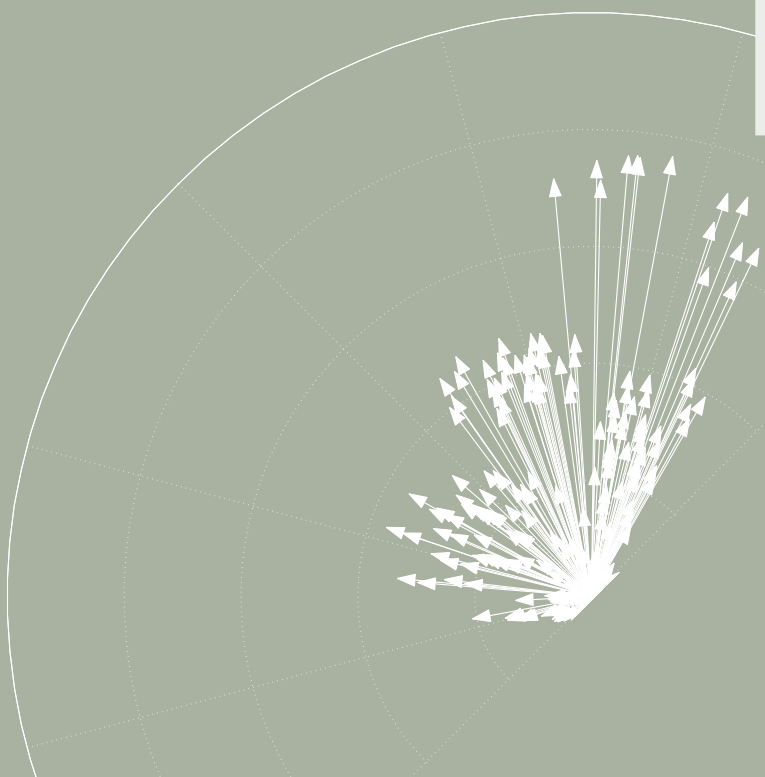
MANUAL DE COMPUTAÇÃO EVOLUTIVA E METAHEURÍSTICA

ANTÓNIO GASPAR-CUNHA
RICARDO TAKAHASHI
CARLOS HENGGELER ANTUNES
COORDENADORES



IMPRENSA DA UNIVERSIDADE DE COIMBRA
COIMBRA UNIVERSITY PRESS

EDITORA ufmg

CAPÍTULO 7

Evolução Diferencial

Frederico G. Guimarães

Departamento de Engenharia Elétrica
Universidade Federal de Minas Gerais

Este capítulo apresenta os algoritmos de *evolução diferencial* para a solução de problemas de otimização. Inicialmente, faz-se uma apresentação do algoritmo em sua versão básica e uma análise de seu comportamento em algumas funções-objetivo usadas como exemplo. Na sequência, são discutidos aspectos avançados e variações do algoritmo de evolução diferencial. Posteriormente, é desenvolvido um esquema geral para os algoritmos de evolução diferencial.

1. Introdução

O algoritmo de Evolução Diferencial é um algoritmo de otimização simples e eficiente que tem recebido cada vez mais destaque no âmbito da otimização não linear com variáveis contínuas. A primeira publicação sobre esse algoritmo ocorreu em 1995, em um relatório técnico de Rainer Storn e Kenneth Price (Storn e Price, 1995). O algoritmo ganhou destaque na comunidade internacional de Computação Evolutiva nos anos seguintes, após apresentar excelente desempenho nas edições de 1996 e 1997 da *International Contest on Evolutionary Optimization* da *IEEE International Conference on Evolutionary Computation* (IEEE ICEC), ver (Storn e Price, 1996; Price, 1997). Na edição de 1996, o algoritmo de evolução diferencial ficou em terceiro lugar. O algoritmo classificado em primeiro

explorava a característica de separabilidade presente nas funções de teste usadas na competição e o algoritmo classificado em segundo lugar se sustentava em quadrados latinos e, por essa razão, não era muito escalável para problemas com muitas variáveis. O algoritmo de evolução diferencial foi o melhor classificado entre os algoritmos de otimização de propósito geral, perdendo para dois métodos mais especializados e menos gerais. Já na edição de 1997, com um novo conjunto de funções de teste, o algoritmo de evolução diferencial apresentou o melhor desempenho entre os algoritmos classificados. Em dezembro do mesmo ano, Storn e Price (1997) publicaram um artigo no *Journal of Global Optimization* apresentando testes experimentais mais amplos e resultados empíricos que ilustravam a robustez do algoritmo.

Em 1999, foi publicado o livro *New Ideas in Optimization*, editado pelos pesquisadores David Corne, Marco Dorigo e Fred Glover, incluindo uma seção de capítulos sobre o algoritmo de evolução diferencial (Lampinen e Zelinka, 1999; Price, 1999; Storn, 1999). Atualmente, já podem ser encontrados alguns livros dedicados exclusivamente aos algoritmos de evolução diferencial, ver por exemplo (Price et al., 2005; Feoktistov, 2006; Chakraborty, 2008), incluindo um livro recente com foco e diversas aplicações em engenharia elétrica e eletrônica (Qing, 2009).

Nos últimos anos, esse algoritmo tem se mostrado versátil e eficaz em muitas aplicações práticas, desde o projeto de filtros digitais (Storn, 1999) até a otimização de sistema reservatório de água (Reddy e Kumar, 2007) e localização do foco sísmico de terremotos (Ruzek e Kvasnicka, 2001). O algoritmo de evolução diferencial também tem se mostrado eficiente para o treinamento de redes neurais (Masters e Land, 1997; Magoulas et al., 2001; Abbass, 2002; Ilonen et al., 2003), para o projeto de dispositivos de engenharia elétrica (Pahner e Hameyer, 2000; Kim et al., 2007) e solução de problemas inversos (Michalski, 2001).

Pesquisas recentes têm se concentrado no estudo de variações do algoritmo (Mezura-Montes et al., 2006), em técnicas para o tratamento de restrições (Kim et al., 2007; Gong e Cai, 2008) e em versões para a solução de problemas de otimização multiobjetivo (Xue et al., 2005; Reddy e Kumar, 2007; Batista et al., 2009).

Este capítulo tem como objetivo apresentar uma visão geral sobre esse algoritmo e seu peculiar mecanismo de busca, sustentado no operador de *mutação diferencial*, que dá nome ao algoritmo. Embora o método de evolução diferencial seja classificado como um algoritmo evolutivo, e se enquadre em um esquema geral de um algoritmo evolutivo, a mutação diferencial não tem base ou inspiração em nenhum processo natural. A forma como esse operador gera perturbações (mutações) nos indivíduos da população se sustenta em argumentos matemáticos e heurísticos que indicam sua adequabilidade para a otimização de funções e não exatamente em argumentos derivados de metáforas da natureza. Contudo, o algoritmo de evolução diferencial segue uma linha histórica de algoritmos e métodos que evoluem uma população de soluções candidatas segundo operadores heurísticos inspirados em mecanismos bastante gerais de adaptação natural. Por essa razão, o algoritmo de evolução diferencial é classificado como uma instância dos algoritmos evolutivos.

A mutação diferencial emprega a diferença entre pares de indivíduos na população corrente para gerar os vetores de perturbação, denominados *vetores-diferença*. Porém, à medida que o algoritmo progride no processo de busca, a distribuição espacial da população se modifica de acordo com o panorama da função-objetivo. Essa mudança, por sua vez, altera as orientações e tamanhos dos vetores-diferença que podem ser criados a partir da população. Por essa razão, observa-se que a distribuição dos vetores-diferença, e portanto a distribuição das direções e tamanhos de passo das perturbações, se ajusta ao panorama da função. Essa característica de autoadaptação da mutação diferencial confere ao algoritmo de evolução diferencial qualidades interessantes do ponto de vista da otimização, tais como robustez, versatilidade e eficiência em diversos problemas. Neste capítulo, mostraremos por meio de exemplos essa autoadaptação dos vetores-diferença à medida que o algoritmo avança no processo de otimização.

2. Evolução Diferencial

Nesta seção é apresentada uma visão geral do algoritmo de evolução diferencial. Considere um problema genérico de otimização não linear com variáveis contínuas, formulado como:

$$\mathbf{x}^* = \arg\min f(\mathbf{x})$$

$$\text{sujeito a:} \begin{cases} \mathbf{x} \in \mathbb{R}^n \\ g_1(\mathbf{x}) \leq 0 \\ \vdots \\ g_m(\mathbf{x}) \leq 0 \end{cases} \tag{7.1}$$

Neste capítulo, vamos considerar o problema irrestrito, isto é, sem as funções de restrição $g_1(\mathbf{x})$ a $g_m(\mathbf{x})$. Lampinen (2002) apresenta uma forma simples de tratar restrições no algoritmo de evolução diferencial. O tratamento de restrições em algoritmos evolutivos será discutido de forma mais geral no Capítulo 14.

Ao longo deste capítulo, usaremos a notação $\mathcal{U}_{[a,b]}$ para indicar a amostragem de uma variável aleatória com distribuição uniforme entre a e b e a notação $\mathcal{N}_{[\mu,\sigma]}$ para indicar a amostragem de uma variável aleatória com distribuição normal com média μ e desvio padrão σ.

Seja uma população de soluções candidatas para o problema, representada por $X_t = \{\mathbf{x}_{t,i};\ i = 1, \ldots, N\}$, em que t é o índice da geração corrente e i é o índice do indivíduo na população. Cada indivíduo na população corrente é representado por um vetor coluna:

$$\mathbf{x}_{t,i} = \begin{bmatrix} x_{t,i,1} \\ x_{t,i,2} \\ \vdots \\ x_{t,i,n} \end{bmatrix} \tag{7.2}$$

dessa forma, o terceiro índice indica uma entre as n variáveis do problema de otimização.

O mecanismo de busca do algoritmo de evolução diferencial utiliza vetores-diferença criados a partir de pares de vetores da própria população. Dois indivíduos são selecionados aleatoriamente da população corrente, criando-se um vetor-diferença que nada mais é do que a diferença entre estes dois indivíduos. Este vetor-diferença, por sua vez, é somado a um terceiro indivíduo, também selecionado aleatoriamente, produzindo uma nova solução mutante. A nova solução mutante é portanto o resultado de uma perturbação em algum indivíduo da população, sendo esta perturbação um vetor-diferença construído aleatoriamente. A equação a seguir ilustra esse procedimento:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right), \quad r_1, r_2, r_3 \in \{1, \ldots, N\} \tag{7.3}$$

em que $\mathbf{v}_{t,i}$ representa a i-ésima solução mutante e F é um fator de escala aplicado ao vetor-diferença e parâmetro do algoritmo de evolução diferencial. O vetor $\mathbf{x}_{t,r_1}$, ao qual é aplicada a mutação diferencial, é denominado *vetor de base*.

Usando este procedimento, obtém-se uma população mutante $V_t = \{\mathbf{v}_{t,i};\ i = 1, \ldots, N\}$. Os próximos passos no algoritmo são bem simples. Os indivíduos da população corrente X_t são recombinados com os indivíduos da população mutante, produzindo a descendência ou população de soluções teste U_t. Na versão clássica do algoritmo de evolução diferencial, emprega-se a recombinação discreta com probabilidade $C \in [0, 1]$:

$$u_{t,i,j} = \begin{cases} v_{t,i,j}, & \text{se } \mathcal{U}_{[0,1]} \leq C \ \vee \ j = \delta_i \\ x_{t,i,j}, & \text{caso contrário} \end{cases} \tag{7.4}$$

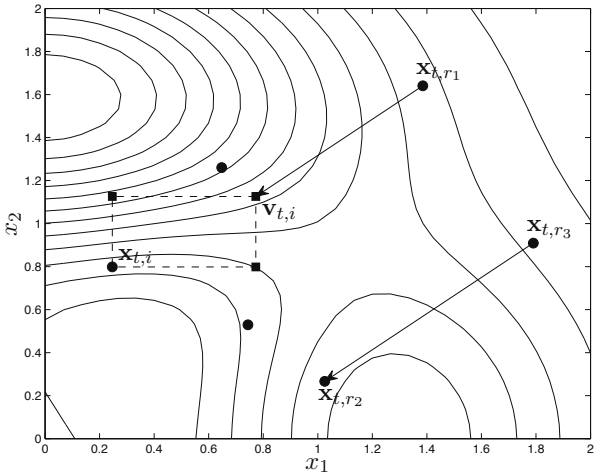


Figura 7.1: Ilustração do procedimento de geração de uma solução mutante.

em que $\delta_i \in \{1, \ldots, n\}$ é um índice aleatório sorteado para o vetor teste i. Como em algum momento a igualdade $j = \delta_i$ será verificada, essa condição garante que pelo menos um dos parâmetros da solução teste será herdado do indivíduo mutante. O parâmetro C controla a fração de valores em $\mathbf{u}_{t,i}$ que são copiados do vetor mutante $\mathbf{v}_{t,i}$. Quanto mais próximo de 1 for o valor de C, maior a chance de que a solução teste contenha muitos valores herdados do vetor mutante. No limite, quando $C = 1$, o vetor teste é igual ao vetor mutante.

A Figura 7.1 ilustra a geração de um vetor mutante e as possíveis soluções teste obtidas após a recombinação, indicadas por ■ na Figura. Note que pelo menos a coordenada $j = \delta_i$ será herdada do vetor mutante, dessa forma, garante-se $\mathbf{u}_{t,i} \neq \mathbf{x}_{t,i}$. Pode-se observar que uma solução teste é o resultado da recombinação de cada solução $\mathbf{x}_{t,i}$ com uma solução mutante gerada a partir de uma perturbação em algum indivíduo da população. A direção e o tamanho dessa perturbação são definidos pela diferença entre as soluções $\mathbf{x}_{t,r_2}$ e $\mathbf{x}_{t,r_3}$, portanto dependem das posições relativas destes indivíduos no domínio de busca.

Finalmente, o valor da função-objetivo é avaliado em $\mathbf{u}_{t,i}$. Cada solução teste $\mathbf{u}_{t,i}$ é comparada com seu correspondente na população corrente, no caso $\mathbf{x}_{t,i}$. Se a solução teste é melhor do que a solução corrente $\mathbf{x}_{t,i}$, a solução corrente é eliminada e seu lugar passa a ser ocupado por $\mathbf{u}_{t,i}$. Caso contrário, a solução teste é descartada e a solução corrente sobrevive, permanecendo na população da próxima geração, representada por X_{t+1}. O processo se repete até que algum critério de parada definido seja satisfeito.

Como pode-se observar, o algoritmo de evolução diferencial em sua versão original é bastante simples. Essa simplicidade torna-se evidente quando escrevemos as equações (7.3) e (7.4) na forma compacta a seguir:

$$u_{t,i,j} = \begin{cases} x_{t,r_1,j} + F\left(x_{t,r_2,j} - x_{t,r_3,j}\right), & \text{se } \mathcal{U}_{[0,1]} \leq C \ \vee \ j = \delta_i \\ x_{t,i,j}, & \text{caso contrário} \end{cases} \tag{7.5}$$

com $r_1, r_2, r_3 \in \{1, \ldots, N\}$, $t = 1, \ldots, t_{\max}$, $i = 1, \ldots, N$, e $j = 1, \ldots, n$.

A seleção para sobrevivência pode ser descrita por:

$$\mathbf{x}_{t+1,i} = \begin{cases} \mathbf{u}_{t,i}, & \text{se } f(\mathbf{u}_{t,i}) \leq f(\mathbf{x}_{t,i}) \\ \mathbf{x}_{t,i}, & \text{caso contrário} \end{cases} \tag{7.6}$$

As operações do algoritmo de evolução diferencial básico são apresentadas na forma de pseudocódigo no Algoritmo 1 a seguir.

Algoritmo 1 Pseudocódigo do algoritmo de evolução diferencial básico

```
1: t ← 1
2: Inicializar população X_t = {x_{t,i}; i = 1, ..., N}
3: enquanto algum critério de parada não for satisfeito faça
4:    para i = 1 até N faça
5:       Selecione aleatoriamente r_1, r_2, r_3 ∈ {1, ..., N}
6:       Selecione aleatoriamente δ_i ∈ {1, ..., n}
7:       para j = 1 até n faça
8:          se U_[0,1] ≤ C ∨ j = δ_i então
9:             u_{t,i,j} = x_{t,r_1,j} + F (x_{t,r_2,j} − x_{t,r_3,j})
10:         senão
11:            u_{t,i,j} = x_{t,i,j}
12:         fim se
13:      fim para
14:   fim para
15:   para i = 1 até N faça
16:      se f(u_{t,i}) ≤ f(x_{t,i}) então
17:         x_{t+1,i} ← u_{t,i}
18:      senão
19:         x_{t+1,i} ← x_{t,i}
20:      fim se
21:   fim para
22:   t ← t + 1
23: fim enquanto
```

A equações (7.5)-(7.6) descrevem todas as operações necessárias no algoritmo de evolução diferencial em sua versão básica. É realmente impressionante que um algoritmo tão simples apresente tantas características computacionais desejáveis, tais como robustez, versatilidade, eficiência e adaptabilidade. Nas próximas seções, tentaremos compreender o comportamento do algoritmo em alguns exemplos de funções-objetivo de forma a elucidar seu mecanismo de funcionamento. Além disso, discutiremos variantes do algoritmo básico e aspectos avançados da evolução diferencial.

3. Comportamento da Mutação Diferencial

Nesta seção discutiremos o comportamento do algoritmo de evolução diferencial, buscando compreender os princípios de funcionamento de seu mecanismo de busca.

O princípio básico de funcionamento do algoritmo de evolução diferencial é perturbar soluções da população corrente gerando vetores mutantes. Essas perturbações são proporcionais à diferença entre pares de soluções escolhidas aleatoriamente na população. Portanto, para entender melhor o comportamento do algoritmo convém verificar a distribuição dos possíveis vetores-diferença em diversos instantes do processo de otimização.

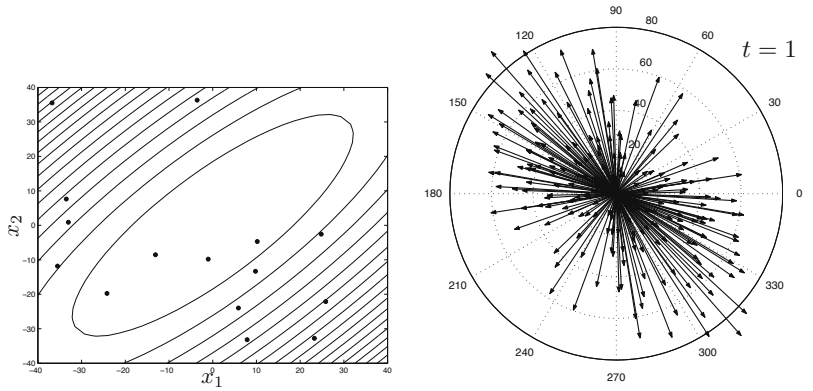
Vamos considerar uma população de tamanho N, supondo inicialmente que os N vetores sejam distintos entre si. Existem ao todo N^2 combinações de diferenças possíveis, das quais N são nulas, pois correspondem a diferenças de um vetor com ele mesmo. As $N(N-1)$ diferenças restantes são não nulas. Além disso, essas $N(N-1)$ diferenças apresentam simetria, porque cada diferença $(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3})$ possui seu simétrico correspondente, bastando para isso trocar os índices r_2 e r_3. Como os índices são escolhidos aleatoriamente com distribuição uniforme, a probabilidade de selecionar uma dada diferença e seu simétrico é a mesma. Fica claro que a distribuição de todos os vetores-diferença possíveis de ser construídos com N vetores distintos quaisquer apresenta média nula, uma vez que cada vetor possui seu correspondente negativo e uma mesma probabilidade de ser selecionado.

Matematicamente, tem-se:

$$\langle \Delta \mathbf{x}_t \rangle = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\mathbf{x}_{t,i} - \mathbf{x}_{t,j}) = \mathbf{0} \tag{7.7}$$

Nesta seção, adotaremos como recurso de visualização o desenho da distribuição dos vetores-diferença em um gráfico polar, considerando que todos tenham a mesma origem em $(\rho = 0, \theta = 0)$. O gráfico polar nos ajuda a visualizar a distribuição dos vetores-diferença de acordo com suas orientações e tamanhos.

Função convexa



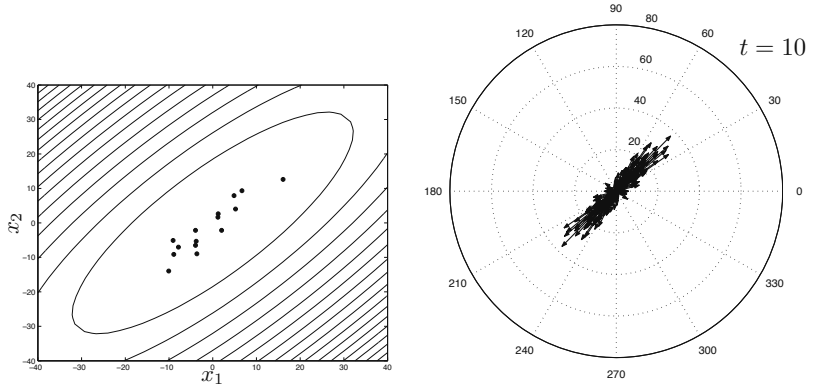
(a) (b)

Figura 7.2: Função-objetivo quadrática. (a) Distribuição espacial da população na geração $t = 1$. (b) Distribuição dos vetores-diferença na geração $t = 1$.

O ponto fundamental para entender o funcionamento do algoritmo de evolução diferencial é perceber que *a distribuição dos vetores-diferença depende da distribuição espacial dos indivíduos da população no problema em questão.* À medida que a população se distribui de acordo com o "contorno" da função, a distribuição dos vetores-diferença também se ajusta a esse contorno.

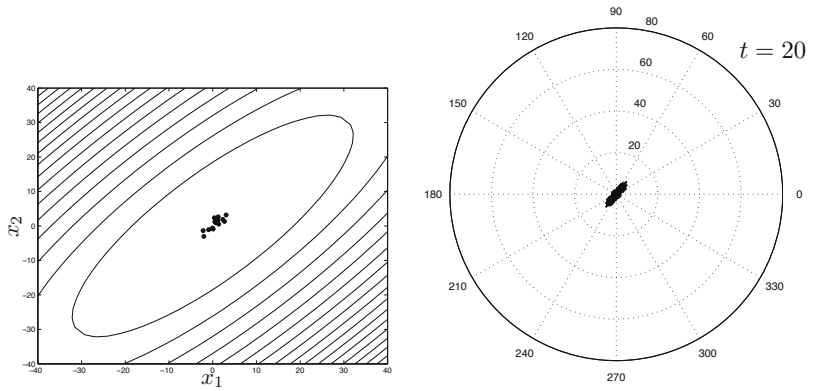
As Figuras 7.2 a 7.4 ilustram essa propriedade do algoritmo em uma função quadrática, cujas curvas de nível correspondem a elipsóides rotacionados de $\pi/4$ no sentido anti-horário em relação aos eixos coordenados. Além disso, um dos eixos desse elipsóide é maior do que o outro, tornando o elipsóide "alongado" numa dada direção.

Na primeira geração, a população é distribuída aleatoriamente com distribuição uniforme em uma região retangular que corresponde aos limites inferiores e superiores de cada variável. Neste primeiro



(a) (b)

Figura 7.3: Função-objetivo quadrática. (a) Distribuição espacial da população na geração $t = 10$. (b) Distribuição dos vetores-diferença na geração $t = 10$.



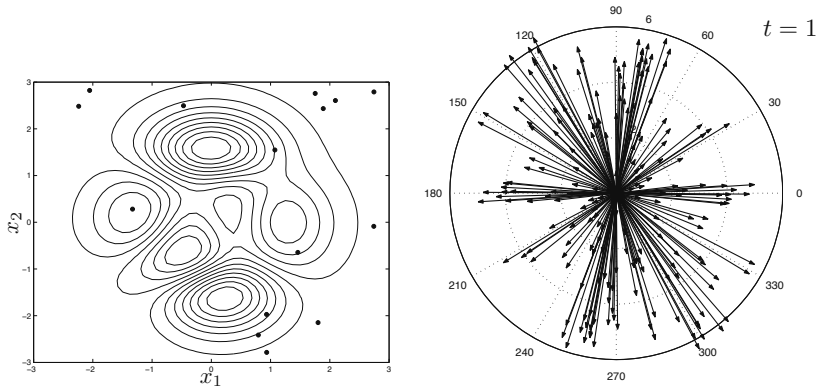
(a) (b)

Figura 7.4: Função-objetivo quadrática. (a) Distribuição espacial da população na geração $t = 20$. (b) Distribuição dos vetores-diferença na geração $t = 20$.

momento, a população não possui nenhuma informação sobre o contorno da função. A Figura 7.2-(a) ilustra a distribuição espacial inicial da população e o gráfico na Figura 7.2-(b) mostra a distribuição dos vetores-diferença correspondente. A distribuição inicial dos vetores-diferença não é polarizada em nenhuma direção, havendo vetores com vários tamanhos distintos e apontando para diversas direções.

Esses vetores-diferença são usados na operação de mutação diferencial para perturbar indivíduos da população, produzindo a população mutante V_t. Cada indivíduo $\mathbf{x}_{t,i}$ sofre recombinação com seu mutante correspondente, produzindo uma solução teste $\mathbf{u}_{t,i}$. Algumas soluções teste serão piores do que as soluções originais e serão descartadas, porém, algumas soluções teste serão melhores e substituirão as soluções originais. Nesse momento, a distribuição espacial da população corrente X_t se altera. A tendência é que, ao longo das gerações, essa distribuição espacial se alinhe com o contorno da função.

A Figura 7.3 ilustra a distribuição espacial da população e a distribuição dos vetores-diferença 10 gerações após a distribuição inicial. Observe que a distribuição dos vetores-diferença agora está mais



(a) (b)

Figura 7.5: Função-objetivo multimodal. (a) Distribuição espacial da população na geração $t = 1$. (b) Distribuição dos vetores-diferença na geração $t = 1$.

alinhada com o eixo maior do elipsóide, ou seja, mais alinhada com a direção mais favorável para a minimização desta função-objetivo quadrática. Outra característica interessante é que os tamanhos de passo estão menores do que na distribuição da Figura 7.2-(b), devido à aglomeração dos indivíduos em torno do ponto de mínimo.

A Figura 7.4 ilustra a distribuição espacial da população e a distribuição dos vetores diferenciais na geração $t = 20$. A população está agora mais próxima do ponto de mínimo e ocupa um volume reduzido em relação à distribuição espacial em $t = 1$. A distribuição dos vetores-diferença continua alinhada com os elipsóides que formam as curvas de nível da função, porém os tamanhos desses vetores estão reduzidos, favorecendo uma busca bem mais local. Nesse momento, o algoritmo converge para o mínimo local.

A sequência de gráficos nas Figuras 7.2 a 7.4 mostra que as direções e os tamanhos de passo dos vetores usados na perturbação das soluções se adaptam ao longo do processo de otimização. As orientações dos vetores-diferença se alinham com a direção mais favorável para a minimização e os tamanhos dos vetores-diferença diminuem à medida que a população se aglomera em torno de algum ponto, favorecendo uma busca cada vez mais local.

Esse exemplo ilustra o comportamento geral do algoritmo de evolução diferencial em funções convexas, mostrando claramente a adaptação dos tamanhos de passo e das direções das mutações. As seções seguintes ilustram o comportamento do algoritmo em uma função multimodal e em uma função unimodal não convexa.

Função multimodal

Na seção anterior vimos como o algoritmo de evolução diferencial se comporta em uma função convexa. Contudo, cabe explorar o comportamento do algoritmo em uma função multimodal, com diversos mínimos locais. Em uma função multimodal, a população tende a se distribuir em torno dos mínimos locais. Nesse caso, convém verificar como ficará a distribuição dos vetores-diferença. Para explorar essas questões, usaremos a mesma metodologia usada anteriormente, mostrando a distribuição espacial da população e o gráfico em coordenadas polares da distribuição de vetores-diferença correspondente em instantes distintos da otimização.

As Figuras 7.5 a 7.7 mostram o resultado obtido para o caso da otimização de uma função-objetivo multimodal.

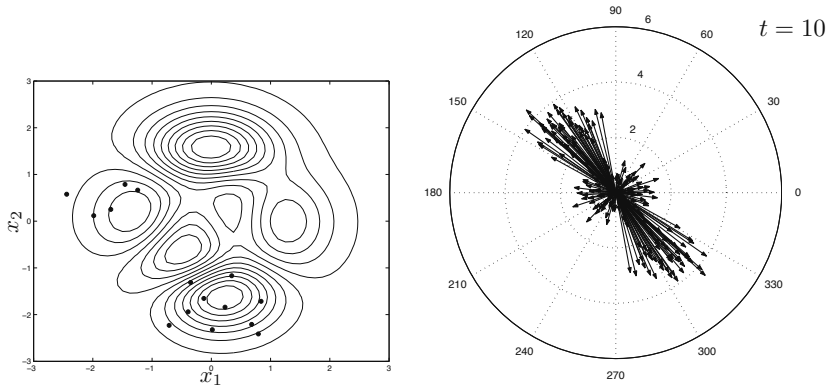


Figura 7.6: Função-objetivo multimodal. (a) Distribuição espacial da população na geração $t = 10$. (b) Distribuição dos vetores-diferença na geração $t = 10$.

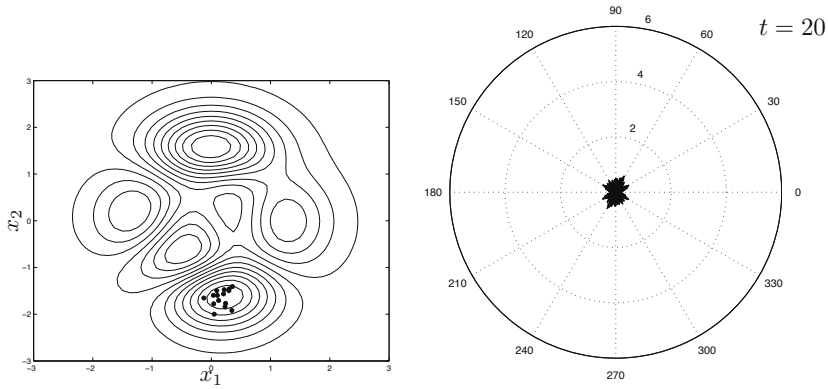


Figura 7.7: Função-objetivo multimodal. (a) Distribuição espacial da população na geração $t = 20$. (b) Distribuição dos vetores-diferença na geração $t = 20$.

Na primeira geração, os vetores-diferença apontam para quase todas as direções e possuem diversos tamanhos devido à geração aleatória da população inicial. Após algumas gerações, ver Figura 7.6, a população se concentra nas duas bacias de atração existentes[1]. Observe a distribuição de vetores-diferença correspondente na Figura 7.6-(b). Pode-se notar dois grupos bem definidos de vetores. Essa distribuição apresenta um conjunto de vetores de pequena amplitude, formados por pares de soluções que se localizam na mesma bacia de atração. Os vetores desse conjunto favorecem uma busca local em cada bacia de atração. A distribuição de vetores-diferença apresenta um segundo conjunto de vetores de maior amplitude, estes por sua vez formados por pares de soluções que se localizam cada uma em bacias distintas. Além disso, as direções desses vetores-diferença de maior amplitude se alinham com a direção que une as duas bacias de atração. Esses vetores-diferença favorecem uma busca global, levando a perturbações que permitem "saltar" de uma bacia à outra, além de gerar soluções na região

[1] Esta função apresenta três bacias de máximo local e duas bacias de mínimo local.

intermediária entre as duas bacias.

Finalmente, na geração $t = 20$, ver Figura 7.7, a população se concentra em uma única bacia, aquela que apresenta melhores valores de função-objetivo. Os vetores-diferença automaticamente diminuem de tamanho, favorecendo uma busca local e mais refinada. A convergência para o mínimo global da função na região considerada é agora iminente.

Esse exemplo ilustra que, no caso de funções multimodais, a distribuição espacial da população se concentra em bacias de atração distintas, causando a geração de grupos de vetores-diferença bem definidos. Alguns conjuntos de vetores causam perturbações que levam a saltos na direção de uma bacia à outra, enquanto outros grupos de vetores-diferença causam perturbações pequenas, favorecendo uma busca local em cada bacia de atração. Após algumas gerações, a população se concentra em uma única bacia e a distribuição de vetores-diferença se reduz ao caso de uma função-objetivo convexa.

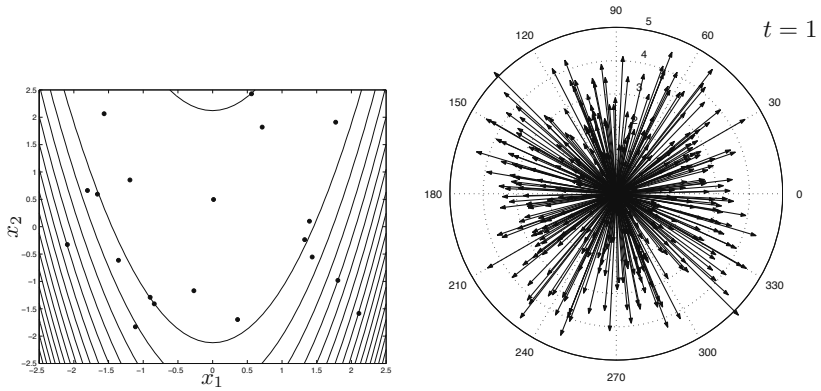
Função unimodal não convexa

Vamos analisar agora o comportamento do algoritmo em uma função unimodal não convexa. Usaremos a função a seguir, uma função-objetivo de teste conhecida como função de Rosenbrock:

$$f(\mathbf{x}) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2 \tag{7.8}$$

Essa função-objetivo é unimodal e suas curvas de nível formam conjuntos não convexos. Observando as curvas de nível desta função, vemos que seu ponto de mínimo se localiza em um vale estreito e longo com curvatura na forma de uma parábola. Essa região apresenta uma inclinação pequena que dificulta a convergência de métodos de otimização baseados em derivadas.

As Figuras 7.8 a 7.10 mostram o comportamento do algoritmo para essa função não convexa. Observe que após algumas gerações ($t = 10$) a população se concentra em torno do vale em curva da função. O gráfico da distribuição de vetores-diferença apresenta características interessantes. Observa-se dois grupos de vetores em direções quase ortogonais, quase formando a figura da letra 'x'. Os vetores-diferença em cada grupo são formados por pares de vetores localizados em uma das metades da curvatura em formato de parábola. Além disso, observa-se um grupo de vetores com amplitude maior que representam perturbações as quais levam de uma metade à outra da curvatura. O comportamento



(a) (b)

Figura 7.8: Função-objetivo unimodal não convexa. (a) Distribuição espacial da população na geração $t = 1$. (b) Distribuição dos vetores-diferença na geração $t = 1$.

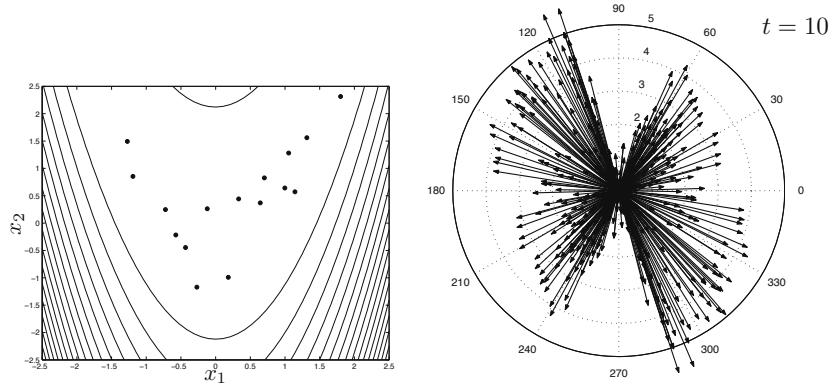


Figura 7.9: Função-objetivo unimodal não convexa. (a) Distribuição espacial da população na geração $t = 10$. (b) Distribuição dos vetores-diferença na geração $t = 10$.

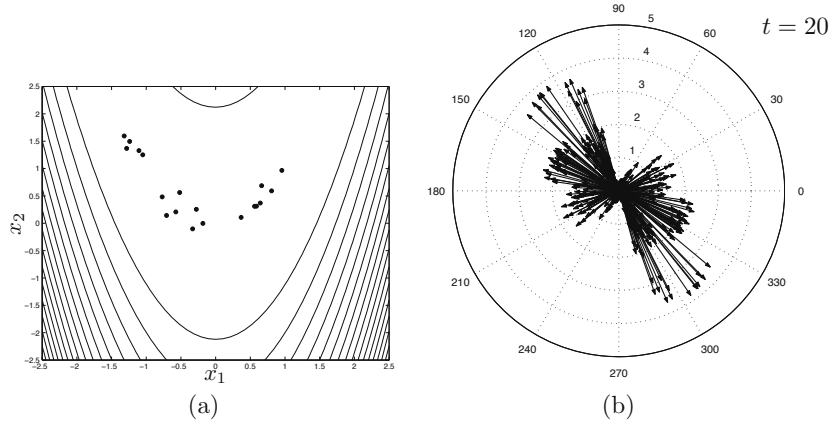


Figura 7.10: Função-objetivo unimodal não convexa. (a) Distribuição espacial da população na geração $t = 20$. (b) Distribuição dos vetores-diferença na geração $t = 20$.

do algoritmo de evolução diferencial nesta função é similar ao comportamento do algoritmo em uma função multimodal.

Quando a distribuição espacial da população apresenta "formato" linear, como no caso da função convexa, dizemos que há uma forte correlação linear entre as variáveis nessa distribuição espacial. O algoritmo de evolução diferencial é capaz de descobrir correlações lineares na distribuição espacial da população e utilizar essa informação para produzir perturbações favoráveis. Entretanto, no caso de distribuições mais complexas do que uma reta, como a distribuição espacial curva na Figura 7.10, o algoritmo trata essa distribuição como uma combinação de distribuições com correlações lineares. Dessa forma, o algoritmo trata funções não convexas, que causariam distribuições espaciais com formatos curvos mais complicados, como se fossem funções multimodais, mesmo que a função não convexa em questão seja unimodal. Cada trecho aproximadamente convexo é enxergado pelo algoritmo como uma bacia de atração e gera um grupo de vetores-diferença.

Na Figura 7.10-(a), pode-se perceber três agrupamentos de indivíduos bem destacados na distribuição espacial da população. Na Figura 7.10-(b), que ilustra a distribuição de vetores-diferença correspondente, pode-se notar também três grupos de vetores-diferença bem distintos. Cada grupo de vetores-diferença está alinhado com a direção que conecta dois agrupamentos distintos e gera perturbações que permitem saltar de um grupo de pontos ao outro. O algoritmo se comporta como se existissem três bacias de atração na função.

Rotação e translação

Uma das características interessantes do operador de mutação diferencial é sua invariância à rotação e translação do sistema de coordenadas. É fácil verificar que a mutação diferencial é invariante à rotação e translação, já que se trata de uma operação vetorial. O que importa na definição dos vetores-diferença são as posições relativas dos indivíduos da população, não suas posições absolutas.

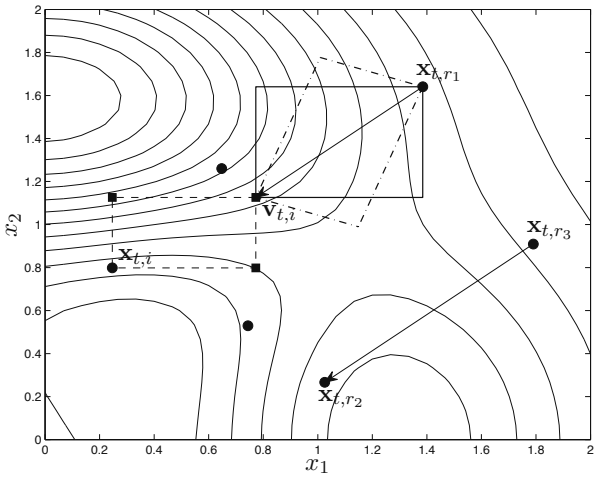


Figura 7.11: Invariância da mutação diferencial em relação à rotação do sistema de coordenadas.

A propriedade de invariância à rotação é uma característica desejável em algoritmos de otimização. Um algoritmo cujo desempenho não depende da orientação do sistema de coordenadas é mais geral, uma vez que a orientação ideal não é conhecida na prática.

Embora a mutação diferencial apresente essas características desejáveis, a recombinação discreta usada no algoritmo de evolução diferencial clássico não possui invariância à rotação. Observando a Figura 7.11, percebe-se claramente que os vetores teste produzidos pela recombinação discreta variam com a rotação do sistema de coordenadas, exceto quando $C = 1$, pois nesse caso não há recombinação, somente mutação, e $\mathbf{u}_{t,i} = \mathbf{v}_{t,i}$.

4. Aspectos Avançados

Combinações degeneradas

Ao apresentarmos a equação do operador de mutação diferencial, ver (7.3), apenas comentamos que os índices r_1, r_2, r_3 são sorteados aleatoriamente para cada i. Entretanto, há uma probabilidade, mesmo que pequena, de que algum par dos índices i, r_1, r_2, r_3 em (7.3) seja coincidente. Nesta seção discutimos esses casos degenerados e procedimentos que podem ser adotados para evitá-los.

As seguintes situações degeneradas podem ocorrer:

1. $r_2 = r_3$. Nesse caso, (7.3) se reduz a:
$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1}$$
e não há perturbação aplicada ao vetor de base. O vetor de base não sofre mutação, causando a recombinação de $\mathbf{x}_{t,i}$ com algum vetor escolhido aleatoriamente da população.

2. $r_1 = r_2$ ou $r_1 = r_3$. Nesse caso a mutação diferencial se reduz a um operador de recombinação aritmética[2]. Para $r_1 = r_2$, temos:
$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_1} - \mathbf{x}_{t,r_3}\right) = (1+F)\mathbf{x}_{t,r_1} - F\mathbf{x}_{t,r_3}$$
Ocorre uma recombinação degenerada, em que a nova solução é gerada externamente ao segmento que une $\mathbf{x}_{t,r_1}$ e $\mathbf{x}_{t,r_3}$, ver Figura 7.12. Para $r_1 = r_3$, ocorre uma recombinação aritmética tradicional entre $\mathbf{x}_{t,r_1}$ e $\mathbf{x}_{t,r_2}$:
$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_1}\right) = (1-F)\mathbf{x}_{t,r_1} + F\mathbf{x}_{t,r_2}$$
se $F \in [0,1]$, ocorrendo cruzamentos degenerados para valores de $F > 1$.

3. $i = r_1$. Uma perturbação é aplicada a $\mathbf{x}_{t,i}$ de acordo com:
$$\mathbf{v}_{t,i} = \mathbf{x}_{t,i} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$$
A solução teste é o resultado da recombinação da solução $\mathbf{x}_{t,i}$ com sua versão perturbada. Na prática, o parâmetro C passa a ter o significado de um parâmetro de mutação, controlando quantas variáveis de $\mathbf{x}_{t,i}$ serão perturbadas.

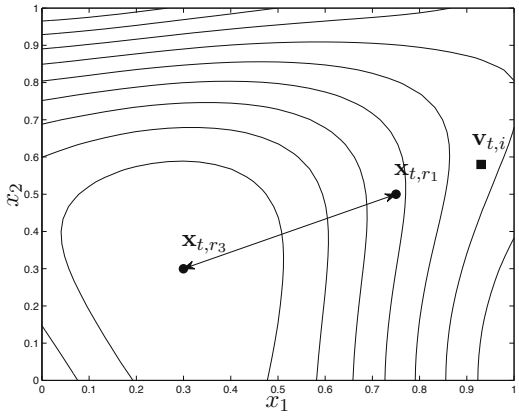
4. $i = r_2$ ou $i = r_3$. Nesse caso o vetor-diferença está na direção que liga $\mathbf{x}_{t,i}$ a algum outro vetor da população, $\mathbf{x}_{t,r_2}$ ou $\mathbf{x}_{t,r_3}$. Essa combinação não é necessariamente indesejável.

Estas combinações degeneradas, mesmo que tenham baixa probabilidade de ocorrência, são em geral indesejáveis e podem prejudicar o desempenho do algoritmo. Por essa razão, recomenda-se adotar índices mutuamente distintos, isto é, $i \neq r_1 \neq r_2 \neq r_3$, em implementações mais práticas do algoritmo de evolução diferencial. Uma maneira simples de garantir essa condição é substituir a linha 5 no Algoritmo 1 pelo código no Algoritmo 2 a seguir.

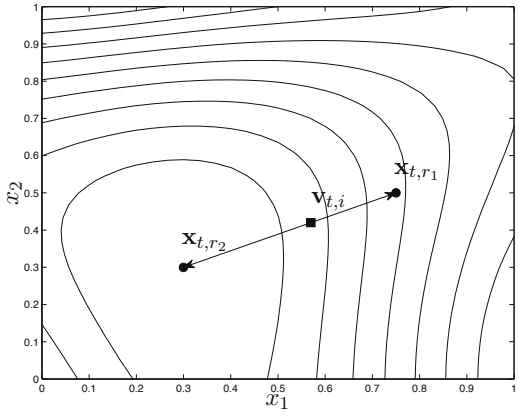
Algoritmo 2 Pseudocódigo para a seleção de índices mutuamente distintos

```
1: repita
2:     Selecione aleatoriamente r1 ∈ {1, ..., N}
3: até r1 ≠ i
4: repita
5:     Selecione aleatoriamente r2 ∈ {1, ..., N}
6: até r2 ≠ i ∧ r2 ≠ r1
7: repita
8:     Selecione aleatoriamente r3 ∈ {1, ..., N}
9: até r3 ≠ i ∧ r3 ≠ r1 ∧ r3 ≠ r2
```

[2] A recombinação aritmética de dois vetores $\mathbf{x}_a$ e $\mathbf{x}_b$ é definida pela combinação convexa $\lambda\mathbf{x}_a + (1-\lambda)\mathbf{x}_b$, para $\lambda \in [0,1]$. Por essa razão, a recombinação aritmética é também conhecida na literatura como recombinação convexa ou recombinação intermediária. Para o caso particular $\lambda = 0.5$, a recombinação é denominada recombinação média. No contexto de algoritmos genéticos, a operação de recombinação é tradicionalmente denominada cruzamento.



(a)



(b)

Figura 7.12: (a) Quando $r_1 = r_2$, ocorre uma recombinação degenerada, em que a nova solução é gerada fora do segmento que une as duas soluções recombinantes. (b) Quando $r_1 = r_3$, ocorre uma recombinação aritmética tradicional. Nos exemplos acima, $F = 0.4$.

Outra maneira de implementar a escolha de índices distintos é utilizar uma rotina de permutação aleatória de um vetor de inteiros, se disponível. Nesse caso, pode-se utilizar essa rotina para gerar uma permutação aleatória do vetor de índices $(1, \ldots, N)$. Em seguida, utilize os três primeiros elementos do vetor resultante como valores para os índices r_1, r_2 e r_3. Esse procedimento assegura $r_1 \neq r_2 \neq r_3$. Para garantir a condição $i \neq r_1 \neq r_2 \neq r_3$, procure o índice i no vetor resultante da permutação aleatória e use os três elementos seguintes como valores para os índices r_1, r_2 e r_3. Nesse caso, deve-se considerar uma contagem circular para as posições do vetor, isto é, ao se chegar à última posição do vetor, a posição seguinte representa um retorno à primeira posição.

Variações do algoritmo

O algoritmo de evolução diferencial em sua versão básica utiliza seleção aleatória com probabilidade uniforme do vetor de base, um vetor-diferença para a mutação, e recombinação discreta entre a solução corrente e seu vetor mutante correspondente. Entretanto, muitas variações desse esquema básico são

possíveis. Nesta seção, comentaremos sobre algumas dessas variações.

Com relação ao vetor de base, este pode ser escolhido aleatoriamente entre os indivíduos da população corrente com probabilidade uniforme ou com probabilidade proporcional à qualidade de cada solução[3]. Ao usar uma seleção com probabilidade uniforme, de fato o algoritmo de evolução diferencial está eliminando a pressão seletiva para a reprodução, uma vez que cada indivíduo possui a mesma probabilidade de ser selecionado como vetor de base, portanto, cada indivíduo produz em média um descendente. Essa forma de seleção corresponde a uma seleção por roleta estocástica em que cada indivíduo ocupa uma área igual na roleta.

A seleção uniforme pode entretanto causar repetição do vetor de base e fazer com que algumas soluções na população não sejam usadas como vetor de base. Para garantir a seleção de um vetor de base único para cada indivíduo, pode-se usar permutação aleatória de um vetor de índices $(1, \ldots, N)$. Dessa forma garante-se que cada indivíduo da população será selecionado uma única vez como vetor de base na mutação diferencial, portanto, cada indivíduo produz um único descendente, um único vetor mutante $\mathbf{v}_{t,i}$. Usando a analogia com a seleção por roleta, essa estratégia de seleção dos vetores de base equivale a um giro de uma roleta com N ponteiros igualmente espaçados, ao invés de N giros de uma roleta com um ponteiro, como na situação anterior. Como cada indivíduo ocupa uma área igual na roleta, cada indivíduo é selecionado uma única vez.

Pode-se ainda utilizar um mesmo vetor como vetor de base em todas as operações de mutação diferencial. A escolha mais comum é usar a melhor solução na população como vetor de base. Assim, temos:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,best} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right) \tag{7.9}$$

em que $\mathbf{x}_{t,best}$ representa o melhor indivíduo na população na geração t.

Pode-se ainda adotar o vetor correspondente à média da distribuição espacial da população como vetor base. As perturbações geradas pela mutação diferencial são aplicadas à média da população:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,mean} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right) \tag{7.10}$$

com:

$$\mathbf{x}_{t,mean} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_{t,i} \tag{7.11}$$

Utilizar um único vetor de base significa adotar uma pressão seletiva forte na reprodução, em contraste à ausência de pressão seletiva da seleção uniforme. Em geral, essa estratégia apresenta uma maior velocidade de convergência e uma rápida redução de diversidade, que podem levar à convergência prematura em alguns problemas, em particular em problemas em que a função-objetivo é multimodal. O algoritmo de evolução diferencial já possui uma pressão seletiva forte na sobrevivência dos indivíduos, em que ocorre uma competição determinística entre o indivíduo $\mathbf{x}_{t,i}$ e a solução teste $\mathbf{u}_{t,i}$. Portanto, se a pressão seletiva para reprodução também for forte, pode ocorrer uma rápida perda de diversidade na população. Na versão clássica do algoritmo, a ausência de pressão seletiva para a reprodução compensa a pressão seletiva forte na etapa de sobrevivência de maneira similar ao que ocorre com as *estratégias evolutivas*, ver Capítulo XX.

Outra possibilidade é gerar soluções na direção do melhor indivíduo para servirem como vetores de base. A equação a seguir ilustra essa abordagem:

$$\mathbf{v}_{t,i} = \underbrace{\mathbf{x}_{t,i} + \lambda\left(\mathbf{x}_{t,best} - \mathbf{x}_{t,i}\right)}_{\mathbf{x}_{t,base}} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right) \tag{7.12}$$

[3] Esquema de seleção proporcional à aptidão muito utilizado em algoritmos genéticos.

com $\lambda \in [0,1]$. Nesse caso, ocorre uma recombinação aritmética entre $\mathbf{x}_{t,i}$ e $\mathbf{x}_{t,best}$. O vetor de base corresponde a um ponto gerado aleatoriamente sobre o segmento que liga $\mathbf{x}_{t,i}$ e $\mathbf{x}_{t,best}$. Essa forma de seleção do vetor de base apresenta menor pressão seletiva do que em (7.9), embora haja uma polarização na direção da melhor solução.

A mutação diferencial pode ser generalizada para empregar mais vetores-diferença na criação do vetor mutante:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + \sum_{k=1}^{d} F_k \Delta \mathbf{x}_{t,k} \tag{7.13}$$

em que a perturbação aplicada ao vetor de base é composta pela soma de d vetores-diferença da forma:

$$\Delta \mathbf{x}_{t,k} = \left(\mathbf{x}_{t,r_{k+1}} - \mathbf{x}_{t,r_{k+1+d}}\right) \tag{7.14}$$

Por exemplo, usando o mesmo valor de F_k para todo k e $d = 3$, a equação (7.13) fica:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_5} + \mathbf{x}_{t,r_3} - \mathbf{x}_{t,r_6} + \mathbf{x}_{t,r_4} - \mathbf{x}_{t,r_7}\right) \tag{7.15}$$

O uso de mais de um vetor-diferença na mutação diferencial aumenta a capacidade de geração de diversidade no algoritmo. Contudo, essa estratégia reduz a capacidade de alinhamento dos vetores-diferença com o contorno da função, embora o alinhamento não seja totalmente perdido. Além disso, fica cada vez mais difícil garantir índices mutuamente distintos. O uso de mais vetores-diferença aumenta a busca global no algoritmo e prejudica a busca local, porque os tamanhos de passo adaptados se somam produzindo perturbações de maior magnitude. Por essa razão, o valor de F deve ser menor para compensar o aumento no tamanho das perturbações.

Existe uma notação sintética para representar as variações do algoritmo de evolução diferencial. Essa notação padrão segue o formato `DE/base/d/rec`. O termo usado no lugar de `base` indica a forma como o vetor de base é selecionado, `d` indica o número de vetores-diferença usados, e o termo usado no lugar de `rec` faz referência ao operador de recombinação utilizado.

Por exemplo, a versão clássica do algoritmo de evolução diferencial, ver Algoritmo 1, pode ser representada pela notação `DE/rand/1/bin`, em que `rand` indica que o vetor base na mutação diferencial é escolhido aleatoriamente com probabilidade uniforme, $d = 1$ indica que apenas um vetor-diferença é usado, e `bin` indica o método de recombinação. O termo `bin` faz referência à distribuição binomial, pois a recombinação discreta com probabilidade C faz com que o número de valores herdados de $\mathbf{v}_{t,i}$ siga uma distribuição binomial. A probabilidade de que p valores sejam herdados do vetor mutante é dada por:

$$P\{X = p\} = \binom{n}{p} C^p (1-C)^{n-p} \tag{7.16}$$

que corresponde a todas as combinações de p sucessos e $n-p$ falhas, isto é, p ocorrências de $\mathcal{U}_{[0,1]}$ que foram menores do que C e $n-p$ ocorrências de $\mathcal{U}_{[0,1]}$ que foram maiores do que C.

Existe uma forma alternativa de implementação da recombinação discreta, que produz uma distribuição exponencial do número de valores herdados da solução mutante. Na recombinação discreta binomial uma realização de $\mathcal{U}_{[0,1]}$ é obtida para cada coordenada e o valor de $u_{t,i,j}$ é copiado de $v_{t,i,j}$ se $\mathcal{U}_{[0,1]} \leq C$ para a coordenada j, isto é, cada coordenada é testada de maneira independente. Na recombinação discreta exponencial, indicada por `exp` na notação padrão, uma posição no vetor é sorteada aleatoriamente com probabilidade uniforme. A partir dessa posição, os valores da solução teste são herdados da solução mutante *enquanto* $\mathcal{U}_{[0,1]} \leq C$. Na primeira ocorrência de $\mathcal{U}_{[0,1]} > C$, os valores para as coordenadas restantes são obtidos de $\mathbf{x}_{t,i}$. Observe que nesse caso a probabilidade de que p valores sejam herdados do vetor mutante é dada por:

$$P\{X = p\} = C^p(1-C) = C^p - C^{p+1} \tag{7.17}$$

que corresponde a p ocorrências *sucessivas* de $\mathcal{U}_{[0,1]}$ que foram menores do que C e uma ocorrência de $\mathcal{U}_{[0,1]}$ que foi maior do que C.

Na recombinação discreta do tipo exponencial, as primeiras coordenadas do vetor teste a partir da posição sorteada possuem maior probabilidade de serem herdadas do vetor mutante do que as últimas coordenadas. Em outras palavras, os valores das coordenadas em posições adjacentes têm maior probabilidade de permanecerem juntas no vetor teste do que os valores em posições não adjacentes. Na terminologia usada em algoritmos evolutivos, dizemos que esse operador apresenta *polarização posicional*, ou seja, uma polarização na probabilidade de que algumas posições do genótipo sejam escolhidas em detrimento de outras. Para contornar a polarização posicional desse tipo de recombinação, os índices das coordenadas podem ser embaralhados aleatoriamente antes de aplicar a recombinação.

Para concluir esta seção, a Tabela 7.1 apresenta alguns exemplos de instâncias do algoritmo de evolução diferencial, cada uma com sua notação padrão correspondente. Se o melhor indivíduo for usado como vetor de base, e utilizarmos três vetores-diferença na mutação diferencial, podemos referenciar o algoritmo de evolução diferencial correspondente como `DE/best/3/bin`. O algoritmo de evolução diferencial que utiliza a abordagem de seleção do vetor de base em (7.12) e recombinação discreta exponencial pode ser representado pela notação `DE/current-to-best/1/exp`.

Notação	Mutação diferencial
`DE/rand/1/bin`	$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$
`DE/best/1/bin`	$\mathbf{v}_{t,i} = \mathbf{x}_{t,best} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$
`DE/mean/1/bin`	$\mathbf{v}_{t,i} = \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_{t,k} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$
`DE/rand-to-best/1/bin`	$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + \lambda\left(\mathbf{x}_{t,best} - \mathbf{x}_{t,r_1}\right) + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$
`DE/current-to-best/1/bin`	$\mathbf{v}_{t,i} = \mathbf{x}_{t,i} + \lambda\left(\mathbf{x}_{t,best} - \mathbf{x}_{t,i}\right) + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)$
`DE/rand/2/bin`	$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + F_1\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_4}\right) + F_2\left(\mathbf{x}_{t,r_3} - \mathbf{x}_{t,r_5}\right)$

Tabela 7.1: Algumas instâncias do algoritmo de evolução diferencial.

Variação do parâmetro de escala

O parâmetro de escala F do vetor-diferença não necessariamente precisa variar, pois os tamanhos de passo são autoadaptados na distribuição de vetores-diferença, conforme vimos anteriormente. À medida que o algoritmo progride, e a população se agrupa em torno de um ponto de mínimo local, os tamanhos de passo automaticamente diminuem devido à proximidade das soluções no domínio de busca. Dessa forma, o parâmetro de escala pode ser mantido fixo no algoritmo de evolução diferencial sem comprometer seu desempenho significativamente.

Entretanto, variar o valor de F dentro de alguma faixa contínua de valores aumenta a diversidade de vetores-diferença e de soluções possíveis que podem ser geradas com o operador. Além disso, variar o parâmetro de escala reduz as chances de estagnação do algoritmo. A estagnação no algoritmo de evolução diferencial pode ocorrer com a escolha de um valor fixo para F em situações degeneradas, quando a população está presa numa dada distribuição espacial que não oferece vetores-diferença que permitam a geração de soluções melhores. Variando o valor de F, a possibilidade de estagnação diminui.

Uma maneira simples de variar o valor de F é fazer:

$$F = \mathcal{U}_{[a,b]} \tag{7.18}$$

Tipicamente, adota-se $\mathcal{U}_{[0.5,1]}$. Valores muito pequenos de F prejudicam a convergência pois reduzem muito o efeito da mutação diferencial, causando perturbações pequenas nos vetores de base. Por outro lado, valores de F maiores do que 1 desaceleram a redução natural dos tamanhos de passo, retardando a convergência do algoritmo. Observe que F pode variar em uma faixa relativamente pequena de valores, uma vez que os tamanhos de passo são adaptados no algoritmo de evolução diferencial de acordo com a distribuição espacial da população. Por exemplo, Zaharie (2002) indica o valor 0.3 como um limite inferior confiável para F. Outros estudos empíricos indicam um limite inferior para F de 0.4 (Ali e Törn, 2000; Gamperle et al., 2002). Até o momento, estudos empíricos na literatura indicaram que não há benefícios em se utilizar valores superiores a 1 para F.

Alguns autores estudaram o emprego de parâmetros de escala independentes para cada variável, fazendo a mutação diferencial da forma:

$$v_{t,i,j} = x_{t,r_1,j} + F_j \left(x_{t,r_2,j} - x_{t,r_3,j}\right) \tag{7.19}$$

ou, escrito na forma matricial:

$$\mathbf{v}_{t,i} = \mathbf{x}_{t,r_1} + \mathbf{F} \left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right) \tag{7.20}$$

com:

$$\mathbf{F} = \begin{bmatrix} F_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & F_n \end{bmatrix} \tag{7.21}$$

O parâmetro de escala F_j é retirado de uma distribuição aleatória independente para cada variável. Contudo, essa escolha se mostrou desvantajosa, porque utilizar valores independentes de F_j descorrelaciona as componentes do vetor-diferença, destruindo a correlação linear entre as variáveis aprendida pelo operador de mutação diferencial e o alinhamento das orientações dos vetores-diferença.

Entretanto, pequenas variações nos valores de F_j em torno de um valor central representam pequenas variações de ângulo nos vetores-diferença e podem ser interessantes para aumentar a diversidade de soluções geradas sem prejudicar gravemente o alinhamento dos vetores-diferença. Para implementar essa abordagem, pode-se utilizar:

$$F_j = F_0 + \mathcal{U}_{[-\alpha,\alpha]}, \quad j \in \{1, \ldots, n\} \tag{7.22}$$

ou ainda

$$F_j = F_0 + \mathcal{N}_{[0,\alpha]}, \quad j \in \{1, \ldots, n\} \tag{7.23}$$

com $\alpha \ll 1$ em ambos os casos. O valor de F_0 é fixo para todo j, mas pode variar para cada i, usando por exemplo (7.18).

Outros operadores de recombinação

Como discutido anteriormente, a recombinação discreta, seja ela do tipo binomial ou exponencial, não é invariante à rotação do sistema de coordenadas. Além disso, como somente algumas coordenadas são herdadas do vetor mutante, o vetor teste produzido é algum ponto obtido com passos ortogonais a partir da solução mutante, veja Figura 7.1 novamente.

A Figura 7.13 ilustra uma função multimodal com variáveis correlacionadas de tal forma que a bacia de atração $\mathcal{B}_2$ é alcançável a partir da bacia de atração $\mathcal{B}_1$ na Figura por meio de um movimento correlacionado em ambas as coordenadas. A recombinação discreta pode causar movimentos ortogonais em que apenas uma variável é modificada. Um movimento em uma única coordenada pode não alcançar

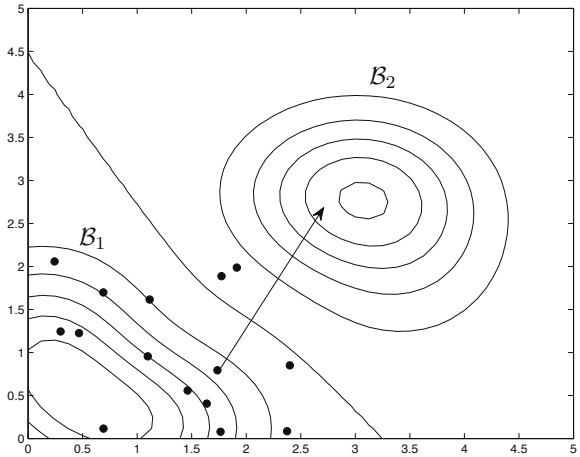


Figura 7.13: Função multimodal com variáveis correlacionadas.

a bacia de atração $\mathcal{B}_2$ e não ser aceito. Dessa forma, a bacia $\mathcal{B}_2$ não seria alcançável, a menos que um movimento seja executado em ambas as coordenadas simultaneamente e na direção adequada.

Em uma função convexa com variáveis correlacionadas, como aquela ilustrada na Figura 7.2, passos ortogonais diminuem a velocidade de convergência que poderia ser obtida com passos correlacionados. Em problemas multimodais, é interessante que os movimentos sejam executados simultaneamente em muitas ou em todas as variáveis. Por essa razão, em problemas convexos com variáveis correlacionadas e em problemas multimodais, o algoritmo de evolução diferencial usando recombinação discreta apresenta melhor desempenho quando C é próximo de 1. Para valores próximos de 1, vários valores da solução mutante são aceitos simultaneamente, permitindo movimentos correlacionados.

Outra forma de contornar o problema, é substituir a recombinação discreta por outros operadores de recombinação, incluindo operadores que executem passos em várias ou em todas as coordenadas simultaneamente e que sejam invariantes à rotação do sistema de coordenadas.

Um operador de recombinação simples que satisfaz esses requisitos é a recombinação aritmética, muito utilizada em algoritmos genéticos e estratégias evolutivas. A solução teste é obtida a partir de:

$$\mathbf{u}_{t,i} = \lambda \mathbf{x}_{t,i} + (1-\lambda)\mathbf{v}_{t,i}, \quad \lambda \in [0,1] \tag{7.24}$$

sempre que $\mathcal{U}_{[0,1]} \leq C$. Dessa forma, o parâmetro C passa a ter o significado de um parâmetro de taxa de recombinação da população atual X_t com a população mutante V_t.

A equação (7.24) pode ser combinada com (7.3) obtendo:

$$\mathbf{u}_{t,i} = \lambda \mathbf{x}_{t,i} + (1-\lambda)\left[\mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)\right], \quad \lambda \in [0,1] \tag{7.25}$$

que combina a mutação diferencial e a recombinação numa única equação.

Novamente, podemos escrever a equação para a geração do vetor teste na forma compacta a seguir:

$$\mathbf{u}_{t,i} = \begin{cases} \lambda \mathbf{x}_{t,i} + (1-\lambda)\left[\mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right)\right], & \text{se } \mathcal{U}_{[0,1]} \leq C \\ \mathbf{x}_{t,r_1} + F\left(\mathbf{x}_{t,r_2} - \mathbf{x}_{t,r_3}\right), & \text{caso contrário} \end{cases} \tag{7.26}$$

em que a seleção para sobrevivência é feita como antes, ver (7.6).

Além da recombinação aritmética, outros operadores de recombinação propostos na literatura podem ser utilizados, veja (Gwiazda, 2006) para uma referência bastante ampla de operadores de recombinação para problemas de otimização numérica.

Esquema geral dos algoritmos de evolução diferencial

Vamos concluir esta seção com a apresentação de um esquema geral para as variações do algoritmo de evolução diferencial. Esse esquema geral é apresentado no Algoritmo 3.

Algoritmo 3 Pseudocódigo geral para a família de algoritmos de evolução diferencial

1: $t \leftarrow 1$
2: Inicializar população $X_t = \{\mathbf{x}_{t,i};\ i = 1, \ldots, N\}$
3: **enquanto** algum critério de parada não for satisfeito **faça**
4: **para** $i = 1$ até N **faça**
5: Selecionar vetor de base $\mathbf{x}_{t,base}$ [Seleção para reprodução]
6: Selecionar conjunto de vetores-diferença $\{\Delta\mathbf{x}_{t,k}\}$, $k = 1, \ldots, d$
7: Selecionar fatores de escala F_k, $k = 1, \ldots, d$
8: Gerar solução mutante usando $\mathbf{v}_{t,i} = \mathbf{x}_{t,base} + \sum_{k=1}^{d} F_k \Delta\mathbf{x}_{t,k}$
9: Adicionar $\mathbf{v}_{t,i}$ à população mutante V_t
10: **fim para**
11: Recombinar X_t e V_t, com parâmetro C, gerando U_t
12: Aplicar seleção para sobrevivência entre X_t e U_t
13: $t \leftarrow t + 1$
14: **fim enquanto**

Na linha 5 do Algoritmo 3, ocorre a seleção do vetor de base, que representa a seleção para reprodução no algoritmo de evolução diferencial. Nas duas linhas seguintes, são selecionados d vetores-diferença e os valores para os parâmetros de escala de cada vetor-diferença, que serão usados na geração do vetor mutante na linha 8.

O laço nas linhas 4 a 10 está relacionado somente à operação de mutação diferencial para produção da população mutante V_t. Após a operação de mutação diferencial, a recombinação é executada na linha 11. Finalmente, deve-se definir um mecanismo de seleção para sobrevivência entre X_t e U_t, formando a população X_t da próxima geração.

Apresentado dessa maneira, podemos observar que a mutação diferencial é um operador de busca que tem um papel primário no algoritmo, enquanto a recombinação tem um papel secundário.

O algoritmo de evolução diferencial básico apresentado no Algoritmo 1 e as variações discutidas neste capítulo podem ser vistos como instâncias particulares do Algoritmo 3, mais geral.

5. Conclusões

Este capítulo apresentou os algoritmos baseados em evolução diferencial para otimização de funções contínuas. Ultimamente, a evolução diferencial tem recebido bastante destaque no contexto da otimização não linear com variáveis contínuas, devido às suas características de versatilidade, robustez e autoadaptação, colocando-o entre os algoritmos evolutivos mais eficientes nesse contexto.

O algoritmo de evolução diferencial possui muitas qualidades desejáveis em algoritmos de otimização de propósito geral: (i) capacidade de adaptar-se à estrutura da função, aprendendo correlações lineares entre as variáveis do problema a partir da distribuição espacial da população; (ii) invariância à rotação e translação do sistema de coordenadas; (iii) poucos parâmetros de controle a serem ajustados, sendo que o parâmetro de escala não precisa ser necessariamente ajustado, uma vez que os tamanhos de passo da mutação são autoadaptados; e (iv) simplicidade de implementação.

Embora o algoritmo de evolução diferencial tenha sido desenvolvido para tratar problemas de otimização não linear contínua, pode-se encontrar na literatura, em especial nos últimos anos, alguns trabalhos que visam adaptar o algoritmo para problemas de otimização combinatória. Por exemplo,

Qian et al. (2008) apresenta uma versão do algoritmo de evolução diferencial para problemas de planejamento e escalonamento da produção. O leitor interessado também pode consultar o recente livro editado por Onwubolu e Davendra (2009), que discute várias abordagens de utilização da evolução diferencial em problemas de otimização baseados em vetores de permutações.

Agradecimentos

O autor gostaria de expressar sua gratidão ao aluno de doutorado Lucas de Souza Batista e aos colegas Felipe Campelo e Jaime Arturo Ramírez pelos comentários e sugestões que contribuiram para o formato final deste texto.