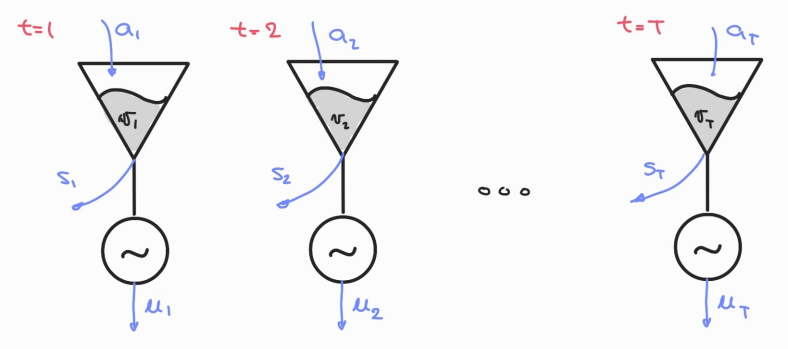


Figura 3.5: Cambio en los volumenes del reservorio en cada etapa

formula como sigue

$$
\begin{aligned}
z^* = \min_{u_t, s_t} \quad & \sum_{t=1}^{T} c_t(u_t) \\
\text{s.t.} \quad & v_{t+1} = v_t - u_t - s_t + a_t, \ t \in \{1, 2, ..., T-1\} \\
& 0 \leq u_t \leq \overline{u}, 0 \leq s_t \quad \forall t \in \{1, 2, ..., T\}
\end{aligned} \tag{3.2}
$$

La variable v_t, el volumen del reservorio en la etapa t, es una variable continua y a la vez es una variable de estado. Así nos interesa discretizar esta variable continua para trabajar solo con algunos de sus valores.
Considando T etapas, el sistema de estados está formado por todos los v_t^m con $m = 1, ..., M$ que toman los valores de niveles de almacenamiento $100\,\%, 90\,\%$, etc hasta $0\,\%$. Podemos escribir el problema 3.5, usando la ecuación de recursion Bellman es

$$
\alpha_t(v_t^m) = \min_{u_t, s_t} \ c_t(u_t) + \alpha_{t+1}(v_{t+1}) \tag{3.3}
$$

$$
\begin{aligned}
s.a. \ & v_{t+1} = v_t - u_t - s_t + a_t \\
& v_{t+1} \leq \overline{v} \\
& u_t \leq \overline{u} \\
& s_t \geq 0
\end{aligned} \tag{3.4}
$$

para cada $m = 1, ..., M$. La manera de construir $\alpha_t(v_t)$ es interpolando los puntos $\alpha_t(v_t^m), m \in \{1, ..., M\}$.
donde

Parámetros:

$\overline{v}$	almacenamiento máximo
$\overline{u}$	cantidad de turbinamiento máximo
a_t	caudal en la etapa t
v_t^m	volumen de almacenamiento m en la etapa t

Variables:

s_t	vertimiento en la etapa t

u_t volumen turbinado en la etapa t

Nuestro problema es minimizar, bajo las restricciones de dinámica y limitaciones máximas y mínimas, los costos de complementación térmica los cuales resultan otro problema de minimización. Estos costos, $c_t(u_t)$, son problemas de minimización fijando la variable de turbinamiento u_t. Entonces se puede ver al problema de despacho hidrotérmico como un problema proyectado en las variables de turbinamiento. Para fines computacionales podemos tener un solo problema de minimización como el que se muestra a continuación

$$
\begin{aligned}
z^* = \underset{u_t, s_t, g_t(1), \ldots, g_t(J)}{\text{minimize}} \quad & \sum_{t=1}^{T} \sum_{i=1}^{J} c_t(i) \cdot g_t(i) \\
\text{s.t.} \quad & v_{t+1} = v_t - u_t - s_t + a_t, \ t \in \{1, 2, ..., T\} \\
& 0 \leq u_t \leq \overline{u}, 0 \leq s_t \quad t \in \{1, 2, ..., T\} \\
& \sum_{j=1}^{J} g_t(j) + \rho u_t = d_t, \quad t = 1, ..., T \\
& g_t(j) \leq \overline{g}(j), \quad j = 1, ..., J, \quad t = 1, ..., T
\end{aligned}
\tag{3.5}
$$

Y así la recursión para el modelo (el problema 3.3) es

$$
\begin{aligned}
\alpha_t(v_t^m) = \min_{u_t, s_t} \quad & \sum_{i=1}^{J} c_t(i) \cdot g_t(i) + \alpha_{t+1}(v_{t+1}) \\
s.a. \ & v_{t+1} = v_t - u_t - s_t + a_t \\
& v_{t+1} \leq \overline{v} \\
& u_t \leq \overline{u} \\
& s_t \geq 0 \\
& \sum_{j=1}^{J} g_t(j) + \rho u_t = d_t, \\
& g_t(j) \leq \overline{g}(j), \quad j = 1, ..., J,
\end{aligned}
\tag{3.6}
$$

para cada $m = 1, ..., M$

Observación:

- El nivel de almacenamiento en el inicio de la primera etapa es conocido y así podemos comenzar la recursión de manera Backward usando la Ecuación de Bellman.
- La restricción 3.4 representa el balance hídrico y a la vez, la función de transición (cambio de estado entre etapas).
- Al resolver el problema 3.3 se obtienen soluciones para cada estado (M en total). Se opta por interpolar para obtener la función de costo de manera continua.

- El caudal termina siendo una variable con incertidumbre lo que vuelve al problema de determinístico a estocástico. Como es la Programación estocástica la que se encarga de estudiar modelos con incertidumbre, el modelo Hidrotérmico estocástico se tratará en el Capítulo 5.

3.2.4. Ejemplo

Consideremos un sistema compuesto por una planta hidráulica y dos plantas térmicas cuyas características se muestran en la Tabla 5.1 y Tabla 5.2.

Almacenamiento máximo (hm^3)	Almacenamiento mínimo (hm^3)	Turninado máximo (hm^3)	Coeficiente de producción ($MW-$mes$/hm^3$)
120	20	50	0.9

Tabla 3.2: Características de la planta hidrotérmica

La carga del sistema, la demanda, es de $45MW-$mes para todas las etapas y la penalidad por fallo al suplir la demanda, que es representada por una planta térmica artificial, es de $1000\$/MW-$mes. Además, los caudales para cada etapa se muestran en la Tabla 5.3.

Térmica	Máxima generación (MW)	Costo de operación ($\$/MW-mes$)
T_1	20	10
T_2	25	20

Tabla 3.3: Características de las plantas térmicas

Para este ejemplo el almacenamiento tendra 3 discretizaciones, las cuales son $100\,\%, 50\$, 0\,\%$.

Etapa	Caudal (hm^3/mes)
1	18
2	17
3	14

Tabla 3.4: Escenarios del caudal para cada etapa

El primer paso consiste en el cálculo del coste óptimo para cada nivel de discretización.
La ecuación de recursión queda como sigue

$$\begin{aligned} \alpha_t(v_t^k) = \min_{u_t} \; & c_t(u_t) + \alpha_{t+1}(v_{t+1}) \\ s.a. \; & v_{t+1} = v_t^k - u_t - s_t + a_t \\ & 0 \leq v_{t+1} \leq 120 \\ & u_t \leq 50 \end{aligned}$$

donde

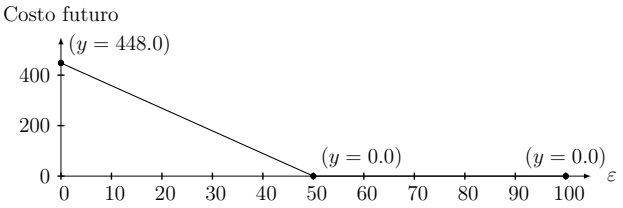
$$
\begin{aligned}
c_t(u_t) = \min_{g_1,g_2,\delta} \; & 10 \cdot g_t(1) \cdot 20 \cdot g_t(2) + 1000 \cdot \delta \\
s.a. \; & g_t(1) + g_t(2) + 0.9u_t + \delta = 45, \;\; t = 1, ..., 3 \\
& g_t(1) \leq 20, \;\; g_t(2) \leq 25, \quad t = 1, ..., 3
\end{aligned}
$$

y la Función de costo futuro $\alpha_t(v_t)$ es la interpolación de los valores $\{v_t^1, v_t^2, v_t^3\}$

Comenzamos la recursión con la etapa 3, cuyos valores optimales y función de costo futuro están en la Figura 3.6.
Pasamos a la etapa 2 donde ya conocemos los valores de $\alpha_3(v_3)$ y obtenemos la solución óptima como se muestra en la Figura 3.7. Finalmente los resultados y función de costo futuro en la etapa 1 se muestran en la Figura 3.6.
En resumen, la función de costo futuro aproximada para todo el planeamiento considerado se muestra en la Figura 3.8 (b). Lo que significa que si el nivel del reservorio estaba en un 50 %, el costo esperado para que el sistema opere en las tres etapas y considerando los dos escenarios es de $601.97.
La implementación en el lenguaje Gams se encuentra en el capítulo Implementaciones

Almacenamiento (%)	v_3	0	50	100
Caudal (hm^3)	a_3	14	14	14
Decisión Optimal	v_4	0	14	64
	u_3	14	50	50
	s_3	0	0	0
	$g_3(1)$	20	0	0
	$g_3(2)$	12.4	0	0
	δ	0	0	0
Costo optimal ($)		448.0	0.0	0.0

(a) Función de costo futuro

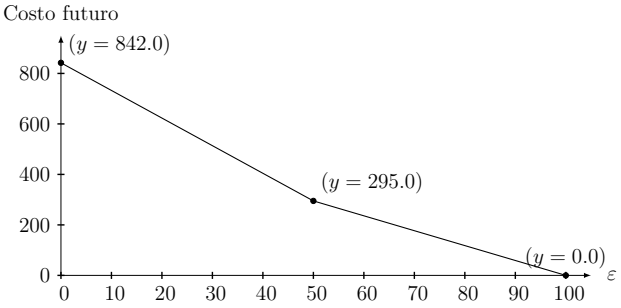


(b) Resultados

Figura 3.6: Resultados etapa 3

Almacenamiento (%)	v_2	0	50	100
Caudal (hm^3)	a_2	17	17	17
Decisión Optimal	v_3	0	50	50
	u_2	17	50	50
	s_2	0	0	0
	$g_2(1)$	20	0	0
	$g_2(2)$	9.7	0	0
	δ	0	0	0
Costo inmediato ($)		394.0	0.0	0.0
Costo futuro ($)		484.0	295.0	0.0
Costo optimal ($)		842.0	295.0	0.0

(a) Función de costo futuro

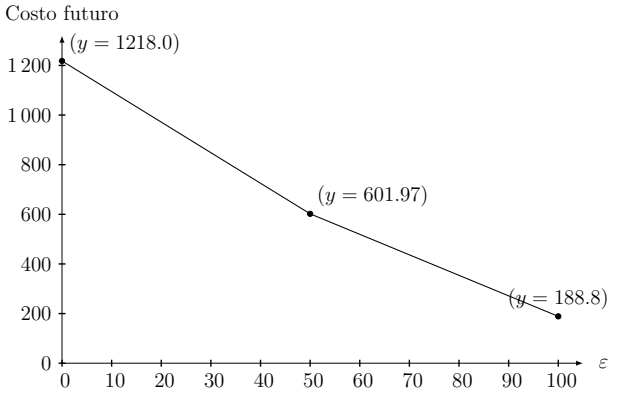


(b) Resultados

Figura 3.7: Resultados etapa 2

Almacenamiento (%)	v_1	0	50	100
Caudal (hm^3)	a_1	18	18	18
Decisión Optimal	v_2	0	40.2	68
	u_2	18	27.8	50
	s_1	0	0	0
	$g_1(1)$	20	20	0
	$g_1(2)$	8.8	0	0
	δ	0	0	0
Costo inmediato ($)		376.0	200.0	0.0
Costo futuro ($)		914.0	401.97	188.8
Costo optimal ($)		1218.0	601.97	188.8

(a) Función de costo futuro



(b) Resultados

Figura 3.8: Resultados etapa 1

Capítulo 4

Programación Dinámica Dual

Como ya se vió, la Programación Dinámica, considerando T etapas, busca resolver T subproblemas en recursión para obtener el valor optimal. A primera vista no resulta tan provechoso ya que conlleva resolver T problemas, a pesar de que el espacio de estados pueda ser pequeño se incurre en un coste computacional relativamente alto, debido a eso se opta por resolver el problema con información del dual. La ventaja de usar Programación Dual Dinámica es que el problema no requiere una representación exacta de la función de costo futuro, solo una que sea considerado suficientemente buena y cuya aproximación se construye con información de las variables duales. Esto significa que solo un subconjunto de todas las restricciones lineales, que proporcionan una aproximación a la función de costo futuro, necesita ser encontrada.
Si el problema Lineal tiene como matriz tecnológica a la matriz en la Figura 4.1, fijando la variable que corresponde a la primera columna de bloques, el problema que queda es separable por bloques, en otras palabras, es más sencillo de resolver

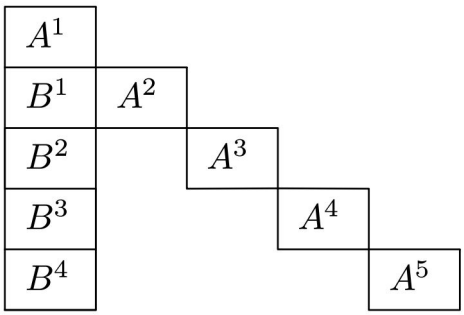


Figura 4.1: Matriz tecnológica conocida como Bloque angular

Por otro lado, si consideremos un problema donde su matriz tecnológica sea la de la Figura 4.2 podemos ver que existen restricciones que enlazan a más de dos subperiodos de tiempo. Estas restricciones pueden clasificarse en intertemporales y estáticas. Las intertemporales enlanzan variables que no pertenecen a un mismo subperiodo de tiempo como el balance hídrico o tiempos mínimos de operación de unidades térmicas; mientras que las estáticas si las enlazan como los límites mínimos y máximos de operación de las unidades térmicas, el balance nodal, etc.

Así si fijamos las intertemporales se obtiene un problema que resulta sencillo de resolver como el poder descomponerse en subproblemas. Por lo tanto, se debe seleccionar el conjunto de restricciones y variables complicantes del problema para poder lograr que el problema pueda separarse o, encontrar estructuras conocidas. [7]

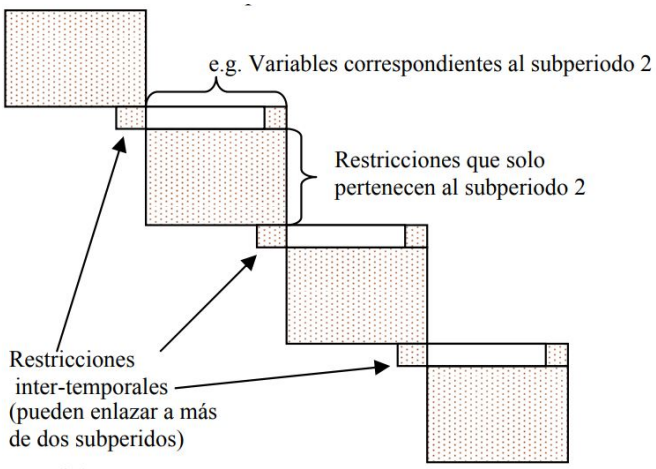


Figura 4.2: Matriz tecnológica [7]

Para un problema de dos etapas el cálculo de las restricciones lineales es hecha por la Descomposición de Benders.

4.1. Descomposición de Benders

Este método de descomposición considera 2 etapas. Para saber que el método puede sernos de utilidad, hacer que el problema sea más sencillo de resolver, el problema debe tener variables complicantes. Las variables complicantes son aquellas que cuando son fijadas, se obtiene un problema más sencillo de resolver.

Consideremos el problema

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + d^t y \\ s.a. \quad & F(x) + By \geq e \qquad (P) \\ & x \in \mathcal{X} \\ & x \geq 0,\ y \geq 0 \end{aligned} \tag{4.1}$$

donde x viene a ser nuestra complicante (fijando x el problema es más sencillo de resolver) y $\mathcal{X}$ es un cono poliedral.

Considerando $\alpha : \mathbb{R}^n \to \overline{\mathbb{R}}$, conocida como la función valor, definida como sigue

$$\begin{aligned} \alpha(x) = \min_y \ & d^t y \\ s.a. \ & By \geq e - F(x) \\ & y \geq 0 \end{aligned} \tag{4.2}$$

donde $\text{dom}\,(\alpha) = \{x : \exists\, y, F(x) + By \geq e, y \geq 0\}$.

Proposición 4.1.1 *El problema 4.1 es equivalente a*

$$\begin{aligned} min \ & f(x)+\alpha(x) \\ s.a. \ & x \in \mathcal{X} \qquad (\overline{P}) \\ & x \geq 0 \end{aligned}$$

En particular $(x^*, y^*) \in sol(P)$ *si y solo si* $x \in sol(\overline{P})$ *y* $y \in sol(Sub)$

Prueba: Si $(\overline{P})$ es infactible, no acotado y tiene solucion optimal si y solo si (P) es infactible, no acotado y tiene solución optimal. Para un x fijado, $z \in sol(\overline{P})$, es igual a $f(x)$ más α con α siendo solución de 4.2.
Sea $(\overline{x}, \overline{y}), \hat{x}, \hat{y}$ soluciones optimales de (P), $(\overline{P})$ y 4.2 respectivamente. Entonces particularmente $(\hat{x}, \hat{y})$ es solución factible y se tiene

$$f(\hat{x}) + b^t\hat{y} \leq f(\overline{x}) + b^t\overline{y}$$

Por otro lado

$$f(\hat{x}) + b^t\hat{y} \geq f(\overline{x}) + \min_y\{b^t y | By \geq e - F(\overline{x}), y \geq 0\}$$

asi $\overline{y}$ es solución optimal de 4.2. Entonces se cumple que

$$f(\overline{x}) + \min_y\{b^t y | By \geq e - F(\overline{x}), y \geq 0\} = f(\overline{x}) + b^t\overline{y}$$

Finalmente se tiene

$$f(\hat{x}) + b^t\hat{y} \geq f(\overline{x}) + b^t\overline{y}$$

y junto con la desigualdad anterior se cumple la igualdad; es decir, los problemas (P) y $(\overline{P})$ comparten soluciones. □

En caso $dom(\alpha) = \varnothing$, el problema es infactible.

Proposición 4.1.2 *Cuando* $F(x)$ *sea convexa y* $dom(\alpha)$ *es no vacío entonces la función* α *es convexa.*

Prueba: Definamos $adm(\alpha(x)) = \{By \geq e - F(x),\ y \geq 0\}$. Sean y_1 y y_2 soluciones de $\alpha(x)$ y $\alpha(z)$ respectivamente.

$$\alpha(x) = d^t y_1 \text{ donde } d^t y_1 \leq d^t y \ \ \forall y \in adm(\alpha(x)) \text{ y}$$
$$\alpha(z) = d^t y_2 \text{ donde } d^t y_2 \leq d^t y \ \ \forall y \in adm(\alpha(z))$$

Particularmente se tiene que $y_1 \in adm(\alpha(x))$ y $y_2 \in adm(\alpha(z))$ entonces

$$By_1 \geq e - F(x)\ ,\ y_1 \geq 0 \text{ y} \tag{4.3}$$
$$By_2 \geq e - F(z)\ ,\ y_2 \geq 0 \tag{4.4}$$

Sea $\lambda \in \langle 0, 1\rangle$. Multiplicamos a las restricciones de 4.3 por λ, a las de 4.4 por $(1-\lambda)$ y sumamos obteniendo

$$\lambda By_1 + (1-\lambda)By_2 \geq \lambda e + (1-\lambda)e - \lambda F(x) - (1-\lambda)F(z) \quad \wedge\ \lambda y_1 + (1-\lambda)y_2 \geq 0$$
$$B(\lambda y_1 + (1-\lambda)y_2) \geq e - F(\lambda x + (1-\lambda)z)) \quad \wedge\ \lambda y_1 + (1-\lambda)y_2 \geq 0$$

Así $\lambda y_1 + (1-\lambda)y_2 \in adm(\alpha(\lambda x + (1-\lambda)z))$ y se cumple

$$\begin{aligned}\alpha(\lambda x + (1-\lambda)z) &\leq d^t(\lambda y_1 + (1-\lambda)y_2)\\ &\leq \lambda d^t y_1 + (1-\lambda)d^t y_2\\ &\leq \lambda\alpha(x) + (1-\lambda)\alpha(z)\end{aligned}$$

Concluyendo que la función α es convexa. □

Suponiendo que la función α es convexa, se puede aproximar por el máximo de cartas obtenidas. En la práctica esta aproximación es iterativa y así mejora la función en cada iteración y esta aproximación iterativa es la base del algoritmo de Bender.

El algoritmo tiene los siguientes pasos

1. Obtener un punto factible inicial resolviendo el problema cinsiderando $y = 0$
2. Resolver el Subproblema y obtener una carta que se añade al problema Maestro como una aproximación inicial de α
3. Resolver el Problema Maestro Restricto usando la aproximación de $\alpha(x)$
4. Resolver el Subproblema y obtener una carta, la carta que se añade ahora al Problema maestro es la carta máxima de la carta anterior y la carta que queremos añadir.
5. Si la aproximación es suficientemente buena el algoritmo se detiene, caso contrario volvemos al Paso 3.

4.1.1. Subproblema

Es el problema que resulta de fijar la variable x en el problema (4.1)

$$\begin{aligned}\alpha(x) = \min_y\ & d^t y && (4.5)\\ s.a.\ & By \leq e - F(x)\ :\ \lambda \qquad (Sub)\\ & y \geq 0\end{aligned}$$

donde λ son las variables duales de las restricciones correspondientes.

Subproblema Dual

La variable dual de un problema como el problema dual de 4.5, nos ayuda a hallar la subdiferencial de la función objetivo con respecto a la variable en cuestión, por lo que la dualidad nos permite construir una restricción lineal con la solución óptima del problema y el valor de la variable de prueba. Así, la descomposición de Benders calcula restricciones lineales, también conocida como Cortes de Bender, para aproximar el subproblema $\alpha(x)$. Consideremos el dual del Subproblema

$$\begin{aligned} \tilde{\alpha}(x) = \max_{\lambda} \ & (e - F(x))^t \lambda \\ s.a. \ & B^t \lambda \leq d \qquad (D) \\ & \lambda \geq 0 \end{aligned} \tag{4.6}$$

Se puede ver que el conjunto factible es independiente de x. Veamos los casos posibles en la solución del Subproblema Dual.

Caso 1: Si el conjunto factible de (D) es vacío entonces el (Sub) es no acotado para algún x, lo que inclica que (P) sea no acotado ($\alpha(x) = -\infty$), o el problema (Sub) es infactible para todo x, en tal caso el problema original (P) sería infactible.

Caso 2: Si el conjunto factible de (D) es no acotado entonces el problema (Sub) es no actoado, así (P) es no acotado.

Caso 3: Si el conjunto factible del problema dual 4.6; es decir (D) es no vacío y acotado; es decir compacto, tendríamos factibilidad en el problema dual $\tilde{\alpha}(x)$ y por definición factibilidad en $\alpha(x)$, entonces tales problemas son equivalentes y alcanzan el mismo valor óptimo.

Asumiremos a partir de ahora que $S \neq \varnothing$ y así $\alpha = \hat{\alpha}$. Además, podemos ver que al conjunto factible de $\tilde{\alpha}(x)$ como un poliedro

$$S = conv\{\lambda_1, ..., \lambda_r\} + cone\{d_1, ..., d_s\}$$

donde el óptimo, si se alcanza, se alcanzará en un elemento del conjunto.

Observación 4.1.1 *Aún no tenemos la certeza de que (D) sea acotado y alcance su óptimo, es por eso que existen direcciones extremas dejando la posibilidad de que el problema sea no acotado.*

Lema 4.1.1 *Dado x, existe una solución optimal del problema (D) si y solo si*

$$(e - F(x))^t d_j \geq 0 \quad , d_j \text{ sea dirección extrema} \tag{4.7}$$

Prueba:

($\Rightarrow$) Si no se cumple 4.7 entonces existe algún d_k dirección extrema tal que

$$(e - F(x))^t d_j < 0 \quad , d_j$$

entonces existe una dirección descendente extrema, así el problema (D) sería no acotado por tanto no tiene solución ($\Rightarrow\Leftarrow$).

($\Leftarrow$) Si se cumple 4.7 no existe uan dirección el cual podamos descender sin salirnos de la region factible, entonces existe solucion optimal para (D).

□

Proposición 4.1.3 *Se cumple que* $\alpha(x) = \sigma_s(e - F(X))$, *siendo*

$$\sigma_S(\overline{v}) = \begin{cases} max\{\lambda^t \overline{v} \ / \ i = 1, ..., r\} & , \ si \quad \overline{v} \in K \\ +\infty & , \ si \quad \overline{v} \notin K \end{cases} \tag{4.8}$$

con $K = \{\overline{v} \ / \ d^t \overline{v} \geq 0 \ \forall i = 1, ..., s\}$

Además cuando $(e - F(x)) \in K$, el problema 4.6 se puede escribir como

$$\begin{aligned} \alpha(x) = \min \ & \alpha \\ s.a. \ & \alpha \geq (e - F(x))^t \lambda^1 \\ & \alpha \geq (e - F(x))^t \lambda^2 \\ & \vdots \qquad\qquad \vdots \\ & \alpha \geq (e - F(x))^t \lambda^r \end{aligned} \tag{4.9}$$

En el problema 4.9 α es mayor o igual que cada $(e - F(x))^t \lambda^i$, $i = 1, ..., r$, particularmente es mayor o igual que $max\{(e - F(x))^t \lambda^i, i = 1, ..., r\}$ y como el problema es de minimización, el valor que debe tomar $\alpha(x)$ es max $[(e - F(x))^t \lambda^i$ para todo $i \in 1, ..., r]$.

Proposición 4.1.4 *Cuando* $F(x)$ *sea lineal, la función* $\alpha(x)$ *es lineal por partes*

Prueba: La representación que tiene $\alpha(x)$ de 4.9 permite ver que el problema es lineal por partes, por otro lado por definición y usando la formulación Ecuación 4.8 también se tiene lo pedido. □

Así cuando $F(x)$ sea lineal, por las Proposiciones 4.1.2 y 4.1.4 se tiene que la función $\alpha(x)$ es una función convexa y lineal por partes.

4.1.2. Problema Maestro:

El problema maestro, con los cortes añadidos par aproximar α, queda como sigue

$$\begin{aligned} \min_{x} \ & c^t x + \alpha && (4.10) \\ s.a. \ & x \in \mathcal{X} \\ & \alpha \geq (e - F(x))^t \lambda^i \quad i = 1, ..., r && (4.11) \\ & 0 \geq (e - F(x))^t d^j \quad j = 1, ..., s \\ & x \geq 0 \end{aligned}$$

Teorema 4.1.1 *$(\overline{x}, \overline{y})$ es solución optimal de 4.1 si y solo si $\overline{y}$ es solución optimal de (Sub) y $(\overline{x}, \alpha)$ es solución optimal de el problema Maestro* 4.12

Prueba: Si existe solución en (P) y $(\overline{P})$ y los problemas son equivalentes, por Teorema se dualidad fuerte se tiene que tienen el mismo valor optimal,

$$\min_{x} \{d^t y \ / \ By \geq e - F(x), y \geq 0\} = \max_{\lambda} \{(e - F(x))^t \lambda \quad B^t \lambda \leq d, \lambda \geq 0\}$$

Reemplazamos

$$\begin{aligned} \min \ & f(x) + \max_{\lambda} \{(e - F(x))^t \lambda \quad B^t \lambda \leq d, \lambda \geq 0\} \\ s.a. \ & x \in \mathcal{X} \qquad (\overline{P}) \\ & x \geq 0 \end{aligned}$$

Como existe solución para (D), esta se alcanza en un vértice. Entonces se cumple que

$$\max_{\lambda} \{(e - F(x))^t \lambda \mid B^t \lambda \leq d, \lambda \geq 0\} = \max_{\lambda_i} \{(e - F(x))^t \lambda_i\}$$

Sustituyendolo en el problema (P) se cumple

$$\begin{aligned} \min_{x} \ & f(x) + \max_{\lambda_i} \{(e - F(x))^t \lambda_i\} && (4.12) \\ s.a. \ & x \in \mathcal{X} \\ & 0 \geq (e - F(x))^t d^j \quad j = 1, ..., s \\ & x \geq 0 \end{aligned}$$

Lo que nos da la equivalencia pedida. □

Gracias a la formulación tenemos otra manera de resolver 4.1; sim embargo, hallar los puntos extremos y direcciones extremas de un problema, en este caso del problema dual, es costoso por

lo que podemos definir al Problema Maestro restricto para un subconjunto de estos conjuntos de puntos mencionados. Considerando $U' \subset \{\lambda^1, ..., \lambda^r\}$ y $V' \subset \{d^1, ..., d^s\}$.

$$\begin{aligned} \min_{x,\alpha} \ & c^t x + \alpha \\ s.a. \ & x \in \mathcal{X} \\ & \alpha \geq (e - F(x))^t \lambda^i \quad \lambda^i \in U' \\ & 0 \geq (e - F(x))^t d^j \quad d^j \in V' \\ & x \geq 0 \end{aligned} \tag{4.13}$$

Para la interrogante de cuántos puntos extremos o cuántas direcciones extremas debemos tener, es que se enuncia el siguiente teorema.

Teorema 4.1.2 *Sea $(\overline{x}, \overline{\alpha})$ una solución optimal del Problema Maestro restricto. Si λ^* es el vertice optimal de la solucion y z^* el valor optimal del Dual $\alpha(x)$ entonces*

(i) Si $\overline{\alpha} > z^*$ entonces λ^* es un nuevo vértice generado

(ii) Si $\overline{\alpha} = z^*$, entonces $(\overline{x}, \overline{\alpha})$ es una solución optimal del Problema Maestro.

Si $\alpha(\overline{x})$ es no acotado entonces una nueva dirección extrema es dada. [8]

Prueba: Por la hipotesis se tiene

$$z^* = (e - F(x))^t \lambda^* = max \ \{(e - F(x))^t \lambda_i \, , \ \lambda^i \text{ punto extrema de } (D)\}$$

Además $\overline{x}$ satisface que $0 \geq (e - F(x))^t d^j$, con d^j dirección extrema, en otro caso existiría una dirección donde el mínimo decrece lo que haría al problema no acotado contradiciendo la hipotesis. Desde que $(\overline{x}, \overline{\alpha})$ es optimal se obtiene

$$\begin{aligned} f(\overline{x}) + \overline{\alpha} &= f(\overline{x}) + max \ \{(e - F(x))^t \lambda_i \, , \ \lambda^i \in U'\} \\ &\geq f(\overline{x}) + max \ \{(e - F(x))^t \lambda_i \, , \ \lambda^i \in U'\} \\ &= f(\overline{x}) + z^* \end{aligned}$$

Así solo se obtienen 2 casos:
Caso 1: $\overline{\alpha} > z^*$, en este caso

$$((e - F(x))^t \lambda^*) = z^* < \overline{\alpha} = max \ \{(e - F(x))^t \lambda_i \, , \ \lambda^i \in U'\}$$

Asi $\lambda^* \notin U'$ tenga que ser un nuevo vértice.
Caso2: $\overline{\alpha} = z^*$, donde se cumple

$$\overline{\alpha} = z^* \leq (e - F(x))^t \lambda_i \, , \ \lambda^i \text{ punto extremo}$$

Se puede ver que $(\overline{x}, \overline{\alpha})$ es una solución factible del Problema Maestro; más aún, es una solución optimal ya que es optimal del problema relajado .
Si el problema ahora es no acotado, existe una dirección extrema que satisface

$$0 < (e - F(x))^t d^*$$

Asi $(\overline{x}, \overline{\alpha})$ satisface las restricciones del problema maestro se tiene

$$0 \geq (e - F(x))^t d^j \quad d^j \in V'$$

por tanto $d^* \notin V'$ termina siendo una nueva dirección extrema. □

Cortes de Bender

Las restricciones del problema 4.9 se pueden ver en la Figura 4.3 donde cada restricción representa un corte de Bender que define un valor optimal del subproblema, $\alpha(x)$. Así la función de costo futuro se aproxima por abajo por los cortes de Benders. Si bien es cierto, a mayor número de cortes obtendremos mas precisión; pero a la vez, se incrementa el número de operaciones.

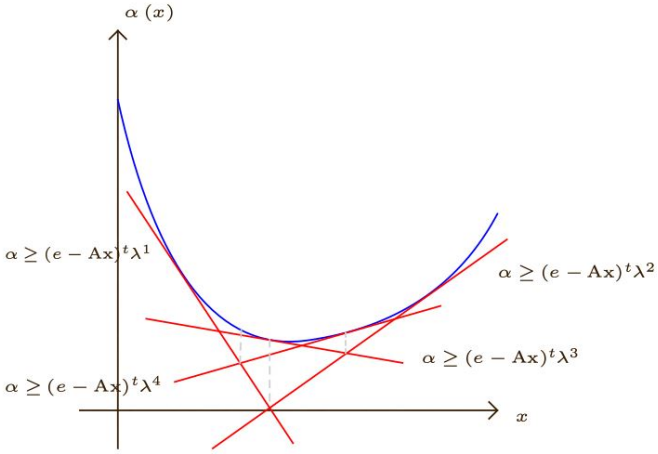


Figura 4.3: Cortes de Bender en 4 iteraciones

4.1.3. Convergencia

En la iteración k. Se puede ver que el problema maestro restricto es una versión relajada del problema original donde las funciones objetivos son iguales. Por eso, con la iteración el valor optimal del problema Maestro es una cota inferior del problema original.

$$z_l = c^t x^k + \alpha(x^k)$$

Además, los cortes de Benders cumplen que α esté arriba del corte pero el problema es de minimización. Entonces, los cortes son añadidos, el valor de α crece con las iteraciones (cada

vez hay más restricciones de tipo $\alpha \geq (e - Ax)^t \lambda^i$) y el problema se vuelve progresivamente menos relajado (más restricto), cuando al añadir restricciones no afecte el valor de α la cote inferior se mantiene ya que el valor de $\alpha(x_k)$ tampoco se afecta, lo que implica que la cota inferior sea monótonamente creciente con las iteraciones, mas no estricta.
Por otro lado el Subproblema es una versión restricta del problema original. Por eso, su valor objetivo es una cota superior del problema original.

$$z_u = c^t x^k + d^t y$$

Debido a que no se tiene la certeza de que el Subproblema nos de una cota superior decreciente, el problema se dice que converge cuando la diferencia de la cota superior e inferior sea lo suficientemente pequeña.

Observación: Como $z^u - z_l = \overline{\alpha} - z^*$, siendo $\overline{\alpha}$ solución del problema (4.15) y z^* siendo valor optimal del subproblema Dual, se le puede considerar como cota superior a $\overline{\alpha}$ y como cota inferior a z^*. Al fin y al cabo son cotas de α solución del Problem Maestro.

4.1.4. Algoritmo

Para resolver un problema de optimización usando la descomposición de Benders se usa el algoritmo que itera entre el Problema Maestro y Subproblema. En cada iteración un corte adicional es calculado, el corte de Bender, que es hallado con el dual del subproblema y es añadido como restricción adicional al problema maestro.

Entrada: Problema de programación lineal con variables complicantes, $\varepsilon > 0$ y pequeño.
Salida: Aproximación a solución óptima, x^k, y^k.

Paso 1: Hacemos $k = 1$, fijamos $z_l = 0$ y $z_u = \infty$.

Paso 2: Resolver el Problema Maestro

$$\begin{aligned} \min_x \ & c^t x \\ s.a. \ & x \in \mathcal{X} \\ & x \geq 0 \end{aligned} \tag{4.14}$$

Si la solución es infactible, **PARAR**; caso contrario, almacenamos $x = x^k$

Paso 3: Resolvemos el Subproblema añadiendo el Corte de Bender si hubiera.

$$\begin{aligned} \alpha(x) = \max_\lambda \ & (e - Ax)^t \lambda \\ s.a. \ & B^t \lambda \leq d \\ & \lambda \geq 0 \end{aligned} \tag{4.15}$$

Paso 3.1: Si tiene solución infactible entonces el problema principal es infactible o no acotado, **PARAR.**

Paso 3.2: Si la solución del subproblema en la iteración k con la variable dual λ^* y con valor optimal z^*.

Paso 3.2.1: Si $\overline{\alpha} > z^*$. Actualizar 4.16 añadiendo una restricción, asociado a un corte optimal, correspondiente al vertice λ^*

$$\alpha \geq (e - Ax)^t \lambda^i$$

e ir al **Paso 5**

Paso 3.2.2: Si $\overline{\alpha} = z^*$, **PARAR**

Paso 3.3: Si el problema es no acotado, actualizamos 4.16 introduciendo el corte factible asociado a una dirección extrema, en este caso q

$$(e - Ax)^t \lambda^q \leq 0, \quad q \in V$$

e ir al Paso 5.

Paso 4: Convergencia (brecha entre Problema Maestro y Subproblema)

Si $\overline{\alpha} - z^* \leq \varepsilon$, **PARAR**, en otro caso continuar con **Paso 5**.

Paso 5: Actualizar iteración $k \leftarrow k + 1$ y resolver el Problema Maestro Restricto

$$\begin{aligned} \min_{x,\alpha} \ & c^t x + \alpha \\ s.a. \ & x \in \mathcal{X} \\ & \alpha \geq (e - Ax)^t \lambda^i \quad \lambda^i \in U' \\ & 0 \geq (e - Ax)^t d^j \quad d^j \in V' \\ & x \geq 0 \end{aligned} \tag{4.16}$$

Si la solución es infactible el algoritmo se detiene; caso contrario, almacenamos el valor óptimo es $(x^k, \overline{\alpha})$.

Paso 6: Volver al **Paso 3**

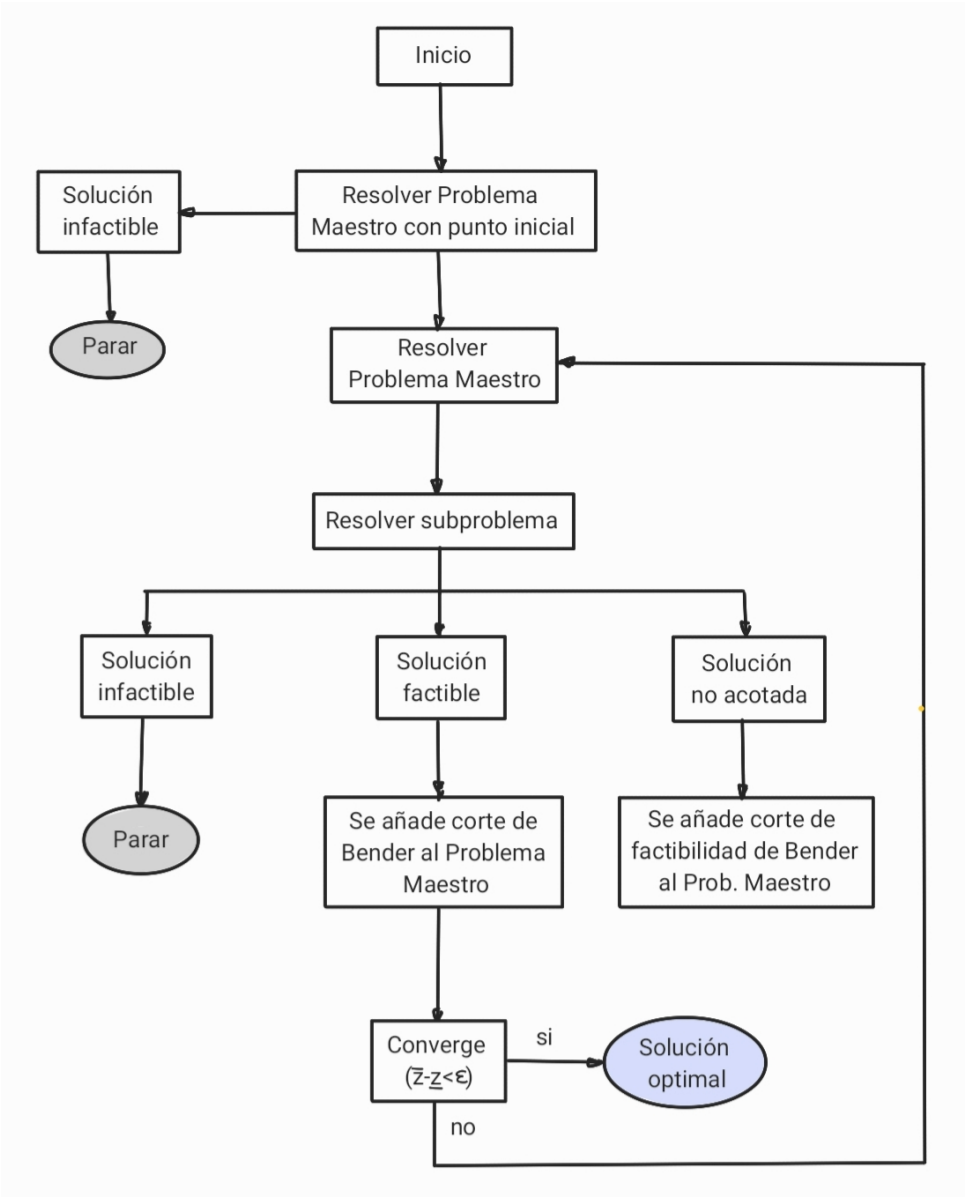


Figura 4.4: Diagrama de flujo para el Algoritmo de Bender

4.1.5. Despacho hidrotérmico con arranque

Extendamos el problema tomando en consideración el arranque de los despachos térmicos que deben cumplir lo siguiente

- Si en la etapa previa y la actual la planta térmica estaban (o no) en funcionamiento, entonces no se incurre en un costo.
- Si en la etapa previa la planta no estaba en funcionamiento y en la actual si lo está, entonces se incurre en un costo.
- Si la planta no está en funcionamiento no debe haber generación térmica.

Si consideramos a las variables e que representa el estado de la planta (prendido o apagado) y d que representa si se incurrirá el costo o no, las dos variables toman valores 0 o 1.

e_{t+1}	e_t	q_{t+1}
1	0	1
0	0	0
1	1	0
0	1	0

Tabla 4.1: Valores que deben tomar e_t y d_t

Las restricciones que se ajustan a lo pedido es, dado la etapa t y J plantas térmicas son:

$$\begin{aligned} e_{t+1,j} - e_{t,j} \leq d_{t+1,j} \quad \forall j \in \{1, ..., J\} \\ g_j \leq e_{t,j} \cdot \overline{g}_j \quad \forall j \in \{1, ..., J\} \end{aligned}$$

Así, cambia el costo de generación térmica $c_t(u_t)$ para el modelo 3.3

$$\begin{aligned} c_t(u_t) = \min_{g_t(1),...,g_t(J),q_{t,1},...,q_{t,J},e_{t,}} \quad & \sum_{j=1}^{J} c_t(j) \cdot g_t(j) + c_q \cdot q_{t,j} \\ s.a. \quad & \sum_{j=1}^{J} g_t(j) + \rho u_t = d_t, \quad j = 1, ..., J \\ & g_j \leq e_{t,j} \cdot \overline{g}_j, \quad j = 1, ..., J, \\ & 0 \leq e_{t,j} - e_{t-1,j} \leq q_{t,j} \quad j \in 1, ..., J \\ & e_{t,j}, q_{t,j} \in \{0, 1\} \end{aligned}$$

Entonces el problema es hallar z^*

$$\begin{aligned} z^* = \min_{u_t, s_t} \quad & \sum_{i=1}^{T} c_t(u_t) \\ \text{s.t.} \quad & v_{t+1} = v_t - u_t - s_t + a_t, \ t \in \{1, 2, ..., T-1\} \\ & u_t \leq \overline{u}, 0 \leq s_t \quad \forall t \in \{1, 2, ..., T\} \end{aligned} \tag{4.17}$$

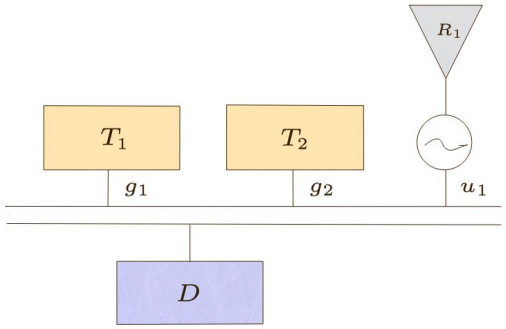


Figura 4.5: Sistema hidrotérmico

El problema sin proyectar, consideransdo 2 generadoras, es como sigue

$$
\begin{aligned}
z^* = \underset{u_t,s_t,g_t(1),g_t(2),q_{t,1},q_{t,2},e_{t,1},e_{t,2}}{\text{Minimize}} \quad & \sum_{i=1}^{T}(c_t(1)\cdot g_t(1) + c_t(2)\cdot g_t(2) + c_q \cdot q_{t,1} + c_q \cdot q_{t,2}) \\
\text{s.t.} \quad & v_{t+1} = v_t - u_t - s_t + a_t,\ t \in \{1,2,...,T\} \\
& u_t \leq \overline{u}, 0 \leq s_t \quad \forall t \in \{1,2,...,T\} \\
& g_t(1) + g_t(2) + \rho u_t = d_t \\
& g_t(1) \leq e_{t,1} \cdot \overline{g}_1, \quad t = 1,...,T, \\
& g_t(2) \leq e_{t,2} \cdot \overline{g}_2, \quad t = 1,...,T, \\
& 0 \leq e_{t+1,1} - e_{t,1} \leq q_{t,1} \quad t \in 1,...,T \\
& 0 \leq e_{t+1,2} - e_{t,2} \leq q_{t,2} \quad t \in 1,...,T \\
& e_{t,1}, q_{t,1} \in \{0,1\} \\
& e_{t,2}, q_{t,2} \in \{0,1\}
\end{aligned}
$$

con $e_{T+1,1}, e_{T+1,2}$ conocidos. En términos matriciales tendriamos:

$$
\begin{aligned}
z^* = \underset{x,y}{\text{Minimize}} \quad & c^t x + d^t y \\
s.a. \quad & Ax + By \leq e \\
& Cx \leq f \\
& Dy = g \\
& x \in \{0,1\}^{4t}, y \geq 0
\end{aligned}
$$

con

$$c = (c_q, ..., c_q, c_q, ..., c_q, 0, ..., 0, 0, ..., 0)$$

$$d = (0, ..., 0, 0, ..., 0, c_1(1), ..., c_t(1), c_1(2), ..., c_t(2))$$

$$C = \left(\begin{array}{c|c|c|c} -I_t & 0_t & P & 0_t \\ \hline 0_t & -I_t & 0_t & P \end{array}\right) \text{ con } P = \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & -1 \end{pmatrix}_{t\times t}$$

$$D = \left(\begin{array}{c|c|c|c} I_t & I_t & 0_t & 0_t \\ \hline \rho I_t & 0_t & I_t & I_t \end{array}\right)$$

$$x = (q_{11}, ..., q_{t1}, q_{12}, ..., q_{t2}, e_{11}, ..., e_{t1}, e_{12}, ..., e_{t2})$$

$$y = (u_1, ..., u_t, s_1, ..., s_t, g_1(1), ..., g_t(1), g_1(2), ..., g_t(2))$$

$$f = (0, ..., -e_{t+1,1}, 0, ..., -e_{t+1,2})_{2t}$$

$$g = (v_1 - v_2 + a_1, ..., v_t - v_{t+1} + a_t, d, .., d, d, ..., d)$$

$$A = \left(\begin{array}{c|c|c|c} 0_t & 0_t & -Q(1) & 0_t \\ \hline 0_t & 0_t & 0_t & -Q(2) \end{array}\right), \qquad B = \left(\begin{array}{c|c|c|c} 0_t & 0_t & I_t & 0_t \\ \hline 0_t & 0_t & 0_T & I_t \end{array}\right)$$

con

$$Q(t) = \begin{pmatrix} \overline{g}_t & 0 & \cdots & 0 \\ 0 & \overline{g}_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & \overline{g}_t \end{pmatrix}$$

$$e = (0, ..., 0)_{2t}$$

Así, tomamos como variable complicante a y, si la fijamos nos queda un problema de Programación Entera con variables binarias la cual es más sencilla de resolver.

El subproblema y su dual quedan como

$$\begin{aligned} \theta = \min_{x,y} \ & d^t y \\ s.a. \ & By \leq e - Ax \\ & Dy = g \\ & y \geq 0 \end{aligned} \qquad\qquad \begin{aligned} \theta = \max_{\lambda} \ & \begin{pmatrix} Ax - e & g & -g \end{pmatrix}^t \lambda \\ s.a. \ & \begin{pmatrix} -B^T & D^T & -D^T \end{pmatrix} \lambda \leq d \\ & \lambda \geq 0 \end{aligned}$$

El problema maestro restricto es el que sigue

$$\begin{aligned} \min_{x} \ & c^t x + \theta \\ s.a. \ & Cx \leq f \\ & \theta \geq \begin{pmatrix} Ax - e & g & -g \end{pmatrix}^t \lambda^i \quad \lambda^i \in U' \\ & 0 \geq \begin{pmatrix} Ax - e & g & -g \end{pmatrix}^t d^j \quad d^j \in V' \\ & x \in \{0,1\}^{4t} \end{aligned} \tag{4.18}$$

con U' y V' los subconjuntos de puntos y direciones extremas respectivamente.
Así podemos comenzar el algoritmo de Benders ya mencionado.

Como una extensión a la programación dinámica dual se tiene la Descomposicion de Benders Aninada o Descomposición de Benders Multietapa.

4.2. Descomposición de Benders multietapa

La descomposición de Benders se puede extender a problemas de multietapa. Aquí el problema se descompone en una serie de problemas maestros y subproblemas donde cada problema individual puede representarse tanto por un problema maestro como por un subproblema. La diferencia con el Algoritmo de Benders clásico es que se añade un pase hacia adelante (forward pass) para calcular los valores proeba de la variable complicada del problema Maestro y un pase hacia atrás (backward pass) para calcular las variables duales y así los Cortes de Bender del subproblema. [9] [1]

Consideremos un problema de programación lineal

$$\begin{aligned} &\min \sum_{t=1}^{T} c_t x_t \\ &s.a.\ A_t x_t \geq b_t - E_{t-1} x_{t-1} \quad \forall t = 1, 2, ..., T \end{aligned}$$

Nota: Se asume que x_0 es conocido.

El problema se puede reescribir como un problema de T etapas

$$\begin{aligned} &\min c_1 x_1 + \alpha_1(x_1) \\ &s.a.\ A_1 x_1 \geq b_1 - E_0 x_0 \end{aligned}$$

donde

$$\begin{aligned} \alpha_t(x_t) = &\min c_t x_t + \alpha_{t+1}(x_{t+1}) \\ &s.a.\ A_t x_t \geq b_t - E_{t-1} x_{t-1} \quad \forall t = 2, ..., T-1 \\ \text{y} \quad \alpha_T(x_T) = &\min c_T x_T \\ &s.a.\ A_T x_T \geq b_T - E_{T-1} x_{T-1} \end{aligned}$$

Entonces podemos solucionar el problema usando el Algoritmo de Benders con la particularidad que nuestro $n-$ésimo Subproblema es el Problema Maestro de la iteración siguiente; es decir el $n+1-$ésimo Problema Maestro, esto para $n = 1, ..., T-1$.
Para la solución partimos teniendo el primer Problema Maestro y primer Subproblema y aplicamos el algoritmo de Benders. Comenzamos con un valor inicial de $\tilde{x}_1$, para luego pasar al Subproblema el cual depende de las variables $x_2, x_3, ..., x_T$.

El problema Maestro de la primera iteración es

$$\begin{aligned} &\min c_1x_1 + \tilde{\alpha}_1 \\ &s.a.\ A_1x_1 \geq b_1 - E_0x_0 \\ &\quad \lambda^j(b_2 - E_1x_1) - \tilde{\alpha}_1 \leq 0 \end{aligned}$$

Y como Subproblema a:

$$\begin{aligned} \alpha_1(\hat{x}_1) = \min\ & c_2x_2 + c_3x_3 + ... + c_tx_t \\ s.a.\ & A_2x_2 \geq b_2 - E_1\hat{x}_1 \\ & A_tx_t \geq b_t - E_{t-1}x_{t-1} \quad \forall t = 3, ..., T \end{aligned}$$

donde $\hat{x}_1$ es el valor prueba de x_1 encontrado desde el problema maestro de arriba.
Debido a que el subproblema resulta complicado de resolver, lo tomamos como un problema que queremos resolver. Y por su forma podemos volver a Aplicar el algoritmo de Bender para descomponerlo en otro Problema Maestro y Subproblema. Nos podemos dar cuenta que el Subproblema es el Problema Maestro de la iteración siguiente.

$$\begin{aligned} \alpha_1(\hat{x}_1) = \min\ & c_2x_2 + \alpha_2 \\ s.a.\ & A_2x_2 \geq b_2 - E_1\hat{x}_1 \\ & \lambda_1^j(b_3 - E_2x_2) - \tilde{\alpha}_2 \leq 0 \end{aligned}$$

Y como segundo Subproblema a:

$$\begin{aligned} \alpha_2(\hat{x}_2) = \min\ & c_3x_3 + ... + c_tx_t \\ s.a.\ & A_3x_3 \geq b_2 - E_2\hat{x}_2 \\ & A_tx_t \geq b_t - E_{t-1}x_{t-1} \quad \forall t = 4, ..., T \end{aligned}$$

El proceso sigue hasta llegar al Problema Maestro T, que es como sigue.

$$\begin{aligned} \alpha_{T-1}(\hat{x}_{T-1}) = \min\ & c_Tx_T \\ s.a.\ & A_Tx_T \geq b_{T-1} - E_{T-1}\hat{x}_{T-1} \\ & \lambda_{T+1}^j(b_{T+1} - E_Tx_T) - \tilde{\alpha}_{T+1} \leq 0 \quad \forall j = 1, 2, ..., J \end{aligned}$$

Pase hacia adelante

En el pase hacia adelante se resuelven subprobemas de la forma

$$\begin{aligned} \min\ & c_1x_1 + c_2x_2 + ... + c_tx_t + \tilde{\alpha}_{t+1} \\ s.a.\ & \quad b_t - E_{t-1}x_{t-1}^* \leq A_tx_t \\ & \lambda_{t+1}^j(b_{t+1} - E_tx_t) - \tilde{\alpha}_{t+1} \leq 0 \quad \forall j = 1, 2, ..., J \end{aligned}$$

Aquí se forma soluciones prueba que son pasados a las siguientes etapas donde son parámetros fijos para el subproblema. Después de todos los pases una solución factible es encontrada, generando así una cota superior para el problema de minimización.

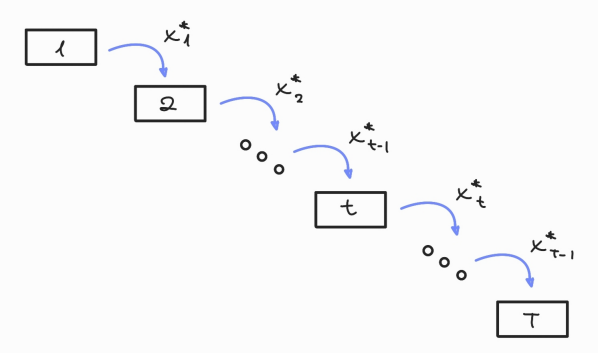


Figura 4.6: Pase hacia adelante

Pase hacia atrás

En el pase hacia atrás comienza desde la última etapa T y acaba en la primera, Aquí las aproximaciones de los subproblemas son mejoradas iterativamente por construcciones de cortes que son pasados desde atrás hacia la etapa previa. De esta manera, si la solución no llega a ser óptima, la solución actual es modificada.

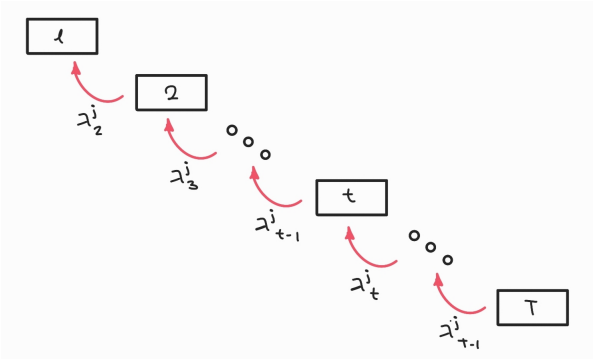


Figura 4.7: Pase hacia atrás

4.2.1. Algoritmo

Paso 1: Definir las funciones de costo $\tilde{\alpha}_{t+1}(x_t)$ con $t = 1, .., T$ que son cotas inferiores de las funciones (desconocidas) $\{\tilde{\alpha}_{t+1}(x_t)\}$. Estas funciones de aproximación son relajaciones de los problemas duales. Comenzamos con lo siguiente:

- J=0
- $\{\alpha_{t+1}(x_t), t = 1, ..., T\} = 0$
- cota superior $= +\infty$
- tolerancia $= \varepsilon$

- variable en la etapa cero $= x_0^* = 0$

Paso 2: Pase hacia adelante. Repetir para $t = 1, ..., T$

Paso 2.1: Resolver el problema de programaciòn aproximada

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} c_t x_t + \tilde{\alpha}_{t+1} \\ s.a. \quad & b_t - E_{t-1} x_{t-1}^* \leq A_t x_t \\ & \lambda_{t+1}^j (b_{t+1} - E_t x_t) - \tilde{\alpha}_{t+1} \leq 0 \quad \forall j = 1, 2, ..., J \end{aligned}$$

la ùltima restricciòn se da excepto para $t = T$ donde $\tilde{\alpha}_{t+1}$ es siempre igual a 0. El primer conjunto de restricciones representa las restricciones de programación de la etapa t; y el segundo conjunto de restricciones, la aproximación (relajación) de la función de costo futuro $\alpha_{t+1}(x_t)$.

Paso 2.2: Almacenar el vector optimal x_t^* y la operación de costo de la etapa t, $C_t x_t^*$.

Paso 3: El valor optimal de la función objetivo del problema del paso 2 para la primera etapa, $C_1 x_1^* + \tilde{\alpha}_2^*$ es una cota inferior $\underline{z}$ de la solución principal, ya que $\tilde{\alpha}_2^*$ es una cota inferior de la función de costo futuro $\tilde{\alpha}_2(x_1)$.
El coste total de la solución, $\sum_{i=1}^{T} C_t x_t^*$, es una cota superior $\overline{z}$, ya que $\{x_t^*, t = 1, ..., T\}$ es factible pero no necesariamente solución optimal del problema original.

Paso 4: Si $\overline{z} - \underline{z} \leq \varepsilon$; en otro caso, ir al **Paso 5**.

Paso 5: Pase hacia atrás. Sea $J = J + 1$. Rpetir para $i = T, T-1, ..., 2$

Paso 5.1: Resolver el problema de programación aproximada

$$\begin{aligned} \min & \sum_{t=1}^{T} c_t x_t + \tilde{\alpha}_{t+1} \\ s.a. & A_t x_t \geq b_t - E_{t-1} x_{t-1}^* \\ & \lambda_{t+1}^j (b_{t+1} - E_t x_t) - \tilde{\alpha}_{t+1} \geq 0 \quad \forall j = 1, 2, ..., J \end{aligned}$$

la última restricción se da excepto para $t = T$ (donde $\tilde{\alpha}_{t+1}$ es siempre igual a 0) donde x_{t-1}^* es obtenida desde la solución del problema del paso 2 del algoritmo.

Paso 5.2: Sea λ_t^J el vector de multiplicadores asociados a las primeras restricciones al conjunto de restricciones. En el problema, λ_t^J mide el cambio del costo de operación del sistema desde la etapa t hasta el final del horizonte; T, causa un cambio marginal en la disponibilidad del almacenamiento al sistema de reservorios al iniciar la etapa t (final de la etapa $t-1$), represemtada por x_{t+1}^*. Este multiplicador es luego usado para formar un segmento lineal adicional para la aproximación de la función de costo futuro de la etapa $t-1, \tilde{\alpha}_t$.

Paso 6: Volver al **Paso 2.**

Observaciones:

- Como se puede observar en el algoritmo y método se consideran cortes de optimalidad, mas no de factibilidad, debido a que el problema se supone toma un valor finito en cada iteración y con conjunto factible acotado.
- El algoritmo acaba en un número finito de pasos debido a que existen puntos y direcciones estremas finitas por iteración.
- Al igual que en el algoritmo de Benders de dos etapas la cota inferior es monótonamente creciente. En cada etapa el problema Maestro restricto, usado para hallar la cota inferior, se vuelve menos relajado (cada vez se le añaden restricciones) y así la cota inferior del problema de T etapas, compuesta por las cotas inferiores acumuladas por etapa, son monótonamente crecientes.

Esta extensión del Algoritmo de Benders se puede extender al caso estocástico y bajo algunas condiciones, vistas en el Capítulo 5, dan paso al algoritmo de SDDP.

Capítulo 5

Programación Estocástica

Los problemas determinísticos, que comúnmente vemos, son formulados con parámetros conocidos. Sin embargo, en la vida real muchos de los problemas tienen parámetros que son desconocidos al momento en el que la decisión es hecha o tomada. La programación estocástica o un programa estocástico es un enfoque para modelar problemas de optimización que involucran incertidumbre. La incertidumbre se usa para indicar que alguna propiedad de una distribución, como su media, es desconocida pero se le puede estimar mediante una distribución probabilística, función de densidad u observaciones previas. Así los parámetros desconocidos son tomados como variables aleatorias con distribución conocida. El problema viene dado como sigue:

$$\min_{x \in X} \{f(x) := \mathbb{E}_{\mathbb{P}} F(x, \xi)\}$$

Y su modelo determinístico asociado es como sigue

$$\begin{aligned} &\min_{x} \sum_{i=1}^{S} p_{\xi_s} \cdot F(x, \xi_s) \\ &s.a.\ x \in \mathcal{X}(\xi_s)\ \forall s = 1, ..., S \end{aligned}$$

donde $p_{\xi_s} = \mathbb{P}(\xi = \xi_s)$

El objetivo de un problema estocástico es minimizar o maximizar la esperanza de una función $F(x, \xi)$ (que depende de la variable aleatoria ξ, con distribución de probabilidad P, y de la decisión x) bajo un conjunto factible X que puede o no depender del parámetro con incertidumbre. Si la esperanza no es tomada, el problema no estaría bien definido ya que nuestra función depende no solo del vector de decisión sino también de la variable aleatoria. Entonces la función objetivo se toma en promedio y usando la Ley de Grandes Numeros (LGN) converge, de forma casi segura, a la esperanza de la función.

5.1. Aplicación al Despacho Térmico Estocástico con escenarios

Luego de presentar el Problema de despacho térmico en el Capítulo 2, consideramos el caso estocástico que toma en cuenta la fluctuación de la demanda y las fallas aleatorias en los equipamientos.
La capacidad del generador se puede representar como variable aleatoria y representa la falla en el equipamiento i. Cuando haya falla $\overline{g}_i = 0$ y cuando no hay, $\overline{g}_i$ toma el valor de un límite operativo. Entonces tal variable sigue una distribución de Bernoulli con cierta probabilidad; es decir,

$$\begin{array}{ll} \overline{g}_i = 0 & \text{con probabilidad } r_i \\ \overline{g}_i = \overline{G}_i & \text{con probabilidad } (1 - r_i) \end{array}$$

donde $\overline{G}_i$ es el límite operativo del generador i.

Consideramos K escenarios de capacidad de generación, obtenidos por la combinación de los escenarios de falla de funcionamiento de cada uno de los generadores. Sea p_k la probabilidad asociada al escenario k $(k = 1, ..., K)$ obtenida por la multiplicación de las probabilidades de falla del funcionamiento de cada generador. Además q_j representa en qué porcentaje se necesita la demanda g_{ijk} (el porcentaje se calcula previamente sabiendo los niveles de demanda que se necesita cubrir por hora). Así obtenemos el siguiente problema.

$$z = \min \sum_{k=1}^{K} p_k \sum_{i=1}^{J} q_j \left(\sum_{i-1}^{I} c_i g_{ijk} + c_\delta \cdot \delta_{jk} \right) \tag{5.1}$$

$$s.a. \sum_{i=1}^{I} g_{ijk} + \delta_{jk} = d_j \quad j = 1, ..., J, \quad k = 1, ..., K \tag{5.2}$$

$$0 \leq g_{ijk} \leq \overline{g}_{ik} \quad i = 1, ..., I \quad j = 1, ..., J, \quad k = 1, ..., K \tag{5.3}$$

$$\delta_{jk} \geq 0$$

donde

Parámetros:

i	indexa los generadores
I	cantidad de generadores
j	indexa los niveles de demanda
J	número de niveles de demanda
k	indexa escenarios de la capacidad
K	cantidad de escenarios de demanda
p_k	probabilidad del escenario k
q_j	duración del nivel de demanda j
c_i	costo unitario de operación del $i-$ésimo generador
d_j	demanda del nivel j
$\overline{g}_{ik}$	capacidad del $i-$ésimo generador escenario k

c_δ penalidad por no cumplir el atendimiento de la demanda

Variables:

g_{ijk} energía producida por el $i-$ésimo generador en el nivel de demanda j, escenario k

δ_{jk} energía no satisfechan del nivel j, escenario k debido a la insuficiencia de generación

El objetivo es minimizar el valor esperado del costo de generación total. La restricción 5.2 representa el atendimiento de la demanda y las restricciones de 5.3, los límites de una generación de cada generador para cada nivel de demanda en cada escenario de capacidades.
Por la gran cantidad de escenarios que se crean (K^I), se opta solucionar el problema por métodos como la simulación de Monte Carlo o descomponiendo el problema en $K \times J$ problemas menores por las independencias de los escenarios.

5.1.1. Ejemplo - Problema de Despacho Térmico

Se considera un sistema compuesto por dos plantas térmicas con las siguientes características

Planta	Capacidad (MW)	Tasa de ruptura (%)	Costo operacional, c_j (MW)
1	500	20	5
2	200	10	10

La penalidad bajo el no atendimiento de la demanda es de $100\$MWh$. Los niveles de demanda son los siguientes

Demanda (MW)	Duración (%) (q_j)
600	50
400	50

Así, al tener 2 escenarios de capacidad y 2 plantas térmicas, los escenarios posibles son $2^2 = 4$.

Escenario	$\overline{g}_{1k}$	$\overline{g}_{2k}$	Probabilidad p_k
1	500	200	$0.8 \times 0.9 = 0.72$
2	500	0	$0.8 \times 0.1 = 0.08$
3	0	200	$0.2 \times 0.9 = 0.18$
4	0	0	$0.2 \times 0.1 = 0.02$

Del problema identificamos que $I = 2, J = 2, K = 4$ Usando el Modelo de Despacho 5.1 el

problema se fórmula como sigue

$$\begin{aligned}
z = \min\ & p_1 \sum_{j=1}^{2} q_j \left(\sum_{i=1}^{2} c_i g_{ij1} + c_\delta \cdot \delta_{j1} \right) + p_2 \sum_{j=1}^{2} q_j \left(\sum_{i=1}^{2} c_i g_{ij2} + c_\delta \cdot \delta_{j2} \right) \\
& + p_3 \sum_{j=1}^{2} q_j \left(\sum_{i=1}^{2} c_i g_{ij3} + c_\delta \cdot \delta_{j3} \right) + p_4 \sum_{j=1}^{2} q_j \left(\sum_{i=1}^{2} c_i g_{ij4} + c_\delta \cdot \delta_{j4} \right) \\
s.a.\ & g_{111} + g_{211} + \delta_{11} = d_1, \quad g_{121} + g_{221} + \delta_{21} = d_2 \\
& g_{112} + g_{212} + \delta_{12} = d_1, \quad g_{122} + g_{222} + \delta_{22} = d_2 \\
& g_{113} + g_{213} + \delta_{13} = d_1, \quad g_{123} + g_{223} + \delta_{23} = d_2 \\
& g_{114} + g_{214} + \delta_{14} = d_1, \quad g_{124} + g_{224} + \delta_{24} = d_2 \\
& 0 \le g_{111} \le \overline{g}_1, \quad 0 \le g_{211} \le \overline{g}_1, \quad 0 \le g_{121} \le \overline{g}_1, \quad 0 \le g_{221} \le \overline{g}_1 \\
& 0 \le g_{112} \le \overline{g}_2, \quad 0 \le g_{212} \le \overline{g}_2, \quad 0 \le g_{122} \le \overline{g}_2, \quad 0 \le g_{222} \le \overline{g}_2 \\
& 0 \le g_{113} \le \overline{g}_3, \quad 0 \le g_{213} \le \overline{g}_3, \quad 0 \le g_{123} \le \overline{g}_3, \quad 0 \le g_{223} \le \overline{g}_3 \\
& 0 \le g_{114} \le \overline{g}_4, \quad 0 \le g_{214} \le \overline{g}_4, \quad 0 \le g_{124} \le \overline{g}_4, \quad 0 \le g_{224} \le \overline{g}_4 \\
& \delta_{11}, \delta_{12}, \delta_{13}, \delta_{14}, \delta_{21}, \delta_{22}, \delta_{23}, \delta_{24} \ge 0
\end{aligned}$$

Se puede ver que el problema es separable, se puede dividir en 4 subproblemas

$$\begin{aligned}
C(k) = \min\ & q_1 \left(c_1 g_{11k} + c_2 g_{21k} + c_\delta \cdot \delta_{1k} \right) + q_2 \left(c_1 g_{12k} + c_2 g_{22k} + c_\delta \cdot \delta_{2k} \right) \\
s.a.\ & g_{11k} + g_{21k} + \delta_{1k} = d_1\ ,\ g_{12k} + g_{22k} + \delta_{2k} = d_2 \\
& 0 \le g_{11k} \le \overline{g}_{1k}, \quad 0 \le g_{21k} \le \overline{g}_{2k}, \\
& 0 \le g_{12k} \le \overline{g}_{1k}, \quad 0 \le g_{22k} \le \overline{g}_{2k} \\
& \delta_{1k}, \delta_{2k} \ge 0
\end{aligned}$$

para $k = 1, ..., 4$. Luego el valor óptimo del problema sería

$$z = p_1 \cdot C(1) + p_2 \cdot C(2) + p_3 \cdot C(3) + p_4 \cdot C(4)$$

Escenario 1:

$$\begin{aligned}
C(1) = \min\ & 0.5 \left(5 g_{111} + 10 g_{211} + 100 \cdot \delta_{11} \right) + 0.5 \left(5 g_{121} + 10 g_{221} + 100 \cdot \delta_{2k} \right) \\
s.a.\ & g_{111} + g_{211} + \delta_{11} = 600\ ,\ g_{121} + g_{221} + \delta_{21} = 400 \\
& 0 \le g_{111} \le 500, \quad 0 \le g_{211} \le 200, \\
& 0 \le g_{121} \le 500, \quad 0 \le g_{221} \le 200 \\
& \delta_{11}, \delta_{21} \ge 0
\end{aligned}$$

La solución óptima se da cuando

$$\begin{aligned}
& g_{111} = 500, \quad g_{121} = 400, \quad \delta_{11} = 0 \\
& g_{211} = 100, \quad g_{221} = 0, \quad \delta_{21} = 0
\end{aligned}$$

Dando un valor óptimo de \$2750. Análogamente podemos desarrollar los subproblemas $C(2)$, $C(3), C(4)$ obteniendo así

$$\begin{aligned}
&g_{112} = 500, \quad g_{122} = 400, \quad \delta_{12} = 100 \\
&g_{212} = 0, \quad g_{222} = 0, \quad \delta_{22} = 0 \\
&g_{113} = 0, \quad g_{123} = 0, \quad \delta_{13} = 400 \\
&g_{213} = 200, \quad g_{223} = 200, \quad \delta_{23} = 200 \\
&g_{114} = 0, \quad g_{124} = 0, \quad \delta_{14} = 600 \\
&g_{214} = 0, \quad g_{224} = 0, \quad \delta_{24} = 400
\end{aligned}$$

Así el valor óptimo es

$$\begin{aligned}
z &= 0.72 \cdot 2750 + 0.08 \cdot 7250 + 0.18 \cdot 32000 + 0.02 \cdot 50000 \\
z &= \$\,93020
\end{aligned}$$

5.2. Aplicación al Despacho Hidrotérmico Estocástico

Si consideramos inicialmente un sistema compuesto por I generadores térmicos y una planta hidroeléctrica, cuya producción de energía esta limitada a un valor máximo E, el límite de agua disponible para la generación puede ser escrito como

$$\sum_{j=1}^{J} q_j h_{jk} \leq E, \quad k = 1, ..., K$$

donde k indexa los escenarios de las capacidades y j indexa los niveles de demanda.
El problema de despacho estaría dado por

$$z = \min \sum_{k=1}^{K} p_k \sum_{j=1}^{J} q_j \left(\sum_{i=1}^{I} c_i g_{ijk} + c_\delta \cdot \delta_{jk} \right) \tag{5.4}$$

$$\begin{aligned}
s.a. \sum_{i=1}^{I} g_{ijk} + h_{jk} + \delta_{jk} &= d_j \quad j = 1, ..., J, \quad k = 1, ..., K \\
0 \leq g_{ijk} &\leq \overline{g}_{ik} \quad i = 1, ..., I \quad j = 1, ..., J, \quad k = 1, ..., K \\
0 \leq h_{jk} &\leq \overline{h}, \quad j = 1, ..., J, \quad k = 1, ..., K \\
\sum_{j=1}^{J} q_j h_{jk} &\leq E, \quad k = 1, ..., K \qquad (5.5) \\
\delta_{jk} &\geq 0
\end{aligned}$$

donde

Parámetros:

i indexa los generadores

I	cantidad de generadores
j	indexa los niveles de demanda
J	número de niveles de demanda
k	indexa escenarios de la capacidad
K	cantidad de escenarios de demanda
p_k	probabilidad del escenario k
q_j	duración del nivel de demanda j
c_i	costo unitario de operación del $i-$ésimo generador
d_j	demanda del nivel j
$\overline{g}_{ik}$	capacidad del $i-$ésimo generador escenario k
c_δ	penalidad por no cumplir el atendimiento de la demanda
E	límite máximo de energia de la hidroeléctrica
$\overline{h}$	capacidad de generación de la hidrotérmica

Variables:

g_{ijk}	energía producida por el $i-$ésimo generador en el nivel de demanda j, escenario k
δ_{jk}	energía no satisfechan del nivel j, escenario k debido a la insuficiencia de generación
h_{jk}	generación de la hidroeléctrica en el nivel de demanda j, escenario k

Los métodos de resolución numérica para problemas estocasticos se basan en técnicas variacionales como la técnica de Monte Carlo donde se representa la incertidumbre a través de escenarios o muestras, la aproximación en media muestral, el Método L-Shaped, entre otros y algunos de ellos serán presentados posteriormente.

Hasta ahora hemos visto problemas con una sola etapa o periodo pero existen problemas con más de un periodo. Más aún un modelo de Despacho hidrotérmico posee una dinmámica donde algunas de sus variables poseen incertidumbre.

5.3. Programación Estocástica de Dos etapas

Aquí el problema se divide en 2 periodos, en el primer periodo tomamos una decisión sobre la información disponible y en el segunda etapa, después de que la realización de los datos con incertidumbre se conoce, una decisión óptima es tomada. El modelo se puede expresar de la siguiente manera

$$\min_{x \in X} \{f(x) := \mathbb{E}_P F(x, \xi)\}$$

donde

$$F(x, \xi) = \min_{y \in Y(x)} \{F_2(y, \xi)\}$$

En el modelo x representa las decisiones de la primera etapa, w las realizaciones de la v.a. ξ e y es un vector de decisiones de la segunda etapa. Por tanto la secuencia que se sigue en problemas de este tipo es la siguiente

$$x \longrightarrow \xi(\omega) \longrightarrow y \tag{5.6}$$

Es decir, primero tomamos una decisión, observamos las realizaciones ω y luego se concluye resolviendo el problema ya sabiendo x y conociendo las incertidumbres.
De un problema de múltiples periodos, en particular dos, se puede desprender un árbol de decision.

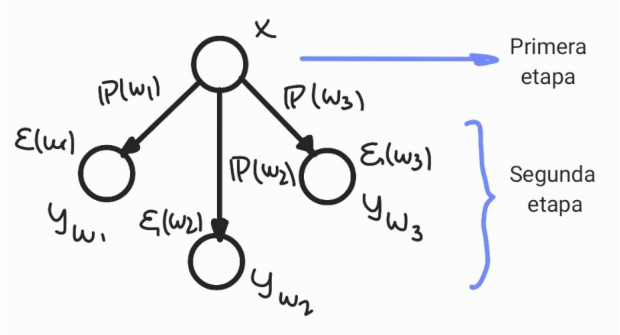


Figura 5.1: Árbol de decisión de un problema de 2 etapas, con 3 observaciones

En la programación lineal, un modelo de un problema estocástico lineal con recurso de dos etapas es el que se muestra a continuación;

$$\begin{aligned} \min \ & c^T x + \mathbb{E}_{\tilde{\xi}} \left[Q(x, \tilde{\xi}) \right] \\ s.a. \ & Ax = b \\ & x \geq 0 \end{aligned}$$

donde

$$Q(x, \xi) = \min \left\{ q^T y(\xi) \ : \ Wy(\xi) \geq \xi - Tx, y(\xi) \geq 0 \right\} \tag{5.7}$$

Luego de tomar la decisión x en la primera etapa, la realización del vector aleatorio $\tilde{\xi}$ es observada para así en la segunda etapa conocer el valor de ξ y posteriormente tomar la decisión $y(\xi)$. Cabe resaltar que $\mathbb{E}_{\tilde{\xi}}$ representa la esperanza con respecto a la variable aleatoria $\tilde{\xi}$.
Si asumimos que el vector de datos aleatorios $(\tilde{\xi})$ tiene un número finito de realizaciones (posibles escenarios) $\xi_k = (q_k, V_k, T_k, h_k)$ con su respectiva probabilidad $p_k, k = 1, ..., K$, es decir $\mathbb{P}(\tilde{\xi} = \xi_k) = p_k$, entonces podemos reescribir el problema como

$$\begin{aligned} \underset{x, y_1, .., y_k}{\text{Min}} \ & c^T x + \sum_{k=1}^{K} p_k \cdot Q(x, \xi_k) \\ s.a. \ & Ax = b \\ & x \geq 0 \end{aligned}$$

donde $Q(x, \xi_k) = \min \left\{ q^T y(\xi_k) \ : \ Wy(\xi_k) \geq \xi_k - Tx, y(\xi_k) \geq 0 \right\}, k = 1, ..., K$

5.3.1. Aproximación a la media Muestral

Consideremos la formulación siguiente de un problema de dos etapas

$$\min_{x \in \mathcal{X}} \left\{ g(x) := c^T x + \mathbb{E}[Q(x, \xi(w))] \right\}$$

donde

$$Q(x,\xi) := \min_{y \in Y} \{q^t y : Wy \geq h - Tx\}$$

es un valor optimal y $\xi = (q, T, W, h)$ denota el vector de parámetros del problema en la segunda etapa. Se asume que para algunas de las componentes (o todas) de ξ son aleatorios, sean $\xi(w)$, y la esperanza es tomada con respecto a la distribución de probabilidad de $\xi(w)$ que es conocida de antemano.
La idea principal de la aproximación por Media Muestral (SAA por sus siglas en inglés) es como sigue. Una muestra $\xi^1, ..., \xi^N$ de N realizaciones del vector aleatorio $\xi(w)$ es generada para luego aproximar la esperanza, $\mathbb{E}[Q(x, \xi(w))]$ por la función por media muestral $N^{-1} \sum_{n=1}^{N} Q(x, \xi^n)$. Obteniendo la aproximación por media muestral

$$\min_{x \in \mathcal{X}} \left\{ \tilde{g}_N(x) := c^t x + N^{-1} \sum_{i=1}^{N} Q(x, \xi^N) \right\}$$

del problema inicial considerado que es resuelto por algoritmos de optimización deterministica. Cabe resaltar que el SAA no es un algoritmo, ya que aún se necesita resolver el problema, pero sirve para considerar un problema de programación estocástica con escenarios.

5.3.2. Método L-Shaped

Consideremos el problema

$$\begin{aligned} \min_{x,y} \ & \mathbb{E}\left[c^t x + q^t \ y \right] \\ s.a. \ & Ax = b, x \geq 0 \\ & Tx + Wy = h, y \geq 0 \end{aligned}$$

La formulación asociada extendida del problema es

$$\begin{aligned} \min_{x,y} \ & c^t x + \sum_{i=1}^{S} \pi^s q^s y^s \\ s.a. & Ax = b, x \geq 0 \\ & T^s x + W^s y^s = h^s, y \geq 0 \ \ \forall s \in [1, S] \end{aligned}$$

Se puede reescribir como

$$\begin{aligned} \min_{x \geq 0} \ & c^t x + \sum_{i=1}^{S} \pi^s Q^s(x) \\ & s.a. Ax = b, x \geq 0 \end{aligned}$$

con

$$\begin{aligned} Q^s(x) := \min_{y \geq 0} \ & q^s y \\ & s.a. W^s y = h^s - T^s x \end{aligned}$$

Asumimos que $\{y^s \geq 0 \mid T^s x + W^s y^s = h^s\} \neq \varnothing \; \forall s \in [1, S]$, obteniendo así la dualidad fuerte.

$$(D_s) \;\; Q^s(x) = \max_{\lambda^s \in \mathbb{R}^m} \lambda^s \cdot (h^s - T^s x)$$
$$s.a. \;\; (W^s)^t \lambda^s \leq q^s$$

Sea P el poliedro $\{\lambda^s \mid (W^s)^t \lambda^s \leq q^s\}$ del conjunto admisible del dual (independiente de x) y $ext(P)$ un conjunto finito de sus puntos extremos. Tenemos que $Q^s(x) = \max_{\lambda^s \in ext(P)} \lambda^s \cdot (h^s - T^s x)$. y así es una función poliedral.

Escribimos la formulación extendida como

$$min \; c^t x + \sum_s \pi^s \theta^s$$
$$s.a. \; Ax = b \; , \; x \geq 0$$
$$\theta \geq Q^s(x) \;\; x \in \mathbb{R}^n, \; \forall s \in [1, S]$$

Como $Q^s(x)$ es una función poliedral de x, $\theta^s \geq Q^s(x)$ puede ser escrito como $\theta \geq \alpha_k^s x + \beta_k^s, \; \forall k \in K^s$ y la descomposición consiste en construir iterativamente cortes con coeficientes α_k^s y β_k^s Para construir los cortes consideremos una solución optimal λ_x^s y un punto de Q^s, x'

$$(D_{x'}) \;\; Q^s(x') = \max_{\lambda^s \in \mathbb{R}^m} \lambda^s \cdot (h^s - T^s x')$$
$$s.a. \;\; (W^s)^t \lambda^s \leq q^s$$

Como λ_x^s es admisible para D_s y $(D_{x'})$ se tiene que

$$Q^s(x') \geq \lambda_x^s \cdot (h^s - T^s x')$$

Para el proceso operativo, sea $x^k \geq 0$ tal que que $Ax^k = b$ y λ_k^s una solución dual, tenemos

$$\alpha_k^s := -(T^s)^t \lambda_k^s \text{ y } \beta_k^s := (\lambda_k^s)^t h^s$$

Los pasos para el método son los siguientes

1. Comenzamos con una colección de K cortes, tal que $Q(x) \geq \alpha_k \cdot x + \beta_k$
2. Resolver el problema maestro, con solución primal optimal x^{K+1}

$$\min_{x \geq 0} \;\; c^t x + \theta$$
$$s.a. \;\; Ax = b$$
$$\theta \geq \alpha_k x = \beta_K \;\; \forall k \in [1, K]$$

3. Resolver S problemas esclavos del dual, para cada solución dual λ_{K+1}^s

$$\max_{x \in \mathbb{R}^m} \;\; \lambda^s \cdot (h^s - T^s x^{K+1})$$
$$s.a. \;\; W^s \cdot \lambda^s \leq q^s$$

4. Construir un nuevo corte con

$$\alpha_k = -\sum_{i=1}^{S} \pi^s (T^s)^t \lambda^s, \quad \beta_{K+1} := \sum_{i=1}^{S} \pi^s h^s \cdot \lambda^s$$

Finitos cortes son añadidos, y el no poder añadir más cortes hace que el algoritmo converja. [10]

5.4. Programación Estocástica Multietapa

Hay situaciones en las que nos enfrentamos a problemas donde las decisiones deben ser tomadas secuencialmente en ciertos periodos de tiempo basados en la información disponible en cada periodo de tiempo. Los datos en el proceso de decisión pueden ser sujetos a incertidumbre el cual es modelado por un espacio de probabilidad a lo largo de vectores aleatorios con valor real $\xi : \omega \to \mathbb{R}$. La incertidumbre puede ser interpretado por etapas definiendo un proceso estocástico $(\xi)_{t=1}^{T}$ con vectores aleatorios $\xi_t \subset \mathbb{R}^{k_1}$. Entonces $\xi = (\xi_1, ..., \xi_T)$.
En la primera etapa se toma la dicisión x_1 para no lidiar con la incertidumbre como en las siguientes etapas. En estas etapas, las decisiones de recurso $x_t(\xi_{[t]})$ puede ser tomada con conocimiento de la historia del proceso de datos de la etapa 1 hasta la t, que se denota por $\xi_{[t]} = (\xi_1, ..., \xi_t)$. Por lo tanto el paradigma es que estas decisiones pueden tomarse después de que se haya desarrollado la incertidumbre correspondiente a la etapa t (las llamadas decisiones de ver y esperar, *wait-and-see*, o proceso de decisión de riesgo, *hazard-decision*) que hace que x_t sea función de $\xi_{[t]}$. Nótese que x_t solo depende de x_t solo depende de las realizaciones hasta la etapa t pero no anticipa eventos o decisiones futuros.

Estos problemas son una generalización de problemas de dos etapas. Un problema de programación estocástica con T etapas puede ser escrita como sigue.

$$\min_{x_1 \in X_1} f_1(x_1) + \mathbb{E}_{\xi_2|\xi_{[1]}} \left[\min_{x_2 \in X_2(x_1,\xi_2)} f_2(x_2,\xi_2) + \mathbb{E}_{\xi_3|\xi_{[2]}} \left[\min_{x_3 \in X_3(x_2,\xi_3)} f_3(x_3,\xi_3) + ... \right. \right. \\ \left. \left. + \mathbb{E}_{\xi_T|\xi_{[T-1]}} \left[\min_{x_T \in X_T(x_{T-1},\xi_T)} f_T(x_T,\xi_T) \right] \right] \right] \tag{5.8}$$

donde $\mathbb{E}_{X|Y}$ representa la esperanza condicional de X dado Y. Además, el proceso de decisiones es análoga a la secuencia 5.6.

$$x_1 \longrightarrow \xi_2 \longrightarrow x_2 \longrightarrow \xi_3 \longrightarrow \quad ... \quad \longrightarrow \xi_T \longrightarrow x_T \tag{5.9}$$

siendo x_i con $i \in 1, ..., T$, las decisiones que se toman en cada etapa y $\xi_i, i \in 1, ...T$ son los vectores estocásticos o variables aleatorias con distribución conocida. Las decisiones que se tomen a la etapa t deben depender de la información $(\xi_1, \xi_2, ..., \xi_t)$ que es la que está disponible a la etapa t pero no de futuras observaciones.

La politica del problema es la secuencia de funciones $(x_t(\xi_{[t]}))_{t=1}^{T}$ y nos da una regla de decisión para todas las etapas y realizaciones de los datos procesados $\xi = (\xi_t, ..., \xi_T)$. Esta política es de *no-anticipavidad*, modelando una secuencia de decisiones condicionales anidadas. El objetivo del problema es determinar una política optimal con respecto a la función objetivo.

El problema lineal de un programa estocástico con T etapas se puede escribir como sigue

$$\begin{aligned} v^* = \min_{x_1} \; & c_1^T x_1 + \mathbb{E}_{\xi_2|\xi_{[1]}} \left[\min_{x_2} c_2(\xi_2)^T x_2(\xi_2) + ... + \mathbb{E}_{\xi_T|\xi_{[1T-1]}} \left[\min_{x_T} c_T(\xi_T)^T x_T(\xi_T) \right] \right] \\ s.a. \; & W_1 x_1 = h_1 \\ & A_1(\xi_2) x_1 + W_2(\xi_2) x_2(\xi_2) = h_2(\xi_2) \\ & A_2(\xi_3) x_2(\xi_2) + W_3(\xi_3) x_3(\xi_3) = h_3(\xi_3) \qquad (P) \\ & \quad \vdots \qquad\qquad \vdots \\ & A_{T-1}(\xi_T) x_{T-1} \xi_{T-1}) + W_T(\xi_T) x_T(\xi_T) = h_T(\xi_T) \\ & x_1, x_2(\xi_2), ..., x_T(\xi) \geq 0 \end{aligned}$$

Algunos o todos los datos en $\xi_t = (c_t, T_1, W_t, h_t)$ puede ser sujeto a la incertidumbre para $t = 2, ..., T$ mientras que para la primera etapa es asumida determinística con la matriz $A_0 = 0$.

El problema usando las ecuaciones de Bellman se reformula como

$$Q_t(x_{t-1}, \xi_{[t-1]}, \xi_t) = \begin{cases} \min\limits_{x_t} & c_t^T(\xi_t) x_t + \mathcal{Q}_{t+1}(x_t, \xi_{[t]}) \\ \text{s.a.} & W_t(\xi_t) x_t = h_t(\xi_t) - A_{t-1}(\xi_t) x_{t-1} \\ & x_t \geq 0 \end{cases} \tag{5.10}$$

donde

$$\mathcal{Q}_{t+1}(x_t, \xi_{[t]}) := \mathbb{E}_{\xi_{t+1}|\xi_{[t]}} \left[Q_{t+1}(x_t, \xi_{[t]}, \xi_{t+1}) \right] \tag{5.11}$$

con $\mathcal{Q}_{T+1}(x_T, \xi_{[T]}) \equiv 0$. A $Q_t(\cdot, \cdot)$ se le llama función valor y a $\mathcal{Q}_t(\cdot, \cdot)$ se le llama función valor esperada, función de costo futuro o función de recurso. Para la primera etapa obtenemos el valor óptimo

$$v^* = \begin{cases} \min\limits_{x_1} & c_1^T x_1 + \mathcal{Q}_1(x_1, \xi_{[1]}) \\ \text{s.a.} & W_1 x_1 = h_1 \\ & x_1 \geq 0 \end{cases} \tag{5.12}$$

Cabe resaltar que en las últimas 3 ecuaciones las variables x_t son determinísticas y no funciones ya que se halla considerando fijo cada realización de ξ_t.
Para asegurar la existencia de soluciones factibles en el problema de optimización, diferentes suposiciones de recurso son considerados el la programación estocástica como el de tener recurso relativamente completo.

Suposición 1: *El problema (P) tiene recurso relativamente completo; es decir, para cualquier*

x_{t-1} factible para la etapa $t-1$, el subproblema de la etapa t es factible para cada realización de ξ_t de forma casi segura.

Cabe mencionar que suponiendo acotamiento y factibilidad de (P) junto con *Suposición 1* entonces se tiene que los problemas 5.10 y 5.12 son acotados y factibles; es decir, todos los $\mathcal{Q}_t(\cdot,\cdot)$ para $t = 2, ..., T$ toman valores finitos.

En caso de vectores aleatorios continuos ξ, la solución de (P) requiere la evaluación de integrales multidimensionales para determinar los valores esperados lo que hace al problema computacionalmente intratable. Así por ahora asumimos al proceso estocástico con un número finito de realizaciones (escenarios), lo que significa que ξ_t es una variable aleatoria discreta y finita para todo $t = 1, ..., T$.

Suposición 2: *Existe un número finito de realizaciones de ξ_t para cada $t = 2, ..., T$*

Además, se requiere que la distribución específica F_ξ de ξ sea conocida. En caso tengamos la Suposición 2, la distribución es simplemente definida por un conjunto finito de salidas de ξ donde se requiere la probabilidad con la que se da cada realización de ξ.

Suposición 3: *Las distribuciones de probabilidad F_ξ del proceso $(\xi)_{t=1}^T$ es conocida.*

Para que el problema no sea dependiente de decisión se asume que las distribuciones de los datos aleatorios no dependen de las decisiones tomadas en las etapas previas.

Suposición 4: *Las distribuciones de probabilidad F_ξ del proceso $(\xi)_{t=1}^T$ no es dependiente de la politica escogida.*

Bajo estas suposiciones, las posibles realizaciones del proceso de datos $(\xi_t)_{t=1}^T$ puede ser representada en forma de un árbol finita de escenarios. La idea es indexar diferentes caminos ξ^s a traves del arbol de escenarios; es decir, los escenarios de la etapa t, por $s \in S$ siendo S el numero de escenarios en esa etapa.

Limitaciones en problemas de Programación Estocástica

- El coste computaional tiende a ser alto, en la práctica, el número de etapas no debe ser alta, ya que en cada etapa necesitamos aproximar la estructura de la información.

- A menudo el problema se basa en la linealidad de los costes y la función dinámica recursiva.

- Principalmente devuelven una estimación del coste óptimo y el control del primer paso.

5.5. Programación Dinámica Estocástica

En este enfoque se introduce la incertidumbre al problema dinámico, apareciendo $\tilde{e}_t$ con $t \in \{1, .., T\}$ como los vectores o variables aleatorias al proceso secuencial y siendo T el número de etapas. La recursión del problema es definido como sigue:

$$\begin{aligned} v_n(s_n) = \min \mathbb{E}_{e_n} \left[f_n(d_n, s_n, e_n) + v_{n+1}(s_{n+1}) \right] \\ s.a.\ s_{n+1} = t_n(d_n, s_n, e_n), \\ d_n \in D_n \end{aligned} \tag{5.13}$$

Observaciones:

- $n \in \{1, ..., T-1\}$
- Para $n = T$ se tiene

$$\begin{aligned} v_T(s_T) = \min \mathbb{E}_{e_T} \left[f_T(d_T, s_T, e_T) \right] \\ s.a.\ s_{T+1} = t_n(d_T, s_T, e_T), \\ d_T \in D_T \end{aligned}$$

  donde s_{T+1} es el estado inicial y es dado.

Observación: Las funciones de recursión de Bellman se hallan para cada variable de estado lo que puede estar más propenso a sufrir la maldición de la dimensionalidad. Más aún si cada escenario de variable aleatoria por etapa es tomada en cuenta.

5.5.1. Aplicación al Despacho Hidrotérmico estocástico

5.5.1.1. Recursión

a) Para cada t (periodo de tiempo) se consideran las variables de estado, M en total indexadas por m. Pueden ser los niveles de almacemaniento como $100\,\%, 90\,\%$ hasta $0\,\%$.

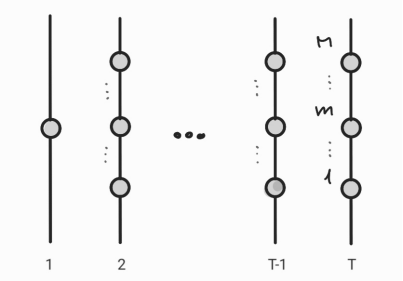


Figura 5.2: Sistema de estados

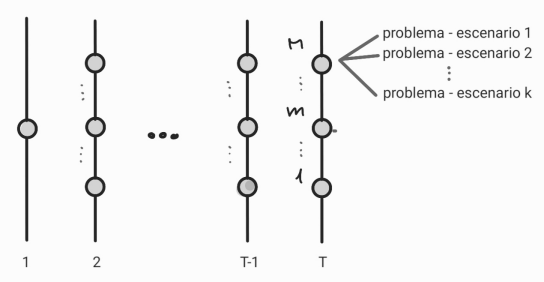


Figura 5.3: Para cada variable de estado en una etapa existen k escenarios

b) Empezando por la etapa T se resuelve el problema de despacho sabiendo el almacenamiento inicial y considerando los escenarios debido a los caudales, K en total.

c) Calculamos la Esperanza de la función, en este caso como el promedio de los K subproblemas y se obtienen puntos debido al discretizamiento del sistema de estados. Resta interpolar los puntos para obtener funciones continuas $\alpha_T(v_T)$ para la etapa $T-1$

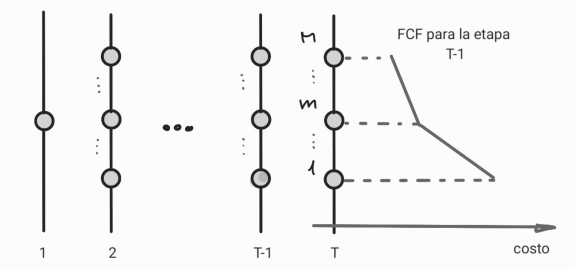


Figura 5.4: La interpolación se da, en este caso, lineal

d) Repetir el procedimiento para cada estado definido en el sistema de estados. Tomar en cuenta que la función ahora esla FCI más la esperanza del costo futuro calculado anteriormente.

e) Repetir el procedimiento d) para las etapas $T-2, T-3, ..., 1$ respectivamente.

La consideración de escenarios para afluencias futuras a embalses introduce estocasticidad y un carácter dinámico al problema.

5.5.1.2. Modelo

Consideremos el despacho hidrotérmico estocástico para una planta hidroeléctrica, con un reservorio, J térmicas y un árbol de caudales con 3 etapas.

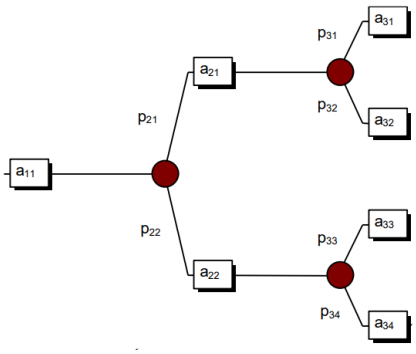


Figura 5.5: Árbol de escenario de caudales

donde

a_{ts} caudal en la etapa t, escenario s

p_{ts} probabilidad condicionada del escenario s de caudal en la etapa t

El problema de despacho se formula como

$$\begin{aligned}
\min\ & c_1(u_{11}) + p_{21}[c_2(u_{21}) + p_{31}c_3(u_{31}) + p_{32}c_3(u_{32})] \\
& \quad + p_{22}[c_2(u_{22}) + p_{33}c_3(u_{33}) + p_{34}c_3(u_{34})] \\
s.a.\ & v_{21} = v_{11} - u_{11} - s_{11} + a_{11} \\
& v_{31} = v_{21} - u_{21} - s_{21} + a_{21} \\
& v_{41} = v_{31} - u_{31} - s_{31} + a_{31} \\
& v_{42} = v_{31} - u_{32} - s_{32} + a_{32} \\
& v_{32} = v_{21} - u_{22} - s_{22} + a_{22} \\
& v_{43} = v_{32} - u_{33} - s_{33} + a_{33} \\
& v_{44} = v_{32} - u_{34} - s_{34} + a_{34} \\
& v_{t+1,s} \leq \overline{v};\ \ u_{t,s} \leq \overline{u} \ \text{ para todas las } t \text{ etapas y todos los escenarios } s \\
& s_{t,s} \geq 0 \ \text{ para todas las } t \text{ etapas y todos los escenarios } s
\end{aligned}$$

Las primeras 7 restricciones corresponden al balance hídrico; mientras el resto, a restricciones de almacenamiento y desfogue.

donde

Parámetros:

$c_t(u_{t,s})$ costo de generación térmica necesaria para complementar la decisión hidroeléctrica

$v_{t+1,s}$ nivel de almacenamiento de embalse al final de la etapa t, escenario s

Variables:

$u_{t,s}$ decisión operativa de la planta hidroeléctrica (volumen turbinado) en la etapa t escenario s

$s_{t,s}$ volumen de vertido en la etapa t, escenario k

Donde la función de complementación térmica $c_t(u_{t,s})$ se representa implicitamente a través de la solución del problema siguiente

$$\begin{aligned} c_t(u_{t,s}) = \min_{g_1,g_2,...,g_J} \quad & \sum_{i=1}^{J} c_t(j) \cdot g_t(j) \\ s.a. \quad & \sum_{j=1}^{J} g_t(j) + \rho u_{t,s} = d_t, \quad t = 1, ..., 3 \\ & 0 \leq g_t(j) \leq \overline{g}(j), \quad j = 1, ..., J, \quad t = 1, ..., 3 \end{aligned}$$

Entonces el problema se puede representar como

$$\begin{aligned} \alpha_t^k(v_t^m) = \min_{u_{t,s}} \quad & c_t(u_{t,s}) + \alpha_{t+1}(v_{t+1}) \\ s.a. \quad & v_{t+1} = v_t^m - u_{t,s} - s_t + a_t^m \\ & v_{t+1} \leq \overline{v} \\ & u_{t,s} \leq \overline{u} \\ & s_{t,s} \geq 0 \end{aligned}$$

Luego calculamos los costos operativos considerando todos los escenarios de caudales

$$\alpha_t(v_t^m) = \sum_{k=1}^{K} p_k \cdot \alpha_t^k(v_t^m)$$

Para finalmente construir la función de costo futuro, $\alpha_t(v_t)$, interpolamos los valores de $\{\alpha_t(v_t^m); m = 1, ..., M\}$ [11].
Podemos extender el problema de despacho hidrotérmico si consideramos I plantas hidroeléctricas con reservorios donde el reservorio i tiene M_i reservorios apilados por arriba. La ecuación de balance hídrico sería

$$v_{i,t+1} = v_{i,t} - u_{i,t} - s_{i,t} + a_{i,t} + \sum_{m \in M_i} (u_{m,t} + s_{m,t})$$

donde

$v_{i,t+1}$ nivel de almacenamiento del embalse i al final de la etapa t
$s_{i,t}$ volumen vertido en la etapa t del reservorio i
$a_{i,t}$ caudal en la etapa t
M_i número de reservorios arriba del reservorio i

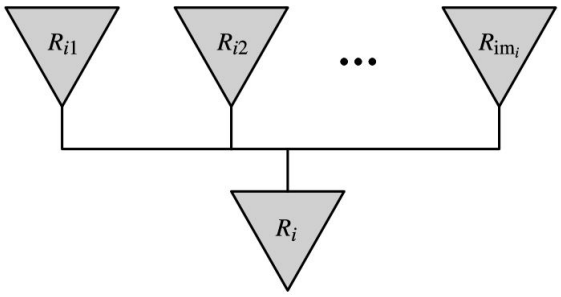


Figura 5.6: Reservorios arriba del reservorio i

5.5.1.3. Ejemplo

Consideremos un sistema compuesto por una planta hidráulica y dos plantas térmicas cuyas características se muestran en la Tabla 5.1 y Tabla 5.2.

Almacenamiento máximo (hm^3)	Almacenamiento mínimo (hm^3)	Turninado máximo (hm^3)	Coeficiente de producción ($MW-$mes$/hm^3$)
120	20	50	0.9

Tabla 5.1: Características de la planta hidrotérmica

La carga del sistema, la demanda, es de $45MW-$mes para todas las etapas y la penalidad por fallo al suplir la demanda es representada por una planta térmica artificial, es de $1000\$/MW-$mes. Además, los escenarios para el caudal son considerados en la Tabla 5.3.

Térmica	Máxima generación (MW)	Costo de operación ($\$/MW-mes$)
T_1	20	10
T_2	25	20

Tabla 5.2: Características de las plantas térmicas

Para este ejemplo el almacenamiento tendra 3 discretizaciones, las cuales son $100\,\%, 50\$, 0\,\%$.

Etapa	Alto (hm^3/mes)	Bajo (hm^3/mes)
1	25	18
2	17	13
3	14	10

Tabla 5.3: Escenarios del caudal para cada etapa

El primer paso consiste en el cálculo del coste medio óptimo para cada nivel de discretización, siendo la media de los costes óptimos calculados para cada escenario de entrada de agua, para cada discretización, en cada etapa.
La ecuación de recursión queda como sigue

$$\begin{aligned} \alpha_t^k(v_t^m) = \min_{u_{t,s},s_t} \quad & c_t(u_{t,s}) + \alpha_{t+1}(v_{t+1}) \\ s.a. \quad & v_{t+1} = v_t^m - u_{t,s} - s_t + a_t^m \\ & 0 \leq v_{t+1} \leq 100 \\ & 0 \leq u_{t,s} \leq 50 \\ & 0 \leq s_t \end{aligned}$$

donde

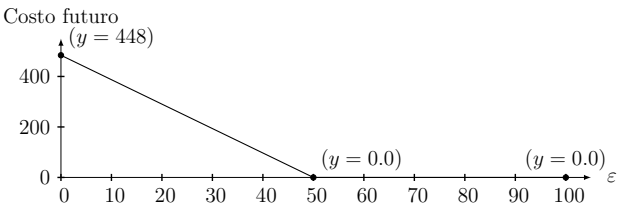
$$\begin{aligned} c_t(u_{t,s}) = \min_{g_1,g_2,\delta} \quad & 10 \cdot g_t(1) \cdot 20 \cdot g_t(2) + 1000 \cdot \delta \\ s.a. \quad & g_t(1) + g_t(2) + 0.9u_{t,s} + \delta = 45, \quad t = 1, ..., 3 \\ & 0 \leq g_t(1) \leq 20, 0 \leq g_t(2) \leq 25, \quad t = 1, ..., 3 \end{aligned}$$

y la Función de costo futuro $\alpha_t(v_t^m) = 0.5 \cdot (\alpha_t^1(v_t^m) + \alpha_t^2(v_t^m))$

Comenzamos la recursión con la etapa 3, cuyos valores optimales y construccion de la FCF están en la Figura 5.7. Pasamos a la etapa 2 donde ya conocemos los valores de $\alpha_3(v_3)$ y obtenemos la solución óptima como se muestra en la figura Figura 5.8. Finalmente los resultados y función de costo futuro en la etapa 1 se muestran en la Figura 5.9.
Podemos concluir que la función de costo futuro aproximada para todo el planeamiento considerado se muestra en la Figura 5.9 (b). Lo que significa que si el nivel del reservorio estaba en un 50 %, el costo esperado para que el sistema opere en las tres etapas y considerando los dos escenarios es de $597.8.
La implementación se ubica en la parte final del trabajo en el capítulo Implementaciones.

Almacenamiento v_3 (%)		0		50		100	
Caudal (hm^3)		14	10	14	10	14	10
Decisión Optimal	v_4	0	0	14	10	64	60
	u_3	14	10	50	50	50	50
	s_3	0	0	0	0	0	0
	$g_3(1)$	20	20	0	0	0	0
	$g_3(2)$	12.4	16	0	0	0	0
	δ	0	0	0	0	0	0
Costo inmediato ($)		448.0	520.0	0.0	0.0	0.0	0.0
Costo optimal promedio ($)		484		0		0	

(a) Resultados

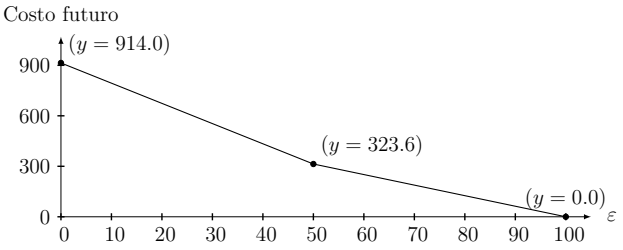


(b) Función de Costo Futuro

Figura 5.7: Resultados etapa 3

Almacenamiento v_2 (%)		0		50		100	
Caudal (hm^3)		17	13	17	13	17	13
Decisión Optimal	v_3	0	0	39.2	35.2	67	63
	u_2	17	13	27.7	27.7	50	50
	s_2	0	0	0	0	0	0
	$g_2(1)$	20	20	20	20	0	0
	$g_2(2)$	9.7	13.3	0	0	0	0
	δ	0	0	0	0	0	0
Costo inmediato ($)		394.0	466.0	200.0	200.0	0.0	0.0
Costo futuro ($)		484.0	484.0	104.3	143.0	0.0	0.0
Costo optimal promedio ($)		914.0		323.6		0.0	

(a) Resultados

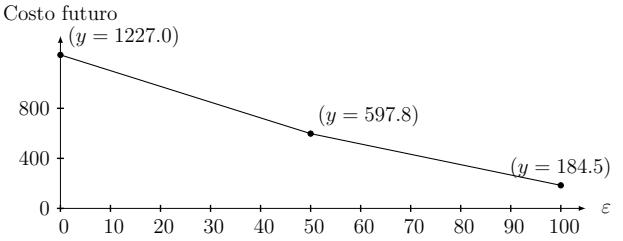


(b) Función de Costo Futuro

Figura 5.8: Resultados etapa 2

Almacenamiento v_2 (%)		0		50		100	
Caudal (hm^3)		25	18	25	18	25	18
Decisión Optimal	v_2	0	0	47.2	40.2	75	68
	u_2	25	18	27.8	27.8	50	50
	s_2	0	0	0	0	0	0
	$g_1(1)$	20	20	20	20	0	0
	$g_1(2)$	2.5	8.8	0	0	0	0
	δ	0	0	0	0	0	0
Costo inmediato		250.0	376.0	200.0	200.0	0.0	0.0
Costo futuro		914.0	914	356.5	439.1	161.8	207.2
Costo optimal promedio ($)		1227.0		597.8		184.5	

(a) Resultados



(b) Función de Costo Futuro

Figura 5.9: Resultados etapa 1

5.5.2. Modelo con caudal como variable de estado

Hasta ahora solo el volumen es variable de estado y además es aquel que cambia con el tiempo, el volumen en la etapa t está relacionado con el volumen en la etapa $t+1$ por la dinámica. Sin embargo, no es la única variable de estado. Cuando la variable caudal en la etapa t se considera más "seco"que el promedio, entonces el caudal en la etapa $t+1$ tiene una tendencia a mantenerse así, y por tanto la consideraremos variable de estado.
La probabilidad ahora dependerá de la correlación que tengan las variables a_t^i y a_{t+1}^j y esta probabilidad será p_{ij} y la correlación entre a_t y a_{t+1} se pueden representar mediante una cadena de Markov como en la Tabla 5.4.

$i \downarrow \setminus \ i+1 \rightarrow$	a_{t+1}^i	$\cdots$	a_{t+1}^l	$\cdots$	a_{t+1}^L
a_t^1	p_{11}	$\cdots$	p_{1l}	$\cdots$	p_{1L}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
a_t^k	p_{k1}	$\cdots$	p_{kl}	$\cdots$	p_{kL}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
a_t^K	p_{K1}	$\cdots$	p_{Kl}	$\cdots$	p_{KL}

Tabla 5.4: Correlacion entre a_t y a_{t+1}

Quedandonos el algoritmo, con las dos variables de estado, como sigue

Inicializar la función de costo futuro de la última etapa $\alpha_{T+1}(v_T, a_T) \to 0$

Repetir para $t = T, T-1, T-2, ..., 1$

Repetir para cada nivel de almacenamiento $v_t = v_t^1, ..., v_t^m, ..., v_t^M$

Repetir para cada escenario de caudales $a_t = a_t^1, ..., a_t^K$

Resolver el problema de despacho hidroeléctrico

$$\begin{aligned} \alpha_t^l(v_t^m, a_t^k) = \min_{u_t} \; & c_t(u_t) + \sum_{l=1}^{L} p_{kl} \cdot \alpha_{t+1}(v_{t+1}^l, a_{t+1}^l) \\ s.a. \; & v_{t+1}^k = v_t^m - u_t - s_t + a_t^l \\ & v_{t+1}^l \leq \overline{v} \\ & u_t \leq \overline{u}, \; 0 \leq s_t \end{aligned}$$

Fin del bucle.

Calculamos el valor esperado del costo

$$\alpha_t(v_t^m, a_{t-1}^k) = \sum_{l=1}^{L} p_{kl}(v_t^m, a_{t-1}^k)$$

Fin del bucle.

Fin del bucle.

Construimos la función de costo futuro $\alpha_t(v_t, a_t)$ interpolando los valores $\alpha_t(v_t^m, a_t^k), m = 1, ..., M, k = 1, ..., K$.

Fin del bucle.

Como vemos la cantidad de operaciones a realizar en el algoritmo crece tanto crece la cantidad de escenarios y valores de la variable de estado.

Se busca solucionar el problema con información del dual del subproblema que resulta al fijar la variable complicada. Para esto escribamos el problema de despacho hidrotérmico multietapa con I hidroeléctricas y J térmicas.

$$\begin{aligned} \alpha_t^k(v_t^m, a_t^k) = \; & \mathbb{E}\left[\min_{u_t, s_t} c_t(u_t) + \sum_{l=1}^{L} p_{kl} \cdot \alpha_{t+1}(v_{t+1}^k, a_{t+1}^l)\right] \\ s.a. \; & v_{t+1} = v_t^m - u_t - s_t + a_t + \sum_{m \in M_i} (u_{m,t} + s_{m,t}) \\ & v_{t+1} \leq \overline{v} \;\; \forall i = 1, ..., I \\ & u_t \leq \overline{u} \;\; \forall i = 1, ..., I \\ & s_t \geq 0 \end{aligned}$$

Y $c_t(u_t)$ representa el costo operativo asociado a la generación hidroeléctrica (generación térmica más hidráulica)

$$\begin{aligned} c_t(u_t) = \min_{g_t} \quad & \sum_{i=1}^{J} c_t(j) \cdot g_t(j) + c_\delta \delta_t \\ s.a. \quad & \sum_{j=1}^{J} g_t(j) + \rho \sum_{i=1}^{I} u_t + \delta_t = d_t, \quad t = 1, ..., 3 \\ & 0 \leq g_t(j) \leq \overline{g}(j), \quad j = 1, ..., J, \quad t = 1, ..., 3 \end{aligned}$$

donde j indexa las plantas térmicas.

Resolvamos ahora el problema haciendo uso de variables duales como se vio en el Capítulo 4 lo que nos lleva a un problema de Programación Dinámica Dual Estocástica.

5.6. Programación Dinámica Dual Estocástica (SDDP)

En esta sección se vio la estructura de un Problema Mutietapa Estocástico. Hasta ahora para considerar algoritmos en caso dinámico era necesario recorrer todas las discretizaciones; más aún, el problema se vuelve computacionalmente intratable para todas las instancias de problemas grandes (considerando cientos o pocos miles de escenarios), ya que el árbol de escenario crece exponencialmente con el número de etapas.
Este problema denominado antes como la *maldición de la dimensionalidad* no se da en el SDDP. La idea principal es que en cada iteración no todos los escenarios $\xi^s, s \in S$ del árbol de escenario son considerados, solo se utiliza una muestra de estos lo que reduce significativamente el número de programas lineales a resolver. El SDDP viene junto con la suposición crucial adicional de que la incertidumbre en las diferentes etapas no depende una de la otra.

Suposición 1: *La incertidumbre de resolver el problema (P) es independiente por etapas; es decir, es independiente de la historia $\xi_{[t-1]}$ del proceso.*

Bajo la Suposición 1 el árbol de escenarios colapsa para recombinar el árbol de escenarios. Además $\mathbb{E}_{\xi_{t+1}|\xi_t}$ se simplifica a $\mathbb{E}_{\xi_{t+1}}$ y $\mathcal{Q}_{t+1}(x_t, \xi_{[t]}) = \mathcal{Q}_{t+1}(x_t)$; es decir, par cada etapa t escencialmente existe una función de valor esperado el cual no depende del proceso de los datos.

5.6.1. Pase o Simulación hacia adelante (Forward Simulation)

El principio del paso hacia adelante es como el visto antes pero considerando un muestreo de escenarios. Los índices de $k \in K$ denotan los escenarios muestreados con $K \subset S$ y $K \ll S$

Los subproblemas a resolver para la iteración i, etapa $t = 2, ..., T$ y muestreo k se denotan

$$\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k) = \begin{cases} \min\limits_{x_t} & c_t^T(\xi_t^k)x_t + \mathfrak{Q}_{t+1}^i(x_t) \\ \text{s.a.} & W_t(\xi_t^k)x_t = h_t(\xi_t^k) - A_{t-1}(\xi_t^k)x_{t-1}^{ik} \\ & x_t \geq 0 \end{cases} \tag{5.14}$$

Como en la recursión vista anteriormente de el pase hacia adelante se tienen valores prueba $(x_t(\xi_t^k))_{t=1}^T$ que corresponden a una política factible del problema (P) que esta implícitamente definida por las aproximaciones de corte actuales $\mathfrak{Q}_t^i(\cdot), t = 2, ..., T$. Sin embargo, en contraste con el algoritmo multietapa de Benders, esta política no es evaluada para todos los escenarios, solo para los muestreados.
Definamos a $\overline{v}^i$

$$\overline{v}^i = \mathbb{E}\left[\sum_{t=1}^T c_t^t(\xi_t)x_t^i(\xi_{[t]})\right] = \sum_{k\in K} p_s \sum_{t=1}^T c_t^t(\xi_t)x_t^i(\xi_{[t]})$$

donde p_s denota la probabilidad del escenario s.
En esta simulación no se puede calcular un límite superior determinístico para v^*, pero la media de la muestra

$$\overline{v}_K^i = \frac{1}{|K|}\sum_{k\in K} \underbrace{\sum_{t=1}^T c_t^T(\xi_t^k)x_t^{ik}}_{=:v^i(\xi^k)} \tag{5.15}$$

viene a ser un estimador insesgado de $\overline{v}^i$; es decir, $\mathbb{E}[\overline{v}_K^i] = \overline{v}^i$ y así $\overline{v}_K^i$ es estadísticamente una cota superior de v^*. Más aún, por la ley de grandes números, $\lim_{k\to\infty} \overline{v}_K^i = \overline{v}^i$ y junto con la varianza muestral

$$(\sigma_{\overline{v},K}^i)^2 := \frac{1}{|K|-1}\sum_{k\in K}(v^i(\xi^k) - \overline{v}_K^i)^2$$

para algún $\alpha \in (0, 1)$, un intervalo de confianza es

$$\left[\overline{v}_K^i - z_{1-\alpha/2}\frac{\sigma_{\overline{v},K}^i}{\sqrt{|K|}}, \overline{v}_K^i + z_{1-\alpha/2}\frac{\sigma_{\overline{v},K}^i}{\sqrt{|K|}}\right]$$

donde $z_{(1-\alpha/2)}$ representa el cuantil de la distribución Normal estandar. El criterio de parada de SDDP sucede cuando la mejor cota inferior $\underline{v}^{i-1}$ es incluida en ese intervalo de confianza.

5.6.2. Pase o Simulación hacia atrás (Backward Simulation)

Aquí se da la generación de cortes al igual que en el capítulo donde se vio el Algoritmo Benders Multietapa; sin embargo, no todos los nodos del árbol de escenarios son recorridos en este pase. Para ser más precisos, para todo $t = T, ..., 2$ y todos los escenarios muestreados en el pase hacia adelante; es decir, para todas las soluciones prueba $x_{t-1}^{ik}, k \in K$ los siguientes subproblemas son resueltos para todas posibles realizaciones de la etapa t (aperturas hacia

atrás) $\xi_{tj}^k \equiv \xi_{tj}, j = 1, ..., q_t$:

$$\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj}) = \begin{cases} \min\limits_{x_t} & c_t^T(\xi_{tj})x_t + \mathfrak{Q}_{t+1}^{i+1}(x_t) \\ \text{s.a.} & W_t(\xi_{tj})x_t = h_t(\xi_{tj}) - A_{t-1}(\xi_{tj})x_{t-1}^{ik} \\ & x_t \geq 0 \end{cases} \tag{5.16}$$

$$= \begin{cases} \min\limits_{x_t, \theta_{t+1}} & c_t^T(\xi_{tj})x_t + \theta_{t+1} \\ \text{s.a.} & W_t(\xi_{tj})x_t = h_t(\xi_{tj}) - A_{t-1}(\xi_{tj})x_{t-1}^{ik} \\ & \theta_{t+1} \geq (\beta_{t+1}^r)^T x_t + \alpha_{t+1}^r, \quad r \in R_{t+10} \\ & x_t \geq 0 \end{cases} \tag{5.17}$$

Otra vez, la excepción es considerar $\mathfrak{Q}_{T+1}(\cdot) \equiv 0$.
Sea π_t^{ij} y $\rho_t^{ijr}, r \in R_{t+1}$ denota los vectores optimales duales de los problemas anteriores si tuviesen.

$$\beta_t := -\sum_{j=1}^{q_t} p_{tj}(\pi_t^{ij})^T A_{t-1}(\xi_{tj})$$

y

$$\alpha_t := \sum_{j=1}^{q_t} p_{tj} \left((\pi_t^{ij})^T h_t(\xi_{tj}) + \sum_{r \in R_{t+1}} \rho_t^{ijr} \alpha_{t+1}^r \right)$$

Así, el corte es dado por

$$\mathcal{Q}_t x_{t-1} \geq \alpha_t + \beta_t^T x_{t-1}$$

para todo x_{t-1}. Este corte no depende de la historia de los datos procesados.

Para la primera etapa, el subproblema siguiente es resuelto

$$\underline{v}^i := \begin{cases} \min\limits_{x_1} & c_1^T x_1 + \mathfrak{Q}_2^{i+1}(x_1, \xi_{[1]}) \\ s.a. & W_1 x_1 = h_1 \\ & x_1 \geq 0 \end{cases}$$

$$= \begin{cases} \min\limits_{x_1} & c_1^T x_1 + \theta_2 \\ s.a. & W_1 x_1 = h_1 \\ & \theta_2 \geq (\beta_2^r(\xi_{[1]})^T x_1 + (\alpha_2^r)^T(\xi_{[1]}), \quad r \in R_2 \\ & x_1 \geq 0 \end{cases}$$

y produce una cota inferior $\underline{v}^i$ para v^*

5.6.3. Convergencia

Las funciones $\mathcal{Q}_t(\cdot)$ al ser funciones lineales por partes pueden representarse exactamente como el máximo de finitos cortes lineales. En principio esto permite la convergencia finita.
Cabe mencionar que las cotas superiores no son decrecientes, ni las cotas inferiores son crecientes. Las propiedades de convergencia dependen del muestreo el cual es aleatorio. Por eso, solo se puede esperar la convergencia en probabilidad. [12]

5.6.4. Aplicación al Despacho Hidrotérmico

El problema de Despacho Hidrotérmico tiene la siguiente forma recursiva

$$\begin{aligned}\alpha(v_{t-1}, a_{t-1}) = \mathbb{E}[\ \min \quad & z_t(e_t) + \alpha_{t+1}(v_t, a_t] \\ s.a. \ & v_t(i) = v_{t-1}(i) + a_t(i) - u_t(i) - s_t(i) + \sum_{m \in M_i} [s_t(m) + u_t(m)] \\ & 0 \le v_t(i) \le \overline{v}_t(i) \\ & e_t(i) = \rho(i) u_t(i)\end{aligned}$$

donde

$$\begin{aligned}z_t(e_t) = \ \min \quad & \sum_{j=1}^{J} c(j) g_t(j) + c_\delta \delta_t \\ s.a. \ & \sum_{j=1}^{J} g_t(j) - \sum_{i=1}^{I} e_t(j) + \delta_t = d_t \\ & 0 \le g_t(j) \le \overline{g}_t(j) \ \ , \ \ j = 1, ..., J\end{aligned}$$

5.6.4.1. Recursión del modelo

Pase hacia atrás (Backward Simulation)

Inicializar el número de segmentos lineales N= número de almacenamientos iniciales M

Inicializar la FCF para la última etapa (φ^n_{T+1} y δ^n_{T+1}) $= 0$ para $n = 1, ..., N$

Repetir para $t = T, T-1, T-2, ..., 1$

Repetir para cada nivel de almacenamiento $v_t = \{v^m_t, m = 1, .., M\}$

Repetir para cada escenario de caudales $a_t = a^1_t, ..., a^k_t, ..., a^K_t$

Resolver el problema de despacho para el volumen v^m_t y caudal a^k_t

$$\begin{aligned}\alpha^k_t(v^m_t) = \ & \min_{u_t} c_t(u_t) + \alpha_{t+1} \\ s.a. \ & v_{t+1} = v^m_t - u_t - s_t + a^k_t \qquad \pi^k_{ht} \\ & v_{t+1} \le \overline{v} \\ & u_t \le \overline{u} \\ & \alpha_{t+1} \ge \varphi^m_{t+1} \cdot v_{t+1} + \delta^m_{t+1} \quad \text{para } n = 1, ..., N \\ & s_t \ge 0\end{aligned}$$

Fin del Bucle.

Calcule el coeficiente y el término constante para el segmento lineal $m-$ésimo de la FCF de la etapa anterior

$$\varphi^m_t = \sum_{k=1}^{K} p_k \cdot \pi^k_{ht} \text{ y } \delta^m_t = \sum_{k=1}^{K} p_k \cdot a^k_t(v^m_t) - \varphi^m_t \cdot v^m_t$$

Fin del Bucle

Fin del Bucle

Calculo del límite inferior
En la Programación Dinámica Dual Estocástica los segmentos lineales se usan para extrapolar los valores de la FCF. Si se usa un conjunto menor de almacenamientos iniciales, se generan menos segmentos lineales. Aún así es el máximo sobre todos los segmentos un límite inferior a la función real. Así el límite inferior está dado por

$$\underline{z} = \alpha_1(v_1)$$

Pase hacia adelante (Forward Simulation)
En esta recursión hacemos uso de las FCF halladas en la recursión Backward.
El esquema tiene los siguientes pasos

Definir un conjunto de escenarios de caudales $a_t = \{a_t^1, ..., a_t^m, ..., a_t^M\}$ para $t = 1, ..., T$.

Repetir para cada escenario $a_t = a_t^1, ..., a_t^m, ..., a_t^M$.

Inicializar el almacenamiento para la etapa 1 como las $v_t^m = v_1$

Repetir para $t = 1, ..., T$

Resolver el despacho para el almacenamiento v_t^m y caudal a_t^m

$$\begin{aligned} \alpha_t^k(v_t^m) = & \min_{u_t, s_t} c_t(u_t) + \alpha_{t+1} \\ & s.a.\ v_{t+1}^m = v_t^m - u_t^m - s_t + a_t^m \\ & v_{t+1}^m \leq \overline{v} \\ & u_t^m \leq \overline{u} \\ & s_t \geq 0 \\ & \alpha_{t+1} \geq \varphi_{t+1}^n \cdot v_{t+1}^m + \delta_{t+1}^n \quad \text{para } n = 1, ..., N \end{aligned}$$

Fin del Bucle

Calcular el costo operativo total z^m para el escenario m como la suma de todos los costos inmediatos a lo largo del periodo de estudio.

$$z^m = \sum_{t=1}^{T} c_t(u_t^m)$$

Fin del Bucle

Intervalo de confianza
El valor esperado del costo operativo se estima por el promedio de los costos considerando todos los escenarios de caudales.

$$\hat{z} = \frac{1}{M} \sum_{t=1}^{T} z^m$$

Cuando se usa una simulacion Monte-Carlo, se puede calcular la incertidumbre alrededor del valor "verdadero"de $\overline{z}$. El intervalo de confianza a 95 % es

$$\overline{z} \in [\hat{z} - 1.96 \cdot \hat{\sigma}; \hat{z} + 1.96 \cdot \hat{\sigma}]$$

donde $\hat{\sigma}$ es la desviación estandar del estimador dado por

$$\hat{\sigma} = \frac{1}{M-1}\sqrt{\sum_{m=1}^{M}(z^m - \overline{z})^2}$$

Verificación de la optimalidad
Cuando el límite inferior $\underline{z}$ se encuentre dentro del intervalo de confianza el algoritmo se detiene; en otro caso, se pasa a la siguiente iteración devolviendonos otra aproximación.

5.6.5. Ejemplo

Consideremos un despacho hidrotérmico con 3 plantas térmicas y 1 planta hidráulica con 3 etapas teniendo en cuenta que el almacenamiento inicial es $v_1 = 25$, y coincide con el almacenamiento máximo de el embalse, los caudales para las 3 etapas son 0, 50 y 100 respectivamente. Los costos de generación de las plantas térmicas son de 50, 100 y 150. La demanda a suplir es de 150 para cada etapa. Se busca hallar la política del planeamiento total y haremos uso del paquete SDDP en el lenguaje Julia, que nos permite solucionar el despacho hidrotérmico usando el algoritmo SDDP (código adjuntado en el Capitulo Implementaciones).

Considerando los datos del problema de Programación Estocástica Dinámica con un volumen inicial de 25, nuestro problema queda

```
# Building the subproblem for each node
function subproblem_builder(subproblem::Model, node::Int)
    # State variables: volume
    @variable(subproblem, 0 <= volume <= 100, SDDP.State, initial_value = 25)
    # Decition variables:
    @variables(subproblem, begin
        thermal_generation1 >= 0
        thermal_generation2 >= 0
        hydro_generation >= 0
        hydro_spill >= 0
    end)
    # Inflow as a random variable
    @variable(subproblem, inflow)
    P = [1 / 3, 1 / 3, 1 / 3] # probability wich case occur
    Inflows = [[15.0,20.0,17.0],[25.0,18.0,19],
               [17.0,13.0,15.0],[14.0,12.0,16.0]]
    # omega is a vector of realizations
    SDDP.parameterize(subproblem, Inflows[node], P) do ω
        return JuMP.fix(inflow, ω)
    end
```

Figura 5.10: Construcción 1

```
    # Transition function and constraints
    @constraints(
        subproblem,
        begin
            # hydric balance
            volume.out == volume.in - hydro_generation - hydro_spill + inflow
            # nodal balance
            demand_constraint, 0.9 * hydro_generation + thermal_generation1 + thermal_generation2 == 45
            thermal_generation1 <= 20
            thermal_generation2 <= 25
            hydro_generation <= 50
        end
    )
    # Stage-objective
    @stageobjective(subproblem, 10 * thermal_generation1 + 20 * thermal_generation2)
    return subproblem
end | subproblem_builder (generic function with 1 method)

# Construct the model
model = SDDP.LinearPolicyGraph(
    subproblem_builder,
    stages = 4,
    sense = :Min,
    lower_bound = 0.0, # we cannot incur in negative cost
    optimizer = HiGHS.Optimizer, # solver
) | A policy graph with 4 nodes.
```

Figura 5.11: Construcción 2

```
# Training our policy
SDDP.train(model; iteration_limit = 20) | ✓

# Get a decision rule
rule = SDDP.DecisionRule(model; node = 1) | A decision rule for node 1
# rule
# Evaluating a decision rule
solution = SDDP.evaluate(
    rule;
    incoming_state = Dict(:volume => 25.0),
    noise = 50.0,
    controls_to_record = [:hydro_generation, :thermal_generation1, :thermal_generation2],
) | (stage_objective = 200.0, outgoing_state = Dict(:volume => 47.22222222222222), controls = Dict(:the…
```

Figura 5.12: Entrenamiento

	$c(u_1)$	$g_1(1)$	$g_1(2)$	u_1
Valores	200	20	0	27.8

Obtenemos los resultados de la simulación y los valores óptimos como se muestran en la Tabla.

Etapa	$vol.out$	$g_t(1)$	$g_t(2)$
1	25.3	20	11.8
2	22.5	20	6.3
3	11.7	20	3.6
4	0	20	0

Los resultados se obtienen con un intervalo de confianza de $(1124.74-16.617954730734084, 1124.74+16.617954730734084)$ y una cota inferior de 1144.0.

```
# Get the outgoing volume
outgoing_volume = map(simulations[1]) do node
    return node[:volume].out
end | 4-element Vector{Float64}:

thermal_generation = map(simulations[1]) do node
    return node[:thermal_generation1], node[:thermal_generation2]
end | 4-element Vector{Tuple{Float64, Float64}}:
```

Figura 5.13: Resultados de la simulación

Capítulo 6

Conclusiones

1. Existe otra manera equivalente al algoritmo Simplex dual para hallar el costo mínimo de generación en plantas térmicas como lo es el Método de Baleriaux.

2. Debido a que la programación Dinámica tiende a caer en la maldición de la Dimensionalidad, se opta por otros métodos de solución como lo es el Algoritmo de Benders. Benders soluciona el problema y lo hace de manera exacta sabiendo todos los puntos y direcciones extremas; sin embargo, en la práctica es complicado conocerlos todos. Por eso resuelve un problema Maestro restricto (relajado del problema Maestro) y aproxima a una función objetivo convexa por abajo devolviendonos como resultado una solución muy próxima a la verdadera o incluso la misma.

3. La solución de problemas de Programación estocástica de dos etapas por escenarios también tiende a caer gran dimensionalidad ya que se recorren todos los escenarios; sin embargo, con el Método L-Shaped subsanamos el coste computacional de los escenarios y aproximamos a la función de la segunda etapa mediante cortes, como en Benders. Así encontramos distintos métodos para resolver problemas de Programación Estocástica.

4. La extensión del algoritmo de Benders al caso multietapa junto con variables estocásticas es un problema previo al Algoritmo SDDP. El algoritmo SDDP toma en cuenta independencia de la incertidumbre por etapa, aprovecha la aproximación a la media muestral y toma muestras por etapa en lugar de considerar todos los escenarios lo cual reduce la dimensionalidad ya que el algoritmo no recorrerá todos los nodos del árbol que se desprende. Además se aproxima, con cierta confianza, a la solución para un planeamiento de T etapas mediante los cortes hallados por información de un subproblema Dual.

Implementaciones

Programación Dinámica

La implementación se dio en el lenguaje GAMS.

```
$offSymXRef
$OnMultiR
$offSymList

option limrow = 0;
option limcol = 0;
option solprint=on;
option sysout=off;

$TITLE DP

scalar a;
scalar v;
set puntos /1*3/;
parameter m(puntos) /1 EPS,2 EPS,3 EPS/ ;
parameter b(puntos) /1 EPS,2 EPS,3 EPS/;
parameter caudal

Variables
u,s,g1,g2,delta,F;

POSITIVE VARIABLES
g1,g2,delta,u,s ;

Equations
funobj, eq1,eq2,eq3,eq4,eq5,eq6;

funobj.. F=e= 10*g1+20*g2+100*delta + max(m("1")*(v-u-s+a)+b("1"), m("2")*(v
   -u-s+a)+b("2"));

eq1.. 0 =l= v-u-s+a ;
eq2.. v-u-s+a =l= 100 ;
eq3.. u =l= 50 ;
eq4.. g1+g2+0.9*u+delta =e= 45 ;
eq5.. g1 =l= 20 ;
eq6.. g2 =l= 25 ;
```

```
Model problema1 /funobj,eq1,eq2,eq3,eq4,eq5,eq6/ ;

parameter almacen(puntos) /1 EPS,2 50,3 100/;
parameter alpha(puntos) /1 EPS,2 EPS,3 EPS/;

set esc /1*1/;
set t /1*3/;

Table caudales(t,esc)
    1
1   14
2   17
3   18;

alias(puntos,puntosp);
loop(t,
    loop(puntos,
        v = almacen(puntos);
        a = caudales(t,'1');
        solve problema1 using DNLP minimizing F;
        alpha(puntos) = F.l + EPS;
    );
    display alpha;

    loop(puntos,
        m(puntos) = (sum(puntosp$[ord(puntosp)=ord(puntos)+1],alpha(puntosp)
   )-alpha(puntos))/50;
        b(puntos) = alpha(puntos)-m(puntos)*almacen(puntos);
    );
);

Execute_unload 'Resultados';
```

Algoritmo de Benders

La implementación se dio en el lenguaje Julia y considera pasar todas las direcciones extremas del subproblema dual, si hubiesen, al problema maestro restricto. Además la inicialización se da dándole una cota inferior a *alpha* denotada por M.

```
# Packages
import Pkg
Pkg.add("GLPK")
Pkg.add("JuMP")
using GLPK
using JuMP
Pkg.add("Polyhedra")
Pkg.add("CDDLib")
```

```
using Polyhedra, CDDLib
import Printf

# ====================== MODEL ======================
#    min  cx + dy
#    s.a. Ax + By >= e
#                Cx <= f
#               x,y >= 0

# Datos
#     - M cota inferior de alpha
#     - epsilon>0 y peque o

function MaestroRestricto(M,c, C,f)
    MR = Model(GLPK.Optimizer)
    @variable(MR, x[1:nvarx] >= 0)
    @variable(MR, alpha >= M)
    @objective(MR, Min, c' * x + alpha)
    @constraint(MR, C * x .<= f)
    return MR
end

function SubDual(x)
    SPD = Model(GLPK.Optimizer)
    lb = length(e)
    @variable(SPD, lambda[1:lb] >= 0)
    c2 = e - A * x
    @objective(SPD, Max, c2' * lambda)
    @constraint(SPD, cons, B' * lambda .<= d )
    return SPD
end

function print_iteration(k, args...)
    f(x) = Printf.@sprintf("%12.4e", x)
    println(lpad(k, 9), " ", join(f.(args), " "))
    return
end

function Benders(epsi, maxiter, M, C,f)
    println("Iteration  Lower Bound  Upper Bound          Gap")
    MR = MaestroRestricto(M,c, C,f)
    x = MR[:x]
    alpha = MR[:alpha]
    xgen = Matrix(undef, 0, nvarx)
    for k in 1:maxiter
        # Solve restrict master
        optimize!(MR)
        lower_bound = objective_value(MR)
        xk = value.(x)
        xgen = vcat(xgen,reshape(xk,(1,nvarx)))
```

```
    # Solve subproblem
    SubD = SubDual(xk)
    lambda = SubD[:lambda]
    optimize!(SubD)
    statusSubp = string(MOI.get(SubD, MOI.TerminationStatus()))

    if statusSubp == "OPTIMAL"
        obj = objective_value(SubD)
           = value.(lambda)
        upper_bound = c' * xk + obj
        gap = (upper_bound - lower_bound)
        print_iteration(k, lower_bound, upper_bound, gap)
        if gap < epsi
            println("Terminating with the optimal solution with gap ",
gap)
            if gap == 0
                println("The objective value is ", upper_bound)
            else
                println("The objective value is near to our numerical
results, is more than ", lower_bound, " and less than", upper_bound)
            end
            break
        end
        Alamb = A' *
        cut = @constraint(MR, alpha >=  e'*   - Alamb' * x)
        @info "Adding the optimality cut $(cut)"
    elseif statusSubp == "INFEASIBLE"
        return print("El problema principal es infactible")
        break
    else # unbounded case
        poly = polyhedron(SubD, CDDLib.Library(:exact))
        hrep(poly)
        B2 = rays(vrep(poly))

        # Extreme directions
        extdirec = Matrix(undef, 0, 9)
        for j in B2
            m = []
            for i in 1:9
                push!(m, vec(j)[i])
            end
            m = reshape(m,(1,54))
            extdirec = vcat(extdirec, m) # almacena las direcciones
extremas en una matriz
        end
        # already have extreme rays
        nrays = size(extdirec)[1] # number of rays
        MR = MaestroRestricto(M,c, C,f)
        x = MR[:x]
```

```
            alpha = MR[:alpha]
            for g in 1:nrays
                ray = extdirec[g,:]
                Aray = A'*vec(ray)
                cut2 = @constraint(MR, 0 >= e'*ray - Aray' * x)
                @info "Adding the feasibility cut $(cut2)"
            end
        end
    end

    print("x values throught iterations: \n",xgen, "\n")
    print("Optimal value x= ", vec(xgen[end,:]))
end

Benders(epsi, maxiter, M, C,f)
```

Programación Dinámica Estocástica

La implementación se dio en el lenguaje GAMS.

```
$OnMultiR

option limrow = 0;
option limcol = 0;
option solprint=on;
option sysout=off;

$TITLE SDP

scalar a;
scalar v;
set puntos /1*3/;
parameter m(puntos) /1 EPS,2 EPS,3 EPS/ ;
parameter b(puntos) /1 EPS,2 EPS,3 EPS/;
parameter caudal

Variables
u,s,g1,g2,delta,F;

POSITIVE VARIABLES
g1,g2,delta,u,s ;

Equations
funobj, eq1,eq2,eq3,eq4,eq5,eq6;

funobj.. F=e= 10*g1+20*g2+100*delta + max(m("1")*(v-u-s+a)+b("1"), m("2")*(v
   -u-s+a)+b("2"));

eq1.. 0 =l= v-u-s+a ;
```

```
eq2.. v-u-s+a =l= 100 ;
eq3.. u =l= 50 ;
eq4.. g1+g2+0.9*u+delta =e= 45 ;
eq5.. g1 =l= 20 ;
eq6.. g2 =l= 25 ;

Model problema1 /funobj,eq1,eq2,eq3,eq4,eq5,eq6/ ;
parameter almacen(puntos) /1 EPS,2 50,3 100/;
parameter alpha(puntos) /1 EPS,2 EPS,3 EPS/;

set esc /1*2/;
parameter caudal(esc) /1 14,2 10/;
parameter alphaesc(esc) /1 EPS,2 EPS/;
set t /1*3/;

Table caudales(t,esc)
    1   2
1   14  10
2   17  13
3   25  18;

* ------ Calculo de la funcion alpha ------
alias(puntos,puntosp);
loop(t,
    loop(puntos,
        v = almacen(puntos);
        loop(esc,
            a = caudales(t,esc);
            solve problema1 using DNLP minimizing F;
            alphaesc(esc) = F.l + EPS;
        alpha(puntos) = (alphaesc('1')+alphaesc('2'))/2;)
    );
    display alpha;

    loop(puntos,
        m(puntos) = (sum(puntosp$[ord(puntosp)=ord(puntos)+1],alpha(puntosp)
   )-alpha(puntos))/50;
        b(puntos) = alpha(puntos)-m(puntos)*almacen(puntos);
    );
);

Execute_unload 'Resultados';
```

SDDP

Para la implementación hacemos uso del paquete SDDP en el lenguaje Julia que permite escribir un despacho hidrotérmico bajo ciertos parámetros. La documentación del paquete se encuentra en: `https://github.com/odow/SDDP.jl`

```
# Packages
Pkg.add("SDDP")
Pkg.add("HiGHS") # Solver
using SDDP
using HiGHS

# Number of stages: 3
graph = SDDP.LinearGraph(4)

# Building the subproblem for each node
function subproblem_builder(subproblem::Model, node::Int)
    # State variables: volume
    @variable(subproblem, 0 <= volume <= 100, SDDP.State, initial_value =
   25)
    # Decition variables:
    @variables(subproblem, begin
        thermal_generation1 >= 0
        thermal_generation2 >= 0
        hydro_generation >= 0
        hydro_spill >= 0
    end)
    # Inflow as a random variable
    @variable(subproblem, inflow)
    P = [1 / 3, 1 / 3, 1 / 3] # probability wich case occur
    Inflows = [[15.0,20.0,17.0],[25.0,18.0,19],
                [17.0,13.0,15.0],[14.0,12.0,16.0]]
    # omega is a vector of realizations
    SDDP.parameterize(subproblem, Inflows[node], P) do
        return JuMP.fix(inflow,   )
    end
    # Transition function and constraints
    @constraints(
        subproblem,
        begin
            # hydric balance
            volume.out == volume.in - hydro_generation - hydro_spill +
   inflow
            # nodal balance
            demand_constraint, 0.9 * hydro_generation + thermal_generation1
   + thermal_generation2 == 45
            thermal_generation1 <= 20
            thermal_generation2 <= 25
            hydro_generation <= 50
        end
    )
    # Stage-objective
    @stageobjective(subproblem, 10 * thermal_generation1 + 20 *
   thermal_generation2)
    return subproblem
end
```

```
# Construct the model
model = SDDP.LinearPolicyGraph(
    subproblem_builder,
    stages = 4,
    sense = :Min,
    lower_bound = 0.0, # we cannot incur in negative cost
    optimizer = HiGHS.Optimizer, # solver
)

# Training our policy
SDDP.train(model; iteration_limit = 10)

# Get a decision rule
rule = SDDP.DecisionRule(model; node = 1)
# rule
# Evaluating a decision rule
solution = SDDP.evaluate(
    rule;
    incoming_state = Dict(:volume => 25.0),
    noise = 50.0,
    controls_to_record = [:hydro_generation, :thermal_generation1, :
   thermal_generation2],
)

# Simulating the polity
simulations = SDDP.simulate(
    # The trained model to simulate.
    model,
    # The number of replications.
    100,
    # A list of names to record the values of.
    [:volume, :thermal_generation1, :thermal_generation2, :hydro_generation,
    :hydro_spill],
)

replication = 1
stage = 4
simulations[replication][stage]

# Get the outgoing volume
outgoing_volume = map(simulations[1]) do node
    return node[:volume].out
end

thermal_generation = map(simulations[1]) do node
    return node[:thermal_generation1], node[:thermal_generation2]
end

# Confidence interval
```

```
objectives = map(simulations) do simulation
    return sum(stage[:stage_objective] for stage in simulation)
end

mean, ci = SDDP.confidence_interval(objectives)
println("Confidence interval: ", mean, "    ", ci)
println("Lower bound: ", SDDP.calculate_bound(model))
```

Bibliografía

[1] Mario VF Pereira. Optimal stochastic operations scheduling of large hydroelectric systems. International Journal of Electrical Power & Energy Systems, 11(3):161–169, 1989.

[2] Alexander Shapiro. Analysis of stochastic dual dynamic programming method. European Journal of Operational Research, 209(1):63–72, 2011.

[3] R Tyrrell Rockafellar and Roger J-B Wets. Variational analysis, volume 317. Springer Science & Business Media, 2009.

[4] Mario V. F. Pereira Panos M. Pardalos, Steffen Rebennack and Niko A. Iliadis. Handbook of Power Systems I. Springer Berlin, Heidelberg, 1 edition, 2010.

[5] Dimitri Bertsekas. Dynamic programming and optimal control: Volume I, volume 1. Athena scientific, 2012.

[6] Richard Bellman, Robert E Kalaba, et al. Dynamic programming and modern control theory, volume 81. Citeseer, 1965.

[7] Alberto Vargas and Wilfredo Sifuentes. Despacho económico hidrotérmico multiembalse multinodal de corto plazo. estado del arte de los mÉtodos de optimización.

[8] PAN Ping-Qi. Linear programming computation. Springer, 2014.

[9] Nikki Newham. Power system investment planning using stochastic dual dynamic programming. April 2018.

[10] Vincent LECLERE. The l-shaped method. Cermics, Ecole des Ponts ParisTech ´ France, December 2019. `http://cermics.enpc.fr/~delara/TEACHING/slides_l-shaped.pdf`.

[11] PSR. Sddp manual de metodología. Mayo 2018. Version 15.0.

[12] Steffen Rebemmack Christian Fullner. Stochastic Dual Dynamic Programming and its variants. Preprint, Karlsruhe Institute of Technology, 2014.