Rodrigo Tomás Nogueira Cardoso

Ferramentas para Programação Dinâmica em Malha Aberta

Tese submetida à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais como requisito parcial para obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Ricardo H. C. Takahashi

Universidade Federal de Minas Gerais
Belo Horizonte, 14 de julho de 2008.

Dedicatória

Para Cibele, minha esposa.

Agradecimentos

Ao término desta tese, e ao final destes quase 30 anos de estudo, quero agradecer a Deus pelo dom da minha vida, e pela minha vocação de mestre e doutor. Que eu possa colocar estes dons a serviço, oferecendo minha parcela de contribuição para a melhoria da sociedade.

Agradeço aos meus pais, pelo apoio, pelo incentivo; por terem me proporcionado a oportunidade de tomar gosto pelos livros e cadernos desde cedo. Agradeço aos meus irmãos, pela amizade, à minha família, pela presença.

Agradeço à minha esposa, Cibele, pois seu amor me faz uma pessoa melhor.

Agradeço ao prof. Ricardo Takahashi, pela indicação do tema e pelo acompanhamento desde o trabalho do mestrado. Agradeço pelo exemplo de respeito e paciência diante do processo pesquisa-ensino-aprendizagem. Agradeço aos professores das minhas bancas, pelas dicas, correções e sugestões. Agradeço também aos demais professores e funcionários do ICEx e da Escola da Engenharia da UFMG.

Agradeço aos amigos e colegas de graduação, de mestrado e de doutorado, pela convivência e ajuda; por tornarem a vida mais leve! Aos colegas do GOPAC, em especial aos amigos e professores do nosso grupo de otimização: André Cançado, André Cruz, Eduardo Carrano, Elizabeth Wanner, Frederico Cruz, Oriane Neto, um agradecimento especial pelas inúmeras sugestões e discussões informais.

Agradeço à CAPES e ao CNPq pelo apoio financeiro.

Resumo

A técnica da programação dinâmica consiste em decompor um problema de otimização dinâmica numa seqüência de sub-problemas, obtendo a solução de maneira incremental, tendo como base o Princípio da Otimalidade de Bellman. Entretanto, sua solução numérica é proibitiva em muitas aplicações práticas, característica de procedimentos enumerativos. Em vista disso, métodos sub-ótimos vêm sendo propostos para tal, dentre eles, métodos de solução em malha aberta. Nesta linha, esta tese propõe ferramentas computacionalmente tratáveis para problemas de programação dinâmica, considerando a dinâmica em malha aberta através da iteração de conjuntos fechados pelo sistema, como no controle preditivo. Consideramos problemas com ações de controle discretas no tempo, tendo como sistemas dinâmicos funções lineares, não-lineares, determinísticas, estocásticas, nos casos mono e multiobjetivo. Consideramos o caso impulsivo como um problema de otimização em tempo discreto, e no caso estocástico, usamos o conceito de dominância estocástica numa abordagem multi-quantil. Apresentamos cinco estudos de casos, mostrando a aplicação da metodologia proposta na solução de relevantes problemas reais. São eles: a otimização da implantação de uma fazenda de gado, o planejamento da expansão de uma rede de distribuição de energia elétrica, o controle biológico de pragas, o planejamento de estratégias de vacinação e o controle de estoque. Os resultados obtidos nos exemplos estudados se mostraram satisfatórios dos pontos de vista computacional e prático.
Palavras-chave: otimização, programação dinâmica, métodos sub-ótimos, matemática aplicada.

Abstract

The dynamic programming technique consists in the decomposition of the dynamic optimization problem in a sequence of sub-problems, obtaining the solution in a swelling way, based on the Bellman's Optimality Principle. However, its numerical solution is prohibitive in a large extent of the practical applications, as it happens in enumerative algorithms. In view of that, suboptimal procedures have been proposed, such as open-loop methods. In this way, this thesis proposes computationally tractable tools for dynamic programming problems, considering the dynamics in open-loop by the iteration of closed sets throughout the system, similar to the predictive control technique. We consider problems with discrete-time control actions, with dynamic systems having deterministic, stochastic, linear and nonlinear functions in the mono and multiobjective cases. The impulsive case is resolved as a discrete-time dynamic optimization problem, and in the stochastic case, the stochastic dominance concept is used in a multi-quantile approach. Five case studies are presented, showing the application of the proposed methodology in same relevant real-world problems. They are: the optimization of a herd cattle implantation, the optimal distribution network design of electrical energy, the biological control of plagues, the optimal vaccination strategies design, and the stock control. The results came from the case studies are satisfactory under the computational and the practical point of view.
Keywords: optimization, dynamic programming, sub-optimal methods, applied mathematics.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Principalmente após o advento dos computadores pessoais, as ferramentas para encontrar numericamente extremos de funções têm sido objeto de estudo na Engenharia Elétrica, e em tantas outras áreas do conhecimento. Estes problemas, chamados de problemas de otimização, podem ser: restritos ou irrestritos, lineares ou não-lineares, mono ou multiobjetivos, determinísticos ou estocásticos, de variáveis discretas ou contínuas, etc. (Goldbarg e Luna, 2000).

Na presente tese de doutorado, estudamos os problemas de otimização dinâmica, cujo objetivo consiste em encontrar extremos de uma função dinâmica ou restrita a um sistema dinâmico controlável. Propomos ferramentas para a solução numérica de problemas de otimização dinâmica em malha aberta, obtendo resultados em alguns casos sub-ótimos, em um tempo computacional razoável. Consideramos sistemas lineares, não-lineares, determinísticos, estocásticos, com ações de controle discretas no tempo, com variáveis reais ou binárias, nos casos mono-objetivo e multiobjetivo. Em todos os casos, assumimos informação perfeita, em que o estado em cada estágio possa ser plenamente conhecido.

Esta tese é seqüência do trabalho de mestrado (Cardoso, 2005), quando nos propusemos a estudar problemas de programação dinâmica linear, determinística, discreta no tempo, com variáveis reais ou que permitissem relaxação. Inspirados no artigo Takahashi et al. (1997), consideramos a dinâmica do sistema em malha aberta via iteração de conjuntos fechados com estrutura parametricamente invariante ao longo dos estágios do problema, a saber: conjuntos elipsoidais e politópicos. Apenas posteriormente, vimos a semelhança desta técnica com o trabalho de Bertsekas e Rhodes (1971) e com o controle preditivo (Garcia et al., 1989).

Neste capítulo, apresentamos um resumo sobre otimização dinâmica na seção 1.1, que é mais uma pequena história da técnica da programação dinâmica. Na seção 1.2, discutimos os objetivos e a relevância desta tese.

1.1 Otimização dinâmica

O objetivo de um problema de **otimização dinâmica** consiste em encontrar extremos de uma função dinâmica ou extremos de uma função restrita a um sistema dinâmico (controlável) (Chiang, 1992; Weber, 2003; Poulsen, 2004). Um sistema dinâmico pode ser visto como um sistema com memória, cujo parâmetro pode ser discreto ou contínuo (Chen, 1999; Monteiro, 2002). Ao longo deste texto, vamos considerar este parâmetro como sendo o tempo, sem perda de generalidade nos resultados[1]. No caso de tempo contínuo, a solução de um problema de otimização dinâmica consiste numa trajetória ótima, cuja variável escolhida em cada instante é a melhor decisão considerando-se todo o horizonte. No caso de tempo discreto, a solução é uma seqüência de ações ótimas, cuja variável escolhida em cada cada estágio é a melhor decisão considerando-se todo o horizonte. Também é possível considerar um horizonte de planejamento infinito.

Como exemplo, considere a alocação ótima dos recursos de uma empresa: um administrador precisa separar parte do lucro da sua produção para reinvestimento imediato na firma e parte para poupança como moeda forte. O objetivo deste problema de otimização dinâmica é maximizar a quantidade total do que for poupado como moeda forte ao longo de um dado período. Considerando que as decisões são tomadas continuamente no tempo, a teoria econômica nos fornece uma equação diferencial que modela o aumento da taxa de produção em função do que foi reinvestido na firma e da taxa de produção inicial do problema, supondo dados o capital inicial e/ou uma meta para o final. Podemos resolver o problema baseado nesta expressão.

Um exemplo considerando o tempo discreto pode ser o planejamento da política de compras de um supermercado ao longo de um ano. Considere que cada decisão acarreta um custo de compra e um custo de estocagem. A meta do supermercado é atender à demanda, gastando o mínimo possível com a compra e a estocagem. Podemos considerar no problema possíveis limitações orçamentárias e as limitações de capacidade no estoque. O problema de otimização dinâmica consiste em encontrar a estratégia de compras mensal que atenda às restrições e às metas do supermercado.

Baseado nestes dois exemplos, podemos listar os ingredientes básicos de um problema de otimização dinâmica:

- um ponto inicial dado e/ou um ponto final dado;
- um conjunto de alternativas (chamadas de caminhos, trajetórias ou políticas) admissíveis entre o ponto inicial e o ponto final;

[1]Isto é porque, geralmente, os demais parâmetros usados são funções monotônicas do tempo, como por exemplo, algum espacial, que meça distâncias.

- um conjunto de índices (números reais) associados às alternativas admissíveis (a cada alternativa admissível está associado um índice, responsável por medir o custo ou o benefício desta escolha);
- um objetivo específico: maximizar ou minimizar uma função matemática desses índices, aditiva sobre o tempo.

Três ferramentas têm sido as mais usadas para a solução de um problema de otimização dinâmica: cálculo variacional, controle ótimo e programação dinâmica.

Cálculo variacional

O **cálculo variacional** foi a primeira maneira de se resolver um problema de otimização dinâmica. Isaac Newton publicou a base desta técnica em 1687 em seu livro *Principia*, procurando encontrar uma superfície de revolução de área mínima. Outros matemáticos (como Euler, Lagrange, Bernoulli, Gauss, etc.) também estudaram problemas de natureza similar.

O problema fundamental de cálculo variacional consiste em determinar a função $x(t)$, ou simplesmente x, que minimiza um funcional dado em $t \in [0,T]$, sujeito às condições iniciais e/ou finais. Esta função x é chamada de **variável de estado** do problema. As variáveis de estado (o sistema pode ter mais de um estado, neste caso x seria uma grandeza vetorial) podem ser intuitivamente interpretadas como as responsáveis por armazenar um sumário da história do sistema.

Este problema pode ser formulado como em (1.1)[2].

$$\min_x \int_0^T g(t,x(t),x'(t))\ dt + g_T(x(T)) \tag{1.1}$$

$$\text{sujeito a: } \begin{cases} t \in [0,T]; \\ x(0) = x_0 \text{ e/ou } x(T) = x^* \text{ dados.} \end{cases}$$

A região factível deste problema, consiste no conjunto de todas as funções x, que chamamos de variáveis de estado admissíveis, tais que $x(0) = x_0$ e/ou $x(T) = x^*$. Como comentado, os valores das variáveis x_0 (estado inicial), x^* (estado final) e T (horizonte do problema) podem ou não serem dadas, em que cada caso define um tipo de problema a ser estudado.

As técnicas do cálculo variacional andam bem próximas das técnicas do cálculo diferencial tradicional, portanto é preciso que as funções envolvidas sejam continuamente diferenciáveis. Além disso, para que o problema fique bem posto, é preciso que o funcional seja integrável.

[2]O problema foi apresentado na forma de minimização, mas também pode ser escrito na forma de maximização.

Controle ótimo

O estudo continuado de problemas variacionais levou ao desenvolvimento da teoria do **controle ótimo**. No controle ótimo existem três, em vez de dois tipos de variáveis. Além do parâmetro tempo e das variáveis de estado, consideramos também variáveis de controle. Estas variáveis de controle desempenham papel fundamental no método, pois são os instrumentos de otimização: procuramos uma política de controle ótima e encontramos, a partir de um estado inicial e desta política ótima, o caminho de estado ótimo correspondente.

Assim, as variáveis de controle ou **ações de controle**, que chamaremos u, influenciam diretamente as variáveis de estado do problema, por isso, também são chamadas de variáveis de entrada do sistema. Matematicamente, no caso de tempo contínuo, esta influência se dá por meio da equação de um sistema dinâmico contínuo no tempo (uma equação diferencial). Esta equação, descrita em (1.2), é chamada de equação de estado ou de equação de movimento.

$$x'(t) = f(t,x(t),u(t)). \tag{1.2}$$

Este problema fundamental de controle ótimo pode ser formulado como em (1.3).

$$\min_u \int_0^T g(t,x(t),u(t))\ dt + g_T(x(T)) \tag{1.3}$$

$$\text{sujeito a: } \begin{cases} x'(t) = f(t,x(t),u(t)); \\ t \in [0,T]; \\ x(0) = x_0 \text{ e/ou } x(T) = x^* \text{ dados.} \end{cases}$$

Note que se consideramos $u(t) = x'(t)$, o problema de controle ótimo (1.3) fica formulado como um problema de cálculo variacional (1.1).

O resultado mais importante da teoria do controle ótimo é conhecido como **Princípio do Máximo**, comumente associado ao matemático russo Pontryagin (a versão em inglês do seu trabalho foi publicada em 1962). O poder desse princípio está na facilidade de se trabalhar com restrições nas variáveis de controle do problema, por exemplo, de se considerar as variáveis de controle pertencentes a conjuntos compactos. Além disso, o método pede apenas que as variáveis de controle sejam contínuas por partes e que as variáveis de estado apenas sejam deriváveis por partes, o que faz dele um pouco mais abrangente que o cálculo variacional.

Note que, assim como o cálculo variacional, o controle ótimo está interessado em resolver o problema dinâmico usando as propriedades do conjunto factível das variáveis de estado (e de controle) para garantir a otimalidade da solução.

Programação dinâmica

O pioneiro da técnica conhecida como **programação dinâmica** foi o matemático americano Richard Bellman, que escreveu seu primeiro livro na área em 1957, época do aparecimento do computador digital.

A programação dinâmica propõe a decomposição do problema de otimização dinâmica numa família de sub-problemas, e para cada sub-problema, a melhor solução é caracterizada pela comparação de índices associados aos valores de função-objetivo. Esta característica está baseada na causalidade dos sistemas dinâmicos envolvidos: como a decisão no tempo presente não influencia as decisões passadas, podemos otimizar em estágios, considerando apenas os estados nos tempos presente e futuros.

A formulação fundamental de um problema de programação dinâmica com tempo contínuo, apresentada na equação (1.4), não difere em nada da formulação do problema de controle ótimo.

$$\min_{u} \int_0^T g(t,x(t),u(t))\ dt + g_T(x(T)) \tag{1.4}$$

$$\text{sujeito a: } \begin{cases} x'(t) = f(t,x(t),u(t)); \\ t \in [0,T]; \\ x(0) = x_0 \text{ e/ou } x(T) = x^* \text{ dados.} \end{cases}$$

O problema fundamental de programação dinâmica com tempo discreto pode ser formulado como em (1.5)[3].

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{1.5}$$

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k],u[k]); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

Neste caso, N é o horizonte (estágio final), $x[k]$ corresponde ao estado no estágio k, etc.

[3]Nas formulações gerais deste texto, nos preocupamos **apenas com a restrição da dinâmica** do sistema em questão. Mas, é claro que nos modelos práticos, outras restrições podem ser consideradas, tanto nas variáveis de estado quanto nas variáveis de controle do problema. Pontuamos que, nos estudos de casos em cada um dos capítulos, consideraremos problemas dinâmicos com demais restrições.

No caso de tempo discreto, a equação de estado é a equação de um sistema dinâmico discreto no tempo (equação de diferenças). Esta equação, descrita em (1.6), também é chamada de mapa, ou ainda de equação de transição ou equação de evolução.

$$x[k+1] = f(x[k],u[k]). \tag{1.6}$$

Repare que a função-objetivo do problema é aditiva sobre o tempo, uma das premissas dos problemas de programação dinâmica. No caso com tempo discreto, sempre temos uma expressão para o custo de cada estágio, e o custo total será a soma dos custos ao longo dos estágios do problema. Note também que cada função g_k funciona como um índice medindo o custo ou benefício de cada par estado-ação de controle.

Considere a **restrição de ponto final**: $x[N] = x^*$, no caso do sistema com tempo discreto com N estágios, ou $x(T) = x^*$, no caso do sistema com tempo contínuo e horizonte de tempo T. Neste texto, chamamos o estado final x^*, o ponto que o estado deve atingir ao final da resolução do problema, de **meta de programação dinâmica**. Como comentado, a restrição de ponto final nem sempre é dada, mas quando dada, a meta de programação costuma ser o vetor nulo ou um ponto fixo do sistema dinâmico autônomo em questão[4].

Como veremos no capítulo 2, a maneira tradicional de se resolver numericamente um problema de programação dinâmica com um número finito de variáveis consiste em montar sua árvore de possibilidades e procurar nela um caminho mínimo. Se as variáveis do problema forem números reais, o algoritmo pode ser usado via discretização. Este algoritmo é ótimo, mas tem complexidade computacional exponencial, sendo inaplicável em grande parte dos problemas reais.

1.2 Propostas do trabalho

1.2.1 Objetivos

Esta tese de doutorado tem como principal objetivo algo, digamos, amplo: propor ferramentas para certas classes de problemas de programação dinâmica concebidas sob um mesmo paradigma, inspirado em técnicas de controle em malha aberta, além de mostrar os resultados da aplicação destas ferramentas na solução de importantes problemas reais. No trabalho de mestrado (Cardoso, 2005), estudamos problemas dinâmicos com sistemas lineares, determinísticos, discretos no tempo, com variáveis reais ou que permitam uma relaxação. Nesta

[4]O sistema sem ação de controle (ou com as variáveis de controle nulas) é chamado de **sistema autônomo**. Neste caso, as variáveis de otimização podem ser, por exemplo, os valores iniciais das próprias variáveis de estado, como no caso do cálculo variacional.

tese, estendemos esta abordagem para outras classes de problemas de programação dinâmica.

Em vez de estudarmos a dinâmica do sistema através dos arcos de um grafo, como classicamente é feito, a idéia central consiste em procurar, estágio-por-estágio, a seqüência ótima de ações de controle, considerando o sistema dinâmico associado em apenas um dos estágios do problema, a partir de uma ótica geométrica: através da iteração de conjuntos fechados. Este fato permite uma redução considerável no custo computacional da solução, pois evita sua característica enumerativa.

Os algoritmos propostos são exatos quando a solução em malha aberta corresponde à solução em malha fechada, como nos problemas com variáveis reais e sistemas determinísticos ou estocásticos com uma equivalência à certeza (confira na seção 2.5), desde que resolvidos por métodos exatos de otimização estática; são assintoticamente exatos para os problemas com variáveis inteiras, desde que permitida a devida relaxação e resolvidos por métodos exatos de otimização estática; e são sub-ótimos nos demais casos.

Consideramos três classes de problemas de programação dinâmica. Para cada classe, propomos ferramentas específicas e apresentamos a aplicação destas ferramentas em estudos de casos:

1. Problemas dinâmicos com sistemas lineares, determinísticos, discretos no tempo. Estudos de casos: a otimização biobjetivo dos recursos envolvidos na implantação de um rebanho de gado e o projeto de redes de distribuição ao longo do tempo.

2. Problemas dinâmicos com sistemas não-lineares, determinísticos, com controle impulsivo. Estudos de casos: problema do controle biológico de pragas e a otimização das estratégias de vacinação numa situação de epidemia.

3. Problemas dinâmicos com sistemas lineares, estocásticos, discretos no tempo, numa abordagem multi-quantil. Estudo de caso: problema de estoque.

1.2.2 Relevância

Existe um algoritmo ótimo para a solução de problemas de programação dinâmica, baseado no Princípio da Otimalidade de Bellman (princípio 2.1). Entretanto, sua solução analítica costuma ser muito difícil ou impossível e sua solução numérica sofre da chamada "maldição da dimensionalidade", pois apresenta um custo computacional exponencial em função do número de variáveis de estado admissíveis, devido a seu caráter enumerativo.

Portanto, torna-se indispensável o desenvolvimento de métodos de otimização que sejam computacionalmente tratáveis e obtenham resultados satisfatórios do

ponto de vista prático, ainda que de modo sub-ótimo. Normalmente, as técnicas sub-ótimas costumam depender da classe de problemas a serem tratados, e problemas não-lineares, estocásticos, multiobjetivos e com variáveis discretas têm sido tidos como problemas difíceis. A relevância do nosso estudo está em propor ferramentas não-enumerativas para estas classes de problemas.

Para os problemas dinâmicos com sistemas lineares, usamos da Álgebra Linear para melhor fundamentar a metodologia proposta. Para a classe de problemas não-lineares, propomos a otimização de problemas impulsivos via métodos de programação dinâmica discreta no tempo. Para a classe de problemas estocásticos, propomos uma abordagem multi-quantil: considerar diversos quantis da função-objetivo do problema, via simulações de Monte Carlo, como critérios de otimização, em vez de apenas o valor esperado.

Além disso, estudos de casos por si mesmos relevantes foram escolhidos para exemplificar cada uma das classes de problemas.

Por exemplo, o estudo do projeto de expansão de redes de energia elétrica é de vital relevância social e econômica, visto sua importância no desenvolvimento de um país. O sistema de distribuição de energia elétrica é a parte mais cara do sistema; portanto, redução nas perdas leva a considerável redução nos custos, permitindo repassar a redução dos preços para a população.

O controle biológico de pragas na lavoura é um método ecologicamente correto, e consiste na redução da população das pragas através da atuação de outros organismos vivos, geralmente conhecidos como inimigos naturais. Um assunto crítico tem sido alcançar sua completa viabilidade através da determinação da política de gerenciamento que conduza a melhores resultados econômicos. Neste sentido, técnicas de otimização dinâmica podem representar um importante diferencial capaz de fazer esta tecnologia atingir o nível de eficiência que seria necessário para sua viabilização econômica.

O controle de estoques é um problema clássico, principalmente considerando a demanda seguindo uma distribuição normal. Neste texto, consideramos demandas com distribuições assimétricas e multi-modais, e principalmente nestes casos a análise multi-quantil se justifica. Afinal de contas, pode ser importante para um tomador de decisões planejar sua política de compras segundo um quantil maior ou menor que a mediana, podendo até variar o critério de otimização como lhe convier no momento.

1.2.3 Estrutura do texto

Além da revisão da literatura e das conclusões, organizamos o restante deste texto separando uma classe de problemas por capítulo.

No capítulo 2, fazemos uma revisão sobre a técnica da programação dinâmica em tempo discreto e contínuo, falamos sobre a programação dinâmica impul-

siva, bem como sobre as principais técnicas de programação dinâmica sub-ótima e sobre a programação dinâmica multiobjetivo. Também apresentamos uma síntese geral das ferramentas propostas, destacando o que elas tem em comum, e relacionando-as com as demais técnicas sub-ótimas.

No capítulo 3, apresentamos as ferramentas para programação dinâmica cujo sistema dinâmico associado é linear. Apresentamos as ferramentas já discutidas na dissertação de mestrado (Cardoso, 2005), acrescentando a abordagem multiobjetivo. Dois estudos de casos são considerados: uma aplicação da abordagem multiobjetivo na otimização dos recursos obtidos com a implantação de um rebanho de gado e a otimização do projeto de distribuição de redes.

No capítulo 4, estendemos para a programação dinâmica não-linear a metodologia apresentada para a programação dinâmica linear discreta no tempo. Consideramos o caso do sistema com tempo discreto e o caso impulsivo. Os estudos de casos deste capítulo são: o controle biológico de pragas numa lavoura e a busca de estratégias ótimas de vacinação numa situação de epidemia.

No capítulo 5, consideramos a programação dinâmica estocástica, estendendo a metodologia via simulações de Monte Carlo. No caso multiobjetivo, consideramos diversos quantis em vez que usar apenas o valor esperado da função objetivo como critério de otimização. Por fim, consideramos como estudo de caso o problema de estoque com demandas assimétricas e bimodais.

No capítulo 6, apresentamos as conclusões e discussões finais, destacando as principais contribuições do nosso trabalho, em três classes: estado da arte, ferramentas propostas e estudos de casos. Discutimos também a perspectiva de trabalhos futuros.

Capítulo 2

Programação Dinâmica

Neste capítulo, fazemos uma revisão sobre a técnica da programação dinâmica. Na seção 2.2, apresentamos a programação dinâmica em tempo discreto (estudando separadamente os casos determinístico e estocástico), na seção 2.3, falamos sobre a programação dinâmica em tempo contínuo e na seção 2.4, sobre a programação dinâmica impulsiva. Na seção 2.5, fazemos um apanhado dos principais métodos de programação dinâmica sub-ótima (com aproximação implícita e explícita) e na seção 2.6, falamos de programação dinâmica multiobjetivo. Na seção 2.7, apresentamos uma síntese geral das ferramentas propostas nesta tese. Na seção 2.8 apresentamos as conclusões deste capítulo.

2.1 Introdução

A **programação dinâmica**, introduzida por Bellman (1957), é uma técnica simples e poderosa para a resolução de problemas de otimização dinâmica. Trabalhos posteriores a este livro, do próprio Bellman e de outros matemáticos, engenheiros e economistas, vêm ressaltando a importância teórica do tema e suas numerosas aplicações. Bertsekas (1985, 1995) apresenta uma série de problemas que vêm sendo resolvidos via programação dinâmica, tais como: controle de estoque, problema de filas, problema das quatro rainhas, problema do caixeiro viajante, problema da mochila, problemas de caminho mínimo em redes, análise de portfólios dinâmicos, problemas de parada, problemas de alocação e seqüenciamento, alocação de recursos ao longo do tempo, orçamento de capital, programas de jogos, etc.[1].

A idéia central da solução de um problema de otimização dinâmica pela técnica da programação dinâmica consiste em decompor o problema original

[1]Atualmente, o *Web of Knowledge* registra 5.675 citações de aplicações de programação dinâmica (*dynamic programming applications*) entre 1960 e 2007; só em 2007 foram 316.

numa família de subproblemas, de modo que resolvendo cada subproblema seqüencialmente, resolve-se o problema todo. A base teórica desta técnica é o Princípio da Otimalidade de Bellman.

Princípio 2.1 (Princípio da Otimalidade de Bellman) *A partir de cada ponto da trajetória ótima, a trajetória restante é a ótima do problema correspondente iniciando nesse ponto.*

A justificativa para o princípio da otimalidade é muito simples: se a política truncada não fosse a melhor para o subproblema, então haveria outra que o seria. E assim fosse, poderíamos melhorar a função-objetivo considerando esta nova política ótima na porção correspondente a este subproblema. Para um exemplo com tráfego: se o caminho mais curto entre Belo Horizonte e a cidade do Rio de Janeiro passar por Juiz de Fora, então a porção desta rota entre Juiz de Fora e o Rio tem que ser o caminho ótimo entre Juiz de Fora e o Rio. É claro que, para ser resolvido por programação dinâmica, o problema deve ter esta sub-estrutura ótima.

Outra característica desta técnica consiste em concentrar os esforços nos valores de índices associados a cada solução candidata a ótima. A vantagem desta abordagem é deixar a análise independente das propriedades das funções envolvidas na otimização.

Nesta seção, vamos aprofundar nesta técnica: vamos mostrar os algoritmos exatos, sua fundamentação teórica e alguns exemplos didáticos; vamos discutir os problemas destes algoritmos exatos, que apesar de serem simples, são inaplicáveis em muitos problemas reais; vamos apresentar e discutir alternativas sub-ótimas a um custo computacional mais razoável.

Vamos apresentar separadamente os casos de tempo discreto (na subseção 2.2) e tempo contínuo (na subseção 2.3). Na subseção sobre tempo discreto, vamos estudar separadamente os casos determinístico e estocástico e na subseção sobre tempo contínuo, omitimos o caso estocástico. O problema de programação dinâmica pode ser resolvido analiticamente ou numericamente, como exemplificaremos. Na subseção 2.5, falaremos sobre os métodos sub-ótimos para programação dinâmica, principal foco deste trabalho.

Atualmente, os principais livros sobre programação dinâmica são: Bertsekas (1985, 1995), de onde tiramos a maior parte das informações deste capítulo. Também serviram de fonte para esta seção os seguintes livros: Dano (1975); Wagner (1986); Weber (2003).

2.2 Caso discreto

Na programação dinâmica discreta, as decisões devem ser tomadas em estágios ou em etapas seqüenciais e a resposta, ou a decisão em cada estágio, influencia apenas nas decisões a serem tomadas nos estágios seguintes. Assim, o algoritmo da programação dinâmica com tempo discreto vai procurar atacar o problema dividindo-o numa seqüência de subproblemas por estágios, de modo a otimizar em dimensões mais baixas, ou seja, com um número menor de variáveis por vez.

Uma característica da solução de um problema de programação dinâmica discreta é que cada subproblema deverá estar escrito em função da variável de decisão em um estágio e da solução ótima do subproblema do estágio posterior. A chave é que a decisão de cada estágio não pode ser tomada isoladamente das outras, mas pode ser tomada seqüencialmente e progressivamente ao longo dos estágios.

Podemos sintetizar as características na solução de um problema de programação dinâmica discreta como:

- separar o problema de controle numa família de subproblemas mais simples de serem resolvidos, de modo que ao final da resolução do último subproblema, o problema original terá sido resolvido;
- o subproblema de cada estágio deverá estar escrito em função da variável de decisão neste estágio e das variáveis do subproblema posterior;
- cada subproblema deve ser resolvido concentrando a atenção nos índices, ou seja, nos valores das funções-objetivo de cada subproblema.

Vamos discorrer separadamente sobre a programação dinâmica com tempo discreto nos casos: determinístico e estocástico.

2.2.1 Caso determinístico

Considere o problema fundamental de programação dinâmica determinístico com tempo discreto, formulado em (1.5) e reescrito aqui para comodidade do leitor[2].

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N])$$

[2]Caso o valor da variável de estado (inicial ou final) não tenha sido dado, ela pode entrar no problema como variável de otimização. Discutiremos esse caso nos exemplos.

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k],u[k]); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

A equação de estados do problema de tempo discreto é a (1.6), reescrita aqui.

$$x[k+1] = f(x[k],u[k]).$$

O objetivo é encontrar a seqüência ótima de variáveis de controle ou de ações de controle: $\{u^*[0],\ldots,u^*[N-1]\}$ que minimizem a função-objetivo ou função-custo do problema (2.1).

$$J = \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]). \tag{2.1}$$

Denotaremos o valor ótimo da função-objetivo por J^*:

$$J^* = \min_{u[0],\ldots,u[N-1]} J. \tag{2.2}$$

Vamos introduzir a seguinte notação: $u[i,k] = \{u[i],\ldots,u[k]\}$ corresponde à seqüência de decisões do estágio i até o estágio k. Consideremos a função-truncada no estágio i, como sendo (2.3).

$$J_i(x[i],u[i,N-1]) = \sum_{k=i}^{N-1} g_k(x[k],u[k]) + g_N(x[N]). \tag{2.3}$$

Esta função está bem definida, pois, dados $x[i]$ e a seqüência $u[i,N-1]$, podemos encontrar a seqüência de estados $x[i+1,k]$ pela equação de estado (1.6).

É fácil ver que:

$$J_i(x[i],u[i,N-1]) = g_i(x[i],u[i]) + J_{i+1}(x[i+1],u[i+1,N-1]). \tag{2.4}$$

Em particular:

$$J = J_0(x[0],u[0,N-1]), \tag{2.5}$$

cujo valor de J é o definido na equação (2.1).

A **função de Bellman** (2.6) é uma função do estado presente e é definida como sendo o ótimo de cada função-truncada (2.3).

$$V_i(x[i]) = \min_{u[i,N-1]} J_i(x[i],u[i,N-1]). \tag{2.6}$$

Note que podemos definir a condição de contorno (2.7).

$$V_N(x[N]) = g_N(x[N]). \tag{2.7}$$

Podemos, por fim, apresentar o teorema fundamental da programação dinâmica, publicado por Bellman em 1957.

Teorema 2.1 (Teorema Fundamental da Programação Dinâmica)
Considere o problema de programação dinâmica determinístico com tempo discreto formulado em (1.5). A função de Bellman (2.6) é dada pela equação recursiva (2.8):

$$V_i(x[i]) = \min_{u[i]} \left(g_i(x[i],u[i]) + V_{i+1}(x[i+1]) \right), \tag{2.8}$$

com condição de contorno (2.7):

$$V_N(x[N]) = g_N(x[N]).$$

O valor ótimo da função-objetivo (2.2) é $J^* = V_0(x[0])$.

Demonstração:
Da definição da função de Bellman (2.6) e da equação recursiva (2.4), temos que:

$$\begin{aligned} V_i(x[i]) &= \min_{u[i,N-1]} J_i(x[i],u[i,N-1]) = \\ &= \min_{u[i,N-1]} g_i(x[i],u[i]) + J_{i+1}(x[i+1],u[i+1,N-1]). \end{aligned}$$

Como $u[i+1,N-1]$ não afeta g_i, podemos reescrever:

$$\begin{aligned} V_i(x[i]) &= \min_{u[i]} \left(g_i(x[i],u[i]) + \min_{u[i+1,N-1]} J_{i+1}(x[i+1],u[i+1,N-1]) \right) = \\ &= \min_{u[i]} \left(g_i(x[i],u[i]) + V_{i+1}(x[i+1]) \right). \end{aligned}$$

A condição de contorno é a mesma da função de Bellman (2.7). Isto conclui a demonstração da primeira parte do teorema.

Aplicando a relação acima, temos que:

$$V_0(x[0]) = \min_{u[0]} \left(g_0(x[0],u[0]) + \min_{u[1,N-1]} J_1(x[1],u[1,N-1]) \right) = J_0(x[0],u[0,N-1]).$$

Das equações (2.5) e (2.2), concluímos que $J^* = V_0(x[0])$. Isto conclui a demonstração do teorema. ■

Nota 2.1 *A equação funcional (2.8) é chamada de* ***equação de Bellman****.*

Repare que este teorema fornece um algoritmo para a resolução de problemas de programação dinâmica. A idéia é otimizar em estágios, a partir do estágio final, sendo que a otimização referente ao estágio k se dá na variável de decisão do estágio k e é função do resultado da otimização referente ao estágio $k+1$. O mesmo raciocínio pode ser feito para a otimização a partir do estágio inicial.

Vamos exemplificar esse algoritmo num caso simples.

Exemplo 2.2 (Problema LQ (linear-quadrático) simples)
Considere o seguinte problema com sistema dinâmico linear e função-objetivo quadrática:

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (x[k]^2 + u[k]^2) + x[N]^2$$

$$\textit{sujeito a: } \begin{cases} x[k+1] = x[k] + u[k]; \\ k = 0,1,\ldots,N-1; \\ x[N] = x^* \textit{ dado.} \end{cases}$$

Para resolver este problema por programação dinâmica, vamos definir:

$$V_N(x[N]) = x[N]^2.$$

Esta é a condição de contorno. A equação de Bellman desse problema é a seguinte:

$$V_i(x[i]) = \min_{u[i]} \left(x[i]^2 + u[i]^2 + V_{i+1}(x[i+1]) \right),$$

Vamos considerar como horizonte $N = 3$ e como estado final dado $x[3] = 1$. Ou seja, $V_3(x[3]) = x[3]^2 = 1$.

Resolvendo a equação de Bellman para o segundo estágio, temos que:

$$V_2(x[2]) = \min_{u[2]} \left(x[2]^2 + u[2]^2 + V_3(x[3]) \right) = \min_{u[2]} \left(x[2]^2 + u[2]^2 + 1 \right).$$

O mínimo, claro, é atingido para $u[2] = 0$. Portanto, $x[2] = x[3] - u[2] = 1$ e $V_2(x[2]) = 2$.

Resolvendo para o primeiro estágio, temos que:

$$V_1(x[1]) = \min_{u[1]} \left(x[1]^2 + u[1]^2 + V_2(x[2]) \right) = \min_{u[1]} \left(x[1]^2 + u[1]^2 + 2 \right).$$

O mínimo é $u[1] = 0$, portanto, $x[1] = 1$ e $V_1(x[1]) = 3$.

Para o estágio inicial, temos que:

$$V_0(x[0]) = \min_{u[0]} \left(x[0]^2 + u[0]^2 + V_1(x[1]) \right) = \min_{u[0]} \left(x[0]^2 + u[0]^2 + 3 \right).$$

O mínimo é $u[0] = 0$, portanto, $x[0] = 1$. Desta forma, $J^* = V_0(x[0]) = 4$, obtido para $u[0] = u[1] = u[2] = 0$. □

A nota 2.1 nos chama a atenção para a necessidade de se resolver uma equação funcional recursiva na solução analítica genérica de um problema de programação dinâmica. Comentamos apenas que resolver analiticamente uma equação funcional recursiva pode não ser tarefa das mais fáceis!

Outro ponto que devemos considerar é o caso cujas variáveis de controle e de estado estejam restritas a conjuntos, por exemplo:

$$u[i] \in \mathcal{U}_i \text{ e/ou } x[i] \in \mathcal{X}_i. \tag{2.9}$$

Neste caso, a região factível da otimização se restringe a:

$$\mathcal{U}_i^*(x[i]) = \{u[i] \in \mathcal{U}_i; f(x[i],u[i]) \in \mathcal{D}_i\}, \tag{2.10}$$

com:

$$\mathcal{D}_i = \{x[i] \in \mathcal{X}_i; \exists u[i] \in \mathcal{U}_i; f(x[i],u[i]) \in \mathcal{D}_{i+1}\}, \tag{2.11}$$

$$\mathcal{D}_N = \mathcal{X}_N.$$

Teorema 2.3 (Teorema 2.1 revisto)
Considere o problema de programação dinâmica determinístico com tempo discreto formulado em (1.5). Suponha que o problema tenha restrição nas variáveis, como em (2.9). A função de Bellman desse problema é dada pela equação recursiva (2.12):

$$V_i(x[i]) = \min_{u[i] \in \mathcal{U}_i^*} \left(g_i(x[i],u[i]) + V_{i+1}(x[i+1]) \right), \tag{2.12}$$

com $x[i] \in \mathcal{D}_i$ e com condição de contorno (2.7):

$$V_N(x[N]) = g_N(x[N]).$$

DEMONSTRAÇÃO:
Segue direto. ■

Exemplo 2.4 (Problema LQ restrito)
Considere o problema LQ do exemplo 2.2 com restrição nas variáveis de controle:

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (x[k]^2 + u[k]^2) + x[N]^2$$

$$\textit{sujeito a: } \begin{cases} x[k+1] = x[k] + u[k]; \\ k = 0,1,\ldots,N-1; \\ x[N] = x^* \textit{ dado}; \\ u[k] \in [1,2]. \end{cases}$$

A equação de Bellman desse problema é a seguinte:

$$V_i(x[i]) = \min_{u[i]\in[1,2]} \left(x[i]^2 + u[i]^2 + V_{i+1}(x[i+1])\right),$$

A condição de contorno desse problema é a mesma:

$$V_N(x[N]) = x[N]^2.$$

Também vamos considerar como horizonte $N = 3$ e como estado final dado $x[3] = 1$. Ou seja, também nesse exemplo $V_3(x[3]) = x[3]^2 = 1$.

Resolvendo a equação de Bellman para o segundo estágio, temos que:

$$V_2(x[2]) = \min_{u[2]\in[1,2]} \left(x[2]^2 + u[2]^2 + V_3(x[3])\right) = \min_{u[2]} \left(x[2]^2 + u[2]^2 + 1\right).$$

O mínimo, claro, é $u[2] = 1$. Portanto, $x[2] = x[3] - u[2] = 0$ e $V_2(x[2]) = 2$.

Resolvendo para o primeiro estágio, temos que:

$$V_1(x[1]) = \min_{u[1]\in[1,2]} \left(x[1]^2 + u[1]^2 + V_2(x[2])\right) = \min_{u[1]} \left(x[1]^2 + u[1]^2 + 2\right).$$

O mínimo é $u[1] = 1$, portanto, $x[1] = -1$ e $V_1(x[1]) = 4$.

Para o estágio inicial, temos que:

$$V_0(x[0]) = \min_{u[0]\in[1,2]} \left(x[0]^2 + u[0]^2 + V_1(x[1])\right) = \min_{u[0]} \left(x[0]^2 + u[0]^2 + 3\right).$$

O mínimo é $u[0] = 1$, portanto, $x[0] = -2$ e $J^* = V_0(x[0]) = 8$. □

Quando as variáveis de estado e de controle do problema discreto no tempo estiverem restritas a conjuntos discretos e finitos um método numérico mais simples pode ser proposto. Este método consiste em considerar o problema de programação dinâmica como a busca seqüencial de um caminho mínimo em um grafo direcionado. Lembramos que em muitas aplicações práticas, é usual o fato das variáveis de estado e/ou de controle estarem restritas a conjuntos discretos e finitos e em outros casos, pode-se tomar uma discretização das variáveis reais e considerá-las entre limites inferiores e superiores.

Este grafo direcionado (Goldbarg e Luna, 2000) é conhecido como **árvore de possibilidades** do problema. Ele é um grafo em estágios cujos nodos do estágio k são os estados $x[k]$ e os arcos entre os dois estados $x[k]$ e $x[k+1] = f(x[k],u[k])$ são

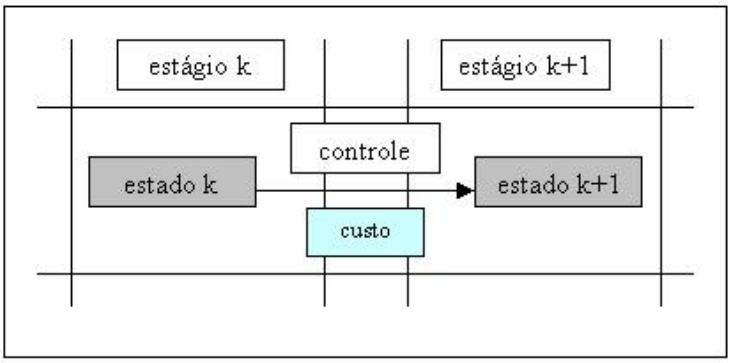

Figura 2.1: Programação dinâmica com variáveis discretas e um número finito de possibilidades. Diagrama mostrando a transição do estado $x[k]$ para o estado $x[k+1]$ num grafo, via controle $u[k]$ com o respectivo custo associado.

os controles $u[k]$ admissíveis. O custo deste arco é $g_k(x[k], u[k])$, que representa o custo de transição do estado $x[k]$ para o estado $x[k+1]$ sob o controle $u[k]$. A figura 2.1 apresenta um diagrama exemplificando a transição de estados via arcos de um grafo.

O algoritmo ótimo para a solução de um problema de programação dinâmica com tempo discreto e com variáveis discretas e finitas é bem simples, e consiste em:

Algoritmo Clássico - Programação Dinâmica

---

1. Montar a árvore de possibilidades do problema: listar, por estágio, os estados e os controles admissíveis;

2. Procurar nesse grafo, seqüencialmente por estágios, para cada estado, um caminho mínimo entre este estado e o estado final.

Ou seja, um problema de programação dinâmica com tempo discreto e número finito de estados pode ser visto como uma busca seqüencial num grafo e vice-versa.

Vamos explicar esse algoritmo por meio de um exemplo de tráfego.

Exemplo 2.5 (Problema de caminho mais curto)
Um viajante quer ir da cidade A para a cidade J em 4 dias de viagem a um mínimo custo. Para o primeiro dia, o viajante deve escolher entre as cidades B, C ou D. Para o segundo dia, ele deve escolher entre as cidades E, F ou G.

Depois, para o terceiro dia, deverá escolher entre as cidades H ou I e finalmente chegar em J no quarto dia. A figura 2.2 apresenta a árvore de possibilidades deste problema num grafo, cujos nodos são as cidades, os arcos são as decisões com os respectivos custos e as barras verticais indicam os dias de viagem. A tabela 2.1 apresenta os custos de deslocamento entre as cidades.

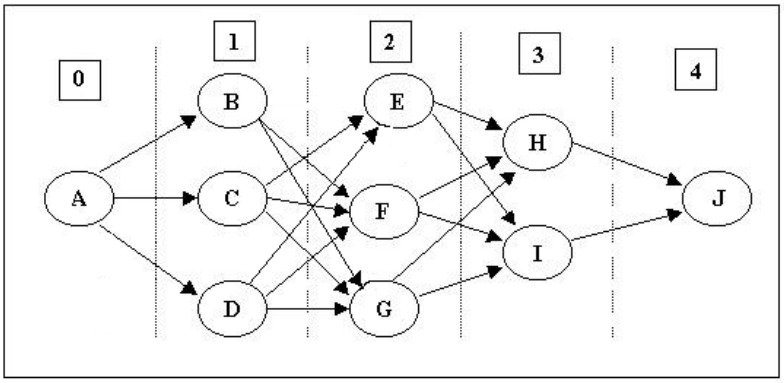

Figura 2.2: Problema de caminho mais curto (exemplo 2.5). Grafo direcionado cujos nodos são as cidades (os estados), os arcos são as decisões com os respectivos custos e as barras verticais indicam os dias de viagem (os estágios do problema).

A solução desse problema pode ser encontrada através da programação dinâmica, resolvendo o problema em estágios. Seja $V(X)$ a menor distância entre o nodo X e o nodo final J. Então, claramente, $V(H) = 3$ e $V(I) = 4$, relativos ao terceiro estágio do problema.

Para resolver o problema no segundo estágio, temos que:

$$\begin{aligned} V(E) &= \min(1 + V(H), 4 + V(I)) = \min(1 + 3, 4 + 4) = 4; \\ V(F) &= \min(6 + V(H), 3 + V(I)) = \min(6 + 3, 3 + 4) = 7; \\ V(G) &= \min(3 + V(H), 3 + V(I)) = \min(3 + 3, 3 + 4) = 6. \end{aligned}$$

Portanto, a partir do estágio 2, a sub-seqüência de caminhos ótima é:

$$\begin{aligned} &\textit{partindo de } E: \quad E \to H \to J; \\ &\textit{partindo de } F: \quad F \to I \to J; \\ &\textit{partindo de } G: \quad G \to H \to J. \end{aligned}$$

	A	B	C	D	E	F	G	H	I	J
A	-	2	4	4	-	-	-	-	-	-
B	-	-	-	-	-	4	6	-	-	-
C	-	-	-	-	3	2	4	-	-	-
D	-	-	-	-	4	1	5	-	-	-
E	-	-	-	-	-	-	-	1	4	-
F	-	-	-	-	-	-	-	6	3	-
G	-	-	-	-	-	-	-	3	3	-
H	-	-	-	-	-	-	-	-	-	3
I	-	-	-	-	-	-	-	-	-	4
J	-	-	-	-	-	-	-	-	-	-

Tabela 2.1: Problema de caminho mais curto (exemplo 2.5). Tabela com os custos de transição de estados para a árvore de possibilidades da figura 2.2. Os nodos de origem estão na vertical e os nodos de destino estão na horizontal. Os "traços" correspondem à ausência de caminho entre os nodos.

Para resolver o problema no primeiro estágio, temos que:

$$\begin{aligned} V(B) &= \min(-*-,4+V(F),6+V(G)) = \min(-*-,4+7,6+6) = 11; \\ V(C) &= \min(3+V(E),2+V(F),4+V(G)) = \min(3+4,2+7,4+6) = 7; \\ V(D) &= \min(4+V(E),1+V(F),5+V(G)) = \min(4+4,1+7,5+6) = 8. \end{aligned}$$

Repare que não existe caminho factível entre os nodos B e E e que houve um empate entre dois caminhos: do nodo D para o nodo E e para o nodo F. Assim, a partir do primeiro estágio, a sub-seqüência de caminhos ótima é:

$$\begin{aligned} &\textit{partindo de } B: && B \to F \to I \to J; \\ &\textit{partindo de } C: && C \to E \to H \to J; \\ &\textit{partindo de } D: && D \to E \to H \to J \textit{ ou} \\ &\textit{partindo de } D: && D \to F \to I \to J \textit{ (empate).} \end{aligned}$$

Finalmente, para o estágio inicial, temos que:

$$V(A) = \min(2+V(B),4+V(C),4+V(D)) = \min(2+11,4+7,4+8) = 11.$$

Portanto, o caminho ótimo por estágios, com comprimento 11, é:

$$A \to C \to E \to H \to J.$$

A figura 2.3 destaca este caminho ótimo e a figura 2.4 mostra toda a solução ótima fornecida pelo algoritmo. □

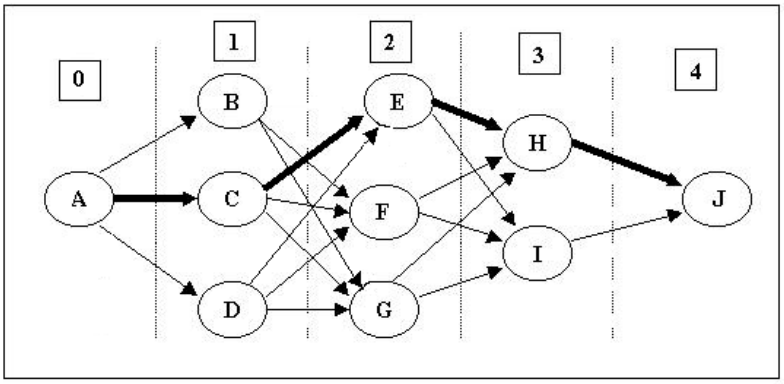

Figura 2.3: Problema de caminho mais curto (exemplo 2.5). Destaque para o caminho ótimo encontrado pelo algoritmo (em negrito).

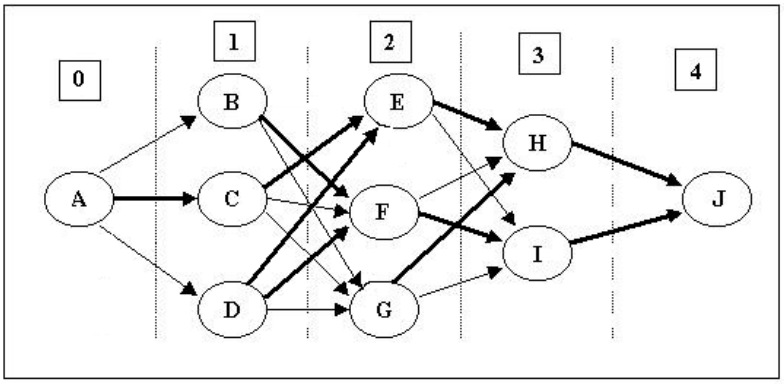

Figura 2.4: Problema de caminho mais curto (exemplo 2.5). Destaque para a solução ótima encontrada pelo algoritmo (em negrito).

Note na figura 2.4 que, além da seqüência ótima de ações de controle, com o respectivo caminho ótimo mostrado na figura 2.3, o algoritmo clássico da programação dinâmica fornece a parte ótima do caminho entre cada estado e o estado final. Desta forma, se por algum motivo, a rota tiver que ser desviada, não é necessária nova otimização.

Nota 2.2 *O algoritmo clássico da programação dinâmica pode ser identificado com o* ***controle em malha fechada*** *(Ogata, 1998), uma vez que, terminada a otimização, para cada estado admissível, sabemos qual deve ser a ação de controle ótima a ser tomada. Em contrapartida, o procedimento de se procurar a seqüência*

de ações de controle propriamente dita pode ser identificado com o ***controle em malha aberta****, uma vez que a otimização é feita sem nenhuma informação sobre os reais valores dos estados ao longo do processo.*

O problema é que montar esta árvore de possibilidades é, em geral, impraticável, exceto em problemas de baixa dimensão, dada a explosão combinatória dos métodos enumerativos. Isso porque a montagem da árvore **tem complexidade computacional exponencial** em função do número de estados por estágio possíveis e do número de estágios do problema. Este fato, conhecido como "maldição da dimensionalidade" torna a aplicação da técnica da programação dinâmica proibitiva em muitos casos práticos.

2.2.2 Caso estocástico

Vamos considerar nesta subseção a presença da aleatoriedade no sistema dinâmico a ser otimizado via programação dinâmica. Este assunto, muito comum em vários problemas práticos, é chamado de **programação dinâmica estocástica**.

A equação de estados de um sistema estocástico discreto é dada por (2.13).

$$x[k+1] = f(x[k],u[k],w[k]). \tag{2.13}$$

Cada $w[k]$ é uma variável aleatória independente para cada estágio do problema discreto no tempo. Neste contexto, pode ser chamada de distúrbio (aleatório) ou ruído (aleatório). Dado um horizonte N, a seqüência $\{w[0],\ldots,w[N-1]\}$ define um processo estocástico.

O custo da decisão $u[k]$ a ser tomada no estágio k a partir do estado $x[k]$ para uma dada variável aleatória $w[k]$ está bem definido: $g_k(x[k],u[k],w[k])$. Portanto, a função J (mostrada na equação 2.14) que soma os custos ao longo dos estágios seria uma boa candidata a função-objetivo problema.

$$J = \sum_{k=0}^{N-1} g_k(x[k],u[k],w[k]) + g_N(x[N]). \tag{2.14}$$

Assim, dado um horizonte, poderíamos formular o problema que procure a seqüência ótima de políticas ou variáveis de controle $\{u[0],\ldots,u[N-1]\}$ que minimizassem esta função-objetivo J, restrita ao sistema dinâmico (2.13). Mas, note que esta função J, bem como as variáveis de estado, também são variáveis aleatórias, pois dependem do processo estocástico $\{w[0],\ldots,w[N-1]\}$. Portanto, precisamos definir o que seria o mínimo de uma função de variáveis aleatórias

a fim de encontrarmos a estratégia ótima correspondente. Mais precisamente, precisamos definir quando o valor de uma função é melhor que outro.

A estratégia que vem sendo adotada consiste em pensar num ponto-de-vista médio e tomar o valor esperado desta função como referência. Estamos, assim, considerando que o valor esperado é uma boa medida para sintetizar as características principais de uma variável aleatória. Considere a notação: E $[X]$ para o valor esperado da variável aleatória X.

O problema fundamental de programação dinâmica estocástica com tempo discreto pode ser formulado como em (2.15). Queremos encontrar a seqüência de ações de controle que minimizem o valor esperado da função-objetivo, sujeito ao sistema dinâmico.

$$\min_{u[0],\ldots,u[N-1]} \text{E} \left[\sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \right] \tag{2.15}$$

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k],u[k],w[k]); \\ k = 0,1,\ldots,N-1; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

Neste caso, o estado inicial $x[0]$ também poderia ser estocástico.

Denotaremos o valor ótimo da função-objetivo por J^*:

$$J^* = \min_{u[0],\ldots,u[N-1]} \text{E } [J]\,. \tag{2.16}$$

A função truncada no estágio i para o caso estocástico depende do estado presente $x[i]$, da seqüência de decisões $u[i,N-1]$ e da seqüência de variáveis aleatórias $\{w[i],\ldots,w[N-1]\}$ ou $w[i,N-1]$.

A **função estocástica de Bellman** também pode ser definida como o ótimo da função truncada, neste novo ponto-de-vista, como em (2.17).

$$V_i(x[i]) = \min_{u[i,N-1]} \text{E } [J_i(x[i],u[i,N-1])]\,. \tag{2.17}$$

Note que podemos definir uma condição de contorno (2.18).

$$V_N(x[N]) = g_N(x[N]). \tag{2.18}$$

Podemos, por fim, reformular o teorema fundamental da programação dinâmica, para o caso estocástico.

Teorema 2.6 (Versão para o caso estocástico do teorema 2.1)
Considere o problema de programação dinâmica determinístico com tempo discreto formulado em (2.15). A função de Bellman (2.17) é dada pela equação recursiva (2.19):

$$V_i(x[i]) = \min_{u[i]} \; E \; [g_i(x[i],u[i],w[i]) + V_{i+1}(x[i+1])] \,, \tag{2.19}$$

com condição de contorno (2.18):

$$V_N(x[N]) = g_N(x[N]).$$

O valor ótimo da função-objetivo (2.16) é $J^* = V_0(x[0])$.

Demonstração:
Análoga à demonstração do teorema 2.1. ■

Se o sistema dinâmico estocástico tem um número finito de estados, pode ser melhor representá-lo por meio de um sistema de equações em termos das probabilidades de transição entre os estados, em vez de considerar sua equação de estados (2.13). Precisamos para isto, listados os estados, conhecer $p_{ij}(u,k)$, ou seja, a probabilidade, no estágio k, do próximo estado $x[k+1]$ ser j, sendo que o estado atual $x[k]$ é i e que o controle $u[k]$ selecionado é u. Este sistema, pode ser descrito pela equação de estados alternativa (2.20).

$$x[k+1] = w[k]. \tag{2.20}$$

Em que a distribuição de probabilidade da variável aleatória $w[k]$ seria: $p_{ij}(u,k) = P\{w[k] = j; x[k] = i, u[k] = u\}$.

Segundo Dano (1975), o processo descrito desta forma é conhecido como processo de decisão de Markov, e o sistema dinâmico desta forma é uma cadeia de Markov. Se o sistema for estacionário, as probabilidades não vão depender do estágio k, portanto podemos suprimi-lo e considerar apenas $p_{ij}(u)$.

Neste caso, o algoritmo apresentado para o caso determinístico, que propõe montar a árvore de possibilidades do problema, pode ser facilmente estendido para o caso estocástico. Este algoritmo consiste em:

Algoritmo Clássico - Programação Dinâmica Estocástica

---

1. Montar a árvore de possibilidades do problema: listar, para cada estágio do problema, os estados possíveis de serem alcançados pelas ações de controle admissíveis;

2. Procurar neste grafo, seqüencialmente por estágios, para cada estado, o caminho mínimo estocástico (baseado no valor esperado) entre este estado e o estado final.

Aqui, os nodos da árvore de possibilidades serão os estados, os arcos serão as ações de controle entre dois estados e o custo de transição entre os estados será ponderado pelas respectivas probabilidades de transição. Reforçamos que este algoritmo só vale quando as variáveis aleatórias assumem apenas um número finito de valores, bem como as variáveis de estado e as de decisão do problema. Caso contrário, este algoritmo pode ser usado via técnicas de discretização.

Vamos apresentar um exemplo, também clássico nesta classe de problemas, o problema de estoque. Vamos resolver este problema procurando na árvore de possibilidades do problema um caminho mínimo estocástico.

Exemplo 2.7 (Problema de estoque)

Considere um problema de estoque simples com apenas $N = 2$ *estágios. As demandas nos dois estágios iniciais* $w[0]$ *e* $w[1]$ *são variáveis aleatórias, sendo que* $w[0]$ *pode ser igual a* 10 *com probabilidade* $p_1 = 0{,}6$ *ou ser igual a* 20 *com probabilidade* $p_1 = 0{,}4$ *e* $w[1]$ *pode ser igual a* 10 *com probabilidade* $p_2 = 0{,}5$ *ou ser igual a* 20 *com probabilidade* $p_2 = 0{,}5$*. Portanto, as compras nos dois estágios iniciais* $u[0]$ *e* $u[1]$ *devem ser em lotes de* 10 *ou de* 20 *unidades. É claro que a quantidade em estoque* $x[k+1]$ *no estágio* $k+1$ *é dada por:*

$$x[k+1] = x[k] + u[k] - w[k].$$

Suponha que o estoque inicial seja nulo $x[0] = 0$*.*

O custo unitário de compra do produto $u[0]$ *é* $d_0 = 1$ *e o custo unitário de compra do produto* $u[1]$ *é* $d_1 = 3$*. Neste exemplo, apenas a quantidade em estoque ao final do processo* $x[2]$ *deve ser penalizada, com* $c_2 = 2$ *por unidade. Deve-se penalizar também a quantidade da demanda não atendida em cada estágio, com* $c_d = 5$ *por unidade. Portanto, a função custo do problema é:*

$$J = \underbrace{d_0 u[0] + c_d \max(0, -x[0])}_{J_1} + \underbrace{d_1 u[1] + c_2 x[2] + c_d \max(0, -x[1])}_{J_2}.$$

Lembramos que o critério de otimização deve ser modificado, devido a presença da aleatoriedade. Assumimos que o objetivo da companhia é minimizar o valor esperado dos gastos com a compra e com a estocagem ao final dos estágios. Para isto, basta considerar como função-objetivo do problema a média ponderada dos custos de cada estágio, com os pesos sendo as respectivas probabilidades das demandas.

Assim, sendo i o indexador das possibilidades do primeiro estágio e j o indexador das possibilidades do segundo estágio, a função-objetivo do problema é:

$$\min E[J] = \min(\sum_i J_1^i p_1^i + \sum_j J_2^j p_2^j).$$

O primeiro passo para a resolução deste problema consiste em montar sua árvore de possibilidades. Vamos sintetizar esta árvore em tabelas por estágios. A tabela do estágio 1 é a tabela 2.2 e a do estágio 2 é a tabela 2.3. Os traços correspondem a soluções infactíveis.

i	$x[0]$	$u[0]$	$w[0]$	$x[1]$	probabilidade p_1	custo $J_1^i p_1^i$
1	0	0	10	-	0,6	$50 \times 0{,}6 = 30$
2	0	10	10	0	0,6	$10 \times 0{,}6 = 6$
3	0	20	10	10	0,6	$20 \times 0{,}6 = 12$
4	0	0	20	-	0,4	$100 \times 0{,}4 = 40$
5	0	10	20	-	0,4	$(10+50) \times 0{,}4 = 24$
6	0	20	20	0	0,4	$20 \times 0{,}4 = 5$

Tabela 2.2: Problema de estoque (exemplo 2.7). Listagem da árvore de possibilidades do estágio 1.

j	$x[1]$	$u[1]$	$w[1]$	$x[2]$	probabilidade p_2	custo $J_2^j p_2^j$
1	0	0	10	-	0,5	$50 \times 0{,}5 = 25$
2	0	10	10	0	0,5	$30 \times 0{,}5 = 15$
3	0	20	10	10	0,5	$(60 + 20) \times 0{,}5 = 40$
4	0	0	20	-	0,5	$100 \times 0{,}5 = 50$
5	0	10	20	-	0,5	$(30 + 50) \times 0{,}5 = 40$
6	0	20	20	0	0,5	$60 \times 0{,}5 = 30$
7	10	0	10	0	0,5	$0 \times 0{,}5 = 0$
8	10	10	10	10	0,5	$(30 + 20) \times 0{,}5 = 25$
9	10	20	10	20	0,5	$(60 + 40) \times 0{,}5 = 50$
10	10	0	20	-	0,5	$50 \times 0{,}5 = 25$
11	10	10	20	0	0,5	$30 \times 0{,}5 = 15$
12	10	20	20	10	0,5	$(60 + 20) \times 0{,}5 = 40$

Tabela 2.3: Problema de estoque (exemplo 2.7). Listagem da árvore de possibilidades do estágio 2.

Feito isto, basta usar o algoritmo recursivo, a partir do estágio final.

O estoque do início do segundo estágio $x[1]$ *pode ser* 0 *ou* 10*. Se* $x[1] = 0$*, a companhia pode comprar 0, 10 ou 20 unidades. Se comprar* $u[1] = 0$*, o custo esperado será* $25+50 = 75$*, se comprar* $u[1] = 10$*, o custo esperado será* $15+40 = 55$ *e se comprar* $u[1] = 20$*, o custo esperado será de* $40 + 30 = 70$*. Portanto, a melhor decisão para* $x[1] = 0$ *é* $u[1] = 10$*. Se* $x[1] = 10$ *e a companhia comprar* $u[1] = 0$*, o custo esperado será* $0 + 25 = 25$ *se a companhia comprar* $u[1] = 10$*, o custo esperado será* $25 + 15 = 40$ *e se a companhia comprar* $u[1] = 20$*, o custo esperado será de* $50 + 40 = 90$*. Portanto, a melhor decisão para* $x[1] = 10$ *é* $u[1] = 0$*.*

Finalmente, o estoque do início do primeiro estágio $x[0]$ *será sempre* 0*. Neste caso, a companhia também pode comprar 0, 10 ou 20 unidades. Se comprar* $u[0] = 0$*, o custo esperado será* $30 + 40 = 70$*, se comprar* $u[0] = 10$*, o custo esperado será* $6 + 24 = 30$ *e se comprar* $u[0] = 20$*, o custo esperado será de* $12 + 5 = 17$*. Portanto, a melhor decisão para* $x[0] = 0$ *é* $u[0] = 20$*.* □

Repare que, em cada estágio e para cada estado, descobrimos qual deve ser a ação de controle ótima. Desta forma, obtemos como resultado do problema, não apenas uma seqüência de decisões ótimas $\{u[0], \ldots, u[N-1]\}$, mas uma seqüência ótima de regras de transição $\{\mu_0, \ldots, \mu_{N-1}\}$, que mapeiam cada $x[k]$ admissível (que é estocástico, pela natureza do problema) no correspondente $u[k]$ ótimo $\mu_k(x[k]) = u[k]$.

Nota 2.3 *Quando o sistema é determinístico, dado um estado inicial e uma seqüência de ações de controle, sabemos previamente qual será o valor de cada estado* $x[k]$*. Desta forma, a otimização pode ser feita em malha aberta, procurando pela seqüência de variáveis de controle* $\{u[0], \ldots, u[N-1]\}$*, como temos formulado. Entretanto, quando o processo é estocástico, não conseguimos saber previamente qual será o valor de cada estado* $x[k]$*, e, desta forma, é necessário fazer a otimização em malha fechada, sobre uma seqüência de leis de controle* $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$*, em que cada* μ_k *mapeia* $x[k]$ *na variável de controle* $u[k] = \mu_k(x[k])$*.*

Lembramos mais uma vez que este procedimento ótimo é muito simples de ser entendido, mas **tem um custo computacional exponencial** em função do número de itens a serem enumerados: variáveis de estado, de controle e número de estágios do problema. O problema é que, enquanto no caso determinístico temos apenas uma possibilidade a ser listada para cada par ($x[k]$,$u[k]$), a saber: $x[k+1] = f(x[k],u[k])$, no caso estocástico, temos tantas opções quanto for a cardinalidade de $w[k]$ para cada par ($x[k]$,$u[k]$). Desta forma, se o algoritmo determinístico já se mostrava proibitivo em muitos casos práticos, este algoritmo estocástico, que envolve um número bem maior de possibilidades, o é muito

mais. Assim como no caso determinístico, este fato é conhecido por "maldição da dimensionalidade".

2.3 Caso contínuo

Considere o problema fundamental de programação dinâmica determinístico com tempo contínuo formulado em (1.4) e reescrito aqui para comodidade do leitor[3].

$$\min_u \int_0^T g(t,x(t),u(t))dt + g_T(x(T))$$

$$\text{sujeito a: } \begin{cases} x'(t) = f(t,x(t),u(t)); \\ t \in [0,T]; \\ x(0) = x_0 \text{ e/ou } x(T) = x^* \text{ dados.} \end{cases}$$

A equação de movimento do problema de tempo contínuo é a (1.2), também reescrita aqui.

$$x'(t) = f(t,x(t),u(t)).$$

Vamos seguir notação semelhante ao caso discreto. Denotaremos o valor ótimo da função-objetivo por J^*:

$$J^* = \min_u J. \tag{2.21}$$

Vamos introduzir a seguinte notação: $u(\tau,T)$ corresponde à função de decisão no intervalo fechado $[\tau,T]$. Consideremos a função-truncada como sendo (2.22).

$$J_\tau(x(\tau),u(\tau,T)) = \int_\tau^T g(s,x(s),u(s))ds + g(T,x(T)). \tag{2.22}$$

Esta equação está bem definida, pois dados $x(\tau)$ e a função $u(\tau,T)$ no intervalo $[\tau,T]$, podemos encontrar o estado no intervalo $[\tau,T]$ resolvendo a equação de movimento (1.2).

Seguindo por analogia com o caso discreto, é fácil ver que a **função de Bellman** para o caso contínuo é a que está em (2.23).

$$V_\tau(x(\tau)) = \min_{u(\tau,T)} J_\tau(x(\tau),u(\tau,T)). \tag{2.23}$$

[3]Também aqui, caso uma das variáveis de estado (inicial ou final) não tenha sido dada, ela pode entrar no problema como variável de otimização.

Note que podemos definir uma condição de contorno (2.24).

$$V_T(x(T)) = g(T,x(T)). \tag{2.24}$$

O teorema 2.8 apresenta uma condição suficiente para a existência da solução do problema de programação dinâmica contínuo.

Teorema 2.8 (Versão para o tempo contínuo do teorema 2.1)
Considere o problema de programação dinâmica determinístico com tempo contínuo formulado em (1.4). A função de Bellman (2.23) satisfaz a equação (2.25):

$$-\frac{\partial V_t(x(t))}{\partial t} = \min_u \left(g(t,x(t),u(t)) + \frac{\partial V_t(x(t))}{\partial x} f(t,x(t),u(t)) \right), \tag{2.25}$$

com condição de contorno (2.24):

$$V_T(x(T)) = g_T(x(T)).$$

O valor ótimo da função-objetivo (2.21) é $J^* = V_0(x(0))$.

Demonstração:
A versão discreta da equação de Bellman é (2.8):

$$V_i(x[i]) = \min_{u[i]} \left(g_i(x[i],u[i]) + V_{i+1}(x[i+1]) \right),$$

com condição de contorno (2.7):

$$V_N(x[N]) = g_N(x[N]).$$

Vamos discretizar o tempo contínuo, ou seja, dividir o intervalo $[0,T]$ em N estágios de tamanho Δt. Assim, se o tempo contínuo τ corresponder ao estágio i, o tempo contínuo $\tau + \Delta t$ corresponderá ao estágio $i+1$. Substituindo esta discretização em (2.8), temos:

$$V_\tau(x(\tau)) = \min_{u(\tau)} \left(\int_\tau^{\tau+\Delta t} g(s,x(s),u(s))ds + V_{\tau+\Delta t}(x(\tau+\Delta t)) \right). \tag{2.26}$$

Fazendo a expansão de Taylor em $V_{\tau+\Delta t}(x(\tau+\Delta t))$, temos:

$$\begin{aligned} V_{\tau+\Delta t}(x(\tau+\Delta t)) &= V_\tau(x(\tau)) + \frac{\partial V_\tau(x(\tau))}{\partial x} x'(\tau)\Delta t + \frac{\partial V_\tau(x(\tau))}{\partial t}\Delta t + TOS = \\ &= V_\tau(x(\tau)) + \frac{\partial V_\tau(x(\tau))}{\partial x} f(\tau,x(\tau),u(\tau))\Delta t + \frac{\partial V_\tau(x(\tau))}{\partial t}\Delta t + TOS. \end{aligned}$$

Onde TOS quer dizer termos de ordem superior em Δt, satisfazendo:

$$\lim_{\Delta t \to 0} \frac{TOS}{\Delta t} = 0.$$

Fazendo a expansão de Taylor na integral, temos:

$$\int_{\tau}^{\tau+\Delta t} g(s,x(s),u(s))ds = g(\tau,x(\tau),u(\tau))\Delta t + TOS.$$

Portanto, (2.26) pode aproximada por:

$$V_\tau = \min_u \left(g\Delta t + V_\tau + \frac{\partial V_\tau}{\partial x} f\Delta t + \frac{\partial V_\tau}{\partial t}\Delta t \right) + TOS.$$

Os parâmetros das funções foram omitidos apenas para melhorar a legibilidade da equação.

Reorganizando os termos, e separando os que dependem da variável de decisão, temos:

$$V_\tau = V_\tau + \frac{\partial V_\tau}{\partial t}\Delta t + \min_u \left(g\Delta t + \frac{\partial V_\tau}{\partial x} f\Delta t \right) + TOS.$$

Donde segue que:

$$-\frac{\partial V_\tau}{\partial t} = \min_u \left(g + \frac{\partial V_\tau}{\partial x} f \right) + TOS. \tag{2.27}$$

Dividindo os dois termos da equação (2.27) por Δt, tomando o limite quando $\Delta t \to 0$ e considerando em cada cada $t = \tau$, obtemos a equação (2.25).

Devido à discretização, a condição de contorno (2.24) vem direto da condição de contorno do caso discreto (2.7), bem como o valor ótimo da função-objetivo (2.21): $J^* = V_0(x(0))$ vem direto o valor ótimo da função-objetivo do caso discreto (2.2): $J^* = V_0(x[0])$. Isto conclui nossa demonstração. ■

Nota 2.4 *A equação diferencial parcial (2.25) é chamada de* ***equação de Hamilton-Jacobi-Bellman*** *ou apenas* ***equação HJB****.*

Nota 2.5 *A problema contínuo (1.4) pode ser resolvido (de maneira aproximada) via árvore de possibilidades, através de sua discretização.*

Exemplo 2.9 (Problema LQ contínuo simples)

Considere o seguinte problema contínuo com sistema dinâmico linear e função-objetivo quadrática:

$$\min_u \int_0^T \frac{1}{2}u(t)^2 dt + \frac{1}{2}x(T)^2$$

$$\textit{sujeito a:} \begin{cases} x'(t) = u(t); \\ t \in [0,T]; \\ x(0) = x_0 \textit{ dado.} \end{cases}$$

A equação HJB desse problema é:

$$-\frac{\delta V_t(x(t))}{\delta t} = \min_u \left(\frac{1}{2}u(t)^2 + \frac{\delta V_t(x(t))}{\delta x}u(t) \right), \tag{2.28}$$

com condição de contorno:

$$V_T(x(T)) = \frac{1}{2}x(T)^2. \tag{2.29}$$

Derivando a função de dentro do parêntesis da equação HJB do problema (2.28) com respeito a u e igualando a zero, obtemos o mínimo, que chamaremos de u^*:

$$u^*(t) = -\frac{\partial V_t(x(t))}{\partial x}. \tag{2.30}$$

Substituindo este u^* na equação HJB do problema (2.28), obtemos:

$$\begin{aligned} -\frac{\partial V_t(x(t))}{\partial t} &= \frac{1}{2}\left(\frac{\partial V_t(x(t))}{\partial x}\right)^2 - \left(\frac{\partial V_t(x(t))}{\partial x}\right)^2 \Rightarrow \\ \frac{\partial V_t(x(t))}{\partial t} &= \frac{1}{2}\left(\frac{\partial V_t(x(t))}{\partial x}\right)^2. \end{aligned}$$

Esta é uma equação diferencial parcial com condição de contorno (2.29). Sua solução é:

$$V_t(x(t)) = -\frac{1}{2(t-T-1)}x(t)^2. \tag{2.31}$$

Tendo encontrado a função V_t, encontramos u^* por meio da equação (2.30), e encontrando u^*, encontramos a função x^* ótima por meio da equação de movimento do problema $x'(t) = u(t)$, com a condição inicial dada $x(0) = x_0$. □

2.4 Caso impulsivo

Em muitas aplicações práticas, gostaríamos que as variáveis de estado contínuas no tempo sofressem influência das variáveis de controle apenas instantaneamente. Já outras aplicações práticas simplesmente não admitem uma ação de controle contínua no tempo, embora o sistema dinâmico autônomo relacionado o seja. Esta maneira de se formular o problema é mais conhecida como **controle impulsivo**. Yang (1999, 2001) apresenta a formalização da teoria e alguns exemplos práticos. Branicky (1995) analisa o controle impulsivo em sistemas híbridos (com dinâmica contínua e discreta).

Seja T o horizonte de otimização do problema. A idéia do controle impulsivo consiste em dividir o intervalo de tempo a ser considerado [0,T] em N estágios a fim de agir no sistema dinâmico impulsivamente apenas nos instantes em que estes estágios se iniciam. Para isto, considere a seqüência de instantes de controle $\Gamma = \{\tau_0, \ldots, \tau_N\}$, em que: $\tau_k < \tau_{k+1}$, $\tau_0 = 0$ e $\tau_N = T$. Não é necessário que esses instantes sejam eqüidistantes.

Uma ação de controle impulsiva no estágio k, que chamaremos de $u[k]$, corresponde a uma ação de controle impulsiva no instante de tempo τ_k. O estado no estágio k, que chamaremos de $x[k]$, corresponde ao estado no instante de tempo τ_k; o estado no estágio inicial $x[0]$ corresponde a $x(\tau_0) = x(0) = x_0$; o estado no estágio final $x[N]$ corresponde a $x(\tau_N) = x(T) = x^*$, etc.

A variável τ_k^+ é definida como o instante de tempo "imediatamente depois" da ação impulsiva em τ_k. Ou seja, dado qualquer $\delta > 0$, τ_k^+ é formalmente definido como o número que cumpra $\tau_k^+ = \tau_k + \epsilon$, para qualquer $0 < \epsilon < \delta$. Como a ação é impulsiva, os dois números τ_k e τ_k^+ correspondem a instantes iguais; pontuamos que o fato de diferenciarmos estas duas variáveis é apenas por uma questão de notação matemática, para melhor explicarmos o efeito da ação impulsiva nas variáveis de estado do problema.

Como o problema é misto, precisamos considerar um sistema de equações de estado, chamado de **equação diferencial impulsiva** (Lakshmikantham et al., 1989): uma equação contínua (referente à evolução dos estados), uma equação discreta (referente à interferência da ação de controle) e uma terceira que relaciona estas outras duas. O sistema dinâmico descrito por uma equação diferencial impulsiva pode ser chamado de sistema dinâmico impulsivo, e vários trabalhos têm discutido a estabilidade destes sistemas (Lakshmikantham et al., 1989; Sun e Zhang, 2003).

A equação de estado desse problema misto está em (2.32).

$$\begin{cases} x'(t) = f_1(t,x(t)); \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k). \end{cases} \tag{2.32}$$

Onde f_1 e f_2 são funções.

No intervalo entre os estágios, a evolução do sistema se dá pela equação diferencial $x'(t) = f_1(t,x(t))$, com condição inicial dada pela equação de diferença $x[k^+] = x[k]+f_2(u[k])$. Portanto, para cada estado inicial, temos N subproblemas de valor inicial para resolver, um para cada estágio do problema. A equação (2.33) mostra o subproblema de valor inicial (PVI) que deverá ser resolvido em cada estágio.

$$\text{PVI do estágio } k\text{: } \begin{cases} x'(t) = f_1(t,x(t)); \\ t \in [\tau_k^+,\tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k). \end{cases} \tag{2.33}$$

Note que cada PVI será resolvido "após" a ação de controle impulsiva, a partir do instante τ_k^+. A condição inicial $x[0] = x_0$ do problema deve ser dada e a condição inicial de cada subproblema $x(\tau_k^+) = x[k^+]$ é função da ação de controle $u[k]$ e do valor final do subproblema anterior $x[k] = x(\tau_k)$. A ação de controle acontece instantaneamente apenas nos estágios discretos $k = 0,1,\ldots,N-1$ e atuará transladando a condição inicial de cada subproblema.

A seqüência de ações de controle $\{u[0],\ldots,u[N-1]\}$ pode ser escrita como uma função contínua no tempo $u(t)$. Para isto, basta usar das chamadas funções-impulso ou funções delta-de-Dirac $\delta(t)$ e considerar $u_k = u[k]$, como em (2.34).

$$u(t) = \sum_{k=0}^{N-1} u_k \delta(t - \tau_k). \tag{2.34}$$

Chamamos de **programação dinâmica impulsiva** à resolução de problemas de controle impulsivo via técnicas de programação dinâmica (discreta ou contínua). Por exemplo, o trabalho de Dar'in et al. (2005) procura resolver problemas de controle impulsivo via programação dinâmica contínua no tempo, por meio da equação HJB (equação 2.25). Em da Cruz et al. (2007b), resolvemos problemas de controle impulsivo via técnicas de programação dinâmica sub-ótima discreta no tempo.

Nesta linha, o problema fundamental de programação dinâmica impulsiva pode ser formulado como em (2.35).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{2.35}$$

$$\text{sujeito a: } \begin{cases} x'(t) = f_1(t,x(t)); \\ t \in [\tau_k^+,\tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

O valor inicial da equação diferencial do subproblema do estágio k é função do valor final da equação de estado do estágio $k-1$ e da ação de controle do estágio k.

Ponto fixo de um sistema dinâmico impulsivo

O ponto fixo[4] de um sistema dinâmico impulsivo está relacionado com o início de cada estágio do problema, instante em que ocorrem as ações impulsivas.

Definição 2.1 *Um ponto fixo $\bar{x}$ do sistema dinâmico impulsivo (2.32) é o ponto em que, para todo estágio k, vale que:*

$$\bar{x} = x[k] = x(\tau_k) = x(\tau_{k+1}) = x[k+1]. \tag{2.36}$$

Note que se os intervalos são eqüidistantes, associado a este ponto-fixo $\bar{x}$, existem duas outras variáveis (fixas): a ação de controle (que chamaremos de $\bar{u}$) e a condição inicial de cada subproblema (que chamaremos de $\bar{x}^+$). Desta forma, para cada estágio k, temos que $x[k^+] = \bar{x}^+$ e $u[k] = \bar{u}$ com $\bar{x}^+ = \bar{x} + f_2(\bar{u})$.

Repare que, definido desta forma, um ponto fixo do sistema impulsivo é um ponto fixo do mapa de primeiro retorno ou mapa de Poincaré (Fiedler-Ferrara e do Prado, 1994) do sistema contínuo, e pode ser analisado como um ponto fixo de um sistema dinâmico discreto.

2.5 Métodos sub-ótimos

Apesar de termos considerado exemplos simples, percebemos nas seções anteriores que encontrar a solução ótima em malha fechada pode ser muito difícil ou mesmo impossível. Ainda quando optamos por resolvê-los numericamente, montando sua árvore de possibilidades, uma vez que este procedimento é simples de ser entendido, mas possui alto custo computacional.

A fim de encontrar uma solução (ainda que não seja ótima) com um custo computacional menos proibitivo, é costume aproximar o problema dinâmico por

[4]Ao longo deste trabalho, consideramos a meta de programação x^* como um ponto fixo do sistema dinâmico autônomo em questão.

outro computacionalmente mais simples. Outra estratégia, costuma ser encontrar algoritmos que aproximam — em algum sentido — o resultado que seria obtido pelo algoritmo clássico da programação dinâmica. O fato é que vários métodos sub-ótimos (aproximativos e heurísticos) vêm sendo propostos para diversas classes de problemas de programação dinâmica[5].

Os métodos aproximativos ou heurísticos procuram relaxar a demanda pela otimalidade na resposta, em troca de um custo computacional mais razoável. Um algoritmo aproximativo (ou de aproximação) apresenta uma cota para o erro da solução, ou seja, tem uma razão de qualidade comprovada matematicamente e/ou uma prova formal de convergência. Uma heurística utiliza informação e intuição a respeito do problema e da sua estrutura para resolvê-lo de forma rápida, mas sem nenhuma garantia de otimalidade. Geralmente, quanto melhor se conhecem as particularidades da classe de problema atacada, melhores são os resultados heurísticos.

Geralmente, os métodos sub-ótimos de programação dinâmica procuram considerar, em cada estágio, **aproximações para a função dinâmica do problema**, que mais eficientemente possa ser calculada e ordenada. A idéia geral consiste em encontrar uma aproximação $\tilde{J}$ para a função de Bellman (equações 2.6, 2.17 ou 2.23, dependendo da classe do problema), que pode ser implícita ou explícita (Bertsekas, 2005):

- **Aproximação implícita** A aproximação $\tilde{J}$ é computada *on-line*, por meio de estratégias específicas.
- **Aproximação explícita** A aproximação $\tilde{J}$ é computada *off-line*, geralmente introduzindo uma arquitetura de aproximação paramétrica, como uma rede neural ou uma soma ponderada de funções-base.

Em geral, os métodos de aproximação implícita, em sua maioria, têm inspiração nos métodos de controle, enquanto que os de aproximação explícita têm mais ligação com a área da ciência da computação.

Nesta seção, apresentamos quatro métodos de aproximação implícita: equivalência à certeza, controle por realimentação em malha aberta, planejamento truncado e controle preditivo[6]. Existem diversas variações destes métodos, vamos apresentar suas versões segundo Bertsekas (1995, 2005). Também comentamos os principais métodos de aproximação explícita: programação dinâmica diferencial e programação neuro-dinâmica. Não pretendemos, com isto, esgotar todo o

[5]Atualmente, constam 724 citações de programação dinâmica aproximada (*approximate dynamic programming*) entre 1964 e 2007 no *Web of Knowledge*; só em 2007 foram 86.

[6]Bertsekas (2005) apresenta o método de estrutura restrita, que dá uma mesma roupagem a estes quatro métodos.

assunto, vamos apenas mostrar alguns exemplos a fim de melhor contextualizar nosso trabalho entre os demais.

Aproximação implícita

O **método de equivalência à certeza** — *certainty equivalent controller (CEC)* — é uma rotina de controle sub-ótimo para programação dinâmica estocástica motivada pelo controle linear-quadrático (LQ). Este método propõe aplicar em cada estágio, o controle que seria ótimo se umas ou todas as variáveis aleatórias fossem fixadas em seus valores esperados. Ou seja, o modelo estocástico é aproximado por um modelo determinístico considerando os valores esperados das variáveis aleatórias.

Este procedimento deve ser executado assim que o valor do estado $x[k] = x_k$ for conhecido, e, em cada estágio, apenas a ação de controle $u[k] = \mu_k(x[k])$ deve ser aplicada. Desta forma, um total de N problemas de otimização determinística deve ser executado em cada processo (um por estágio), só que a um custo computacional bem menor que o custo da montagem da árvore de possibilidades.

Assim, para cada estágio k, se fixarmos as variáveis aleatórias em seus valores esperados, ou seja, se: $\bar{w}[k] = E[w[k]]$, o problema de se encontrar a ação de controle $u[k]$ a ser aplicado pelo método em cada estágio k pode ser formulado como em (2.37).

$$\min_{u[k],\ldots,u[N-1]} \sum_{i=k}^{N-1} g_i(x[i],u[i],\bar{w}[i]) + g_N(x[N]) \tag{2.37}$$

$$\text{sujeito a: } \begin{cases} x[i+1] = f(x[i],u[i],\bar{w}[i]); \\ i = k,\ldots,N-1; \\ x[k] = x_k \text{ dado.} \end{cases}$$

O método de otimização determinística pode ser escolhido de acordo com a natureza do problema, podendo ser qualquer método de otimização estática. Como em problemas determinísticos, a solução usando métodos exatos de otimização estática corresponde à solução dada pelo algoritmo clássico da programação dinâmica (Bertsekas, 1995), em média, o método de certezas equivalentes apresenta bons resultados na prática, com seqüências de ações de controle próximas da ótima. Entretanto, quando o valor esperado não for uma boa previsão para a variável aleatória, o método pode apresentar resultados realmente muito ruins.

Uma alternativa para não se resolver os N problemas *on-line*, seria resolver o problema determinístico *a priori* pelo algoritmo clássico da programação dinâmica determinística, fixando todas as variáveis aleatórias em seus valores

esperados. Infelizmente, como temos comentado, esta alternativa é computacionalmente cara.

O método de **controle por realimentação em malha aberta** — *open-loop feedback control (OLFC)* — leva em conta as incertezas sobre as variáveis aleatórias durante a otimização. A questão é que, como no método de equivalência à certeza, o controle por realimentação em malha aberta deve ser rodado N vezes, e cada subproblema deve ser resolvido usando métodos de otimização estática. Note que a computação será mais complicada, pois o cálculo do custo envolve o valor esperado das variáveis aleatórias.

Em cada estágio k, assim que o valor do estado $x[k] = x_k$ for conhecido, e levando em conta as incertezas sobre as variáveis aleatórias $w[k], \ldots, w[N-1]$, o procedimento para se calcular a ação de controle $u[k] = \mu_k(x[k])$ a ser aplicada no estágio k pode ser formulado como em (2.38).

$$\min_{u[k],\ldots,u[N-1]} E\left[\sum_{i=k}^{N-1} g_i(x[i],u[i],w[i]) + g_N(x[N])\right] \tag{2.38}$$

$$\text{sujeito a: } \begin{cases} x[i+1] = f(x[i],u[i],w[i]); \\ i = k,\ldots,N-1. \end{cases}$$

Uma propriedade interessante do controle por realimentação em malha aberta é que este método conduz a uma política ótima pelo menos tão boa quanto à política do método em malha aberta[7], ou seja, se $J_{\bar{\pi}}$ é o custo da política ótima do controle por realimentação em malha aberta e J_o^* é o custo da política ótima em malha aberta, então $J_{\bar{\pi}} \leq J_o^*$ (Bertsekas, 2005). Entretanto, não sabemos quão perto do ótimo do problema o resultado controle por realimentação em malha aberta se encontra. Por outro lado, o método das equivalência à certeza não apresenta esta propriedade.

Uma maneira prática de se reduzir o custo computacional de um algoritmo de programação dinâmica consiste em truncar o horizonte de tempo e planejar as ações de controle do futuro considerando apenas um número pequeno de estágios. Esta técnica é conhecida como **planejamento truncado** (*limited lookahead policies*).

A possibilidade mais simples de planejamento truncado consiste em usar o passo com apenas um estágio; este algoritmo é conhecido como *rollout*. Por exemplo, no caso de tempo discreto, para o cálculo da variável de controle de cada estágio k, em vez de considerar todo o horizonte de otimização, basta considerar

[7]Na seção 2.7, discutimos melhor sobre a otimização dinâmica sobre o ponto de vista da otimização em malha aberta e em malha fechada.

a seguinte aproximação para a função de Bellman (equação 2.6):

$$V_k(x[k]) = \min_{u[k]} J_k(x[k],u[k,k+1]) \tag{2.39}$$

ou no caso estocástico (equação 2.17):

$$V_k(x[k]) = \min_{u[k]} \text{ E } \left[J_k(x[k],u[k,k+1])\right]. \tag{2.40}$$

Note que, para avaliar a aproximação da função de Bellman, o planejamento truncado pode ser combinado com outros métodos sub-ótimos, como o método das certezas equivalentes, ou o controle por realimentação em malha aberta. O planejamento truncado também conduz a uma política ótima pelo menos tão boa quanto à política em malha aberta (Bertsekas, 2005).

A redução computacional em relação ao método clássico é devida a um número menor de possibilidades que deverá ser verificada em cada estágio. O planejamento truncado considerando mais passos pode ser similarmente definido, levando em conta o *trade-off* entre a precisão do método e o número de passos a serem considerados. De toda forma, esta é uma boa alternativa para o caso de horizonte ilimitado, supondo que se busque uma política estacionária.

O **controle preditivo** — *model predictive control (MPC)* — foi motivado pela necessidade de introduzir não-linearidades e restrições no controle LQ, de forma a obter um sistema em malha fechada sub-ótimo, mas estável — no sentido de que o estado final deva ser zero ou se aproximar de uma vizinhança pequena em torno de zero. Ou seja, a restrição de ponto final é $x[N] = 0$, em que 0 é o vetor nulo, e a vizinhança em torno da meta é suposta sendo poliedral ou elíptica.

O processo do controle preditivo pode ser descrito como: para cada estágio k, e cada cada estado já conhecido $x[k] = x_k$, resolver o problema de programação dinâmica em malha aberta com m estágios (planejamento truncado com m passos), considerando como restrição que o estado final se estabiliza em zero, ou seja $x[i] = 0$, para $i = k+m, \ldots, N$, e implementando apenas o controle $u[k] = \mu_k(x[k])$, descartando os demais.

Neste método, a seqüência de ações de controle pode não ser seqüencialmente consistente, mas a função-objetivo total pelo menos não piora estágio-por-estágio (Bertsekas, 2005).

Aproximação explícita

Uma classe de aproximações explícitas bem conhecida é a chamada **programação dinâmica diferencial** (Jacobson e Mayne, 1970). Esta técnica propõe usar aproximações quadráticas sucessivas para a função dinâmica do problema, num algoritmo iterativo. Com o passar dos anos, esta técnica tem sofrido

diversas melhorias quando aplicadas em problemas específicos, muitas vezes via algoritmos híbridos. Por exemplo, o trabalho de Tospornsampan et al. (2005) apresenta uma combinação da programação dinâmica diferencial discreta com um algoritmo genético (chamado GA-DDDP) para problemas de otimização da operação de sistemas de múltiplos reservatórios.

Outra classe de aproximações relevante é a chamada **programação dinâmica iterativa** (Luus, 1990, 2000). Esta técnica propõe uma maneira iterativa de se fazer a enumeração dos pontos candidatos à solução ótima do problema. Ou seja, procura montar uma seqüência de grades: a partir de uma grade inicial relativamente grosseira, busca redução sistemática no tamanho desta grade até a convergência à solução ótima. O trabalho de Thompson e Cluett (2005) propõe simulação de Monte Carlo para calcular a integral relativa ao valor esperado na função-objetivo de um problema unidimensional de controle ótimo dual a ser resolvido — usando discretização — via programação dinâmica iterativa.

Outra classe de aproximações que vem sendo estudada é a **programação neuro-dinâmica** (Bertsekas e Tsitsiklis, 1996), que usa aproximações para a função dinâmica usando uma rede neural via simulação de Monte Carlo. Para uma dada seqüência de ações de controle, o método procura a seqüência de pesos ótima para cada aproximação. Já a **programação dinâmica nebulosa** (Kacprzyk e Esogbue, 1996) usa sistemas nebulosos para representar as funções e restrições dinâmicas.

O trabalho de Lincoln e Rantzer (2006) também propõe uma maneira de se parametrizar a função dinâmica para uma dada seqüência de ações de controle, conseguindo controlar a distância ao ótimo global do problema. A distância da otimalidade é mantida dentro de limites pré-especificados, e o tamanho desta distância determina a complexidade do problema. Por meio de parametrizações, o problema é resolvido iterativamente. O trabalho ainda apresenta uma série de exemplos computacionais, a saber: considerando sistemas lineares com custos descontínuos e considerando sistemas estocásticos, como controle de estoque e processos de decisão markovianos parcialmente observáveis.

Por fim, destacamos os trabalhos de de Farias (2002); de Farias e Roy (2003b,a), que propõem aproximações via parametrização da função dinâmica usando certas classes de funções lineares, no estilo da regressão linear. Desta forma, métodos de programação linear podem ser usados para encontrar soluções aproximadas, obtendo cotas para o erro cometido. Uma dificuldade seria o elevado número de restrições do problema linear aproximado, que é contornada via simulações de Monte Carlo, trabalhando apenas com amostras tratáveis do conjunto de restrições.

2.6 Programação dinâmica multiobjetivo

Muitos problemas de otimização vindos do mundo real, são compostos por dois ou mais objetivos ou critérios que deveriam ser satisfeitos ao mesmo tempo. Chamamos a estes problemas de otimização multiobjetivo, otimização multicritério ou ainda otimização vetorial (Chankong e Peng, 1983; Miettinen, 1998; Takahashi, 2004). O problema de otimização multiobjetivo com n variáveis de decisão e m funções-objetivo pode ser formulado como em (2.41).

$$\min_{y} J = (J_1(y), \ldots, J_m(y)) \tag{2.41}$$

$$\text{sujeito a: } \begin{cases} g(y) \leq 0; \\ h(y) = 0. \end{cases}$$

Em que: $J : \mathbb{R}^n \to \mathbb{R}^m$ e g e h são funções vetoriais de n variáveis.

O fato é que geralmente estes objetivos são conflitantes, fazendo com que seja necessário redefinir o conceito de otimalidade para esta classe de problemas, trabalhando-se com *trade-offs*, chamados de soluções de compromisso, soluções não dominadas ou **soluções Pareto-ótimas**. Um ponto factível será o ótimo de um problema de otimização multiobjetivo quando não existir outro ponto factível em que se possa melhorar um critério sem tornar outro pior. Matematicamente, se $y \in \mathbb{R}^n$ denota o vetor variáveis de decisão factíveis e $D \subset \mathbb{R}^n$ é o conjunto de todas as soluções y, o conjunto de soluções Pareto-ótimas $Y \subset D$ é caracterizado como em (2.42).

$$\begin{aligned} Y = \{\bar{y} \in D; \neg\exists\ y \in D : J_i(y) \leq J_i(\bar{y}), \\ \forall i = 1, \ldots, m; J(y) \neq J(\bar{y})\}. \end{aligned} \tag{2.42}$$

Assim, cada uma das soluções Pareto-ótimas representa um compromisso entre os critérios de otimização. A imagem deste pontos no espaço de objetivos é chamada de fronteira Pareto-ótima ou **fronteira de Pareto**. O ponto cujas coordenadas correspondem ao mínimo de cada respectivo funcional é conhecido como **solução utópica** e, via de regra, não é uma solução existente. Este ponto funciona como uma referência, visto que a fronteira de Pareto encontra-se contida no hiperparalelogramo definido pela solução utópica e pelas soluções correspondentes aos mínimos individuais de cada funcional.

A comunidade de otimização vem, desde 1950, desenvolvendo metodologias para se resolver problemas de otimização multiobjetivo. A intenção destas metodologias é gerar um conjunto de pontos que descreva de forma representativa a fronteira de Pareto. Uma das mais simples e intuitivas é conhecida como escalarização e consiste em agregar numa função mono-objetivo uma combinação

convexa dos funcionais, via ponderação. Como desvantagens deste método, apresentamos duas: várias ponderações podem estar associadas à mesma solução eficiente (podendo haver computação desnecessária) e o fato de que nos problemas não-convexos não é possível gerar todas as soluções. Uma alternativa tem sido o uso de algoritmos evolucionários, como o NSGA e o NSGA-II (Deb et al., 2000, 2002). Estes algoritmos são métodos heurísticos de busca, baseados nos conceitos de evolução natural, que lidam com uma população de indivíduos que evoluem na direção de regiões melhores de espaço de busca. Por lidar com populações de indivíduos, esses algoritmos permitem que vários membros do conjunto de Pareto sejam gerados de uma só vez. Outra vantagem desses algoritmos é que eles são menos suscetíveis à forma da fronteira de Pareto.

Também os problemas dinâmicos podem ser tratados através de uma abordagem multiobjetivo, chamada de **programação dinâmica multiobjetivo**. Nesta classe de problemas de programação dinâmica, a "maldição da dimensionalidade" costuma ser ainda mais séria e portanto, métodos sub-ótimos são ainda mais importantes e necessários.

Matematicamente, um problema de programação dinâmica multiobjetivo pode ser visto como a otimização de uma função-objetivo vetorial, restrita a um sistema dinâmico. Formalmente, esta função-objetivo é uma função vetorial $J : \mathbb{R}^{N(n+p)} \rightarrow \mathbb{R}^m$, na qual: n é o número de variáveis de estado em um estágio, p é o número de variáveis de controle em um estágio, N é o número de estágios do processo de decisão e m é o número de funções-objetivo do problema. Se o problema for com tempo discreto, podemos escrever cada coordenada de J como o funcional:

$$J_i = J_i(x[0], u[0], \ldots, x[N-1], u[N-1], x[N]).$$

O problema fundamental de programação dinâmica determinística em tempo discreto com m objetivos pode ser formulado como em (2.43).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (J_1, \ldots, J_m) \tag{2.43}$$

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k], u[k]); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

O trabalho de Abo-Sinna (2004) apresenta uma síntese sobre a programação dinâmica multiobjetivo, e propõe resolver esta classe de problemas via sistemas nebulosos. O trabalho de Liao e Li (2002) propõe a solução do caso multiobjetivo via forma adaptativa da programação dinâmica diferencial. O trabalho de Sarkar e Mondak (2005) adapta um problema de controle ótimo (contínuo no tempo) multiobjetivo em química para resolvê-lo via algoritmo NSGA-II.

2.7 Síntese das ferramentas propostas

Procuramos nesta tese, soluções numéricas para problemas de programação dinâmica com ações de controle discretas e horizonte finito.

Todas as ferramentas propostas se baseiam num mesmo paradigma: a idéia de considerarmos a dinâmica do sistema via iteração de conjuntos fechados ao longo dos estágios do problema, em vez de estudarmos a dinâmica através dos arcos de um grafo. A esta abordagem chamamos de **ponto de vista geométrico**. A nossa proposta consiste em definirmos ou escolhermos, de acordo com o problema a ser tratado, uma meta de programação: um ponto que deve ser atingido no estágio final. Feito isto, permitimos uma relaxação de tamanho admissível nesta meta, por meio de um conjunto fechado ao seu redor, e consideramos a pré-imagem deste conjunto no estágio inicial por meio do sistema dinâmico do problema[8]. A figura 2.5 ilustra esta idéia considerando um conjunto politópico.

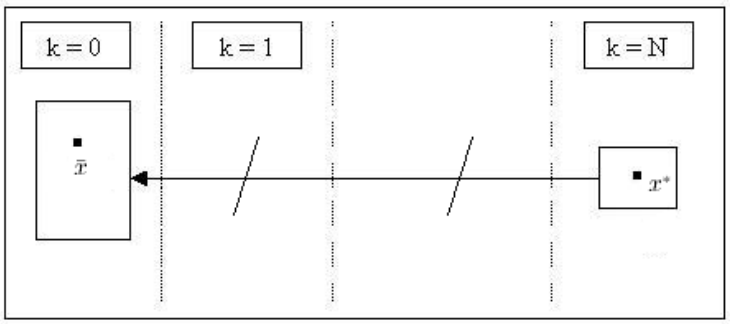

Figura 2.5: Síntese das ferramentas propostas. Ponto de vista geométrico: préimagem no estágio inicial de um hipercubo em torno da meta de programação.

Assim sendo, a otimização pode ser feita usando métodos de otimização estática, exatos ou heurísticos, de acordo com a natureza do problema, considerando como variáveis apenas a seqüência de ações de controle. Conseguimos isto, reescrevendo formalmente a restrição de ponto final em função do estado inicial e da seqüência de ações de controle, por meio da equação de estados do sistema dinâmico. De acordo com a nota 2.2, este procedimento pode ser identificado com o **controle em malha aberta**. O ponto central é que evitamos o caráter enumerativo da solução.

Voltando ao exemplo 2.5, nossa abordagem forneceria apenas a resposta mostrada na figura 2.3, enquanto que a solução pelo algoritmo clássico da programação dinâmica forneceria a resposta mostrada na figura 2.4. Note que a

[8]Este procedimento é semelhante ao proposto em Bertsekas e Rhodes (1971), que considera na otimização, uma bola e um politopo em torno da origem do sistema, a fim de garantir estabilidade de um problema de horizonte ilimitado.

seqüência de ações de controle depende dos valores dos estados, mas na nossa abordagem, esta dependência se está implicitamente, por meio da pré-imagem da meta de programação em um dos estágios do problema.

Como dissemos na nota 2.3, quando o sistema é determinístico, dado um estado inicial e uma seqüência de ações de controle, sabemos previamente qual será o valor de cada estado. Portanto, o resultado da otimização em malha aberta equivale ao da otimização em malha fechada. O problema é que quando o processo é estocástico, não conseguimos saber previamente qual será o valor de cada estado, e desta forma, a otimização deveria ser feita sobre uma seqüência de leis de controle.

Sendo assim, no caso estocástico, a fim de atualizarmos a informação dos estados, propomos que apenas a decisão atual seja implementada, e que nova otimização seja executada em malha aberta a cada estágio, se for o caso. Desta forma, as ferramentas propostas são similares ao controle preditivo e se enquadram nos **métodos sub-ótimos de aproximação implícita** descritos na seção 2.5.

Podemos sintetizar a metodologia aqui proposta desta forma:

Algoritmo Proposto - Versão Geral

1. Encontrar ou definir a meta de programação do problema.

2. Executar a otimização em malha aberta como num problema estático, com o sistema reescrito em função do estado inicial e da seqüência de ações de controle, sujeito à restrição de ponto-final com ou sem a relaxação na meta.

3. Implementar apenas a decisão atual e repetir a otimização no estágio seguinte, se for o caso, atualizando a novas informações sobre os estados.

A complexidade computacional das formulações aqui propostas será a de um problema de otimização estático com p variáveis, sendo p é número de controles possíveis.

No problema determinístico, se as variáveis forem números reais, ao resolvermos pela técnica aqui proposta usando métodos exatos de otimização estática, encontraremos o ponto ótimo (a menos da limitação do computador que só trabalha com um número finito de casas decimais). Se as variáveis forem números inteiros, o procedimento proposto corresponde a permitir a devida relaxação nas variáveis e encontrar um limite inferior para o valor da função-objetivo. Neste caso, nosso procedimento é assintoticamente exato, no sentido que que depende assintoticamente da distância relativa entre o conjunto discreto e sua relaxação.

Entretanto, se quisermos trabalhar com as variáveis inteiras enquanto tais, basta resolvermos a otimização de forma exata via métodos de programação inteira a um custo exponencial, ou, de forma sub-ótima, usando heurísticas. Nos problemas estocásticos, os resultados obtidos pelas nossas ferramentas são sub-ótimos.

As ferramentas propostas foram aplicadas em problemas que tenham como sistemas dinâmicos funções: determinísticas lineares, determinísticas não-lineares e estocásticas lineares. Ao longo dos capítulos desta tese, aprofundamos e discutimos as especificidades das ferramentas em cada classe.

2.8 Conclusões do capítulo

No presente capítulo, apresentamos a técnica da programação dinâmica. Esta técnica consiste em decompor o problema original de otimização dinâmica (com solução geralmente difícil) numa seqüência de problemas possivelmente mais simples de serem resolvidos. Assim, as soluções, obtidas de uma maneira incremental, determinam a estratégia ótima do problema, baseada no Princípio da Otimalidade de Bellman (princípio 2.1). A solução de cada estágio é obtida seqüencialmente, através de um ordenamento baseado na soma do custo presente e do custo futuro já computado. Esta solução iterativa pode ser obtida analiticamente ou numericamente. Consideramos os casos: tempo discreto (determinístico e estocástico), tempo contínuo, e com controle impulsivo.

Entretanto, vimos que estes algoritmos, embora simples de serem entendidos, são proibitivos em muitas aplicações práticas. Por exemplo, a solução analítica um problema de programação dinâmica pode ser muito complicada ou com formulação muito restritiva, e a solução numérica de um problema com um número finito de variáveis sofre da chamada "maldição da dimensionalidade", característica de métodos enumerativos. Em vista disso, métodos sub-ótimos vêm sendo propostos. Neste texto, listamos e discutimos alguns métodos em duas classes: de aproximação implícita (como os métodos em malha aberta, vindos da teoria de controle) e explícita (vindos de uma tradição mais ligada à computação).

Falamos também sobre a programação dinâmica multiobjetivo, que procura um conjunto de políticas que atenda, em certo sentido, a diversos critérios simultaneamente.

Além disso, apresentamos uma visão geral dos métodos que estão propostos neste texto. Vimos que os métodos propostos que encaixam como métodos subótimos de aproximação implícita, similares ao controle preditivo.

Capítulo 3

Ferramentas para Programação Dinâmica Linear

Neste capítulo, apresentamos as ferramentas para programação dinâmica cujo sistema dinâmico associado é linear (não necessariamente a função-objetivo e as demais restrições precisam ser lineares). A seção 3.1 apresenta a motivação e a formulação do problema de programação dinâmica linear. Apresentamos na seção 3.2, as ferramentas já discutidas na dissertação de mestrado (Cardoso, 2005), acrescentando a abordagem multiobjetivo (subseção 3.2.3). Dois estudos de caso são considerados: uma aplicação da abordagem multiobjetivo na otimização dos recursos obtidos com a montagem de um rebanho de gado (subseção 3.3.1)[1] e a otimização do projeto de redes de distribuição (subseção 3.3.2). A seção 3.4 apresenta as conclusões do capítulo.

3.1 Introdução

Considere uma função-objetivo linear. O problema de programação dinâmica linear, determinística, com tempo discreto pode ser formulado como em (3.1).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (c_k^T x[k] + d_k^T u[k]) + c_N^T x[N] \tag{3.1}$$

$$\text{sujeito a: } \begin{cases} x[k+1] = Ax[k] + Bu[k]; \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

[1] Apresentamos a versão mono-objetivo deste estudo de caso no trabalho de mestrado (Cardoso, 2005).

Nesta formulação, os custos unitários c_k e d_k são vetores-colunas (para cada estágio k), e A e B são matrizes de dimensão apropriada.

A equação de estado deste problema é a (3.2).

$$x[k+1] = Ax[k] + Bu[k]. \tag{3.2}$$

Repare que se as variáveis de controle são identicamente nulas em cada estágio do problema, então todos os estados do sistema dependem apenas do estado inicial. Do mesmo modo, se o estado inicial é zero, então obtemos os demais estados do sistema em cada estágio exclusivamente em função das variáveis de controle. Ou seja, o sistema dinâmico linear é separável.

Princípio 3.1 (Princípio da Superposição) *A seqüência de estados de um sistema separável pode ser obtida em duas etapas distintas: em função do estado inicial (supondo as variáveis de controle nulas) e em função das variáveis de controle (supondo o estado inicial nulo). O resultado final será a soma destas duas respostas.*

Portanto, podemos estudar a seqüência de estados da parte autônoma do sistema linear separadamente da ação das variáveis de controle. O resultado final será a soma destas duas respostas. No caso do sistema dinâmico linear autônomo, cada iteração nada mais é que uma transformação linear, e se a matriz da transformação for não-singular, cada iteração é uma mudança de coordenadas. Além disso, cada ação de controle é uma translação. Portanto, cada iteração é uma transformação afim, ou seja, uma transformação linear composta com uma translação.

3.2 Formulações propostas

Queremos trabalhar com o sistema em malha aberta. Desta forma, propomos considerar, em apenas um dos estágios do problema, a pré-imagem de cada estado pensado como a pré-imagem de um conjunto por meio de uma função contínua. Isto quer dizer que a dinâmica do sistema será considerada pela pré-imagem de conjuntos, em vez de pelos arcos de um grafo representando a transição de estados. Isto pode ser feito através de manipulação algébrica das equações destes conjuntos.

Matematicamente, precisamos fazer a iteração das variáveis de estado ao longo dos estágios para o mesmo estágio, por exemplo, o estágio inicial, nas restrições e na função objetivo do problema. A proposição 3.1 mostra como escrever o estado em qualquer estágio K em função do estado inicial $x[0]$ e da seqüência de controles anteriores, entre o estágio inicial e o estágio $K-1$.

Proposição 3.1 *Considere o sistema dinâmico com a ação de controle como em (3.2). Podemos escrever x[K], em cada estágio K, em função apenas do estado inicial x[0] e da seqüência dos controles anteriores, desta forma:*

$$x[K] = A^K x[0] + Bu[K-1] + \sum_{k=1}^{K-1} A^k Bu[K-(k+1)]. \tag{3.3}$$

Demonstração:
Vamos mostrar por indução em K.
Quando $K = 1$, por (3.2), temos que:

$$x[1] = Ax[0] + Bu[0].$$

Suponha, agora que a tese seja válida para até o estágio $n = K$.

$$x[n] = A^K x[0] + Bu[n-1] + \sum_{k=1}^{n-1} A^k Bu[n-(k+1)].$$

Esta é a hipótese de indução. Vamos mostrar que, então, a tese também vale para o estágio $n+1$. Também por (3.2), temos que:

$$x[n+1] = Ax[n] + Bu[n].$$

Substituindo a hipótese de indução na equação acima, temos que:

$$x[n+1] = A(A^n x[0] + Bu[n-1] + \sum_{k=1}^{n-1} A^k Bu[n-(k+1)]) + Bu[n].$$

Daí, segue que:

$$x[n+1] = A^{n+1} x[0] + ABu[n-1] + \sum_{k=2}^{n} A^k Bu[n+1-(k+1)]) + Bu[n].$$

Ou seja:

$$x[n+1] = A^{n+1} x[0] + Bu[n] + \sum_{k=1}^{n} A^k Bu[n+1-(k+1)]),$$

como queríamos mostrar. ■

Podemos melhorar o resultado da otimização (o tempo de execução e o valor da função-objetivo) se pudermos aproximar o problema por outro semelhante com

região factível maior. A chave para isto consiste substituir a meta de programação x^* por conjunto fechado de tamanho admissível ao seu redor.

A metodologia aqui proposta é a seguinte:

Algoritmo Proposto - Programação Dinâmica Linear

1. Permitir a relaxação na meta de programação, que será substituída por um conjunto fechado ao seu redor.

2. Fazer o retorno deste conjunto estágio por estágio, para o estágio inicial do problema pelo sistema autônomo.

3. Fazer a otimização restrita a este conjunto transladado pela seqüência de ações de controle, a fim de encontrar a solução do problema.

Note que a região factível do problema será o resultado da iteração do conjunto ao redor da meta de programação por meio do sistema autônomo (uma transformação linear), transladado pela seqüência de ações de controle. Pelo Princípio da Superposição (princípio 3.1), a soma desses conjuntos é a soma algébrica das equações que o representam.

Estamos considerando as variáveis do problema como números reais. Respeitando, claro, a limitação da precisão do computador, a solução encontrada pelo procedimento sem relaxação na meta será a ótima, visto que num problema determinístico a solução em malha aberta equivale à solução em malha fechada (Bertsekas, 1995). Se as variáveis forem números inteiros, o procedimento proposto corresponde a permitir a devida relaxação nas variáveis e encontrar um limite inferior para a função-objetivo. Entretanto, se quisermos trabalhar com as variáveis inteiras enquanto tais, basta acrescentarmos na formulação (3.1) a restrição de integralidade para as variáveis[2].

3.2.1 Otimização restrita à pré-imagem da meta

Vamos considerar o caso da otimização da função objetivo restrita à pré-imagem da meta da programação dinâmica x^*. A pré-imagem da meta no estágio

[2]Só que, neste caso, teríamos um problema de programação inteira, que requer uma solução de complexidade não-polinomial em função de suas variáveis (Goldbarg e Luna, 2000). Assim, resolver o problema com variáveis inteiras via metodologia proposta e métodos de programação inteira, pode não ter nenhum ganho computacional em relação a resolvê-lo pelo algoritmo ótimo da programação dinâmica discreta. A menos que usemos heurísticas, como fazemos no estudo de caso deste capítulo (seção 3.3.2).

inicial pelo sistema autônomo é um ponto (se a matriz A for não-singular) ou uma variedade linear - retas, planos, hiperplanos (caso contrário). Como o sistema é separável, aplicar a ação de controle consiste em transladar a pré-imagem da meta por meio das ações de controle de cada estágio.

Pela proposição (3.1), é possível escrever $x[N]$, o estado no estágio N, em função apenas do estado inicial $x[0]$ e da seqüência dos controles anteriores. Como queremos no estágio final $x[N] = x^*$, a restrição do sistema dinâmico pode ser escrita como em (3.4)[3].

$$A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] = x^*. \tag{3.4}$$

De fato, a restrição do sistema dinâmico (3.4) é a soma da restrição do sistema sem a ação de controle com a seqüência de controles iteradas pelo sistema autônomo, como diz o Princípio da Superposição (princípio 3.1).

Podemos, então, formular este problema como em (3.5).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (c_k^T x[k] + d_k^T u[k]) + c_N^T x[N] \tag{3.5}$$

$$\text{sujeito a: } \begin{cases} A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] = x^*; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Podemos resolver este problema de otimização estática via algoritmos de pontos interiores (Gonzaga, 1990), por exemplo, a um custo computacional polinomial da ordem do número de restrições e com um número de variáveis igual ao número de variáveis de controle. Só que esta formulação pode trazer consigo as possíveis dificuldades de um problema com restrição de igualdade: em alguns casos, a região factível do problema com restrição de igualdade pode ficar muito pequena e encontrar um ponto factível pode ser muito difícil.

3.2.2 Otimização restrita à pré-imagem de um hipercubo

Vamos considerar a otimização em malha aberta, permitindo uma relaxação politópica na meta de programação x^*. A idéia consiste em considerar um hipercubo em torno da meta de programação (veja figura 2.5).

Um hipercubo P de raio ϵ em torno de uma meta x^* no estágio N pode ser escrito como o conjunto dos pontos $x[N]$ tais que:

$$||x[N] - x^*||_\infty \leq \epsilon \quad \text{ou} \quad x_i[N] = x_i^* \pm e_i\epsilon, \text{ para cada } i, \tag{3.6}$$

[3]Esta restrição é conhecida como restrição de controlabilidade (Chen, 1999).

em que e_i é o i-ésimo vetor da base canônica do $\mathbb{R}^n$ e n é a dimensão de $x[N]$.

Como o sistema é linear, a pré-imagem deste hipercubo no estágio $k = 0$ pelo sistema é um politopo (pré-imagem pelo sistema autônomo) transladado pela seqüência de ações de controle. Usando o Princípio da Superposição (princípio 3.1), a equação da região factível será a soma da equação da pré-imagem do hipercubo pelo sistema autônomo somada com as pré-imagens das ações de controle. Portanto, a restrição do sistema dinâmico pode ser escrita como em (3.7).

$$\begin{cases} e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* + e_i\epsilon)) \leq 0, \text{ para cada } i; \\ -e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* - e_i\epsilon)) \geq 0, \text{ para cada } i. \end{cases} \tag{3.7}$$

Este problema pode ser formulado como em (3.8).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (c_k^T x[k] + d_k^T u[k]) + c_N^T x[N] \tag{3.8}$$

$$\text{sujeito a: } \begin{cases} e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* + e_i\epsilon)) \leq 0, \text{ para cada } i; \\ -e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* - e_i\epsilon)) \geq 0, \text{ para cada } i; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Podemos, também aqui, resolver este problema através de métodos de pontos interiores com a complexidade computacional polinomial. Além disso, como a região factível é possivelmente maior, esta formulação pode conseguir uma função custo consideravelmente menor, num tempo computacional menor, apesar de não garantir mais que o estado final atinja a meta x^*.

Note que o raio da relaxação na meta ϵ pode ser visto como um parâmetro, que define um *trade-off* entre o valor da função-objetivo e a precisão na meta de programação do problema: se este parâmetro cresce, a meta de programação se torna menos satisfeita, mas o valor da função-objetivo é possivelmente menor. Mostramos dois comportamentos típicos deste *trade-off* nas figuras 3.1, que considera uma função-objetivo linear, e 3.2, que considera uma função-objetivo quadrática. Note que principalmente no segundo caso, a relaxação na meta se mostra uma boa alternativa, pois uma pequena redução na precisão da meta (valor do parâmetro) permite uma considerável redução no valor da função-objetivo.

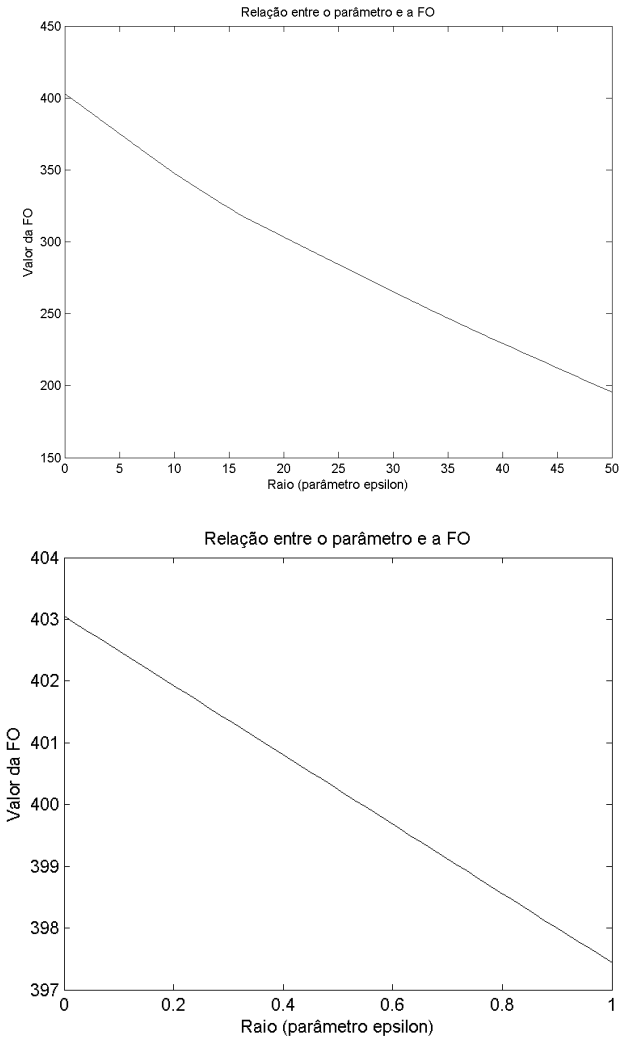

Figura 3.1: *Trade-off* entre o raio da relaxação ϵ e o valor de uma função-objetivo J linear. O segundo gráfico é um *zoom* do primeiro.

3.2.3 Abordagem multiobjetivo

Vimos na subseção 2.6, que a programação dinâmica multi-objetivo diz respeito aos problemas dinâmicos compostos por vários objetivos (ou critérios) que deveriam ser satisfeitos ao mesmo tempo. Nesta subseção, estendemos para o caso multiobjetivo as duas formulações propostas nas subseções 3.2.1 e 3.2.2 para problemas dinâmicos lineares.

A equação (3.9) apresenta a formulação multiobjetivo para a otimização res-

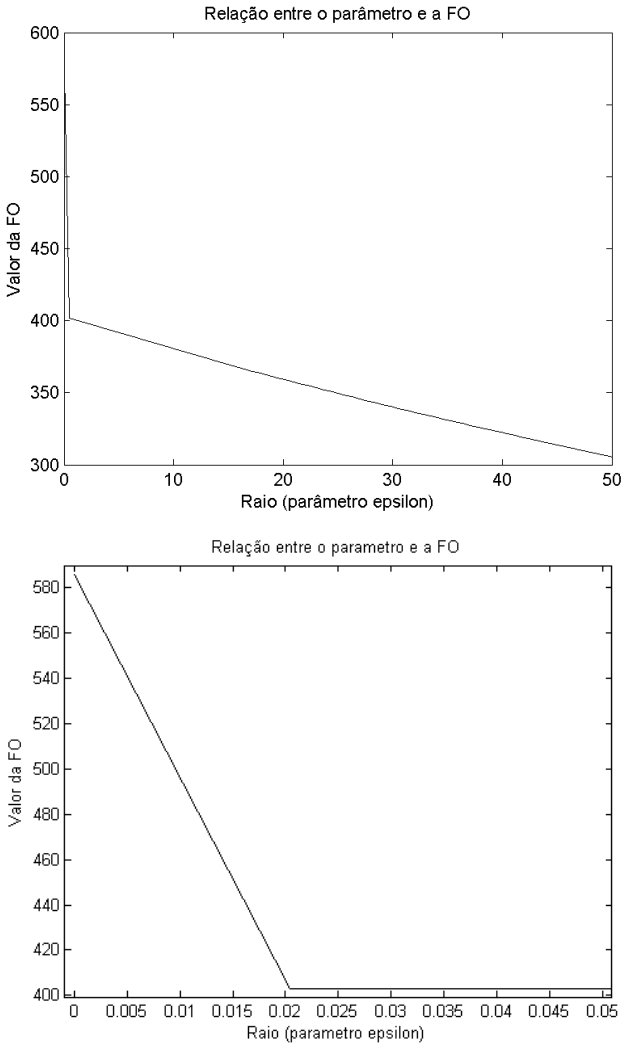

Figura 3.2: *Trade-off* entre o raio da relaxação ϵ e o valor de uma função-objetivo J quadrática (não-linear). O segundo gráfico é um *zoom* do primeiro.

trita à pré-imagem da meta de programação.

$$\min_{u[0],\ldots,u[N-1]} J = (J_1, \ldots, J_m) \tag{3.9}$$

$$\text{sujeito a: } \begin{cases} A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] = x^*; \\ x[0] \text{ pode ser dado.} \end{cases}$$

E a equação (3.10) apresenta a formulação multiobjetivo para a otimização restrita à pré-imagem de um hipercubo de raio ϵ (dado) em torno da meta de

programação.

$$\min_{u[0],\ldots,u[N-1]} J = (J_1, \ldots, J_m) \tag{3.10}$$

$$\text{sujeito a: } \begin{cases} e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* + e_i\epsilon)) \leq 0, \text{ para cada } i; \\ -e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* - e_i\epsilon)) \geq 0, \text{ para cada } i; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Temos considerado funções-objetivos lineares. Portanto, cada funcional linear J_i pode ser escrito da forma mostrada na equação (3.11).

$$J_i = \sum_{k=0}^{N-1} (c_{ik}^T x[k] + d_{ik}^T u[k]) + c_{iN}^T x[N]. \tag{3.11}$$

Neste caso, em que as funções-objetivo, bem como as demais restrições são lineares, podemos resolver este problema em malha aberta, via algoritmos de programação linear multiobjetivo. Existem métodos específicos para esta classe de problemas, como o simplex multiobjetivo (Zenely, 1974). Usando o fato de que o problema é linear, portanto, convexo, outra alternativa seria encontrar uma parte significativa do conjunto de Pareto via escalarização.

3.3 Estudos de caso

3.3.1 Evolução de um rebanho de gado

Este estudo de caso foi inspirado no artigo de Takahashi et al. (1997), que tratou do caso autônomo. O caso não-autônomo mono-objetivo está na dissertação de mestrado de Cardoso (2005), que apresenta todos os detalhes deste problema, e no artigo de Cardoso e Takahashi (2005). No presente estudo[4], tratamos o caso não-autônomo multi-objetivo.

Este estudo de caso está relacionado com a evolução de um rebanho de gado no tempo, cujo comportamento médio pode ser modelado por um sistema dinâmico linear discreto no tempo. Sabemos que depois de um tempo o rebanho adquire estabilidade, no sentido de que o número de cabeças permanece aproximadamente constante. Usando este modelo, propomos formulações de otimização da aquisição do rebanho (até atingir a estabilidade) procurando a política ótima de compra e venda de cabeças de gado que tenha o menor custo[5].

[4]Este estudo de caso foi submetido para publicação (Cardoso et al., 2008c).

[5]Esta mesma abordagem pode ser empregada em diversos tipos de rebanhos com diferentes funções-objetivo e restrições.

Modelagem Matemática

Matematicamente, cada estágio k do problema corresponde a 1 ano e cada variável de estado é um vetor $x[k]$, para cada estágio k, com oito posições. Cada uma das posições representa uma faixa etária de cada um dos sexos, como descrito abaixo. O valor de cada posição representa a quantidade de indivíduos desta faixa: $x_1[k]$: número de fêmeas entre 0 e 1 ano no estágio k;
$x_2[k]$: número de fêmeas entre 1 e 2 anos no estágio k;
$x_3[k]$: número de fêmeas entre 2 e 3 anos no estágio k;
$x_4[k]$: número de fêmeas acima de 3 anos com bezerros no estágio k;
$x_5[k]$: número de fêmeas acima de 3 anos sem bezerros no estágio k;
$x_6[k]$: número de machos entre 0 e 1 ano no estágio k;
$x_7[k]$: número de machos entre 1 e 2 anos no estágio k;
$x_8[k]$: número de machos entre 2 e 3 anos no estágio k. Tendo em vista a estabilidade do sistema, todos os machos acima de 3 anos são descartados, assim como uma porcentagem R do número de fêmeas, a ser calculada.

As variáveis de controle do problema são os vetores $u[k]$, para cada estágio k, com até oito posições, correspondendo à faixa etária e ao sexo que se quer negociar (cada posição representa uma faixa semelhante à descrita acima para as variáveis de estado).

Como o número de cabeças é alto, e tendo em vista a metodologia aqui proposta, justificamos a relaxação na integralidade das variáveis. Assim, ao resolver através das formulações propostas encontraremos como resultado uma cota inferior para a função-objetivo do problema com variáveis inteiras. Note que quanto maior for o número de cabeças de gado do problema, maior seria o número de possibilidades a serem listadas pelos métodos enumerativos tradicionais, o que dificultaria encontrar a solução ótima. Por outro lado, quanto maior for o número de cabeças de gado do problema, menor será o erro relativo do resultado aqui obtido.

O sistema dinâmico com a ação de controle será modelado pela equação:

$$\begin{bmatrix} x_1[k+1] \\ x_2[k+1] \\ x_3[k+1] \\ x_4[k+1] \\ x_5[k+1] \\ x_6[k+1] \\ x_7[k+1] \\ x_8[k+1] \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 0 & m_3f_3/2 & 0 & (m_4-R)f/2 & 0 & 0 & 0 \\ m_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & m_3f_3 & 0 & (m_4-R)f & 0 & 0 & 0 \\ 0 & 0 & m_3(1-f_3) & (m_4-R) & (m_4-R)(1-f) & 0 & 0 & 0 \\ 0 & 0 & m_3f_3/2 & 0 & (m_4-R)f/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & m_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \\ x_3[k] \\ x_4[k] \\ x_5[k] \\ x_6[k] \\ x_7[k] \\ x_8[k] \end{bmatrix} +$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1[k] \\ u_2[k] \\ u_3[k] \\ u_4[k] \\ u_5[k] \\ u_6[k] \\ u_7[k] \\ u_8[k] \end{bmatrix};$$

eur pode ser reescrita na na forma sintética:

$$x[k+1] = Ax[k] + Bu[k]. \tag{3.12}$$

A matriz B acima é a matriz identidade, indicando que o controle será feito em todas as coordenadas de $u[k]$ (podemos negociar qualquer faixa-etária de qualquer sexo do rebanho). Se quisermos fazer o controle em coordenadas específicas, por exemplo, apenas na coordenada m de $u[k]$, basta fazermos a posição $B_{mm} = 1$ e as demais posições de B iguais a zero ($B_{ij} = 0$ quando $(i,j) \neq (m,m)$). Tendo feito isto, podemos considerar (computacionalmente) apenas a coluna m da matriz identidade em vez de toda a matriz e do mesmo modo, considerar como variável somente a coordenada m de cada vetor $u[k]$.

Os parâmetros do modelo são:

- m_1 : um menos a taxa de mortalidade para a faixa etária de 0 a 1 ano;

- m_2 : um menos a taxa de mortalidade para a faixa etária de 1 a 2 anos;

- m_3 : um menos a taxa de mortalidade para a faixa etária de 2 a 3 anos;

- m_4 : um menos a taxa de mortalidade para a faixa etária acima de 3 anos;

- f : taxa de natalidade para vacas acima de 3 anos;

- f_3 : taxa de natalidade para vacas com 3 anos;

- R : taxa de descarte anual das fêmeas acima de 3 anos.

Para que o sistema dinâmico autônomo seja estável, a matriz A do problema deverá ter raio espectral (máximo do valor absoluto dos autovalores da matriz) menor ou igual a 1. Podemos obter a matriz A com raio espectral igual a 1, variando o parâmetro R. O autovetor associado a este autovalor unitário será a meta de programação do problema, x^*, escolhida de forma que a soma das coordenadas de x^* (quantidade total de cabeças ao final do processo) esteja tão próximo quanto se queira da capacidade final da fazenda (dada) L. Como x^* será um ponto fixo do problema autônomo, então $A^k x^* = x^*$, para todo estágio k, ou seja, o rebanho adquire estabilidade.

Otimização

O problema a ser formulado é o de encontrar o conjunto de políticas ótimas que minimizem (do ponto de vista da otimização multiobjetivo) o custo inicial de aquisição do rebanho (que será chamado de J_1) e o número total de indivíduos a serem comprados ao longo dos estágios do problema (que será chamado de J_2). Assim, o estado inicial $x[0]$ também será uma variável de decisão. Lembramos que nas formulações que serão propostas, o sistema dinâmico (3.12) será reescrito em função do estado inicial e da seqüência de ações de controle anteriores, como explicado na seção 3.2.3.

Para as funções-objetivo J_1 e J_2, vamos considerar c um vetor-coluna de 8 posições que associa aos indivíduos associados à posição x_i o custo unitário médio c_i de cada um. Vamos considerar também vetores-colunas d_k de 8 coordenadas, com cada posição i tendo o mesmo valor de c_i nas faixas onde se quer fazer o controle e zero nas demais (esta é uma outra maneira de se decidir em quais coordenadas fazer o controle, sem mexer na matriz B).

As restrições deste problema são:

1. Os valores das coordenadas do vetor $x[k]$ deverão ser não-negativos em cada estágio k.

2. Existe uma limitação no espaço da fazenda ao final do processo, que tem uma capacidade para L cabeças de gado.

3. Depois de um horizonte de N anos o rebanho deve ter atingido um estado de estabilidade com uma população que oscile menos que $\epsilon\%$, para um ϵ dado.

4. Cada variável de controle está limitada entre um valor mínimo e um valor máximo.

Este problema pode, portanto, ser formulado como em (3.13).

$$\min_{x[0],u[0],\ldots,u[N-1]} (J_1 = c^T x[0], J_2 = \sum_{k=0}^{N} d_k^T u[k]) \tag{3.13}$$

$$\text{sujeito a: } \begin{cases} A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] = x^*; \\ x[0] \geq 0; \\ c^T x^* = L; \\ u_{min} \leq u_i[k] \leq u_{max}. \end{cases}$$

Poderíamos também permitir a relaxação na meta de programação do problema. A formulação otimizando via pré-imagem de uma bola de raio ϵ dado pode ser como em (3.14).

$$\min_{x[0],u[0],\ldots,u[N-1]} (J_1 = c^T x[0], J_2 = \sum_{k=0}^{N} d_k^T u[k]) \tag{3.14}$$

$$\text{sujeito a: } \begin{cases} e_i^T (A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* + e_i\epsilon)) \leq 0, \text{ para cada } i; \\ -e_i^T (A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad -(x^* - e_i\epsilon)) \geq 0, \text{ para cada } i; \\ x[0] \geq 0; \\ c^T x^* = L; \\ u_{min} \leq u_i[k] \leq u_{max}. \end{cases}$$

Vamos resolver usando a formulação (3.14), com relaxação na meta. Os dados deste exemplo são de um típico rebanho de gado de corte zebu da região de Minas Gerais (Takahashi et al., 1997). Os parâmetros do modelo são: $m_1 = 0{,}95$, $m_2 = 0{,}97$, $m_3 = 0{,}98$, $m_4 = 0{,}98$, $f = 0{,}7$, $f_3 = 0{,}625$. O tempo necessário para que o sistema entre em regime de estabilidade é $N = 7$ anos. A taxa de descarte anual, calculada pelo método da bisseção é $R = 0{,}1781$. Com estes parâmetros, montamos a matriz A do problema, cujo maior autovalor é 1.

Vamos permitir uma ação de controle em cada faixa, ou seja, em cada uma das coordenadas dos vetores. Em outras palavras, a matriz B será a identidade. Os vetores c e d_k, ou seja, os custos unitários por faixa etária, são da forma $[\ 0{,}3 \quad 0{,}8 \quad 1 \quad 1 \quad 0{,}8 \quad 0{,}2 \quad 0{,}5 \quad 0{,}8\ ] \in \mathbb{R}^8$. Cada coordenada da variável de controle $u_i[k]$ poderá variar entre -20 (vender 20 cabeças) e $+20$ (comprar 20 cabeças) e cada coordenada da variável $x[0]$ poderá variar entre 0 e 200 cabeças.

Precisamos encontrar a meta de programação x^*, autovetor de A associado ao autovalor unitário (um ponto-fixo do sistema autônomo). O tamanho da fazenda

é de $L = 1000$ cabeças de gado. Para obedecer à restrição 2, queremos a soma dos valores das coordenadas de $x[k]$ seja menor ou igual a L, para cada estágio k. Em particular, queremos que no estágio final a soma das coordenadas de $x[N]$ seja bem próximo de L. Se calcularmos x^* de modo que a soma dos valores das coordenadas de x^* seja L, esta restrição não precisará ser considerada. Assim, a meta de programação x^* deste problema (já com o arrredondamento) é:

$$x^* = [\ 98 \quad 94 \quad 91 \quad 195 \quad 242 \quad 98 \quad 93 \quad 90\ ].$$

Este problema multiobjetivo foi resolvido via escalarização, pois as funções-objetivo e as restrições são todas lineares, portanto, convexas. Cada solução escalar foi obtida via função `linprog` do Matlab, que usa métodos de pontos interiores.

A figura 3.3 mostra uma pequena amostra da fronteira de Pareto para diferentes valores do parâmetro de relaxação na meta de programação, que são: $\epsilon = 0, 10, 50, 100, 150$ e 200. Note o *trade-off* entre a precisão na meta (valor de ϵ) e a redução no valor da função-objetivo (as curvas se deslocam para a esquerda). Note também, para cada valor de parâmetro, o *trade-off* entre o gastar mais no início, para ter um retorno ao longo do tempo (caso com maior custo inicial), ou gastar sistematicamente ao longo dos estágios (caso com menor custo inicial).

Consideremos o erro admissível após $N = 7$ anos como sendo $\epsilon = 1$. A figura 3.3 mostra em detalhe a fronteira de Pareto para este caso. Vamos mostrar a evolução do rebanho de gado no tempo e a correspondente ação de controle ótima para dois pontos escolhidos dentre este conjunto, situados em extremos distintos da fronteira: um que prioriza gastar de maneira uniforme ao longo do processo, e outro que prioriza a aquisição inicial, e portanto usufrui de um retorno ao longo dos estágios.

A figura 3.4 apresenta os gráficos para o ponto do lado esquerdo da fronteira de Pareto, com valores das funções correspondentes ao ponto (19,404), isto é, com custo inicial menor e custo ao longo do processo maior. O custo inicial envolve 64 cabeças e o custo ao longo do processo envolve 594 cabeças. As tabelas 3.1 e 3.2 apresentam, respectivamente, a quantidade de cabeças no pasto (variáveis de estado) e a quantidade negociada (variáveis de controle) em cada faixa (coordenada) e em cada estágio do processo.

A figura 3.5 apresenta os dados para o ponto do lado direito da fronteira de Pareto, com valores das funções correspondentes ao ponto $(798, -582)$, com custo inicial maior e custo ao longo do processo menor. O custo inicial envolve 1063 cabeças e ao longo do processo 679 cabeças são vendidas. As tabelas 3.3 e 3.4 apresentam, respectivamente, a quantidade de cabeças no pasto e a quantidade negociada em cada faixa e em cada estágio.

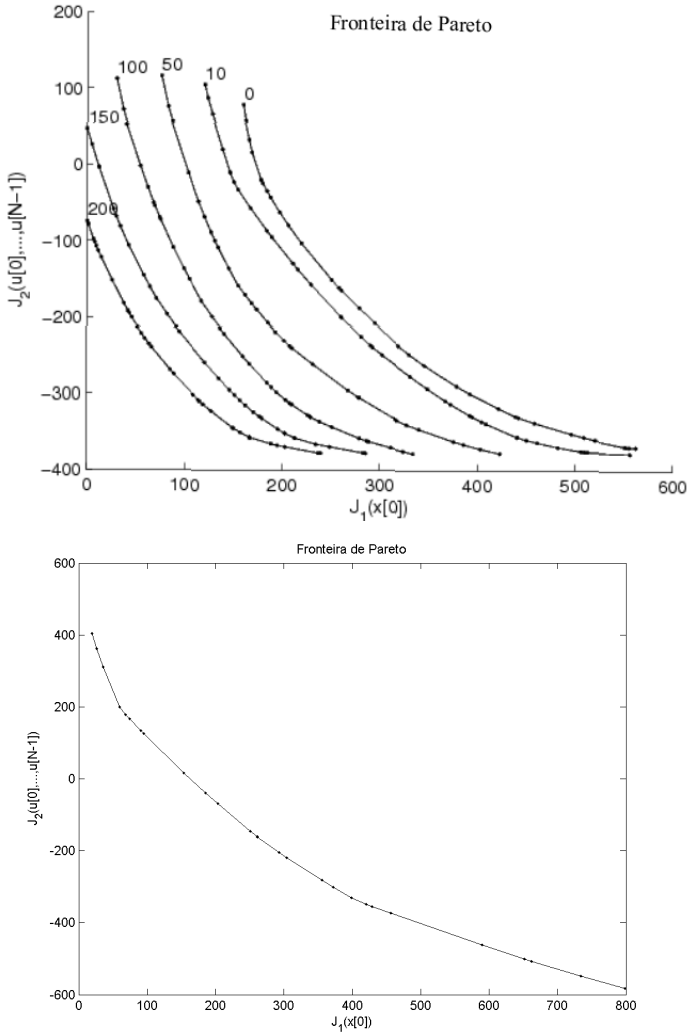

Figura 3.3: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Fonteira de Pareto do exemplo do problema do rebanho de gado bi-objetivo para diversos parâmetros de relaxação $\epsilon = 0, 10, 50, 100, 150$ e 200 (cada curva corresponde a um valor de parâmetro). Detalhe para a curva referente ao valor de parâmetro $\epsilon = 1$.

Observamos que o tempo de execução computacional para se montar uma amostra da fronteira de Pareto contendo 10.000 pontos foi em torno de 530 segundos[6].

[6]Todas as simulações mostradas nesta tese foram rodadas num mesmo computador.

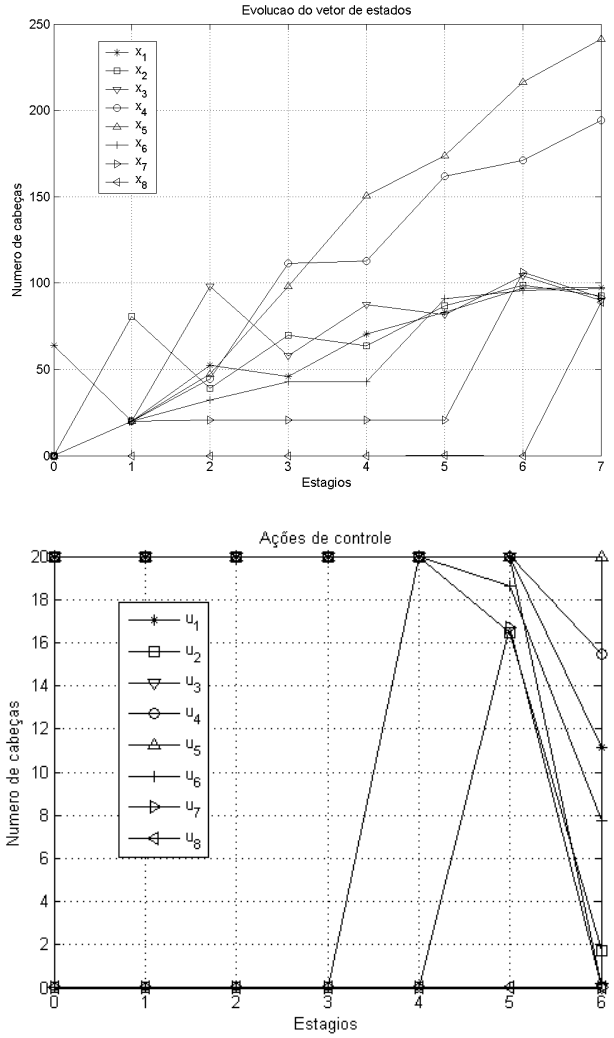

Figura 3.4: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Exemplo de uma solução Pareto-ótima do lado esquerdo da fronteira de Pareto, com custo inicial menor.

3.3.2 Projeto de redes de distribuição

O projeto ótimo de redes de distribuição consiste em encontrar o conjunto de arestas que conecta todos os vértices de um grafo, obedecendo a uma estrutura pré-definida, minimizando uma função-custo.

Neste estudo[7], consideramos as redes de energia elétrica, de relevância social e econômica vital, visto sua importância no desenvolvimento de um país. O sis-

[7]Este estudo de caso foi aceito para publicação (Carrano et al., 2008).

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	Total
$x[0]$	64	0	0	0	0	0	0	0	64
$x[1]$	20	81	20	20	20	20	20	0	201
$x[2]$	52	39	98	44	47	32	20	0	332
$x[3]$	46	70	58	111	98	43	21	0	447
$x[4]$	70	64	87	113	150	43	21	0	548
$x[5]$	83	87	82	162	174	91	21	0	700
$x[6]$	97	99	104	171	216	96	106	0	889
$x[7]$	97	93	90	194	241	97	92	89	993
x^*	98	94	91	195	242	97	93	90	1000

Tabela 3.1: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Evolução do rebanho referente a uma solução Pareto-ótima do lado esquerdo da fronteira de Pareto, com custo inicial menor.

tema de distribuição de energia elétrica é a parte mais cara do sistema; portanto, redução nas perdas leva a considerável redução nos custos, donde se justifica o uso de técnicas de otimização no projeto. O problema é que sua formulação matemática recai num problema combinatório com restrições e função-objetivo não-lineares, e desta forma, tem sido oportuno trabalhar com algoritmos evolucionários (Park et al., 1998; Carrano et al., 2005, 2006, 2007).

O projeto da rede ao longo do tempo permite levar em conta as constantes alterações a que o sistema de distribuição está sujeito, por exemplo, a necessidade de sucessivas expansões. Assim, pode ser modelado como um problema de programação dinâmica discreta no tempo (Park et al., 1998), com o sistema sendo expandido com passos incrementais.

Modelagem Matemática

Carrano (2007) propõe uma codificação que permite considerar redes com m arestas como m-uplas ordenadas de números inteiros, e mostra que esta codificação herda o conceito de vizinhança no $\mathbb{R}^m$. Como exemplo, considere o vetor $\bar{x}$ na figura 3.6, referente a uma rede com m arestas e n nodos. Suponha que cada tipo de ramo corresponda a um número inteiro pré-determinado. Assim, nesta codificação, o valor da coordenada i do vetor da rede corresponde à codificação do tipo de ramo da aresta i.

Desta forma, o sistema dinâmico que modela a expansão da rede descrita como vetores pode ser posto como em (3.15).

$$x[k+1] = x[k] + u[k]. \tag{3.15}$$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	Total
$u[0]$	20	20	20	20	20	20	20	20	160
$u[1]$	20	20	20	20	20	20	2	-20	102
$u[2]$	20	20	20	20	20	-3	-10	-20	67
$u[3]$	20	20	20	20	20	-4	-20	-20	56
$u[4]$	20	20	20	20	20	20	-20	-20	80
$u[5]$	20	20	20	20	20	20	20	-20	120
$u[6]$	4	0	-6	4	20	1	0	-14	9

Tabela 3.2: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Negociação do rebanho referente a uma solução Pareto-ótima do lado esquerdo da fronteira de Pareto, com custo inicial menor.

Neste caso, $x[k]$ é a codificação da rede no estágio k e $u[k]$ é a codificação do incremento da rede no estágio k (novas instalações e redimensionamentos)[8].

O algoritmo deve determinar as etapas da expansão, a partir de uma rede inicial até uma rede final. Supomos como dada a rede inicial $x[0] = x_0$, que pode por ventura ser a rede correspondente ao vetor nulo (o projeto começa do zero). A partir de um projeto anterior, define-se a rede x^* que será a meta de programação do problema, ou seja, queremos que ao final do processo, tenhamos $x[N] = x^*$, como mostra a equação (3.16).

$$x[N] = x_0 + \sum_{k=0}^{N-1} u[k] = x^*. \tag{3.16}$$

Como exemplo, considere a figura 3.7, referente à expansão de uma rede com 9 nodos e 4 estágios. A primeira rede mostrada é a rede inicial ($x[0] = x_0$ dado), e as demais redes são implementadas seqüencialmente, até que no estágio final $N = 4$, tenhamos a rede final igual à meta de programação $x[4] = x^*$, obtida via otimização numa etapa anterior a este projeto.

A codificação das variáveis de cada estágio ($x[k]$ e $u[k]$) associada à expansão desta rede está na figura 3.8. A correspondente seqüência dos valores das variáveis de decisão $\{u[0],u[1],u[2],u[3]\}$ está na figura (3.9) e a correspondente seqüência dos valores das variáveis de estados $\{x[0],x[1],x[2],x[3],x[4]\}$ está na figura 3.10.

[8]Como existe uma bijeção, podemos considerar que a codificação da rede (vetor) e a rede são a mesma coisa.

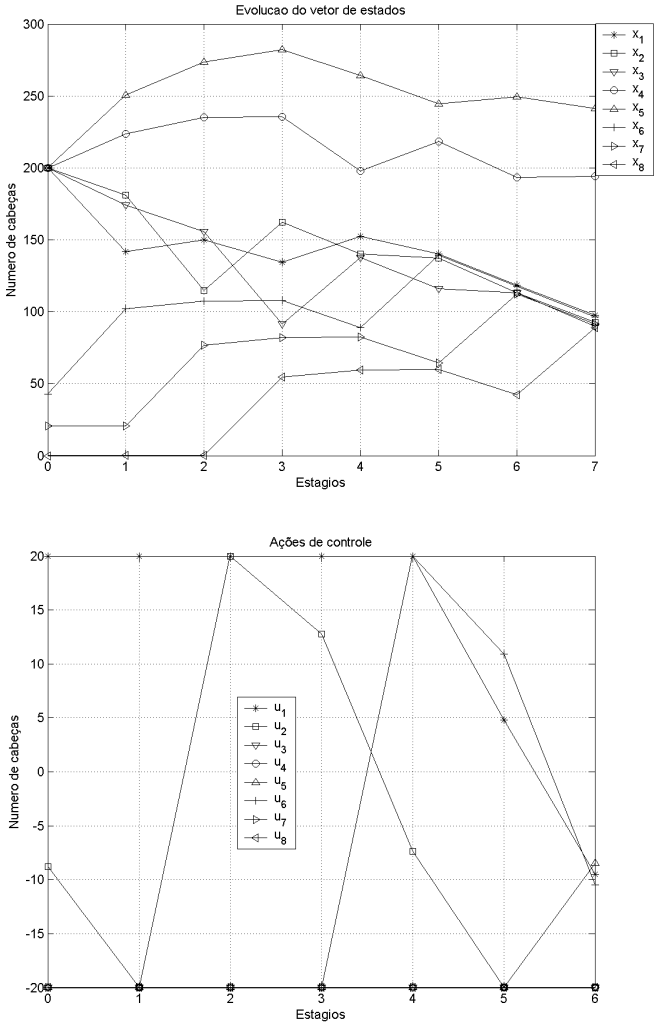

Figura 3.5: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Exemplo de solução Pareto-ótima do lado direito da fronteira de Pareto, com custo inicial maior.

$\bar{x}$	$=$	$[$	x_1	x_2	$\ldots$	x_{n-1}	x_n	x_{n+1}	$\ldots$	x_m	$]$
aresta			1	2	$\ldots$	$n-1$	n	$n+1$	$\ldots$	m	
do nodo			1	1	$\ldots$	1	2	2	$\ldots$	$n-1$	
para o nodo			2	3	$\ldots$	n	3	4	$\ldots$	n	

Figura 3.6: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Codificação da rede como um vetor. Nesta codificação, o valor da coordenada i do vetor $\bar{x}$ corresponde à codificação do tipo de ramo da aresta i da rede em questão.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	Total
$x[0]$	200	200	200	200	200	43	20	0	1063
$x[1]$	142	181	174	224	251	102	21	0	1095
$x[2]$	150	115	156	235	274	107	77	0	1114
$x[3]$	134	162	91	236	282	108	82	55	1150
$x[4]$	152	140	138	198	264	89	82	60	1123
$x[5]$	140	137	116	218	244	139	64	60	1118
$x[6]$	118	113	113	193	249	117	112	42	1057
$x[7]$	97	93	90	194	241	97	92	89	993
x^*	98	94	91	195	242	97	93	90	1000

Tabela 3.3: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Evolução do rebanho referente a uma solução Pareto-ótima do lado direito da fronteira de Pareto, com custo inicial maior.

Otimização

Voltando ao caso geral, o custo de cada estágio pode ser formulado como em (3.17).

$$g_k(x[k],u[k]) = \text{custop}(x[k]) + \text{custinst}(u[k]). \tag{3.17}$$

Em que custop$(\cdot)$ são os custos de manutenção e perda de energia e custinst$(\cdot)$ são os custos de instalação e redimensionamento do sistema.

Dado o grafo de busca G, contendo n_n nodos e n_a possíveis arestas, definimos X como o conjunto dos vetores que representam todas as árvores geradoras contidas em G ou em subgrafos de G. O grafo G considerado deve conter todos os nodos refentes aos centros consumidores ao final do horizonte de otimização, mas em estágios iniciais, podemos considerar apenas subgrafos de G: os nodos referentes a centros já existentes (com carga instalada), e as arestas que conectam apenas pares destes nodos. Desta forma, a rede de cada estágio $x[k]$ deverá ser um elemento de X. Além disso, o custo de cada estágio g_k deve ser menor que uma quantia pré-determinada cmax.

Para cada rede $x[k]$, as demais restrições são:

1. A carga demandada pelos consumidores deve ser atendida.
2. A estrutura radial da árvore deve ser mantida.
3. Os níveis de corrente $I(x[k])$ e de tensão $V(x[k])$ regulamentados devem ser cumpridos.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	Total
$u[0]$	20	-8	-20	-20	-20	-20	-20	-20	-108
$u[1]$	20	-20	-20	-20	-20	-20	-20	-20	-120
$u[2]$	20	20	-20	-20	-20	-20	-20	-20	-80
$u[3]$	20	13	-20	-20	-20	-20	-20	-20	-87
$u[4]$	20	-7	-20	-20	-20	20	-20	-20	-67
$u[5]$	5	-20	-20	-20	-20	11	-20	-20	-87
$u[6]$	-10	-20	-20	-20	-9	-11	-20	-20	-130

Tabela 3.4: Estudo de caso: evolução de um rebanho de gado (seção 3.3.1). Negociação do rebanho referente a uma solução Pareto-ótima do lado direito da fronteira de Pareto, com custo inicial maior.

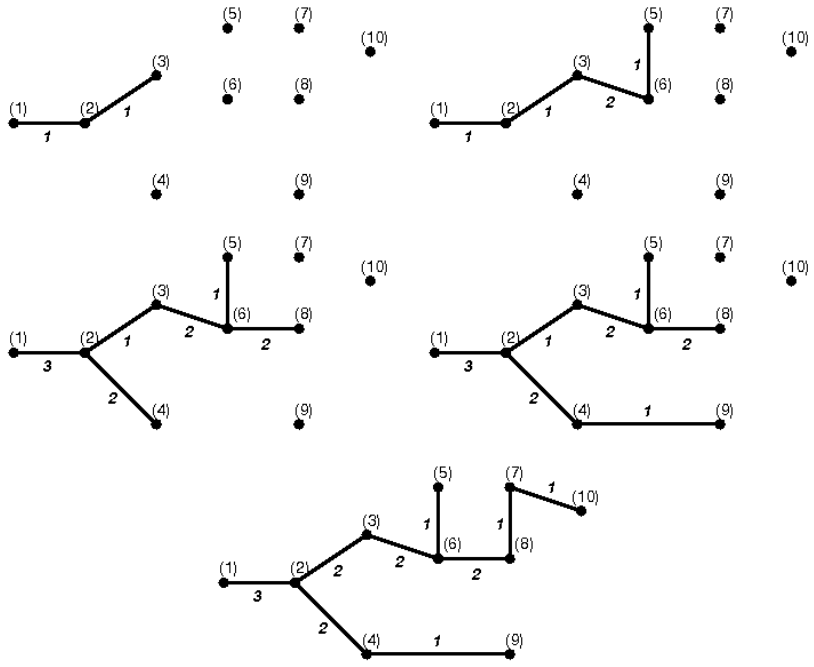

Figura 3.7: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Exemplo de expansão de uma rede com 9 nodos e 4 estágios: as redes em cada um dos estágios. A primeira rede mostrada é a rede inicial ($x[0] = x_0$ dado), e as demais redes são implementadas seqüencialmente, até que no estágio final $N = 4$, tenhamos a rede final igual à meta de programação $x[4] = x^*$, obtida via otimização numa etapa anterior a este projeto.

$x[k]$	=	[	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	]
$u[k]$	=	[	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	]
do nodo			1	2	2	3	4	5	6	7	7	
para			2	3	4	6	9	6	8	8	10	

Figura 3.8: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Codificação das variáveis de cada estágio para o exemplo da figura 3.7.

$$\begin{bmatrix} u[0] \\ u[1] \\ u[2] \\ u[3] \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Figura 3.9: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Codificação das variáveis de controle referentes às redes da figura 3.7.

$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 3 & 1 & 2 & 2 & 0 & 1 & 2 & 0 & 0 \\ 3 & 1 & 2 & 2 & 1 & 1 & 2 & 0 & 0 \\ 3 & 2 & 2 & 2 & 1 & 1 & 2 & 1 & 1 \end{bmatrix}$$

Figura 3.10: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Codificação das variáveis de estado referentes às redes da figura 3.7.

O problema consiste em determinar as etapas da expansão da rede, minimizando a soma dos custos de cada estágio, atendendo às especificações dos órgãos reguladores.

A formulação matemática deste problema pode ser posta como em (3.18).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) \tag{3.18}$$

$$\text{sujeito a: } \begin{cases} x_0 + \sum_{k=0}^{N-1} u[k] = x^* \\ x[k] \in X; \\ \min \leq V(x[k]) \leq \max; \\ I(x[k]) \leq \max; \\ g_k(x[k],u[k]) \leq \text{cmax}. \end{cases}$$

Note que a função-objetivo e algumas restrições deste problema não são lineares, o que não impede de usarmos a metodologia proposta na seção 3.2, pois o sistema dinâmico é linear. A diferença é que, neste caso, a otimização em malha

aberta é feita heuristicamente por meio de um Algoritmo Genético[9]. A fim de trabalhar apenas com pontos factíveis no algoritmo, usamos os operadores de cruzamento, mutação e correção propostos por Carrano (2007), tendo como base a codificação para redes já apresentada. Outra vantagem de usar estes operadores específicos, é que a relaxação nas variáveis (que são inteiras) não precisa ser feita, ou seja, podemos considerá-las enquanto tais.

Consideramos, agora, uma rede com estado final x^* contendo 100 nodos. Supomos que a rede inicial x_0 tenha 70 nodos. Consideramos como horizonte $N = 10$ anos, com estágios anuais (a intervenção no sistema se dá a cada ano). Esperamos que o crescimento anual da carga seja de 5% por nodo existente. A figura 3.11 apresenta a expansão: a rede inicial dada $x[0] = x_0$, a rede $x[5]$ no quinto ano, a rede $x[7]$ no sétimo ano, e a rede final igual à meta de programação $x[10] = x^*$.

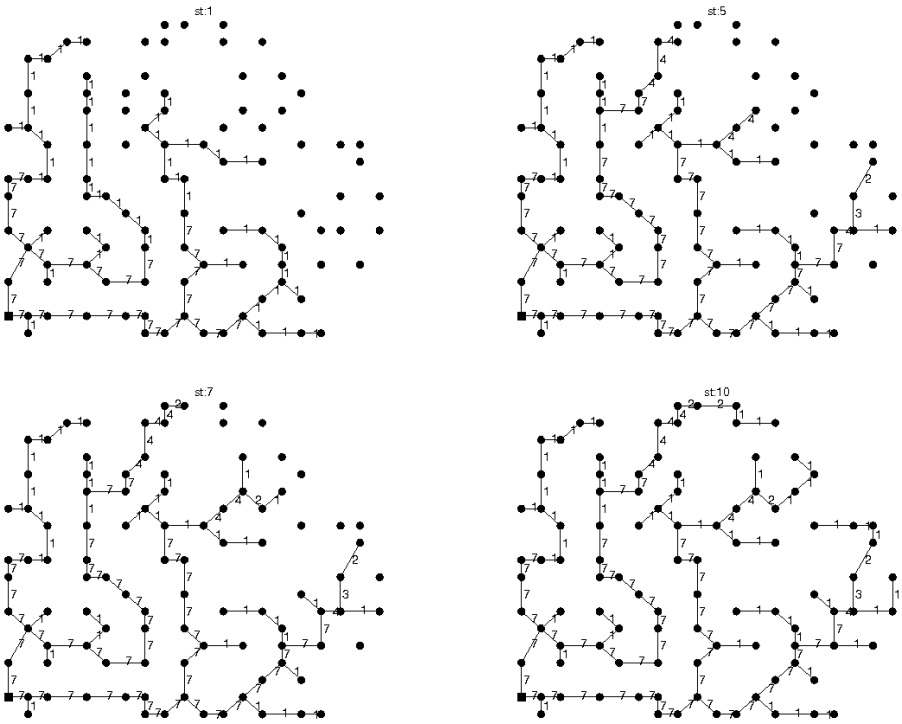

Figura 3.11: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Exemplo de expansão de uma rede com 100 nodos e 10 estágios: a rede inicial: $x[0] = x_0$, a rede no quinto ano: $x[5]$, a rede no sétimo ano: $x[7]$ e a rede final: $x[10] = x^*$.

[9]Baseados nos conceitos de evolução natural, estes métodos lidam com uma população de indivíduos que evoluem na direção das melhores regiões do espaço de busca (Goldberg, 1989; Davis, 1991; Takahashi, 2004).

Considere o efeito da discretização do tempo no processo. Simulamos o problema para 9 níveis de discretização: 1 (que equivale a uma abordagem estática), 2, 10, 20, 30, 40, 60 e 120 estágios (que equivale a uma intervenção por mês). Consideramos como função-objetivo o custo total do processo de implantação somado ao longo dos estágios, trazidos para um mesmo valor presente. A figura 3.12 apresenta o gráfico. Note que ocorre uma redução considerável do custo quando se passa da abordagem estática para uma abordagem dinâmica, até que com 10 estágios (tempo de intervenção igual a 1 ano) o custo estabiliza. Como o valor da função-objetivo é similar para diferentes discretizações, podemos intuir certa robustez da resposta encontrada.

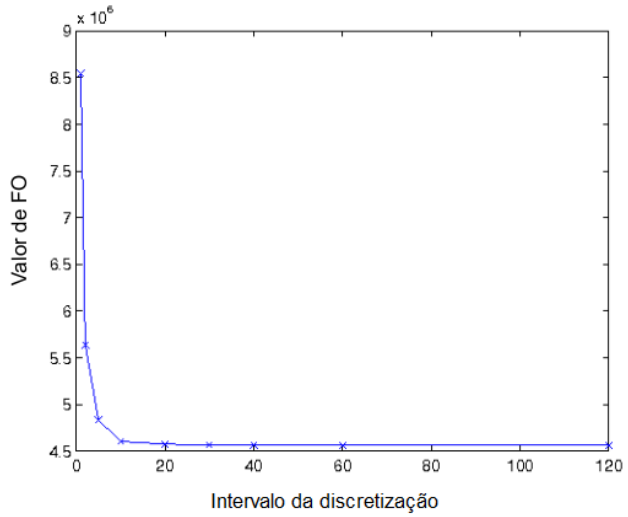

Figura 3.12: Estudo de caso: projeto de redes de distribuição (seção 3.3.2). Efeito da discretização do tempo no valor da função-objetivo. A função-objetivo é o custo total do processo de implantação somado ao longo dos estágios, trazidos para um mesmo valor presente.

3.4 Conclusões do capítulo

Neste capítulo, reproduzimos as principais idéias discutidas na dissertação de mestrado (Cardoso, 2005), que apresenta ferramentas para problemas de programação dinâmica linear. Esta proposta consiste em estudar a dinâmica do sistema pela iteração de conjuntos fechados em torno de uma meta de programação, a fim de otimizar em malha aberta com as variáveis de estado em apenas um dos estágios. Além disso, nesta tese, acrescentamos a abordagem multiobjetivo, via

técnicas de escalarização. Reforçamos que em problemas dinâmicos multiobjetivos, a "maldição da dimensionalidade" costuma ser ainda mais séria.

Consideramos dois relevantes estudos de caso. No primeiro, estudamos a otimização dos recursos envolvidos na implantação de uma fazenda com 1000 cabeças de gado ao longo de 7 anos, quando o rebanho adquire a condição de estabilidade. No segundo, consideramos a otimização da expansão de uma rede de distribuição de energia elétrica com 100 nodos em 10 anos, cuja abordagem proposta é bastante eficiente e não-convencional. No mínimo, podemos dizer que é notável o fato de termos obtido bons resultados num problema complicado, considerando um sistema dinâmico tão simples.

Estes dois exemplos puderam validar a metodologia proposta. Além disso, assinalamos que um algoritmo de programação dinâmica clássico teria muita dificuldade para resolver estes dois problemas num tempo razoável (resolvemos o primeiro problema na ordem de alguns minutos, e o segundo na ordem de algumas horas). Acreditamos que obteríamos bons resultados na aplicação desta metodologia em problemas com dinâmica similar, principalmente se propusermos operadores específicos que tornem o problema computacionalmente tratável, por exemplo, via algoritmos genéticos.

Como estudos futuros, propomos investigar esta abordagem em sistemas lineares com funções-objetivo convexas não-quadráticas via metodologia LMI (Boyd et al., 1994), que tem se mostrado muito eficiente nesta classe de problemas. Propomos também considerar a forma estocástica dos estudos de casos deste capítulo, por exemplo, considerando as incertezas nas demandas de carga no projeto de expansão de redes de distribuição.

Capítulo 4

Ferramentas para Programação Dinâmica Não-Linear

Neste capítulo, estendemos para a programação dinâmica não-linear a metodologia apresentada para a programação dinâmica linear discreta no tempo. Propomos otimizar em malha aberta, escrevendo o estado em cada estágio do problema como função apenas do estado inicial e da seqüência de ações de controle anteriores, por meio da pré-imagem de um conjunto fechado em torno da meta de programação do problema. Consideramos problemas com variáveis reais (ou que permitam a relaxação) e ações de controle discretas no tempo.

Na seção 4.1, apresentamos as principais formulações para problemas de programação dinâmica não-linear. Consideramos o caso do sistema com tempo discreto na subseção 4.2.1 e o sistema impulsivo na subseção 4.2.2, permitindo que a evolução do sistema dinâmico entre os estágios se dê por meio da equação diferencial, mas com ações de controle impulsivas encontradas via métodos propostos de programação dinâmica para tempos discretos. Na subseção 4.2.3, apresentamos a extensão para o caso multiobjetivo. Por fim, consideramos dois estudos de casos: na subseção 4.3.1 estudamos o controle biológico de pragas numa lavoura e na subseção 4.3.2 estudamos a busca de estratégias ótimas de vacinação numa situação de epidemia. Na seção 4.4, apresentamos as conclusões desse capítulo.

4.1 Introdução

Uma maneira de se trabalhar com problemas dinâmicos não-lineares é através da sua **linearização**[1]. Para se resolver um problema de programação dinâmica

[1]Um sistema linearizado é obtido a partir da aproximação de Taylor de primeira ordem da função não-linear em torno de um dado ponto fixo (ou ponto de equilíbrio) do sistema (Fiedler-Ferrara e do Prado, 1994).

não-linear via linearização, poderíamos usar os procedimentos propostos para os problemas lineares, usando o sistema dinâmico linearizado em torno do ponto fixo escolhido. Mas, a fim de obtermos resultados mais precisos, preferimos neste trabalho considerar os sistemas não-lineares enquanto tais.

O problema fundamental de programação dinâmica com tempo discreto pode ser formulado como em (1.5), reescrito aqui.

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N])$$

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k],u[k]); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

A equação de estado deste problema é a (1.6), também reescrita aqui.

$$x[k+1] = f(x[k],u[k]).$$

O problema com tempo contínuo pode ser formulado como em (1.4), e está reescrito aqui.

$$\min_{u} \int_0^T g(t,x(t),u(t))dt + g(T,x(T))$$

$$\text{sujeito a: } \begin{cases} x'(t) = f(t,x(t),u(t)); \\ t \in [0,T]; \\ x(0) = x_0 \text{ e/ou } x(T) = x^* \text{ dados.} \end{cases}$$

A equação de movimento deste problema é a (1.2), também reescrita aqui para comodidade do leitor.

$$x'(t) = f(t,x(t),u(t)).$$

Nesta tese, procuramos soluções numéricas para os problemas, e estas geralmente estão relacionadas a ações de controle em tempos discretos ou discretizados. Evitando a discretização do sistema, propomos ações de controle impulsivas, permitindo que a dinâmica autônoma seja regida pela equação diferencial (contínua no tempo). Chamamos este problema de programação dinâmica impulsiva, que pode ser formulado como em (2.35), reescrito aqui para comodidade do leitor.

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N])$$

$$\text{sujeito a: } \begin{cases} x'(t) = f_1(t,x(t)); \\ t \in [\tau_k^+,\tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados.} \end{cases}$$

A equação de estado deste problema é a (2.32), também reescrita aqui.

$$\begin{cases} x'(t) = f_1(t,x(t)); \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k). \end{cases}$$

Para que o problema seja considerado como de programação dinâmica não-linear, basta que o sistema dinâmico seja não-linear.

4.2 Formulações propostas

Pretendemos resolver o problema de otimização em malha aberta, escrevendo o estado em cada estágio como função apenas do estado inicial e da seqüência de ações de controle anteriores, estendendo a abordagem geométrica apresentada para o caso linear. Desta forma, a região factível do problema será a pré-imagem pelo sistema dinâmico de um conjunto fechado em torno da meta de programação.

4.2.1 Caso discreto

Vamos, primeiro, considerar o problema autônomo, com função f não-linear. A equação de estado deste problema é a (4.1).

$$x[k+1] = f(x[k]). \tag{4.1}$$

Podemos escrever qualquer estado $x[k]$ em função do estado inicial $x[0]$ recursivamente, pela composição de funções, como na equação (4.2).

$$x[k] = f^k(x[0]) = \underbrace{f \circ \cdots \circ f}_{k \text{ vezes}}(x[0]). \tag{4.2}$$

No caso não-autônomo, fazer a iteração das variáveis de estado para um mesmo estágio também implica em reproduzir formalmente nas variáveis em cada estágio a restrição do sistema dinâmico. Assim, para cada estado inicial $x[0]$ dado

e para uma dada seqüência de ações de controle $u[0], \ldots, u[N-1]$, de acordo com a equação (1.6), temos que:

$$\begin{aligned} x[1] &= f(x[0],u[0]) \Rightarrow \\ x[2] &= f(x[1],u[1]) = f(f(x[0],u[0]),u[1]) \Rightarrow \\ &\vdots \\ x[N] &= f(x[N-1],u[N-1]) = f(\ldots f(f(x[0],u[0]),u[1]),\ldots,u[N-1]). \end{aligned}$$

Ou seja, podemos escrever o estado de cada estágio $x[k]$ recursivamente em função do estado inicial e da seqüência de ações de controle anteriores pela composição de funções, como na equação (4.3).

$$x[k] = f(\ldots f(f(x[0],u[0]),u[1]),\ldots,u[k-1]). \tag{4.3}$$

Embora de escrita um pouco complicada, este procedimento é de simples implementação computacional, através das funções recursivas.

Dada uma meta de programação x^*, o problema de programação dinâmica discreta não-linear pode ser formulado como em (4.4).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{4.4}$$

$$\text{sujeito a: } \begin{cases} f(\ldots f(f(x[0],u[0]),u[1]),\ldots,u[N-1]) = x^*; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Computacionalmente teremos que substituir também cada $x[k]$ presente na função-objetivo de acordo com a equação (4.3).

Esta formulação encontra o ótimo do problema com variáveis reais, mas traz consigo as possíveis dificuldades de um problema com restrição de igualdade: a região factível pode ficar muito pequena, podendo atrasar a convergência do método. Este possível inconveniente, já comentado para o caso linear, costuma se agravar em problemas não-lineares (Wanner, 2006). Assim como no caso linear, podemos melhorar o resultado da otimização (o tempo de execução e o valor da função-objetivo) se considerarmos como região factível do problema a pré-imagem de uma vizinhança em torno da meta de programação, de tamanho admissível.

A metodologia aqui proposta é a seguinte:

Algoritmo Proposto - Programação Dinâmica Não-Linear

1. Permitir a relaxação na meta de programação, que será substituída por um conjunto ao seu redor.

2. Fazer o retorno deste conjunto, estágio por estágio, até o estágio inicial via equação de estado do sistema dinâmico (a região factível será o resultado desta iteração).

3. Por fim, basta fazer a otimização restrita a esta região factível para encontrar a solução do problema.

Consideramos o problema de programação dinâmica restrito à pré-imagem de uma bola de raio ϵ em torno da meta de programação x^* no estágio final N. Uma bola de raio ϵ em torno de uma meta x^* no estágio N pode ser escrita como o conjunto dos pontos $x[N]$ tais que:

$$||x[N] - x^*||_2 \leq \epsilon \quad \text{ou} \quad (x[N] - x^*)^t(x[N] - x^*) \leq \epsilon^2. \tag{4.5}$$

Podemos escrever o estado final $x[N]$ desta equação (4.5) como função apenas do estado inicial $x[0]$ e da seqüência de ações de controle anteriores $u[0], \ldots, u[N-1]$, usando a equação (4.3).

Este problema pode ser formulado como em (4.6).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{4.6}$$

$$\text{sujeito a: } \begin{cases} ||(f(\ldots f(f(x[0],u[0]), u[1]), \ldots, u[N-1]) - x^*)||_2 \leq \epsilon; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Computacionalmente, também teremos que substituir cada estado $x[k]$ presente na função-objetivo como na equação (4.3).

Genericamente, não temos *a priori* informações sobre a pré-imagem de um conjunto em torno da meta de programação no estágio inicial, uma vez que o sistema é não-linear. Alguns dados sobre a geometria ou a topologia do problema (características da função-objetivo, da função da equação de estado do sistema e/ou das demais restrições), tais como: diferenciabilidade, unimodalidade ou convexidade, podem ajudar na escolha do melhor método de otimização a ser usado para se resolver o problema em malha aberta[2].

Principalmente quando não dispomos destas informações, sugerimos métodos estocásticos de otimização, como os algoritmos genéticos (AGs). Baseados nos

[2]O livro de Takahashi (2004) apresenta um estudo detalhado e formal sobre os métodos de otimização para problemas não-lineares e comenta, com exemplos práticos, em que situação cada método deve ser usado. Outras referências sobre otimização não-linear podem ser: Luenberger (1984); Miettinen (1998); Bazaraa et al. (1993).

conceitos de evolução natural, estes métodos lidam com uma população de indivíduos que evoluem na direção das melhores regiões do espaço de busca (Goldberg, 1989; Davis, 1991; Takahashi, 2004). Estes métodos são meta-heurísticas: sem a necessidade de nenhum conhecimento prévio sobre a função a ser otimizada, trabalham sobre uma população de soluções candidatas, procurando soluções sub-ótimas a um custo computacional razoável, sem nenhuma garantia de otimalidade.

Assinalamos que a discussão sobre o melhor método a ser usado na resolução de cada problema vale para todas as proposições ao longo deste capítulo. Entretanto, repare que as formulações aqui propostas são gerais, no sentido de que independem das características do sistema dinâmico, da função-objetivo do problema, bem como das características das demais restrições.

4.2.2 Caso impulsivo

Propomos estudar o problema de programação dinâmica impulsiva como discreto, considerando como relevante apenas o valor dos estados nos instantes em que ocorrem as ações impulsivas. Poderíamos resolver este problema via métodos enumerativos descritos na seção 2.2, mas preferimos otimizar em malha aberta, trabalhando com as variáveis de estado apenas em um dos estágios do problema, evitando a "maldição da dimensionalidade". Assim, a diferença em relação aos métodos formulados na subseção 4.2.1 será quanto à maneira de se calcular o valor de cada estado $x[k]$.

Supomos que a restrição de ponto final $x[N] = x(T) = x^*$ seja dada. O problema de programação dinâmica impulsiva pode ser descrito como em (4.7).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{4.7}$$

$$\text{sujeito a: } \begin{cases} x[N] = x^*; \\ x[0] = x_0 \text{ pode ser dado.} \end{cases}$$

em que, a partir de um estado inicial (dado ou candidato a ótimo), o valor de cada estado $x[k]$ deve ser calculado resolvendo-se numericamente os k problemas de valor inicial (PVIs) mostrados na equação (2.33), e reescrito a seguir.

$$\text{PVI do estágio } k\text{: } \begin{cases} x'(t) = f_1(t,x(t)); \\ t \in [\tau_k^+,\tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k] + f_2(u[k]); \\ x[k] = x(\tau_k). \end{cases}$$

Cada PVI pode ser facilmente resolvido por meio de métodos numéricos, por exemplo, pelo método de Runge-Kutta de quarta ordem (Campos-filho, 2001).

Apresentamos também a formulação considerando um conjunto em torno da meta de programação, de tamanho admissível. O problema restrito à pré-imagem de uma bola de raio ϵ (dado) em torno da meta de programação pode ser formulado como em (4.8).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \tag{4.8}$$

$$\text{sujeito a: } \begin{cases} (x[N]-x^*)^t(x[N]-x^*) \leq \epsilon^2; \\ x[0] \text{ pode ser dado.} \end{cases}$$

em que cada $x[k]$ deve ser calculado resolvendo-se numericamente os k PVIs mostrados na equação (2.33).

Note que não temos, *a priori*, informações topológicas ou geométricas sobre a pré-imagem de uma bola em torno da meta de programação pelo sistema dinâmico impulsivo não-linear. Entretanto, de acordo com a simulação a ser mostrada na subseção 4.3.2, a pré-imagem da bola pelo sistema autônomo segue a órbita do sistema, e neste sentido, a ação de controle apenas translada a condição inicial de cada PVI.

4.2.3 Abordagem multiobjetivo

Na seção 2.6 dissemos que um problema de programação dinâmica multiobjetivo pode ser visto como a otimização de uma função-objetivo vetorial, restrita a um sistema dinâmico. Nesta seção, estendemos as formulações propostas nas seções 4.2.1 e 4.2.2 (referentes aos casos discreto e impulsivo, respectivamente) para o caso multiobjetivo.

A equação (4.9) apresenta a formulação multiobjetivo para a otimização restrita à pré-imagem da meta de programação.

$$\min_{u[0],\ldots,u[N-1]} J = (J_1,\ldots,J_m) \tag{4.9}$$

$$\text{sujeito a: } \begin{cases} f(\ldots f(f(x[0],u[0]),u[1]),\ldots,u[N-1]) = x^*; \\ x[0] \text{ pode ser dado.} \end{cases}$$

A equação (4.10) apresenta a formulação multiobjetivo para a otimização restrita à pré-imagem de uma bola de raio ϵ (dado) em torno da meta de programação.

$$\min_{u[0],\ldots,u[N-1]} J = (J_1,\ldots,J_m) \tag{4.10}$$

$$\text{sujeito a: } \begin{cases} ||(f(\ldots f(f(x[0],u[0]),u[1]),\ldots,u[N-1]) - x^*)||_2 \leq \epsilon; \\ x[0] \text{ pode ser dado.} \end{cases}$$

Cada J_i é um funcional (linear ou não-linear), que pode ser escrito da forma geral mostrada na equação (4.11).

$$J_i = \sum_{k=0}^{N-1} g_{ik}(x[k],u[k]) + g_{iN}(x[N]). \tag{4.11}$$

Computacionalmente falando, no caso de tempo discreto, o valor de cada estado $x[k]$ deverá ser considerado como explicado na seção 4.2.1 e no caso impulsivo, o valor de cada estado $x[k]$ deverá ser calculado como explicado na seção 4.2.2.

Como a otimização é feita em malha aberta, a resolução de cada problema pode ser feita via métodos de otimização multiobjetivo. Como os problemas são não-lineares, destacamos os algoritmos estocásticos, como o NSGA-II[3], usado no estudo de caso da seção 4.3.1.

4.3 Estudos de caso

4.3.1 Controle biológico de pragas

O controle biológico de pragas é uma prática antiga, mas que atualmente tem merecido destaque na sociedade: temos observado uma maior consciência da necessidade de conservação e do uso inteligente dos recursos naturais disponíveis.

Para realizar o controle biológico, parasitóides são criados em laboratórios até que, atingindo um número adequado, são liberados no meio-ambiente na expectativa de que seja criado um sistema presa-predador com as pragas das lavouras. Dessa forma, o processo permite a substituição do uso de agrotóxicos, resultando em produtos agrícolas de melhor qualidade nutricional, com melhor aceitação no mercado. Em uma lavoura, o número de insetos ou animais que atacam certo tipo de plantação é considerado praga ou peste a partir do momento em que a quantidade de espécie por área seja capaz de causar danos ou prejuízos econômicos. No controle biológico, agricultores liberam inimigos naturais que atacam as pestes, porém é importante saber a quantidade ideal desses inimigos para evitar um desequilíbrio ecológico e outras perdas.

[3]O NSGA-II é baseado na classificação de indivíduos por camadas não-dominadas no espaço de objetivos, atribuindo um valor de aptidão para cada camada. Além disso, uma função de partilha degrada os indivíduos numa mesma camada que se encontram muito próximos entre si (Deb et al., 2002).

O presente estudo de caso[4] tem como objetivo otimizar o controle biológico de pragas numa lavoura através da inserção de predadores em intervalos de tempo discretos, a um custo mínimo. Consideraremos o controle biológico da lagarta da soja (*Anticarsia gematalis*), usando seus inimigos naturais, tais como aranhas e vespas.

Nossa inspiração foi no trabalho de Feltrin e Rafikov (2002), que propõe um controle ótimo com ação contínua no tempo, de modo que a densidade das pragas (e também dos predadores) fique abaixo de um nível considerado tolerável. Os resultados obtidos por eles, apesar de serem ótimos do ponto de vista matemático, se mostram inviáveis sob o ponto de vista prático, devido à necessidade de inserções de predadores em um tempo contínuo. Pontuamos que foi deste artigo que retiramos as principais informações para este estudo de caso: o modelo dinâmico e os valores dos parâmetros.

Modelo Matemático

A interação entre as pragas numa lavoura e seus inimigos naturais pode ser representada por um sistema dinâmico, dado pelas equações diferenciais ordinárias (EDOs) de um sistema presa-predador como em (4.12).

$$\begin{cases} x'(t) = xf_1(x,y); \\ y'(t) = yf_2(x,y). \end{cases} \tag{4.12}$$

As variáveis $x = x(t)$ e $y = y(t)$ representam, respectivamente, as densidades das presas e predadores por metro quadrado em cada instante de tempo t. Supomos que neste sistema não há interferência significativa de agentes externos.

Neste trabalho, consideramos como sistema presa-predador o modelo (4.13).

$$\begin{cases} x'(t) = x(a - \gamma x - \alpha y); \\ y'(t) = y(-b + \beta x). \end{cases} \tag{4.13}$$

Os parâmetros a, γ, α, b e β são constantes positivas com valores conhecidos. Os produtos ax e $-by$ modelam a taxa de crescimento das presas e a taxa de mortalidade dos predadores; os termos $-\alpha xy$ e βxy representam as interações entre as duas populações em ambas as equações, em que cada encontro tende a ser favorável ao predador, pois possibilita o crescimento dessa espécie e desfavorável à presa por reduzir a sua população; o produto $-x\gamma x$ determina a competição intra-específica. A condição $\gamma > 0$ é suficiente para a estabilidade global do sistema, mas este equilíbrio pode ser inaceitável do ponto de vista prático.

[4]Este trabalho foi publicado (por partes) em anais de dois congressos nacionais e um internacional (da Cruz et al., 2006, 2007a,b) e aceito para publicação em um periódico (Cardoso et al., 2008b).

Vamos considerar a relação entre a lagarta da soja (*Anticarsia gematalis*) e seus predadores naturais (como aranhas e vespas). Os valores dos parâmetros do sistema correspondentes a esta relação estão reproduzidos na tabela 4.1.

a	b	α	γ	β
0,16	0,19	0,02	0,001	0,0029

Tabela 4.1: Estudo de caso: controle biológico de pragas (seção 4.3.1). Tabela de constantes do sistema presa-predador utilizadas neste estudo de caso.

Consideramos os quatro tipos de condição inicial para o problema, variando as coordenadas do ponto inicial como nulas ou não-nulas. Assim, mapeamos, via simulação, todos os pontos fixos ($\bar{x}$,$\bar{y}$) deste sistema dinâmico autônomo (4.13). Estes dados, para cada tipo de condição inicial, estão apresentados na tabela 4.2. A primeira condição inicial representa a maior parte dos casos, na qual o sistema tende para um ponto estável, a saber, um foco assintoticamente estável. O segundo descreve uma situação sem predadores, no qual o equilíbrio é determinado pela competição intra-específica. A terceira condição descreve uma condição inicial nula e a última apresenta a situação com predadores e sem presas, no qual a população de predadores se extingue no final. A figura 4.3 apresenta um exemplo da evolução desse sistema autônomo e seu diagrama de fases para a condição inicial $x_0 = 50$ e $y_0 = 9{,}2$. Note que o sistema converge para o ponto fixo apresentado na tabela 4.2.

Segundo dados da EMBRAPA, o nível tolerável da densidade por metro quadrado das presas — as lagartas da soja — é de $x^* = 20$, um valor bem abaixo da condição de equilíbrio, que é $\bar{x} = 65{,}5172$. Assim, a fim de evitar danos na lavoura, uma ação de controle neste sistema deve ser implementada.

Otimização

Queremos transladar o número de presas (lagartas da soja) do ponto fixo $\bar{x} = 65{,}5172$ para abaixo do limite tolerado pela EMBRAPA, que é $x^* = 20$, por

condição	x_0	y_0	$\bar{x}$	$\bar{y}$
1	não-nulo	não-nulo	65,5172	4,7241
2	não-nulo	nulo	160	0
3	nulo	nulo	0	0
4	nulo	não-nulo	0	0

Tabela 4.2: Estudo de caso: controle biológico de pragas (seção 4.3.1). Pontos fixos para os quatro tipos de condição inicial (x_0,y_0) do sistema cujos parâmetros constam na tabela 4.1.

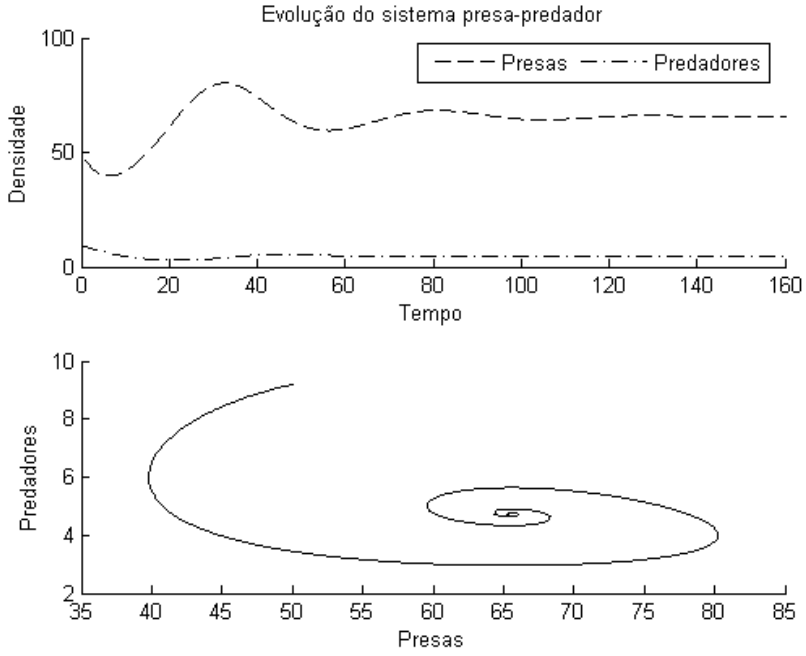

Figura 4.1: Estudo de caso: controle biológico de pragas (seção 4.3.1). Exemplo do comportamento do sistema autônomo para a condição inicial $x_0 = 50$ e $y_0 = 9{,}2$ com os parâmetros da tabela 4.1. Note que o sistema converge para o ponto fixo apresentado na tabela 4.2.

meio da inserção na lavoura dos seus predadores naturais ao longo do tempo. Consideremos o número inicial de presas por área como $x_0 = 100$, o número inicial de predadores por área como $y_0 = 0$ e o tempo total de $T = 200$.

Feltrin e Rafikov (2002) propõem uma ação de controle contínua no tempo (formulação 1.4) para este problema, considerando como modelo o descrito na formulação (4.14).

$$\begin{cases} x'(t) = x(a - \gamma x - \alpha y); \\ y'(t) = y(-b + \beta x) + U. \end{cases} \tag{4.14}$$

Esta função de controle U é responsável por transladar o sistema até ao ponto fixo desejado e garantir a estabilidade desse novo ponto.

A função objetivo a ser minimizada é o funcional quadrático (4.15), sendo, m_1 e m_2 constantes positivas. Este funcional pondera o custo de inserção de predadores com uma penalidade por não atingir a meta ao final do processo.

$$J = \int_0^{\infty} (m_1(x - x^*)^2 + m_2(y - y^*) + u^2)dt. \tag{4.15}$$

A função x^* deve ser constante igual à meta de programação dada e a função x^* é encontrada resolvendo o sistema mostrado na equação 4.14 com $x(t) = x^*$ e $x'(t) = y'(t) = 0$.

Este problema de otimização dinâmica foi resolvido via programação dinâmica contínua no tempo, através da resolução da respectiva equação HJB (equação 2.25). A função de controle ótima U^* (contínua no tempo) encontrada por esta formulação com $m_1 = 2{,}0 \times 10^{-5}$ e $m_2 = 3{,}02 \times 10^{-3}$ está na equação (4.16).

$$U^*(t) = 0{,}01886y^*(t) - \frac{0{,}15391}{2}(y^*(t) - 7). \tag{4.16}$$

A figura 4.2 apresenta o comportamento do sistema (x^*,y^*) com a ação de controle contínua no tempo proposta na equação (4.16), para a condição inicial $x_0 = 100$ e $y_0 = 0$, bem como a ação de controle ótima U^*.

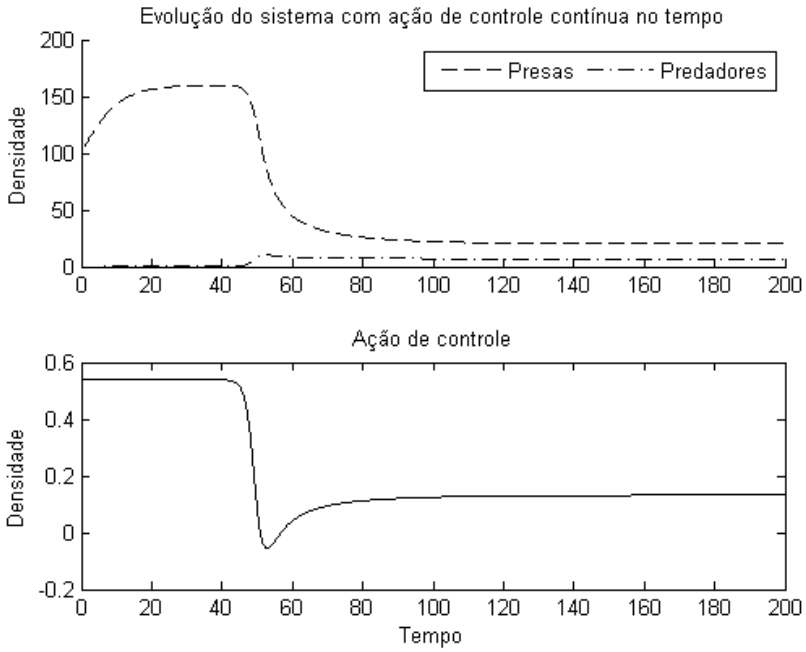

Figura 4.2: Estudo de caso: controle biológico de pragas (seção 4.3.1). Evolução do sistema presa-predador com ação de controle contínua no tempo proposta na equação (4.16), para a condição inicial $x_0 = 100$ e $y_0 = 0$. A primeira figura apresenta o comportamento ótimo do sistema (x^*,y^*) e a segunda, apresenta a ação de controle ótima U^*, todas em função do tempo.

Achamos de pouca aplicabilidade prática esta opção de controle ótimo contínuo no tempo. Neste sentido, usamos a técnica da programação dinâmica impulsiva

(formulação 2.35): propomos uma ação de controle discreta no tempo, apesar de permitir que as presas e predadores evoluam por meio da equação diferencial.

Para isto, dividimos o horizonte de tempo T em N intervalos, escolhendo uma seqüência de instantes de tempo $\{\tau_0, \ldots, \tau_N\}$, em que: $\tau_k < \tau_{k+1}$, $\tau_0 = 0$ e $\tau_N = T$. Supomos que esses intervalos sejam eqüidistantes, de tamanho δT. Faremos a otimização como um problema em tempo discreto com N estágios. Neste estudo, supomos que a duração de cada estágio é $\delta T = 20$, portanto, temos $N = 10$ estágios.

Primeiramente, consideramos uma função-objetivo linear: queremos minimizar a soma do número de presas com a soma do números de predadores introduzidos em cada estágio do problema. A nossa meta de programação será a estipulada pela EMBRAPA, assim a restrição de ponto final será $x^* \leq 20$. Considere $u[k]$ como a densidade de predadores por metro quadrado a serem lançados no estágio k.

A formulação deste problema pode ser como em (4.17).

$$\min_{u[0],\ldots,u[N-1]} \sum_{k=0}^{N-1} (x[k] + u[k]) + x[N] \tag{4.17}$$

$$\text{sujeito a: } \begin{cases} x'(t) = x(a - \gamma x - \alpha y); \\ y'(t) = y(-b + \beta x); \\ t \in [\tau_k^+, \tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k]; \\ x[k] = x(\tau_k); \\ y(\tau_k^+) = y[k^+] = y[k] + u[k]; \\ y[k] = y(\tau_k); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e } y[0] = y_0 \text{ são dados}; \\ x[N] \leq 20. \end{cases}$$

Repare que a ação de controle, em cada estágio, incide apenas sobre o número de predadores. Dados x_0 e y_0 e uma seqüência de ações de controle $\{u[0], \ldots, u[k]\}$, os estados $x[k+1]$ e $y[k+1]$ podem ser encontrados através da resolução do PVI mostrado em (4.18).

$$\text{PVI do estágio } k\text{: } \begin{cases} x'(t) = x(a - \gamma x - \alpha y); \\ y'(t) = y(-b + \beta x); \\ t \in [\tau_k^+, \tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k]; \\ x[k] = x(\tau_k); \\ y(\tau_k^+) = y[k^+] = y[k] + u[k]; \\ y[k] = y(\tau_k). \end{cases} \tag{4.18}$$

Como não temos conhecimento sobre a iteração da meta de programação ao longo deste sistema, propomos um algoritmo genético (AG) para a otimização em malha aberta. Cada indivíduo ou cada solução-tentativa é um vetor com n coordenadas em que o valor da k-ésima posição representa a densidade de predadores a serem liberados no estágio k (o valor de $u[k]$). Num problema de otimização com restrições, pode ser difícil para os AG's conseguirem manter a população factível em todas as gerações. Para contornar esta situação, usamos penalidades na função-objetivo e propomos, na seleção, dar maior prioridade aos indivíduos factíveis. A *fitness* deste AG tem adequabilidade quadrática e a seleção é feita através do resto estocástico (Silva et al., 2006). A população é de 50 soluções-tentativas, com taxa de cruzamento de 0,7 e de mutação de 0,1. O critério de parada é pelo número máximo de 100 gerações.

A figura 4.3 apresenta o gráfico mostrando o comportamento do sistema presa-predador impulsivo ótimo. A fim de melhor entendermos o comportamento dinâmico do sistema impulsivo ótimo, a figura 4.4 mostra seu diagrama de fases (também em escala logarítmica). O valor do custo para esta solução encontrada é $J^* = 446{,}2467$.

Note que o resultado ótimo obtido corresponde a um nível de presas abaixo da meta $x^* = 20$ desde o primeiro estágio, portanto, o prejuízo com as pragas na lavoura é muito pequeno. Repare também que, como especificado, o número de presas: $x[k] = x[k^+]$, para cada estágio k, ou seja, o salto é apenas no número de predadores.

Observe que a otimização via programação dinâmica impulsiva, como sugerimos aqui, é mais fácil de ser implementada nas lavouras, do ponto de vista prático, do que a estratégia usando uma ação de controle contínua no tempo (Feltrin e Rafikov, 2002). Uma comparação direta entre a estratégia impulsiva proposta e a estratégia usando ação de controle em tempo contínua não pode ser feita, uma vez que na estratégia contínua a libertação dos predadores presume-se ser realizada continuamente ao longo do tempo. No entanto, as ações de controle contínuas podem ser discretizadas da seguinte forma: definimos alguns instantes de tempo para a realização de ações de controle (como fazemos na programação dinâmica impulsiva), e aplicamos a integral da ação de contínua, impulsivamente, nestes instantes. A figura 4.5 apresenta a evolução do sistema discretizado considerando o mesmo intervalo de discretização do problema impulsivo, e as respectivas ações de controle discretizadas. Neste caso, o valor do custo função valor que é atingido se torna $J^* = 1177{,}60$, bem maior do que aquela que é conseguido com a metodologia aqui proposta, que é $J^* = 446{,}2467$.

O problema de controle químico de pragas numa lavoura também poderia ser resolvido pela metodologia proposta. Este problema consiste em descobrir a densidade ótima de inseticidas e agrotóxicos a serem lançados na lavoura de modo a minimizar o custo com os produtos químicos ou o prejuízo na lavoura, ou uma

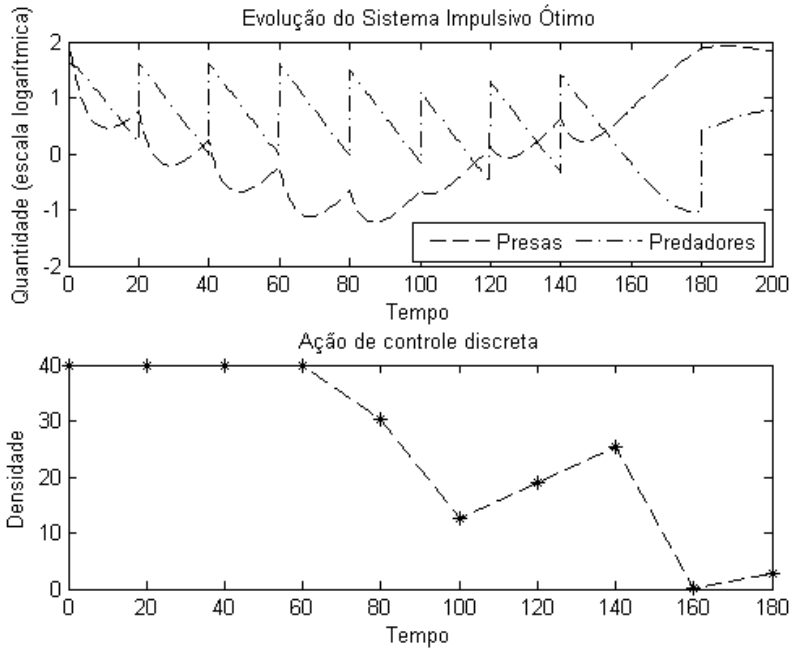

Figura 4.3: Estudo de caso: controle biológico de pragas (seção 4.3.1). A primeira figura apresenta a evolução ótima de presas e predadores ao longo dos estágios (em escala logarítmica com base 10). A condição inicial não é mostrada no gráfico, por causa da escala. A segunda figura apresenta o gráfico com a quantidade de predadores a serem lançados em cada estágio. Reforçamos que os predadores são lançados impulsivamente apenas nos inícios dos estágios, a linha tracejada serve apenas para melhor compreensão visual do resultado.

média entre esses dois custos. Também o controle híbrido (integrando o biológico com o químico), que costuma ser mais barato e eficiente do que aplicar apenas um deles isoladamente (Liu et al., 2006), poderia ser resolvido via programação dinâmica impulsiva. Para isto, bastaria reescrever a equação diferencial impulsiva (4.18) de acordo com o modelo em questão, e resolver o problema de otimização através do mesmo AG comentado anteriormente.

Abordagem multiobjetivo

Em vez de considerarmos como função-objetivo do problema a média entre o custo com a inserção de predadores e prejuízo com as pragas, como fizemos anteriormente, propomos uma formulação biobjetivo (como apresentado na seção 4.2.3), para encontrar soluções de compromisso entre estes dois objetivos: o custo com a inserção de predadores e o prejuízo com as pragas ao longo dos estágios

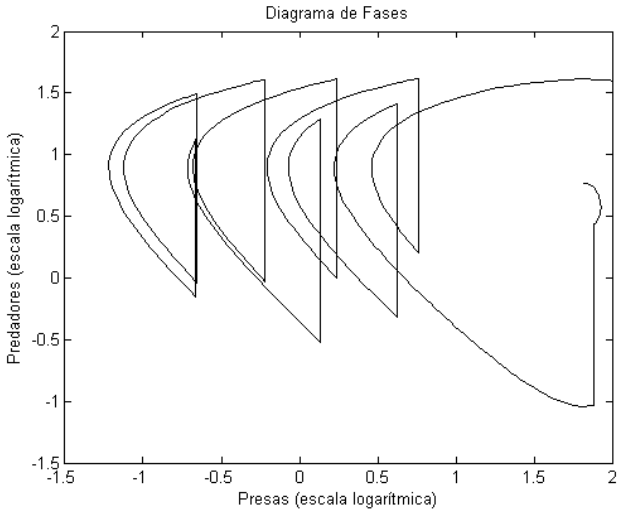

Figura 4.4: Estudo de caso: controle biológico de pragas (seção 4.3.1). Diagrama de fases do sistema presa-predador impulsivo ótimo mostrado na figura 4.3.

do problema.

Este problema pode ser formulado como em (4.19).

$$\min_{u[0],\ldots,u[N-1]} (J_1 = \sum_{k=0}^{N} x[k], J_2 = \sum_{k=0}^{N-1} u[k]) \tag{4.19}$$

$$\text{sujeito a: } \begin{cases} x'(t) = x(a - \gamma x - \alpha y); \\ y'(t) = y(-b + \beta x); \\ t \in [\tau_k^+, \tau_{k+1}]; \\ x(\tau_k^+) = x[k^+] = x[k]; \\ x[k] = x(\tau_k); \\ y(\tau_k^+) = y[k^+] = y[k] + u[k]; \\ y[k] = y(\tau_k); \\ k = 0,1,\ldots,N-1; \\ x[0] = x_0 \text{ e } y[0] = y_0 \text{ são dados}; \\ x[N] \leq 20. \end{cases}$$

Os valores dos parâmetros do modelo são os mesmos dos considerados no caso mono-objetivo.

Resolvemos este problema por meio de um AG multiobjetivo, o NSGA-II (Deb et al., 2002), com codificação real, seleção por torneio binário e mutação polinomial. Usamos os seguintes parâmetros no algoritmo: 200 indivíduos, critério de

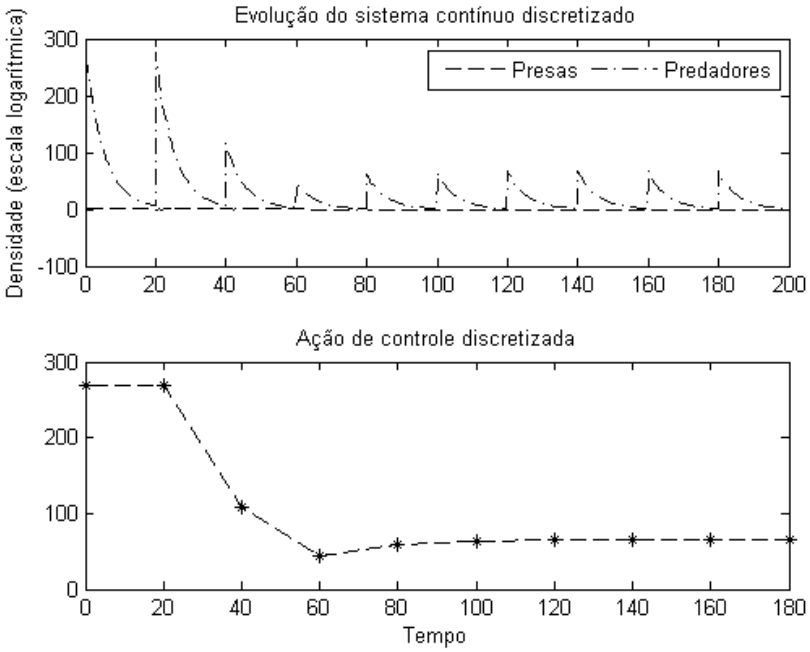

Figura 4.5: Estudo de caso: controle biológico de pragas (seção 4.3.1). Comportamento do sistema presa-predador mostrado na figura 4.2 com ação de controle discretizada. A primeira figura apresenta a evolução ótima de presas e predadores ao longo dos estágios (em escala logarítmica com base 10) e a segunda figura apresenta o gráfico com a quantidade de predadores a serem lançados em cada estágio.

parada pelo número máximo de 200 gerações, espaço de busca de cada dimensão entre 1 e 50 indivíduos. A taxa de combinação é 0,70 e a taxa de mutação é 0,05.

A figura 4.6 apresenta uma amostra da fronteira de Pareto deste problema, com 200 soluções, todas obedecendo às restrições do problema. Note que há duas regiões distintas neste conjunto, que chamamos de *região A* e *região B*. Note que na região A há uma saturação do benefício obtido com um aumento significativo do custo da ação de controle: o prejuízo com as pragas se tornou quase constante a partir do custo 300 da ação de controle. Na região B, que corresponde a variação do custo da ação de controle ao redor de 75 a 280 predadores lançados por metro quadrado, a marginal alcançada pelo lançamento de predadores na lavoura de soja permanece constante se a variável de controle se estende. Nessa região, uma relação aparentemente linear permanece entre o custo da ação de controle e o prejuízo com a lavoura. Há uma pequena região de transição entre essas duas regiões principais.

As figuras 4.7 e 4.8 apresentam duas soluções Pareto-ótimas, escolhidas por

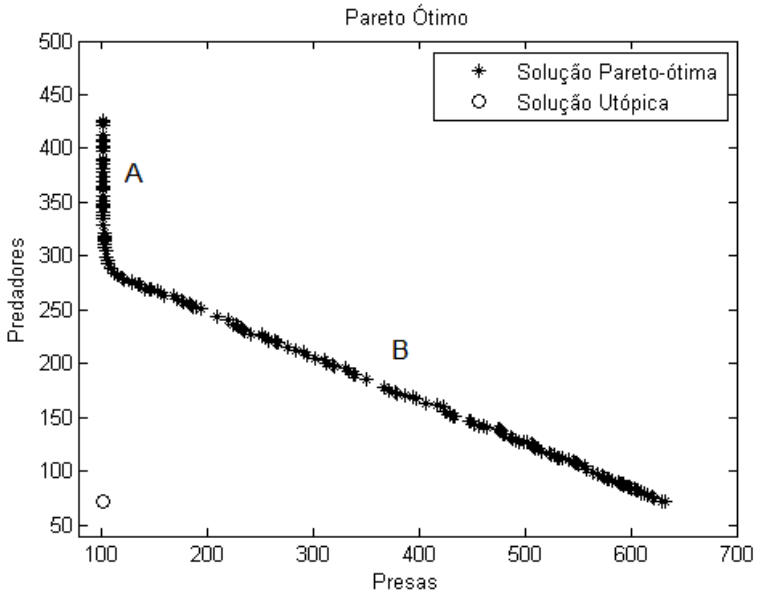

Figura 4.6: Estudo de caso: controle biológico de pragas (seção 4.3.1). Amostra da fronteira de Pareto do sistema presa-predador impulsivo biobjetivo. Os eixos correspondem à soma do número de presas e à soma do número de predadores.

estarem em posições extremas na fronteira. A primeira é referente a maior inserção de predadores, cujos valores das funções são (102,5970; 421,7071), e a segunda, referente a menor inserção de predadores, com valores das funções (512,4661; 120,9854). Mostramos em cada figura a evolução no sistema no tempo (em escala logarítmica) e a ação de controle ótima. A figura 4.9 apresenta uma comparação entre os diagramas de fases destas duas soluções de compromisso (o diagrama da direita corresponde à maior inserção de predadores, e o da esquerda, corresponde à menor inserção de predadores).

Observe o *trade-off* entre um maior número de predadores inseridos, com uma redução mais rápida no número de presas, e um custo com predadores moderado, com a redução no número de presas sendo feita mais gradativa. Note, também, que um ponto intermediário na fronteira de Pareto corresponde ao ótimo do problema com apenas um objetivo, considerando a soma das duas funções.

Do ponto de vista econômico, assinalamos que:

- O emprego de uma ação de controle de custo acima de 300 é ineficiente. Esse deve ser o valor máximo recomendado na prática.

- A estratégia a ser empregada deve ser provavelmente escolhida de um conjunto de duas alternativas: a ação de controle com custo por volta de 75 e a ação de controle com custo por volta de 290.

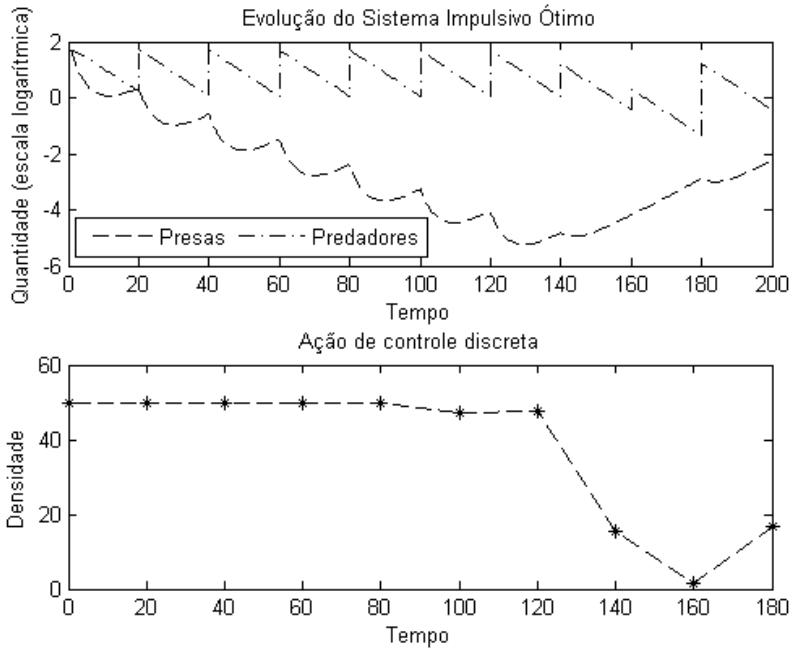

Figura 4.7: Estudo de caso: controle biológico de pragas (seção 4.3.1). Exemplo de solução Pareto-ótima do sistema presa-predador impulsivo biobjetivo com maior inserção de predadores, cujos valores das funções são (102,5970; 421,7071). A primeira figura apresenta a evolução ótima de presas e predadores ao longo dos estágios (em escala logarítmica com base 10) e a segunda figura apresenta o gráfico com a quantidade de predadores a serem lançados em cada estágio.

- Deve existir um preço crítico para a venda da soja no qual qualquer ação de controle entre 75 e 290 pode ser aceitável. Para esse preço, o custo econômico do lançamento de mais predadores (aumentando a eficiência do controle) deve ser exatamente compensado pelo ganho econômico obtido da soja adicional que deve ser colhida no fim do ciclo da lavoura.

- Por outro lado, os preços relativos da soja e da ação de controle devem definir qual estratégia deverá ser selecionada, necessariamente entre os dois extremos da região B, entre os custos 75 e 290. Um baixo preço de mercado para a soja deve direcionar a ação de controle para o que possui o custo 75: nesse caso, o ganho econômico vem da soja adicional que deverá ser colhida se mais predadores lançados não cobrir o custo com a ação de controle. Porém, um alto preço de mercado para a soja tende a escolher a ação de controle de custo por volta de 290, desde de que seja compensado com o ganho econômico obtido pela colheita.

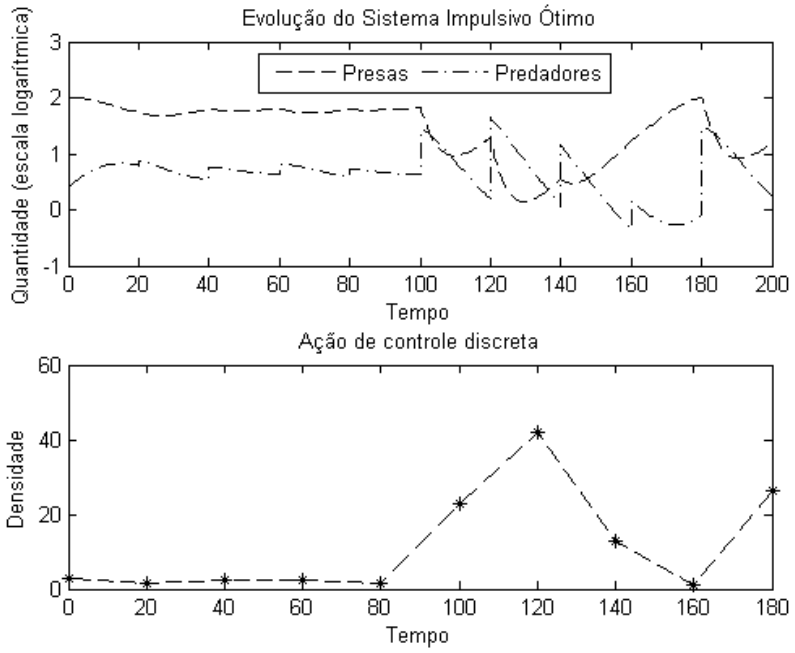

Figura 4.8: Estudo de caso: controle biológico de pragas (seção 4.3.1). Exemplo de solução Pareto-ótima do sistema presa-predador impulsivo biobjetivo com menor inserção de predadores, com valores das funções (512,4661; 120,9854). A primeira figura apresenta a evolução ótima de presas e predadores ao longo dos estágios (em escala logarítmica com base 10) e a segunda figura apresenta o gráfico com a quantidade de predadores a serem lançados em cada estágio.

A relação crítica entre o preço de mercado da soja e o custo da ação de controle que separam as duas estratégias é dada pela inclinação da reta que ajusta a região B. Do ponto de vista do produtor de soja, essa relação de fato define uma política de controle a ser adotada. No entanto, sob o ponto de vista do produtor de predadores, essa relação crítica define um preço limiar para ser seguido.

4.3.2 Controle de epidemias

Epidemia é a alteração de uma ou mais características em um número significativo de indivíduos de uma população, normalmente relacionada à perda da saúde. A interação entre os indivíduos de uma população e o meio constitui um sistema imunológico. Entendemos como sendo o meio tudo o que se relaciona com outros indivíduos.

A epidemiologia é a ciência que estuda os sistemas imunológicos, bem como a eficácia e o impacto de intervenções para se controlar os problemas de saúde.

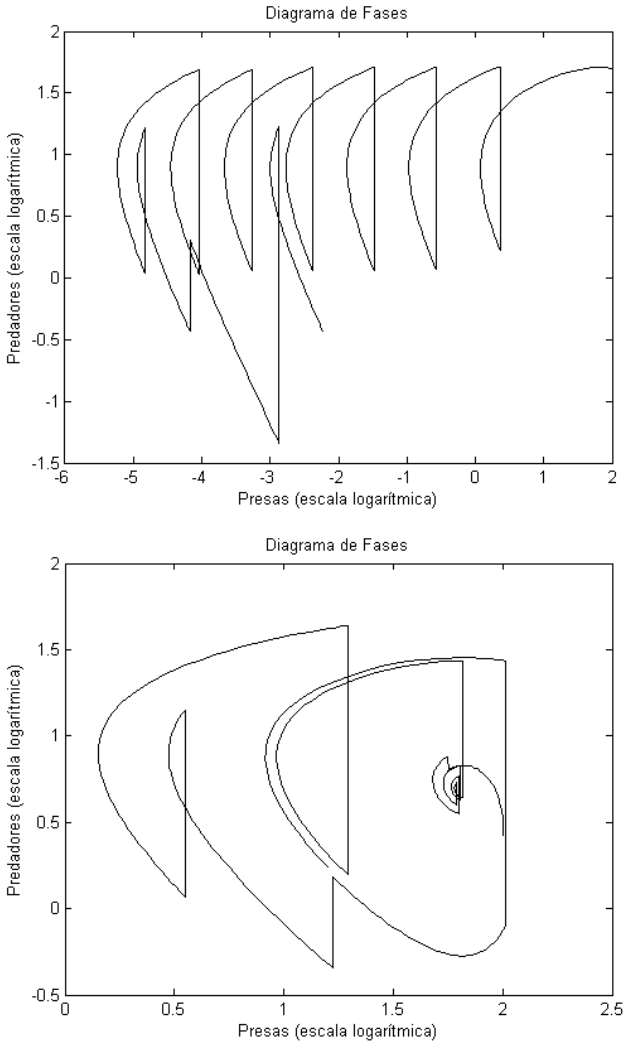

Figura 4.9: Estudo de caso: controle biológico de pragas (seção 4.3.1). Comparação entre os diagramas de fases das suas soluções Pareto-ótimas apresentadas nas figuras 4.7 e 4.8.

A epidemiologia matemática busca modelar matematicamente esta interação, com o objetivo de ajudar numa melhor compreensão dos mecanismos reais de transmissão das doenças, e assim proporcionar estratégias mais baratas e efetivas para o controle de infecções.

Neste estudo de caso[5], propomos estratégias ótimas de vacinação usando o

[5]Este trabalho foi publicado nos anais de um congresso nacional (Cardoso e Takahashi, 2007) e aceito para publicação em um periódico (Cardoso e Takahashi, 2008).

modelo SIR como sistema dinâmico e a programação dinâmica impulsiva como técnica de controle. Este estudo foi inspirado no trabalho de Nepomuceno (2005), que propõe uma estratégia de vacinação impulsiva com pulsos constantes.

Modelagem Matemática

A abordagem mais comum para se estudar a evolução de infecções, consiste em classificar os indivíduos em estados. O modelo matemático SIR classifica os indivíduos em três estados: suscetíveis (S), infectados (I) e recuperados (R). Estes três estados são relacionados por meio do sistema de equações diferenciais ordinárias (4.20).

$$\begin{cases} S'(t) = \mu N - \mu S - \beta IS/N; \\ I'(t) = \beta IS/N - \gamma I - \mu I; \\ R'(t) = \gamma I - \mu R; \end{cases} \tag{4.20}$$

em que: N representa o tamanho da população, μ é a taxa de novos suscetíveis por unidade de tempo, γ é a taxa de indivíduos infectados que são recuperados e β é o coeficiente de transmissão, ou a taxa de contatos por unidade de tempo.

Assumindo que a população total N é fixa, então $R(t) = N - S(t) - I(t)$, para todo tempo t, e podemos redefinir as variáveis como razões: $s(t) = S(t)/N$, $i(t) = I(t)/N$ e $r(t) = R(t)/N$. Assim, o sistema (4.20) pode ser escrito apenas em duas variáveis como em (4.21).

$$\begin{cases} s'(t) = \mu - \mu s - \beta is; \\ i'(t) = \beta is - \gamma i - \mu i. \end{cases} \tag{4.21}$$

Nepomuceno (2005) mostra que este modelo é epidemiológica e matematicamente bem formulado, além de apresentar uma lista mais completa de resultados, além de outras referências. Foi deste trabalho que retiramos as definições, o modelo e os parâmetros usados nesta seção. A estabilidade deste sistema é estudada através de um parâmetro: a taxa básica de reprodução R_0, que é definida em (4.22). Esta taxa representa o número médio de infecções secundárias produzidas quando um indivíduo infectado é introduzido numa população inteiramente suscetível.

$$R_0 = \frac{\beta}{\gamma + \mu} \tag{4.22}$$

Prova-se que se $R_0 \leq 1$ ou se a condição inicial é $i(0) = i_0 = 0$, as trajetórias factíveis atingirão o ponto de erradicação da doença $(s,i) = (1,0)$. Se $R_0 > 1$, as trajetórias das soluções com $i_0 > 1$ atingirão um equilíbrio endêmico dado por:

$$\bar{s} = \frac{1}{R_0} \text{ e } \bar{i} = \frac{\mu}{\beta}(R_0 - 1).$$

De fato, se $R_0 \leq 1$, então $i'(0) = \beta i(0)s(0) - (\gamma + \mu)i(0) \leq 0$, para toda condição inicial $(s(0),i(0))$, ou seja, a doença tende a desaparecer.

Os valores para as constantes utilizados neste trabalho são baseados na epidemia de sarampo no Reino Unido entre 1950-68 (Nepomuceno, 2005), e estão apresentados na tabela 4.3.

μ	γ	β	N	R_0
1/70	1/24	0,95	2×10^6	17

Tabela 4.3: Estudo de caso: controle de epidemias (seção 4.3.2). Tabela de constantes do sistema SIR usadas neste estudo de caso.

O sistema dinâmico (4.21) com os parâmetros da tabela 4.3 possui dois pontos fixos. O ponto $(\bar{s},\bar{i}) = (1,0)$, correspondente à erradicação da doença, é uma sela, assim, instável. O outro ponto fixo do sistema, para a condição inicial com $i_0 \neq 0$, é um nó assintoticamente estável, e está na tabela 4.4.

Número de suscetíveis:	$\bar{s} = 0{,}0588$
Número de infectados:	$\bar{i} = 0{,}2406$

Tabela 4.4: Estudo de caso: controle de epidemias (seção 4.3.2). Ponto fixo endêmico estável do sistema SIR cujos parâmetros estão na tabela 4.3.

A evolução do sistema autônomo (sem ação de controle) e seu diagrama de fases para a condição inicial $s_0 = 0{,}999$ e $i_0 = 0{,}001$ está na figura 4.10. Repare que na situação de equilíbrio, 24% da população estaria infectada em cada unidade de tempo - de fato, uma situação endêmica, que pede por uma ação de controle. Note que, neste exemplo, o valor do parâmetro R_0 é bem maior que 1, o que ratifica o caráter endêmico da infecção.

Otimização

O controle de epidemias tem sido amplamente estudado e as técnicas de controle mais usadas têm sido vacinação, isolamento ou ambas. Vamos trabalhar apenas com a estratégia de vacinação, mas os resultados obtidos poderiam ser facilmente adaptados para outras estratégias mudando-se apenas o sistema dinâmico em questão.

Considere o exemplo da tabela 4.3 com condição inicial $s_0 = 0{,}999$ e $i_0 = 0{,}001$. Matematicamente, como $R_0 > 1$, a erradicação da doença só aconteceria por vacinação em instantes discretos se toda a população fosse vacinada. Neste caso, a taxa de suscetíveis seria reduzida para zero. Só que esta solução costuma ser inviável do ponto de vista prático, ou de implementação muito difícil. Nosso

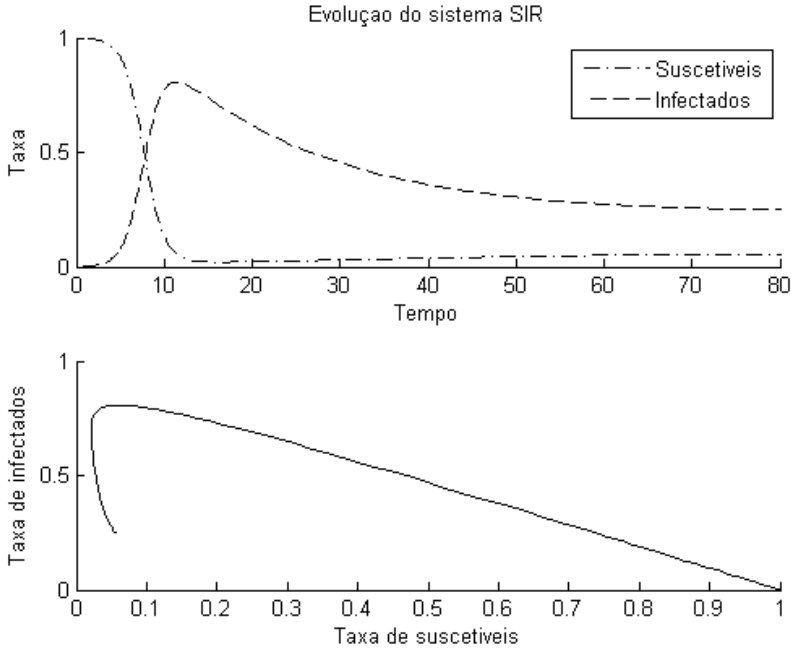

Figura 4.10: Estudo de caso: controle de epidemias (seção 4.3.2). Comportamento do sistema SIR autônomo para a condição inicial $s_0 = 0{,}999$ e $i_0 = 0{,}001$ e os parâmetros da tabela 4.3.

trabalho é baseado na estratégia de vacinação por pulsos, que é definida como a vacinação de uma parcela fixa da população (de suscetíveis) em instantes de tempo igualmente espaçados (Zhou e Liu, 2003), como no trabalho de Nepomuceno (2005). Assim, campanhas permanentes e periódicas de vacinação em massa são propostas, permitindo que o número de infectados seja levado para um patamar aceitável.

Nesta forma, vamos discretizar o horizonte de tempo T em N estágios com um período constante, de tamanho δT. Matematicamente, isto consiste em escolher uma seqüência eqüidistante de instantes de tempo $\{\tau_0, \ldots, \tau_N\}$, em que: $\tau_k < \tau_{k+1}$, $\tau_0 = 0$, $\tau_N = T$ e $\tau_{k+1} - \tau_k = \delta T$. Seja $s[k] = s(\tau_k)$ o percentual de suscetíveis no estágio k, $i[k] = i(\tau_k)$ o percentual de infectados no estágio k e $p[k]$ o percentual de vacinados no estágio k. Consideramos, neste estudo, o horizonte como sendo $T = 100$ unidades de tempo e supomos estágios de tamanho $\delta T = 1$.

A vacinação por pulsos supõe intensidade constante, ou seja $p[k] = \bar{p}$, para todo estágio k. Neste caso, para um intervalo de tempo δT dado, a vacinação por pulsos leva o sistema para pontos fixos do sistema impulsivo $(\bar{s},\bar{i})$ - veja a definição 2.1. Recentemente, a caracterização e a estabilidade dos pontos fixos impulsivos do modelo SIR tem sido matematicamente e computacionalmente estudada (Zhou

e Liu, 2003; d'Onofrio, 2005).

Mapeamos, via simulação computacional, parte dos pontos fixos associados a campanhas de vacinação por pulsos usando os dados do nosso problema. A figura 4.11 apresenta o diagrama de fases com 1000 destes pontos fixos $(\bar{s},\bar{i})$, obtidos variando-se a taxa de vacinação $\bar{p}$ entre 0 e 1. A figura 4.12 mostra, para o exemplo dado, a relação de compromisso entre o gasto com a vacinação e o resultado da vacinação, ou seja, a taxa de vacinação $\bar{p}$ e o percentual de infectados ao final do processo $\bar{i}$.

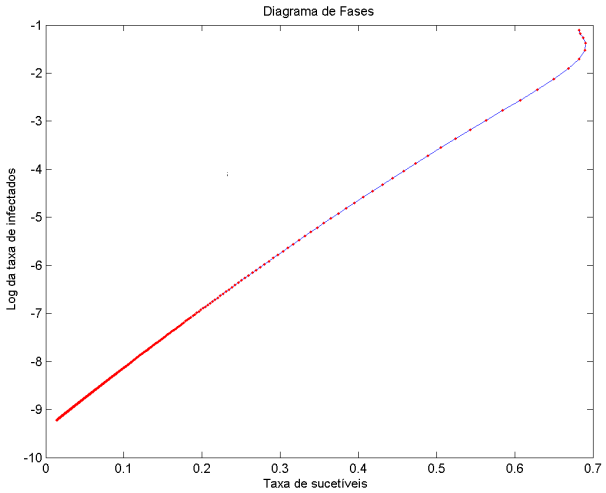

Figura 4.11: Estudo de caso: controle de epidemias (seção 4.3.2). Mapeamento dos pontos fixos $(\bar{s},\bar{i})$ do sistema SIR impulsivo com vacinação constante. Apenas $\bar{i}$ está em escala logarítmica com base 10. Cada ponto no gráfico corresponde a uma taxa de vacinação constante $\bar{p}$ entre 0 e 1.

Como exemplo, considere a vacinação por pulsos constantes com taxa $\bar{p} = 0{,}4$. A tabela 4.5 apresenta o ponto fixo do sistema impulsivo correspondente a esta taxa. A figura 4.13 mostra a evolução do sistema e seu diagrama de fases (em escala logarítmica), considerando esta estratégia de vacinação.

Número de suscetíveis: s^*	$3{,}47 \times 10^{-1}$
Número de infectados: i^*	$4{,}47 \times 10^{-8}$

Tabela 4.5: Estudo de caso: controle de epidemias (seção 4.3.2). Ponto-fixo do sistema impulsivo associado à taxa de vacinação constante $\bar{p} = 0{,}4$.

Neste trabalho, de modo a permitir pulsos não-constantes, formulamos o problema como um problema de programação dinâmica impulsiva. Assim, a variável

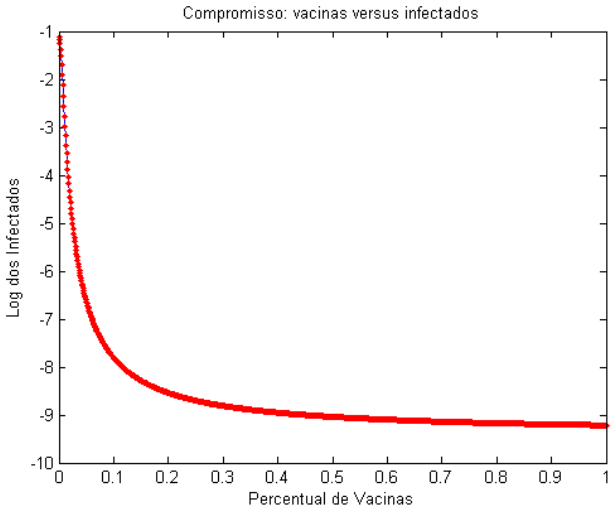

Figura 4.12: Estudo de caso: controle de epidemias (seção 4.3.2). Compromisso na escolha de um ponto fixo do sistema SIR impulsivo: percentual de vacinados a uma taxa de vacinação constante $\bar{p}$ versus percentual de infectados ao final do processo $\bar{i}$ (em escala logarítmica com base 10).

de decisão do problema é a seqüência da proporção de indivíduos suscetíveis a serem impulsivamente vacinadas em cada estágio: $\{p[0], \ldots, p[N-1]\}$.

Dados $s[0] = s_0$, $i[0] = i_0$ e uma seqüência de ações de controle $\{p[0], \ldots, p[k]\}$, os estados $s[k+1]$ e $i[k+1]$ podem ser encontrados através da resolução do PVI mostrado em (4.23).

$$\text{PVI do estágio } k\text{:} \begin{cases} s'(t) = \mu - \mu s - \beta i s; \\ i'(t) = \beta i s - \gamma i - \mu i; \\ t \in [\tau_k^+, \tau_{k+1}]; \\ s(\tau_k^+) = s[k^+] = s[k](1 - p[k]); \\ s[k] = s(\tau_k); \\ i(\tau_k^+) = i[k^+] = i[k]; \\ i[k] = i(\tau_k). \end{cases} \tag{4.23}$$

Para sabermos o número de recuperados em cada estágio k, podemos usar a expressão: $r(\tau_k) = r[k] = 1 - s[k] - i[k]$ (antes da vacinação do estágio) ou $r(\tau_k^+) = r[k^+] = 1 - s[k^+] - i[k^+]$ (depois da vacinação do estágio). Podemos também escrever em função do percentual de vacinados, pela equação impulsiva: $r(\tau_k^+) = r[k^+] = r[k] + s[k]p[k]$.

Sugerimos que a meta de programação dinâmica (s^*,i^*), a ser escolhida pelo executor de políticas públicas, seja um dos pontos fixos associados a taxas de vacinação por pulsos constantes. Posto desta forma, o problema de otimização

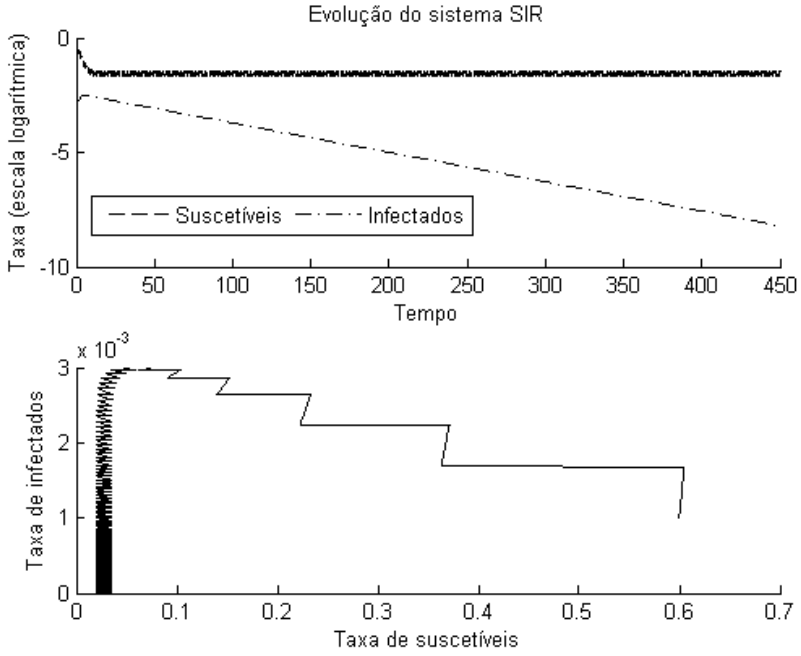

Figura 4.13: Estudo de caso: controle de epidemias (seção 4.3.2). Exemplo mostrando a evolução para o ponto fixo (mostrado na tabela 4.5) do sistema SIR impulsivo com taxa de vacinação constante $\bar{p} = 0{,}4$.

consiste em procurar a melhor estratégia para se atingir um dado ponto fixo impulsivo. Esta meta pode ser escolhida baseada na relação de compromisso mostrada na figura 4.12. O problema é que, computacionalmente, pode ser difícil otimizar na pré-imagem da meta de programação, por isso permitimos uma relaxação. Se consideramos uma folga de $\epsilon = 10^{-4}$ na meta, e se $i^* = 10^{-5}$, estamos permitindo que ao final do processo de otimização menos de 0,01% da população esteja infectada em cada unidade de tempo, o que pode ser razoável do ponto de vista prático.

Não temos, em geral, maiores informações geométricas sobre a pré-imagem de conjuntos por este sistema dinâmico, mas como ilustração, a figura 4.14 mostra a evolução das pré-imagens de um retângulo em torno de um ponto fixo (endêmico) pelo sistema SIR autônomo. A figura de cada tonalidade representa a pré-imagem da bola um estágio atrás. Repare que, embora não saibamos descrever geometricamente as pré-imagens desses conjuntos, percebemos que eles seguem a forma da órbita do sistema dinâmico (compare com a figura 4.10).

A fim de trabalharmos com funções convexas[6], consideramos uma função-

[6]Mesmo usando função-objetivo convexa, globalmente o problema continua não-convexo

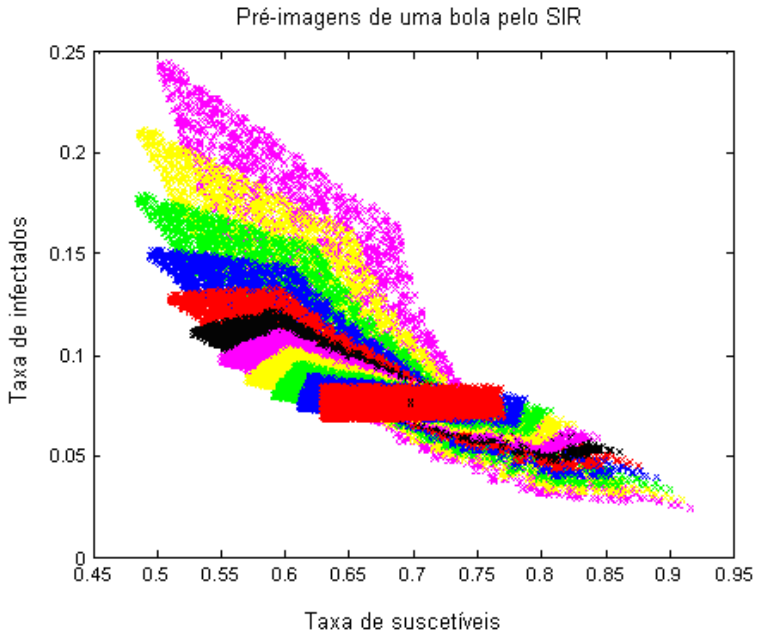

Figura 4.14: Estudo de caso: controle de epidemias (seção 4.3.2). Evolução das pré-imagens de uma bola quadrada em torno de um ponto fixo do sistema SIR autônomo. A figura de cada tonalidade representa a pré-imagem da bola um estágio atrás. Assim, a primeira figura (por cima de todas) é o retângulo, a figura logo atrás representa a pré-imagem deste retângulo por um estágio, a figura logo atrás representa a imagem deste retângulo dois estágios atrás, depois o mesmo para as demais figuras, e assim sucessivamente.

objetivo quadrática: desejamos minimizar uma média quadrática entre o custo com infectados e o custo com as vacinas.

A formulação do problema de otimização considerando a pré-imagem de uma bola de um raio ϵ dado em torno da meta de programação pode ser como em (4.24).

$$\min_{p[0],\ldots,p[N-1]} \sum_{k=0}^{N-1} (c_k i[k]^2 + d_k p[k]^2) + c_N i[N]^2 \tag{4.24}$$

devido à restrição dinâmica. A escolha da função-objetivo apenas melhora as propriedades do problema global, o que pode ser verificado nos experimentos computacionais.

$$\text{sujeito a: } \begin{cases} s'(t) = \mu - \mu s - \beta i s; \\ i'(t) = \beta i s - \gamma i - \mu i; \\ t \in [\tau_k^+, \tau_{k+1}]; \\ s(\tau_k^+) = s[k^+] = s[k](1 - p[k]); \\ s[k] = s(\tau_k); \\ i(\tau_k^+) = i[k^+] = i[k]; \\ i[k] = i(\tau_k); \\ s[0] = s_0 \text{ e/ou } i[0] = i_0 \text{ podem ser dados}; \\ ||s[N] - s^*|| \leq \epsilon^2 \text{ e } ||i[N] - i^*|| \leq \epsilon^2. \end{cases}$$

Em que: $||\cdot||$ é uma norma qualquer, c_k e d_k são constantes associadas às taxas de infectados e de vacinados, respectivamente. Repare que, nos dois casos, a ação de controle (vacinação impulsiva) incide apenas sobre o número de suscetíveis, em cada estágio.

Resolvemos esse problema em malha aberta pelo método de programação quadrática seqüencial (SQP)[7] implementado pela função `fmincon` do Matlab. Consideramos como meta de programação o ponto-fixo mostrado na tabela 4.5, referente à taxa de vacinação com percentual 0,4. O percentual de vacinados de cada estágio é procurado entre 0,4 (pois este é o valor fixo de vacinação necessário para se estabelecer o ponto-fixo correspondente à meta de programação) e 0,8 (supondo ser muito difícil vacinar mais que 80% de uma população). Os valores das constantes da função-objetivo são $c_k = 0{,}9 \times 10^7$ e $d_k = 0{,}1 \times 10$.

A figura 4.15 apresenta o gráfico mostrando a evolução ótima da taxa de suscetíveis e de infectados ao longo dos estágios (em escala logarítmica), bem como o gráfico que mostra o percentual ótimo de vacinados em cada um dos estágios. O valor da função-objetivo ótima encontrado foi 190,20.

Repare que o o número de suscetíveis atinge o ponto fixo desde pouco antes de $T = 50$, ou seja, a partir da metade do horizonte de otimização. Repare também que, como especificado, $i[k] = i[k^+]$, para cada estágio k.

A figura 4.16 apresenta os diagramas de fases referentes, respectivamente, à estratégia por pulsos constantes, como proposto em Nepomuceno (2005), com taxa $\bar{p} = 0{,}4$ (figura da esquerda) e à estratégia aqui proposta, tendo como meta a taxa $\bar{p} = 0{,}4$ (figura da direita). Na primeira figura, a proporção de infectados chega a 10^{-4} em aproximadamente 120 estágios e no processo proposto, a proporção de infectados chega a 10^{-4} em aproximadamente 65 estágios. O valor da função objetivo para a vacinação constante com esta taxa é de 18.529 (um valor muito maior que o encontrado pela nossa formulação). Acreditamos que isto seja devido o fato de se vacinar 80% da população logo nos 20 primeiros estágios do

[7]Este método resolve um problema de programação quadrática a cada iteração, com a estimativa da Hessiana da função Lagrangeana dada pela fórmula BFGS (Fletcher e Powell, 1963).

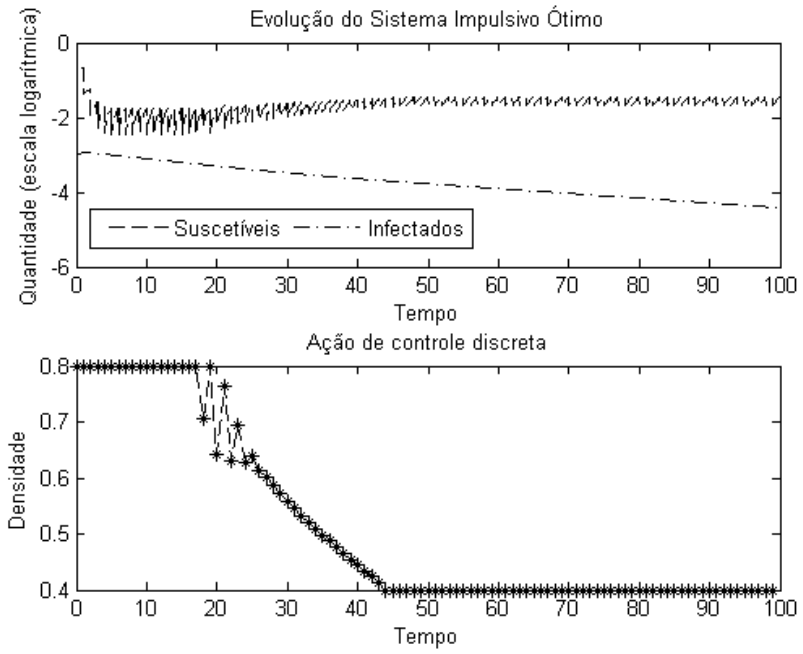

Figura 4.15: Estudo de caso: controle de epidemias (seção 4.3.2). Evolução do sistema SIR impulsivo ótimo. A primeira figura apresenta a evolução ótima de suscetíveis e infectados ao longo dos estágios (em escala logarítmica com base 10) e a segunda figura apresenta o gráfico com o percentual ótimo de suscetíveis a serem vacinados em cada estágio.

problema. Considerando-se que se vacinar 80% de uma população pode não ser muito fácil, a alternativa proposta de começar com uma taxa de vacinação mais alta, reduzir gradativamente nos estágios seguintes, e só aí começar a vacinação impulsiva constante com $\bar{p} = 0{,}4$ se mostra bastante eficaz, também do ponto de vista prático.

A vacinação por pulsos tem sido colocada na literatura como promissora, mas a técnica da programação dinâmica impulsiva se mostra ainda mais eficiente, do ponto de vista da redução de custos, além de incorporar a facilidade da programação da ação em tempo discreto, mesmo com a evolução do sistema em tempo contínuo.

Não apresentaremos os resultados aqui, mas fizemos testes com funções-objetivos lineares, não-convexas e descontínuas. Testamos também com outros métodos de otimização, como elipsoidal e algoritmos genéticos. Fizemos também testes otimizando na pré-imagem de outros pontos fixos. Em todos os casos, os resultados obtidos foram satisfatórios, do ponto-de-vista computacional e do ponto-de-vista prático. Reforçamos que poderíamos facilmente usar esta técnica

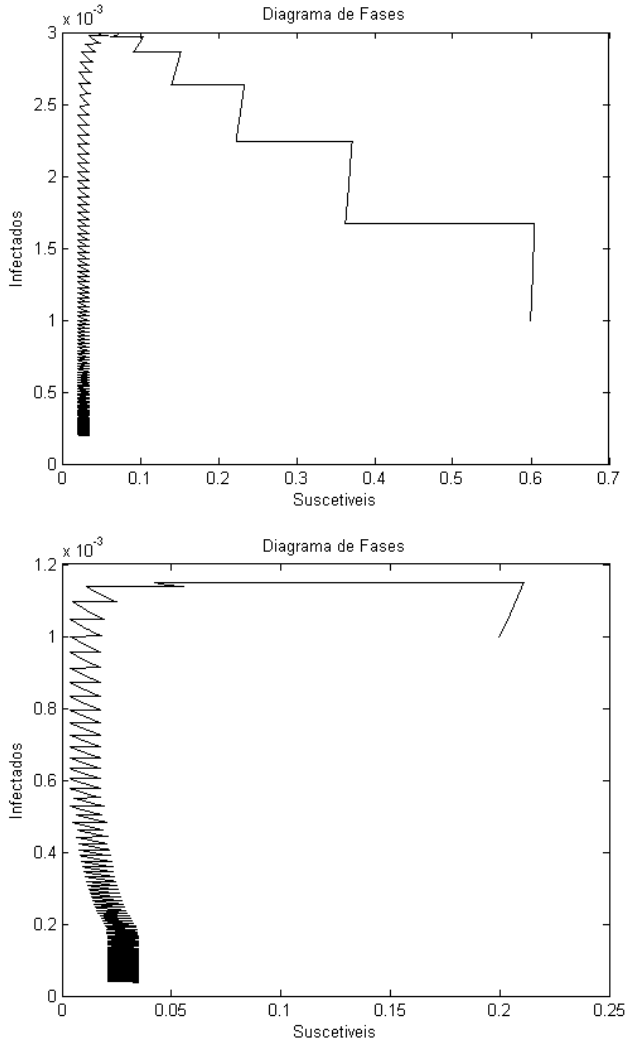

Figura 4.16: Estudo de caso: controle de epidemias (seção 4.3.2). Comparação entre os diagramas de fases das duas soluções apresentadas: a primeira é referente à solução com pulsos constantes (mostramos a evolução no tempo na figura 4.13) e a segunda, à solução por programação dinâmica impulsiva (mostramos a evolução no tempo na figura 4.15).

em outras estratégias de controle, como isolamento ou até mesmo numa estratégia mista, mudando apenas o sistema dinâmico em questão.

Abordagem multiobjetivo

Em vez de considerarmos como função-objetivo deste problema uma média entre entre o custo com infectados e o custo a vacinação, podemos pensar numa formulação biobjetivo (como apresentado na seção 4.2.3): encontrar soluções de compromisso entre a minimização do prejuízo com os infectados e o custo de vacinação ao longo dos estágios do problema.

Este caso, o problema de otimização restrita à pré-imagem de uma bola de raio ϵ em torno da meta de programação, pode ser formulado como em (4.25).

$$\min_{p[0],\ldots,p[N-1]} (J_1 = \sum_{k=0}^{N} c_k i[k]^2, J_2 = \sum_{k=0}^{N-1} d_k p[k]^2) \tag{4.25}$$

$$\text{sujeito a: } \begin{cases} s'(t) = \mu - \mu s - \beta i s; \\ i'(t) = \beta i s - \gamma i - \mu i; \\ t \in [\tau_k^+, \tau_{k+1}]; \\ s(\tau_k^+) = s[k^+] = s[k](1 - p[k]); \\ s[k] = s(\tau_k); \\ i(\tau_k^+) = i[k^+] = i[k]; \\ i[k] = i(\tau_k); \\ s[0] = s_0 \text{ e/ou } i[0] = i_0 \text{ podem ser dados}; \\ ||s[N] - s^*|| \leq \epsilon^2 \text{ e } ||i[N] - i^*|| \leq \epsilon^2. \end{cases}$$

Resolvemos este problema via técnicas de escalarização e usamos os mesmos parâmetros do problema mono-objetivo e resolvemos cada problema escalarizado em malha aberta também pela função `fmincon` do Matlab.

A figura 4.17 apresenta uma amostra da fronteira de Pareto deste problema e a figura 4.18 apresenta uma amostra do conjunto de soluções Pareto-ótimas deste problema.

Repare na figura 4.18 que todas as soluções seguem o mesmo comportamento:

1. a vacinação da maior parte da população (pelo menos 80%) nos primeiros estágios do problema, por meio de uma ampla campanha na mídia;

2. a redução gradativa do percentual de vacinados;

3. implantar a vacinação por pulsos constantes com taxa $\bar{p}$ previamente escolhida, de acordo com patamar de infectados aceitável para a infecção.

O que muda nas soluções Pareto-ótimas é a duração de cada uma destas fases, que varia de acordo com as prioridades do executor de políticas públicas: gastar mais com a profilaxia (valorizando um maior tempo de vacinação em massa, reduzindo mais rapidamente o número de infectados) ou com o possível tratamento.

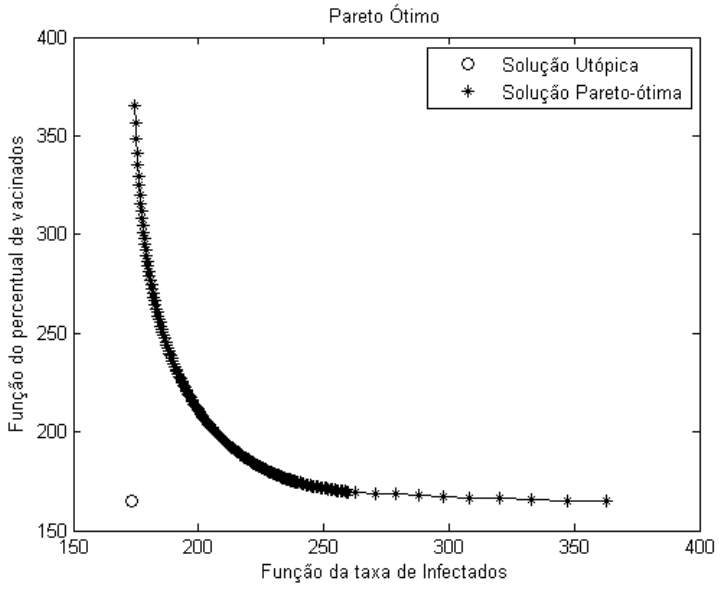

Figura 4.17: Estudo de caso: controle de epidemias (seção 4.3.2). Amostra da fronteira de Pareto do sistema SIR impulsivo biobjetivo. Os eixos são as funções J_1 (função da soma da taxa de infectados) e J_2 (função da soma do percentual de suscetíveis a serem vacinados.)

4.4 Conclusões do capítulo

Neste capítulo, estendemos para o caso não-linear a abordagem proposta para o caso linear. Apresentamos formulações para o problema com sistema dinâmico em tempo discreto e para o problema dinâmico impulsivo, nos casos mono e multiobjetivo. Discutimos dois estudos de casos relevantes, usando a abordagem proposta: o controle biológico de pragas numa lavoura via modelo presa-predador e a busca de estratégias ótimas de vacinação via modelo de epidemias SIR.

Estes exemplos nos levam a concluir que a abordagem proposta: (i) do ponto de vista computacional, é mais barata que a técnica ótima de programação dinâmica em tempo discreto; (ii) do ponto de vista da função-objetivo, é mais barata que as técnicas impulsivas por pulsos constantes; (iii) do ponto de vista prático, é mais realística que as técnicas de controle em tempo contínuo, ou é mesmo a única possibilidade, uma vez que o controle contínuo teria que ser discretizado a fim de ser implementado; (iv) parece ser mais precisa e robusta que as técnicas de controle em tempo contínuo seguidas de discretização, uma vez que esta depende fortemente da forma como a discretização é feita.

Parece ser plausível extrair conclusões genéricas a partir do exame destes casos particulares, conjecturando, assim, que estas propriedades sejam extensivas

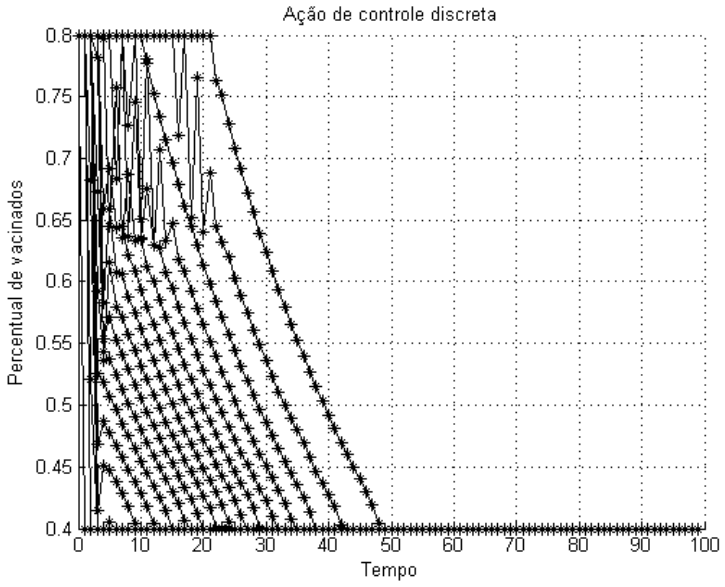

Figura 4.18: Estudo de caso: controle de epidemias (seção 4.3.2). Amostra do conjunto de soluções Pareto-ótimas do sistema SIR impulsivo biobjetivo, em que cada linha corresponde a uma solução Pareto-ótima. Cada solução corresponde a uma seqüência de percentuais de suscetíveis a serem vacinados ao longo dos estágios do problema. Todas as soluções apresentam a mesma tendência: começam no nível 0,8 e gradativamente descem para o nível 0,4.

às classes de problemas dinâmicos com natureza similar.

As informações sobre a função-objetivo e/ou sobre a função da equação de estado do sistema, tais como diferenciabilidade, unimodalidade ou convexidade, ajudam na escolha do melhor o método de otimização a ser usado para se resolver o problema em malha aberta. Apesar disto, observamos que as formulações aqui propostas, de certa forma, independem destas características.

Como estudos futuros para esta classe de problemas, propomos estudar aplicações em que o sistema dinâmico seja modelado por outras equações não-lineares discretas no tempo. Por exemplo, o problema da vacinação ótima para controle de epidemias poderia ser estudada via modelos baseados em indivíduos (Nepomuceno, 2005), considerando a probabilidade de erradicação da doença como critério de otimização.

Capítulo 5

Ferramentas para Programação Dinâmica Estocástica

Neste capítulo, estendemos para a programação dinâmica estocástica a metodologia apresentada para a programação dinâmica determinística. Propomos a otimização em malha aberta, via simulações de Monte Carlo, numa abordagem multi-quantil.

Na seção 5.1, apresentamos a formulação geral de um problema de programação dinâmica estocástica em tempo discreto. Na seção 5.2, apresentamos a nossa proposta para programação dinâmica estocástica: na subseção 5.2.1, consideramos o caso linear, na subseção 5.2.2, consideramos o caso geral não-linear, e na subseção 5.2.3, o caso multiobjetivo. Na subseção 5.3.1, apresentamos um estudo de caso clássico para esta classe de problemas: o controle de estoques ótimo, supondo demandas assimétricas e multimodais.

5.1 Introdução

O problema de programação dinâmica estocástica com tempo discreto foi discutido na subseção 2.2.2 e sua formulação está na equação (2.15), reescrita aqui para comodidade do leitor.

$$\min_{u[0],\ldots,u[N-1]} \mathrm{E}\left[\sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N])\right]$$

$$\text{sujeito a: } \begin{cases} x[k+1] = f(x[k],u[k],w[k]); \\ k = 0,1,\ldots,N-1; \\ \text{distribuição de cada } w[k] \text{ dada}; \\ x[0] = x_0 \text{ e/ou } x[N] = x^* \text{ dados}. \end{cases}$$

O distúrbio de cada estágio do problema $w[k]$ é uma variável aleatória independente. Assim, dado um horizonte N, a seqüência de distúrbios aleatórios $\{w[0], \ldots, w[N-1]\}$ é um processo estocástico. Também o estado inicial pode ser aleatório.

A equação de estados de um sistema estocástico discreto está na equação (2.13), reescrita aqui para comodidade do leitor.

$$x[k+1] = f(x[k],u[k],w[k]).$$

Como discutido na subseção 2.2.2, se o número de variáveis for finito, a maneira tradicional de se encontrarem soluções numéricas para esta classe de problemas pode ser vista como uma busca em estágios seqüenciais num grafo direcionado, a um custo computacional exponencial em função do número de variáveis de estado, das possibilidades de transição e do número de estágios do problema. Assim, a solução sofre da chamada "maldição da dimensionalidade". Se o problema for com variáveis contínuas, as soluções numéricas podem ser encontradas pelo algoritmo tradicional através da discretização destas variáveis.

5.2 Formulações propostas

A maioria dos exemplos de programação dinâmica estocástica considera variáveis aleatórias com distribuição normal[1] e, em tantas vezes, com média zero (Kushner, 1971). Isto pode ser verdade em algumas aplicações práticas, mas não em todas. Muitos autores modelam o problema assim, porque a forma analítica do valor esperado da função-objetivo de uma variável com outras distribuições pode ficar muito complicada ou até mesmo impossível de ser encontrada. Entretanto, a possibilidade de se considerar certas distribuições "não-padrões" pode tornar o modelo do problema bem mais realístico.

De fato, recentemente, funções com distribuições "não-padrões" têm sido consideradas em problemas de programação dinâmica, via técnicas sub-ótimas. Por exemplo, o artigo de Gallego et al. (2007) afirma que em muitas situações, como num problema de estoque, quando o desvio padrão da demanda cresce, a distribuição normal não deve ser usada; os autores propõem usar distribuições não-negativas, como a lognormal, a gama, ou a binomial negativa. Também o trabalho de Chen et al. (1999) usa a demanda prevista (também num problema de estoque) com distribuição lognormal.

[1] Ao longo deste capítulo usamos vários termos de estatística e de probabilidade, tais como: variáveis aleatórias, distribuições de probabilidade, valor esperado, quantil, boxplot, histograma, etc. Estes conceitos estão bem explicados nos livros da área, tais como: Meyer (1983); Ross (2002a); Triola (2005).

Mais ainda, em problemas de otimização estocástica, o valor esperado da função-objetivo tem sido usado como (único) critério de otimização, fato que pode ser razoável, desde que o distúrbio do sistema siga uma distribuição com certa regularidade, como simetria e unimodalidade. Entretanto, este critério pode não ser a melhor referência em problemas, por exemplo, com distribuições assimétricas e/ou multimodais. Nesta linha, propomos uma abordagem multi-quantil: procurar a variável de decisão ótima referente a diversos quantis da função-objetivo.

Em vista disso, tratamos a aleatoriedade dos sistemas via simulações de Monte Carlo. Mais especificamente, usamos de simulações para computar e ordenar os valores dos quantis da função-objetivo de diferentes candidatos a seqüência de ações de controle ótima. O estudo de problemas de programação dinâmica estocástica via simulações de Monte Carlo não é uma novidade, embora esta técnica ainda tenha sido pouco usada, por causa da sua alta complexidade computacional. Quando usada, tem sido baseada em métodos sub-ótimos (vide referências citadas na seção 2.5).

Simulação de Monte-Carlo, ou apenas simulação é o método de se determinar empiricamente probabilidades por meio de experimentos (computacionais). Desde o avanço dos microcomputadores, as simulações de Monte Carlo têm sido usadas nos mais diversos contextos e na solução de vários problemas do mundo real (Ross, 2002a,b). Analisar um modelo estocástico via simulações de Monte Carlo costuma ser uma boa alternativa, principalmente quando é difícil ou impossível analisar analiticamente um processo estocástico, relativo a certas distribuições de probabilidade, porque um gerador de números aleatórios pode facilmente ser usado para gerar valores de variáveis aleatórias de uma distribuição arbitrária.

Propomos, também para a classe de problemas de programação dinâmica estocástica, a otimização em malha aberta, considerando o problema a partir de um ponto de vista geométrico, como discutido nos capítulos 3 e 4 desta tese. Entretanto, na solução dos problemas estocásticos, costuma-se adiar a escolha do valor da variável de decisão de cada estágio para quando o valor do estado atual for conhecido. Quando a otimização é feita em malha fechada, como todos os caminhos são previamente listados, ou seja, como é conhecida uma regra de decisão ótima para cada estado possível, não existe penalidade alguma em adiar esta escolha.

Desta forma, ao propormos a resolução de um problema de programação dinâmica estocástica em malha aberta, precisamos reforçar **a necessidade de atualizar o resultado da otimização estágio a estágio**, buscando considerar o valor exato da variável de estado anterior no cálculo da ação de controle ótima do estágio em questão. Este procedimento sub-ótimo é similar ao controle por realimentação em malha aberta - *open-loop feedback control (OLFC)* - descrito

na seção 2.5. Isso significa que, apesar de gerarmos ações de controle em malha aberta, a estratégia proposta torna-se, mesmo que de forma sub-ótima, em malha fechada, devida a atualização passo-a-passo, em cada nova execução, das condições do estado do problema, de alguma forma observadas.

Assim sendo, a metodologia proposta para se calcular a variável de decisão a ser implementada em cada estágio é a seguinte:

Algoritmo Proposto - Programação Dinâmica Estocástica

1. Escolher um quantil da função-objetivo para ser a referência do critério de otimização.

2. Gerar aleatoriamente várias seqüências de distúrbios com a distribuição dada.

3. Para cada seqüência de distúrbios gerada, encontrar a seqüência de ações de controle ótima, que minimize o quantil em questão da função-objetivo do problema, como em um problema de otimização em malha aberta.

4. Para cada seqüência de ações de controle encontrada no passo anterior, calcular o respectivo valor do quantil da função-objetivo do problema.

5. Escolher a seqüência de ações de controle referente ao valor mínimo.

Do ponto de vista computacional, não importa se cada seqüência de distúrbios é gerada *a priori* ou *on line*. Nas nossas simulações, geramos *a priori*, e usamos a mesma seqüência de distúrbios gerada para calcular as variáveis de decisão ótimas (no passo 3) e para computar o valor da função-objetivo de cada seqüência candidata à otima (no passo 4), a fim de efetuarmos uma comparação mais coerente.

Reforçamos mais uma vez que em cada processo, o procedimento descrito neste algoritmo deverá ser rodado por estágio, assim que o valor do estado atual for conhecido, e em cada estágio, apenas a ação de controle atual deve ser aplicada. Este processo de realimentação, quando consideramos a restrição de ponto final, como tem acontecido nos exemplos desta tese, torna-se semelhante à recente técnica do **controle preditivo** - *model predictive control* (MPC) - também descrita na seção 2.5.

Por fim, assinalamos que a convergência do algoritmo acima, para encontrar a ação de controle ótima de cada estágio, depende da convergência da simulação de Monte Carlo: quanto maior o número de seqüências geradas, maior será a precisão da resposta. E depende também da convergência do método de otimização escolhido para cada solução em malha aberta.

5.2.1 Caso linear

A equação (5.1) mostra o estado de um sistema dinâmico linear estocástico discreto no tempo. As matrizes A, B e C devem ter a dimensão apropriada.

$$x[k+1] = Ax[k] + Bu[k] + Cw[k]. \tag{5.1}$$

Inspirado no lema 3.1, a equação (5.2) nos mostra como reescrever o estado de cada estágio como função não apenas do estado inicial e da seqüência de ações de controle anteriores, mas também como função da seqüência de variáveis de distúrbio anteriores (fixadas).

$$x[k] = A^k x[0] + \sum_{i=0}^{k-1} A^i Bu[k-(i+1)] + \sum_{i=0}^{k-1} A^i Cw[k-(i+1)] \tag{5.2}$$

Consideremos uma função-objetivo linear. A equação (5.3) apresenta a formulação de um problema de programação linear estocástica para a otimização restrita à pré-imagem da meta de programação.

$$\min_{u[0],\ldots,u[N-1]} Q_i \left[\sum_{k=0}^{N-1} (c_k^T x[k] + d_k^T u[k]) + c_N^T x[N] \right] \tag{5.3}$$

$$\text{sujeito a: } \begin{cases} A^N x[0] + \sum_{i=0}^{N-1} A^i Bu[N-(i+1)] + \\ \quad \sum_{i=0}^{N-1} A^i Cw[N-(i+1)] = x^*; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] = x_0 \text{ pode ser dado.} \end{cases}$$

Em que $Q_i[\cdot]$ representa o quantil i da variável aleatória em questão e os custos unitários c_k e d_k são vetores-colunas.

Devido à aleatoriedade do sistema, pode ser difícil exigir que a meta de programação x^* dada seja plenamente atingida. Assim, temos mais uma motivação para considerar a formulação otimizando na pré-imagem de um conjunto em torno da meta, de forma a tentar, de certa forma, controlar a distância do valor do estado final em relação ao valor da meta.

A equação (5.4) apresenta a formulação estocástica de um problema de programação linear estocástica para a otimização restrita à pré-imagem de um hipercubo de raio ϵ em torno da meta de programação.

$$\min_{u[0],\ldots,u[N-1]} Q_i \left[\sum_{k=0}^{N-1} (c_k^T x[k] + d_k^T u[k]) + c_N^T x[N] \right] \tag{5.4}$$

$$\text{sujeito a: } \begin{cases} e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad + \sum_{i=0}^{N-1} A^i Cw[N-(i+1)] - (x^* + e_i\epsilon)) \leq 0, \\ \quad \text{para cada } i; \\ -e_i^T(A^N x[0] + Bu[N-1] + \sum_{k=1}^{N-1} A^k Bu[N-(k+1)] + \\ \quad + \sum_{i=0}^{N-1} A^i Cw[N-(i+1)] - (x^* - e_i\epsilon)) \geq 0, \\ \quad \text{para cada } i; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] = x_0 \text{ pode ser dado.} \end{cases}$$

Para cada um dos casos, apresentamos apenas a formulação do primeiro problema a ser resolvido, em que encontraríamos a seqüência ótima:

$$\{u[0], \ldots, u[N-1]\},$$

mas só implementaríamos a decisão atual $u[0]$. Assim que o valor do estado $x[1] = Ax[0] + Bu[0] + Cw[0]$ for conhecido, um novo problema de otimização deverá ser resolvido para o intervalo de tempo $k = 1, \ldots, N$, que encontraríamos nova seqüência ótima

$$\{u[1], \ldots, u[N-1]\}.$$

Da mesma forma, apenas a decisão atual $u[1]$ seria implementada, e assim sucessivamente.

Se as demais restrições do problema também forem lineares, podemos resolver cada problema em malha aberta via algoritmos de pontos interiores (Gonzaga, 1990), a um custo computacional polinomial da ordem do número de restrições e com um número de variáveis igual ao número de variáveis de controle.

5.2.2 Caso não-linear

A equação (2.13) mostra o estado de um sistema estocástico discreto, reescrita aqui para a comodidade do leitor. Para que este sistema seja não-linear, basta que a função f o seja.

$$x[k+1] = f(x[k],u[k],w[k]).$$

Para cada seqüência de distúrbios fornecida pelo gerador de números aleatórios do computador, consideramos, via recursividade, o estado em cada estágio em função do estado inicial, da seqüência de ações de controle anteriores e da seqüência destes distúrbios anteriores (fixada). A equação (5.5) apresenta este resultado.

$$x[k] = f(\ldots f(f(x[0],u[0],w[0]), u[1],w[1]), \ldots, u[k-1],w[k-1]). \tag{5.5}$$

Assim, dada uma meta de programação x^*, o problema de programação dinâmica discreta não-linear estocástica pode ser formulado como em (5.6).

$$\min_{u[0],\ldots,u[N-1]} Q_i \left[\sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \right] \tag{5.6}$$

$$\text{sujeito a: } \begin{cases} f(\ldots f(x[0],u[0],w[0]),\ldots,u[N-1],w[N-1]) = x^*; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] \text{ pode ser dado.} \end{cases}$$

O problema de programação dinâmica estocástica restrito à pré-imagem de uma bola de raio ϵ em torno da meta no estágio final pode ser formulado como em (5.7).

$$\min_{u[0],\ldots,u[N-1]} Q_i \left[\sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]) \right] \tag{5.7}$$

$$\text{sujeito a: } \begin{cases} ||(f(\ldots f(x[0],u[0],w[0]),\ldots,u[N-1],w[N-1]) - x^*)||_2 \leq \epsilon; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] \text{ pode ser dado.} \end{cases}$$

Para cada um dos casos, apresentamos apenas a formulação do primeiro problema a ser resolvido (referente ao primeiro estágio de otimização).

Como comentamos no caso determinístico, computacionalmente falando, temos que substituir o valor de cada estado $x[k]$ presente na função-objetivo, bem como nas demais restrições do problema, via equação recursiva (5.5). E se soubermos sobre diferenciabilidade, unimodalidade ou convexidade das funções envolvidas na otimização, poderemos escolher o método de otimização mais apropriado para cada iteração da simulação a ser resolvida. De toda forma, podemos dispor de métodos de populações para resolver cada problema determinístico, que independem das características das funções do problema, como os algoritmos genéticos.

5.2.3 Abordagem multiobjetivo

No contexto do que estamos propondo, a motivação para a abordagem multiobjetivo consiste em considerar, simultaneamente, diversos quantis da função-objetivo estocástica como critérios de otimização.

Esta abordagem é baseada na dominância estocástica de primeira ordem, que muitas vezes a literatura chama simplesmente de dominância estocástica ou

método de comparações intervalares (Hadar e Russell, 1969). Desde o início da década de 1970, esta noção vem sendo largamente aplicada na análise de risco, em diversos contextos como economia, finanças, estatística e pesquisa operacional. A idéia central da dominância estocástica consiste em estabelecer critérios para que possamos ordenar duas variáveis aleatórias (dizer qual seria estocasticamente maior), e assim escolher uma opção.

Considere duas funções de distribuição acumuladas F e G sobre uma variável aleatória X. Segundo Levy (1992), num problema de maximização, a função de distribuição acumulada F tem dominância estocástica de primeira ordem sobre a função G quando:

$$F(x) \leq G(x), \text{ para todo } x \in X. \tag{5.8}$$

Esta definição vale entre quaisquer duas distribuições em que uma distribuição acumulada se ponha inteiramente ou parcialmente sobre a outra. O artigo apresenta também outras formas de se definir dominância estocástica, vantajosas em outras áreas.

Relações de dominância mais fracas, ditas como de ordem superior, também podem ser consideradas. Por exemplo, a dominância estocástica de segunda ordem de F sobre G ocorre quando:

$$\int_{-\infty}^{+\infty} [G(t) - F(t)]dt \geq 0, \text{ para todo } x \in X. \tag{5.9}$$

Esta dominância é dita mais fraca, porque dominância de primeira ordem implica em dominância de segunda ordem. Usando integrais múltiplas, define-se dominância de qualquer ordem.

Para uma abordagem usando quantis, que costuma ser mais útil na prática, considere $Q_i[F]$ e $Q_i[G]$ representando o quantil i das distribuições F e G, respectivamente. Dizemos que a função F tem dominância estocástica de primeira ordem sobre a função G quando:

$$Q_i[F] \geq Q_i[G], \text{ para todo } i, \tag{5.10}$$

com a desigualdade estrita valendo para pelo menos um i.

O algoritmo básico para se verificar a dominância estocástica de primeira ordem usando m quantis consiste em: suponha que existam n distribuições a serem comparadas e m observações para cada distribuição, que chamaremos de X_i. Considere X_{ij} como sendo a observação j da distribuição i. Assume-se cada distribuição esteja em ordem crescente, ou seja, $X_{i1} \leq X_{i2} \leq \ldots \leq X_{im}$, para cada i. Desta forma, X_1 domina X_2 se $X_{1j} \geq X_{2j}$, para todo j, com a desigualdade valendo para pelo menos um j; e assim, sucessivamente.

Usando este conceito no nosso trabalho, gostaríamos de encontrar a seqüência de ações de controle que pudessem, ao mesmo tempo, minimizar todos os quantis da função-objetivo, ou, pelo menos, ser menor ou igual aos demais e estritamente menor que pelo menos um quantil. Se isto não for possível, vamos tentar obter soluções de compromisso, **como em qualquer problema de otimização multiobjetivo**. Por questões computacionais de caráter prático, em vez de todos os quantis, vamos considerar como referência na otimização m quantis - a serem previamente escolhidos.

A equação (5.11) apresenta a formulação multiobjetivo do problema não-linear restrito à pré-imagem de uma bola de raio ϵ em torno da meta de programação x^* no estágio final.

$$\min_{u[0],\ldots,u[N-1]} (Q_1[J], \ldots, Q_m[J]). \tag{5.11}$$

$$\text{sujeito a: } \begin{cases} ||(f(\ldots f(x[0],u[0],w[0]), \ldots, u[N-1],w[N-1]) - x^*)||_2 \leq \epsilon; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] \text{ pode ser dado.} \end{cases}$$

Sendo $Q_i[J]$, o quantil i da seguinte função estocástica:

$$J = \sum_{k=0}^{N-1} g_k(x[k],u[k]) + g_N(x[N]),$$

uma vez que cada estado $x[k]$ é uma variável aleatória.

Do ponto de vista computacional, como a otimização é feita em malha aberta, a resolução de cada problema (em cada estágio, como já comentado) pode ser feita via métodos de otimização multiobjetivo, com os descritos em Takahashi (2004).

Ao final do processo de otimização de cada estágio, basta que o responsável pela decisão escolha como resposta do problema, a seqüência de variáveis referentes ao quantil que melhor lhe convier. O fato é que, desta forma, o decisor tem maior liberdade de escolha, de acordo com as características específicas do problema em questão.

5.3 Estudos de caso

5.3.1 Problema de estoque

Apresentamos uma aplicação simples da metodologia proposta num estudo de caso clássico em se tratando de programação dinâmica estocástica: o problema

de estoque (Bertsekas, 1995)[2].

Um dos itens mais importantes do ativo de uma empresa comercial é o estoque (Slack et al., 2002). Essa importância advém não só de sua alta participação percentual no total do ativo, mas também do fato de ser a partir dele que se determina o custo das mercadorias ou produtos vendidos. Por um lado, devemos considerar que os produtos precisam ser estocados o menor tempo possível, mas que a demanda não atendida, ou um prazo de entrega não cumprido por falta do produto no estoque, pode causar sérios danos à imagem da empresa. Através de uma política de estoques eficiente, a companhia não fica à mercê da sorte, podendo controlar seus gastos e aumentar sua lucratividade.

O objetivo do nosso estudo é determinar a política de compras ótima de uma empresa, de modo a atender a uma demanda (estocástica) em tempos discretos, a um custo mínimo[3]. Além disso, queremos que ao final do horizonte de otimização, o estoque final da empresa seja próximo da média da demanda no último mês. Poderíamos ter considerado outros valores como meta para o estoque final, por exemplo: o estoque final nulo (supondo que ao final do processo será iniciado outro ciclo de negócios, com o lançamento de outro produto, por exemplo) ou que o valor do estoque final fosse o mesmo valor do estoque inicial (a fim de aplicar praticamente a mesma política a cada ano), etc. Consideramos, também, o estoque inicial nulo (a empresa acaba de ser inaugurada).

Modelagem Matemática

As variáveis deste problema dinâmico são: o número de produtos a serem considerados: n; o horizonte de otimização: N, em que cada estágio k corresponde a um mês do ano; cada estado $x[k]$ é um vetor de n coordenadas, cujo valor de cada coordenada i corresponde à quantidade do produto i em estoque ao final do estágio k; cada ação de controle $u[k]$ é um vetor de n coordenadas, cujo valor de cada coordenada i corresponde à quantidade do produto i que será comprado ao início do estágio k; cada distúrbio $w[k]$ é um vetor de n coordenadas, cujo valor de cada coordenada i corresponde à quantidade do produto i demandada durante o estágio k; cada coordenada do distúrbio é uma variável aleatória com a distribuição dada.

As variáveis deste problema são discretas, mas consideramos o caso de uma grande empresa, que compra em grandes quantidades. Neste caso, o número de possibilidades de compras é grande demais para ser prática a solução via métodos clássicos de programação dinâmica em tempo discreto. Desta forma, a fim de melhor usarmos a metodologia proposta, consideramos as variáveis, ou suas

[2]No exemplo 2.7, consideramos este problema no contexto do algoritmo clássico da programação dinâmica.

[3]Este estudo de caso foi submetido para publicação (Cardoso et al., 2008a).

relaxações, como números reais, ou seja, permitimos o arredondamento. Para este caso, métodos de otimização determinística em malha aberta apresentam solução assintoticamente exata.

O sistema dinâmico discreto no tempo referente ao controle de estoque pode ser dado pela equação (5.12).

$$x[k+1] = x[k] + u[k] - w[k]. \tag{5.12}$$

Devido ao uso da abordagem determinística em malha aberta, o nível de estoque em cada estágio $x[k]$ na função-objetivo e em cada restrição deve ser reescrito em função apenas do estado inicial, da seqüência de variáveis de controles variáveis e da seqüência de distúrbios (pré-fixados) anteriores. A equação (5.13) apresenta o estado reescrito.

$$x[k] = x[0] + \sum_{i=0}^{k-1} u[k-(i+1)] + \sum_{i=0}^{k-1} w[k-(i+1)]. \tag{5.13}$$

A meta de programação deste problema x^* e o estado inicial x_0 são dados. As demais restrições deste problema são quea quantidade em estoque e a quantidade a ser comprada devem ser não-negativas em cada estágio do problema.

Otimização

O problema de otimização restrita à meta de programação pode ser formulado como em (5.14).

$$\min_{u[0],\ldots,u[N-1]} Q_i \left[\sum_{k=0}^{N-1} g_k\big(x[k],u[k]\big) + g_N\big(x[N]\big) \right] \tag{5.14}$$

$$\text{sujeito a: } \begin{cases} e_i^T(x[0] + u[N-1] + \sum_{k=1}^{N-1} u[N-(k+1)] + \\ + \quad \sum_{i=0}^{N-1} w[N-(i+1)] - (x^* + e_i\epsilon)) \leq 0, \text{ para cada } i; \\ -e_i^T(x[0] + u[N-1] + \sum_{k=1}^{N-1} u[N-(k+1)] + \\ + \quad \sum_{i=0}^{N-1} w[N-(i+1)] - (x^* - e_i\epsilon)) \geq 0, \text{ para cada } i; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] + \sum_{i=0}^{k-1} u[k-(i+1)] + \sum_{i=0}^{k-1} w[k-(i+1)] \geq 0; \\ x[0] = x_0 \text{ dado;} \\ x[0] + \sum_{i=0}^{k-1} u[k-(i+1)] + \sum_{i=0}^{k-1} w[k-(i+1)] \geq 0; \\ u[k] \geq 0, \\ k = 0,\ldots,N-1. \end{cases}$$

Em que: $Q_i[\cdot]$ corresponde ao quantil i da variável aleatória em questão. Também na função-objetivo, o estado de cada estágio deve ser reescrito como mostrado na equação (5.13).

Consideramos o custo de cada estágio $g_k\big(x[k],u[k]\big)$ como sendo composto por um custo de compras linear, representado com um vetor linha d_k por unidade comprada, somada com um custo fixo D_k quando $u[k] \neq 0$; e um custo de penalidade por estoque não-nulo, representado po um vetor linha c_k, somado com uma penalidade por estoque negativo (interpretado como custo por não atender à demanda), representado por um vetor linha b_k.

Apresentamos simulações para dois casos:

- **Caso I:** demanda com distribuição de probabilidade assimétrica;

- **Caso II:** demanda com distribuição de probabilidade bimodal.

Note que quando a distribuição da demanda for assimétrica ou multi-modal, sua solução analítica costuma ser bem mais difícil, enquanto que nas nossas simulações este fato é indiferente, desde se tenha um algoritmo que gere dados com esta distribuição.

O horizonte de otimização é $N = 12$, as constantes da função-objetivo são: $D_k = 100$, $d_k = [20; 10; 10; 10; 10; 15; 15; 15; 10; 10; 10; 20]$, $c_k = 1$, $b_k = 40$. Procuramos cada coordenada das variáveis de decisão entre 0 e 100 no caso I e entre 0 e 1000 no caso II. A meta de programação deste problema x^* é a média da demanda (dada). O estado inicial x_0 é o vetor nulo.

Usamos o comando `randn` do Matlab para gerar dados seguindo uma distribuição normal. Em todas as simulações, o programa gerou 20.000 seqüências de distúrbios aleatórios $w[k]$, entretanto, em alguns casos, a convergência foi atingida com apenas 1.000 simulações (convergência considerando a mediana da função-objetivo). Cada problema determinístico foi resolvido através de um algoritmo genético, implementado pela função `ga` do Matlab. Em média, o tempo gasto em cada simulação foi de 16.226 segundos (quase 5 horas) numa máquina padrão.

Para o caso I, consideramos um pequeno supermercado, cuja média da demanda seja pequena, ou seja, se considerássemos a demanda deste produto com distribuição normal, teríamos na simulação demandas negativas, fato que não ocorre na prática. Escolhemos a distribuição lognormal com média 2 e desvio padrão 1 truncada no zero.

A figura 5.1 mostra a quantidade de produtos a ser comprada considerando os quantis Q_i com $i = 0{,}1$; $0{,}5$ e $0{,}9$ da função-objetivo. A figura 5.2 mostra o boxplot[4] do nível de estoque ao comprar segundo a política referente aos quantis 0,1, 0,5 (a mediana) e 0,9 da função-objetivo simulada.

[4]O retângulo no boxplot é traçado de tal maneira que suas bases têm alturas correspondentes

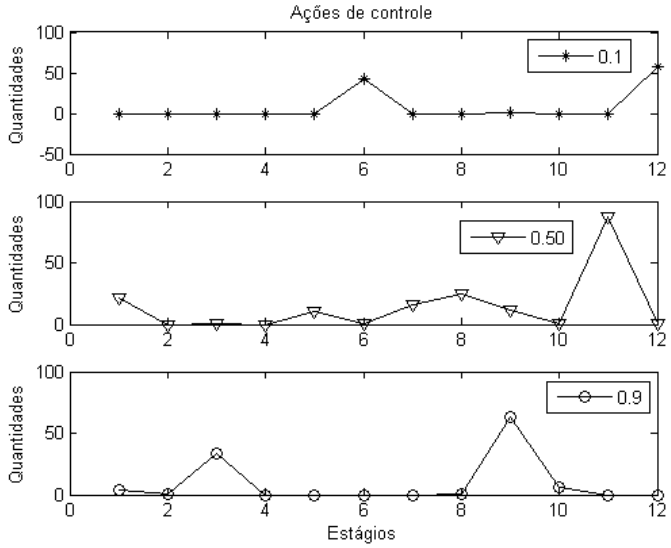

Figura 5.1: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição lognormal (caso I). Ações de controle (seqüência da quantidade de produtos a serem comprados) para os quantis 0,1, 0,5 e 0,9 da função-objetivo.

A figura 5.3 apresenta a Fronteira de Pareto deste problema, considerando como funções de decisão a probabilidade de sobrar estoque (a compra foi maior que a demanda) versus a probabilidade de faltar estoque (a demanda foi maior) ao final do processo.

Pela análise multi-quantil, um decisor poderia escolher um montante de compras mais elevado do que o quantil referente à mediana, e neste caso, a probabilidade de sobrar produtos (nível de estoque positivo) é maior do que a probabilidade de faltar estoque. E assim, um decisor mais pessimista pode escolher uma encomenda a partir de um quantil inferior à média, quando a probabilidade de não ter estoque (faltar mercadoria) é superior à probabilidade de se ter estoque.

Para o caso II, consideramos que o produto a ser modelado é o sorvete numa cidade tropical, com uma média de vendas em dias de frio e outra bem maior

aos primeiro e terceiro quartis da distribuição (Q_1 e Q_3, respectivamente). O retângulo é cortado por um segmento paralelo às bases, na altura correspondente ao segundo quartil. Assim, o retângulo do boxplot correponde aos 50% valores centrais da distribuição. Um segmento paralelo ao eixo parte do ponto médio da base superior do retângulo até o maior valor observado que não supera o valor de $Q_3 + (1{,}5)DEQ$, sendo DEQ a distância entre quartis. O mesmo é feito a partir do ponto médio da base inferior do retângulo, até o menor valor que não é menor do que $Q_1 - (1{,}5)DEQ$. As observações que estiverem acima de $Q_3 + (1{,}5)DEQ$ ou abaixo de $Q_1 - (1{,}5)DEQ$ são chamadas pontos exteriores e representadas por cruzes. Estas observações destoantes das demais são o que chamamos de *outliers* ou valores atípicos (Triola, 2005).

em dias de calor. Desta forma, a demanda por este produto é modelada com distribuição bimodal. Suponhamos que a demanda nos dias de frio siga uma distribuição normal com média 100 e desvio padrão 10 com uma probabilidade de 0,3 (probabilidade de fazer frio) e que nos dias de calor a demanda siga uma distribuição normal com média 200 e desvio padrão 10 com uma probabilidade de 0,7 (probabilidade de fazer calor).

A figura 5.4 mostra a quantidade de produtos a ser comprada considerando os quantis Q_i com $i = 0{,}1$; 0,5 e 0.9 da função-objetivo. A figura 5.5 mostra o boxplot do nível de estoque ao comprar segundo a política referente aos quantis 0,1, 0,5 (a mediana) e 0,9 da função-objetivo simulada.

Simulamos, apenas para o caso II, o processo de otimização em malha aberta com realimentação: para cada estágio, assim que o nível de estoque se tornar conhecido, o processo de otimização em malha aberta é executado, obtendo uma seqüência de ações de controle (uma política de compras), aplicando apenas a decisão atual.

As figuras 5.6 e 5.7 mostram a seqüência de ações de controle e os níveis de estoque simulados a cada mês, para as compras segundo os quantis 0,1 e 0,9, respectivamente, da função-objetivo.

Segundo Bertsekas (2005), a política ótima em malha aberta é uma cota superior em relação ao custo da seqüência que seria encontrada pelos algoritmos clássicos de programação dinâmica estocástica. Entretanto, um algoritmo clássico teria muita dificuldade para resolver estes dois problemas num tempo razoável, enquanto que cada simulação gastou poucas horas, um tempo razoavelmente menor que a duração de cada estágio (um mês).

Abordagem multiobjetivo

Consideramos o caso I (demanda com distribuição assimétrica). Procurando representar toda a extensão, escolhemos os seguintes quantis: $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, $Q_{0,75}$ e $Q_{0,90}$ da função-objetivo estocástica, sendo Q_i o quantil i.

A equação (5.15) apresenta a formulação multiobjetivo deste problema.

$$\min_{u[0],\ldots,u[N-1]} \left(Q_{0,10}\left[J\right], Q_{0,25}\left[J\right], Q_{0,50}\left[J\right], Q_{0,75}\left[J\right], Q_{0,90}\left[J\right]\right). \tag{5.15}$$

$$\text{sujeito a: } \begin{cases} x[0] + \sum_{i=0}^{N-1} u[N-(i+1)] + \sum_{i=0}^{N-1} w[N-(i+1)] = x^*; \\ \text{distribuição de cada } w[k] \text{ dada;} \\ x[0] + \sum_{i=0}^{k-1} u[k-(i+1)] + \sum_{i=0}^{k-1} w[k-(i+1)] \geq 0; \\ x[0] = x_0 \text{ dado;} \\ x[k] \geq 0; \\ u[k] \geq 0, \\ k = 0, \ldots, N-1. \end{cases}$$

Quantis	0,1	0,9
Probabilidade de faltar	0,74	0,01
Probabilidade de sobrar	0,26	0,99

Tabela 5.1: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição lognormal (caso I). Mapeamento da probabilidade de sobrar e de faltar estoque ao final do processo de otimização, considerando as políticas ótimas referentes aos quantis 0,1 e 0,9.

Neste caso, a função J é a mesma considerada no caso mono-objetivo.

Resolvemos este problema por meio do NSGA-II (Deb et al., 2002), com codificação real, seleção por torneio binário e mutação polinomial. Usamos os seguintes parâmetros no algoritmo: 200 indivíduos, critério de parada pelo número máximo de 200 gerações. A taxa de combinação é 0,70 e a taxa de mutação é 0,05.

A fim de analisarmos os resultados obtidos, ordenamos os 200 indivíduos Pareto-ótimos obtidos ao final do processo de otimização em ordem crescente dos valores da primeira função ($Q_{0,10}$). A figura 5.8 apresenta cada um dos indivíduos (no eixo horizontal) e os respectivos valores de suas funções $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, $Q_{0,75}$ e $Q_{0,90}$ (no eixo vertical) em ordem crescente de baixo para cima. Cada um dos indivíduos corresponde a uma política Pareto-ótima.

Note que, pelo menos aparentemente, o primeiro indivíduo minimiza os três primeiros quantis (0,10; 0,25 e 0,50), enquanto que o último indivíduo minimiza os dois últimos quantis (0,75 e 0,90). Desta forma, vamos concentrar nossa análise nestes dois indivíduos, que chamaremos, respectivamente de A e B. O indivíduo A aparenta ser uma boa aproximação da resposta que seria obtida se considerássemos o valor esperado da função-objetivo como critério de otimização, por conter o quantil referente à mediana. Por outro lado, o indivíduo B apresenta solução mais robusta, com menor variabilidade.

Para a etapa de decisão, consideramos como critério a probabilidade de não atingir a demanda ao final do processo (estoque negativo) e a probabilidade de sobrar estoque ao final do processo (estoque positivo). A tabela 5.1 apresenta o mapeamento da probabilidade de sobrar e de faltar estoque ao final do processo de otimização, para estes dois indivíduos.

Note que, pelo menos em princípio, a política referente ao indivíduo A parece atender melhor a um decisor mais otimista, pois apresenta o menor custo, mas também a maior probabilidade de faltar estoque. De fato, de acordo com a figura 5.8, a decisão A pode levar ao menor custo total possível (menos de 20), se tudo ocorrer como o decisor tinha previsto, mas pode levar ao maior custo possível (mais de 180), se as coisas lhe fugirem ao controle, tendo que pagar,

provavelmente por não atender à demanda, um alto custo de oportunidade. Note que esta solução é a que apresenta maior variabilidade, ou seja, maior risco. De acordo com estas simulações, o indivíduo B atende melhor a um decisor mais pessimista, com menor probabilidade de sobrar estoque (praticamente 100%) ao final do processo. Neste caso, seus custos estariam provavelmente entre 80 e 120, solução que dissemos ser mais robusta, ou com menor variabilidade. Isto quer dizer que, escolhendo esta política, o decisor correria um risco menor, no sentido de que seus custos não passariam de 120, apesar de que também não seriam menores que 80. A esperança deste decisor é que a demanda supere todas as suas expectativas, de modo que o alto custo seja recompensado pela alta receita correspondente às vendas.

Suponha que o decisor considere o risco como critério de decisão. Desta forma, para cada valor máximo que ele estiver disposto a gastar, a figura 5.8 apresenta a política Pareto-ótima mais adequada. Por exemplo, se o decisor puder gastar no máximo 140, a política ótima para ele seria a referente ao indivíduo 100, e no melhor caso, ele gastaria apenas 50. Note que, da forma como ordenamos os resultados, a política que minimiza o risco é a oposta à política que minimiza a mediana (que tomamos como aproximação do valor esperado).

Por fim, assinalamos que nesta simulação, apresentamos apenas os primeiros resultados, os obtidos literalmente em malha aberta. Isto quer dizer que, como temos proposto, esta análise deve ser refeita a cada estágio do problema, a fim de obter cada ação de controle a ser implementada. De toda forma, tal análise permite ao decisor, por meio de simulações, uma maior quantidade de informação para que ele possa fazer uma decisão mais confiante.

5.4 Conclusões do capítulo

Neste capítulo, apresentamos uma abordagem multi-quantil para o problema de programação dinâmica estocástica: em vez de considerar apenas o valor esperado da função-objetivo como referência no critério de otimização, propomos considerar as variáveis de decisão referentes a outros e/ou diversos quantis da função-objetivo estocástica do problema, pensadas enquanto variáveis aleatórias. Cada problema é resolvido de maneira sub-ótima, em malha aberta, via simulações de Monte Carlo, que permite considerar variáveis aleatórias com diferentes funções de distribuição.

O estudo de caso deste capítulo é clássico nesta classe de problemas: o controle de estoques de uma empresa. Apresentamos os resultados das simulações para os casos de demandas com distribuições assimétricas e bimodais. Apresentamos o resultado da otimização considerando diversos quantis da função-objetivo separadamente e simultaneamente (como num problema multiobjetivo), este último,

baseado no conceito de dominância estocástica.

Os resultados obtidos foram satisfatórios, pois permitem que o decisor tenha a oportunidade de escolher a política ótima de acordo com seus interesses, variando-se o critério de otimização como melhor lhe convier no momento. Acreditamos que esta abordagem tenha ampla aplicabilidade em problemas com distribuições arbitrárias, bem como em problemas em tempos discretos com dinâmicas não-lineares.

Como estudos futuros, propomos a aplicação desta metodologia em problemas estocásticos não-lineares relevantes, como o problema da vacinação ótima para controle de epidemias, usando equações diferenciais estocásticas (Braumann, 1983) e modelos discretos, baseados em indivíduos (Nepomuceno, 2005).

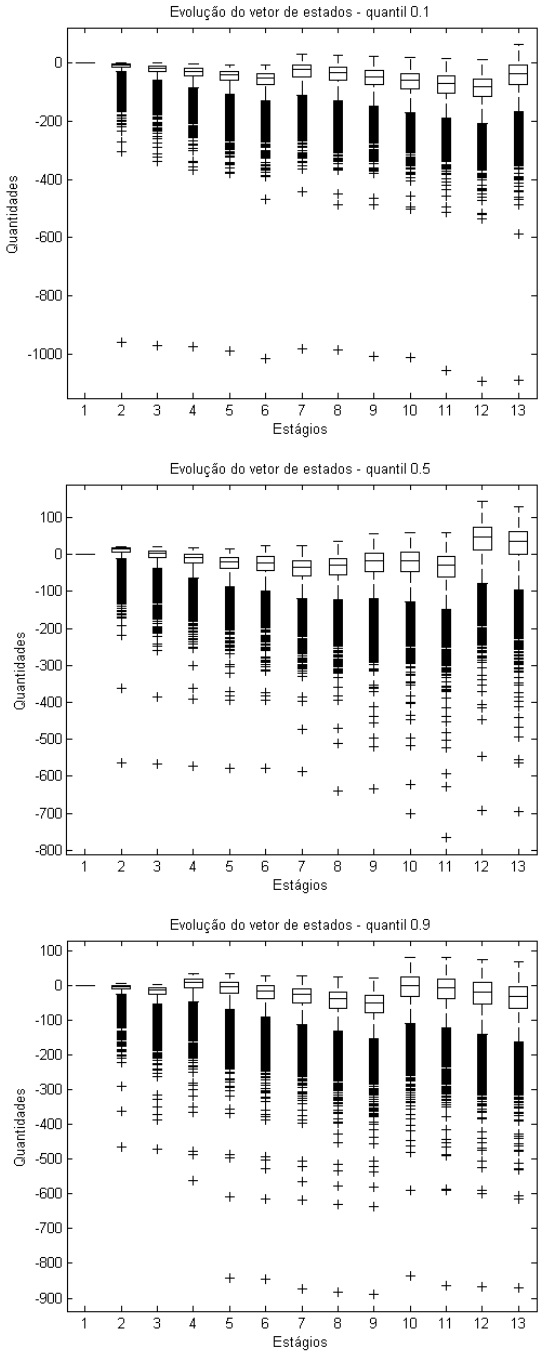

Figura 5.2: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição lognormal (caso I). Boxplot da seqüência do vetor de estados (nível de estoque) com quantidade comprada referente aos quantis 0,1, 0,5 e 0,9 (respectivamente) da função-objetivo.

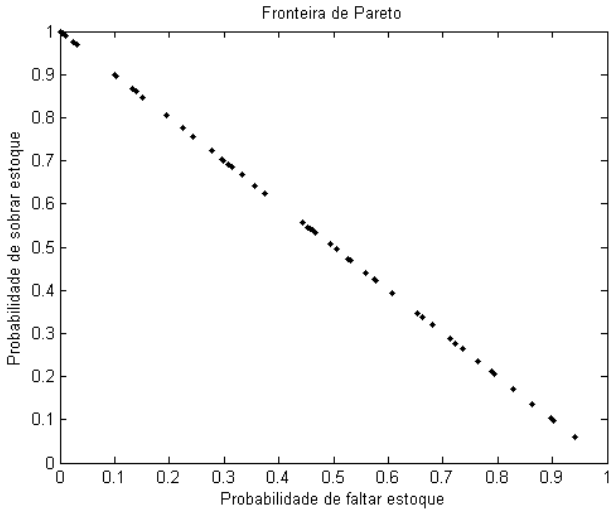

Figura 5.3: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição lognormal (caso I). Fronteira de Pareto considerando como eixos a probabilidade de sobrar e de faltar estoque ao final do processo de otimização.

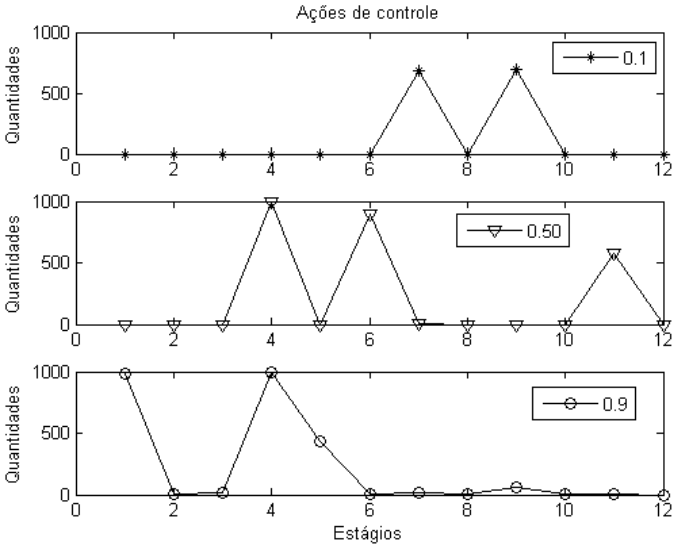

Figura 5.4: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição bimodal (caso II). Ações de controle (seqüência da quantidade de produtos a serem comprados) para os quantis 0,1, 0,5 e 0,9 da função-objetivo.

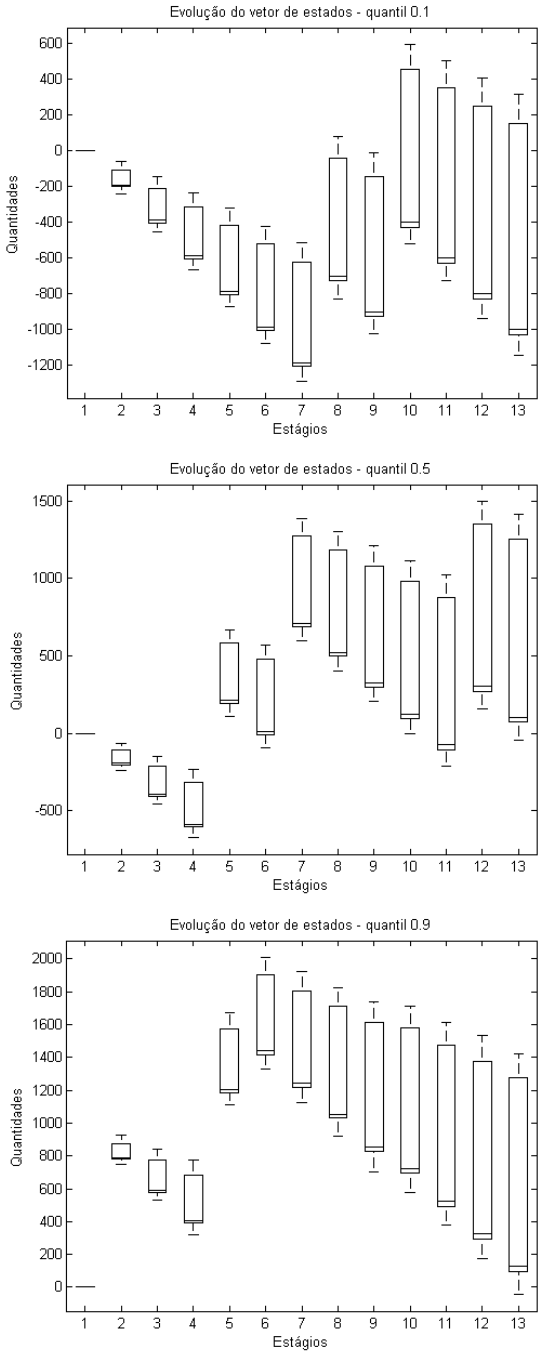

Figura 5.5: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição bimodal (caso II). Boxplot da seqüência do vetor de estados (nível de estoque) com quantidade comprada referente aos quantis 0,1, 0,5 e 0,9 (respectivamente) da função-objetivo.

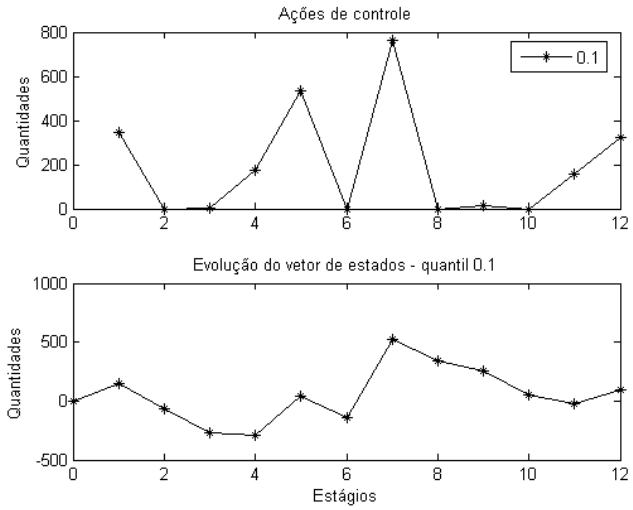

Figura 5.6: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição bimodal (caso II). Seqüência da quantidade a ser comprada segundo o quantil 0,1 da função-objetivo e referente nível de estoque simulado no processo de otimização em malha aberta com realimentação.

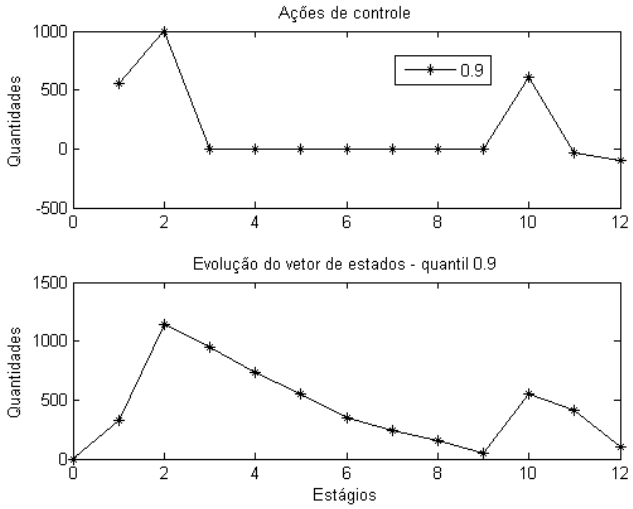

Figura 5.7: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição bimodal (caso II). Seqüência da quantidade comprada segundo o quantil 0,9 da função-objetivo e referente nível de estoque simulado no processo de otimização em malha aberta com realimentação.

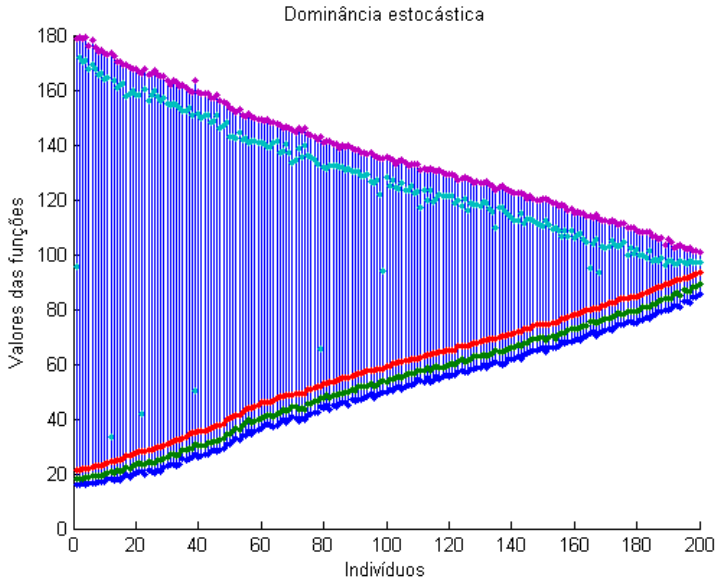

Figura 5.8: Estudo de caso: controle de estoque (seção 5.3.1) - demanda com distribuição lognormal (caso I). Amostra das soluções Pareto-ótimas via dominância estocástica. Cada indivíduo, correspondente a uma política Pareto-ótima, está marcado no eixo horizontal em ordem crescente em relação aos respectivos valores de $Q_{0,10}$. Os respectivos valores de suas funções $Q_{0,10}$, $Q_{0,25}$, $Q_{0,50}$, $Q_{0,75}$ e $Q_{0,90}$ estão no eixo vertical, em ordem crescente (em relação aos quantis) de baixo para cima.

Capítulo 6

Conclusões e trabalhos futuros

Procuramos relatar nesta tese o que de mais relevante fizemos durante o doutoramento. O tema por nós escolhido é abrangente: nos dispusemos a propor ferramentas computacionalmente tratáveis para problemas de programação dinâmica com ações de controle discretas, nos casos: linear, não-linear, determinístico, estocástico, mono e multiobjetivo.

As ferramentas foram desenvolvidas sob um mesmo paradigma: permitir que a otimização seja executada em malha aberta, considerando a restrição dinâmica por meio da pré-imagem no estágio inicial de um conjunto fechado em torno da meta de programação do problema. Para cada classe de problemas estudada, apresentamos estudos de casos mostrando a aplicação dos algoritmos na solução de problemas reais relevantes.

6.1 Contribuições do trabalho

6.1.1 Estado da arte

Inicialmente, esta tese apresenta uma revisão bibliográfica sobre métodos numéricos para problemas de programação dinâmica. Expusemos a teoria e discutimos a solução de alguns problemas discretos e contínuos, além de falarmos sobre os casos impulsivo, multiobjetivo e sub-ótimo, apresentando as principais referências em cada caso.

Podemos sintetizar o estado da arte desta forma:

- Existe um algoritmo que encontra a solução ótima em malha fechada para os problemas de programação dinâmica discreta e determinística, baseado no Princípio da Otimalidade de Bellman. Se as variáveis do problema pertencerem a um conjunto discreto e finito, este algoritmo consiste em montar a árvore de possibilidades e procurar nela um caminho ótimo, em

etapas seqüenciais. No caso de variáveis contínuas, este algoritmo pode ser usado via discretização.

- O problema é que este algoritmo sofre da chamada "maldição da dimensionalidade", ou seja, tem um custo computacional exponencial em função do número de variáveis de estado e do número de estágios do problema. Em vista disto, o desenvolvimento de métodos computacionalmente tratáveis, ainda que sub-ótimos, se faz necessário. Os métodos sub-ótimos se classificam pela forma com que a aproximação é feita: *on-line* (como o controle por realimentação em malha aberta e o controle preditivo) ou *off-line* (como a programação neuro-dinâmica).

- Recentemente, diversos problemas de otimização dinâmica têm sido modelados via controle impulsivo, e sua resolução via programação dinâmica geralmente tem usado métodos variacionais.

- Poucos problemas têm sido modelados como programação dinâmica multiobjetivo, devido à alta complexidade computacional da sua solução. E neste caso, apenas métodos sub-ótimos têm sido usados, como os algoritmos genéticos e a programação neuro-dinâmica. Estes métodos encontram uma aproximação da fronteira de Pareto em questão.

A literatura sobre programação dinâmica ainda está fragmentada, dispersa em várias áreas do conhecimento, principalmente na pesquisa operacional, na teoria de controle e na área de risco. O livro mais completo da área (Bertsekas, 1995), que é uma edição revisada do anterior (Bertsekas, 1985), por exemplo, só na edição lançada em 2007[1] trata sobre controle preditivo, programação dinâmica multiobjetivo, dentre outros tópicos tratados nesta tese.

6.1.2 Ferramentas propostas

Neste texto, propusemos ferramentas, ou algoritmos numéricos, para certas classes de problemas de programação dinâmica, concebidas sob um ponto de vista geométrico. O fio condutor que une as ferramentas propostas e as diferencia das demais é a seguinte:

- Propomos considerar a dinâmica do sistema via iteração de conjuntos fechados pelo sistema dinâmico (isto é o que chamamos de ponto de vista geométrico), através da pré-imagem dos estados em um dos estágios do problema. Tal iteração é possível para um número arbitrário de estágios.

[1]Infelizmente, ainda não tivemos acesso a esta edição.

- Fazemos a resolução em malha aberta, implementando apenas a decisão atual e executando nova otimização a cada estágio, assim que o valor do estado for conhecido, se for o caso.

- Procuramos analisar cada problema do ponto de vista multiobjetivo.

Devido à abordagem em malha aberta, e ao fato de considerarmos a restrição de ponto-final, nossos métodos se assemelham ao controle preditivo, um recente método de aproximação implícita, descrito na seção 2.5.

As ferramentas foram aplicadas em três classes de problemas dinâmicos: determinísticos lineares, determinísticos não-lineares e estocásticos lineares. Cada uma destas foi estudada num capítulo desta tese, em que discutimos as especificidades de cada classe.

Problemas determinísticos lineares

Quando o sistema dinâmico é linear, as operações com matrizes nos ajudam a encontrar uma expressão para a pré-imagem no estágio inicial de coAs ferramentas foram aplicadas em três classes de problemas dinâmicos: determinísticos lineares, determinísticos não-lineares eema. E a Álgebra Linear nos garante que estes conjuntos pertencem a uma mesma família parametricamente invariante pelo sistema dinâmico, no sentido de que continuam, em cada iteração, politopos ou elipsóides, e suas degenerações.

Conhecer as características paramétricas dos conjuntos iterados pelo sistema é uma vantagem que ajuda na escolha o método de otimização em malha aberta a ser usado. Por exemplo, considerando um conjunto politópico em torno da meta, e se as demais restrições do problema forem lineares, podemos realizar a otimização em malha aberta usando métodos de programação linear (mono ou multiobjetivo),executando a otimização a um custo computacional eficiente, em comparação com problemas não-lineares de mesma dimensão.

Além disso, a possibilidade de fazer uma relaxação na meta de tamanho admissível pode ser útil a fim de encontrar uma solução aproximada com valor de função objetivo consideravelmente melhor.

Problemas determinísticos não-lineares

Apresentamos a extensão da metodologia proposta para o caso não-linear, contando, computacionalmente falando, com a recursividade de funções para trabalhar com a equação da pré-imagem de um conjunto fechado em torno da meta. Principalmente em problemas não-lineares, trabalhar com um conjunto em torno da meta pode ser vantajoso, pela dificuldade de se trabalhar com restrições de igualdade. Uma vez que não temos muito conhecimento sobre a pré-imagem de

conjuntos por funções não-lineares, a otimização em malha aberta foi feita usando métodos estocásticos.

Nesta classe, demos atenção aos problemas impulsivos, uma classe relativamente nova, que modela situações relevantes do mundo real. Escolhemos resolver estes problemas via técnicas desenvolvidas para a programação dinâmica discreta no tempo, com a vantagem de podermos usar nossa metodologia, em vez de resolvê-los como um problema contínuo, como geralmente tem sido feito.

Nos problemas que estudamos, vimos que a abordagem proposta é mais realística que as técnicas de controle em tempo contínuo, tem custo menor que as técnicas impulsivas por pulsos constantes, além de ser muito mais barata (do ponto de vista computacional) que a técnica ótima de programação dinâmica em tempo discreto.

Problemas estocásticos lineares

No caso estocástico, propusemos uma abordagem multi-quantil: permitir a otimização usando outros e/ou diversos quantis da função-objetivo em questão, em vez de usar apenas o valor esperado como referência no critério de otimização. A fim de considerarmos distúrbios aleatórios com diferentes distribuições, usamos simulações de Monte Carlo para computar e ordenar os valores dos quantis.

Para cada distúrbio fixo, resolvemos o problema em malha aberta por meio das abordagens propostas para o caso determinístico. No caso em que a dinâmica do sistema é linear, a equação da pré-imagem de um conjunto pelo sistema, para cada distúrbio fixo, é facilmente obtida usando álgebra matricial, e no caso não-linear, computacionalmente falando, podemos usar de recursividade.

Através da abordagem multi-quantil, num problema mono ou multiobjetivo, o decisor tem mais possibilidades de escolha, o que geralmente costuma ser bem-vindo. Além disso, reforçamos que pouquíssimos problemas de programação dinâmica estocástica são modelados da forma multiobjetivo, pelo seu alto custo computacional. Nosso estudo de caso mostra que, ao ser resolvido pela nossa abordagem, esta modelagem pode ser interessante e útil, ainda que o resultado encontrado seja sub-ótimo.

6.1.3 Estudos de casos

Apresentamos, nesta tese, cinco estudos de casos relevantes para as três classes de problemas consideradas, e obtivemos resultados interessantes em cada aplicação.

Problemas determinísticos lineares

Para este caso, apresentamos dois estudos de casos: uma aplicação da abordagem multiobjetivo na otimização dos recursos obtidos com a compra e venda de gado, estendendo o estudo de caso apresentado no trabalho de mestrado, e a otimização do projeto de distribuição de redes de energia elétrica.

O primeiro estudo apresentado é a otimização biobjetivo dos recursos obtidos com a montagem de um rebanho de 1000 cabeças de gado ao longo de 7 anos, modelado como um problema determinístico linear. Vimos que quanto maior for o número de cabeças de gado do problema, maior seria o número de possibilidades a serem listadas pelos métodos enumerativos tradicionais, o que dificultaria encontrar a solução ótima. Por outro lado, quanto maior for o número de cabeças de gado do problema, assintoticamente menor será o erro relativo do resultado obtido pela relaxação das variáveis aqui proposta. Devido à simplicidade do modelo, o custo computacional da solução é irrisório (poucos segundos), e os resultados obtidos se mostraram viáveis do ponto de vista prático.

No segundo estudo de caso, de grande relevância dentro da Engenharia Elétrica, estudamos a otimização da expansão de uma rede de distribuição de energia elétrica com 100 nodos em 10 anos. A abordagem proposta é bastante eficiente e não-convencional, pois modelamos a expansão do sistema como um problema dinâmico linear, apenas com a função-objetivo e as demais restrições sendo não-lineares. Propomos operadores para algoritmos genéticos especialmente desenvolvidos para este fim, que melhoraram consideravelmente a eficiência dos algoritmos. Os resultados obtidos se mostraram melhores que os possíveis de serem obtidos com as técnicas tradicionais de projeto. O uso da otimização neste problema permite uma maior flexibilização de políticas de preços das concessionárias de energia, proporcionando uma maior competitividade em mercados abertos, ou uma maior margem de lucro em mercados fechados.

Problemas determinísticos não-lineares

No caso não-linear, apresentamos dois estudos de casos relevantes, resolvidos via programação dinâmica impulsiva: estudamos a aplicação da metodologia proposta no controle biológico de pragas numa lavoura e na busca de estratégias ótimas de vacinação num modelo de epidemias. Estudamos os casos mono e multiobjetivo.

O controle biológico de pragas em lavouras propõe estratégias que viabilizam (a um custo mínimo) a produção de alimentos com qualidade, sem o uso de pesticidas, ou seja, sem agredir a natureza. Estudamos o caso da soja, de grande importância na economia do Brasil, o maior exportador do complexo (grãos, farelo e óleo) e o segundo produtor mundial. A solução ótima por nós encontrada

é mais fácil de ser implementada nas lavouras, do ponto de vista prático, do que estratégias que vêm sendo propostas, usando ações de controle contínuas no tempo. A abordagem multiobjetivo nos permitiu encontrar soluções de compromisso entre o custo com a inserção de predadores e o prejuízo com as pragas ao longo dos estágios do problema.

O controle de epidemias busca proporcionar estratégias mais baratas e efetivas para o controle de infecções. Em 2004, completou-se um século da primeira campanha de vacinação em massa feita no Brasil, idealizada por Oswaldo Cruz, que tinha por objetivo controlar a varíola, que então dizimava boa parte da população do Rio de Janeiro. Ainda hoje, diante da escassez de recursos públicos e das dificuldades logísticas, tratar a prevenção de doenças previsíveis por meio de vacinações de forma eficiente é de fundamental importância para o país. A política ótima encontrada neste trabalho se mostrou melhor, do ponto de vista da redução de custos, que a estratégia de vacinação por pulsos constantes. A abordagem multiobjetivo nos permitiu encontrar soluções de compromisso entre a minimização do prejuízo com os infectados e o custo de vacinação ao longo dos estágios do problema.

Concluímos que, de fato, a abordagem via programação dinâmica impulsiva é mais realística (do ponto de vista prático) que as técnicas via controle em tempo contínuo e pode ser mais barata (do ponto de vista da função-objetivo) que as técnicas impulsivas por pulsos constantes. Esta técnica se apresenta também como boa alternativa à discretização do problema contínuo e à solução via métodos enumerativos.

Problemas estocásticos lineares

Consideramos o controle de estoques de uma empresa como estudo de caso, apresentando os resultados das simulações para os casos de demandas com distribuições assimétricas e bimodais, para diversos quantis da função-objetivo. Este estudo de caso é clássico nesta classe de problemas, e a abordagem proposta parece ser original.

Os resultados obtidos com a otimização multiobjetivo, de certa forma, quebram um paradigma, pois permitem ao decisor a oportunidade de planejar sua política de compras de acordo com sua visão de mercado, variando-se o critério de otimização como melhor lhe convier no momento.

Sabemos que boa administração da política de compras é de vital importância para a saúde financeira de uma empresa, uma vez que grande parte do seu capital está nos materiais envolvidos na produção. Assim, reduções no montante estocado, bem como reduções nas perdas de vendas devido à falta de produtos em estoque, se traduzem em margens maiores de lucros diretos, necessários ao bom andamento do negócio como um todo.

6.1.4 Discussão final

Por fim, destacamos as seguintes vantagens das ferramentas propostas:

- Evita a montagem da árvore de possibilidades, que tem alto custo computacional.
- A opção pela solução em malha aberta nos permite resolver o problema via métodos de otimização estática, e assim considerar diferentes tipos de funções-objetivo e restrições.
- O número total de variáveis é menor, pois consideramos as variáveis de estado em apenas um dos estágios do problema.
- Podemos facilmente considerar uma relaxação na meta de programação do problema. Este artifício permite aproximar o problema por outro com região factível maior, possivelmente acelerando a convergência dos métodos numéricos usados na otimização.
- Podemos facilmente fazer a extensão para o caso multiobjetivo.

Destacamos também as limitações das ferramentas propostas:

- Perde a característica de malha fechada. Assim, a fim de atualizar desvios causados por possíveis incorreções no modelo ou por causa da estocasticidade do processo, nova otimização deve ser feita em cada estágio.
- Desta forma, em geral, o procedimento é sub-ótimo.

Melhor dizendo, as ferramentas propostas nesta tese são:

- exatas, desde que usemos algoritmos exatos para a solução em malha aberta, nos problemas:
  - determinísticos com variáveis reais;
  - determinísticos com variáveis inteiras;
  - estocásticos, quando houver uma equivalência à certeza.
- assintoticamente exatas para os problemas com variáveis inteiras, quando permitida a devida relaxação, desde que usemos algoritmos exatos para a solução em malha aberta.
- sub-ótimas nos problemas:
  - estocásticos, quando não houver uma equivalência à certeza;

– em que a resolução em malha aberta utilizar métodos estocásticos.

As soluções não-exatas fornecem um limitante inferior para o valor do ótimo exato. Infelizmente, ainda não conseguimos fornecer uma cota para a distância máxima entre a solução encontrada e o ótimo exato.

A complexidade computacional de solução está diretamente associada ao método de otimização escolhido para resolver o problema em malha aberta. A primeira execução conta com pN variáveis, sendo que N é o número de estágios e p é o número de ações de controle em cada estágio, e quando a otimização deve ser refeita em cada estágio, o número total de variáveis é, no máximo, $pN(N-1)/2$.

No caso geral, vimos que a estratégia proposta, quando executada em cada estágio, se torna, de certa forma, em malha fechada, podendo adaptar eventuais modificações no sistema de variáveis com relação a distúrbios externos e/ou para incorreções do modelo. A execução do algoritmo uma vez por estágio do problema se justifica nestes casos, principalmente quando a escala de tempo da execução do algoritmo para solucionar o problema em malha aberta for suficientemente pequena em relação ao tempo entre a leitura do estado atual e a efetiva execução da próxima ação de controle.

Por fim, assinalamos que um algoritmo de programação dinâmica clássica teria dificuldade para resolver, num tempo razoável, cada um dos problemas considerados nesta tese. Enquanto isso, os algoritmos propostos e usados nos estudos de casos os resolveram em questões de horas ou minutos, dependendo do caso, por não terem um caráter enumerativo, fato que queríamos evitar. Conjecturamos que as propriedades das ferramentas propostas verificadas nos estudos de casos desta tese sejam extensivas as classes de problemas dinâmicos similares.

6.2 Trabalhos futuros

Ao final deste doutoramento, surge uma gama de possibilidades para trabalhos futuros.

- Na nossa próxima etapa, a ser tratada no pós-doutoramento, propomos o estudo de aplicações em que o sistema dinâmico seja modelado por equações estocásticas não-lineares. Por exemplo, queremos estudar o problema da vacinação ótima para controle de epidemias via modelos baseados em indivíduos (Nepomuceno, 2005), considerando a probabilidade de erradicação da doença como critério de otimização, num problema discreto tri-objetivo. Propomos também o uso de equações diferenciais estocásticas (Braumann, 1983), aplicando a técnica impulsiva.

- Além disso, gostaríamos de aprofundar o caso com horizonte infinito (Bertsekas, 2005), estudando melhor a idéia de propor uma estratégia ótima para

levar o estado de um regime transitório para um regime permanente, acoplada com uma estratégia ótima estacionária para o regime permanente, como fizemos nesta tese com o problema da vacinação ótima.

- Para o problema de expansão do sistema de distribuição de energia, gostaríamos de considerar as incertezas na evolução da carga (Carrano et al., 2007), a fim posicionar as novas estações considerando, num problema biobjetivo, o custo financeiro nominal e a taxa de factibilidade da solução encontrada.

- Como usamos algoritmos genéticos para a solução da maioria dos problemas nesta tese, proporíamos que seja estudado, inspirado em Carrano (2007), o desenvolvimento de operadores que melhorem o custo computacional de cada solução, levando em conta as especificidades da classe de problemas em questão.

- Poderíamos estudar o processo de tomada de decisão nos problemas dinâmicos multiobjetivos, como em Parreiras (2006), principalmente no caso estocástico, como apenas começamos a fazer.

- Poderíamos procurar estabelecer uma cota para o erro da solução subótima, tentando medir a distância da otimalidade, como feito por de Farias (2002). Poderíamos nos basear, por exemplo, na identificação e análise do problema dual e/ou no relaxamento de restrições.

- Outra linha de estudo seria a síntese de planos de fase, como em Takahashi et al. (1999). Neste caso, um dado plano de fases seria nossa meta de programação, em vez de apenas um ponto fixo, em que procuraríamos por regiões invariantes.

- Para cada classe de problemas, gostaríamos de estudar aplicações em outros problemas relevantes com modelagens semelhantes. Mais ainda, gostaríamos de verificar a aplicação da metodologia proposta no tratamento com dados reais. Temos feito contato com diversos órgãos, a fim de viabilizar esta proposta.

6.3 Produção bibliográfica

Apresentamos a produção bibliográfica durante o doutoramento.

Artigos publicados em periódicos

- Carrano, E. G., Cardoso, R. T. N., Takahashi, R. H. C., Fonseca, C. M., Neto, O. M., 2008. Power distribution network expansion scheduling using dynamic programming genetic algorithm. IET Generation, Transmission & Distribution. DOI `10.1049/iet-gtd:20070174`

Artigos submetidos para publicação em periódicos

- Cardoso, R. T. N., da Cruz, A. R., Wanner, E. F., Takahashi, R. H. C., 2008. Biological control of plagues in soy farmings using a multiobjective evolutionary impulsive dynamic optimization scheme.

- Cardoso, R. T. N., Takahashi, R. H. C., 2008. Solving impulsive control problems by discrete-time dynamic optimization methods.

- Cardoso, R. T. N., Wanner, E. F., Takahashi, R. H. C., 2008. An LMI Approach for Continuous-Variable Dynamic Programming Problems Based on Invariant Families.

- Cardoso, R. T. N., Cruz, F. R. B., Takahashi, R. H. C., 2008. Pareto Optimal Solutions for Stochastic Dynamic Programming Problems via Monte Carlo Simulation.

Artigos publicados em conferências

- Cardoso, R. T. N., Takahashi, R. H. C., 2007. Solving impulsive control problems by discrete-time dynamic optimization methods. Anais do XXX Congresso Nacional de Matemática Aplicada e Computacional.

- Cardoso, R. T. N., Takahashi, R. H. C., 2005. Algoritmos para programação dinâmica baseados em famílias invariantes. Anais do XXXVII Simpósio Brasileiro de Pesquisa Operacional.

- da Cruz, A. R., Cardoso, R. T. N., Takahashi, R. H. C., 2007. Uma abordagem multiobjetivo para o problema de controle biológico através da programação dinâmica não-linear via NSGA-II. Anais do XXXIX Simpósio Brasileiro de Pesquisa Operacional.

- da Cruz, A. R., Cardoso, R. T. N., Wanner, E. F., Takahashi, R. H. C., 2007. A multiobjective non-linear dynamic programming approach for optimal biological control in soy farming via NSGA-II. Proceedings of the 2007 IEEE Congress on Evolutionary Computation.

- da Cruz, A. R., Cardoso, R. T. N., Takahashi, R. H. C., 2006. Uma Abordagem Via Programação Dinâmica Não-Linear para o Problema do Controle Biológico de Pragas Numa Lavoura. Anais do XXXVIII Simpósio Brasileiro de Pesquisa Operacional.

Artigos submetidos para publicação em conferências

- Cardoso, R. T. N., Cruz, F. R. B., Takahashi, R. H. C., 2008. A multi-quantile approach for stochastic dynamic programming problems via monte carlo simulation.

- Cardoso, R. T. N., Wanner, E. F., Takahashi, R. H. C., 2008. An LMI approach for open-loop multiobjective continuous-variable dynamic programming problems.

- Alvarenga, L. R., Cardoso, R. T. N., Nepomuceno, E. G., Takahashi, R. H. C., 2008. Reduction in the Cost Computational Simulation of Epidemiological Systems by means Neural Networks.

- da Cruz, A. R., Cardoso, R. T. N., Takahashi, R. H. C., 2008. Uma Abordagem Multiobjetivo para o Planejamento de Campanhas de Vacinação via NSGA-II.

Resumos publicados em conferências e apresentações orais

- Cardoso, R. T. N., Cruz, F. R. B., Takahashi, R. H. C., 2007. Heurísticas para Programação Dinâmica Estocástica via Simulação de Monte Carlo. XII Escuela Latinoamericana de Verano de Investigación Operativa.

- Cardoso, R. T. N., Takahashi, R. H. C., 2006. An iterated invariant-set approach for linear multi-objective dynamic programming problems. Proceedings of the XIX International Symposium on Mathematical Programming.

Bibliografia

Abo-Sinna, M. A., 2004. Multiple objective (fuzzy) dynamic programming problems: a survey and some applications. Applied Mathematics and Computation 157 (3), 861–888.

Bazaraa, M. S., Sherali, H. D., Shetty, C. M., 1993. Nonlinear programming: Theory and algorithms, 2nd Edition. John Wiley & Sons, New York, NY.

Bellman, R., 1957. Dynamic Programming. Princeton University Press, Princeton, NJ.

Bertsekas, D. P., 1985. Dynamic Programming — Deterministic and Stochastic Models. Prentice-Hall, Upper Saddle River, NJ.

Bertsekas, D. P., 1995. Dynamic Programming and Optimal Control. Vol. 1 and 2. Athena Scientific, Nashua, NH.

Bertsekas, D. P., 2005. Dynamic programming and suboptimal control: a survey from ADP to MPC. European Journal of Control 11 (4-5).

Bertsekas, D. P., Rhodes, I. B., 1971. On the minimax reachability of target sets and target tubes. Automatica 7, 233–247.

Bertsekas, D. P., Tsitsiklis, J. N., 1996. Neuro-Dynamic Programming. Athena Scientific, Nashua, NH.

Boyd, S., El-Ghaoui, L., Feron, E., Balakrishnan, V., 1994. Linear Matrix Inequalities in System and Control Theory. Vol. 15. SIAM — Studies in Applied Mathematics, Philadelphia, PA.

Branicky, M. S., 1995. Studies in hybrid systems: Modeling, analysis, and control. Phd dissertation, Electrical Engineering and Computer Science Department, Massachusetts Institute of Technology, USA.

Braumann, C. A., 1983. Population growth in random environments. Bulletin of Mathematical Biology 45 (4), 635–641.

Campos-filho, F. F., 2001. Algoritmos Numéricos. Editora LTC, Rio de Janeiro.

Cardoso, R. T. N., 2005. Algoritmos para programação dinâmica baseados em famílias invariantes. Dissertação de mestrado, Programa de Pós-Graduação em Matemática, Universidade Federal de Minas Gerais.

Cardoso, R. T. N., Cruz, F. R. B., Takahashi, R. H. C., 2008a. A multi-quantile approach for stochastic dynamic programming problems via Monte Carlo simulation, (Manuscrito a ser publicado).
URL `http://www.cpdee.ufmg.br/\~{}rodrigoc/papers/artigoc1.pdf`

Cardoso, R. T. N., da Cruz, A. R., Wanner, E. F., Takahashi, R. H. C., 2008b. Biological control of plagues in soy farmings using a multiobjective evolutionary impulsive dynamic optimization scheme, (Manuscrito a ser publicado).
URL `http://www.cpdee.ufmg.br/\~{}rodrigoc/papers/artigop1.pdf`

Cardoso, R. T. N., Takahashi, R. H. C., 2005. Algoritmos para programação dinâmica baseados em famílias invariantes. Anais do XXXVII Simpósio Brasileiro de Pesquisa Operacional .

Cardoso, R. T. N., Takahashi, R. H. C., 2007. Solving impulsive control problems by discrete-time dynamic optimization methods. Anais do XXX Congresso Nacional de Matemática Aplicada e Computacional .

Cardoso, R. T. N., Takahashi, R. H. C., 2008. Solving impulsive control problems by discrete-time dynamic optimization methods, (Manuscrito a ser publicado).
URL `http://www.cpdee.ufmg.br/\~{}rodrigoc/papers/artigop2.pdf`

Cardoso, R. T. N., Wanner, E. F., Takahashi, R. H. C., 2008c. An LMI approach for open-loop multiobjective continuous-variable dynamic programming problems, (Manuscrito a ser publicado).
URL `http://www.cpdee.ufmg.br/\~{}rodrigoc/papers/artigoc2.pdf`

Carrano, E. G., 2007. Algoritmos evolucionários eficientes para otimização de redes. Tese de doutorado, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

Carrano, E. G., Cardoso, R. T. N., Takahashi, R. H. C., Fonseca, C. M., Neto, O. M., 2008. Power distribution network expansion scheduling using dynamic programming genetic algorithm. IET Generation, Transmission & Distribution 2 (3), 444–455.

Carrano, E. G., Guimaraes, F. G., Takahashi, R. H. C., Neto, O. M., Pinto, F. C., 2007. Electric distribution network expansion under load evolution uncertainty

using immune system inspired algorithms. IEEE Transactions on Power Systems 22 (2), 851–861.

Carrano, E. G., Soares, L. A. E., Takahashi, R. H. C., Saldanha, R. R., Neto, O. M., 2006. Electric distribution multiobjective network design usign a problem-specific genetic algorithm. IEEE Transactions on Power Delivery 21 (2), 995–1005.

Carrano, E. G., Takahashi, R. H. C., Cardoso, E. P., Saldanha, R. R., Neto, O. M., 2005. Optimal substation location and energy distribution network design using a hybrid GA-BFGS algorithm. IEE Proceedings on Generation, Transmission and Distribution 152 (6), 919–926.

Chankong, V., Peng, T. K. C., 1983. Multiobjective Decision Making — Theory and Methodology. Elsevier.

Chen, C., 1999. Linear System Theory and Design. Oxford University Press, Oxford.

Chen, V. C. P., Ruppert, D., Shoemaker, C. A., 1999. Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. Operations Research 47 (1), 38–52.

Chiang, A. C., 1992. Elements of Dynamic Optimization. Mc Graw Hill.

da Cruz, A. R., Cardoso, R. T. N., Takahashi, R. H. C., 2006. Uma abordagem via programação dinâmica não-linear para o problema do controle biológico de pragas numa lavoura. Anais do XXXVIII Simpósio Brasileiro de Pesquisa Operacional .

da Cruz, A. R., Cardoso, R. T. N., Takahashi, R. H. C., 2007a. Uma abordagem multiobjetivo para o problema de controle biológico através da programação dinâmica não-linear via NSGA-II. Anais do XXXIX Simpósio Brasileiro de Pesquisa Operacional .

da Cruz, A. R., Cardoso, R. T. N., Wanner, E. F., Takahashi, R. H. C., 2007b. A multiobjective non-linear dynamic programming approach for optimal biological control in soy farming via NSGA-II. Proceedings of the 2007 IEEE Congress on Evolutionary Computation .

Dano, S., 1975. Nonlinear and Dynamic Programming. Springer-Verlag.

Dar'in, A. N., Kurzhanskii, A. B., Selesznev, A. V., 2005. The dynamic programming method in impulsive control synthesis. Ordinary Differential Equations 41 (11), 1491–1500.

Davis, L., 1991. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, NY.

de Farias, D. P., 2002. The linear programming approach to approximate dynamic programming. Phd thesis, Department of Management Science and Engineering, Stanford University, USA.

de Farias, D. P., Roy, B. V., 2003a. Approximate linear programming for average-cost dynamic programming. Advances in Neural Information Processing Systems 15, 1587–1594.

de Farias, D. P., Roy, B. V., 2003b. The linear programming approach to approximate dynamic programming. Operations Research 51 (6), 850–865.

Deb, K., Agrawal, S., Pratab, A., Meyarivan, T., 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Proceedings of the VI Conference in Parallel Problem Solving from Nature 1917, 849–858.

Deb, K., Pratab, A., Agrawal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6 (2), 182–197.

d'Onofrio, A., 2005. On pulse vaccination strategy in the SIR epidemic model with vertical transmission. Applied Mathematics Letters 18 (7), 729–732.

Feltrin, C. C., Rafikov, M., 2002. Aplicação da função de Lyapunov num problema de controle ótimo de pragas. Tendências em Matemática Aplicada e Computacional 3 (2), 83–92.

Fiedler-Ferrara, N., do Prado, C. P. C., 1994. Caos: uma introdução. Edgard Blücher, São Paulo.

Fletcher, R., Powell, M. J. D., 1963. A rapidly convergent descent method for minimization. Computer Journal 6, 163–168.

Gallego, C. G., Katircioglu, K., Ramachandran, B., 2007. Inventory management under highly uncertain demand. Operations Research Letters 35, 281–289.

Garcia, C. E., Prett, D. M., Morari, M., 1989. Model predictive control - theory and practice - a survey. Automatica 25 (3), 335–348.

Goldbarg, M. C., Luna, H. L. P., 2000. Otimização Combinatória e Programação Linear. Editora Campus, Rio de Janeiro.

Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley.

Gonzaga, C. C., 1990. Algoritmos de Pontos Interiores para Programação Linear. XVII Colóquio Brasileiro de Matemática, Instituto de Matemática Pura e Aplicada do CNPq, Rio de Janeiro.

Hadar, J., Russell, W. R., 1969. Rules for ordering uncertain prospects. The American Economic Review 59 (1), 25–34.

Jacobson, D. H., Mayne, D. Q., 1970. Differential Dynamic Programming. Elsevier.

Kacprzyk, J., Esogbue, A. O., 1996. Fuzzy dynamic programming and its extensions. Fuzzy Sets and Systems 81 (1), 31–45.

Kushner, H., 1971. Introduction to Stochastic Control. Holt, Rinehart and Winston, Inc.

Lakshmikantham, V., Bainov, D. D., Simeonov, P. S., 1989. Theory of impulsive differential equations. Vol. 6. World Scientific.

Levy, H., 1992. Stochastic dominance and expected utility: Survey and analysis. Management Science 38 (4), 555–585.

Liao, L., Li, D., 2002. Adaptative differential dynamic programming for multiobjective optimal control. Automatica 38 (6), 1003–1015.

Lincoln, B., Rantzer, A., 2006. Relaxing dynamic programming. IEEE Transactions on Automatic Control 51 (8), 1249–1260.

Liu, B., Teng, Z., Chen, L., 2006. Analysis of a predator-prey model with Holling II functional response concerning impulsive control strategy. Journal of Computational and Applied Mathematics 193 (2), 347–362.

Luenberger, D. G., 1984. Linear and Nonlinear Programming. Addison-Wesley.

Luus, R., 1990. Optimal control by dynamic programming using systematic reduction in grid size. International Journal of Control 51 (5), 995–1010.

Luus, R., 2000. Iterative Dynamic Programming. Chapmam & Hall/CRC.

Meyer, P. L., 1983. Probabilidade: Aplicações à Estatística. Livros Técnicos e Científicos, Rio de Janeiro.

Miettinen, K., 1998. Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Norwell, MA.

Monteiro, L. H., 2002. Sistemas Dinâmicos. Livraria da Física, São Paulo.

Nepomuceno, E. G., 2005. Dinâmica, modelagem e controle de epidemias. Tese de doutorado, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

Ogata, K., 1998. Engenharia de Controle Moderno. Prentice-Hall, Rio de Janeiro.

Park, Y. M., Park, J. B., Won, J. R., 1998. A hybrid genetic algorithm/dynamic programming approach to optimal long-term generation expansion planning. International Journal of Electrical Power and Energy Systems 20 (4), 295–303.

Parreiras, R. O., 2006. Algoritmos evolucionários e técnicas de tomada de decisão em análise multicritério. Tese de doutorado, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

Poulsen, N. K., 2004. Dynamic optimization, versão *on line*.
URL `www2.imm.dtu.dk/pubdb/views/edoc/download.php/3283/pdf/imm3283.pdf`

Ross, S. M., 2002a. A First Course in Probability. Macmillan.

Ross, S. M., 2002b. Simulation. Academic Press.

Sarkar, D., Mondak, J. M., 2005. Pareto-optimal solutions for multi-objective optimization of fed-batch bioreactors using nondominated sorting genetic algorithm. Chemical Engineering Science 60 (2), 481–492.

Silva, V. L. S., Cruz, A. R., Carrano, E. G., Takahashi, R. H. C., aes, F. G., 2006. On nonlinear fitness functions for ranking-based selection. IEEE World Congress on Computational Intelligence .

Slack, N., Chambers, S., Harrison, A., 2002. Administração da Produção, 2nd Edition. Editora Atlas, São Paulo.

Sun, J. T., Zhang, Y. P., 2003. Stability analysis of impulsive control systems. IEE Proceedings in Control Theory and Applications 150 (4), 331–334.

Takahashi, R. H. C., 2004. Otimização escalar e vetorial, notas de aula.
URL `www.mat.ufmg.br/~taka/Download/otev04.pdf`

Takahashi, R. H. C., Caldeira, C. P., Peres, P. L. D., 1997. A linear dynamic system approach for cattle herd optimal shaping. International Journal of Systems Science 28 (9), 913–918.

Takahashi, R. H. C., Peres, P. L. D., Barbosa, L. L. S., 1999. A sliding mode controlled sinusoidal voltage source with ellipsoidal switching surface. IEEE Transactions on Circuits and Systems 46 (6), 714–721.

Thompson, A. M., Cluett, W. R., 2005. Stochastic iterative dynamic programming: A Monte Carlo approach to dual control. Automatica 41 (4), 767–778.

Tospornsampan, J., Kita, I., Ishii, M., Kitamura, Y., 2005. Optimization of a multiple reservoir system operation using a combination of genetic algorithm and discrete differential dynamic programming: a case study in mae klong system, thailand. Journal Paddy and Water Environment 3 (1), 29–38.

Triola, M. F., 2005. Introdução à estatistica. LTC, Rio de Janeiro.

Wagner, H. M., 1986. Pesquisa Operacional. Prentice-Hall do Brasil, São Paulo.

Wanner, E. F., 2006. Operadores para algoritmos genéticos baseados em aproximações quadráticas de funções de variáveis contínuas. Tese de doutorado, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

Weber, R., 2003. Optimization and control, versão *on line*.
URL `www.statslab.cam.ac.uk/~rrw1/oc/index.html`

Yang, T., 1999. Impulsive control. IEEE Transactions on Automatic Control 44 (5), 1081–1083.

Yang, T., 2001. Impulsive Control Theory. Spring-Verlag.

Zenely, M., 1974. Linear Multiobjective Programming. Springer-Verlag.

Zhou, Y., Liu, H., 2003. Stability of periodic solutions for an SIS model with pulse vaccination. Mathematical and Computer Modelling 38 (13), 299–308.