Cotas superiores para problemas de otimização estocástica

Projeto ONS – FGV EMAp

Bernardo Freitas Paulo da Costa,
Cleyton Vinicius Prado dos Santos,
Luís Henrique Domingues Bueno,
Maria Soledad Aronna,
Vincent Gérard Yannick Guigues
FGV EMAp – Escola de Matemática Aplicada
Praia de Botafogo, 190, 5º andar
22250-900 – Rio de Janeiro – RJ – Brasil

Lucas de Souza Khenayfis,
Débora Dias Jardim Penna,
Alessandra Mattos Ramos de Oliveira,
Rogerio José Menezes Alves,
Mariana Simões Noel da Silva
ONS – Operador Nacional do Sistema Elétrico
Rua Júlio do Carmo, 251 - Cidade Nova
20211-160 – Rio de Janeiro – RJ – Brasil

Dezembro 2024

Conteúdo

Sumário

Este projeto se propõe a considerar o cálculo de cotas superiores para problemas de otimização estocástica multi-estágio, *com aversão ao risco*. Tais cotas superiores têm duas finalidades principais: em primeiro lugar, avaliar a qualidade das soluções obtidas por algoritmos de programação dinâmica estocástica; em segundo lugar, fornecer um critério de parada efetivo para estes algoritmos, que permita interromper a busca por soluções quando estas estiverem suficientemente boas. Vale lembrar que, em casos concretos, a própria noção de *solução* para um problema de otimização de grande porte é alterada, pois esta requereria especificar todas as variáveis de decisão ao longo de todos os cenários possíveis, o que é inviável em problemas com um grande número de estágios. Assim, ao invés de construir uma solução viável, os algoritmos de programação dinâmica estocástica constroem uma aproximação para as funções de custo futuro, que por sua vez podem ser usadas para determinar uma solução viável de forma *implícita*, também conhecida como *política*.

No caso de otimização neutra ao risco, a obtenção de cotas superiores a partir de uma política é relativamente simples: como a função objetivo é uma *média* dos custos em todos os cenários, podemos escolher (uniformemente) diversos cenários possíveis, calcular os custos de implementar a política em cada um deles, e obter como estimativa de Monte Carlo a média dos custos no subconjunto de cenários explorado. Com um número suficientemente grande de cenários, esta estimativa será suficientemente próxima da média real, e poderemos compará-la com a cota inferior obtida pelo algoritmo de programação dinâmica dual estocástica (PDDE). Este procedimento permite estimar, *estatisticamente*, o custo médio de qualquer política desejada, e com isto avaliar a qualidade da mesma.

Já no caso com aversão ao risco, a dedução de estimativas "de Monte Carlo" para a função objetivo é mais complexa, pois a função objetivo é uma esperança *ponderada* dos custos em todos os cenários, de acordo com a aversão ao risco do tomador de decisão. Ista dá mais peso a alguns cenários do que outros no cálculo do risco (que substitui a média), e com isto uma estimativa de Monte Carlo direta não é mais válida.

Destacamos, neste projeto, duas frentes de pesquisa para o cálculo de cotas superiores em problemas multi-estágio com aversão ao risco. Em primeiro lugar, as abordagens para a obtenção de cotas superiores **determinísticas** para problemas de otimização estocástica multi-estágio. A principal vantagem delas é que o método de construção destas cotas superiores passa por aproximações, agora também superiores, das funções de custo futuro de cada estágio e que, por isto, podem ser adaptadas para o caso com aversão ao risco. Dentro desta linha, destacamos os métodos *primais*, que se valem diretamente da recorrência de programação dinâmica entre as funções de custo futuro, e os métodos *duais*, que deduzem uma recorrência entre os duais do problema original, para em seguida construir cotas superiores.

Em segundo lugar, as abordagens para a obtenção de cotas superiores **estatísticas** para problemas de otimização estocástica multi-estágio. Estas são variantes dos métodos de Monte Carlo, especializadas para o caso com aversão ao risco, e se dividem também em duas abordagens principais. A primeira, via *importance sampling*, busca reduzir a variância dos estimadores de Monte Carlo, ao explorar a estrutura do problema e a aversão ao risco do tomador de decisão. Já a segunda, via *funções de custo futuro com variáveis extras*, busca construir uma função de custo futuro que, ao ser avaliada em um cenário, forneça uma estimativa do custo médio da política induzida, e com isto permita avaliar a qualidade da política de forma estatística.

Dentre os algoritmos estudados, o que se mostrou mais imediatamente promissor foi a combinação da PDDE com uma etapa *backward* de programação dinâmica interior final, para avaliação de uma cota superior e cálculo do *gap* de convergência. Esta escolha implica num custo computacional

dobrado com relação ao uso puro da PDDE no caso com aversão ao risco, porém fornece um critério de parada com intervalos para os valores de custo, atualmente faltantes na metodologia.

Este projeto teve os seguintes produtos:

- O desenvolvimento e aplicação de metodologias de otimização estocástica multi-estágio, incluindo técnicas avançadas de programação dinâmica, aversão ao risco, representação de funções de custo futuro, e em múltiplos algoritmos para obtenção de cotas superiores.
- A capacitação da equipe do ONS nestas novas metodologias, aprofundando os conhecimentos em otimização matemática e nos diversos algoritmos e conceitos necessários para a construção de funções de custo futuro além das aproximações clássicas por planos cortantes.
- Realização de estudos computacionais para prova de conceito, com problemas de otimização multi-estágio representativos do planejamento da operação do sistema elétrico brasileiro.
- O desenvolvimento de um protótipo computacional flexível que pode ser utilizado para diferentes estudos posteriores de problemas de PDDE, seja via `SDDP.jl` e `DualSDDP.jl`.

1 Introdução

O planejamento da operação do setor elétrico brasileiro é realizado, atualmente, a partir de uma *cadeia de modelos de otimização*. A motivação para esta abordagem é a complexidade do Sistema Interligado Nacional, que é composto por diversas usinas geradoras, com diferentes matrizes energéticas, interligadas por uma rede de transmissão de alta tensão para atender à demanda de energia elétrica em todo o território nacional [ONS]. Além disto, a operação do sistema é afetada por incertezas, como a variação da demanda de energia, a disponibilidade de geração e a ocorrência de falhas na rede de transmissão. No Brasil, em que a matriz energética tem uma forte componente hidrelétrica, a operação do sistema é ainda mais complexa, pois a geração de energia é fortemente dependente das condições hidrológicas, que possuem variações sazonais e regionais, além de serem afetadas por condições de longo prazo, com períodos de alta e baixa afluência, relativos à média histórica, que podem durar anos. Para mitigar estas incertezas, a operação do sistema é planejada a partir de modelos de otimização que buscam minimizar custos, garantindo a segurança energética, considerando as restrições físicas e operacionais do sistema.

Dadas as incertezas de médio prazo, a cadeia de modelos atualmente vigente no planejamento da operação do sistema elétrico brasileiro é composta por modelos de otimização estocástica, que consideram a incerteza na afluência das usinas hidrelétricas. O primeiro modelo desta cadeia é um modelo de médio prazo, que divide o período de planejamento em etapas mensais, considerando a operação do sistema em um horizonte de planejamento de cinco anos. Do ponto de vista da cadeia de modelos, o principal resultado deste modelo é a construção de *funções de custo futuro*, que determinam o valor da água, e serão usadas no modelo seguinte, que é um modelo de otimização de curto prazo, para realizar o planejamento da operação do sistema em etapas de uma semana, com um horizonte de dois meses. Esta função de custo futuro serve, portanto, como a *condição de fronteira* do problema de curto prazo, e é responsável por transmitir a informação de médio prazo para o modelo de curto prazo.

Recentemente, com a crescente participação de fontes intermitentes, como a energia eólica e solar, na matriz energética brasileira, a operação do sistema elétrico tem se tornado ainda mais complexa, pois estas fontes possuem características de geração que são diferentes das usinas hidrelétricas e térmicas, que são despacháveis. Com o aumento da complexidade do sistema, tem havido um movimento para aprimorar a modelagem da operação do sistema, com o objetivo de melhorar a eficiência do planejamento e a segurança energética. Entretanto, ao aumentar a complexidade dos modelos, a qualidade da solução dos problemas de otimização correspondentes pode ser afetada, pois a solução destes problemas se torna mais difícil, devido ao aumento da dimensionalidade das variáveis, e das correspondentes funções de custo futuro.

1.1 Qualidade das soluções e aversão ao risco

Tradicionalmente, ao resolver um problema de otimização convexa (que é o caso do modelo de médio prazo), é comum dispor tanto de cotas superiores quanto de cotas inferiores para a função objetivo. Por exemplo, ao avaliar a função objetivo em uma solução viável, é possível obter uma cota superior para o valor ótimo de um problema de minimização, enquanto que o problema dual correspondente fornece uma cota inferior. Com isto, é possível avaliar a qualidade da solução obtida, e determinar um critério de parada para o algoritmo de otimização, baseado em uma tolerância especificada pelo usuário. Entretanto, ao considerar problemas de otimização estocástica multi-estágio, a obtenção de *cotas superiores* para a função objetivo é mais complexa, pois a função objetivo é uma esperança, ao longo de todos os cenários possíveis, o que necessita avaliar um número exponencial de valores, através da árvore de cenários.

Ao resolver um problema de otimização estocástica multi-estágio através de técnicas de programação dinâmica, o algoritmo de Programação Dinâmica Dual Estocástica (PDDE) [PP91] é capaz de obter cotas inferiores para a função objetivo, ao construir aproximações lineares por partes, e também inferiores, para as funções de custo futuro de cada estágio do problema. A vantagem, inclusive, de construir tais aproximações, em oposição à construção de uma solução viável para o problema de otimização multi-estágio, é que tal solução viável requer especificar todas as variáveis de decisão ao longo de todos os cenários possíveis, o que é inviável em problemas com um grande número de estágios, pois estes em geral levam a um número exponencial de variáveis de decisão. Assim, ao invés de construir uma solução viável, o algoritmo de PDDE constrói uma aproximação para as funções de custo futuro. Estas funções de custo futuro, por sua vez, podem ser usadas para determinar uma solução viável de forma *implícita*: para cada estágio, para cada valor da variável de estado, e para cada realização possível da incerteza, a função de custo futuro do estágio seguinte permite determinar uma variável de decisão viável *para aquele estágio*, sem a necessidade de especificar uma trajetória futura até o final do horizonte. Este procedimento de construção de uma solução viável é conhecido como *política de controle induzida pelas funções de custo futuro*, e é uma característica dos algoritmos de programação dinâmica.

Ao construir tais políticas a partir das funções de custo futuro, a única garantia que possuímos é que o custo médio da política será maior do que ou igual à cota inferior obtida, o que é pouco útil na prática. Originalmente, o algoritmo de PDDE foi proposto para resolver problemas de otimização estocástica multi-estágio onde a função objetivo era a média dos custos de operação em todos os estágios, ao longo dos cenários. Neste caso, é possível calcular uma estimativa de Monte Carlo para a média dos custos, e comparar com a cota inferior obtida pelo algoritmo. Realizando um número suficiente de simulações, é possível reduzir a variância do estimador de Monte Carlo, e com isto obter uma garantia *estatística* do custo da política induzida. Quando este custo for suficientemente próximo da cota inferior, é possível considerar a solução obtida como satisfatória, e interromper o algoritmo de PDDE.

Uma das mudanças da modelagem do planejamento da operação do sistema elétrico brasileiro é a consideração de mecanismos de aversão ao risco na função objetivo [STdCS13]. A principal motivação desta mudança é representar a assimetria entre os custos de déficit de energia (em situações de escassez hídrica) e as economias realizadas (em situações de afluências acima da média). Com isto, a operação do setor se tornou, de fato, mais conservadora, pois na função de custo futuro está embutido, implicitamente, um prêmio por risco, que reduz a dispersão da solução obtida em relação à média.

Entretanto, esta mudança na modelagem da função objetivo torna a avaliação da qualidade da solução obtida mais complexa: se, por um lado, as cotas inferiores (construídas a partir de aproximações lineares por partes das funções de custo futuro) podem ser adaptadas para considerar a aversão ao risco, a presença de mecanismos de aversão ao risco na função objetivo torna *inválidas* as estimativas de Monte Carlo para a avaliação da função objetivo ao longo da política induzida. Com isto, perde-se a garantia estatística da qualidade da solução obtida através das funções de custo futuro, e também perde-se um critério de parada para o algoritmo de PDDE no caso com aversão ao risco. Mais ainda, sem avaliar um critério de qualidade da política, também abrimos mão de avaliar a qualidade das funções de custo futuro construídas pelo algoritmo de PDDE, que são fundamentais para os modelos seguintes da cadeia de planejamento da operação do setor elétrico.

1.2 Objetivo e organização do relatório

Neste projeto, analisamos algoritmos e modelos cujo objetivo é construir cotas superiores para a função objetivo de um problema de otimização estocástica multi-estágio, com aversão ao risco,

visando obter um critério de parada para o algoritmo e uma avaliação da qualidade das soluções obtidas.

Este relatório está dividido em três partes. A primeira contém uma apresentação sucinta do arcabouço teórico relativo à otimização estocástica multi-estágio, enfatizando a abordagem via programação dinâmica e funções de custo futuro, que unificam tanto a apresentação do algoritmo de PDDE como a incorporação da aversão ao risco utilizando medidas de risco. A segunda irá focar na discussão dos modelos matemáticos e algoritmos correspondentes para o cálculo de cotas superiores, detalhando tanto algoritmos que calculam cotas superiores exatas, como algoritmos estatísticos, que resultam em intervalos de confiança para as cotas superiores. A terceira parte apresenta e discute os resultados numéricos de exemplos computacionais para problemas multi-estágio representativos do planejamento da operação, para comparar os métodos apresentados na segunda parte. Encerramos o relatório com uma conclusão do projeto, e indicamos direções de trabalhos futuras.

Parte I
Preliminares

2 Otimização estocástica

Neste projeto, tratamos de problemas de otimização estocástica multi-estágio. A ideia principal é partir de uma decomposição do problema em *estágios*, acoplados através de *variáveis de estado*, e construir a solução para o problema estocástico através da construção das *funções de custo futuro* de cada estágio. Esta decomposição é também conhecida como *Programação Dinâmica*, pois a solução de um estágio depende da solução dos estágios subsequentes.

Neste capítulo, iremos apresentar uma formulação geral para problemas de otimização estocástica multi-estágio, discutir brevemente o algoritmo *L-shaped*, que é um método para resolver o caso de problemas de 2 estágios. Com isto, estaremos prontos para apresentar as técnicas de programação dinâmica para a solução de problemas de otimização estocástica multi-estágio.

No próximo capítulo, descreveremos o algoritmo de *PDDE*, que é uma generalização do *L-shaped* para problemas de mais de 2 estágios, no caso em que é possível explorar a decomposição temporal do problema. No capítulo seguinte, mostraremos como generalizar estes problemas de otimização estocástica para incorporar aversão ao risco, adaptando os algoritmos *L-shaped* e de PDDE.

Com isto feito, a próxima parte irá abordar algoritmos que fornecem cotas superiores para a função objetivo de um problema de otimização estocástica multi-estágio, e discutir a aplicação destes métodos para o caso com aversão ao risco.

2.1 Problemas de otimização estocástica multi-estágio

Um problema de programação linear estocástica multi-estágio pode ser formulado da seguinte maneira:

$$\begin{array}{lllll} \min\limits_{x_i} & c_1^\top y_1 + & \mathbb{E}_{\omega_2}\Big[& c_2^\top y_2 + & \mathbb{E}_{\omega_3}\big[\quad c_3^\top y_3 + \quad \ldots \big] \Big] \\ \text{s.a.} & A_1 x_1 + W_1 y_1 = d_1, & & A_2 x_2 + B_2 x_1 + W_2 y_2 = d_2, & A_3 x_3 + B_3 x_2 + W_3 y_3 = d_3, \quad \ldots \\ & x_1, y_1 \geq 0, & & x_2, y_2 \geq 0, & x_3, y_3 \geq 0, \quad \ldots \end{array} \tag{2.1}$$

em que x_i são as variáveis de estado (por exemplo: armazenamento), y_i são as variáveis de decisão (por exemplo: geração térmica, transmissão), c_i são vetores de custo, A_i e B_i as matrizes de transição de estado entre os estágios, W_i as matrizes de restrições sobre as variáveis de decisão, e d_i são vetores de termos independentes (por exemplo: demanda, energia afluente). Na função objetivo, indicamos por $\mathbb{E}_{\omega_i}$ o operador de esperança condicional, que indica que estamos tomando o valor esperado (média) com relação à variável aleatória ω_i do estágio i, que representa a incerteza do problema.

Notemos que os termos do primeiro estágio são determinísticos, e correspondem à forma-padrão de um problema de programação linear, enquanto os termos dos estágios subsequentes são *estocásticos*. Já nos estágios subsequentes, a restrição de igualdade relaciona o estado inicial x_{i-1} e o estado final x_i, e é conhecida como *restrição dinâmica*. Esta restrição é o que acopla os estágios do problema, e torna as decisões tomadas em um estágio dependentes das decisões tomadas nos estágios anteriores.

Uma forma de visualizar estes problemas é através de uma *árvore de cenários*. Por exemplo, um problema de 3 estágios, com 4 possíveis realizações da incerteza no segundo estágio, e 3 realizações subsequentes no terceiro, pode ser representado pela árvore da Figura 2.1. O primeiro estágio está

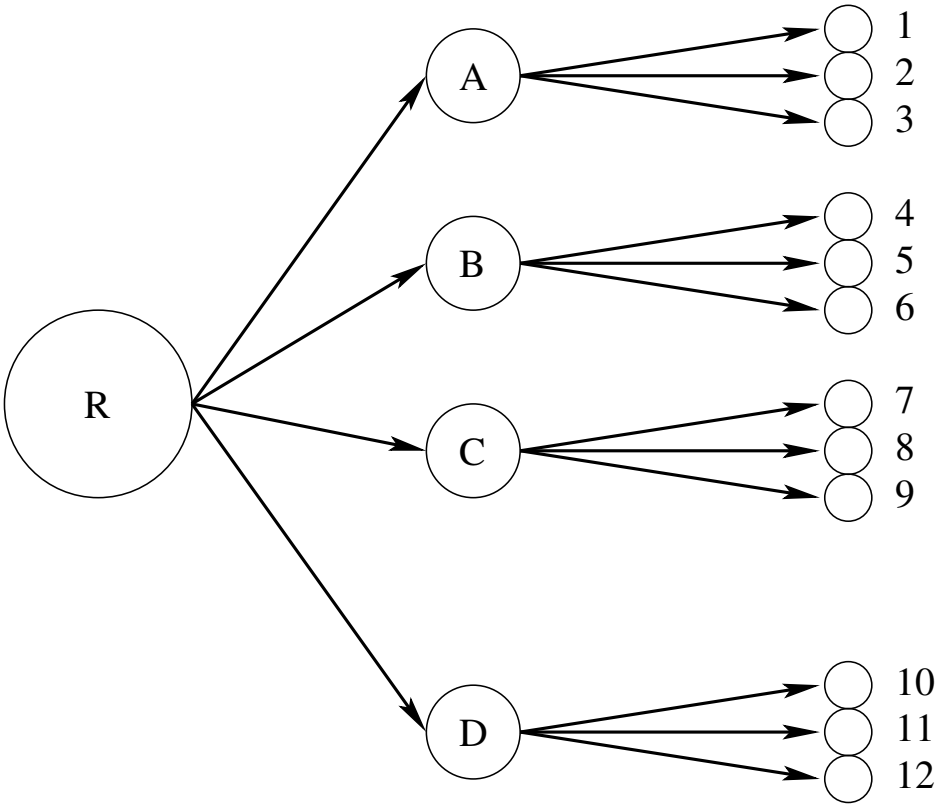


Figura 2.1: Representação em árvore de cenários para um problema de três estágios.

indicado por "R", correspondente à raiz da árvore, e os 4 cenários do segundo estágio pelas letras A, B, C e D. Já os 12 cenários possíveis para o terceiro estágio estão simplesmente numerados.

Da forma como escrito em (2.1), não aparece explicitamente a dependência dos dados do problema (os custos c_t e as matrizes A_t, B_t, W_t e d_t das restrições) em função das variáveis aleatórias ω_i; deve-se subentender que, por exemplo, c_2 depende de ω_2, e assim por diante. Mais ainda, como o estado x_3 depende de x_2, que depende de ω_2, as variáveis de estado x_t dependem, de fato, de toda a história dos cenários *até o tempo* t, $\omega_{[t]} = (\omega_1, \omega_2, \ldots, \omega_t)$. O mesmo se aplica às variáveis de decisão y_t: mesmo que elas não dependam diretamente das decisões anteriores y_{t-1}, elas também dependem de x_{t-1}, que carrega consigo a dependência ao longo de todo o cenário.

Quando o número de cenários é muito grande, a resolução direta de um problema como (2.1) se torna inviável. Ora, se o número de estágios é T e o número de realizações ("aberturas") em cada estágio é N, o número total de cenários é N^T, que cresce muito rapidamente: para $N = 20$ e $T = 12$, este número já é da ordem de $4 \cdot 10^{15}$. Por isto, para problemas já com um número moderado de estágios e cenários, é necessário recorrer a métodos de resolução que explorem a estrutura temporal do problema. A estrutura mais importante aparece quando o problema é *sequencialmente independente*, isto é, quando a variável aleatória ω_t do estágio t é independente de todas as variáveis de estado $x_{[t-1]}$ e variáveis aleatórias $\omega_{[t-1]}$ anteriores.[1] Neste caso, é possível utilizar técnicas de *programação dinâmica*, que constroem a solução do problema de forma recursiva, partindo do último

[1] No caso em que o processo ω_t é exógeno, e por isto independente por construção das variáveis de estado x_{t-1} (e também, naturalmente, das variáveis de decisão y_{t-1}), é comum se referir a esta propriedade como *stagewise independence*. Note que as *coordenadas* das variáveis ω_t podem ser dependentes entre si: o importante é que a distribuição de probabilidade do vetor ω_t seja independente das variáveis anteriores.

estágio e indo até o primeiro.

Para isso, introduzimos as *funções de custo futuro* $V_t(x_{t-1})$, que representam o custo esperado a partir do estágio t, começando com o estado final do estágio anterior, x_{t-1}. Assim, $V_{T+1}(x_T) = 0$, e as funções de custo futuro são definidas recursivamente por:

$$V_t(x_{t-1}) = \mathbb{E}_{\omega_t}\left[V_t(x_{t-1}, \omega_t)\right], \tag{2.2}$$

$$\begin{aligned} V_t(x_{t-1}, \omega_t) = \min_{x_t \geq 0} \quad & c_t^\top y_t + V_{t+1}(x_t) \\ \text{s.a.} \quad & A_t x_t + B_t x_{t-1} + W_t y_t = d_t. \end{aligned} \tag{2.3}$$

Nesta formulação, a dependência das variáveis de estado e decisão, x_t e y_t, do estágio t em relação ao estado x_{t-1} do estágio anterior e à realização ω_t da incerteza no estágio atual é explícita. Assim, o problema de otimização (2.3) é um problema de programação linear determinístico, onde aparece a função de custo futuro V_{t+1} do estágio seguinte.

2.2 Solução *vs* Política

Matematicamente, a solução de um problema de otimização estocástica de T estágios é uma função que associa a cada possível cenário $\omega_{[T]}$ o valor de todas as variáveis de estado e decisão $x_{[T]}$ e $y_{[T]}$ naquele cenário. Se temos um número finito de cenários, uma possibilidade para representar esta solução é através de uma tabela, contendo os valores das variáveis de decisão correspondentes a cada um deles.

Na prática, entretanto, é mais comum descrever a solução de forma indireta, através de uma *política de decisão*, que é uma regra que associa a cada estado x_{t-1} do estágio $t-1$ e uma incerteza ω_t as correspondentes decisões y_t e estado final x_t para o estágio t. Notemos que tal representação só é possível no caso em que o processo estocástico é *sequencialmente independente*; se este não fosse o caso, a política ótima de decisão poderia depender de *toda* a história dos cenários $\omega_{[t-1]}$. Dito de outra forma, se a incerteza futura depender do passado, não basta saber o estado atual x_{t-1} e a realização ω_t, mas é necessário considerar a trajetória inteira $\omega_{[t]}$ porque esta pode influenciar a distribuição da incerteza nos estágios futuros.

Esta forma de determinar uma política é bastante próxima da solução do problema de programação dinâmica: de fato, se conhecemos as funções de custo futuro $V_{t+1}(x_t)$, uma decisão ótima para o estágio t é simplesmente aquela que minimiza o problema (2.3). Mais ainda, se temos apenas uma aproximação para as funções de custo futuro, digamos $\hat{V}_t$, podemos utilizar estas aproximações para determinar uma política de decisão, através da solução de um problema análogo a (2.3), mas substituindo V_{t+1} por $\hat{V}_{t+1}$. Naturalmente, esperamos que, quanto mais próximas das reais funções V_t forem as aproximações $\hat{V}_t$, melhor será a política determinada por elas.

2.3 O algoritmo *L-shaped*

O algoritmo *L-shaped* [VSW69] é um método para resolver problemas de programação estocástica de dois estágios. Este método está na base dos algoritmos de programação dinâmica que iremos apresentar em seguida, e parte da decomposição temporal que apresentamos na seção anterior. De fato, para o caso de dois estágios, a decomposição temporal é elementar: temos $V_3 \equiv 0$, e a única função a ser construída é $V_2(x_1)$, que é a função de custo futuro do primeiro estágio, dada a solução x_1 deste estágio. Note também que neste caso podemos identificar os *cenários* $\omega_{[2]}$ com a realização da incerteza ω_2 no segundo estágio, que é a única variável aleatória do problema (pois também assumimos que não há incerteza no primeiro estágio).

A ideia do algoritmo *L-shaped* é aproximar a função de custo futuro V_2, construída iterativamente através da resolução de problemas de otimização que correspondem ao segundo estágio do

problema original. Suponha que, em uma dada iteração, tenhamos uma aproximação inferior $\underline{V}_2$ para a função V_2. Podemos então resolver o problema de primeiro estágio

$$\begin{aligned} \min_{x_1, y_1 \geq 0} \quad & c_1^\top y_1 + \underline{V}_2(x_1) \\ \text{s.a.} \quad & A_1 x_1 + B_1 x_0 + W_1 y_1 = d_1. \end{aligned} \tag{2.4}$$

Como $\underline{V}_2 \leq V_2$, o valor ótimo deste problema, denotado por $\underline{z}$, é uma cota inferior para o valor ótimo z do problema original.

Seja $(\hat{x}_1, \hat{y}_1)$ uma solução ótima do problema de primeiro estágio. Desejamos completá-la em uma solução viável para o problema original, isto é, encontrar os valores de x_2 e y_2 (um para cada cenário) tais que $A_2 x_2 + B_2 \hat{x}_1 + W_2 y_2 = d_2$. Mais ainda, podemos calcular *exatamente* o valor de $V_2(\hat{x}_1)$ através da resolução do problema de segundo estágio para cada realização ω_2 (lembre que $V_3 \equiv 0$), que também fornecerá os valores de x_2:

$$\begin{aligned} V_2(\hat{x}_1, \omega_2) = \min_{x_2 \geq 0} \quad & c_2^\top y_2 \\ \text{s.a.} \quad & A_2 x_2 + W_2 y_2 = d_2 - B_2 \hat{x}_1. \end{aligned} \tag{2.5}$$

$$V_2(\hat{x}_1) = \mathbb{E}_{\omega_2}\left[V_2(\hat{x}_1, \omega_2)\right]. \tag{2.6}$$

Note que, como a função de custo futuro V_2 já depende de x_1, não é preciso saber o valor de y_1, já que este não aparece no problema de segundo estágio. A última equação reforça que $V_2(\hat{x}_1)$ é a média dos custos ótimos obtidos para cada realização ω_2 da incerteza de segundo estágio.

Se $V_2(\hat{x}_1) = \underline{V}_2(\hat{x}_1)$, então $\hat{x}_1$ é uma solução ótima do problema de primeiro estágio, e o algoritmo termina, pois a solução do problema de primeiro estágio dá $\underline{z} \leq z$, e o cálculo exato de $V_2(\hat{x}_1)$ dá um ponto viável com custo $z = \underline{z}$.

Se, por outro lado, $V_2(\hat{x}_1) > \underline{V}_2(\hat{x}_1)$, a solução dos problemas de programação linear (2.5) nos fornece multiplicadores de Lagrange que permitem escrever uma desigualdade para a função V_2:

$$V_2(x_1, \omega_2) \geq V_2(\hat{x}_1, \omega_2) + \pi(\omega_2)^\top (x_1 - \hat{x}_1). \tag{2.7}$$

Esta desigualdade é conhecida como *corte de Benders para a função* $V_2(\cdot, \omega_2)$. Tomando a média, obtemos

$$V_2(x_1) \geq V_2(\hat{x}_1) + \pi^\top (x_1 - \hat{x}_1), \tag{2.8}$$

onde $\pi = \mathbb{E}_{\omega_2}[\pi(\omega_2)]$. O uso destas desigualdades para a média (e não as desigualdades para cada cenário) é conhecido como *método de cortes médios* e *método "single-cut"*, por oposição aos *métodos "multi-cuts"*, que consideram os cortes obtidos individualmente para cada um dos $\pi(\omega_2)^\top$, resultando em cortes para cada uma das funções $V_2(x_1, \omega_2)$ como em (2.7).

Na prática, o algoritmo *L-shaped* substitui o problema de primeiro estágio (2.4) por um problema com uma variável suplementar θ_1, que representa a aproximação $\underline{V}_2$ para a função de custo futuro, acompanhada por uma restrição inicial para esta variável, para evitar que o problema tenha uma solução com $\theta_1 \to -\infty$:

$$\begin{aligned} \min_{x_1, y_1 \geq 0} \quad & c_1^\top y_1 + \theta_1 \\ \text{s.a.} \quad & A_1 x_1 + B_1 x_0 + W_1 y_1 = d_1, \\ & \theta_1 \geq \theta_{\min}. \end{aligned} \tag{2.9}$$

A partir daí, o algoritmo calcula sucessivamente uma solução $\hat{x}_1$ deste problema, e adiciona cortes (ou seja, restrições ligando θ_1 e x_1) para melhorar a aproximação correspondente $\underline{V}_2$ para a função de custo futuro. Este processo é repetido até que a solução do problema de primeiro estágio dê

Algoritmo 1 Algoritmo *L-shaped*

Require: Estado inicial x_0, tolerância $\varepsilon > 0$.
Inicialize $t = 1$, $\underline{z} = -\infty$, $\hat{V}_t = 0$ para $t = 1, \dots, T$.
while não convergiu **do**
  Resolva o problema de primeiro estágio aproximado (2.9) ▷ Primeiro estágio
  Seja $\hat{x}_1$ o estágio final candidato, e $\hat{\theta}$ o custo futuro estimado.
  $\underline{z} = \max\{\underline{z}, c_1^\top \hat{y}_1 + \hat{\theta}\}$
  for ω_2 nos N_2 cenários do segundo estágio **do** ▷ Segundo estágio
    Resolva o problema de segundo estágio (2.5) para obter $V_2(\hat{x}_1, \omega_2)$
    Também calcule os multiplicadores de Lagrange $\hat{\pi}(\omega_2)$
  end for
  $\hat{v} = V_2(\hat{x}_1) =$ média dos custos $V_2(\hat{x}_1, \omega_2)$
  if $V_2(\hat{x}_1) \leq \hat{\theta} + \varepsilon$ **then**
    convergiu
  end if
  $\hat{\pi} =$ média dos multiplicadores $\hat{\pi}(\omega_2)$ ▷ Cálculo dos cortes médios
  Adicione o corte $\theta_1 \geq \hat{v} + \hat{\pi}^\top (x_1 - \hat{x}_1)$ ao problema do primeiro estágio (2.9).
end while

uma cota inferior suficientemente próxima do custo calculado exatamente para o segundo estágio, e está descrito no Algoritmo 1.

Uma das vantagens do algoritmo *L-shaped* é que ele garante a convergência para a solução ótima do problema original. Mais ainda, a cada iteração possuímos tanto uma cota inferior para o valor ótimo do problema, $\underline{z}$ calculada ao resolver o problema de primeiro estágio aproximado (2.9), quanto uma solução viável para o problema original, $\hat{x}_1$, e com ela uma cota superior para o mesmo, $\overline{z} = c_1^\top \hat{y}_1 + V_2(\hat{x}_1)$, calculada ao final da iteração em (2.6). Vale notar que, como a cada iteração incluímos mais restrições no problema de primeiro estágio, a função de custo futuro $\underline{V}_2$ é crescente ao longo das iterações, e com isto a cota inferior $\underline{z}$ é monotonamente crescente. Por outro lado, a cota superior $\overline{z}$ depende de $\hat{x}_1$ e $\hat{y}_1$, e pode oscilar ao longo das iterações.

3 O Algoritmo de PDDE

Neste capítulo, iremos apresentar o algoritmo de Programação Dinâmica Dual Estocástica (PDDE), que é uma extensão do algoritmo *L-shaped* para problemas de mais de dois estágios, proposto em [PP91] exatamente no caso do problema de planejamento da operação do setor elétrico. Nosso objetivo é dar o arcabouço geral que será, em seguida, detalhado nos capítulos seguintes para os algoritmos de aproximação interior que permitem calcular cotas superiores para problemas de otimização estocástica multi-estágio.

Iremos começar com uma breve revisão dos problemas de otimização estocástica multi-estágio, em particular definindo as funções de custo futuro (também conhecidas como *funções valor ótimo*) de cada estágio. Em seguida, apresentaremos o algoritmo de PDDE no caso neutro ao risco. No próximo capítulo, ao introduzirmos aversão ao risco, iremos discutir como incorporá-la ao algoritmo.

3.1 Recorrência para problemas de otimização estocástica multi-estágio

Relembramos, aqui, as equações (2.1) que descrevem um problema de programação linear estocástica multi-estágio:

$$\begin{array}{lllll} \min\limits_{x_i} & c_1^\top y_1 + & \mathbb{E}_{\omega_2}\Big[& c_2^\top y_2 + & \mathbb{E}_{\omega_3}[\quad c_3^\top y_3 + \quad \ldots] \Big] \\ \text{s.a.} & A_1 x_1 + W_1 y_1 = d_1, & & A_2 x_2 + B_2 x_1 + W_2 y_2 = d_2, & A_3 x_3 + B_3 x_2 + W_3 y_3 = d_3, \quad \ldots \\ & x_1, y_1 \geq 0, & & x_2, y_2 \geq 0, & x_3, y_3 \geq 0, \quad \ldots \end{array} \tag{3.1}$$

e a recorrência de programação dinâmica (2.2) ligando as funções de custo futuro V_t de cada estágio t no caso em que as realizações das incertezas são *stagewise independent*:

$$V_{T+1}(x_T) = 0, \tag{3.2}$$

$$V_t(x_{t-1}) = \mathbb{E}_{\omega_t}\left[V_t(x_{t-1}, \omega_t)\right], \tag{3.3}$$

$$\begin{aligned} V_t(x_{t-1}, \omega_t) = \min_{x_t \geq 0} \quad & c_t^\top y_t + V_{t+1}(x_t) \\ \text{s.a.} \quad & A_t x_t + B_t x_{t-1} + W_t y_t = b_t. \end{aligned} \tag{3.4}$$

Ao longo dos algoritmos de programação dinâmica que veremos, será importante ter uma notação não apenas para as funções de custo futuro V_t, mas também para a operação de recorrência dada pelas equações (3.3) e (3.4). Esta operação é chamada de "operador de Bellman", e pode ser vista como tomando uma função f qualquer como função de custo futuro, e retornando o custo "uma etapa antes". Mais formalmente, definimos $\mathcal{B}_t f$ como a função que, para um estado x_{t-1}, retorna o custo esperado de um estágio t:

$$\mathcal{B}_t f(x_{t-1}) = \mathbb{E}_{\omega_t}\left[\min_{x_t, y_t \geq 0}\left\{c_t^\top y_t + f(x_t) \quad \text{s.a.} \quad A_t x_t + B_t x_{t-1} + W_t y_t = d_t\right\}\right]. \tag{3.5}$$

Como o problema para um estágio pode ser diferente em relação aos outros (por exemplo ao mudar a distribuição da incerteza, ou as próprias restrições), cada estágio dá origem a um operador de Bellman diferente, e por isto temos os índices em $\mathcal{B}_t$.

As funções de custo futuro do problema de otimização estocástica são dadas por $V_t = \mathcal{B}_t V_{t+1}$ para todo $t \in \{1, 2, \ldots, T\}$, e $V_{T+1} \equiv 0$. Na prática, estas funções de custo futuro são desconhecidas, e o objetivo dos algoritmos de programação dinâmica é aproximar estas funções, de forma a obter uma solução para o problema de otimização (3.1). Todos os algoritmos que veremos nesta seção e em várias das seguintes são baseados na aproximação iterativa destas funções de custo futuro, pela aplicação repetida do operador de Bellman (3.5).

3.2 O algoritmo de PDDE

O algoritmo de *Programação Dinâmica Dual Estocástica* (PDDE; em inglês, *Stochastic Dual Dynamic Programming*, SDDP) é uma extensão do algoritmo *L-shaped* para problemas de mais de dois estágios, no caso em que o processo estocástico é *sequencialmente independente* (em inglês, *stagewise independent*). Esta hipótese permite reduzir a complexidade em memória do algoritmo, pois basta calcular e armazenar as aproximações das funções de custo futuro V_t para cada estágio t.

De forma sucinta, cada iteração do algoritmo de PDDE é composta por duas fases:

1. **Fase *Forward***: selecionamos um cenário $\omega_{[T]} = (\omega_1, \ldots, \omega_T)$ e resolvemos os problemas de cada estágio t, indo de $t = 1$ até $t = T$, para determinar uma *trajetória viável* $\hat{x}_{[T]} = (\hat{x}_1, \ldots, \hat{x}_T)$ para este cenário.

2. **Fase *Backward***: a partir dos estados finais $\hat{x}_{[T]} = (\hat{x}_1, \ldots, \hat{x}_T)$ obtidos na fase anterior, calculamos *cortes de Benders* para as funções de custo futuro V_t de cada estágio t, indo de $t = T$ até $t = 1$. Nesta fase, é preciso calcular os multiplicadores de Lagrange π_t para cada possível realização da variável aleatória ω_t, e é a etapa mais custosa computacionalmente. Por esta razão, ao calcular os cortes para cada estágio, já utilizamos a nova função de custo futuro V_{t+1} atualizada pelos cortes desta iteração.

De forma mais detalhada, o algoritmo de PDDE mantém uma coleção de funções de custo futuro aproximadas $\hat{V}_t$ para cada estágio t. Assim, o algoritmo aproxima a recorrência dada pelas equações (3.3) e (3.4) através de funções $\hat{V}_t$ que são monotonamente crescentes para as funções de custo futuro V_t. Portanto, em vez de resolver o problema dado pela equação (3.4), o algoritmo de PDDE resolve um problema de programação linear estocástica com a seguinte função objetivo:

$$\min_{x_t, y_t \geq 0} \left\{ c_t^\top y_t + \hat{V}_{t+1}(x_t) \right\}, \quad \text{s.a.} \quad A_t x_t + B_t x_{t-1} + W_t y_t = d_t. \tag{3.6}$$

Note que o resultado desta operação *em geral não é igual* a $\hat{V}_t(x_{t-1})$. De fato, o objetivo do algoritmo é justamente construir aproximações para $\hat{V}_t$ através da avaliação "um passo a frente" dada pela equação (3.6), e que com isto irão convergir para as funções de custo futuro V_t. Com a notação introduzida no final da seção 3.1, o problema corresponde a calcular $\mathcal{B}_t \hat{V}_{t+1}$ no ponto x_{t-1}, e em geral é maior do que a aproximação atual $\hat{V}_t(x_{t-1})$.

O algoritmo de PDDE constrói aproximações para as funções de custo futuro, ou seja (introduzindo um índice k para a iteração do algoritmo) as funções de custo futuro aproximadas $\hat{V}_t^{(k)}$ satisfazem para todo estágio t e toda iteração k:

- $\hat{V}_t^{(k)} \leq V_t$; e
- $\hat{V}_t^{(k)} \leq \hat{V}_t^{(k+1)}$.

A primeira propriedade é a garantia de que as aproximações $\hat{V}_t$ são cotas inferiores para as funções de custo futuro V_t, e depende da escolha inicial das funções $\hat{V}_t^{(0)}$, que é responsabilidade de quem irá utilizar o algoritmo. Em vários problemas, é simples obter uma cota inferior absoluta, por exemplo $\hat{V}_t(0) \geq 0$ em um problema onde os custos c_t são não-negativos. Já a segunda decorre naturalmente dos passos do algoritmo, pois a cada iteração incluímos um novo corte para a função $\hat{V}_t$. Portanto, a PDDE constrói aproximações *crescentes* para as funções de custo futuro, e é apresentada de forma mais detalhada no Algoritmo 2.

Ao contrário do algoritmo *L-shaped*, o algoritmo de PDDE é *estocástico*, no sentido que sua execução depende da escolha dos cenários $\omega_{[T]}$ na fase *Forward*. Assim, as aproximações $\hat{V}_t^{(k)}$ construídas ao longo das iterações podem ser diferentes, em função dos cenários e dos consequentes estados $\hat{x}_{[T]}$ obtidos. Entretanto, este é um algoritmo convergente, o que significa que, ao longo das iterações, as aproximações $\hat{V}_t^{(k)}$ convergem para as funções de custo futuro V_t do problema original.

Mas a principal diferença entre este algoritmo e o *L-shaped* é que, a cada iteração, não calculamos o custo total da política de decisão corrente, pois apenas calculamos o custo em um (ou um pequeno número de) cenários. Assim, apesar de ainda possuirmos uma cota inferior $\underline{z}$ para o valor ótimo do problema, obtida através da solução do problema de primeiro estágio (e usando o fato que as aproximações $\hat{V}_t$ são monotonamente crescentes para as reais funções de custo futuro V_t, e portanto $\hat{V}_2 \leq V_2$), o algoritmo não calcula uma cota superior $\overline{z}$.

Algoritmo 2 Algoritmo de Programação Dinâmica Dual Estocástica (PDDE)

Require: Estado inicial x_0, número de iterações K
Inicialize $t = 1$, $\underline{z} = -\infty$, $\hat{V}_t = 0$ para $t = 1, \dots, T$.
for $k = 1, \dots, K$ **do**
  Amostre o cenário $\omega_{[T]}$ a partir das incertezas $\omega_{t,i}$ de cada estágio
  $\hat{x}_0 = x_0$
  for $t = 1, \dots, T$ **do** ▷ Fase *Forward*
    Resolva o problema (3.6) do estágio t com incerteza ω_t para obter o estado final $\hat{x}_t$
  end for
  for $t = T, \dots, 2$ **do** ▷ Fase *Backward*
    $\hat{\pi}_t = 0$, $\hat{v}_t = 0$
    for $\omega_{t,i}$ nas N_t realizações de ω_t do estágio t **do**
      Resolva o problema (3.6) para obter $\hat{\pi}_t(\omega_{t,i})$ e $\hat{v}_t(\omega_{t,i})$
      $\hat{\pi}_t += \hat{\pi}_t(\omega_{t,i})$
      $\hat{v}_t += \hat{v}_t(\omega_{t,i})$
    end for
    $\hat{\pi}_t = \hat{\pi}_t / N_t$ e $\hat{v}_t = \hat{v}_t / N_t$ ▷ Cálculo dos cortes médios
    Adicione o corte $\theta_t \geq \hat{v}_t + \hat{\pi}_t^\top (x_{t-1} - \hat{x}_{t-1})$ à função $\hat{V}_t$ do problema do estágio $t-1$.
  end for
end for

4 Aversão ao risco

Existem diversas maneiras de se incorporar a aversão ao risco em problemas de otimização estocástica. Neste projeto, utilizaremos uma abordagem que consiste em *ponderar* os cenários, de forma a dar mais importância aos cenários com maior custo. Assim, o objetivo deixa de ser minimizar o custo esperado, e substituímos esta função objetivo por uma *medida de risco*. Uma medida de risco, de forma geral, *transforma uma variável aleatória em um número real*, para podermos, em seguida, comparar estas variáveis aleatórias (no caso do problema de planejamento do setor elétrico, o custo de operação) entre si. O uso do valor esperado como métrica de comparação é uma escolha comum, mas vale notar que, de forma geral, uma comparação entre variáveis aleatórias envolve transformar uma informação complexa (um custo para cada um dos cenários considerados, no exemplo) em um único número real.

Neste projeto, nos concentramos em medidas de risco que são combinações entre a média e o CVAR, que são bastante comuns na literatura. O CVAR (do inglês *Conditional Value-at-Risk*, Valor Condicional ao Risco) de nível α corresponde a calcular a média dos α cenários com maior custo, e exatamente por isto é uma medida que permite reduzir o impacto dos cenários mais extremos. Na próxima sessão, iremos apresentar uma definição do CVAR (e de um análogo clássico, o VAR), e em seguida apresentaremos como incorporar a aversão ao risco nos algoritmos de otimização estocástica, primeiro no *L-shaped*, e em seguida na PDDE.

4.1 *Value-at-Risk* e *Conditional Value-at-Risk*

Dada uma variável aleatória Z, o seu *Value-at-Risk* a um nível de confiança α (denotado por VAR_α) é o valor q tal que a probabilidade de Z ser maior ou igual a q é igual a α. A notação q remete a "quantil". O *Conditional Value-at-Risk* a um nível de confiança α (denotado por CVAR_α) é a média dos valores de Z que estão acima do seu VAR_α. Assim, o CVAR_α é a média dos α piores casos,

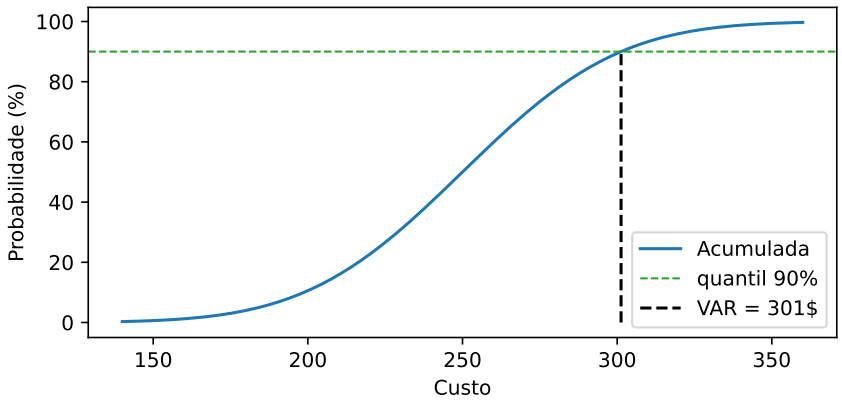


Figura 4.1: Determinação do *Value-at-Risk* a partir da função de probabilidade acumulada.

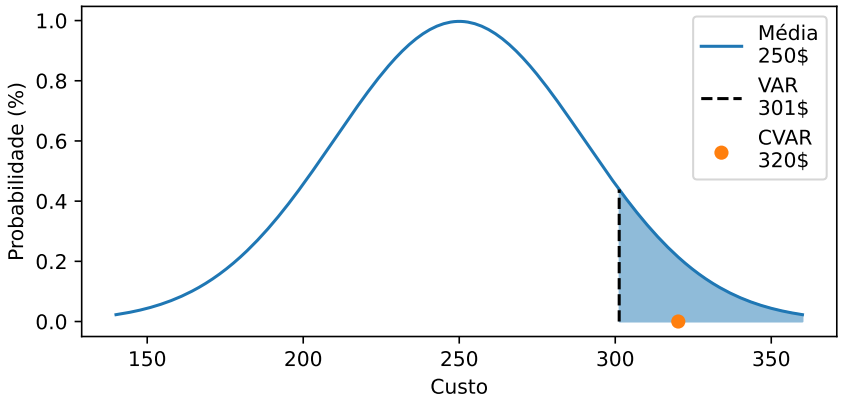


Figura 4.2: *Value-at-Risk* e *Conditional Value-at-Risk* em uma distribuição de custos.

onde a noção de "pior" é dada pelo mais alto. O cálculo do VAR a partir da distribuição acumulada pode ser visto na Figura 4.1, e a relação entre o VAR e o CVAR é ilustrada na Figura 4.2.

Comentário 4.1. *Esta definição é apropriada para o caso em que a variável Z representa custos, e portanto a probabilidade relevante para quantificar riscos é a de que o custo (Z) ultrapasse um certo valor (o VAR). Adotamos aqui esta definição porque o foco do projeto são problemas de otimização onde buscamos* minimizar *os custos. Se fosse o caso de* maximizar *os lucros, a definição do Value-at-Risk mudaria, e, consequentemente, o Conditional Value-at-Risk seria a média de Z condicionada a estar* abaixo *do α-quantil, que representariam as maiores* perdas. *Assim, é importante ter em mente que a interpretação do VAR_α (e, consequentemente, do CVAR_α) depende do sentido atribuído à variável Z.*

Matematicamente, temos

$$\begin{aligned} \text{VAR}_\alpha &= \inf \{\, q \in \mathbb{R} \mid \mathbb{P}(Z \leq q) \geq 1 - \alpha \,\} \text{ e} \\ \text{CVAR}_\alpha &= \mathbb{E}\,[\, Z \,|\, Z \geq \text{VAR}_\alpha(Z) \,]\,, \end{aligned}$$

onde, na definição do VAR, temos a expressão $\mathbb{P}(Z \leq q)$, que indica a probabilidade de a variável aleatória Z ser inferior ou igual a q. Esta escolha (em vez de trabalhar com a probabilidade de

Z ser superior a q como mais natural pela definição) vem do fato que a função de distribuição acumulada de Z, $F_Z(q)$, é definida por $\mathbb{P}(Z \leq z)$.

O CVAR de uma variável aleatória discreta Z pode ser calculado diretamente, a partir de suas realizações z_i e suas probabilidades p_i, bastando ordenar as realizações z_i em ordem decrescente e acumular as probabilidades até que elas atinjam a probabilidade acumulada igual a α. Entretanto, tal procedimento não é conveniente no contexto em que o CVAR aparece em um problema de otimização (e, em particular, para o cálculo de cortes). Por isto, damos a seguir duas outras formas de calcular o CVAR, ambas através de um problema de otimização:

$$\begin{aligned} \text{CVAR}_\alpha(Z) = \min_{q,u_i} \quad & q + \tfrac{1}{\alpha}\textstyle\sum_{i=1}^n p_i \cdot u_i && (4.1)\\ \text{s.a.} \quad & q + u_i \geq z_i, \\ & u_i \geq 0. \\ = \max_{\zeta_i} \quad & \textstyle\sum_{i=1}^n \zeta_i p_i \cdot z_i && (4.2)\\ \text{s.a.} \quad & \textstyle\sum_{i=1}^n \zeta_i p_i = 1, \\ & 0 \leq \zeta_i \leq 1/\alpha. \end{aligned}$$

A primeira, equação (4.1), é a fórmula de Rockafellar e Uryasev [RU00], e permite transformar um problema de otimização estocástico com aversão ao risco em um problema neutro ao risco, com a introdução da variável auxiliar q, como veremos na seção seguinte onde adaptaremos o algoritmo *L-shaped* para incorporar aversão ao risco.

Já a segunda expressão (4.2), dada por um problema de maximização, é bastante próxima da interpretação de CVAR como a média ponderada dos cenários com maior custo. De fato, ao resolver este problema, obtemos um vetor ζ que corresponde a novos pesos para os cenários: o fato de a soma $\sum \zeta_i p_i$ ser igual a 1 garante que ainda obtemos uma distribuição de probabilidade. Já o limite da reponderação ser $1/\alpha$ (que é maior do que 1, já que $\alpha < 1$) garante que vamos incluir cenários cuja probabilidade original some, pelo menos, α. De fato, temos que $\sum p_i = \alpha \sum p_i/\alpha \geq \alpha \sum \zeta_i p_i = \alpha$.

Uma observação interessante é que os problemas de otimização correspondentes às fórmulas (4.1) e (4.2) são *duais* um do outro.

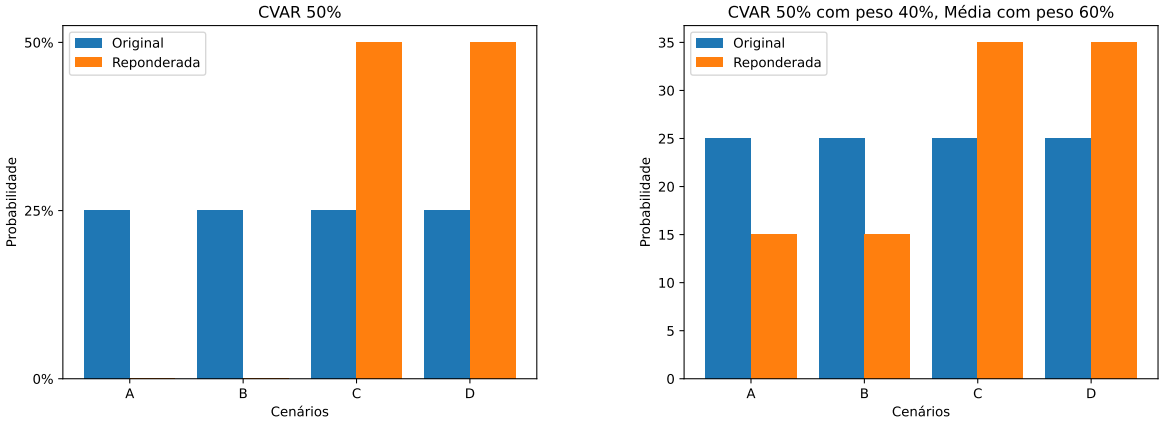
As fórmulas (4.1) e (4.2) possuem versões para o caso contínuo, que generalizam as anteriores:

$$\begin{aligned} \text{CVAR}_\alpha(Z) &= \min_q \; q + \frac{1}{\alpha}\mathbb{E}\left[(Z-q)_+\right] && (4.3)\\ &= \max_{\zeta \in Q_\alpha} \; \mathbb{E}\left[\zeta Z\right]. && (4.4) \end{aligned}$$

Na primeira, vemos a expressão $(Z-q)_+$, que é a parte positiva da diferença $Z-q$, e corresponde, intuitivamente, à parte das variáveis aleatórias que está acima do valor (determinístico) q. Na formulação discreta da equação (4.1), esta parte positiva é representada explicitamente pela variável auxiliar u_i. Na segunda, o conjunto Q_α corresponde à generalização das restrições do problema (4.2), e é dado por $\{\, \zeta \mid 0 \leq \zeta \leq 1/\alpha, \mathbb{E}[\zeta] = 1 \,\}$. Note que a restrição $\zeta \leq 1/\alpha$ garante que ζ é uma variável aleatória limitada, ou seja, em L^∞.

Vale notar que o valor ótimo da variável q, presente nos problemas de minimização, coincide com o VAR_α de Z. Por isto, utilizaremos a fórmula (4.3) para representar o CVAR em problemas de otimização multi-estágio, pois ela é ao mesmo tempo compacta e de fácil interpretação.

No caso em que a medida de risco é uma combinação da média com o CVAR de nível α (em vez de tomarmos apenas a média ou o CVAR), por exemplo $\rho = (1-\beta)\mathbb{E} + \beta\text{CVAR}_\alpha$, com $\beta \in [0,1]$



(a) $\text{CVAR}_{50\%}$. (b) 60% Esperança + 40% do $\text{CVAR}_{50\%}$.

Figura 4.3: Reponderação de cenários com diferentes medidas de risco

sendo o peso do CVAR, as formulações acima podem ser generalizadas. Denotando esta medida de risco por $\rho_{\alpha,\beta}$, temos:

$$\begin{aligned} \rho_{\alpha,\beta}(Z) = \min_{q,u_i} \quad & (1-\beta)\sum_{i=1}^{n} p_i \cdot z_i + \beta\left(q + \tfrac{1}{\alpha}\sum_{i=1}^{n} p_i \cdot u_i\right) && (4.5) \\ \text{s.a.} \quad & q + u_i \geq z_i, \\ & u_i \geq 0. \\ = \max_{\zeta_i} \quad & \sum_{i=1}^{n} \zeta_i p_i \cdot z_i && (4.6) \\ \text{s.a.} \quad & \sum_{i=1}^{n} \zeta_i p_i = 1, \\ & (1-\beta) \leq \zeta_i \leq (1-\beta) + \beta/\alpha. \end{aligned}$$

Vejamos um exemplo para ilustrar o efeito das medidas de risco sobre o custo futuro. Imagine que a medida de risco utilizada é o CVAR com $\alpha = 0.5$, e que há 4 realizações possíveis para a incerteza de segundo estágio, correspondentes aos cenários A, B, C e D, que supomos equiprováveis (ou seja, $p_i = 25\%$). Suponha que o custo futuro para cada um destes cenários seja $z_i = 1$, 2, 3 e 4, respectivamente. Ora, os dois cenários mais caros serão C e D, e portanto o CVAR irá calcular a média destes dois cenários, o que corresponde a $\zeta_i p_i = 50\%$ para eles, e 0 para os demais. A média ponderada dos custos futuros será 3.5, e os pesos correspondentes são $\zeta = (0, 0, 2, 2)$.

Já para uma medida de risco que combina o CVAR com a média, digamos 60% para a média e 40% para o CVAR, a probabilidade ajustada de cada cenário será, respectivamente, 15%, 15%, 35% e 35%. Neste caso, apenas a parte correspondente ao CVAR (40%) será redistribuída, ficando nos cenários C e D. O risco será $0.6 \cdot (1+2+3+4)/4 + 0.4 \cdot (3+4)/2 = 2.9$, e os pesos correspondentes são $\zeta = (0.6, 0.6, 0.6 + 0.4 \cdot 2, 0.6 + 0.4 \cdot 2) = (0.6, 0.6, 1.4, 1.4)$.

Esses exemplos de transferência de probabilidade podem ser vistos nas figuras 4.3a e 4.3b, onde indicamos as probabilidades finais $\zeta_i p_i$ de cada cenário, para os dois exemplos.

Enfim, ressaltamos que a fórmula (4.2) do CVAR como um máximo de médias ponderadas pode ser generalizada ainda mais de forma relativamente simples. De fato, basta considerar um conjunto viável Q mais geral para os pesos ζ, desde que ainda satisfaçam a condição de que a soma $\sum_{i=1}^{n} \zeta_i p_i$ seja 1. Esta abordagem é bastante conveniente para a aplicação em problemas de otimização estocástica, pois é possível adaptar a construção dos cortes médios para qualquer medida de risco que possa ser expressa como um problema de maximização dos pesos ζ, como veremos adiante.

4.2 *L-shaped* para problemas com aversão ao risco

Como anunciamos, a representação do CVAR através do problema de otimização (4.2) também é a chave para calcular os *cortes* para a medida de risco correspondente: ao determinar o CVAR a partir dos custos de cada realização ω_t, podemos construir um vetor $\hat{\zeta}$ correspondente aos valores $1/\alpha$ para as realizações com maior custo, e 0 para as demais (e talvez um valor intermediário, para que a soma dê um, para a realização que corresponde exatamente ao quantil $1-\alpha$).

Retomando a decomposição de um problema de dois estágios visto na seção 2.3, o problema de primeiro estágio continua inalterado

$$\begin{aligned} \min_{x_1, y_1 \geq 0} \quad & c_1^\top y_1 + \underline{V}_2(x_1) \\ \text{s.a.} \quad & A_1 x_1 + B_1 x_0 + W_1 y_1 = d_1. \end{aligned} \tag{4.7}$$

Quem muda é apenas a função de custo futuro, que agora é dada pela medida de risco ρ_{ω_2} aplicada aos custos de cada cenário (cuja definição também continua a mesma):

$$\begin{aligned} V_2(\hat{x}_1, \omega_2) = \min_{x_2 \geq 0} \quad & c_2^\top y_2 \\ \text{s.a.} \quad & A_2 x_2 + W_2 y_2 = d_2 - B_2 \hat{x}_1. \end{aligned} \tag{4.8}$$

$$V_2(\hat{x}_1) = \rho_{\omega_2}\left[V_2(\hat{x}_1, \omega_2)\right]. \tag{4.9}$$

Seja, então, $\hat{x}_1$ uma solução viável para o problema (4.7). Ao resolver o problema (4.8) para cada cenário $\omega_{2,i}$, obtemos os custos $\hat{v}_2(\omega_{2,i})$ e os multiplicadores de Lagrange $\hat{\pi}_2(\omega_{2,i})$. Em seguida, devemos determinar o CVAR dos custos $\hat{v}_2(\omega_{2,i})$. Ora, agora os custos são números reais (e não variáveis dentro de um problema de otimização), e portanto podemos simplesmente ordená-los e calcular a média dos α mais caros. Isto dá uma solução direta para o problema (4.2), pois os pesos serão exatamente $\frac{1}{\alpha}$ para os cenários mais caros, e 0 para todos os outros. Seja $\hat{\zeta}$ esta solução, correspondente aos custos associados a $\hat{x}_1$.

Como $\hat{\zeta}$ é uma solução viável para (4.2), independentemente dos valores de z_i, podemos substituir $z_i = V_2(x_1, \omega_{2,i})$ (ou seja, os custos de cada cenário para um outro estado x_1) e obter a seguinte desigualdade:

$$V_2(x_1) \geq \sum_{i=1}^{n} \hat{\zeta}_i p_i \cdot V_2(x_1, \omega_{2,i}) \quad \text{para todo } x_1. \tag{4.10}$$

Em seguida, aplicamos a desigualdade (2.7) para cada realização $\omega_{2,i}$ e obtemos:

$$\begin{aligned} V_2(x_1) &\geq \sum_{i=1}^{n} \hat{\zeta}_i p_i \cdot V_2(x_1, \omega_{2,i}) \\ &\geq \sum_{i=1}^{n} \hat{\zeta}_i p_i \cdot \Big(V_2(\hat{x}_1, \omega_{2,i}) + \pi_2(\omega_{2,i})^\top (x_1 - \hat{x}_1)\Big) \\ &= V_2(\hat{x}_1, \omega_2) + \left(\sum_{i=1}^{n} \hat{\zeta}_i p_i \cdot \pi_2(\omega_{2,i})\right)^\top (x_1 - \hat{x}_1). \end{aligned} \tag{4.11}$$

A última passagem é uma igualdade (e não uma desigualdade com o sinal contrário) porque, para os valores de $z_i = V_2(\hat{x}_1, \omega_{t,i})$, os pesos $\hat{\zeta}_i$ são exatamente os pesos ótimos do problema (4.2).

Visto de outra forma, o corte correspondente ao CVAR é obtido diretamente a partir da ponderação dos cortes de Benders com o mesmo peso que o cálculo do CVAR. O fato que este corte é *válido* (ou seja, está de fato abaixo da função de custo futuro V_2) para todos os outros estados

x_1, independente de a reponderação ser eventualmente diferente, é uma consequência da formulação (4.2) do CVAR, que garante que o CVAR está sempre *acima* de todas as reponderações viáveis.

Do que vimos, a fórmula (4.2) dá um procedimento para construir cortes para o CVAR. Se tivermos uma outra medida de risco ρ, a formulação terá restrições diferentes (correspondentes a um conjunto Q de reponderações), mas o espírito é o mesmo: basta calcular uma solução ótima $\hat{\zeta}$ para os custos correspondentes ao estado atual x. Naturalmente, quanto mais complexo o conjunto Q, mais complexo será o procedimento para calcular esta solução, possivelmente necessitando de fato a resolução de um problema de otimização (que não foi necessária no caso do CVAR, como vimos acima).

4.3 PDDE com aversão ao risco

Por questões de consistência da política de decisões, a forma mais simples de introduzir medidas de risco em problemas de otimização multi-estágio consiste em alterar, nas equações da formulação de programação dinâmica, a equação (2.2) que define a função de custo futuro:

$$V_t(x_{t-1}) = \rho_{\omega_t}\left[V_t(x_{t-1}, \omega_t)\right]. \tag{4.12}$$

Nela, ρ_{ω_t} substitui a esperança condicional com relação à variável aleatória ω_t, e é chamada de *medida de risco condicional*, pois se refere apenas à incerteza do estágio t. [2] A maior vantagem desta formulação é que ainda mantemos a interpretação das funções de custo futuro V_t, e basta adaptar a construção dos cortes médios a partir dos cortes de Benders para cada cenário. Quando for claro pelo contexto, o operador de Bellman *risco-avesso* correspondente continuará sendo denotado por $\mathcal{B}_t$.

Aqui é um bom momento para observar que o algoritmo de PDDE funciona como uma sequência de aplicações do algoritmo *L-shaped* para cada estágio: a fase *Forward* é análoga à resolução do problema de primeiro estágio, que calcula os estados x_t onde serão construídos os cortes para aprimorar a aproximação da função de custo futuro, e a fase *Backward* é análoga à resolução de todos os problemas para as realizações das incertezas do estágio seguinte, que calcula os multiplicadores de Lagrange π_t e os custos v_t que serão usados para construir os cortes ("médios" ou *multicut*, em função da escolha). Assim, para incorporar a aversão ao risco no algoritmo de PDDE, basta substituir a esperança condicional pela medida de risco ρ_{ω_t}, e adaptar o cálculo dos cortes de Benders para a medida de risco escolhida. Isso pode ser feito diretamente ao substituir a média dos coeficientes $\hat{v}_t$ e $\hat{\pi}_t$ (da linha 15 do algoritmo 2) pela sua reponderação de acordo com a medida de risco escolhida, utilizando as desigualdades (4.11) que são a generalização dos cortes para a aversão ao risco.

O algoritmo resultante é o seguinte:

[2] Em princípio, a medida de risco poderia ser diferente para cada estágio; por simplicidade, vamos considerar que será a mesma para não pesar a notação.

Algoritmo 3 Algoritmo de Programação Dinâmica Dual Estocástica (PDDE) com aversão ao risco

Require: Estado inicial x_0, número de iterações K, medida de risco ρ
Inicialize $t = 1$, $\underline{z} = -\infty$, $\hat{V}_t = 0$ para $t = 1, \dots, T$.
for $k = 1, \dots, K$ **do**
  Amostre o cenário $\omega_{[T]}$ a partir dos cenários de cada estágio
  $\hat{x}_0 = x_0$
  for $t = 1, \dots, T$ **do** ▷ Fase *Forward*
    Resolva o problema (3.6) do estágio t com incerteza ω_t para obter o estado final $\hat{x}_t$
  end for
  for $t = T, \dots, 2$ **do** ▷ Fase *Backward*
    for ω_t nos N_t cenários do estágio t **do**
      Resolva o problema (3.6) para obter $\hat{\pi}_t(\omega_t)$ e $\hat{v}_t(\omega_t)$
    end for
    Calcule $\hat{v}_t = \rho_{\omega_t}[\hat{v}_t(\omega_t)]$ ▷ Cálculo da medida de risco
    Deduza os pesos ζ_i tais que $\hat{v}_t = \sum_{i=1}^{n} \zeta_i p_i \cdot \hat{v}_t(\omega_{t,i})$
    Defina $\hat{\pi}_t := \sum_{i=1}^{n} \zeta_i p_i \cdot \hat{\pi}_t(\omega_{t,i})$
    Adicione o corte $\theta_t \geq \hat{v}_t + \hat{\pi}_t^{\top}(x_{t-1} - \hat{x}_{t-1})$ à função $\hat{V}_t$ do problema do estágio $t-1$.
  end for
end for

Parte II
Fundamentação metodológica

Nesta parte, veremos tanto algoritmos que produzem cotas superioes determinísticas, como variações do método de Monte Carlo que fornecem cotas superiores estatísticas, ou seja, com um intervalo de confiança associado. Todos os métodos determinísticos irão trabalhar com a construção de funções de custo futuro auxiliares, cujo objetivo final é a obtenção das cotas superiores: no próximo capítulo apresentamos os *métodos interiores*, e, no seguinte, os métodos derivados a partir das *conjugadas* das funções de custo futuro. A seguir, passamos aos métodos estatísticos, cujo objetivo é a avaliação rápida de uma política dada, utilizando simulação de Monte Carlo dos cenários para calcular uma estimativa estatística do custo (ou risco) de uma política. Também veremos dois métodos: o primeiro, utilizando uma modificação das funções de custo futuro, com uma variável suplementar; e o segundo, no último capítulo desta parte, utilizando ideias de *importance sampling*.

5 Métodos determinísticos por aproximação interior

Neste capítulo, veremos três métodos para a construção de cotas superiores determinísticas para problemas de otimização estocástica. O primeiro foi concebido como um método para avaliar a qualidade da solução da PDDE, após um certo número de iterações [PdMF13]. O segundo é uma modificação natural do algoritmo de PDDE, proposto em [FL23b] como uma versão puramente primal de um algoritmo primal-dual. Já o terceiro foi desenvolvido como parte de um novo algoritmo para solução de problemas de otimização estocástica multi-estágio, de forma a atender um critério de convergência [BDZ17]. Uma das novidades de [BDZ17] é que o algoritmo é *determinístico*, e não depende de amostragem de cenários, como é o caso da PDDE.

Os métodos de aproximação interior são, em vários aspectos, "duais" com respeito ao algoritmo de PDDE. Em primeiro lugar, ao invés de construir cotas inferiores para as funções de custo futuro

V_t, eles constroem cotas superiores. Em segundo lugar, eles não usam cortes para representar estas aproximações, mas sim uma interpolação construída a partir de um conjunto finito de pontos amostrados do espaço X de estados viáveis. No próximo capítulo, ao explorarmos as construções de funções conjugadas e o algoritmo de PDDE dual, veremos uma terceira razão de interpretar estes métodos como duais da PDDE.

Enfim, vale notar que estes métodos interiores são mais próximos dos métodos de otimização clássicos, que calculam uma cota superior a partir de uma solução viável.

Antes de apresentar os algoritmos multi-estágio, iremos ilustrar as ideias de aproximação interior no caso do algoritmo *L-shaped*, primeiro recapitulando sua construção das aproximações inferiores por cortes para uma função V, e em seguida mostrando como obter aproximações superiores desta mesma função, utilizando *vértices*.

5.1 O algoritmo *L-shaped* e aproximações inferiores por cortes

Para entender o mecanismo de aproximação interior, e contrastar com o algoritmo de PDDE, vamos primeiramente considerar um problema de otimização convexa simples, e comparar a aproximação interior com o algoritmo *L-shaped* da seção 2.3. Suponha que desejamos resolver:

$$v^* = \min_{x \in X} V(x), \tag{5.1}$$

onde $X \subset R^n$ é um conjunto compacto e convexo (representando o conjunto viável do "primeiro estágio"), e $V(x)$ é uma função convexa, α-Lipschitz em X (análoga à função de custo futuro de um problema de dois estágios). Supomos, como por exemplo no caso de um problema de dois estágios, que V não possui uma fórmula simples, mas que pode ser calculada (junto com sua derivada) para qualquer $x \in X$. O algoritmo *L-shaped* da seção 2.3 constrói uma cota inferior para v^*, a partir de aproximações lineares por partes $\underline{V}(x) \leq V(x)$, usando cortes.

De forma genérica, seja $S = \{x_1, \ldots, x_K\}$ um conjunto de K pontos em X. A função de aproximação inferior do método *L-shaped*, denotada a seguir por $\underline{V}^S$, construída a partir de cortes calculados nos pontos $x_i \in S$, é dada por:

$$\underline{V}^S(x) = \max_{1 \leq j \leq K} \; V(x_j) + \langle \nabla V(x_j), x - x_j \rangle. \tag{5.2}$$

Como V é α-Lipschitz, seus gradientes são limitados por α, e daí cada um dos cortes na fórmula acima também é α-Lipschitz, e portanto $\underline{V}^S$ também.

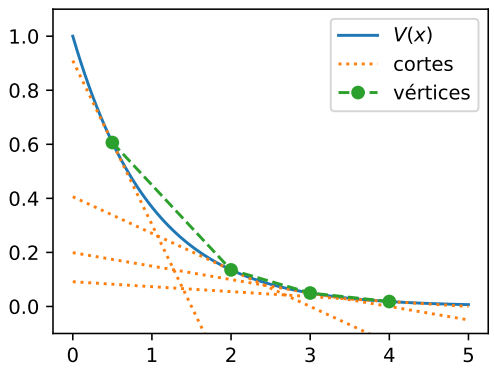
Nesta formulação mais geral, podemos re-enunciar o algoritmo de planos cortantes como a seguir:

Pela forma como o algoritmo *L-shaped* foi construído, para avaliar o critério de parada basta guardar o ponto x_i, com $1 \leq i \leq K$, para o qual a função $V(x_i)$ atingiu o menor valor: este será o candidato natural para a solução do problema (5.1). Entretanto, para a construção de cotas superiores (e, com elas, um critério de parada) no caso multi-estágio, será necessário manter, além das aproximações inferiores (por cortes) de cada função de custo futuro V_t, também aproximações superiores, como veremos a seguir.

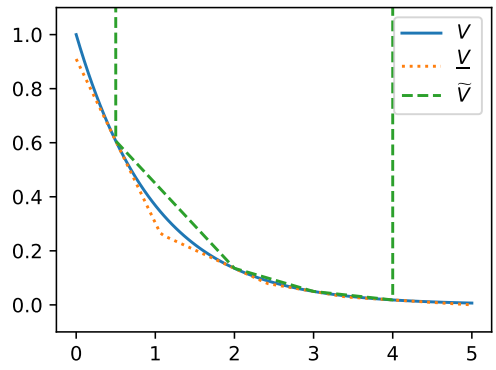
Algoritmo 4 Método de planos cortantes para uma função V

Require: Tolerância $\epsilon > 0$

1: Inicialize $S = \{x_1\}$, $k = 1$ e calcule $\underline{V}^S$.
2: **repeat**
3: Incremente k.
4: Resolva o problema aproximado $\min_{x \in X} \underline{V}^S(x)$.
5: Defina x_k como uma solução ótima, e a cota inferior $\underline{v}_k$ como o valor ótimo do problema.
6: Atualize $S = S \cup \{x_k\}$.
7: Atualize $\underline{V}^S$ com um novo corte para V (avaliando exatamente V e seu gradiente em x_k)
8: Atualize a cota superior $\overline{v}_k = \min(\overline{v}_{k-1}, V(x_k))$.
9: **until** $\overline{v}_k - \underline{v}_k < \epsilon$



(a) Cortes e vértices para V



(b) Gráfico de V, $\widetilde{V}^S$ e $\underline{V}^S$

Figura 5.1: Aproximações de V

5.2 Aproximações superiores para problemas de 2 estágios

A primeira função de aproximação superior que vamos usar é dada pelos pontos $x_i \in S$ e os valores correspondentes de V. Esta função será denotada por $\widetilde{V}^S$ e é dada por:

$$\begin{aligned} \widetilde{V}^S(u) = \quad & \min_{\sigma_j} \quad \sum_{j=1}^K \sigma_j V(x_j) \\ & \text{s.a} \quad \sum_{j=1}^K \sigma_j x_j = u \\ & \qquad \sum_{j=1}^K \sigma_j = 1 \\ & \qquad \sigma_j \geq 0 \end{aligned} \tag{5.3}$$

Esta função é convexa em x, pois é dada um problema de otimização linear, e x aparece apenas no lado direito das restrições. Mais ainda, observe que esta função é a *maior* função convexa que passa pelos pontos $(x_i, V(x_i))$, e portanto é uma cota superior para V. De fato, o gráfico de $\widetilde{V}^S$ é a envoltória convexa dos pontos $(x_i, V(x_i))$ do gráfico de V em $\mathbb{R}^d \times \mathbb{R}$. Ilustramos esta função, junto com as funções de aproximação externa, na Figura 5.1b para uma função V de uma variável, com $K = 4$ pontos amostrados.

Uma diferença entre as aproximações é que $\widetilde{V}^S$ não é α-Lipschitz no conjunto viável X. Isso ocorre porque ela não está definida em todos os pontos de X, como indicado no exemplo da Figura 5.1b através das linhas verticais. Por esta mesma razão, $\widetilde{V}^S$ não pode ser usada diretamente para determinar uma *política* viável, pois poderia acontecer que não fosse possível encontrar um

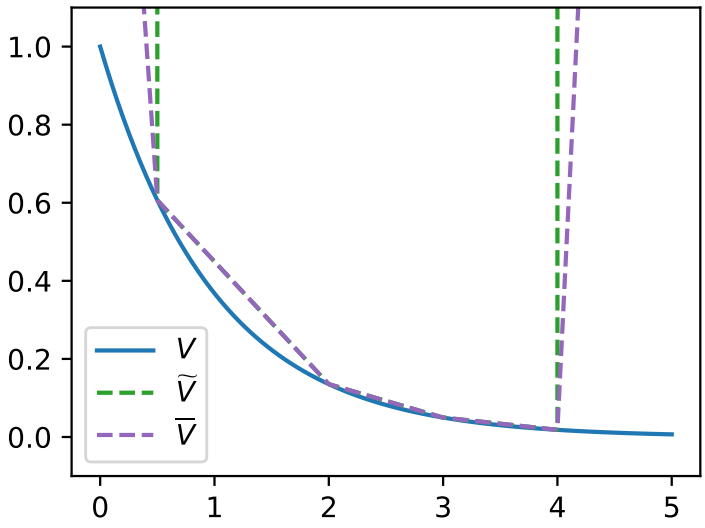


Figura 5.2: Comparação entre aproximações para V: $\overline{V}$, Lipschitz e $\widetilde{V}$, definida apenas entre 0.5 e 4.

ponto x onde $\widetilde{V}^S(x)$ fosse finito.

Para contornar este problema, vamos considerar uma função de aproximação superior que seja α-Lipschitz, e que seja uma cota superior para V. A forma mais natural para fazer isso é construir uma *relaxação Lipschitz* da função $\widetilde{V}^S$. Esta pode ser obtida ao introduzir uma variável de folga ξ no problema (5.3), com uma penalização de magnitude α na função objetivo:

$$\begin{aligned} \overline{V}^S(u) = \quad & \min_{\sigma_j, \xi_i} \quad \sum_{j=1}^K \sigma_j V(x_j) + \alpha \sum_{i=1}^n |\xi_i| \\ & \text{s.a} \quad \sum_{j=1}^K \sigma_j x_j = u - \xi \\ & \qquad \sum_{j=1}^K \sigma_j = 1 \\ & \qquad \sigma_j \geq 0. \end{aligned} \tag{5.4}$$

Esta função é α-Lipschitz por construção, e, como V também é α-Lipschitz (por hipótese), $\overline{V}^S$ também será uma cota superior para V. Para garantir a validade de $\overline{V}$ como uma aproximação por cima de V, é necessário que a penalidade α utilizada seja uma cota superior para a constante de Lipschitz de V. Vale notar que, na prática, a constante de Lipschitz *exata* para V não é conhecida, mas em muitos casos é razoável supor que uma cota superior para α é conhecida.

Vemos, na figura 5.2, uma ilustração do efeito da regularização de $\widetilde{V}^S$ para obter $\overline{V}^S$: a vertical que aparece no gráfico de $\widetilde{V}^S$ é suavizada, e a função resultante é Lipschitz contínua, com uma inclinação constante em cada lado da figura. Também podemos notar que a distância entre $\overline{V}^S$ e V é muito maior em uma das direções, em que a regularização é muito mais forte do que necessário.

Com isto, chegamos a uma versão modificada do algoritmo 4, que agora constrói uma cota superior para v^*, que indicamos no algoritmo 5. Neste ponto, ainda não é óbvia a utilidade das funções interiores $\widetilde{V}$ e $\overline{V}$ — nem a necessidade que $\overline{V}$ esteja definida para todos os pontos de X. Isto aparecerá de forma mais clara quando considerarmos o problema multi-estágio.

Algoritmo 5 Método *L-shaped* com aproximação interior

1: Inicialize $S = \{x_1\}$, $k = 1$ e calcule $\underline{V}^S$ e $\overline{V}^S$.
2: **repeat**
3: Incremente k.
4: Resolva o problema aproximado $\min_{x \in X} \underline{V}^S(x)$.
5: Defina x_k como uma solução ótima, e a cota inferior $\underline{v}_k$ como o valor ótimo do problema.
6: Atualize $S = S \cup \{x_k\}$.
7: Atualize $\underline{V}^S$ (com um novo corte) e $\overline{V}^S$ (com um novo vértice) de acordo com (5.2) e (5.4)
8: Atualize a cota superior $\overline{v}_k = \min(\overline{v}_{k-1}, \overline{V}^S(x_k))$.
9: **until** $\overline{v}_k - \underline{v}_k < \epsilon$

5.3 Programação dinâmica para cotas superiores

Agora, vamos considerar o problema multi-estágio (2.1) e a construção de cotas superiores para a função de custo futuro V_t. Consideramos o caso de um problema de dois estágios, com $t = 1$ e $t = 2$. A diferença principal para o caso anterior, e que permitirá passar para o caso multi-estágio de forma direta, é que *não supomos* que a função de custo futuro V_3 seja conhecida exatamente.

Assim, e de forma análoga ao algoritmo de PDDE, vamos considerar que possuímos apenas aproximações da função de custo futuro V_3, que denotaremos $\underline{V}_3$ e $\overline{V}_3$. Dado um estado final x_1 para o primeiro estágio, podemos resolver os problemas de otimização aproximados do segundo estágio, tanto por baixo quanto por cima, para obter as funções $\underline{V}_2$ e $\overline{V}_2$:

$$\begin{aligned} \underline{V}_2(x_1, \omega_2) = \min_{x_2, y_2 \geq 0} \quad & c_2^\top y_2 + \underline{V}_3(x_2) \\ \text{s.a} \quad & A_2 x_2 + W_2 y_2 = d_2 - B_2 x_1; \end{aligned} \tag{5.5}$$

$$\begin{aligned} \overline{V}_2(x_1, \omega_2) = \min_{x_2, y_2 \geq 0} \quad & c_2^\top y_2 + \overline{V}_3(x_2) \\ \text{s.a} \quad & A_2 x_2 + W_2 y_2 = d_2 - B_2 x_1. \end{aligned} \tag{5.6}$$

Para obter as funções de custo futuro, basta calcular a medida de risco ρ para as funções $\underline{V}_2$ e $\overline{V}_2$. Naturalmente, não serão as funções exatas que iremos utilizar, mas, respectivamente, cortes e vértices para $\underline{V}_2$ e $\overline{V}_2$. No caso de $\overline{V}_2$, basta o valor da função, e portanto o procedimento é direto. Para $\underline{V}_2$, entretanto, é necessário calcular o gradiente da função, e portanto é necessário realizar a ponderação dos gradientes dos cortes de cada cenário amostrado pelo mesmo fator que foi utilizado para calcular o risco da função de custo futuro. De fato, o raciocínio é análogo ao aplicado nas desigualdades da equação (4.11), que partem de desigualdades válidas para os custos de cada realização da incerteza $\omega_{2,i}$.

A observação fundamental nesta etapa é a seguinte: se as funções $\underline{V}_3$ e $\overline{V}_3$ forem cota inferior e cota superior para a função de custo futuro V_3, então as funções $\underline{V}_2$ e $\overline{V}_2$ são, respectivamente, cota inferior e cota superior para a função de custo futuro V_2. Mais ainda, ao construir, respectivamente, aproximações inferiores e superiores para $\underline{V}_2$ e $\overline{V}_2$ (por cortes e vértices), obtemos, respectivamente, aproximações inferiores e superiores para a função de custo futuro V_2.

Como no caso do algoritmo *L-shaped*, a função $\underline{V}_2(x_1)$ é representada no problema de primeiro estágio como uma variável θ_1 que está acima de todos os cortes construídos para ela.

$$\begin{aligned} \min_{x_1, y_1 \geq 0} \quad & c_1^\top y_1 + \theta_1 \\ \text{s.a.} \quad & A_1 x_1 + B_1 x_0 + W_1 y_1 = d_1, \\ & \theta_1 \geq \theta_{\min} \\ & \theta_1 \geq \hat{v}_{(k)} + \hat{pi}_{(k)}^\top (x_1 - x_{(k)}), \end{aligned} \tag{5.7}$$

Já no caso da função $\overline{V}_2$, ela é representada como uma combinação convexa dos vértices construídos para ela, e é dada pelo problema de minimização (5.4). Note que este problema inclui as variáveis de folga ξ, penalizadas pelo fator L_t. Este termo garante que a função $\overline{V}_t$ seja no máximo L_t Lipschitz, e portanto esteja definida para todos os pontos do domínio, e não apenas as combinações convexas dos pontos amostrados x_t^k. Para incorporar $\overline{V}_t$ ao problema do estágio anterior, como este também é um problema de minimização, basta adicionar as variáveis σ e ξ, com suas restrições, e somar as funções objetivo. Por exemplo, no caso do problema de primeiro estágio que estamos considerando, teríamos:

$$\begin{aligned} \min_{x_1,y_1,\sigma_k,\xi} \quad & c_1^\top y_1 + \textstyle\sum_{k=1}^K \sigma_k V_2(x_{1,(k)}) + L_2 \left|\xi\right| \\ \text{s.a} \quad & A_1 x_1 + W_2 y_2 = d_1 - B_1 x_0 - \xi \\ & \textstyle\sum_{k=1}^K \sigma_k x_{1,(k)} = x_1 - \xi \\ & \textstyle\sum_{k=1}^K \sigma_k = 1 \\ & \sigma_k \geq 0. \end{aligned} \tag{5.8}$$

Este é o ponto de partida para a construção de um algoritmo de aproximação simultânea exterior e interior para o problema multi-estágio, o que envolve construir aproximações inferiores e superiores para as funções de custo futuro de cada estágio, de forma recursiva. Resta, então, determinar em quais pontos serão calculadas as aproximações inferiores e superiores para as funções de custo futuro. Há várias possibilidades para tal.

5.4 Uma etapa *backward* final

A maneira mais simples de calcular as aproximações inferiores e superiores, e que implica na menor mudança em relação ao algoritmo de PDDE, é a de calcular as aproximações em um conjunto de pontos amostrados do espaço de decisão, como proposto por [PdMF13]. De fato, este método consiste em utilizar o algoritmo de PDDE usual, e salvar os estados amostrados durante as etapas *forward*. A partir daí, calculamos as aproximações interiores (as exteriores já foram calculadas) para as funções de custo futuro de forma recursiva, a partir do último estágio, utilizando os pontos amostrados de cada estágio para calcular as aproximações $\overline{V}_t$ analogamente ao que foi feito para $\overline{V}$ na equação (5.4). Isto resulta no algoritmo 6.

A desvantagem deste método é que ele não pode ser facilmente utilizado como um critério de parada para o algoritmo de PDDE usual, pois o cálculo das cotas superiores se dá *a posteriori*. Naturalmente, se o *gap* entre as cotas superior e inferior não for pequeno o suficiente, é possível continuar o algoritmo de PDDE por mais iterações (novamente, sem um critério claro de parada) e repetir o procedimento de cálculo das cotas superiores.

5.5 Uso da etapa *backward* combinada

A primeira modificação natural que o algoritmo de aproximação interior apresentado acima sugere é a de utilizar os pontos amostrados para calcular as aproximações inferiores e superiores para as funções de custo futuro, de forma simultânea. Isto resulta num algoritmo combinado, que calcula as trajetórias na etapa *forward* e as cotas inferiores e superiores na etapa *backward*, simultaneamente. Este algoritmo foi proposto em [FL23b] com o nome de *Stochastic Inner Dynamic Programming* (abreviado por SIDP). Apresentamos no algoritmo 7 uma versão adaptada à terminologia deste relatório, buscando manter o máximo de similaridade com os algoritmos anteriormente descritos. A principal vantagem deste algoritmo é que ele possui um critério de parada claro, como podemos ver na linha 23, baseado no *gap* entre as cotas inferior e superior, calculadas a cada iteração.

Podemos ter uma ideia de como este algoritmo procede através da Figura 5.3.

Algoritmo 6 Cotas superiores para a função de custo futuro

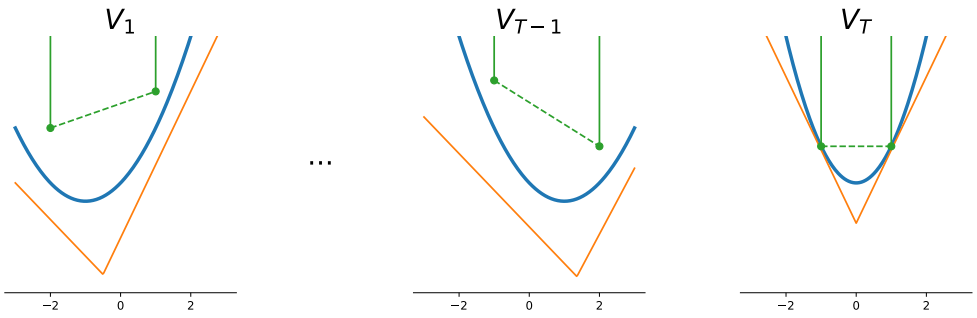
Require: Constantes de Lipschitz L_t para $t = 1, \dots, T$

Aplique o algoritmo de PDDE 2 para obter as aproximações $\underline{V}_t$, e salve os estados amostrados x_t^k para cada estágio t e iteração k
Defina $\overline{V}_{T+1} = V_{T+1}$, conhecida
for $t = T, \dots, 2$ **do**
    for $k = 1, \dots, K$ **do**
        for $\omega_{t,i}$ nas N_t realizações da incerteza do estágio t **do**
            Resolva o problema (3.6) com a função de custo futuro $\overline{V}_{t+1}$ para obter $\overline{V}_t(x_{t-1}^k, \omega_t)$
        end for
        Avalie a medida de risco ρ para calcular $\overline{V}_t(x_{t-1}^k)$
    end for
    Construa a função $\overline{V}_t$ a partir dos pontos amostrados e da constante de Lipschitz L_t, usando a equação (5.4)
end for
Resolva o problema de primeiro estágio com a função $\overline{V}_2$ para obter uma cota superior para o valor ótimo

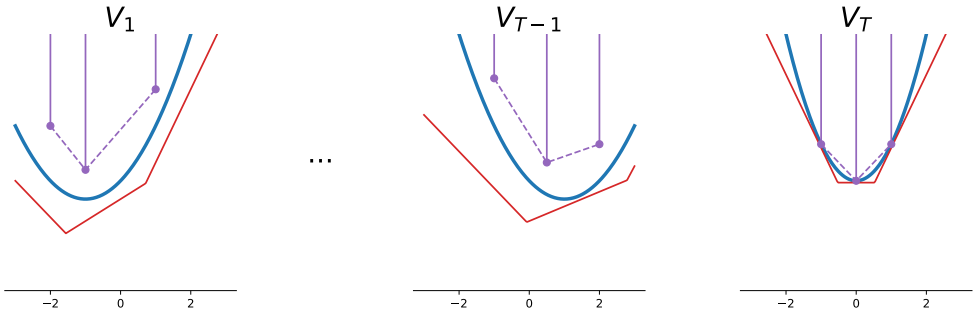
Algoritmo 7 Algoritmo combinando cotas superiores e inferiores

Require: Estado inicial x_0, constantes de Lipschitz L_t para $t = 1, \dots, T$, tolerância ϵ.

Inicialize $t = 1$, $\underline{z} = -\infty$, $\underline{V}_t = 0$ para $t = 1, \dots, T$.
for $k = 1, \dots$ **do**
    Amostre o cenário $\omega_{[T]}$ a partir das realizações das incertezas de cada estágio
    $\hat{x}_0 = x_0$
    for $t = 1, \dots, T$ **do** ▷ Fase *Forward*
        Resolva o problema (3.6) do estágio t com incerteza ω_t para obter o estado final $\hat{x}_t$
    end for
    for $t = T, \dots, 2$ **do**
        for $\omega_{t,i}$ nas N_t realizações da incerteza do estágio t **do** ▷ Fase *Backward*, cortes
            Resolva o problema (3.6) com $\underline{V}_{t+1}$ para obter $\hat{\pi}_t(\omega_t)$ e $\hat{v}_t(\omega_t)$
        end for
        Calcule a medida de risco ρ sobre os valores $\hat{v}_t(\omega_t)$ para obter $\hat{v}_t$ ▷ Cálculo do corte
        Deduza $\hat{\pi}_t$ a partir da mesma ponderação aplicada aos multiplicadores $\hat{\pi}(\omega_t)$
        Adicione o corte $\theta_t \geq \hat{v}_t + \hat{\pi}_t^\top (x_{t-1} - \hat{x}_{t-1})$ à função $\underline{V}_t$ do problema do estágio $t - 1$.
        for $\omega_{t,i}$ nas N_t realizações da incerteza do estágio t **do** ▷ Fase *Backward*, vértices
            Resolva o problema (3.6) com $\overline{V}_{t+1}$ para obter $\hat{v}_t(\omega_t)$
        end for
        Calcule a medida de risco ρ sobre os valores $\hat{v}_t(\omega_t)$ para obter $\hat{v}_t$ ▷ Cálculo do vértice
        Adicione o vértice $(x_{t-1}, \hat{v}_t)$ à função $\overline{V}_t$ do problema do estágio $t - 1$.
    end for
    Resolva o problema de primeiro estágio com $\underline{V}_2$ para obter uma cota inferior z_{inf}^k
    Resolva o problema de primeiro estágio com $\overline{V}_2$ para obter uma cota superior z_{sup}^k
    if $z_{\text{sup}}^k - z_{\text{inf}}^k < \epsilon$ **then**
        break
    end if
end for



(a) Após 2 iterações



(b) Após adicionar o terceiro corte e terceiro vértice

Figura 5.3: Exemplo de execução do algoritmo *Stochastic Inner Dynamic Programming*

5.6 Etapa *forward* determinística

A segunda possibilidade para a escolha dos pontos onde calcular as funções de custo futuro é a de utilizar um método determinístico, como proposto por [BDZ17]. Há algumas diferenças entre este método e a PDDE usual, principalmente devido ao fato que ele não depende de amostragem de cenários. Para tal, este algoritmo irá calcular aproximações inferiores e superiores para as funções de custo futuro para cada estágio, ou seja, será um algoritmo *multi-cut*. A razão desta escolha é que, para cada realização da incerteza, também é possível calcular o *gap* da aproximação entre a cota superior e a cota inferior, e, se este *gap* for pequeno o suficiente, o algoritmo pode ser interrompido. Exatamente por isto, o algoritmo irá *escolher a realização que mais contribui para o gap* para construir o cenário da etapa *forward*, e melhorar as aproximações para as funções de custo futuro correspondentes a este cenário durante a etapa *backward*. Assim, este algoritmo não depende de uma amostragem de cenários, e é determinístico.

Por outro lado, este algoritmo requer uma modificação mais significativa do algoritmo de PDDE usual, para incorporar um novo método para a trajetória *forward*, além da manutenção das aproximações inferiores e superiores para as funções de custo futuro de cada estágio. Vale notar que, ao calcular um corte apenas para um dos cenários amostrados, numa formulação *multi-cut*, o algoritmo determinístico de [BDZ17] resulta em problemas de otimização com o mesmo número de cortes que o algoritmo de PDDE usual a cada iteração. A diferença é que o algoritmo de PDDE irá aproximar a função de custo futuro em média (ou para a medida de risco escolhida), enquanto o algoritmo determinístico irá distribuir os cortes (e vértices, para a aproximação interior) entre os cenários de forma a minimizar o *gap*.

Da forma como descrito em [BDZ17], o algoritmo de programação dinâmica também considera

uma formulação do tipo *decisão-acaso*, em vez da formulação *acaso-decisão*, mais usual para a aplicação da PDDE em problemas multi-estágio.[3] A escolha de uma formulação *decisão-acaso* pode ser explicada pela dedução do algoritmo estocástico a partir de um exemplo de dois estágios, no qual a função de custo futuro é escrita a partir da decisão de primeiro estágio, que é tomada antes da incerteza ser revelada. Isto resulta em uma segunda diferença na formulação dos problemas de cada estágio. Para evitar esta complexidade suplementar, vamos considerar a formulação *acaso-decisão* usual, e adaptar o algoritmo de programação dinâmica determinística para esta formulação. Assim, no estágio t, o algoritmo de programação dinâmica irá resolver o problema

$$\begin{aligned} V_t(x_{t-1}, \omega_t) = \min_{x_t, y_t \geq 0} \quad & c_t^\top y_t + \rho[\underline{V}_{t+1}(x_t, \omega_{t+1})] \\ \text{s.a} \quad & A_t x_t + W_2 y_2 = d_t - W_t x_{t-1}. \end{aligned} \tag{5.9}$$

Na prática, o algoritmo é *multi-cut*, e portanto inclui N_t representações da função de custo futuro, uma para cada realização da incerteza do estágio seguinte. Além disto, ele também necessita a representação da medida de risco ρ na função objetivo. Para o caso em que ρ é uma combinação entre a média e o CVAR, o problema fica:

$$\begin{aligned} V_t(x_{t-1}, \omega_t) = \min_{x_t, y_t \geq 0} \quad & c_t^\top y_t + (1-\beta) \textstyle\sum_{i=1}^{N_{t+1}} p_i \theta_i + \frac{\beta}{\alpha}[q + \sum_{i=1}^{N_{t+1}} p_i u_i] \\ \text{s.a} \quad & A_t x_t + W_2 y_2 = d_t - B_t x_{t-1} \\ & \theta_i \geq v_i^k + (\pi_i^k)^\top (x_t - x_t^k) \quad \forall i, k \\ & q + u_i \geq \theta_i \quad \forall i. \end{aligned} \tag{5.10}$$

onde p_i são as probabilidades associadas a cada realização $\omega_{t,i}$, e θ_i a estimativa do respectivo custo futuro pelos cortes indexados por $k = 1, \ldots$. A representação do CVAR é dada pelas variáveis auxiliares q (α-quantil superior) e u_i.

Naturalmente, no caso em que necessitamos calcular as cotas superiores, a função objetivo é a mesma, bastando apenas substituir os cortes para as variáveis θ_i pelas respectivas desigualdades através da representação de vértices v_i^k.

Com isto, apresentamos o Algoritmo 8, que é um algoritmo determinístico com aproximação interior para a solução de problemas de otimização estocástica multi-estágio. Ele também conhecido como *problem-child*, porque os cenários são escolhidos em cada nó da árvore de cenários de forma a escolher a abertura ("descendente") que apresenta o maior *gap* entre as cotas inferior e superior ("problema" para a convergência).

Note que a etapa *backward* do algoritmo não percorre todas as realizações da incerteza a cada estágio, mas apenas aquela que mais contribui para o *gap* entre as cotas inferior e superior. Desta forma, apenas um corte (ou vértice) é adicionado ao problema do estágio anterior.

6 Funções conjugadas e PDDE para o problema dual

Nesta seção, introduzimos as funções conjugadas V_t^* e $V_t^{\boxtimes}$, e como elas estão relacionadas com o problema dual do problema multi-estágio (2.1). Em particular, veremos que elas podem ser interpretadas como funções de custo futuro no problema dual. Em seguida, iremos descrever como o algoritmo de PDDE pode ser adaptado para resolver o problema dual, calculando aproximações

[3] Vale notar que a formulação *decisão-acaso* é, muitas vezes, mais próxima do processo de programação e planejamento, pois em geral não sabemos de antemão a realização da incerteza ao longo do estágio considerado para a operação. A formulação *acaso-decisão*, por outro lado, torna mais simples a solução dos problemas de cada estágio, pois basta resolver o problema de otimização para cada cenário amostrado, e depois calcular a média ou a medida de risco destas soluções.

Algoritmo 8 Algoritmo determinístico para aproximação interior, via *problem-child*

Require: Estado inicial x_0, constantes de Lipschitz L_t para $t = 1, \dots, T$, tolerância ϵ.
1: $\underline{V}_t = 0$ e $\overline{V}_t = +\infty$ para $t = 2, \dots, T$.
2: **for** $k = 1, \dots$ **do**
3: $\quad \hat{x}_0 = x_0$
4: $\quad$ **for** $t = 1, \dots, T$ **do** ▷ Fase *Forward*
5: $\quad\quad$ Resolva o problema (5.9) do estágio t para obter o estado final $\hat{x}_t$
6: $\quad\quad$ Calcule o *gap* entre as estimativas em cada realização, $\overline{V}_t(\hat{x}_t, \omega_{t+1}) - \underline{V}_t(\hat{x}_t, \omega_{t+1})$.
7: $\quad\quad$ Escolha a realização $\omega_{t+1,i}$ da incerteza do estágio t que mais contribui para o *gap*.
8: $\quad$ **end for**
9: $\quad$ **for** $t = T, \dots, 2$ **do**
10: $\quad\quad$ Resolva o problema (5.9) com $\underline{V}_{t+1}$ para obter $\hat{\pi}_t(\omega_t)$ e $\hat{v}_t(\omega_t)$ ▷ Fase *Backward*, cortes
11: $\quad\quad$ Adicione o corte $\theta_i \geq \hat{v}_t + \hat{\pi}_t^\top (x_{t-1} - \hat{x}_{t-1})$ à função $\underline{V}_{t,i}$ da realização $\omega_{t+1,i}$.
12: $\quad\quad$ Resolva o problema (5.9) com $\overline{V}_{t+1}$ para obter $\hat{v}_t(\omega_t)$ ▷ Fase *Backward*, vértices
13: $\quad\quad$ Adicione o vértice $(x_{t-1}, \hat{v}_t)$ à função $\overline{V}_{t,i}$ da realização $\omega_{t+1,i}$.
14: $\quad$ **end for**
15: $\quad$ Resolva o problema de primeiro estágio com $\underline{V}_2$ para obter uma cota inferior z^k_{inf}
16: $\quad$ Resolva o problema de primeiro estágio com $\overline{V}_2$ para obter uma cota superior z^k_{sup}
17: $\quad$ **if** $z^k_{\text{sup}} - z^k_{\text{inf}} < \epsilon$ **then**
18: $\quad\quad$ **break**
19: $\quad$ **end if**
20: **end for**

destas funções conjugadas. Enfim, veremos como as funções conjugadas V_t^* podem ser utilizadas para construir cotas superiores para as funções de custo futuro V_t do problema original.

Vale notar que, até onde sabemos, esta é a única forma de obter cotas superiores, convergentes para o valor ótimo do problema multi-estágio (2.1), que não necessita utilizar o algoritmo de PDDE primal como um auxiliar.

6.1 Funções conjugadas e recorrência dual

Começamos com a definição da função conjugada de Fenchel de uma função $f : \mathbb{R}^n \to \mathbb{R}$.

Definição 6.1 (Conjugada de Fenchel)**.** *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função. A conjugada de Fenchel de f é a função dada por*

$$f^*(\pi) = \sup_{x \in \mathbb{R}^n} \left\{ \pi^\top x - f(x) \right\}. \tag{6.1}$$

A conjugada de Fenchel é sempre uma função convexa de π, pois é o máximo de funções afins. A principal propriedade que vamos utilizar das funções conjugadas é que, se f for uma função convexa, própria e semi-contínua inferiormente, então $(f^*)^* = f$.[4] Este é um resultado análogo à *dualidade forte*, e em particular vale para funções convexas lineares por partes. Esse resultado tem, também, uma interpretação muito simples: a fórmula $f(x) = \sup_\pi \left\{ \pi^\top x - f^*(\pi) \right\}$ implica que $f(x)$ é dada pelo máximo de funções afins, com inclinação π, que será um (sub-)gradiente de

[4] Uma função é *própria* se nunca toma o valor $-\infty$, e é *semi-contínua inferiormente* se os conjuntos de nível $\{x \mid f(x) \leq \alpha\}$ são fechados para todo $\alpha \in \mathbb{R}$.

f. Em particular, se f for Lipschitz, os gradientes serão limitados; um cálculo simples mostra que $f^*(\pi)$ será finito apenas quando π for menor do que a constante de Lipschitz de f.

Da mesma forma que as equações (2.2) e (2.3) definem uma recorrência entre as funções de custo futuro V_t e V_{t+1}, é possível mostrar que as funções conjugadas V_t^* e V_{t+1}^* estão relacionadas por uma recorrência. Mais importante, é possível obter uma formulação desta recorrência como um problema de otimização, derivado a partir do problema de otimização do estágio t. A forma mais natural de obter esta recorrência passa por aplicar a definição de conjugada de Fenchel à função de custo futuro V_t, expandir a expressão de V_t como um problema de otimização e usar dualidade forte para trocar a ordem entre minimização e maximização.

Para não carregar excessivamente a notação, iremos utilizar a seguinte fórmula para o problema do t-ésimo estágio, que dá a recorrência entre V_t e V_{t+1}:

$$\begin{aligned} V_t(x) = \min_{y,z\geq 0} \quad & \mathbb{E}\left[c_t^\top y + V_{t+1}(z)\right] \\ \text{s.a} \quad & Az + Wy = d - Bx, \end{aligned} \tag{6.2}$$

onde substituímos o estado inicial x_{t-1} por x e o estado final por z, e omitimos a dependência de todas as variáveis em t. Assim:

$$V_t^*(\pi) = \sup_{x\in\mathbb{R}^n} \left\{ \pi^\top x - \min_{\substack{y,z\geq 0 \\ Az+Wy=d-Bx}} \mathbb{E}\left[c^\top y + V_{t+1}(z)\right] \right\} \qquad \text{pela definição de } V_t \tag{6.3}$$

$$= \sup_{x\in\mathbb{R}^n} \max_{y,z\geq 0} \quad \pi^\top x - \mathbb{E}\left[c^\top y + V_{t+1}(z)\right] \qquad \text{trocando } -\min \text{ por } \max - \tag{6.4}$$
$$\text{s.a} \quad Az + Wy = d - Bx$$

$$= \sup_{x\in\mathbb{R}^n} \max_{y,z\geq 0} \quad \pi^\top x - \mathbb{E}\left[c^\top y + V_{t+1}(z)\right] \qquad \text{dualizando a restrição} \tag{6.5}$$
$$+ \inf_\lambda \mathbb{E}[\lambda^\top (d - Bx - Az - Wy)]$$

$$= \inf_\lambda \sup_{x\in\mathbb{R}^n} \max_{y,z\geq 0} \quad \pi^\top x - \mathbb{E}\left[c^\top y + V_{t+1}(z)\right] \qquad \text{por dualidade forte} \tag{6.6}$$
$$+ \mathbb{E}[\lambda^\top (d - Bx - Az - Wy)]$$

$$\begin{aligned} = \inf_{\lambda,\nu} \quad & \mathbb{E}\left[\lambda^\top d + V_{t+1}^*(\nu)\right] \\ \text{s.a.} \quad & \mathbb{E}[B^\top \lambda] = \pi, \\ & c + W^\top \lambda \geq 0, \\ & \nu + A^\top \lambda \geq 0. \end{aligned} \tag{6.7}$$

onde simplificamos os supremos usando que:

$$\sup_x \pi^\top x - \mathbb{E}[\lambda^\top B x] = \begin{cases} 0 & \text{se } \mathbb{E}[B^\top \lambda] = \pi \\ +\infty & \text{caso contrário} \end{cases}$$

$$\sup_{y\geq 0} \mathbb{E}\left[-c^\top y - \lambda^\top W y\right] = \begin{cases} 0 & \text{se } c + W^\top \lambda \geq 0 \\ +\infty & \text{caso contrário} \end{cases}$$

$$\begin{aligned} \sup_{z\geq 0} \mathbb{E}\left[-V_{t+1}(z) - \lambda^\top A z\right] &= \sup_{z\geq 0} \mathbb{E}\left[-V_{t+1}(z) - \lambda^\top A z\right] + \inf_{\mu\geq 0} \mathbb{E}[\mu^\top z] \\ &= \inf_{\mu\geq 0} \sup_{z\geq 0} \mathbb{E}\left[(\mu - A^\top \lambda)^\top z - V_{t+1}(z)\right] \\ &= \inf_{\mu\geq 0} V_{t+1}^*(\mu - A^\top \lambda) \\ &= \inf_{\nu + A^\top \lambda \geq 0} V_{t+1}^*(\nu). \end{aligned} \tag{6.8}$$

Note que a recorrência da equação (6.7) é um problema de otimização linear, porém com uma restrição *em média*, que liga todos os cenários com a variável de estado inicial π. Esta restrição tem duas particularidades. Em primeiro lugar, ela não permite que os problemas de cada cenário sejam resolvidos de forma independente, pois as variáveis de decisão λ e ν devem ser compatíveis com uma restrição global. Isto torna o problema dual mais complexo que o problema primal, que pode ser resolvido independentemente para cada cenário na formulação primal clássica de programação dinâmica estocástica.

Em segundo lugar, a restrição em média poderia não ser viável para alguns valores de π. Isto traz dificuldades para a etapa *forward* do algoritmo de PDDE, que precisa encontrar um valor para a variável de estado final (neste caso, o valor de ν) para prosseguir para a próxima etapa. Na prática, esta segunda observação não é tão crítica, pois em geral a variável de estado primal z também possui um limite superior, o que dá uma desigualdade suplementar e um multiplicador de Lagrange associado. Tal multiplicador funciona como uma variável de folga, penalizada pela cota superior da variável de estado na função objetivo.

Entretanto, esta mesma razão nos conduz a uma dificuldade prática para a implementação do algoritmo de PDDE para as funções conjugadas V_t^*: também será necessário limitar o domínio da variável de estado dual π para garantir a convergência do algoritmo. A forma mais simples de fazê-lo é notar que π corresponde aos preços associados à variável de estado x_t (gradientes de V_t), e portanto pode ser limitado pela constante de Lipschitz da função de custo futuro V_t. Assim, tanto o método dual como os métodos de aproximação interior necessitam da mesma informação suplementar para serem implementados, a saber, uma constante de Lipschitz para cada função de custo futuro V_t.

6.2 Função conjugada com aversão ao risco

Se, em vez de a recorrência entre as funções de custo futuro ser dada em média pela equação (2.2), mas por uma medida de risco, como em (4.12), as operações que fizemos para deduzir a recorrência da equação (6.7) não são mais válidas. Uma forma de contornar esta dificuldade, proposta em [FL23a] é observar que uma medida de risco implica numa mudança das distribuições de probabilidade dos cenários. Assim, a função de custo futuro, para cada cenário, passa a depender de uma nova variável, que é o fator de risco associado.

De forma mais precisa, se a medida de risco ρ é dada por uma fórmula como (4.2), iremos substituir o termo $\mathbb{E}\left[c_t^\top z + V_{t+1}(z)\right]$ por

$$\rho\left(c_t^\top y + V_{t+1}(z)\right) = \max_{\zeta \in Q} \sum_i \zeta_i p_i \left(c_t^\top y_i + V_{t+1}(z_i)\right), \tag{6.9}$$

onde indicamos por i os cenários, p_i suas probabilidades, z_i os valores associados à variável de estado final, ζ o vetor de pesos associado a cada cenário e Q é o conjunto de reponderações associado à medida de risco ρ.

Desta forma, antes de eliminar a variável z para a construção da função conjugada V_{t+1}^*, na equação (6.8), teremos agora a expressão

$$\max_z \sum_i p_i \left[\nu_i^\top z_i - \lambda_i^\top A_i z_i - \zeta_i V_{t+1}(z_i)\right].$$

Aplicando diretamente a definição da função conjugada, obteríamos em cada cenário a conjugada

da função $\zeta_i V_{t+1}$, avaliada no ponto $\nu_i - A_i^\top \lambda_i$. Ora, se $\zeta \neq 0$, temos:

$$\begin{aligned}(\zeta V_{t+1})^*(\pi) &= \sup_{z\in\mathbb{R}^n} \left\{\pi^\top z - \zeta V_{t+1}(z)\right\} \\ &= \zeta \sup_{z\in\mathbb{R}^n} \left\{(\pi/\zeta)^\top z - V_{t+1}(z)\right\} = \zeta V_{t+1}^*(\pi/\zeta). \end{aligned} \tag{6.10}$$

A expressão acima é a combinação da conjugada (que leva V em V^*) com uma transformação conhecida como *perspectiva*:

Definição 6.2. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função e $\zeta > 0$ um escalar. Definimos a perspectiva de f como a função $F : \mathbb{R}^n \times R_+ \to \mathbb{R}$ dada por:*

$$F(\pi, \zeta) = \zeta f(\pi/\zeta). \tag{6.11}$$

A perspectiva de uma função é uma função homogênea de grau 1 em π e ζ: se multiplicarmos π e ζ por um número positivo $r > 0$, o valor da função também é multiplicado por r.

Assim, para poder eliminar a variável z no caso risco-avesso, introduzimos uma nova função dual, que chamamos de $V_t^{\boxtimes}$, e é definida por

Definição 6.3 (Coperspectiva de uma função)**.** *A coperspectiva de uma função f é a perspectiva da conjugada de Fenchel de f, e é dada por*

$$f^{\boxtimes}(\pi, \zeta) = \max_{z\in\mathbb{R}^n} \left\{\pi^\top z - \zeta f(z)\right\} \tag{6.12}$$

para $\zeta > 0$. Esta função não está definida para $\zeta < 0$.

A coperspectiva de uma função é uma outra função, com dois argumentos: o primeiro é uma variável dual, aqui denotada por π, e o segundo é um escalar ζ. Esta outra função também deve ser pensada como uma função no espaço dual, e, no caso da aplicação à PDDE, será chamada de função de custo futuro *dual*.

Com isto, no caso com aversão ao risco, temos uma recorrência entre as coperspectivas $V_t^{\boxtimes}$ e $V_{t+1}^{\boxtimes}$, generalizando o caso neutro ao risco. De forma análoga à equação (6.7), teremos

$$\begin{aligned} V_t^{\boxtimes}(\pi, \gamma) = \inf_{\lambda,\nu,\zeta} \quad & \mathbb{E}\left[\lambda^\top b + V_{t+1}^{\boxtimes}(\nu, \zeta)\right] \\ \text{s.a} \quad & \mathbb{E}[B^\top \lambda] = \pi, \\ & \zeta c + W^\top \lambda \geq 0, \\ & \nu + A^\top \lambda \geq 0, \\ & \zeta \in \gamma \cdot Q, \end{aligned} \tag{6.13}$$

em que incluímos uma nova variável de estado γ, análoga a ζ para o estado inicial, e incluímos a restrição $\zeta \in \gamma \cdot Q$ onde Q é o conjunto de reponderações associado à medida de risco ρ. Note que a restrição $\zeta \in \gamma \cdot Q$ já estava presente na equação (6.9) quando definimos a medida de risco, e continua quando calculamos a recorrência entre as coperspectivas. Observamos que as variáveis de estado iniciais são π e γ (a primeira, dual ao estado primal, e a segunda, um escalar correspondente ao *change-of-measure*). As variáveis finais são, respectivamente, μ e ζ.

O papel dos escalares γ e ζ, aumentando a dimensão das variáveis de estado, é o de *possibilitar* a construção de uma recorrência no caso com aversão ao risco. De fato, podemos interpretar a aversão ao risco como uma mudança na probabilidade atribuída a cada cenário, em função do custo do mesmo. Assim, a variável ζ irá representar esta mudança de probabilidade para as diferentes

realizações da incerteza ao longo do estágio t, e a recorrência entre as coperspectivas $V_t^{\boxtimes}$ e $V_{t+1}^{\boxtimes}$ generaliza o caso neutro ao risco (que pode ser visto como o caso onde a reponderação é sempre igual a 1, e não precisaríamos acompanhar a sua evolução ao longo dos estágios).

Vale notar que a aversão ao risco do problema primal está presente na recorrência do problema dual através do conjunto Q, mas não diretamente na função objetivo, que contém apenas a esperança ao longo dos cenários. Com isto, é importante que o conjunto Q seja suficientemente simples para ser representado em um problema de otimização. No caso do CVAR, a restrição $\zeta \in \gamma \cdot Q$ pode ser representada por restrições de caixa para cada cenário, além da restrição de que $\sum \zeta_i = 1$ que é natural por ser uma probabilidade. Assim, a recorrência para a coperspectiva $V_t^{\boxtimes}$ é um problema de otimização linear, com um número modesto de restrições suplementares para representar a aversão ao risco.

6.3 PDDE dual para coperspectivas

As recorrências vistas nas seções anteriores, seja para a função conjugada V_t^*, seja para a coperspectiva $V_t^{\boxtimes}$, podem ser resolvidas por um algoritmo análogo à PDDE. De fato, cada uma das funções V_{t+1}^* ou $V_{t+1}^{\boxtimes}$ é uma função de custo futuro, convexa, que pode ser aproximada por cortes de Benders, da mesma forma que a função de custo futuro original V_{t+1}.

Como já mencionado no caso das conjugadas V_t^*, e continua sendo o caso das coperspectivas $V_t^{\boxtimes}$, a recorrência de programação dinâmica resulta em um problema de otimização que não pode ser decomposto em problemas independentes, um para cada realização $\omega_{t,i}$ da incerteza, pois as restrições dinâmicas (a primeira e a última, uma em π e a outra em γ) acoplam variáveis de todas as realizações às variáveis de estado *inicial*.

Enfim, observamos que, pela própria definição da coperspectiva, as funções $V_t^{\boxtimes}$ são convexas em π e γ. Mais ainda, como elas são perspectivas de V_t^*, estas são funções homogêneas no par (π, γ), isto é, para todo $\alpha > 0$ temos

$$V_t^{\boxtimes}(\alpha\pi, \alpha\gamma) = \alpha V_t^{\boxtimes}(\pi, \gamma). \tag{6.14}$$

Assim, os cortes para as funções $V_t^{\boxtimes}$ também serão homogêneos (ou seja, o *intercept* é sempre zero), o que pode ser incorporado ao algoritmo. De forma genérica, iremos indicar um corte para $V_t^{\boxtimes}$ como $V_t^{\boxtimes}(\pi, \gamma) \geq y^\top \pi + v \cdot \gamma$. Isto pode ser usado tanto para verificar a implementação como para reduzir a instabilidade numérica do método.

Na prática, o problema que será resolvido a cada iteração é uma versão modificada de (6.13). De fato, o objetivo do algoritmo é construir aproximações $D_t \leq V_{t+1}^{\boxtimes}$ para as funções de custo futuro. Mais ainda, estas aproximações serão representadas por um conjunto de cortes lineares. Para vermos concretamente, reintroduzimos aqui a dependência dos problemas e variáveis de estado no tempo e na realização da incerteza $\omega_{t,i}$. Também usamos uma notação uniforme para as variáveis de estado, que serão π_{t-1}, γ_{t-1} e $\pi_{t,i}, \gamma_{t,i}$ para os estágios $t-1$ e t, respectivamente (note que temos que resolver todas as realizações conjuntamente, mas temos apenas um estado inicial $(\pi_{t-1}, \gamma_{t-1})$ para o problema do t-ésimo estágio). A esperança será substituída pela soma em todas as incertezas,

com as respectivas probabilidades p_i, resultando no seguinte problema de otimização:

$$\begin{array}{lll} \inf\limits_{\lambda_t,\pi_t,\gamma_t} & \sum_i p_i \left[\lambda_{t,i}^\top b_{t,i} + \theta_{t,i}\right] & \\ \text{s.a.} & \sum_i p_i [B_{t,i}^\top \lambda_{t,i}] = \pi_{t-1}, & \text{Restrições acopladas} \\ & \gamma_{t,i} \in \gamma_{t-1} \cdot Q, & \\ & \gamma_{t,i} c_{t,i} + W_{t,i}^\top \lambda_{t,i} \geq 0 \quad \forall i, & \text{Restrições por realização} \\ & \pi_{t,i} + A_{t,i}^\top \lambda_{t,i} \geq 0 \quad \forall i, & \\ & \theta_{t,i} \geq y_{(k)}^\top \pi_{t,i} + v_{(k)} \gamma_{t,i} \quad \forall i, \forall k. & \text{Cortes de Benders} \end{array} \tag{6.15}$$

Esta não é, estritamente falando, uma formulação *multi-cut*, pois temos apenas uma função de custo futuro $V^{\boxtimes}$ por estágio, e não uma função de custo futuro para cada abertura. Isto fica evidente quando observamos que os cortes são indexados apenas pela iteração k do algoritmo, e não pela abertura i. Mesmo assim, devido à não-decomposição devido às restrições que acoplam todas as realizações, este problema também necessita de uma variável $\theta_{t,i}$ para representar a função de custo futuro em cada realização da incerteza - só que agora do próprio estágio t, em vez de corresponder ao estágio seguinte $t+1$.

O problema de primeiro estágio é especial, porque em vez de começarmos com um estado dual inicial π_0, o dado inicial que possuímos é o estado primal x_0. Por isso, *ambos* π_0 e π_1 serão determinados a partir do problema

$$\begin{array}{lll} \inf\limits_{\pi_0,\lambda_1,\pi_1,\gamma_1} & \sum_i p_i \left[\lambda_{1,i}^\top b_{1,i} + \theta_{1,i}\right] & \\ \text{s.a.} & \sum_i p_i [B_{1,i}^\top \lambda_{1,i}] = \pi_0, & \\ & \gamma_{1,i} \in 1 \cdot Q, & \\ & \gamma_{1,i} c_{1,i} + W_{1,i}^\top \lambda_{1,i} \geq 0 \quad \forall i, & \\ & \pi_{1,i} + A_{1,i}^\top \lambda_{1,i} \geq 0 \quad \forall i, & \\ & \theta_{1,i} \geq y_{(k)}^\top \pi_{1,i} + v_{(k)} \gamma_{1,i} \quad \forall i, \forall k. & \end{array} \tag{6.16}$$

Esta formulação mostra que π_0 corresponde ao gradiente da função de custo futuro *primal* em x_0. Naturalmente, ao longo das iterações do algoritmo este valor pode oscilar, até a convergência.

A recorrência acima se traduz em um algoritmo de otimização, similar à PDDE, com etapas *forward* e *backward*, que é conhecido como PDDE Dual, e é descrito no algoritmo 9. Uma das novidades deste algoritmo é que a construção dos cenários durante a etapa *forward* é feita por *importance sampling*, isto é, os cenários são selecionados com pesos proporcionais a $\gamma_{t,i} p_i + \epsilon$. Isto é diferente da PDDE usual, onde o cenário da etapa *forward* é selecionado antes da solução dos problemas de otimização, e é feito de forma uniforme. Como, ao resolver o problema do t-ésimo estágio, temos uma estimativa, dada pelos fatores $\gamma_{t,i}$, de quais cenários são mais importantes, a seleção por *importance sampling* é uma forma de acelerar a convergência do algoritmo. O parâmetro ϵ, por sua vez, tem um papel de regularização para controlar a exploração do espaço de cenários: ele garante que todos os cenários podem ser explorados, o que é importante para a convergência do algoritmo. Na prática, como explorado em [FL23a], o valor de ϵ não é muito crítico, mas um valor pequeno, relativamente às probabilidades p_i, é suficiente para garantir a convergência do algoritmo e ainda manter o efeito de acelerar a convergência.

Enfim, observamos que a escolha de cenários por *importance sampling* é muito útil para acelerar a convergência do algoritmo, como visto em [FML24]. Uma das razões para isto é que o uso de *importance sampling* irá melhorar as aproximações da função de custo futuro ao longo dos estados que mais vão contribuir para o custo futuro. Isto pode ser explicado, entre outros, pela

Algoritmo 9 PDDE dual para problemas com aversão ao risco

Require: Estado inicial x_0, número de iterações K.
Require: Parâmetro ϵ para *importance sampling*, constantes de Lipschitz L_t.
1: Inicialize $V_{T+1}^{\boxtimes}(\pi_T, \gamma_T)$.
2: **for** $k = 1, \dots, K$ **do**
3: Resolva o problema de primeiro estágio (6.16) para determinar π_1 ▷ Etapa *forward*
4: **for** $t = 2, \dots, T$ **do**
5: Resolva (6.15) e calcule os estados finais $\pi_{t,i}$ e $\gamma_{t,i}$.
6: Selecione o cenário j^* por *importance samplig* com pesos $\gamma_{t,i} p_i + \epsilon$.
7: Tome $\pi_t := \pi_{t,j^*}$ e $\gamma_t := \gamma_{t,j^*}$.
8: **end for**
9: **for** $t = T, \dots, 2$ **do** ▷ Etapa *backward*
10: Resolva (6.15) para calcular os multiplicadores de Lagrange para π_{t-1} e γ_{t-1}.
11: Inclua o corte correspondente na função de custo futuro $V_t^{\boxtimes}$ do estágio anterior.
12: **end for**
13: **end for**

homogeneidade das coperspectivas: quanto maior o valor de γ, também esperamos que o valor de $V_t^{\boxtimes}(\pi, \gamma)$ seja maior.

Analogamente ao algoritmo SIDP (algoritmo 7), que calcula cortes e vértices simultaneamente, vejamos uma ilustração do algoritmo PDDE dual, em paralelo com o próprio algoritmo de PDDE, na Figura 6.1, onde agora ambos algoritmos calculam cortes, seja para as funções de custo futuro, seja para as conjugadas de Fenchel (ou coperspectivas, no caso com aversão ao risco).

6.4 Cotas superiores para as funções de custo futuro

A construção da conjugada *inverte a ordem*, ou seja, se $f \leq g$ então $f^* \geq g^*$. Ora, no caso em que as funções valor ótimo são convexas, e temos que $V_t = (V_t^*)^*$, se possuímos uma aproximação inferior $\underline{V^*}_t$ para V_t^*, então sua conjugada $(\underline{V^*}_t)^*$ é uma aproximação *superior* para a função V_t. Em particular, no primeiro estágio do problema, a função conjugada V_1^* permite calcular uma cota superior para o valor do problema de otimização multi-estágio (2.1).

Agora, vejamos como as aproximações D_t das coperspectivas (funções de custo futuro duais) $V_t^{\boxtimes}$ podem ser utilizadas para construir cotas superiores para as funções de custo futuro primais V_t. Sabemos que $D_t \leq V_t^{\boxtimes}$, e que $V_t^{\boxtimes}(\pi, 1) = V_t^*(\pi)$. Assim, se $f(\pi) = D_t(\pi, 1)$, então $f(\pi) \leq V_t^*(\pi)$. Tomando a conjugada de Fenchel de ambos os lados, obtemos

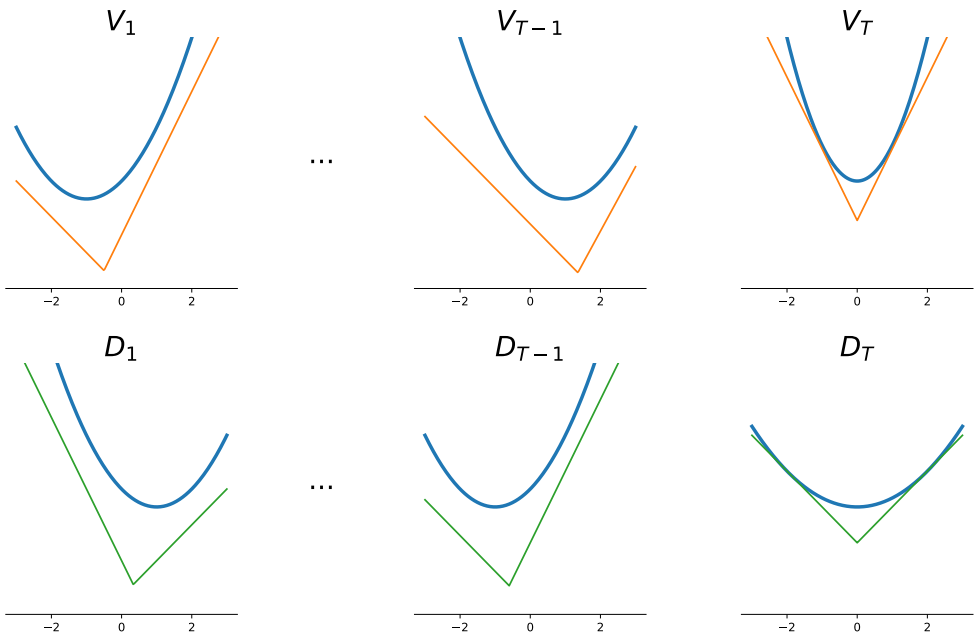
$$f^*(x) \geq V_t(x). \tag{6.17}$$

Assim, a função f^* é uma cota superior para a função de custo futuro V_t. Entretanto, esta função não estará definida para todos os valores de x. De fato, como f é uma aproximação por cortes lineares, a função conjugada f^* estará definida apenas para os valores de x que sejam uma combinação convexa dos cortes que definem f.

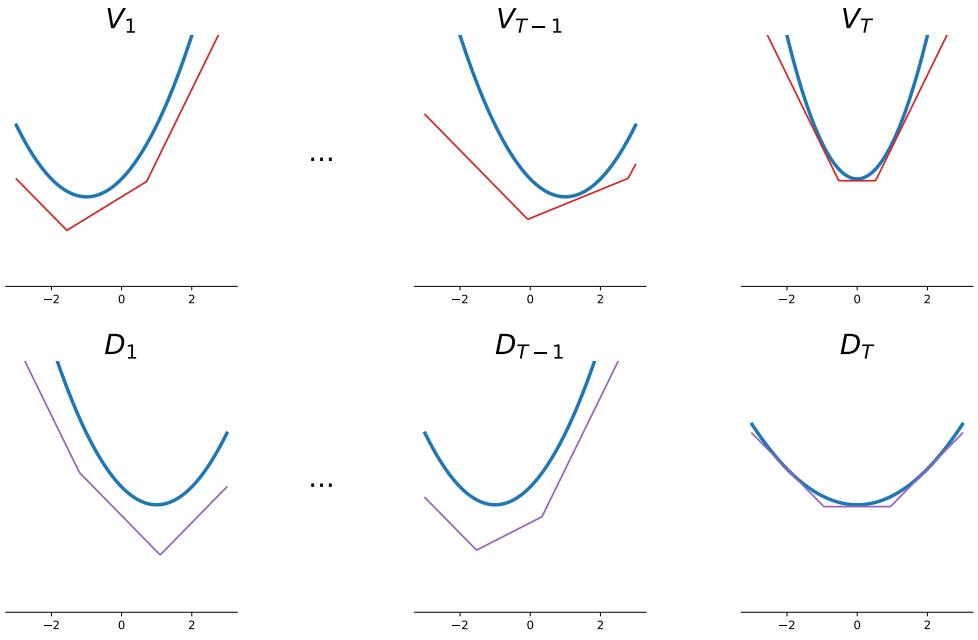
Isso é similar ao que acontece com as aproximações (primais) por vértices, que estão definidas apenas na envoltória convexa dos mesmos. Da mesma forma, também podemos relaxar esta cota utilizando uma constante de Lipschitz L_t para obter uma função que estará definida para todos os valores de x.

Vejamos um exemplo. Suponha que os cortes para D_t sejam dados por

$$D_t(\pi, \gamma) \geq (y_{(k)})^\top \pi + (v_{(k)}) \cdot \gamma \quad \forall k. \tag{6.18}$$



(a) Após 2 iterações



(b) Após adicionar o terceiro corte para ambas funções de custo futuro

Figura 6.1: Exemplo dos algoritmos de *PDDE* (funções V_t) e *PDDE dual* (funções $D_t = (V_t)^*$).

Daí, vemos que

$$f(\pi) = D_t(\pi, 1) \geq (y_{(k)})^\top \pi + v_{(k)} \quad \forall k. \tag{6.19}$$

Ora, a conjugada da função linear $\phi(\pi) = y^\top \pi + v$ é a função indicadora do conjunto $\{y\}$, transladada de v. Isto é fácil de ver pela própria definição:

$$\phi^*(x) = \max_\pi x^\top \pi - y^\top \pi - v = \begin{cases} -v, & \text{se } x = y, \\ +\infty, & \text{caso contrário.} \end{cases} \tag{6.20}$$

Assim, a conjugada de ϕ corresponde ao *vértice* $(y, -v)$. Como f é o máximo de várias funções lineares, a conjugada de f será a combinação convexa dos vértices correspondentes a cada corte.

Assim, para representar uma cota superior para V_t, basta utilizar a mesma fórmula para as funções primais, construídas a partir de vértices, só que agora utilizando os coeficientes dos cortes de D_t para determinar os valores dos vértices.

7 Aumento do espaço de estados

Nesta seção, passamos aos métodos estatísticos para o cálculo de cotas superiores em problemas de otimização estocástica multi-estágio.

Após uma breve revisão do problema de otimização e sua notação, veremos que a dificuldade de utilizar métodos estatísticos advém da necessidade de calcular uma medida de risco dada por uma expressão com CVAR's aninhados, que é a formulação típica dos problemas de planejamento da operação do setor elétrico. Vale lembrar que esta formulação, apesar da dificuldade acima exposta, é o que permite o uso de técnicas de programação dinâmica e também garante a *consistência temporal* das decisão de planejamento obtidas através do modelo de otimização.

Apresentaremos a seguir uma modificação da formulação do problema multi-estágio, que introduz uma variável de estado auxiliar para as funções de custo futuro, cuja função é contornar as limitações impostas pelas medidas de risco aninhadas. Ainda é possível aplicar o algoritmo de PDDE para estas novas funções, e com isto obter uma política (aproximada) para a operação, e daí veremos que uma modificação do algoritmo de simulação de Monte Carlo irá, de fato, produzir cotas superiores para o risco associado à política.

Veremos, no entanto, que estas funções de custo futuro estendidas podem sofrer de uma propagação exponencial de erros de estimação, que, muitas vezes, levam a cotas superiores demasiado grandes para serem utilizadas como critério de parada ou avaliação da qualidade das soluções obtidas pela PDDE.

7.1 Problemas multi-estágio com CVAR

Consideramos o clássico problema multi-estágio neutro a risco:

$$\min\ \mathbb{E}\left[\sum_{t=1}^{T} c^\top y_t\right],$$

onde, por simplicidade de notação, omitimos as restrições subjacentes ao problema minimização envolvendo as variáveis de estado x_t e as variáveis de decisão (ou "controle") y_t. Para fixar a notação e o significado das variáveis com relação ao tempo, tomamos o estado inicial como x_0, e a restrição dinâmica é dada por $A_t x_t + B_t x_{t-1} + W_t y_t = d_t$.

Sob a hipótese de que a incerteza é *stagewise independent*, podemos reformular através de uma recorrência de programação dinâmica:

$$V_t(x_{t-1}) = \min\ \mathbb{E}_t\left[c_t^\top y_t + V_{t+1}(x_t)\right] \quad \text{para } t = 1, \ldots, T-1,$$

onde $\mathbb{E}_t$ denota a esperança em relação à incerteza ξ_t no estágio t. Esta recorrência define as funções de custo futuro $V_t(x_{t-1})$, que representam o valor ótimo do problema a partir do estágio t, cujo estado inicial é x_{t-1}. A função de custo futuro do último estágio, $V_{T+1}(x_T)$, é identicamente nula, por definição, e permite iniciar a programação dinâmica.

No caso averso a risco, iremos substituir $\mathbb{E}$ por CVAR_α. Isto leva à recorrência modificada:

$$V_t(x_{t-1}) = \min\ \text{CVAR}_\alpha\left[c_t^\top y_t + V_{t+1}(x_t)\right]. \tag{7.1}$$

Esta recorrência define um problema de otimização multi-estágio com CVAR, que é o objeto de estudo deste projeto. Note que ele leva a problemas com "CVAR's aninhados". Por exemplo, desenvolvendo duas etapas obtemos

$$V_1(x_1) = \min\ c^\top y_1 + \text{CVAR}_\alpha\left[c^\top y_2 + \text{CVAR}_\alpha\left[c_t^\top y_3 + \cdots\right]\right].$$

Naturalmente, é possível utilizar outras medidas de risco no lugar do CVAR. Uma alternativa muito comum no setor elétrico é utilizar uma combinação entre o CVAR e o valor esperado, que permite equilibrar os aspectos de risco e custo médio da operação. Por simplicidade das discussões que serão feitas neste relatório, especialmente referentes às deduções matemáticas das fórmulas, nos concentraremos no caso em que a medida de risco é dada simplesmente pelo CVAR.

7.2 Inclusão do quantil como variável de estado

Quando o número de cenários da árvore é pequeno, é possível calcular o CVAR de uma política de controle y_t diretamente: calcula-se o custo $c_T^\top y_T$ do último estágio, e a partir deste obtemos o CVAR em cada nó da árvore do estágio anterior. Daí, somamos este CVAR ao custo imediato do estágio anterior, obtemos o CVAR desta etapa, e assim por diante, até o primeiro estágio.

Poderíamos tentar generalizar esta abordagem para um método de Monte Carlo, amostrando cenários em vez de considerá-los todos. Entretanto, o cálculo do CVAR em um dado estágio envolve a estimativa dos custos futuros de todos os cenários, o que torna o método *exponencial* no número de estágios.

Um primeiro passo para abordar este problema é a reformulação da recorrência de programação dinâmica (7.1) correspondente à otimização multi-estágio com CVAR. Generalizando a formulação da equação (4.3), introduzimos as variáveis auxiliares q_t para calcular o CVAR dos custos futuros, resultando no seguinte problema:

$$\min_{x_t, y_t} \; c^\top y_1 + \min_{q_1} q_1 + \frac{1}{\alpha}\mathbb{E}_2\left[\left(c^\top y_2 + \min_{q_2} q_2 + \frac{1}{\alpha}\mathbb{E}_3\left[\left(c^\top y_3 + \cdots - q_2\right)_+\right] - q_1\right)_+\right]. \tag{7.2}$$

Entretanto, para determinar o valor de $q_2(\xi_2)$ é necessário calcular os custos de todos os cenários (ou uma amostra deles) descendentes do nó ξ_2. Portanto, para calcular o valor de q_1, que depende recursivamente de todos os valores $q_2(\xi_2)$, teremos que continuar resolvendo um número exponencial de cenários.

Para contornar esta situação, iremos transformar os q_t's em variáveis de estado. Assim, em vez de serem consideradas variáveis auxiliares para o cálculo do CVAR, elas passam a ser consideradas variáveis de estado, e a recorrência de programação dinâmica passa a ser

$$\begin{aligned} Q_t(x_{t-1}, q_{t-1}) &= \min_{y_t, q_t} \; \mathbb{E}_t\left[\left(c^\top y_t + q_t + \frac{1}{\alpha}Q_{t+1}(x_t, q_t) - q_{t-1}\right)_+\right], \quad t \le T, \\ Q_{T+1}(x_T, q_T) &= 0. \end{aligned}$$

Essa reformulação calcula os valores de y_t e q_t sem ter que avaliar os cenários seguintes. Simbolicamente, ela corresponde a estimar o cálculo do CVAR através de uma função de custo futuro $Q(x, q)$, em vez de calcular exatamente o CVAR para obter a função $V(x)$. Por causa disto, esta nova função de custo futuro tem uma interpretação mais complexa, visto que ela depende também do valor de q, que não é uma variável física do sistema. Mais ainda, pode parecer estranho que a função de custo futuro dependa de uma variável auxiliar, quando sabemos que, de fato, o custo futuro em um dado instante t (mais precisamente, o *risco futuro* devido ao CVAR), só depende do estado corrente $\hat{x}_t$.

Com (aproximações para) as funções $Q(x, q)$, é possível simular os valores de um único cenário independentemente dos outros, já que tanto a decisão de controle y_t como a estimativa do VAR, q_t, são feitas em função do cenário atual e da função de custo futuro estendida $Q(x, q)$. Como o problema com as variáveis auxiliares q_t só possui esperanças, é razoável esperar que um método de Monte Carlo possa ser utilizado para estimar o valor do problema.

7.3 Método de Monte Carlo com variável de estado suplementar

O método de Monte Carlo para estimar uma esperança $\mathbb{E}[f(\omega)]$ é simples: basta gerar uma amostra ω_i da variável aleatória ω e calcular a média de $f(\omega_i)$. No caso de otimização multi-estágio, a variável aleatória é um cenário completo, ou seja, uma realização $(\xi_2, \xi_3, \ldots, \xi_T)$ das incertezas a cada estágio[5]. Ora, por mais que o problema possua apenas esperanças, a função que queremos estimar não está sob a forma $\mathbb{E}[f(\omega)]$, mas sim com uma sequência intercalada de esperanças e partes positivas.

Para entender este fenômeno, vamos retomar a equação (7.2), imaginando que o horizonte é de 3 estágios. Se já temos uma política definida (por funções $Q_t(x_{t-1}, q_{t-1})$), podemos ignorar o problema de minimização (tanto em x_t, y_t, como em q_t), já que o valor da função objetivo[6] seguindo esta política será dado por

$$c^\top y_1 + q_1 + \frac{1}{\alpha}\mathbb{E}_2\left[\left(c^\top y_2 + q_2 + \frac{1}{\alpha}\mathbb{E}_3\left[(c^\top y_3 - q_2)_+\right] - q_1\right)_+\right], \tag{7.3}$$

onde os valores das variáveis y_t e q_t foram determinados pela política. Calculando a operação ao longo da k-ésima trajetória, e obtendo os custos e quantis em cada estágio, temos apenas acesso a expressões sem valor esperado, da forma

$$c^\top y_1^{(k)} + q_1^{(k)} + \frac{1}{\alpha}\left(c^\top y_2^{(k)} + q_2^{(k)} + \frac{1}{\alpha}\left(c^\top y_3^{(k)} - q_2^{(k)}\right)_+ - q_1\right)_+. \tag{7.4}$$

Se tomarmos a média de tais valores para K trajetórias, como faríamos em um método de Monte Carlo, obteríamos uma estimativa para

$$c^\top y_1 + q_1 + \frac{1}{\alpha}\mathbb{E}_2\left[\mathbb{E}_3\left[\left(c^\top y_2 + q_2 + \frac{1}{\alpha}\left(c^\top y_3 - q_2\right)_+ - q_1\right)_+\right]\right], \tag{7.5}$$

onde o valor esperado com relação à incerteza do terceiro estágio está *fora* da parte positiva, se comparado com a expressão (7.3). Note que y_1 e q_1 são determinados no primeiro estágio, e por isso são independentes da incertezas do problema.

Como a função parte positiva é convexa, temos, pela desigualdade de Jensen:

$$\mathbb{E}_3\left[f(z)\right] \geq f\left(\mathbb{E}_3[z]\right) \tag{7.6}$$

e portanto

$$\mathbb{E}_2\Big[\mathbb{E}_3\left[f(z)\right]\Big] \geq \mathbb{E}_2\Big[f\left(\mathbb{E}_3[z]\right)\Big], \tag{7.7}$$

o que quer dizer que a expressão (7.5) é sempre maior do que a expressão (7.3), devido à diferença entre o momento do cálculo da esperança e da parte positiva.

Em um método de Monte Carlo, temos acesso aos valores de (7.4) para um conjunto de K cenários. A média deles é um estimador não-viesado para a expressão dada pela equação (7.5). Ora, este é exatamente o lado esquerdo da equação (7.7), onde a função f é a parte positiva mais externa. Portanto, média obtida por Monte Carlo está acima da expressão correspondente ao lado direito da equação (7.7), que, no caso, é a expressão (7.3). Assim, o estimador de Monte Carlo construído a partir da inclusão da variável de estado suplementar q está de fato *superestimando* o CVAR aninhado da política induzida pelas funções $Q_t(x_{t-1}, q_{t-1})$, que seria obtido se calculássemos

[5] o primeiro estágio é considerado determinístico

[6] "*Risk-adjusted cost*"

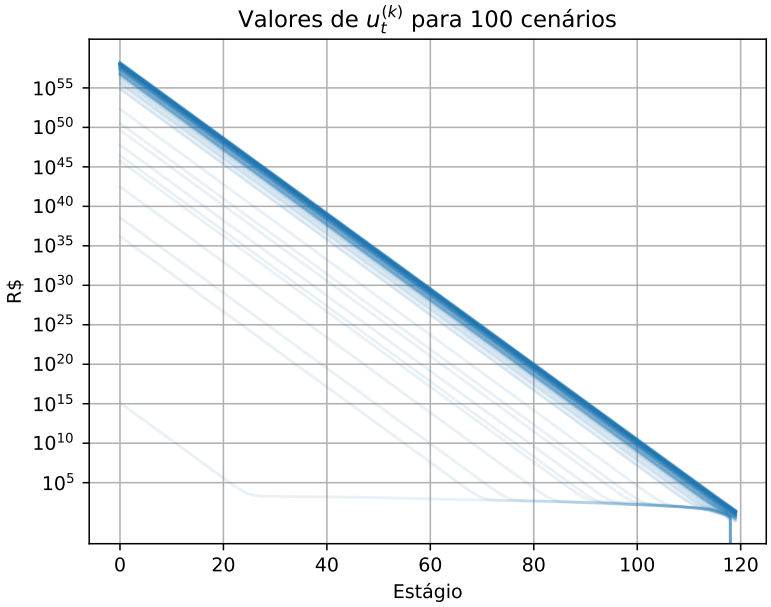


Figura 7.1: Estimativa da cota superior em 120 estágios.

exatamente a expressão (7.3). Teoricamente, isto não é um problema, já que desejamos de fato cotas *superiores* para o problema de otimização e a política induzida, e o viés está no sentido correto.

Entretanto, isso gera um problema significativo em problemas com um grande número de estágios. Se, em vez de 3 estágios, tivermos 120 estágios, ao realizar a média de uma trajetória, o erro de Jensen ocorrerá recursivamente ao longo do tempo. Este erro será, a cada etapa da média, multiplicado por um fator $\frac{1}{\alpha}$, o que leva a uma propagação exponencial do erro, como visto na Figura 7.1, inspirada da análise de [DS16]. Para compreender o fenômeno representado na figura, note que o cálculo da expressão (7.5) para um cenário é feito de forma recursiva, onde a cada estágio incluímos o custo imediato e subtraímos o valor de q_t do estágio anterior, multiplicamos a parte positiva por $\frac{1}{\alpha}$ e somamos novamente o valor de q_t, o que dá uma recorrência da forma:

$$\begin{aligned} u_T^{(k)} &= q_{T-1}^{(k)} + \frac{1}{\alpha}\left(c_T^\top y_T^{(k)} - q_{T-1}^{(k)}\right)_+, \\ &\vdots \\ u_2^{(k)} &= q_1^{(k)} + \frac{1}{\alpha}\left(c_2^\top y_2^{(k)} + u_3^{(k)} - q_1^{(k)}\right)_+, \\ u_1^{(k)} &= c_1^\top y_1^{(k)} + u_2^{(k)}. \end{aligned} \tag{7.8}$$

Assim, os valores de $u_t^{(k)}$ vão crescendo do último até o primeiro estágio, e são eles que estão representados na Figura 7.1.

Observe que a escala do gráfico é *logarítmica*, que mostra o crescimento exponencial dos valores de $u_t^{(k)}$ conforme nos aproximamos do primeiro estágio. Como mencionamos, este crescimento pode ser atribuído ao fator multiplicativo $\frac{1}{\alpha}$ (neste exemplo em que $\alpha = 0.2$, o fator é 5), que amplifica o erro de Jensen a cada estágio.

Note que o valor de $q_t^{(k)}$ é estimado para cada estado inicial $x_t^{(k)}$, ao mesmo tempo que a política de controle, e portanto também depende do cenário. Entretanto, como este está sempre abaixo da

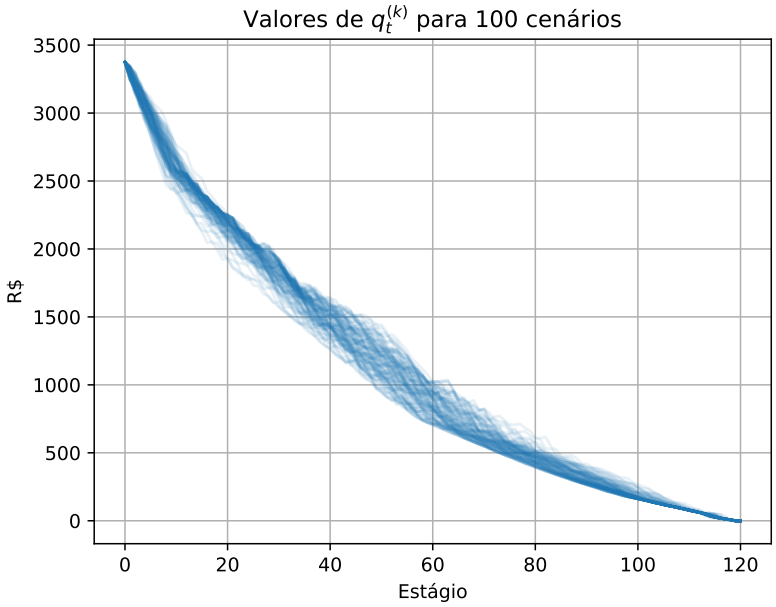


Figura 7.2: Estimativa de $q_t^{(k)}$ em 120 estágios.

cota inferior (porque o custo imediato de cada estágio será pelo menos q_t!), ele cresce de forma no máximo linear a cada estágio, como vemos na Figura 7.2.

8 Importance Sampling

A ideia fundamental de *Importance Sampling* é de alterar as probabilidades dos cenários, de forma a aumentar as chances de obter cenários que ativarão o CVAR. Isto permite reduzir a variância do estimador da média feito por simulações de Monte Carlo.

O CVAR (e, na verdade, qualquer medida de risco), induz uma distorção nas probabilidades dos cenários. Retomando a fórmula da equação (4.2), esta distorção é capturada pelas "novas" probabilidades $\zeta_i p_i$, que entram tanto como peso na soma, como continuam somando 1. Assim, uma ideia natural para o *Importance Sampling* é de usar estas probabilidades distorcidas para amostrar os cenários. Se pudéssemos fazê-lo, teríamos uma fórmula para o CVAR da forma $\mathbb{E}[\zeta Z]$, que não inclui uma função de parte positiva como na seção anterior. Isto evita o problema da composição de erros pela aplicação sucessiva de desigualdades de Jensen para a parte positiva.

Entretanto, estas probabilidades são desconhecidas, em geral, pois a probabilidade de um cenário específico depende de *todos os cenários* da árvore, o que impede o seu cálculo para problemas de horizonte longo como considerado neste projeto. Assim, uma possibilidade seria estimar estas probabilidades, e usá-las para o *Importance Sampling*. Antes de discutir como isto pode ser feito, vamos recapitular o método de Monte Carlo para estimar o custo esperado de uma política, no caso neutro ao risco.

8.1 Estimativas de Monte Carlo

Dado um problema de otimização estocástica multi-estágio, neutro ao risco, suponha que possuímos uma política π, dada por funções $\pi_t : X \times \Omega \to Y \times X$. Cada uma destas funções π_t determina,

para o estágio t, a ação y_t e o estado final correspondente x_t, para um estado inicial x_{t-1} e uma realização da incerteza ξ_t. Uma forma de construír uma política, bastante comum no setor elétrico, é partir de um conjunto de (aproximações das) funções de custo futuro $\{\hat{V}_t\}$. A partir das $\{\hat{V}_t\}$, a política é dada pela solução de problemas de otimização (*forward*), onde substituímos a função de custo futuro "ideal" pela estimativa $\hat{V}_t$:

$$\pi_t(x_{t-1}, \xi_t) = \underset{x_t, y_t \text{ viáveis}}{\operatorname{argmin}} \; c_t^\top y_t + \hat{V}_t(x_t), \tag{8.1}$$

onde a incerteza ξ_t pode modificar o conjunto viável e/ou o vetor de custos c_t.

O método tradicional de estimativa de Monte Carlo para o valor esperado do custo da política π é simplesmente simular um grande número de cenários, e calcular a média dos custos obtidos. Vamos utilizar um problema de três estágios para ilustrar o método de Monte Carlo, como pode ser visto na Figura 8.1. Suponha que escolhemos 4 cenários dos 12 possíveis na árvore de cenários, que estão destacados em azul e correspondem aos cenários 3, 7, 8 e 10. Para cada um destes cenários,

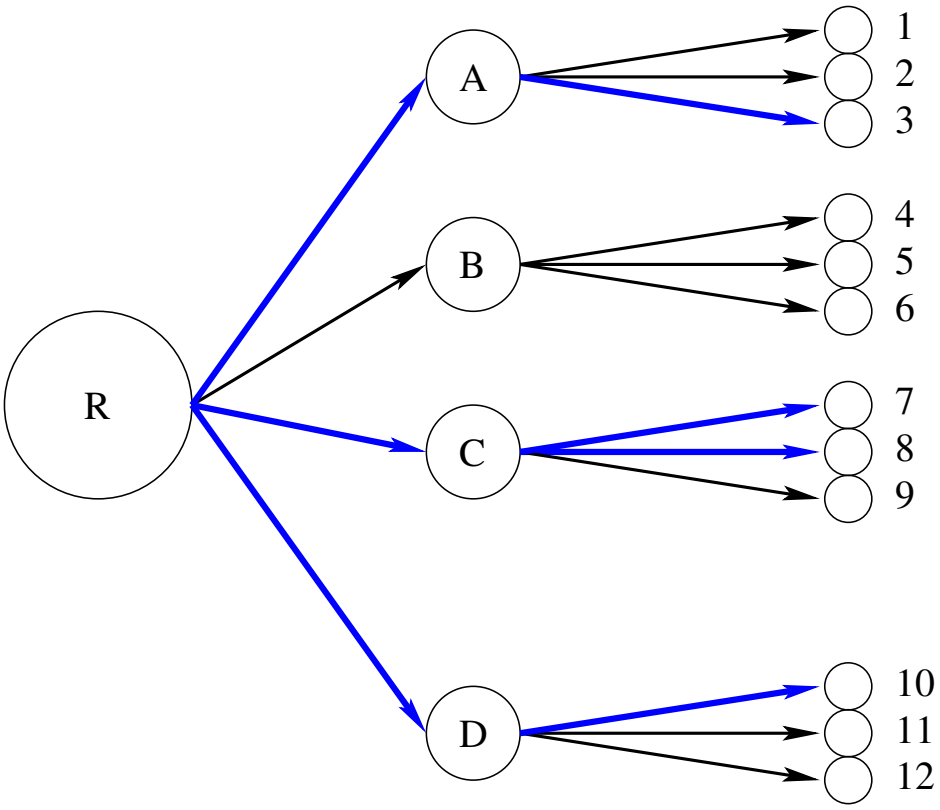


Figura 8.1: Exemplo de simulação de Monte Carlo para um problema de três estágios, explorando 4 cenários da árvore.

podemos calcular o custo de implementar a política π.

Em um problema com um maior número de estágios, o número total de cenários cresce exponencialmente, donde a simulação de Monte Carlo irá visitar apenas uma parte muito reduzida dos cenários possíveis. Ainda assim, a média dos custos obtidos através da simulação de Monte Carlo é uma estimativa não-viesada do custo esperado da política π.

Mais formalmente, este método pode ser descrito pelo Algoritmo 10, que calcula tanto uma estimativa da média como da variância dos custos de cada cenário, e com isto pode ser utilizado para construir intervalos de confiança do custo esperado da política π.

Algoritmo 10 Estimativa de Monte Carlo para o valor esperado da função objetivo

Require: Política $\{\pi_t\}$ para $t = 1, \ldots, T$
Require: Número de cenários N
Inicialize $C = []$
for $i = 1, \ldots, N$ **do**
  Simule uma realização $\xi_1, \ldots, \xi_T$ das incertezas a cada estágio, definindo um cenário
  for $t = 1, \ldots, T$ **do**
    Calcule as decisões e o estado final do estágio t pela política, $y_t, x_t = \pi_t(x_{t-1}, \xi_t)$
    Calcule o custo imediato $c_t^{(i)} = c_t^\top y_t$
  end for
  Calcule $c^{(i)} = \sum_{t=1}^{T} c_t^{(i)}$
  Adicione $c^{(i)}$ à lista C
end for
Calcule a média dos custos obtidos, $\hat{V} = \frac{1}{N} \sum_{i=1}^{N} c^{(i)}$
Calcule o desvio-padrão $\hat{\sigma}$ dos custos obtidos
return $\hat{V}, \hat{\sigma}$

8.2 Importance Sampling na PDDE dual

Um primeiro artigo que trata do uso de *Importance Sampling* em problemas de otimização multi-estágio é o da PDDE dual para problemas risco-avessos [FL23a]. Nele, as probabilidades $\zeta_i p_i$ aparecem naturalmente na recorrência de programação dinâmica como as variáveis de estado auxiliares γ_i, necessárias para a incorporação da aversão ao risco no problema dual. Se o problema dual estiver completamente resolvido, o valor das variáveis γ_i será exatamente a probabilidade do nó correspondente na árvore de cenários. Na prática, com as aproximações feitas pelo algoritmo, as variáveis γ_i são uma estimativa destas probabilidades.

Como o problema dual, a cada estágio, permite estimar as probabilidades dos seus descendentes imediatos, o procedimento abaixo mostra como é possível usar estas probabilidades para o *Importance Sampling*:

1. Resolve-se o problema dual para o estágio t, obtendo as probabilidades γ_i dos cenários (e estados futuros π_i);
2. Sorteia-se o cenário futuro i^* de acordo com as probabilidades γ_i;
3. O estado inicial do próximo estágio é o estado final π_{i^*} do cenário sorteado.
4. Repete-se o processo até o último estágio.

Uma diferença interessante desta abordagem para a construção de cenários clássica da PDDE é que a escolha do cenário futuro depende da solução do cenário atual. Isto não é um problema em si, mas diversas implementações do algoritmo de PDDE realizam o sorteio da trajetória completa antes de resolver os problemas de otimização de cada estágio. No artigo [FML24], os autores mostram que esta abordagem acelera a convergência do algoritmo, por visitar mais frequentemente os cenários que ativam o CVAR, e que com isto contribuem para a construção da função de custo futuro.

Vale notar que o uso das probabilidades γ_i para o *Importance Sampling* é uma abordagem apenas para buscar acelerar a convergência do algoritmo. O fato que a PDDE dual calcula uma cota *superior* para o problema de otimização estocástica multi-estágio com CVAR vem exclusivamente da sua formulação, e é independente da forma como são feitas as trajetórias *forward* do algoritmo.

8.3 Importance Sampling na PDDE primal

O uso de *Importance Sampling* na PDDE primal vem de uma outra reformulação, pois a solução do problema primal não fornece diretamente as probabilidades dos cenários. Assim, em [GLCP23], os autores propõem o uso da formulação *multicut* da PDDE para obter uma estimativa destas probabilidades. A ideia é que, ao utilizar a formulação *multicut*, a solução do problema de cada estágio irá estimar o custo futuro para cada uma das aberturas possíveis do estágio seguinte, através de funções $V_{t+1,i}(x_t)$ correspondentes a cada abertura i.

Matematicamente, a formulação da recorrência de programação dinâmica para a PDDE primal com *multicut* para um problema com CVAR é

$$\begin{aligned} V_{t,j}(x_{t-1}) = \min_{y_t} \quad & c^\top y_t + q_t + \tfrac{1}{\alpha}\textstyle\sum u_i \\ \text{s.a.} \quad & q_t + u_i \geq z_i \\ & u_i \geq 0 \\ & z_i \geq V_{t+1,i}(x_t) \\ & A_t x_t + B_t x_{t-1} + W_t y_t = d_{t,j} \end{aligned} \tag{8.2}$$

Aqui, reconhecemos inicialmente a formulação de Rockafeller-Uryasev para o CVAR, com a introdução de variáveis auxiliares q_t e u_i para a parte positiva, como descrito na equação 4.1. Em seguida, observamos a substituição da função de custo futuro $V_{t+1}(x_t)$ por funções $V_{t+1,i}(x_t)$, que correspondem a cada abertura i do estágio seguinte, que serão em seguida representadas por cortes, diferentes (especializados) para cada abertura.

Assim, mesmo sem resolver explicitamente os cenários futuros de cada abertura, a formulação *multicut* possui uma estimativa dos custos de cada um deles. Mais ainda, no caso específico do CVAR, a formulação ainda dá, para cada abertura, uma variável indicadora se o cenário irá ou não contribuir para o CVAR: se $u_i > 0$, o cenário i tem um custo acima do VAR. Portanto:

$$\begin{aligned} u_i = 0 &\Rightarrow \text{probabilidade zero para o CVAR} \\ u_i > 0 &\Rightarrow \text{probabilidade positiva para o CVAR.} \end{aligned}$$

A menos do cenário que está exatamente no valor do VAR, a probabilidade dos cenários que ativam o CVAR é estimada em $\frac{p_i}{\alpha}$, que é maior do que a probabilidade original, pois $\alpha < 1$ (e, em geral, α é relativamente pequeno). Isto corresponde à transferência de probabilidade entre os cenários "baratos" e os mais caros.

Vejamos novamente o exemplo da seção 4, no contexto do *Importance Sampling*. Imagine que a medida de risco utilizada é o CVAR com $\alpha = 0.5$, para o primeiro estágio do problema da Figura 8.1. Há 4 realizações possíveis para a incerteza de segundo estágio, correspondentes aos cenários A, B, C e D, que supomos equiprováveis. Suponha que o custo futuro (aproximado pelos cortes) para cada um destes cenários seja $V_2(x_{1,i}) = 1$, 2, 3 e 4, respectivamente. Ora, os dois cenários mais caros serão C e D, e portanto o *Importance Sampling* irá atribuir 50% de probabilidade para cada um destes cenários, e 0 para os demais. Assim, realizando simulações de Monte Carlo, iremos apenas considerar os cenários descendentes de C e D. Isso resultaria numa exploração de cenários como vemos na Figura 8.2.

Naturalmente, é mais típico utilizar uma medida de risco que combina o CVAR com a média. Neste caso, apenas a parte correspondente ao CVAR será distribuída. Voltando ao exemplo em questão, se considerarmos um peso de 60% para a média e 40% para o CVAR, a probabilidade ajustada de cada cenário para o *Importance Sampling* será, respectivamente, 15%, 15%, 35% e 35%. Note que, se houver uma componente de média, todos os cenários terão uma probabilidade de serem selecionados, ela será apenas aumentada ou diminuída conforme o custo futuro estimado.

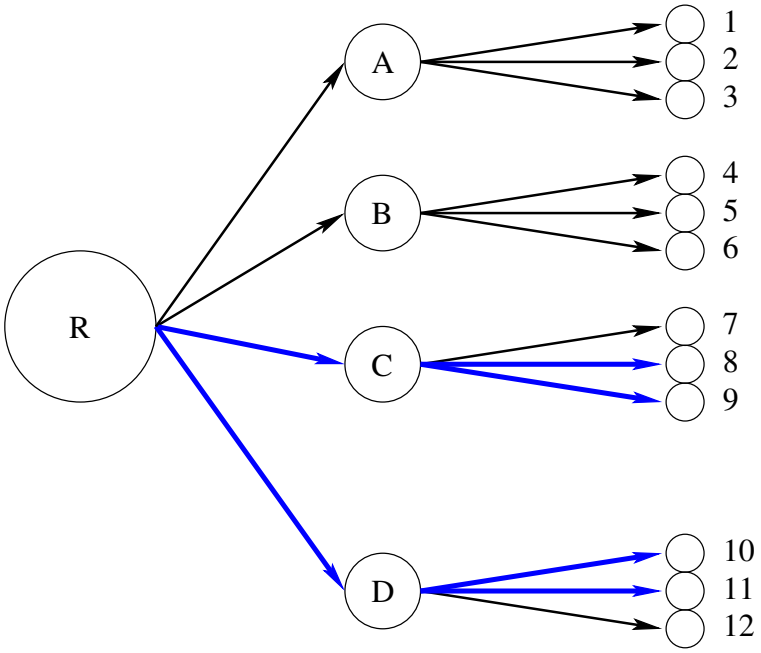
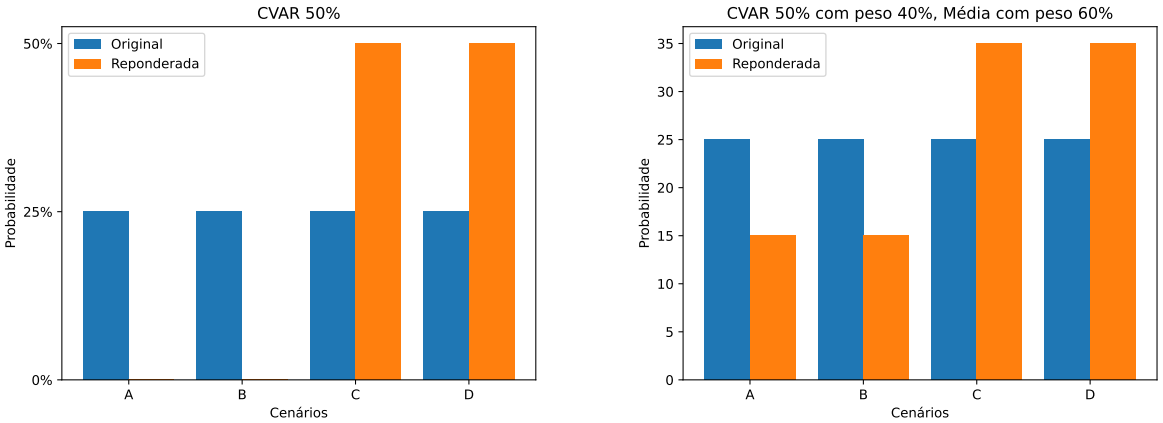


Figura 8.2: Método de Monte Carlo com *Importance Sampling*, explorando 4 cenários.



(a) $\text{CVAR}_{50\%}$. (b) 60% Esperança + 40% do $\text{CVAR}_{50\%}$.

Figura 8.3: Reponderação de cenários com diferentes medidas de risco

Esses exemplos de transferência de probabilidade podem ser vistos nas figuras 8.3a e 8.3b.

Assim, a PDDE primal com *multicut* já possui elementos que permitem estimar as probabilidades dos cenários que ativam o CVAR, e estas podem ser usadas para o *Importance Sampling*. Em certo sentido, ao utilizar as funções de custo futuro *multicut*, o algoritmo de PDDE se permite "olhar um passo à frente" e com isto produzir uma estimativa das probabilidades dos cenários que ativam o CVAR. Ao sortear os cenários futuros de acordo com as probabilidades alteradas ao inspecionar as variáveis u_i, o algoritmo de PDDE primal também constrói uma etapa *forward* de forma incremental, estágio por estágio, que eles denominam "amostra de cenário ajustada pelo risco".

Com isto, a ideia de [GLCP23] é de usar o *Importance Sampling* para que os custos $\sum_{t=1}^{T} c_t^\top y_t$ de uma trajetória sejam calculados com uma probabilidade ajustada, que dá mais peso aos cenários que ativam o CVAR aninhado correspondente ao problema multi-estágio. Assim, este método não irá empregar as variáveis auxiliares q_t para o cálculo do CVAR, nem tomar partes positivas como nos métodos da seção anterior, mas *apenas* calcular o custo acumulado em cada um dos cenários amostrados por Monte Carlo. Desta forma, o método proposto irá calcular um estimador de Monte Carlo para $\mathbb{E}_Q[\sum_{t=1}^{T} c_t^\top y_t]$, onde Q é a distribuição de probabilidade ajustada pelas estimativas dadas pelo *multicut*.

O algoritmo proposto em [GLCP23], e adaptado à notação deste relatório, está descrito no Algoritmo 11.

Algoritmo 11 Simulação com *Importance Sampling* para PDDE primal com *multicut*

Require: Política $\{\pi_t\}$ para $t = 1, \ldots, T$
Require: Número de cenários N
Inicialize $C = []$
for $i = 1, \ldots, N$ **do**
    Sorteie o cenário do primeiro estágio, ξ_1 de acordo com as probabilidades ajustadas pelos diferentes *lower bounds* correspondentes.
    for $t = 1, \ldots, T$ **do**
        Calcule as decisões e o estado final do estágio t pela política, $y_t, x_t = \pi_t(x_{t-1}, \xi_t)$
        Calcule o custo imediato $c_t^{(i)} = c_t^\top y_t$
        Sorteie a realização ξ_{t+1} da incerteza no estágio seguinte de acordo com as probabilidades ajustadas pelos custos das diferentes aberturas do *multicut*
    end for
    Calcule $c^{(i)} = \sum_{t=1}^{T} c_t^{(i)}$
    Adicione $c^{(i)}$ à lista C
end for
Calcule a média dos custos obtidos, $\hat{V} = \frac{1}{N} \sum_{i=1}^{N} c^{(i)}$
Calcule o desvio-padrão $\hat{\sigma}$ dos custos obtidos
return $\hat{V}, \hat{\sigma}$

Esta abordagem tem duas vantagens claras: a primeira é que ela não requer a formulação da PDDE dual, que é mais custosa computacionalmente, e também implica em uma implementação totalmente diferente da atual, que é baseada na PDDE primal. A segunda é que ela praticamente não requer a resolução de problemas de otimização adicionais, pois a formulação *multicut* já dá uma estimativa dos custos futuros para cada abertura, e permite realizar o *Importance Sampling* a cada estágio. A terceira é que ela é muito similar à abordagem clássica do método de Monte Carlo

para estimar o custo médio de uma política em problemas de otimização multi-estágio, e portanto requer poucas modificações na implementação atual.

Entretanto, como inclusive é possível observar nos resultados experimentais de [GLCP23], a qualidade do estimador depende fortemente da qualidade das estimativas dadas pelo *multicut*. Em particular, o valor real do CVAR aninhado da política é dado por

$$\sup_{P \in \mathcal{Q}} \mathbb{E}_P \left[\sum_{t=1}^{T} c_t^\top y_t \right],$$

onde $\mathcal{Q}$ é o conjunto de todas as distribuições de probabilidade sobre a árvore de cenários que satisfazem, a cada nó, a condição de calcular um CVAR das suas aberturas. Portanto, o estimador de Monte Carlo assim construído tem um viés *para baixo*, pois a probabilidade utilizada é apenas uma das possíveis distribuições que satisfazem a condição de CVAR. Este viés é um defeito quando o uso do estimador é para calcular um *upper bound* para o CVAR da política, que possa ser utilizado como um limitante superior para o CVAR real.

Parte III
Implementação e Resultados

9 Ambiente computacional e infraestrutura de pacotes

Para a realização das simulações computacionais, contamos com uma infra-estrutura de pacotes em Julia específicos para otimização estocástica multi-estágio e aplicações auxiliares em Python:

- `SDDP.jl`, que implementa uma generalização do algoritmo de programação dinâmica dual estocástica, primal [DK17], `https://github.com/odow/SDDP.jl/`;
- `DualSDDP.jl`, que implementa todos os algoritmos de programação dinâmica dual estocástica: primal, dual, *problem-child* [FL23a], `https://github.com/bfpc/DualSDDP.jl/`;
- `SDDPLab.jl`, que implementa uma interface para o pacote `SDDP.jl` para representar problema de planejamento da operação de sistemas hidrotérmicos, contemplando a leitura de arquivos de dados e o armazenamento de resultados, `https://github.com/rjmalves/sddp-lab`;
- `Lab2MSLBO.jl`, que converte uma representação do `SDDP.jl` para o formato necessário para o pacote `DualSDDP.jl`, `https://github.com/lkhenayfis/lab2mslbo`;
- `SDDP-Lab-Manager`, que gerencia a execução de simulações em paralelo, `https://github.com/rjmalves/sddp-lab-manager`.

O primeiro pacote está disponível diretamente para instalação em Julia, e registrado no repositório oficial de pacotes. O segundo pacote foi desenvolvido por Bernardo Costa, um dos autores deste relatório, em conjunto com Vincent Leclère, e está disponível em código aberto para instalação a partir do repositório. Os três últimos pacotes estão disponíveis em repositórios públicos, porém não se encontram registrados no repositório oficial de pacotes. Estes são desenvolvidos pela equipe do ONS.

O ambiente para realização das execuções foi virtualizado em servidores internos do ONS. Foi utilizada uma instância *c6i* com processador Intel Xeon Gold 6248R de 3.00GHz e 16 núcleos com 128GB de RAM. A execução dos casos foi limitada a 4 simultâneas, cada uma monoprocessada, mantendo o patamar de uso da infraestrutura longe de sua capacidade máxima. O solver utilizado foi o CPLEX v22.1.1 [IBM22].

9.1 Contribuições para pacotes públicos

O pacote `SDDP.jl` é amplamente utilizado pela comunidade de otimização estocástica, e já é bastante maduro, com boa parte das funcionalidades estável desde 2019. Entretanto, ao realizar testes de comparação com os algoritmos implementados no pacote `DualSDDP.jl`, identificamos que o pacote `SDDP.jl` calcula de forma diferente a cota inferior no caso em que o problema de otimização possui incerteza no primeiro estágio, pois não considerava uma medida de risco dos custos do primeiro estágio. Mesmo que esta diferença não levasse a um cálculo diferente das funções de custo futuro, as cotas inferiores obtidas não eram compatíveis com os problemas em que haveria incerteza no primeiro estágio e uma medida de risco diferente da média. Isto levou à criação de um pedido de aprimoramento no pacote `SDDP.jl`, cuja discussão pode ser vista em `https://github.com/odow/SDDP.jl/issues/803`, que foi aceito e implementado, conforme `https://github.com/odow/SDDP.jl/pull/804`.

9.2 Desenvolvimento de pacotes públicos

O pacote `DualSDDP.jl`, disponível em código aberto para instalação a partir do seu repositório no GitHub, e desenvolvido por um dos autores deste relatório, teve diversas melhorias devido às necessidades do projeto. Dentre as mais relevantes, podemos mencionar:

- Construtores adicionais para a representação de problemas de otimização estocástica multi-estágio, para facilitar a criação de problemas típicos do planejamento da operação de sistemas hidrotérmicos, `https://github.com/bfpc/DualSDDP.jl/commit/23e799e`;
- Adição da possibilidade de utilizar fases *backward* nos algoritmos primais e duais, para melhorar a convergência dos algoritmos, especialmente em problemas com muitos estágios, `https://github.com/bfpc/DualSDDP.jl/commit/9d68097`;
- Adição de uma função para exportar a política obtida (cortes para todos os algoritmos, vértices no caso dos algoritmos de aproximação interior) em um formato muito similar ao do pacote `SDDP.jl`, para facilitar a comparação entre os algoritmos, `https://github.com/bfpc/DualSDDP.jl/commit/8682bdf`;
- Generalização do cálculo das cotas no caso do algoritmo de programação dinâmica dual estocástica *problem-child* para levar em conta as medidas de risco do primeiro estágio, `https://github.com/bfpc/DualSDDP.jl/commit/203cc79`;
- Melhoria geral da documentação do código e de utilização do pacote.

10 Resultados

10.1 Descrição dos casos de estudo

Para este relatório, realizamos diferentes simulações computacionais para avaliar o desempenho dos algoritmos de programação dinâmica dual estocástica primal, PDDE dual e *problem-child* em problemas de otimização estocástica multi-estágio referentes ao planejamento da operação de sistemas hidrotérmicos. Incluímos duas séries de casos:

1. Um sistema simples para fins ilustrativos e de validação dos algoritmos, referido como **1d_toy**: um reservatório equivalente de energia, duas usinas térmicas, e vazão *stagewise independent* estágio a estágio, com 10 aberturas simuladas a partir de uma única distribuição;
2. Um sistema referido como **1d_sin**: também com um reservatório de energia, porém com dados em escala comparável com o sistema elétrico brasileiro, um total de 144 usinas térmicas e vazão novamente *stagewise independent*, porém simuladas das distribuições sazonais, com 10 aberturas para cada estágio;
3. Um sistema mais complexo, inspirado da operação do sistema elétrico brasileiro, referido como **4ree**: 4 reservatórios equivalentes de energia, 126 usinas térmicas, 4 submercados interconectados por uma rede de transmissão e vazão *stagewise independent* simulada de distribuições sazonais;
4. Um sistema referido como **1d_sin_ar**: com configuração idêntica ao **1d_sin**, exceto pelo modelo de geração de cenários de afluência, que passa a ser um autorregressivo periódico PAR de ordem 2, considerando a afluência como variável de estado, e 10 aberturas por estágio correspondentes à inovação do processo PAR.

Em todas as séries, calculamos a política neutra ao risco, além de uma ou mais políticas com diferentes níveis de aversão ao risco.

10.2 Algoritmos utilizados

Como indicado, temos diferentes algoritmos para resolver o problema de otimização estocástica multi-estágio correspondente ao planejamento da operação. Estes algoritmos estão implementados em dois pacotes diferentes, `SDDP.jl` e `DualSDDP.jl`. O primeiro é mais antigo e mais maduro, com diferentes extensões para a formulação de problemas de otimização estocástica, e uma interface simples através da biblioteca `JuMP.jl`, mas implementa apenas o algoritmo de PDDE primal. O segundo utiliza uma formulação restrita a problemas de horizonte finito, com uma interface matricial para a formulação dos problemas de cada estágio, e implementa todos os algoritmos de PDDE: primal, aproximação interior, *problem-child* e dual. Indicamos a seguir as principais características de cada algoritmo, assim como uma abreviação para facilitar a referência futura:

- **Primal via SDDP.jl**: algoritmo clássico de PDDE [PP91], implementado na biblioteca `SDDP.jl`; constrói aproximações inferiores das funções de custo futuro a partir de cortes, e retorna uma *cota inferior* para o problema.
- **Programação dinâmica via SDDP.jl**: algoritmo de programação dinâmica para aproximações interiores seguindo o método descrito em [PdMF13], utilizando as trajetórias primais calculadas pelo algoritmo **Primal via SDDP.jl**; implementado em uma função `inner-dp.jl` desenvolvida para este projeto, e compatível com a interface *single-cut* do pacote `SDDP.jl`.
- **Problem-child via DualSDDP.jl**: algoritmo de programação dinâmica determinístico, similar ao descrito em [BDZ17], porém implementado em uma versão *single-cut* para ser mais próximo aos outros algoritmos; constrói aproximações inferiores e superiores das funções de custo futuro, a partir de cortes e vértices, respectivamente, e com isto produz *cotas inferiores e superiores* para o problema.
- **Dual via DualSDDP.jl**: algoritmo de PDDE dual com aversão ao risco, descrito em [FL23a], implementado na biblioteca `DualSDDP.jl`; constrói aproximações superiores das funções de custo futuro a partir de cortes para a conjugada de Fenchel, e retorna uma *cota superior* para o problema.

Observe que a implementação *single-cut* do algoritmo **Problem-Child**, mesmo que não seja o algoritmo originalmente proposto, também produz funções de custo futuro do mesmo tipo que todos os outros algoritmos, e daí estas também serão mais fáceis de comparar.

10.3 Simulações realizadas

Cada caso de estudo foi submetido aos algoritmos descritos anteriormente para que fossem calculadas as suas *cotas inferiores* e *cotas superiores*, produzindo as *políticas operativas* correspondentes. A política de cada algoritmo foi submetida a uma simulação com 2 000 cenários *out-of-sample* para fins de comparação, monitorando uma série de variáveis de interesse: afluência, geração hidroelétrica, armazenamento, geração termoelétrica, déficit e distância da cobertura de vértices, sendo esta última exclusiva das políticas interiores.

Retomando a discussão na seção 5.2, uma política interior definida apenas com base num conjunto de vértices é **indefinida** fora da envoltória convexa destes vértices. A solução para este

problema foi a introdução de uma relaxação da representação original que penaliza o acesso à aproximação interior fora da envoltória convexa de um fator α, a constante de Lipschitz. A distância de cobertura de vértices ilustra esta penalização; de fato, corresponde ao termo $\sum_{i=1}^{n} |\xi_i|$ na Equação (5.4)

10.4 Modelo 1d *toy*

Este é um modelo simples, cujo objetivo é ilustrar o funcionamento dos algoritmos de programação dinâmica dual estocástica primal, dual e *problem-child* em um problema de otimização estocástica multi-estágio referente ao planejamento da operação de sistemas hidrotérmicos.

Começamos apresentando nas figuras 10.1, 10.2 e 10.3 a evolução das cotas superior e inferior ao longo das iterações, para observar a convergência dos algoritmos. Os casos correspondem, respectivamente, a horizontes de 1, 3 e 10 anos de planejamento, e a aversão ao risco é dada por um peso de 50% para o CVAR no quantil 50% (denotado por CVAR_{50}), sendo complementado com 50% de peso para a média. Em todos eles, podemos observar que as cotas superior e inferior convergem para um valor comum, indicando que o algoritmo convergiu para a solução ótima, o que é esperado visto que se trata de um problema simples. Note que a escala do eixo horizontal é logarítmica, para facilitar a visualização da convergência ao longo das primeiras iterações, e que os valores das 10 primeiras foram desconsiderados para não distorcer o eixo vertical.

Ainda assim, podemos já observar uma diferença entre o método de cota superior via programação dinâmica com uma etapa *backward* final (indicado como *DP UB*, para *dynamic programming upper bound*) nas legendas e os demais métodos para o cálculo de cotas superiores, seja o algoritmo via PDDE dual (indicado por *Dual UB*) ou *problem-child* (indicado por *PC UB*). De fato, por realizar uma única etapa *backward*, com a construção de todos os vértices de uma única vez, este método se mostrou o mais rápido do ponto de vista da convergência em número de iterações, pois ele utiliza a informação de todos os estados de forma conjunta. Por outro lado, os outros dois métodos de cota superior realizam uma etapa *backward* para cada estado amostrado, o que pode levar a uma convergência mais lenta, mas que é compensada pela garantia de convergência para a solução ótima.

Vemos, também, que os três métodos correspondem também a algoritmos que calculam cotas inferiores. Tanto o algoritmo de programação dinâmica com uma única etapa *backward* e a PDDE dual constróem as cotas inferiores via o algoritmo de PDDE usual, e são, muitas vezes, bastante similares. O algoritmo *problem-child* tem a construção da etapa *forward* diferente, e por isso poderia apresentar maiores diferenças quanto à convergência, mas que não vemos nestes gráficos.

Para efeito de comparação, também incluímos os resultados de convergência para o caso *neutro ao risco* com 10 anos de horizonte de planejamento. Podemos observar que a convergência das cotas inferiores é mais lenta do que a convergência da cota superior via programação dinâmica, mas que os métodos de cota superior via PDDE dual e *problem-child* convergem de forma ainda mais lenta.

Apresentamos nas figuras 10.5 e 10.6 os resultados da simulação das políticas correspondentes.

Podemos observar que as políticas obtidas são bastante similares, com a política de 3 anos de horizonte de tempo sendo mais suave, o que é esperado devido ao maior horizonte de tempo. Também notamos pela observação da distância da cobertura de vértices que nenhum dos métodos de aproximação interior foi penalizado por explorar um estado de armazenamento fora da envoltória convexa dos estados para os quais foram construídos os vértices das políticas correspondentes. Isto é esperado, já que a baixa dimensão do problema torna bastante fácil uma exploração suficiente do espaço de estados.

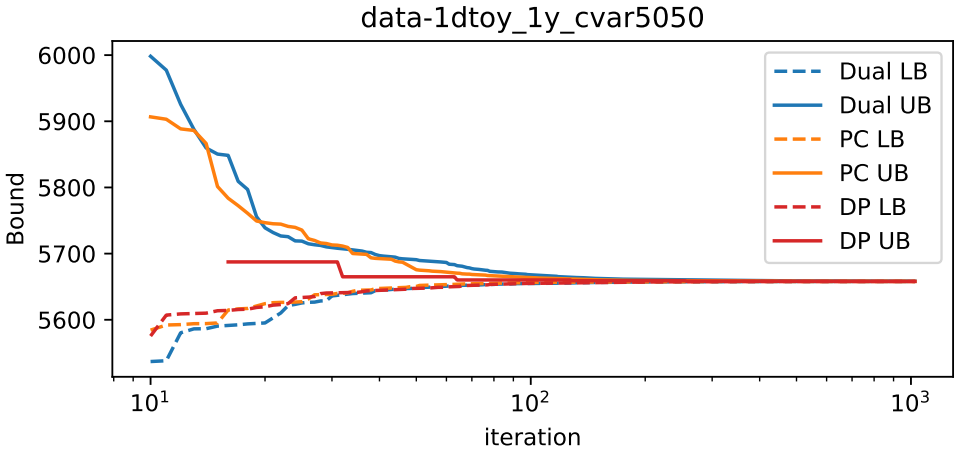


Figura 10.1: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 1 ano de horizonte de tempo.

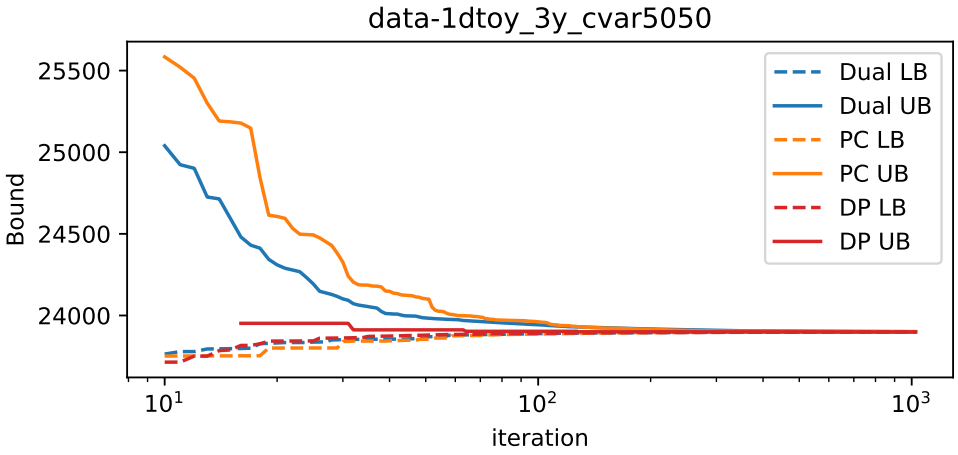


Figura 10.2: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 3 anos de horizonte de tempo.

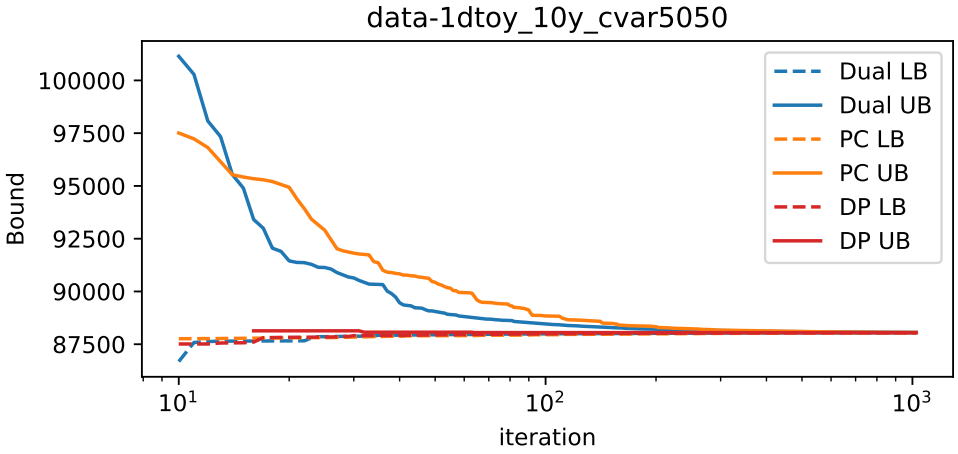


Figura 10.3: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 10 anos de horizonte de tempo.

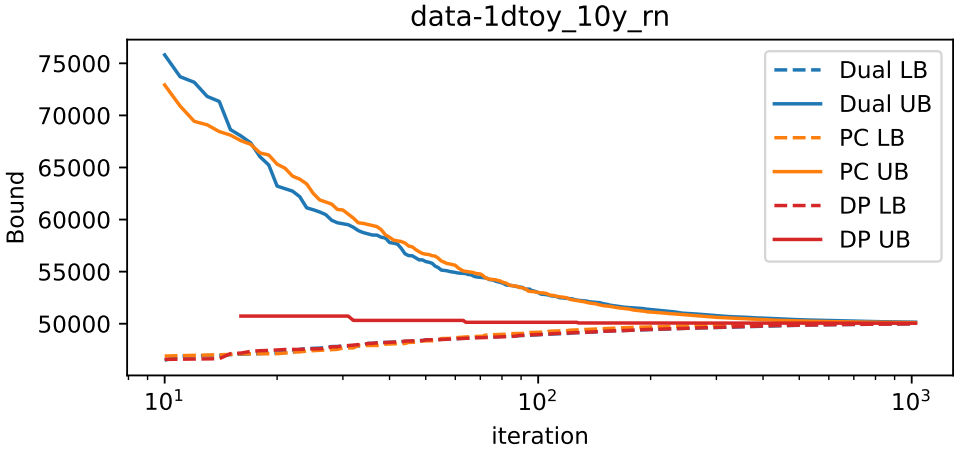


Figura 10.4: Evolução das cotas superior e inferior ao longo das iterações, no caso neutro ao risco, para 10 anos de horizonte de tempo.

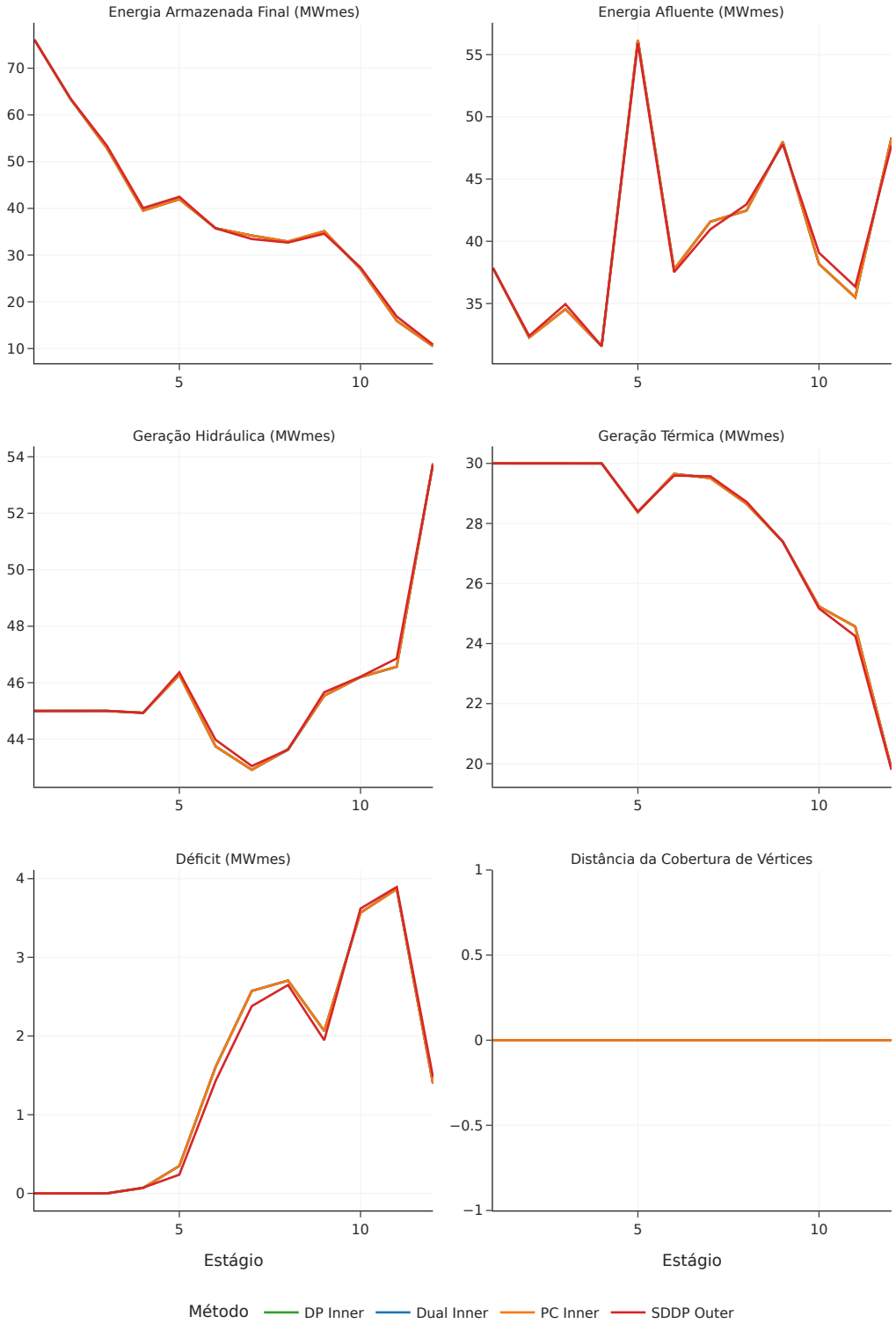


Figura 10.5: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco avesso (CVAR_{50} com peso 50%) para 1 ano de horizonte de tempo.

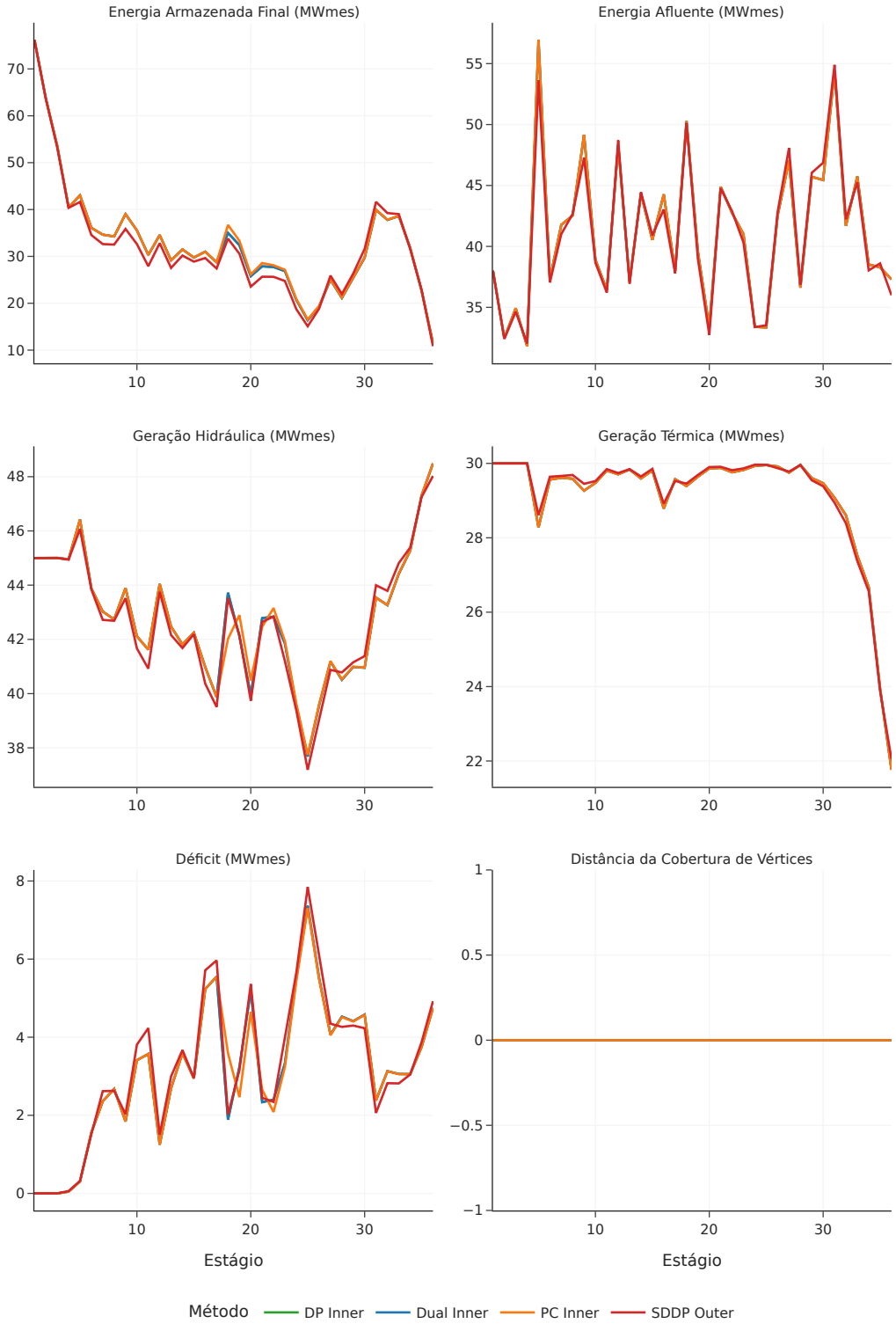


Figura 10.6: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco avesso ($CVAR_{50}$ com peso 50%) para 3 anos de horizonte de tempo.

10.5 Modelo de 1 REE

Este modelo já é um pouco mais realista, pois, apesar de possuir apenas uma variável de estado, tem mais variáveis de decisão para a geração térmica, o que resulta em funções de custo futuro mais detalhadas.

Apresentamos nas figuras 10.7, 10.8 e 10.9 a evolução das cotas superior e inferior ao longo das iterações, para observar a convergência dos algoritmos. Os casos correspondem, respectivamente, a horizontes de 1, 3 e 10 anos de planejamento, e a aversão ao risco é dada por um peso de 50% para o CVAR no quantil 50% (denotado por CVAR_{50}), sendo complementado com 50% de peso para a média. Novamente, observamos que as cotas superiores e inferiores convergem para um valor comum, indicando que todos os algoritmos convergem para a solução ótima. Como na seção anterior, utilizamos uma escala logarítmica para o eixo horizontal, e desconsideramos os valores das 10 primeiras iterações.

Aqui, no problema mais simples, notamos que os métodos parecem estar mais próximos do que estavam no caso **1d_toy**, mas conforme o horizonte de planejamento aumenta, o método de cota superior via programação dinâmica com uma etapa *backward* final (indicado como *DP UB*, para *dynamic programming upper bound*) se torna progressivamente melhor com relação aos outros dois, pela mesma razão.

Apresentamos nas figuras 10.10, 10.11 e 10.12 os resultados da simulação das políticas correspondentes a um planejamento de 10 anos, para o caso com aversão ao risco, um caso com CVAR 50% com peso de 50%, e um caso com bastante aversão ao risco, onde consideramos um peso também de 50% mas agora para apenas os 10% cenários mais caros. Neles, vemos que a operação da política habitual, dada pelos cortes da PDDE, e indicada por *SDDP Outer*, é mais suave na geração térmica, e também é a menos conservadora de todas, mantendo menores energias armazenadas no reservatório. Estas diferenças ficam mais exacerbadas conforme a aversão ao risco aumenta, pois no caso neutro ao risco, mesmo havendo uma diferença visível na geração térmica, esta é bem pequena e não altera de forma significativa as energias armazenadas e a geração hidroelétrica, que está numa escala da ordem de 10 vezes maior. Entretanto, ao considerar a aversão ao risco, as políticas calculadas através das funções de custo futuro interiores, que são cotas *superiores* da função de custo futuro (em vez de serem cotas inferiores, como no caso dos cortes da PDDE) se mostram mais conservadoras, o que é esperado.

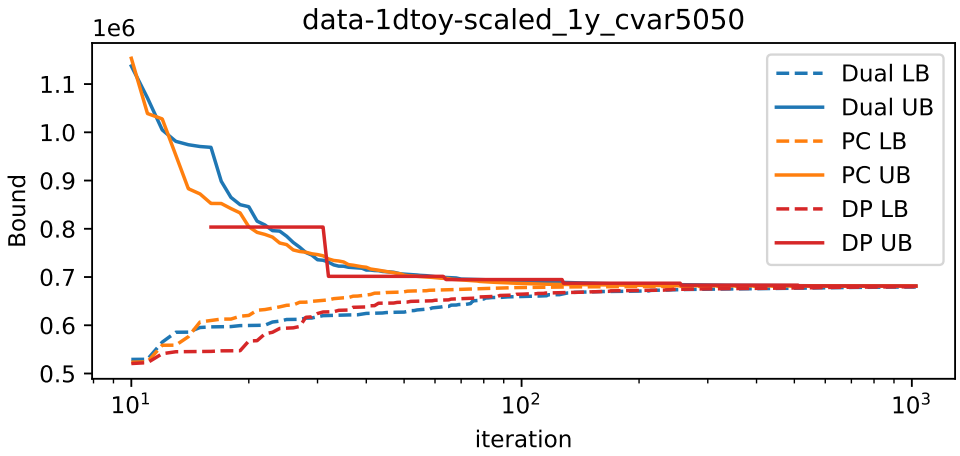


Figura 10.7: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 1 ano de horizonte de tempo.

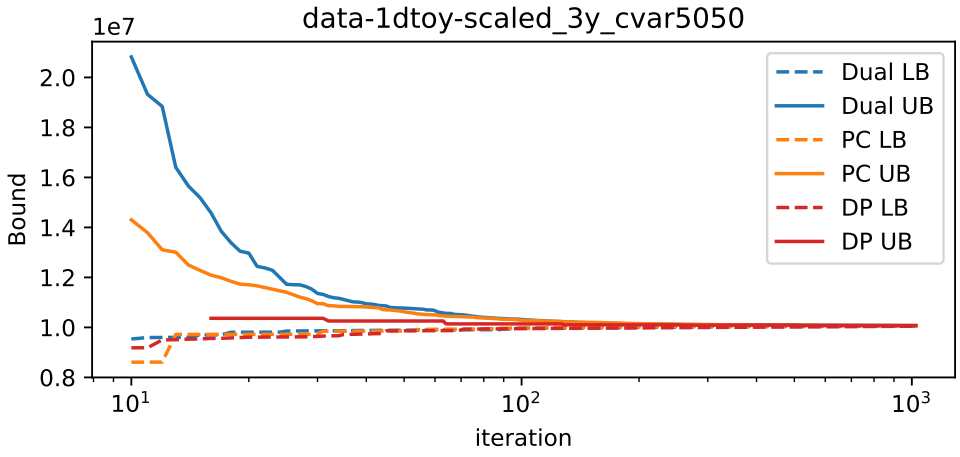


Figura 10.8: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 3 anos de horizonte de tempo.

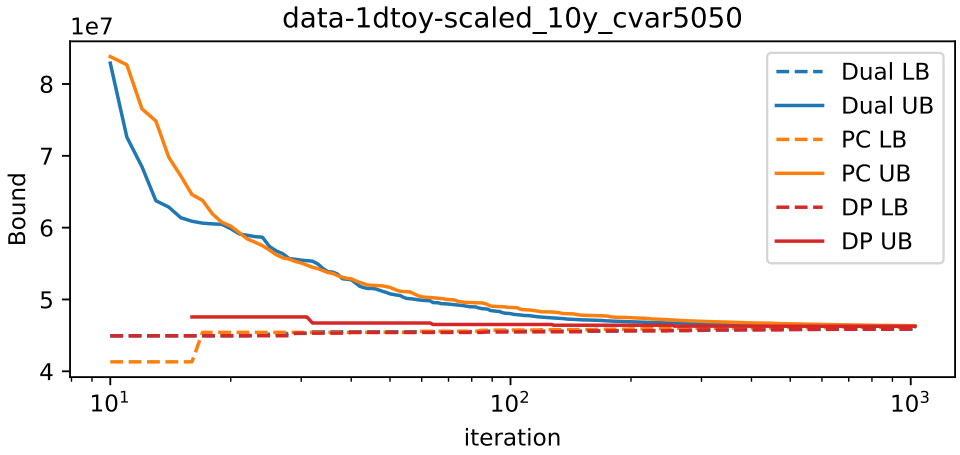


Figura 10.9: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 10 anos de horizonte de tempo.

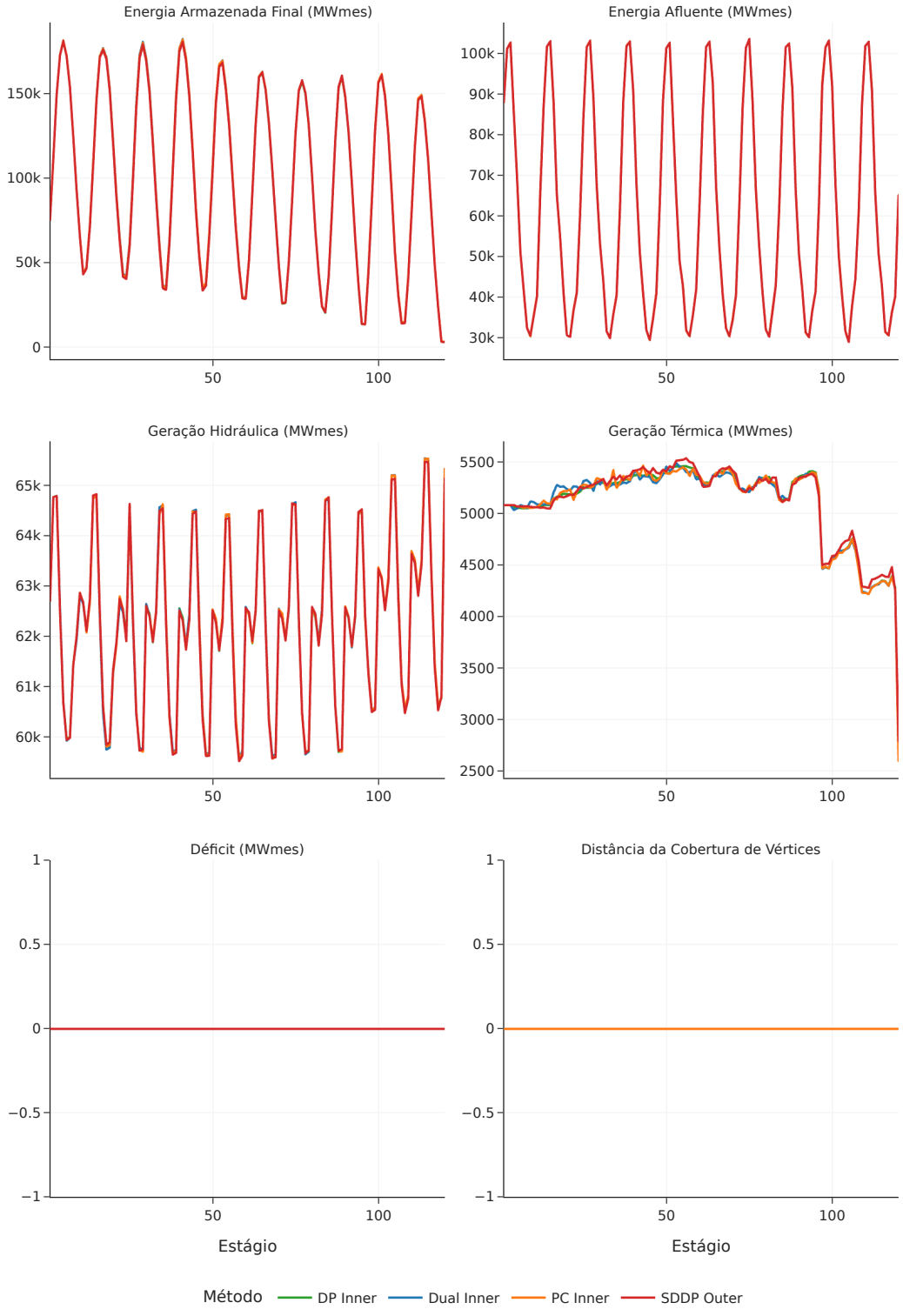


Figura 10.10: Simulação das políticas exterior (cortes) e interior (vértices) no caso neutro ao risco para 10 anos de horizonte de tempo.

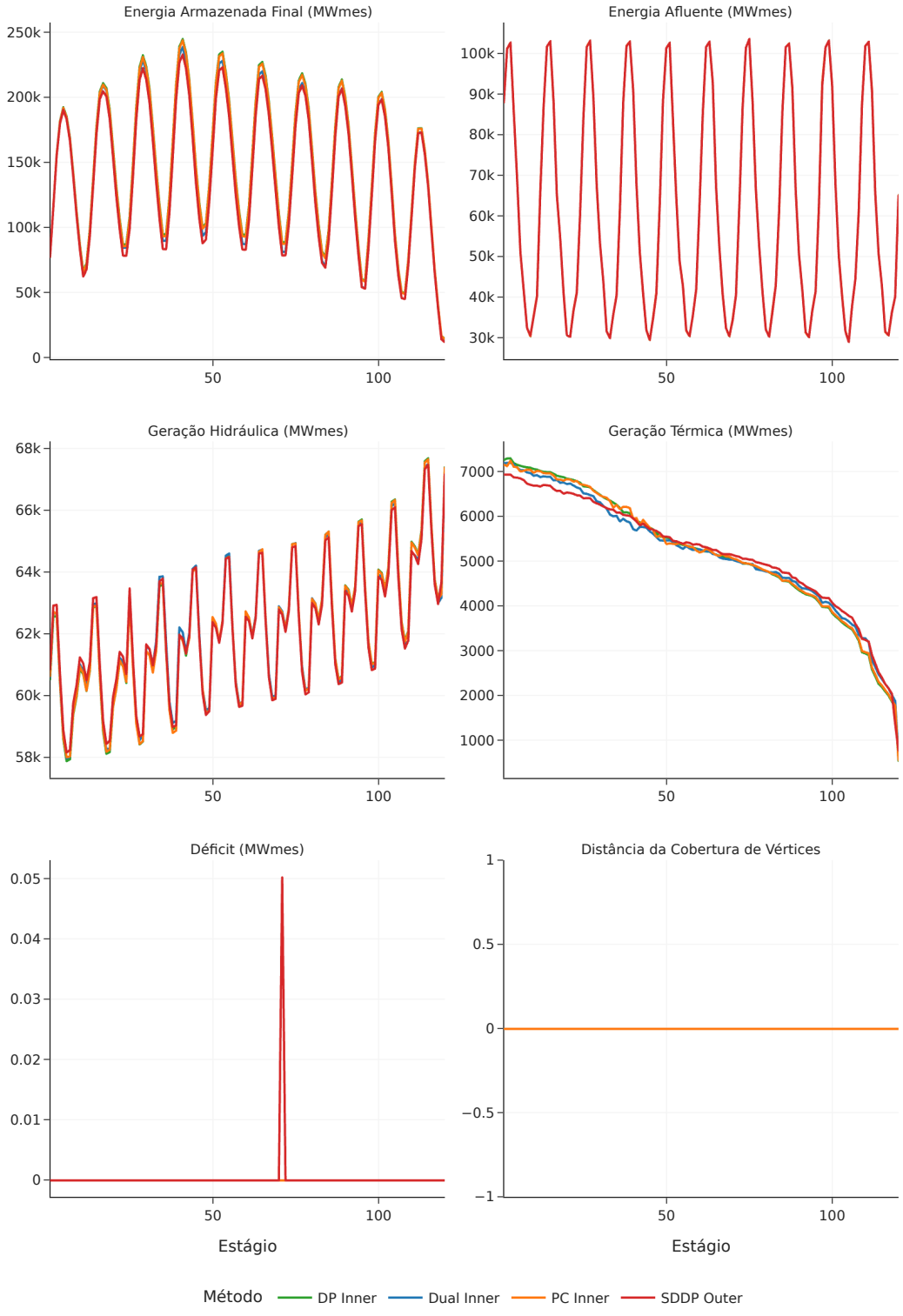


Figura 10.11: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco avesso ($CVAR_{50}$ com peso 50%) para 10 anos de horizonte de tempo.

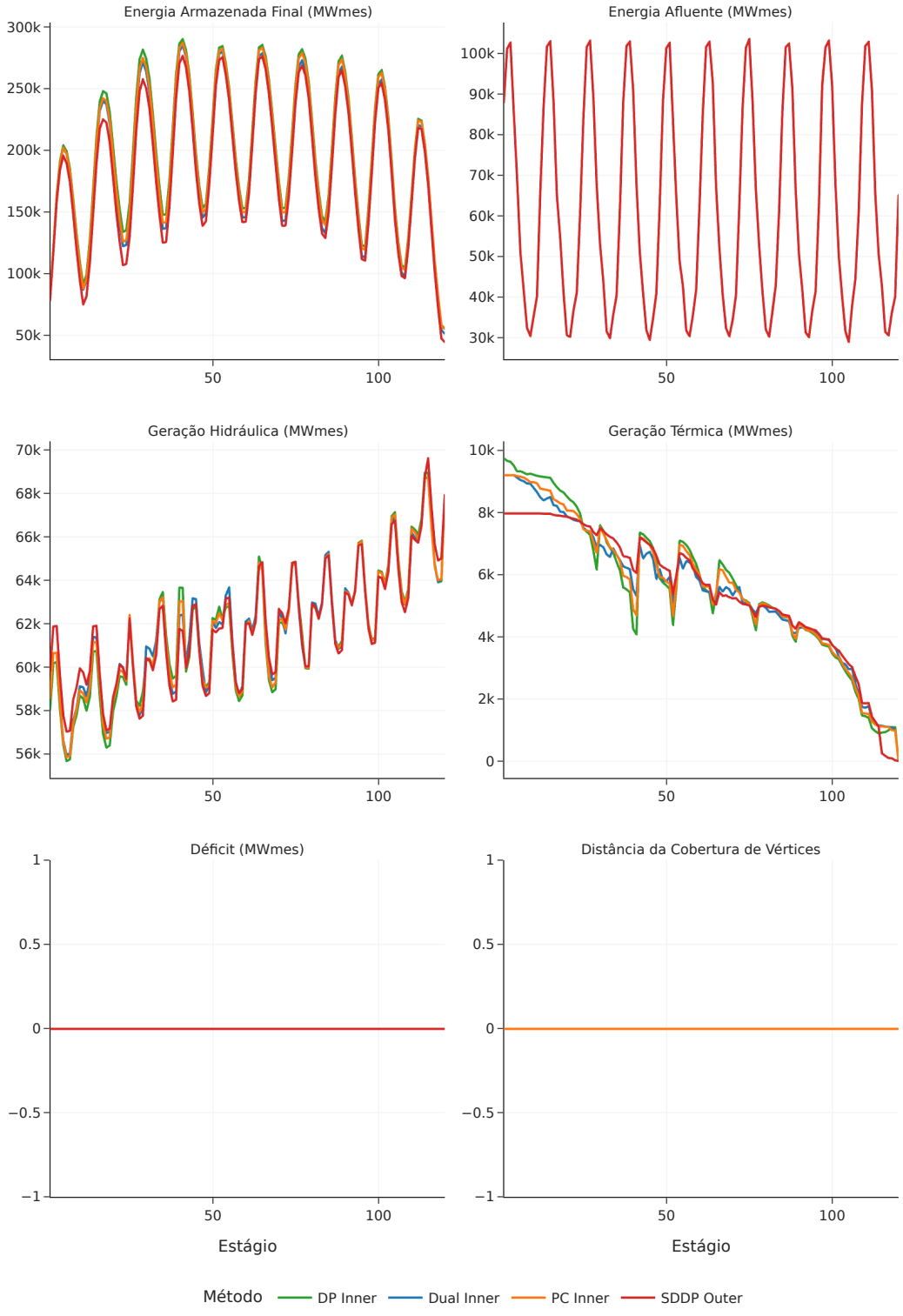


Figura 10.12: Simulação das políticas exterior (cortes) e interior (vértices) no caso bastante risco avesso ($CVAR_{10}$ com peso 50%) para 10 anos de horizonte de tempo.

10.6 Modelo de 4 REE

Este modelo corresponde a um caso clássico de planejamento da operação, com 4 reservatórios equivalentes de energia, cada um em um submercado, interligados por uma rede de transmissão elétrica, além de uma grande quantidade de usinas termoelétricas.

Com o aumento do espaço de estados, teremos uma maior dificuldade para a convergência dos algoritmos, seja para as cotas inferiores como para as cotas superiores. Estes resultados podem ser vistos nas figuras 10.13, 10.14 e 10.15, onde observamos a evolução das cotas superior e inferior ao longo das iterações, em casos de 1, 3 e 10 anos com aversão ao risco. Inclusive, para melhor observarmos a convergência, também alteramos a escala das cotas obtidas para que sejam *logarítmicas*, evitando assim a distorção que ocorre devido às cotas superiores das iterações iniciais, que são muito maiores que as cotas inferiores e do que o valor final, o que distorce a visualização. Aqui, já podemos ver que o algoritmo *problem-child* apresenta um pouco mais de dificuldades para chegar à cota superior, se comparado com o método Dual ou com o método de programação dinâmica com uma única etapa *backward* final. Também vemos que, de forma geral, o algoritmo dual converge de forma mais lenta do que o algoritmo de programação dinâmica, mas que ao final de 1024 iterações (cada uma delas realizando um único corte, no caso dual, ou vértice, no caso da programação dinâmica), ambas estão bastante próximas, mesmo que ao longo do processo o algoritmo dual tenha calculado cotas superiores sistematicamente maiores.

As cotas inferiores, aparentemente, apresentam muito menos variação de acordo com o método utilizado, sem uma tendência clara para um método (PDDE ou *problem-child*) ser mais ou menos eficiente que o outro.

Enfim, as figuras 10.16, 10.17 e 10.18 ilustram os resultados obtidos para o modelo de 4 REE, com 10 anos de horizonte de tempo, nos casos neutro ao risco, avesso ao risco (CVAR_{50} com peso 50%) e muito avesso ao risco (CVAR_{10} com peso 50%), respectivamente.

Novamente, observamos que as políticas obtidas são bastante similares para cada caso, com um comportamento esperado de compromisso entre geração térmica e hidroelétrica de acordo com a aversão ao risco correspondente. Uma diferença esperada é que o custo futuro simulado das políticas interiores, que é dado por estimativas *superiores* das funções de custo futuro, é de fato maior do que o calculado pela política dos cortes da PDDE (indicada novamente por *SDDP Outer*). A política de programação dinâmica é em seguida a mais próxima neste quesito, e podemos também notar que, conforme a aversão ao risco aumenta, estas diferenças vão diminuindo em termos relativos.

Uma maior diferença é o aumento do déficit gerado pela política *problem-child*, que se revela muito mais conservadora, especialmente conforme a aversão ao risco aumenta. Isto é devido ao algoritmo *problem-child* ainda estar com uma função de custo futuro mais superestimada com relação aos outros métodos de cotas superiores, como vimos nos gráficos de convergência, onde este algoritmo não obtém uma cota superior tão boa quanto os outros nos casos com aversão ao risco. Portanto, obtemos uma política ainda mais conservadora, que podemos ver nos gráficos de energia armazenada, e um consequente aumento, também, no vertimento observado.

Além disso, nos dois casos avessos ao risco, a geração térmica da política *dual* parece ser levemente menor do que as outras. Isto não é, necessariamente, uma indicação de uma melhor política, já que o critério é o *risco* da política, e não apenas o seu custo em média. Entretanto, como a política dual (como todas as políticas interiores) tem o seu risco controlado pela cota superior obtida, isto pode ser um sinal que a política dada pelos cortes da PDDE também não está ótima.

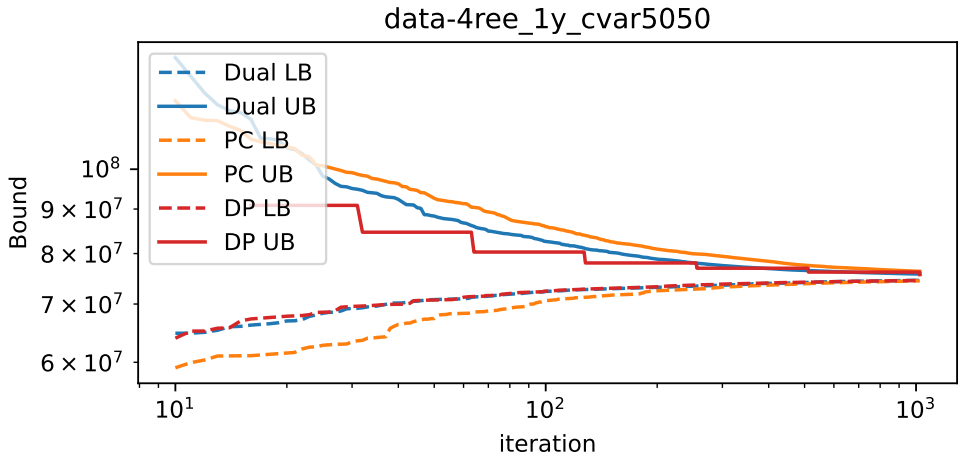


Figura 10.13: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 1 ano de horizonte de tempo.

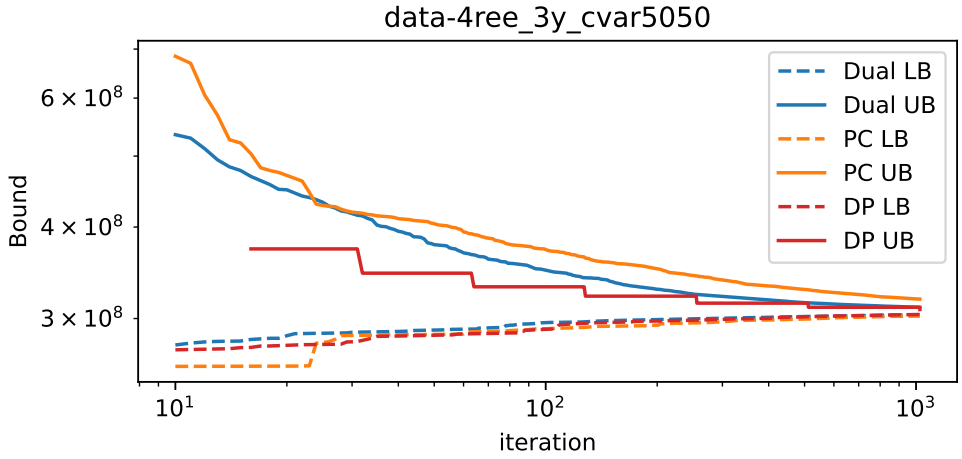


Figura 10.14: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 3 anos de horizonte de tempo.

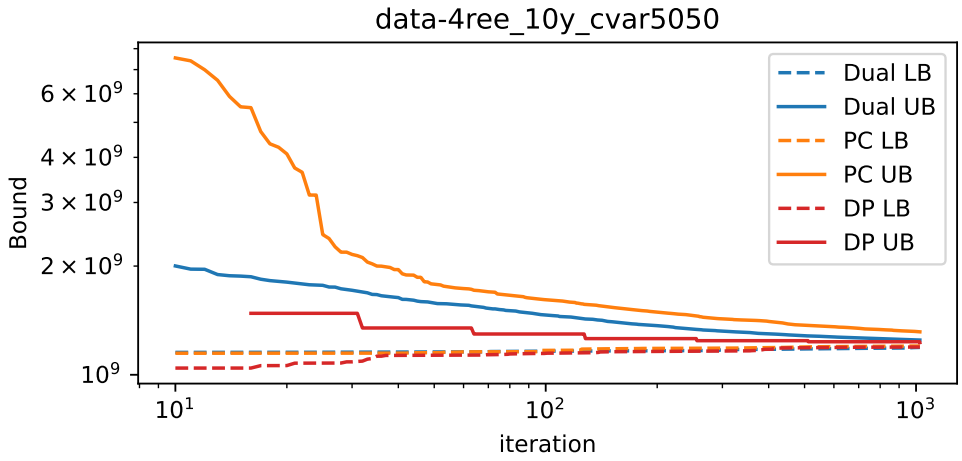


Figura 10.15: Evolução das cotas superior e inferior ao longo das iterações, no caso avesso ao risco, para 10 anos de horizonte de tempo.

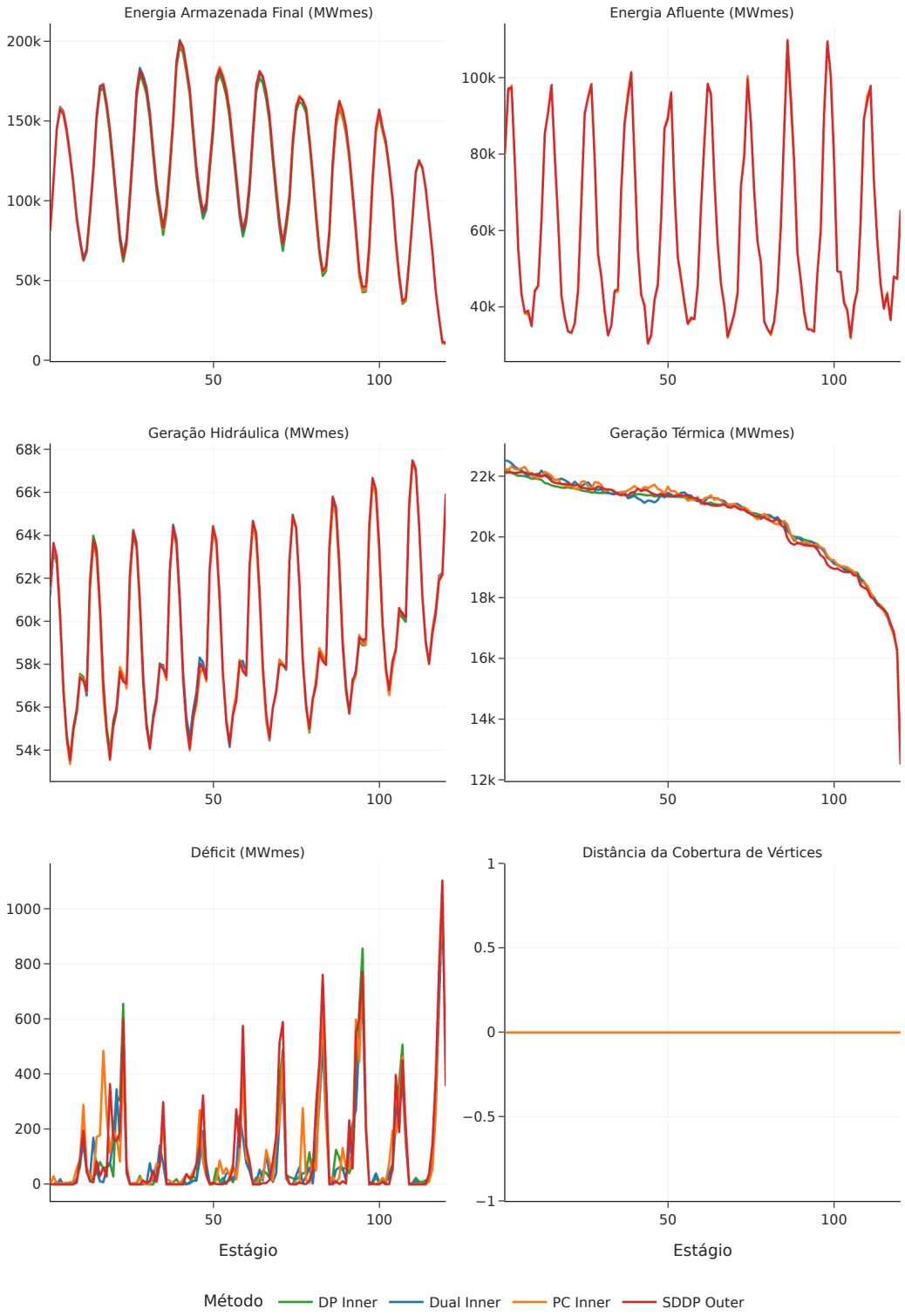


Figura 10.16: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco neutro.

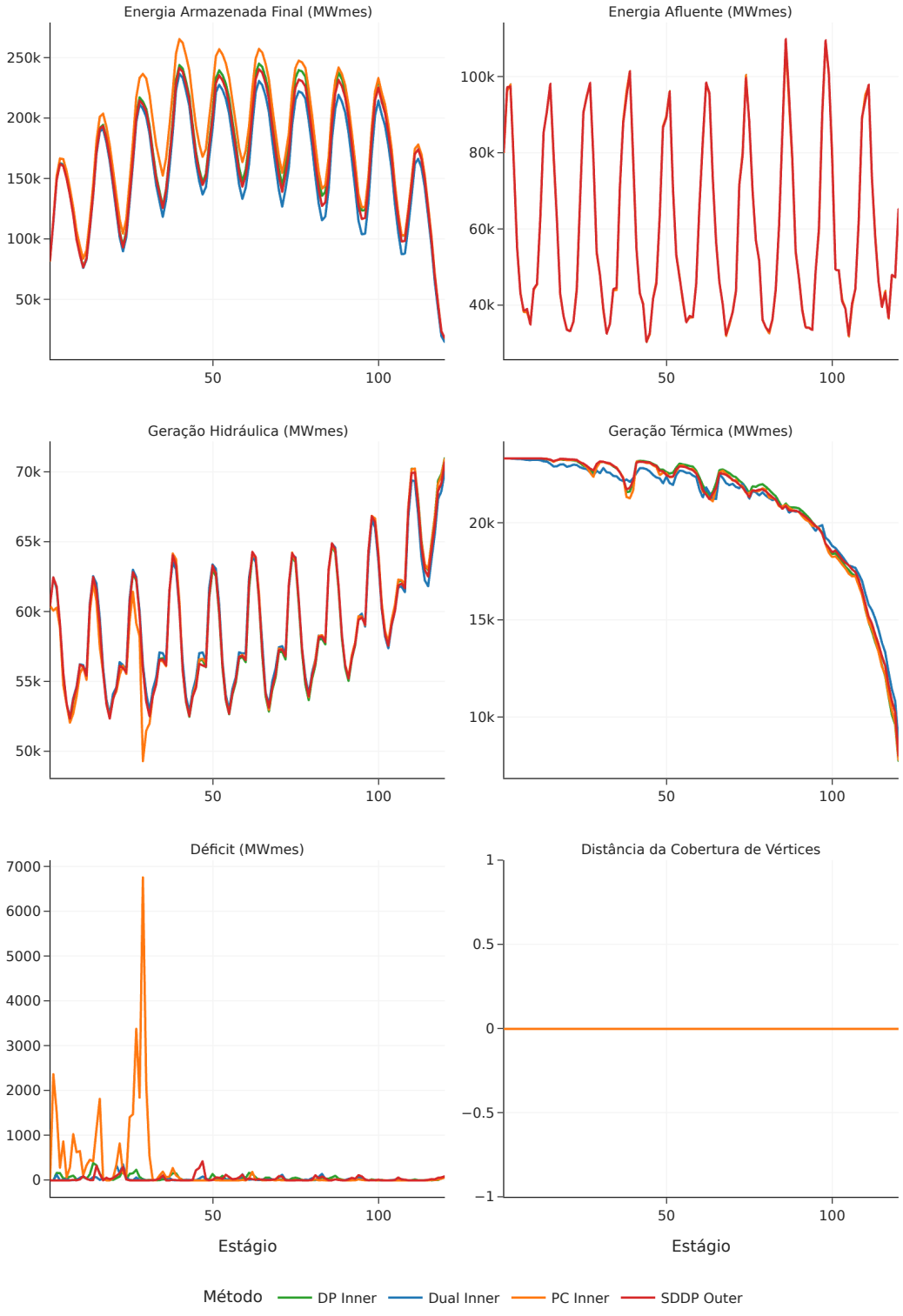


Figura 10.17: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco avesso ($CVAR_{50}$ com peso 50%).

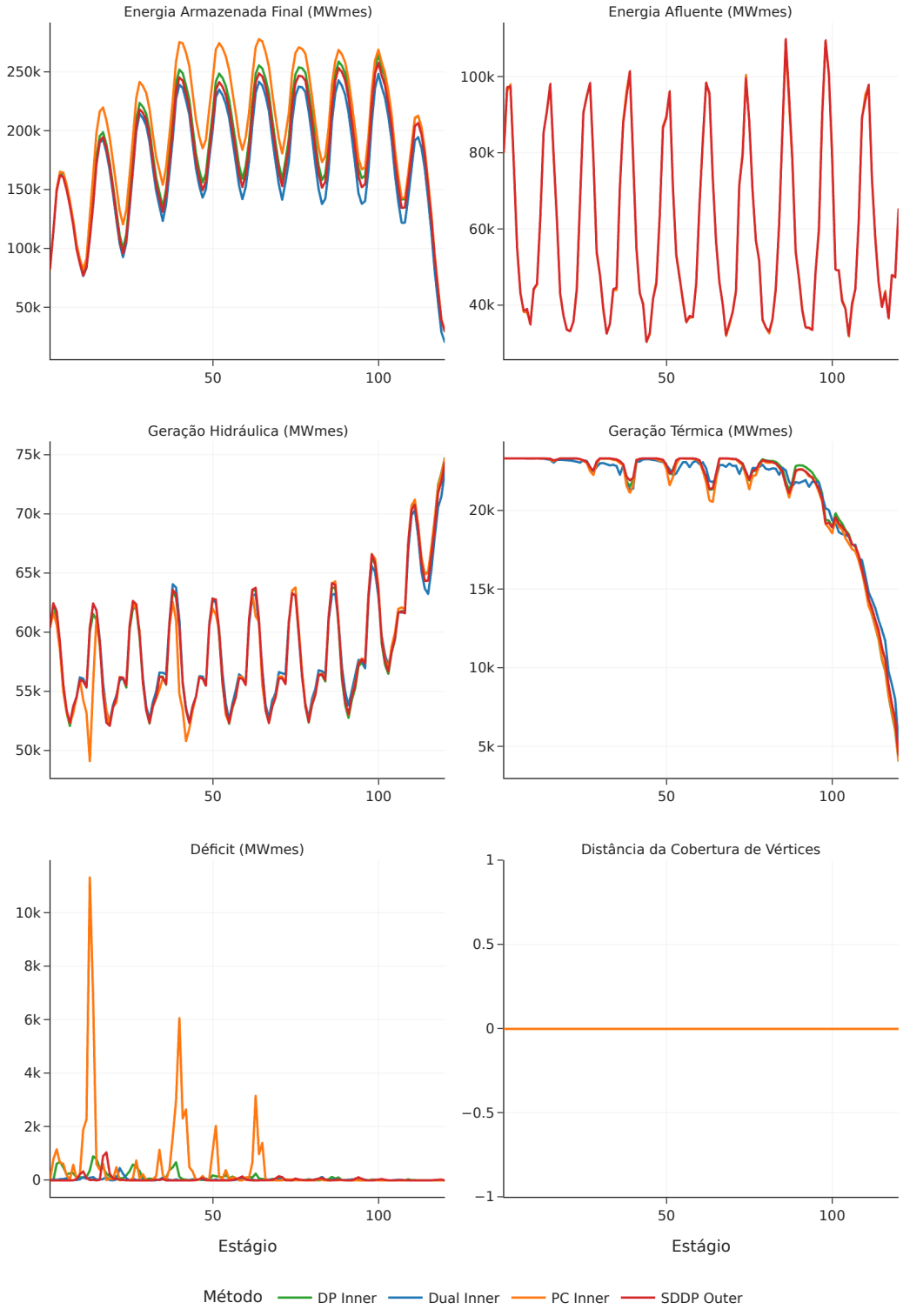


Figura 10.18: Simulação das políticas exterior (cortes) e interior (vértices) no caso muito risco avesso ($CVAR_{10}$ com peso 50%).

10.7 Modelo de 1 REE com afluências autorregressivas

Este modelo tem uma configuração igual ao modelo de 1 REE da seção 10.5, com a única diferença que o processo de afluências não é mais *stagewise independent*, mas segue um processo autorregressivo, periódico, com dois *lags*. Esta mudança do processo aumenta o espaço de estados para a função de custo futuro, que deve agora levar em consideração a tendência hidrológica de afluências. Mais ainda, estas variáveis de estado são, de certa forma, exógenas, pois não possuem uma variável de controle associada: de fato, elas servem apenas para representar a incerteza que afeta o sistema, e que é mais complexa do que a incerteza representada pela amostragem de cenários históricos de afluências.

Para este caso, utilizamos apenas o algoritmo de PDDE primal, para o cálculo de cotas inferiores, e a programação dinâmica interior através de uma etapa *backward* final para o cálculo de cotas superiores, que já havia se mostrado bastante eficiente nos outros três casos considerados.

A figura 10.19 ilustra os resultados da simulação das políticas correspondentes a um planejamento de 10 anos, para o caso com CVAR 50% com peso de 50%. A operação da política habitual, dada pelos cortes da PDDE, e indicada por *SDDP Outer*, é mais adequada do que a política calculada através das aproximações interiores. Uma das razões mais importantes para isto pode ser observada na curva de distância da cobertura de vértices. Isto se deve ao fato de o estado de tendência hidrológica não ser controlável, e portanto qualquer estado que não esteja na envoltória convexa gera uma penalização muito grande na função de custo futuro. Para reduzir tal penalização, a política pode apenas controlar o armazenamento (se necessário, através de déficit e vertimento), o que leva a uma política que gera tanto mais déficit como mais vertimento.

Estes resultados indicam que tanto o cálculo de cotas superiores quanto a política interior resultante necessitam de aprimoramentos para lidar com o caso de variáveis de estado suplementares, como no caso da modelagem de afluências autorregressivas. De fato, apesar de a dimensão do problema ser apenas três, e portanto menor do que o caso de 4 dimensões do modelo de 4 REE, o fato de as variáveis de estado auxiliares não serem controláveis gera uma dificuldade muito maior para o modelo, que resulta em aproximações também menos eficientes.

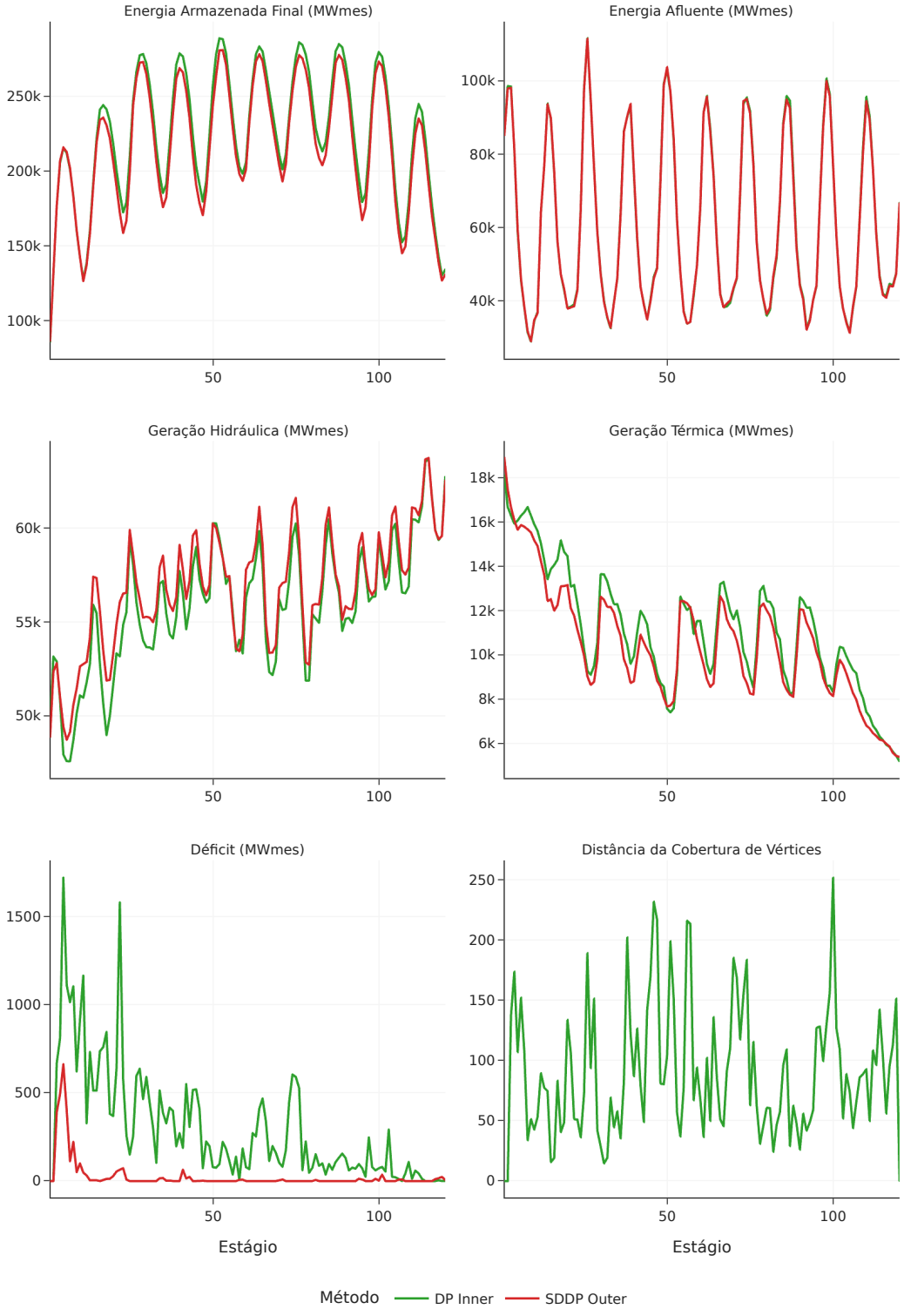


Figura 10.19: Simulação das políticas exterior (cortes) e interior (vértices) no caso risco avesso ($CVAR_{50}$ com peso 50%) para 10 anos de horizonte de tempo com modelo autorregressivo de afluências.

Parte IV
Conclusão

11 Desenvolvimentos realizados

O foco principal dos estudos desenvolvidos durante este projeto consiste em obter cotas superiores para problemas de otimização estocástica, que também permitam estabelecer um critério de parada para o algoritmo de otimização utilizado para obter as funções de custo futuro para o planejamento da operação do setor elétrico. Para tanto, e levando em conta que o algoritmo atualmente utilizado, PDDE, fornece um limite inferior para as funções de custo futuro, exploramos algoritmos para a construção de cotas superiores para problemas de otimização estocástica multi-estágio, com aversão ao risco. Vimos que as metodologias de programação dinâmica, baseadas na construção de funções de custo futuro, são capazes de calcular cotas superiores efetivas para a função objetivo, sem a necessidade de intervalos de confiança. Tanto os algoritmos primais, que constroem cotas superiores a partir de funções de custo futuro interiores, como os algoritmos duais, que constroem cotas superiores a partir de funções de custo futuro conjugadas, também fornecem uma *política*, correspondente às novas funções de custo futuro construídas, cuja performance, do ponto de vista da aversão ao risco considerada no problema, é *garantida* de ser melhor do que a cota superior obtida.

Também vimos que os métodos que utilizam amostragem de cenários, apesar de menos custosos computacionalmente, podem sofrer de divergência exponencial, dada a composição do viés da estimativa com a medida de risco. Especialmente no caso em que o problema possui um grande número de estágios, isso pode levar a estimativas de Monte Carlo pouco úteis para as cotas superiores. Por outro lado, um método recente, utilizando *importance sampling* para rebalancear o peso dos cenários, pode sofrer de um viés para baixo, o que não é interessante para a construção de cotas superiores, pois estas se revelariam otimistas.

Como contribuições práticas, desenvolvemos um protótipo computacional flexível que pode ser utilizado para diferentes estudos posteriores de problemas de PDDE, com interfaces para as bibliotecas `SDDP.jl` e `DualSDDP.jl`. Este protótipo está disponível em código aberto, e como os pacotes `SDDP.jl` e `DualSDDP.jl` também são abertos, os testes realizados neste projeto também levaram a melhorias nos pacotes, que podem ser utilizadas por outros pesquisadores e profissionais. Também desenvolvemos uma extensão para a biblioteca `SDDP.jl` que permite o cálculo de cotas superiores através da construção de funções de custo futuro por aproximações interiores.

12 Resultados obtidos

Vimos que os métodos de cálculo de cotas superiores baseados em funções de custo futuro interiores e duais são capazes de fornecer cotas superiores efetivas para o problema de otimização estocástica considerado, com garantias de performance para a política resultante.

Os métodos que utilizam as funções interiores são de mais simples implementação, e geralmente mais rápidos, por aproveitar a estrutura do problema de otimização estocástica e a decomposição dos problemas em cada estágio. Vimos, entretanto, que a escolha dos métodos também depende do problema, por exemplo da dimensão das variáveis de estado, da aversão ao risco considerada, e da complexidade do processo estocástico associado.

Também observamos que os métodos de aproximação das funções de custo futuro, sejam primais ou duais, devem realizar uma exploração do espaço de estados suficientemente abrangente para

garantir a convergência do algoritmo de otimização. Neste sentido, o algoritmo *problem-child* garante a exploração, mas pode às vezes ser mais lento para a redução do *gap* entre as cotas superiores e inferiores.

Enfim, os métodos estatísticos baseados na amostragem de cenários, apesar de menos custosos computacionalmente, ainda não se mostram plenamenente satisfatórios para o seu uso. Por um lado, os métodos baseados na construção de funções de custo futuro com variáveis auxiliares podem sofrer de divergência exponencial, dada a composição do viés da estimativa com a medida de risco, resultando em cotas superiores demasiado grandes, e por isto pouco úteis seja como critério de parada, seja como avaliação do risco da política considerada. Por outro lado, o método de *importance sampling* para rebalancear o peso dos cenários sofre de um viés sistemático para baixo, o que não é interessante para a construção de cotas superiores, pois estas se revelariam otimistas, resultando em uma parada prematura do algoritmo de otimização.

Em suma, o método que combina o cálculo das cotas inferiores pela PDDE usual, e que complementa com uma cota superior baseada em programação dinâmica, é o que possui um desempenho mais consistente ao longo dos testes realizados, com geralmente uma das mais rápidas reduções do *gap* de convergência. Assim, mesmo que seja necessário repetir o processo se o *gap* ainda estiver acima da tolerância desejada, esta metodologia também pode fornecer um critério de parada efetivo e determinístico para o algoritmo de otimização, com tempo de cálculo razoável. Mais ainda, por ser um procedimento relativamente simples, é de fácil implementação dentro do arcabouço de programação dinâmica já existente para o problema de otimização estocástica do planejamento da operação do setor elétrico.

13 Sugestões de trabalhos futuros

Uma das possibilidades mais relevantes para o desenvolvimento futuro dos algoritmos e metodologias explorados neste projeto é a aplicação de ideias de *importance sampling* no algoritmo *problem-child*. Como visto tanto na PDDE primal como na PDDE dual, esta modificação da construção de cenários da etapa *forward* é capaz de acelerar a convergência do algoritmo, e esta ideia pode ser estendida para o algoritmo *problem-child*, que com isto poderia explorar ainda mais eficientemente o espaço de estados.

Uma segunda direção, também relacionada à exploração do espaço de estados, é uma melhor construção das cotas superiores para o caso em que se utilizam processos estocásticos mais complexos, como por exemplo da família PAR-p. De fato, estes processos mais gerais são capazes de modelar com mais fidelidade as correlações temporais entre os cenários, mas também aumentam a complexidade do problema de otimização estocástica, ao aumentar a dimensão do espaço de estados. Como, no entanto, não há uma verdadeira variável de decisão associada à dinâmica do processo, a penalização necessária para o caso das aproximações interiores acaba se tornando uma barreira muito mais significativa para a convergência dos algoritmos, o que não foi observado em problemas com dimensão maior de variáveis de estado, mas todas controláveis.

Referências

[BDZ17] Regan Baucke, Anthony Downward, and Golbon Zakeri. A deterministic algorithm for solving multistage stochastic programming problems, July 2017. https://optimization-online.org/2017/07/6138/.

[DK17] Oscar Dowson and Lea Kapelevich. SDDP.jl: a Julia package for Stochastic Dual Dynamic Programming. *Optimization Online*, 2017.

[DS16] Lingquan Ding and Alexander Shapiro. Upper bound for optimal value of risk averse multistage problems, 2016.

[FL23a] Bernardo Freitas Paulo da Costa and Vincent Leclère. Dual SDDP for risk-averse multistage stochastic programs. *Operations Research Letters*, 51(3):332–337, 2023.

[FL23b] Bernardo Freitas Paulo da Costa and Vincent Leclère. Duality of upper bounds in stochastic dynamic programming, Jul 2023. https://optimization-online.org/?p=23738.

[FML24] Bernardo Freitas Paulo da Costa, Lucas Merabet, and Vincent Leclère. Policy with guaranteed risk-adjusted performance for multistage stochastic linear problems, Jan 2024. https://optimization-online.org/?p=25462.

[GLCP23] Joaquim Dias Garcia, Iago Leal, Raphael Chabar, and Mario Veiga Pereira. A multicut approach to compute upper bounds for risk-averse sddp, 2023.

[IBM22] IBM. Ibm ilog cplex 22.1.1 user's manual, 2022.

[ONS] O sistema em números. `http://www.ons.org.br/paginas/sobre-o-sin/o-sistema-em-numeros`. Accessed: 2024-06-04.

[PdMF13] Andy Philpott, Vitor de Matos, and Erlon Finardi. On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970, 2013.

[PP91] Mario VF Pereira and Leontina MVG Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical programming*, 52(1-3):359–375, 1991.

[RU00] R Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

[STdCS13] Alexander Shapiro, Wajdi Tekaya, Joari Paulo da Costa, and Murilo Pereira Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European journal of operational research*, 224(2):375–391, 2013.

[VSW69] R M Van Slyke and Roger Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, Jul 1969.