

Selbstadaptive Mutation bei evolutionären Strategien

Stefan Etschberger

Otto Opitz

Institut für Statistik und mathematische Wirtschaftstheorie
Universität Augsburg, 86135 Augsburg, BRD

Abstract: Die Forschungsaktivitäten auf dem Gebiet evolutionärer Algorithmen haben in den letzten Jahren sprunghaft zugenommen. Vor allem auf dem Gebiet der Selbstadaptation von Strategieparametern können Fortschritte verzeichnet werden. In dieser Arbeit werden nach einer allgemeinen Einführung beispielhaft verschiedene Varianten vorgestellt, den Mutationsoperator bei evolutionären Strategien selbstadaptiv zu steuern. Anhand eines neu entwickelten und frei verfügbaren Softwaremoduls werden drei Varianten verglichen.

1 Einführung und Überblick

Evolutionäre Algorithmen sind globale, parallele und stochastische Optimierungsverfahren, die sich an Beobachtungen biologischer Auswahl- und Variationsmechanismen orientieren und als „heuristische Schweizer Taschenmesser“ in vielen Bereichen zur Anwendung kommen. Seit der erstmaligen Beschreibung solcher Systeme ist das Interesse und damit die Anzahl wissenschaftlicher Studien in diesem Bereich sprunghaft angestiegen (siehe Tabelle 1). Nach einem kurzen Überblick über Grundlagen und Basisrichtungen evolutionärer Algorithmen (Abschnitt 2) wird in dieser Arbeit das Problem der Selbstadaptation strategischer Parameter bei evolutionären Algorithmen behandelt (Abschnitt 3). Neben den klassischen Ansätzen von globalen Schrittweitenparametern, individuellen Schrittweiten und korrelierten Mutationen werden die neueren Methoden der Kovarianz Adaption, der Kumulation und Evolutionspfade, sowie die Kombination die-

Zeitraum	Anzahl	Zeitraum	Anzahl
-1960	7	1981-1985	100
1961-1965	13	1986-1990	520
1966-1970	21	1991-1995	3400
1971-1975	38	1996- ca. 2000	6000
1976-1980	50		

Tabelle 1: Größenordnung der Anzahl von Veröffentlichungen über evolutionäre Algorithmen (Daten aus [Ala03])

ser Ansätze kurz vorgestellt. Abschließend wird die tendenzielle Überlegenheit neuerer Ansätze beispielhaft gezeigt.

2 Grundlagen und Basisrichtungen

Evolutionäre Algorithmen gehen von einer Menge von potentiellen Lösungen für ein vorgegebenes Problem aus, die zu einer **Population** $P = \{i_1, \dots, i_n\}$ zusammengefasst werden. Jedes **Individuum** $i_k \in P$ wird bezüglich seiner Lösungsgüte $\Phi(i_k)$, die als **Fitness** bezeichnet wird, evaluiert. Die Kernidee evolutionärer Algorithmen besteht darin, die jeweils aktuelle Population P_t in eine neue, möglichst höher bewertete Population P_{t+1} zu überführen. Dabei sind die biologischen Phänomene Selektion, Replikation und Mutation geeignet über stochastische Operatoren zu modellieren und schrittweise anzuwenden. In der Phase der **Selektion** werden **Eltern** als Individuen der aktuellen Population ausgewählt, die auf Grund ihrer Fitness geeignet sind, ihre Eigenschaften auf die Folgepopulation oder nächste **Generation** zu übertragen. Durch verschiedene Formen der **Replikation** werden Individuen vermischt und auf diese Weise zu neuen Individuen oder **Nachkommen** geformt. Schließlich soll der **Mutationsoperator** potentiell jedes einzelne Individuum verändern und damit die evolutionäre Variation simulieren. Die Basisvariante eines evolutionären Algorithmus kann wie in Abbildung 1 dargestellt werden.

1. Wahl der erforderlichen Strategieparameter
2. Generierung der Ausgangspopulation P_0
3. Setze $t := 0$
4. Wiederholung von (a) - (f), bis ein Abbruchkriterium erfüllt ist
 - (a) Setze $t := t + 1$
 - (b) Berechnung von $\Phi(i_k) \quad \forall i_k \in P_{t-1}$
 - (c) Selektion der Eltern aus P_{t-1}
 - (d) Replikation der Nachkommen
 - (e) Mutation der Nachkommen
 - (f) Generierung von P_t aus e)

Abbildung 1: Evolutionärer Basisalgorithmus

Bedingt durch verschiedene Forschungsrichtungen werden evolutionäre Algorithmen in nachfolgende Klassen unterteilt:

Genetische Algorithmen gehen auf Arbeiten von Holland [Hol75] zurück. Potentielle Lösungen werden meist binär kodiert, also $i_k \in \{1, 0\}^L$ mit $L \in \mathbb{N}$, und übernehmen die Rolle von Individuen einer Population. Die Replikation hat die zentrale Funktion bei genetischen Algorithmen. So versucht man mit einem geschickten Crossover-Operator,

schwächer bewertete Eltern tendenziell in höherwertige Nachkommen zu überführen. Nach Hollands Schema-Theorem und der „*building block*“-Hypothese verbreiten sich tendenziell erfolgreiche Teilmuster eines Individuums in der Population, wodurch sich die durchschnittliche Fitness der Population (z.B. in [Gol89], S. 28-33, oder [Hol75], S. 102 f.) erhöht. Kritiker führen allerdings an, dass die Erhöhung der durchschnittlichen Fitness nicht dem eigentlichen Ziel der Optimierungsaufgabe entspricht.

Als Modifikation zu genetischen Algorithmen befasst sich die **Genetische Programmierung** ursprünglich mit der automatischen Entwicklung von Programmen (z.B. in [Koz92]). Im Unterschied zur reinen Form genetischer Algorithmen besteht die Population hier aus einer Menge von Programmen als den Individuen, die aus problemadäquaten elementaren Funktionen, Variablen und Konstanten zusammengesetzt sind und in einer Baumstruktur kodiert werden. Dabei kann jedes Individuum oder Programm beispielsweise einen Datensatz bzgl. eines Zielwertes evaluieren. Die Fitness errechnet sich dann aus der Abweichung der Programmresultate zum errechneten Zielwert. Alternative Methoden, die Fitnessfunktionen bei genetischer Programmierung zu modellieren findet man z.B. in [GS95].

Evolutionäre Strategien wurden durch Rechenberg und Schwefel [Rec73] eingeführt. Potentielle Lösungen einer vorliegenden Optimierungsaufgabe werden als reelle Vektoren $x \in \mathbb{R}^n$ dargestellt, die dann den Individuen einer Population entsprechen. Die Fitnessfunktion ist hier von der Form $F : \mathbb{R}^n \rightarrow \mathbb{R}$. Nachkommen einer Generation entstehen abweichend vom Crossover bei genetischen Algorithmen durch diskrete oder intermediäre Rekombination von Eltern mit anschließender Mutation, bei der jede rekombinierte Lösungsvariante über einen Zufallsmechanismus komponentenweise erhöht oder erniedrigt wird. Traditionell spielen selbstadaptive Mutationsoperatoren bei evolutionären Strategien eine zentrale Rolle. Darauf wird in Abschnitt 3 genauer eingegangen.

Die **Evolutionäre Programmierung** hat ihren Ursprung in der Erforschung künstlich-intelligenter Automaten [FOW66]. Dabei werden mittels simulierter Evolution endliche Automaten („*finite state machines*“) generiert. Bei der evolutionären Programmierung geht man davon aus, dass die Anwendung von Operatoren auf der Kodierungsebene der Individuen die Fitness nur unzureichend verbessert. Deshalb wird die Mutation als einziger Suchoperator verwendet, der hier direkt auf der Verhaltensebene der Individuen wirkt. Die Verwandtschaft zu evolutionären Strategien ist insbesondere durch die Verwendung selbstadaptiver Techniken gekennzeichnet.

Die Grenzen zwischen den genannten Kategorien evolutionärer Algorithmen verwischen sich in neuester Zeit. Aktuell findet man beispielsweise Arbeiten zu genetischen Algorithmen, in denen die Wichtigkeit der Mutation betont wird ([TS93] oder [BFN96]). Viel Energie wird auch auf die Erforschung der Modellierung selbstadaptiver Strategieparameter nicht nur bei evolutionären Strategien verwendet. Genetische Algorithmen mit reell kodierten Individuen werden ebenso untersucht wie evolutionäre Strategien unter Einbeziehung der Replikation [GAB⁺02].

Mit einer modularen Sichtweise auf alle Strömungen und Teilaspekte könnte ein spezielles Problemlösungsverfahren die Stärken aller Varianten nutzen, wenn es gelingt, die geeigneten Bausteine aus den Elementen Modelle der Populationsbildung, Kodierung der

Individuen, Behandlung von Nebenbedingungen und Operatoren für die Selektion, Replikation und Mutation auszuwählen und sinnvoll zu kombinieren ([Nis97], S. 237 ff).

Im Folgenden wird auf die Selbstadaption strategischer Parameter näher eingegangen.

3 Selbstadaption strategischer Parameter

Eine Reihe von Arbeiten beschäftigt sich mit optimalen statischen Werten für strategische Parameter. Unter strategischen Parametern können Zahlenwerte bestimmender Größen des Algorithmus aber auch die Ausgestaltung von Regeln für die Akzeptanz schlechterer Lösungen verstanden werden (wie z.B. in [DS90]). Letztere Ansätze werden in diesem Beitrag nicht weiter behandelt. So liefern Jongs Experimente [Jon75] bei genetischen Algorithmen für die Mutationswahrscheinlichkeit jeder binären Individuumskomponente den Wert $p_M \approx \frac{1}{1000}$, Grefenstette [Gre86] empfiehlt auf der Grundlage von Simulationen $p_M \approx \frac{1}{100}$. Schaffer [Müh92] kommt in Abhängigkeit von der Populationsgröße m und der Dimension der Individuen n zum Ergebnis $p_M \approx \frac{1.7}{m \cdot \sqrt{n}}$, Mühlenbein [Müh92] zu $p_M \approx \frac{1}{n}$.

Mitchell [Mit02] stellt fest, dass statische Empfehlungen vor allem dann nicht pauschal anwendbar sind, wenn eine Fitnessfunktion ohne unimodalen Charakter, mit starkem Rauschen oder fehlender globaler Differenzierbarkeit zugrunde liegt. Heuristiken wie evolutionäre Algorithmen kommen aber oft gerade dann zur Anwendung, wenn analytisch wenig über die zu optimierende Funktion bekannt ist. Seit einiger Zeit wird unter anderem deshalb intensiv auf dem Feld der Selbstadaption dieser Strategieparameter geforscht (z.B. in [SB92], [GAB⁺02], S. 111-247, [Bäc92], [HB91]). Im Folgenden sollen einige Konzepte zur Selbstadaption des Mutationsoperators beispielhaft bei evolutionären Strategien betrachtet werden.

Geht man von einer evolutionären Strategie mit einer Repräsentation der Individuen in der k -ten Generation als $x^{(k)} \in \mathbb{R}^n$ und einer Fitnessfunktion $f : \mathbb{R}^n \rightarrow \mathbb{R}$ aus, wird der Mutationsoperator in evolutionären Strategien typischerweise als Addition eines normalverteilten Zufallsvektors z mit Mittelwert 0 und einer Kovarianzmatrix C realisiert.

$$x^{(k+1)} = x^{(k)} + z, \quad \text{mit } z \sim \mathcal{N}(0, C), \quad 0 \in \mathbb{R}^n, \quad C \in \mathbb{R}^{n \times n}$$

Die Parametrisierung dieser Normalverteilung steuert offensichtlich die Geschwindigkeit zur Verbesserung der Repräsentation bzgl. der Fitness. Bei selbstadaptiver Steuerung des Algorithmus wird die Kovarianzmatrix im Schritt $k = 0$ geeignet initialisiert und anschließend in jedem Iterationsschritt mittels unterschiedlicher Konzepte verändert.

Ein globaler Schrittweitenparameter: Im einfachsten Fall wird mit der Kovarianzmatrix $C = \sigma^2 \cdot E$ (mit $E \in \mathbb{R}^{n \times n}$ als Einheitsmatrix, $\sigma \in \mathbb{R}^+$) eine isotrope Normalverteilung gewählt. Damit ergibt sich ein globaler Schrittweitenparameter σ . Die Oberflächen gleicher Wahrscheinlichkeitsdichte sind dann Hypersphären, im zweidimensionalen Fall Kreise (Abbildung 2, links). Angewendet wird diese globale Standardabweichung bei Rechenbergs „1/5-Regel“ ([Rec73], [Sch81]), die er auf der Grundlage von Überlegungen

an einer sog. (1+1)-ES aufstellt, einer evolutionären Strategie mit einem Elter und einem Kind ohne Rekombination, bei der das fittere dieser beiden die nächste Population bildet. Nach n Mutationen wird der Anteil der Mutationen mit Fitnessverbesserung p_s unter den letzten $10n$ Mutationen berechnet und anschließend die Mutationsrate σ mit einem Faktor $\delta = 0.85$ nach folgender Regel angepasst:

$$\sigma = \begin{cases} \sigma \cdot \delta & , \text{ falls } p_s < 0.2 \\ \sigma & , \text{ falls } p_s = 0.2 \\ \sigma / \delta & , \text{ falls } p_s > 0.2 \end{cases}$$

Rechenberg untersuchte als Grundlage für diese Regel nur die spezielle Form der (1+1)-ES und konstruiert den Wert 0.2 als Quasi-Mittelwert aus zwei Resultaten, die auf der Analyse zweier Fitnessfunktionen basieren, nämlich

$$f_{\text{sphere}}(x) = \sum_{i=1}^n x_i^2, \text{ dem sog. Sphärenmodell und}$$

$$f_{\text{corridor}}(x) = \begin{cases} f(x_1) & \text{wenn } -b < x_i < b \text{ (} i = 2, \dots, n \text{)} \\ -\infty & \text{sonst} \end{cases}, \text{ dem Tunnelmodell.}$$

Schwefel [Sch81] kritisiert, dass die 1/5-Regel auf sehr speziellen Überlegungen basiert und bei der Parameteradaption auch nicht im klassischen Sinne mutiert bzw. selektiert wird. Vorgeschlagen wurde deshalb, die Parametrisierung der Normalverteilung als Bestandteil in die Kodierung der Individuen mit aufzunehmen und den Regeln des Mutationsoperators zu unterwerfen. Dabei sollten die Mutationsschrittweiten immer positiv bleiben und deren Wachstum in positiver wie in negativer Richtung mit der gleichen Wahrscheinlichkeit und dem gleichen Faktor erfolgen, um eine deterministische Drift zu vermeiden. Deswegen wird σ mit einer log-normalverteilten Zufallsvariable mutiert. Ein Individuum hat dann die Form $I = (x_1, \dots, x_n, \sigma)$ und die Mutation verläuft in zwei Schritten:

$$\sigma^{(k+1)} = \sigma^{(k)} \exp(\zeta) \quad \text{mit} \quad \zeta \sim \frac{1}{\sqrt{2n}} N(0, 1) \quad (1)$$

$$x_i^{(k+1)} = x_i^{(k)} + z_i \quad \text{mit} \quad z_i \sim \sigma^{(k+1)} \cdot N(0, 1) \quad (2)$$

Individuelle Schrittweiten für jede Koordinatenachse: Erweitert man das Konzept eines globalen Schrittweitenparameters auf n individuelle Schrittweitenparameter σ_i , wobei σ_i der i -ten Komponente eines Individuums zugeordnet wird, ergibt sich eine Kovarianzmatrix der Form $C = (c_{ij})$ mit $c_{ij} = \begin{cases} \sigma_i^2 & \text{wenn } i=j \\ 0 & \text{sonst} \end{cases}$. Die Oberflächen gleicher Wahrscheinlichkeitsdichten sind jetzt achsenparallele Hyperellipsoide (siehe Abbildung 2, Mitte). Nimmt man diese n Strategieparameter in die Kodierung der Individuen auf, ergibt sich $I = (x_1, \dots, x_n, \sigma_1, \dots, \sigma_n)$. Eine Mutation kann dann wie folgt durchgeführt

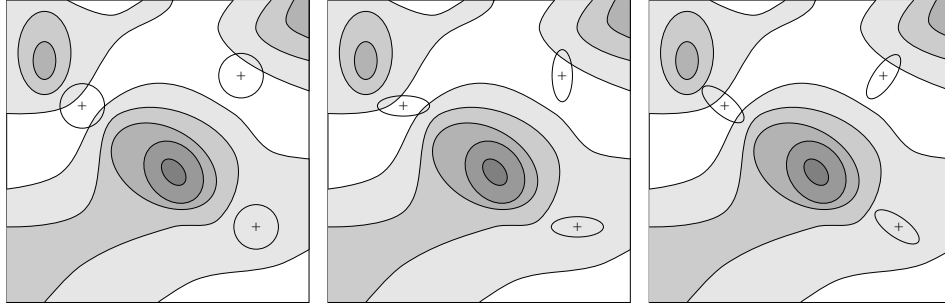


Abbildung 2: Fitnesslandschaft einer Funktion mit $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, mit Individuen (+) und der Linie gleicher Wahrscheinlichkeitsdichte der Mutationsschrittweite bei globalem Schrittwertenparameter (links), individuellen aber achsenparallelen Schrittwerten (Mitte) und beliebigen Normalverteilungen mit Mittelwert 0

werden [Bäc93]:

$$\sigma_i^{(k+1)} = \sigma_i^{(k)} \exp(\zeta + \zeta_i) \text{ mit } \zeta \sim \frac{1}{\sqrt{2n}} N(0, 1) \text{ und } \zeta_i \sim \frac{1}{\sqrt{2}\sqrt{n}} N(0, 1) \quad (3)$$

$$x_i^{(k+1)} = x_i^{(k)} + z_i \text{ mit } z_i \sim \sigma_i^{(k+1)} \cdot N(0, 1) \quad (4)$$

Hier bezeichnen ζ bzw. ζ_i die Ausprägungen von Zufallsvariablen, die für alle σ_i nur einmal bzw. für jede Komponente individuell bestimmt werden. Problematisch bei dieser individuellen Schrittwertenanpassung ist die Achsenparallelität der Hyperellipsoide. Beispielsweise können sich die Strategieparameter dieses Modells kaum an schmale, tiefe Gräben im Fitnessgebirge mit Neigungswinkeln $\alpha \neq 0 \bmod \frac{\pi}{2}$ zu den Koordinatenachsen anpassen.

Korrelierte Mutationen: Schwefel [Sch81] schlägt eine Erweiterung der individuellen Schrittwertenparameter auf beliebig normalverteilte Mutationen vor. In seinem Konzept der *korrelierten Mutationen* wird die Verteilung durch n Standardabweichungen σ_i und $n(n-1)/2$ Rotationswinkel α_{lm} mit $l = 1, \dots, n$ und $m < l$ beschrieben. Ein Individuum wird damit durch $I = (x_1, \dots, x_n, \sigma_1, \dots, \sigma_n, \alpha_{2,1}, \dots, \alpha_{n,n-1})$ dargestellt. Die Komponenten der Standardabweichung der $(k+1)$ -ten Generation $\sigma_i^{(k+1)}$ ergeben sich wie in Formel 3, die Rotationswinkel werden mit der Regel

$$\alpha_{lm}^{(k+1)} = \left(\alpha_{lm}^{(k)} + \zeta_{lm} + \pi \right) \bmod 2\pi - \pi \text{ mit } \zeta_{lm} \sim w \cdot \frac{\pi}{180} \cdot N(0, 1) \quad (5)$$

mutiert. Dabei bezeichnen die ζ_{lm} für jedes α_{lm} individuelle Ausprägungen standardnormalverteilter Zufallsvariablen mit Mittelwert 0 und einer Standardabweichung des Winkels w (bei Schwefel $w = 5^\circ$). Die Rotationswinkel können Werte von $-\pi < \alpha_{lm}^{(k+1)} < \pi$ annehmen.

Für die Generation $(k+1)$ ergibt sich die Kovarianzmatrix $C^{(k+1)}$ der Mutationsschrittweitenverteilung dann zu $C^{(k+1)} = \left(T^{(k+1)}S^{(k+1)}\right) \left(T^{(k+1)}S^{(k+1)}\right)'$. Dabei stellt $S^{(k+1)}$ die Diagonalmatrix der Standardabweichungen $\sigma_i^{(k+1)}$ und $T^{(k+1)}$ eine orthogonale Matrix dar, die aus der Verknüpfung aller $n(n-1)/2$, durch die $\alpha_{lm}^{(k+1)}$ beschriebenen elementaren Rotationen entsteht.

Ist R_{lm} eine Einheitsmatrix bis auf die Einträge $r_{ll} = r_{mm} = \cos(\alpha_{lm})$ und $r_{lm} = -r_{ml} = \sin(\alpha_{lm})$, so beschreibt R_{lm} eine Rotation in der (l,m) -Ebene und es ergibt sich $T = \prod_{m<l=1}^n R_{lm}$. Für die Mutation der Individuen in der $(k+1)$ -ten Generation gilt dann:

$$x^{(k+1)} = x^{(k)} + z^{(k+1)} \quad \text{mit} \quad z^{(k+1)} \sim \mathcal{N}(0, C^{(k+1)}) = T^{(k+1)} \cdot S^{(k+1)} \cdot \mathcal{N}(0, E) \quad (6)$$

Die sphärische $\mathcal{N}(0, E)$ -Verteilung (mit $E \in \mathbb{R}^{n \times n}$ Einheitsmatrix) wird hier durch die Multiplikation mit den Schrittweitenparametern in $S^{(k+1)}$ in ein achsenparalleles Hyperellipsoid überführt und anschließend sukzessive in allen $\frac{n(n-1)}{2}$ 2-dimensionalen Unterräumen rotiert, die durch kanonische Basisvektoren aufgespannt werden (Abbildung 2, rechts). Mittels dieser Mutationsvariante kann im Verlauf des Algorithmus jede n -dimensionale Normalverteilung mit Mittelwert 0 entstehen [Rud92]. Dieser Ansatz hat aber einige Nachteile.

- Die Anzahl der strategischen Parameter, die sich unter der Kontrolle des Mutationsoperators befinden, wächst hier quadratisch mit der Dimension des Problems. Erfahrungen zeigen, dass die Populationsgröße ungefähr mit der Anzahl der freien Parameter übereinstimmen sollte. Für Problemdimensionen mit $n > 10$ nimmt die Performance des Algorithmus rapide ab.
- Wegen der Modellierung der Kovarianzmatrixmutation über Drehungswinkel und der (konstant) normalverteilten Änderung dieser Drehungswinkel ist die Sensitivität, mit der ein Hyperellipsoid auf eine Drehung reagiert, stark von der aktuellen Belegung der Standardabweichungen abhängig. Ist zum Beispiel bei einem Problem mit Dimension $n = 2$ der aktuelle Quotient $\sigma_1/\sigma_2 \approx 1$ (fast ein Kreis), führt eine Drehung um 5° zu einer sehr großen Überdeckung der Verteilungen und somit zu fast keiner Veränderung. Andererseits führt die Drehung um den gleichen Winkel bei $\sigma_1/\sigma_2 \approx 100$ zu einer sehr kleinen Überdeckung der Verteilungen und damit zu einer großen Mutation. Holzheuer [Hol96] untersucht die Auswirkungen dieses Effekts bei verschiedenen Funktionsklassen. Eine evolutionäre Strategie mit korrelierter Mutation zeigt danach bei konvex-quadratischen Funktionen mit hohem Verhältnis der Achsenskalierungen einen um mehrere Größenordnungen langsameren Fortschritt als bei dem Sphärenproblem ($f(x) = \sum_{i=1}^n x_i^2$).
- Die Leistungsfähigkeit dieser Selbstadaption hängt stark von der ursprünglichen Ausrichtung des gegebenen Koordinatensystems ab, ist also nicht invariant gegenüber orthogonalen Transformationen der Fitnessfunktion.

Diese Kritikpunkte wurden in den Arbeiten von Hansen und Ostermeier [HO96] diskutiert. Darin wird die Kombination zweier Verfahren vorgeschlagen, nämlich der *Kovarianz Matrix Adaption*, bei der die Kovarianzmatrix unter Verwendung der letzten Mutationschritte der zuletzt ausgewählten Individuen durch eine Hauptkomponentenanalyse adaptiert wird und zweitens der Technik der *Kumulation*, bei der der Mutationsoperator mit einem Gedächtnis über die vergangenen Mutationsschrittweiten und Selektionen ausgestattet wird, und die dort gespeicherte Information in die Adaption der Mutation einfließt.

Die Kovarianz Matrix Adaption: Bei diesem Ansatz wird ausgenutzt, dass eine beliebige n -dimensionale Normalverteilung mit Mittelwert 0 durch ausreichend viele den \mathbb{R}^n aufspannenden Vektoren generiert werden kann, also existieren für jede Kovarianzmatrix C geeignete v_1, \dots, v_m mit $m \geq n$, so dass

$$\mathcal{N}(0, C) \sim \sum_{i=1}^m \mathcal{N}(0, v_i v_i') \sim \sum_{i=1}^m N(0, 1) \cdot v_i.$$

Bei der Kovarianz Matrix Adaption wird C gemäß dieser Beziehung in jeder Iteration des Algorithmus aus den realisierten Mutationsschrittweiten der in früheren Generationen ausgewählten Individuen generiert. In der ersten Generation wird die Kovarianzmatrix $C^{(0)}$ aus den kanonischen Basisvektoren e_1, \dots, e_n zusammengesetzt und damit als Einheitsmatrix initialisiert, was einer isotropen Normalverteilung entspricht. In jedem weiteren Schritt wird ein Individuum ausgewählt, dann die bisherige Verteilung mit einem Dämpfungsfaktor $d \in (0, 1)$ versehen und schließlich die Mutationsschrittweite v_k des aktuell ausgewählten Individuums addiert. Die Verteilung der (k)-ten Generation ergibt sich also zu

$$\mathcal{N}(0, C^{(k)}) \sim d^k \sum_{j=1}^n N(0, 1) \cdot e_j + \sum_{i=1}^k \left(N(0, 1) \cdot v_i \cdot d^{k-i} \right). \quad (7)$$

Damit ist

$$C^{(k)} = d^{2k} \sum_{j=1}^n e_j e_j' + \sum_{i=1}^k d^{2 \cdot (k-i)} \cdot v_i v_i' = \sum_{j=1}^n w_{j,k}^2 u_{j,k} u_{j,k}'. \quad (8)$$

Hier bezeichnen $w_{j,k}^2 \in \mathbb{R}$ die Eigenwerte und $u_{j,k} \in \mathbb{R}^n$ die normalisierten Eigenvektoren von $C^{(k)}$. Mit dieser Technik wird die bei der korrelierten Mutation auftretende unterschiedliche Wirkung beseitigt, die von Verteilungshauptachsen unterschiedlicher Größenordnung bei gleichen Drehwinkeln verursacht wird. Auch spielt die Lage des Koordinatensystems keine Rolle mehr, der Mutationsoperator ist also invariant bezüglich der Rotation.

Kumulation und Evolutionspfade: Die Idee der Evolutionspfade besteht darin, die realisierten Mutationsschrittweiten der ausgewählten Individuen über mehrere Generationen

in einem Vektor zu kumulieren und dann den Mutationsoperator im Hinblick auf diese Pfadvektoren anzupassen. Auf diese Weise soll einer Fortschrittsrate größeres Gewicht beigemessen werden als der punktuellen Wahrscheinlichkeit einer Auswahl. Der kumulierte Evolutionspfad $p^{(k+1)} \in \mathbb{R}^n$ der $(k+1)$ -ten Generation wird nach [OGH94] über einen Gewichtungsfaktor $g \in (0, 1]$ beschrieben als

$$p^{(k+1)} = (1 - g)p^{(k)} + \sqrt{g(2 - g)} \cdot v_k .$$

v_k bezeichnet den Mutationsvektor des ausgewählten Individuums. Der Normalisierungsfaktor $\sqrt{g(2 - g)}$ sorgt für konstante Varianz $\text{var}(p^{(k+1)}) = \text{var}(p^{(k)})$; denn es gilt $(1 - g)^2 + \left(\sqrt{g(2 - g)}\right)^2 = 1$. Der Gewichtungsfaktor g beschreibt, wie schnell die in $p^{(k)}$ eingebrachten Informationen über die Generationen wieder an Bedeutung verlieren, $g = 1$ bedeutet maximalen Verlust dieser Information, also beinhaltet der Pfad nur die Information der letzten Mutation. Empfohlen wird ein Wert von $\sqrt{\frac{\pi}{2}} < \frac{1}{g} < n$ in Abhängigkeit von der Problemdimension n [Han98].

Kombination der Verfahren: Die Verknüpfung dieser Konzepte mit der Erweiterung um einen globalen Schrittweitenparameter $\sigma^{(k)} \in \mathbb{R}$ führt zu einem Algorithmus, der in [HO01] mit CMA-ES bezeichnet wird. Er basiert auf dem Selektionskonzept der gewichteten Rekombination und kann folgendermaßen beschrieben werden.

Aus λ Individuen werden gemäß ihrer Fitness die besten μ Individuen ausgewählt, mit entsprechenden fitnessproportionalen Gewichten multipliziert und rekombiniert. Auf diese Weise erhält man für die Individuen $x_j^{(k+1)}$ der $(k+1)$ -ten Generation

$$x_j^{(k+1)} = \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i x_i^{(k)} + \sigma^{(k)} \cdot z_j^{(k+1)} \text{ mit } z_j^{(k+1)} \sim \mathcal{N}(0, C^{(k)}) . \quad (9)$$

Die Entwicklung von $C^{(k)}$ geschieht mit Hilfe von Evolutionspfaden, wobei der Pfad $p_C^{(k+1)}$ über den gewichteten Durchschnitt der selektierten Mutationsschrittweiten konstruiert wird:

$$p_C^{(k+1)} = (1 - g_c) \cdot p_C^{(k)} + \sqrt{g_c(2 - g_c)} \cdot \underbrace{\frac{\sum_{i=1}^{\mu} w_i}{\sqrt{\sum_{i=1}^{\mu} w_i^2}}}_{a^{(k+1)}} \cdot \underbrace{\frac{1}{\sum_{i=1}^{\mu} w_i} \cdot \sum_{i=1}^{\mu} w_i z_j^{(k+1)}}_{b^{(k+1)}} \quad (10)$$

$a^{(k+1)}$ ist ein Normalisierungsfaktor, der dafür sorgt, dass $b^{(k+1)}$ bzw. $z_j^{(k+1)}$ identische Varianzen und Verteilungen besitzen. Die Kovarianzmatrix der $(k+1)$ -ten Generation berechnet sich dann aus

$$C^{(k+1)} = (1 - c_{\text{cov}}) C^{(k)} + c_{\text{cov}} \cdot p_C^{(k+1)} \left(p_C^{(k+1)} \right)' . \quad (11)$$

$c_{\text{COV}} \in [0, 1)$ bezeichnet dabei einen Faktor, der die potentielle Änderungsrate der Kovarianzmatrix beschreibt.

Die globale Schrittweitenänderung $\sigma^{(k)}$ wird ebenfalls über evolutionäre Pfade $p_{\sigma}^{(k)}$ generiert. Allerdings wird in der Konstruktion dieser Pfade nur der Rotationsanteil der Kovarianzmatrix C^k selbstadaptiv gesteuert. Ist also $C^{(k)} = \left(S^{(k)}T^{(k)}\right)' \left(S^{(k)}T^{(k)}\right)$ mit einer Rotationsmatrix $T^{(k)}$ und einer Diagonalmatrix $S^{(k)}$, so ergibt sich

$$p_{\sigma}^{(k+1)} = (1 - g_{\sigma}) \cdot p_{\sigma}^{(k)} + \sqrt{g_{\sigma}(2 - g_{\sigma})} \cdot a^{(k+1)} \cdot T^{(k)} \left(S^{(k)}\right)^{-1} \left(T^{(k)}\right)^{-1} \cdot b^{(k+1)}. \quad (12)$$

Hier sind $a^{(k+1)}$ der Normalisierungsfaktor und $b^{(k+1)}$ der gewichtete Durchschnitt der $\mathcal{N}(0, C^{(k)})$ -verteilten ausgewählten Schrittweiten aus Gleichung 10. $g_{\sigma} \in (0, 1)$ ist der Gewichtungsfaktor für die Schritte in früheren Generationen. Gleichung 12 erfordert die Zerlegung von $C^{(k)}$ in $S^{(k)}$ und $T^{(k)}$, da diese in diesem Schritt des Algorithmus explizit getrennt sind. Die neue globale Schrittweite ergibt sich dann zu

$$\sigma^{(k+1)} = \sigma^{(k)} \cdot \exp \left(\frac{1}{d_{\sigma}} \left(\frac{|p_{\sigma}^{(k)}|}{\mathbb{E}[|\mathcal{N}(0, E)|]} - 1 \right) \right). \quad (13)$$

Dabei gibt $d_{\sigma} > 1$ die potentielle Änderungsrate von $\sigma^{(k)}$ an und der innere Klammerausdruck entspricht der Länge von $p_{\sigma}^{(k)}$, normiert und zentriert durch den Erwartungswert der Länge eines $(0, E)$ -normalverteilten n -dimensionalen Zufallsvektors. Eine gute Näherung wird in [Ost97] mit $\mathbb{E}[|\mathcal{N}(0, E)|] \approx \sqrt{n} \left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right)$ angegeben.

Die zuletzt beschriebene Variante der Selbstadaption von Strategieparametern in evolutionären Strategien ist außer in theoretischen Simulationsstudien [HO01] schon in einigen Applikationen erfolgreich eingesetzt worden, beispielsweise zur Optimierung von Potentialfeldmodellen [Alv98], zur Modellkalibrierung von Kläranlagenmodellen [Hol98] oder zur Optimierung von Flugzeugformen [LW98].

Realisierungen basieren dabei auf einer Implementierung in Matlab, einem kommerziellen Softwarepaket für computergestützte Berechnungen. Eine frei verfügbare, einfach anzuwendende Implementierungsvariante ist deswegen wünschenswert. Im Folgenden soll ein für diesen Zweck in der Sprache Perl entwickeltes Modul beschrieben werden.

4 Implementierung und Beispiel

Selbstadaptive Mutation in Perl: Perl ist eine weit verbreitete und frei verfügbare objektorientierte Programmier- und Skriptsprache. Das im Rahmen dieser Arbeit entwickelte Modul `MutCorrAdapt` [Ets03] implementiert die globale Mutationsadaption analog Gleichung 1 und 2, die achsenparallele individuelle Mutationsadaption wie in Gleichung 3 und 4 sowie die in Gleichung 9 bis 13 beschriebene Variante (CMA-ES). `MutCorrAdapt`

setzt die Perl-Bibliotheken Algorithm::Evolutionary und PDL voraus und ist über eine einfach zu bedienende XML-Schnittstelle steuerbar.

Simulationsbeispiel: Als Beispiel soll die Funktion $f : \mathbb{R}^n \rightarrow \mathbb{R}$ mit

$$f(x) = \sum_{i=1}^n \left[\frac{1}{2} \cdot \left(10^{\frac{i-1}{n-1}} \cdot h_i(x) \right)^2 + \sin^2 \left(2\pi \cdot 10^{\frac{i-1}{n-1}} \cdot h_i(x) \right) \right] \quad (14)$$

minimiert werden. $h : \mathbb{R}^n \rightarrow \mathbb{R}^n$ bezeichnet dabei eine zufällige Rotation und Spiegelung des Koordinatensystems, z.B. von der Form $h(x) = \left(\prod_{m < l=1}^n R_{lm} \right) \cdot x$ mit R_{lm} als Rotation in der (l, m) -Ebene um einen zufälligen, gleichverteilten Winkel $\alpha_{lm} \sim U(0, \frac{\pi}{2})$. Durch diese Rotation wird die Parallelität der Funktionstopologie zu den Koordinatenachsen verhindert. $f(x)$ ist multimodal, skalierbar mit der Dimension n , nicht linear, nicht separabel und kann mit lokalen Gradientenverfahren nicht global minimiert werden. Das Minimum von $f(x)$ liegt bei $x = 0$.

Anhand dieser Funktion wurden die in MutCorrAdapt implementierten Varianten mittels einer Simulationsstudie gegenübergestellt. Die Populationsgröße wurde hierbei mit $\lambda = 120$ und die Anzahl der Eltern mit $\mu = 24$ gewählt. Die Parameter waren wie folgt belegt:

- Ein globaler Schrittweitenparameter: $\sigma^{(0)} = 1$ (Variante a)
- Individuelle, achsenparallele Schrittweiten: $\sigma_i^{(0)} = 100/\sqrt{n}$ (Variante b)
- CMA-ES (Empfehlungen für Parameter aus [HO01]): (Variante c)
 $\sigma^{(0)} = 10000$, $w_i = \ln\left(\frac{\lambda+1}{2}\right) - \ln(i)$,
 $g_\sigma = g_C = \frac{4}{4+n}$, $c_{\text{cov}} = \frac{2}{(n+\sqrt{2})^2}$, $d_\sigma = g_\sigma^{-1} + 1$

Für jede der drei Algorithmusvarianten wurden jeweils 50 Simulationen für $n = 2, 5, 10$ durchgeführt und die Ergebnisse mit dem jeweils schlechtesten Funktionswert festgehalten. Abbruchkriterien orientierten sich an Schranken für Fitnesswerte oder der Laufzeit des Verfahrens. Die Auswahl der Startpositionen der Individuen erfolgte gemäß einer Gleichverteilung mit $x_i^{(0)} \sim U(-100, 100)$.

Ergebnis: In Abbildung 3 werden für alle drei Verfahrensvarianten die Simulationsläufe mit den jeweils höchsten Funktionswerten in Abhängigkeit der Anzahl der Generationen für die Dimensionen $n = 2, 5, 10$ grafisch dargestellt. Bei Variante a (—) wird das Optimum dieser Funktion zumindest mit Standardeinstellungen generell verfehlt. Variante b (---) zeigt hier vor allem bei der Rate der Verbesserung einen kleinen Vorteil aufgrund der unterschiedlich skalierten Schrittweitenkomponenten. Mit Variante c (·····) konnten in allen untersuchten Fällen, auch in den schlechtesten Simulationsläufen, Funktionswerte unter 10^{-10} erreicht werden.

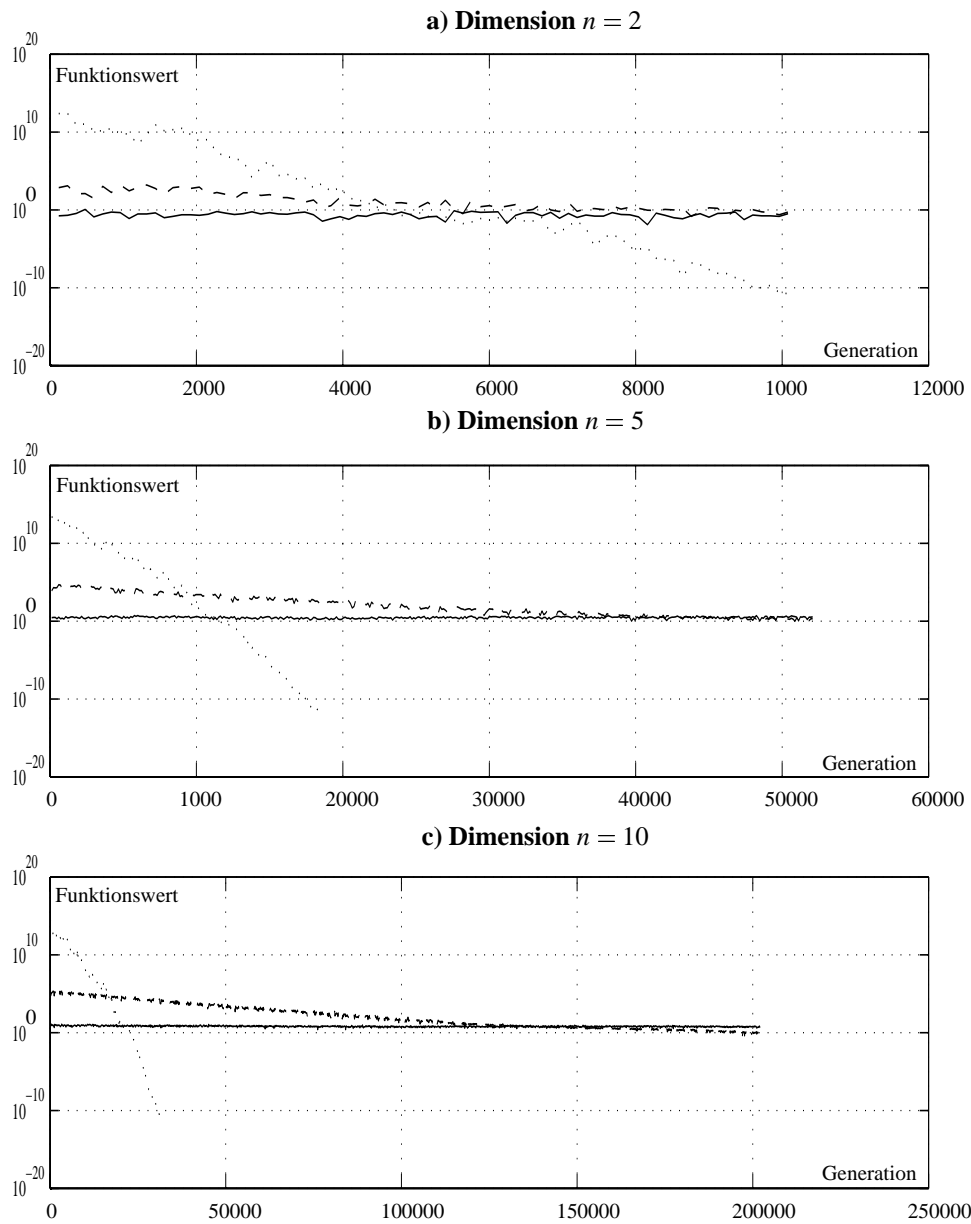


Abbildung 3: Funktionswerte in Abhängigkeit der Anzahl der Generationen für drei Algorithmusvarianten bei a) $n = 2$, b) $n = 5$, c) $n = 10$

5 Zusammenfassung

Viele Optimierungsaufgaben lassen sich mit evolutionären Algorithmen erfolgreich bewältigen. Selbstadaptive Techniken zur dynamischen Verbesserung der Strategieparameter entscheiden über die Geschwindigkeit, mit der sich ein evolutionärer Algorithmus dem Optimum nähert. Vor allem bei evolutionären Strategien ist die Entwicklung neuerer Ansätze zur Selbstadaption des Mutationsoperators aktuell. Mit Hilfe des hier vorgestellten Softwaremoduls und einer bestimmten Klasse von nicht linearen, multimodalen und mit lokalen Gradientenverfahren nicht global minimierbaren Testfunktionen (siehe Formel 14) zeigt sich, dass die neueren Ansätze älteren Methoden tendenziell überlegen sein können.

Literatur

- [Ala03] Jarmo T. Alander. An Indexed Bibliography of Genetic Algorithms Implementations. Report 94-1-IMPLE, University of Vaasa, Department of Information Technology and Production Economics, 2003. <ftp://ftp.uwasa.fi/cs/report94-1/gaIMPLEbib.ps.Z>.
- [Alv98] M. Alvers. *Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle*. Berliner geowissenschaftliche Abhandlungen, Reihe B: Geophysik. Selbstverlag Fachbereich Geowissenschaften, Freie Universität Berlin, 1998.
- [Bäc92] Thomas Bäck. Self-Adaptation in Genetic Algorithms. In *Proceedings of the 1st European Conference on Artificial Life, December 11-13, 1991*, S. 263–271, Paris, France, 1992. MIT Press.
- [Bäc93] Thomas Bäck. An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation*, S. 1–23, 1993.
- [BFN96] Wolfgang Banzhaf, Frank D. Francone und Peter Nordin. The Effect of Extensive Use of the Mutation Operator on Generalization in Genetic Programming Using Sparse Data Sets. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg und Hans-Paul Schwefel (Hrsg.), *Parallel Problem Solving from Nature IV, Proceedings of the International Conference on Evolutionary Computation*, volume 1141, S. 300–309, Berlin, Germany, 22-26 1996. Springer Verlag.
- [DS90] G. Dück und T. Scheuer. Threshold Accepting: A General Purpose Optimization Algorithm Superior to Simulated Annealing. *Journal of Computational Physics*, 90:161–175, 1990.
- [Ets03] Stefan Etschberger. *MutCorrAdapt: Eine Bibliothek für selbstadaptive evolutionäre Strategien*. *Arbeitspapiere zur Mathematischen Wirtschaftsforschung, Universität Augsburg*, 2003.
- [FOW66] L. J. Fogel, A. J. Owens und M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley and Sons, New York, 1966.
- [GAB⁺02] Juan Julian Merelo Guervos, Panagiotis Adamidis, Hans-Georg Beyer, Jose-Luis Fernandez-Villacanas und Hans-Paul Schwefel (Hrsg.). *Parallel Problem Solving From Nature - PPSN VII*, volume 7, 2002.

- [Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
- [Gre86] J. J. Grefenstette. Optimization of Control Parameters for Genetic Algorithms. In *IEEE Transactions on Systems, Man and Cybernetics SMC-16*, S. 122–128, 1986.
- [GS95] Andreas Geyer-Schulz. *Fuzzy Rule-Based Expert Systems and Genetic Machine Learning*. Physica Verlag, 1995.
- [Han98] N. Hansen. *Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie*. Mensch und Buch Verlag, Berlin, Germany, 1998.
- [HB91] Frank Hoffmeister und Thomas Bäck. Genetic Self-Learning. In *Proceedings of the 1st European Conference on Artificial Life, December 11-13, 1991*, S. 227–235, Paris, France, 1991. MIT Press.
- [HO96] N. Hansen und A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, 1996.
- [HO01] N. Hansen und A. Ostermeier. Completely Derandomized Self-Adaption in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195, 2001.
- [Hol75] J. H. Holland. *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press, 1975.
- [Hol96] C. Holzheuer. Analyse der Adaptation von Verteilungsparametern in der Evolutionsstrategie. Master’s thesis, TU Berlin, 1996.
- [Hol98] D. Holste. *Industrielle Anwendungen Evolutionärer Algorithmen*, chapter 4, „Modellkalibrierung am Beispiel von Kläranlagenmodellen“, S. 37–44. Oldenbourg Verlag, 1998.
- [Jon75] K. De Jong. *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, 1975.
- [Koz92] J. R. Koza. *Genetic Programming*. MIT Press, Cambridge/MA, 1992.
- [LW98] T. Lutz und S. Wagner. Drag reduction and shape optimization of airship bodies. *Journal of Aircraft*, 1998.
- [Müh92] H. Mühlenbein. How Genetic Algorithms Really Work I. Mutation and Hillclimbing. In R. Männer und B. Manderick (Hrsg.), *Parallel Problem Solving from Nature. Proceedings of the Second Conference on Parallel Problem Solving from Nature, Amsterdam*. North-Holland, 1992.
- [Mit02] Melanie Mitchell. *An introduction to genetic algorithms*. The MIT Press, Cambridge, Massachusetts, London, 2002.
- [Nis97] Volker Nissen. *Einführung in evolutionäre Algorithmen*. Vieweg, Braunschweig u.a., 1997.
- [OGH94] A. Ostermeier, A. Gawelczyk und N. Hansen. Step-size adaptation based on non-local use of selection information. In Y. Davidor et al. (Hrsg.), *Proceedings of PPSN IV, Parallel Problem Solving from Nature*, Berlin, Germany, 1994. Springer.
- [Ost97] A. Ostermeier. *Schrittweitenadaptation in der Evolutionsstrategie mit einem entstochastisierten Ansatz*. PhD thesis, Technische Universität Berlin, 1997.

- [Rec73] I. Rechenberg. *Evolutionstrategie. Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, 1973.
- [Rud92] G. Rudolph. On correlated mutations in evolution strategies. In R. Männer und B. Manderick (Hrsg.), *Parallel Problem Solving from Nature. Proceedings of the Second Conference on Parallel Problem Solving from Nature, Amsterdam*. North-Holland, 1992.
- [SB92] Nici Schraudolph und Rick Belew. Dynamic Parameter Encoding in Genetic Algorithm. *Machine Learning*, 1992.
- [Sch81] H.-P. Schwefel. *Numerical Optimization of Computer Models*. Wiley, Chichester, England, 1981.
- [TS93] David M. Tate und Alice E. Smith. Expected Allele Coverage and the Role of Mutation in Genetic Algorithms. In Stephanie Forrest (Hrsg.), *Proc. of the Fifth Int. Conf. on Genetic Algorithms*, S. 31–37, San Mateo, CA, 1993. Morgan Kaufmann.