Evolution Strategies

Fachhochschule Vorarlberg
Master Informatik

Dozent

Hans-Georg Beyer

Vorgelegt von

Griesser, Riedmann

Dornbirn, 24.01.2017

Kurzreferat

Diese Übung dient zum Vergleich verschiedener evolutionärer Strategien. Begonnen wird mit der (1+1)-ES, welche in Folge mit der 1/5-Regel erweitert und an der Sharp- und Parabolic-Ridge Funktion evaluiert wird. Anschließend wird noch die (1, λ)-σ SA und die (μ/μ_I, λ)-σ SA Strategie am Kugelmodell evaluiert.

Inhaltsverzeichnis

1. Implementierung der (1+1)-ES in Octave

1.1 Evaluation an Kugelmodell

Zuerst wird die (1+1)-Evolutionsstrategie anhand des vorgelegten Pseudocodes in Octave implementiert. Zur Evaluation wird als Testfunktion die Kugelfunktion herangezogen. Als Parameter wird ein zehndimensionaler Startelter, initialisiert mit $y_i = 10$, verwendet und als Sigma für die Evolutionsstrategie wird 1 gewählt.

Ein Testdurchlauf mit 2000 Generationen (vgl. Abbildung 1) liefert ein vergleichbares Bild wie die Vorlage. Anfangs wird mit jeder Generation eine große Verbesserung erzielt, allerdings wird nach einigen Generationen dieser Fortschritt nach und nach geringer und jede weitere Verbesserung wird wiederum erst nach sehr vielen Generationen gefunden.

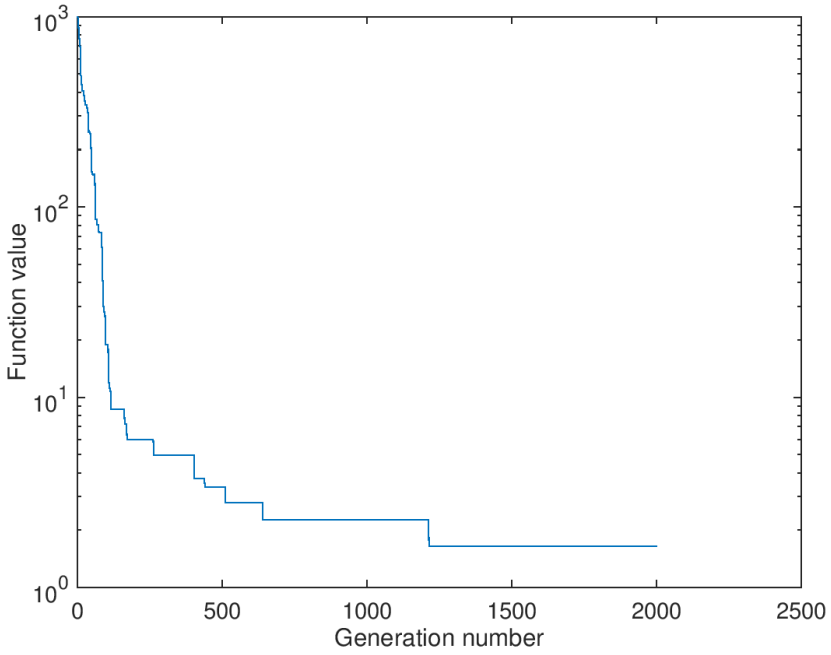

Abbildung 1: (1+1)-ES an Kugelmodell

1.2 Die (1+1)-ES im binären Suchraum

Damit der Algorithmus speziell auf den Suchraum $\mathbb{B}^N$ angewandt werden kann, werden minimale Anpassungen vorgenommen. Für den Startelter wird $y_i \in \{0, 1\}$ vorausgesetzt. Damit zur Mutation jedes Bit mit einer bestimmten Wahrscheinlichkeit gekippt wird, wird die Mutation wie folgt durchgeführt: Der Elter wird mit einem Zufallsvektor $P(y_i = 1) = p_m$ addiert und anschließend wird auf den Vektor die Modulo-2 Operation angewandt, um den Nachkommen zu erhalten. Ebenfalls wird mit diesem Algorithmus die Maximierung der OneMax-Funktion angestrebt.

Zur Evaluation wird der Startelter zufällig mit $N = 100$ und $P(y_i = 1) = 0.5$ initialisiert. Als Mutationswahrscheinlichkeit wird $1/100$ gewählt. Wie in Abbildung 2 zu sehen ist, gibt es anfangs eine für die (1+1)-ES typisch große Verbesserung pro Generation. Diese Verbesserungen werden dann immer seltener, bis der Algorithmus schließlich nach 755 Generationen das Maximum findet.

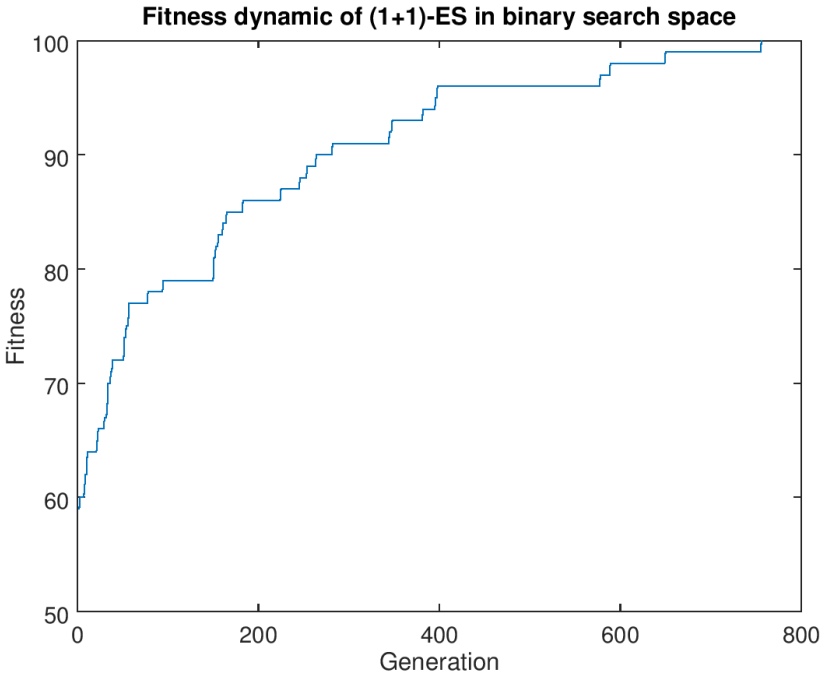

Abbildung 2: (1+1)-ES im binären Suchraum

Um die Laufzeitkomplexität für diesen Algorithmus zu bestimmen, wird dieser einer Variation von unterschiedlich großen Starteltern aufgerufen und jeder dieser Aufrufe wird wiederum mehrmals repliziert, um eine bessere Sicht auf das zufällig gestreute Verhalten des Algorithmus zu bekommen. Für jede zehnte Dimension im Intervall $[10, 300]$ wird ein zufälliger Startelter gewählt und als Mutationswahrscheinlichkeit wird $1/N$ gewählt, wobei N die Anzahl der Dimensionen des Startelters ist. Dies wird für jede gewählte Dimension 50 Mal repliziert. Schlussendlich wird der Mittelwert über die jeweils erhaltenen Anzahl an Generationen, die notwendig waren, bis das Maximum gefunden wurde, ermittelt und in einer Graphik abgebildet (siehe Abbildung 3). Diese Abbildung legt nahe, dass es sich dabei um eine quadratische Laufzeitkomplexität handelt. Anfangs benötigt der Algorithmus etwa 10 Generationen mehr pro zusätzlichem Bit, aber entfernt sich dann mehr und mehr von diesem Faktor.

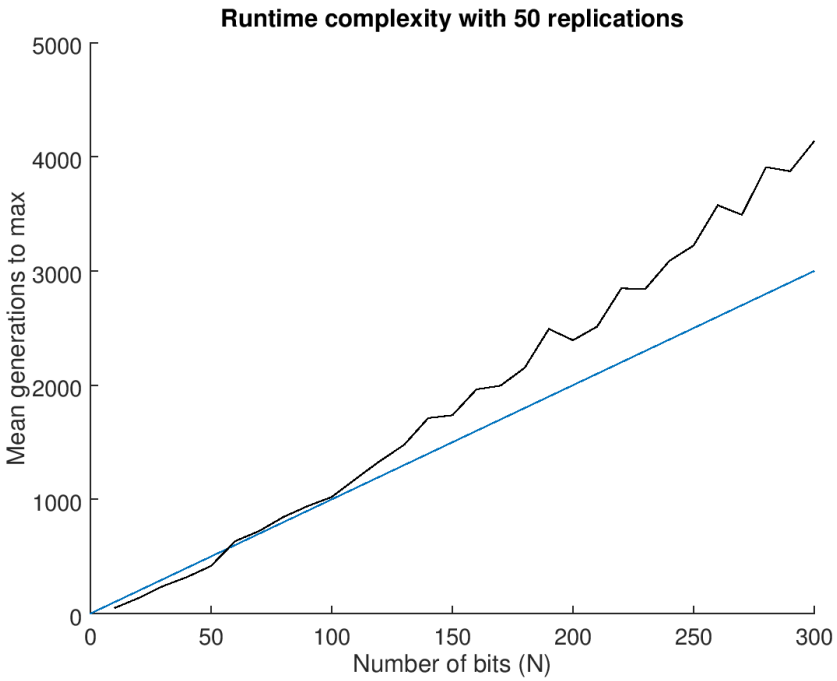

Abbildung 3: Aufgabe 2 - Laufzeitentwicklung

2. Erweiterung durch 1/5-Regel

Zur Verbesserung der (1+1)-ES wird diese nun mit der 1/5-Regel erweitert. Dabei wird das Sigma iterativ mitverbessert, um auch im weiteren Verlauf des Algorithmus noch gute Verbesserungen pro Generation zu erzielen.

Getestet wird die Implementierung wiederum am Kugelmodell. Die Parameter für den Testaufruf sind dabei wie folgt:

- N = 10
- y = (10, ... , 10)$^{\mathrm{T}}$
- $\sigma = 1$
- $\sigma_{\mathrm{stop}} = 10^{-2}$

Im Unterschied zur klassischen (1+1)-ES liefert der erweiterte Algorithmus nun konstant mit jeder Generation eine ähnliche Verbesserung (vgl. Abbildung 8).

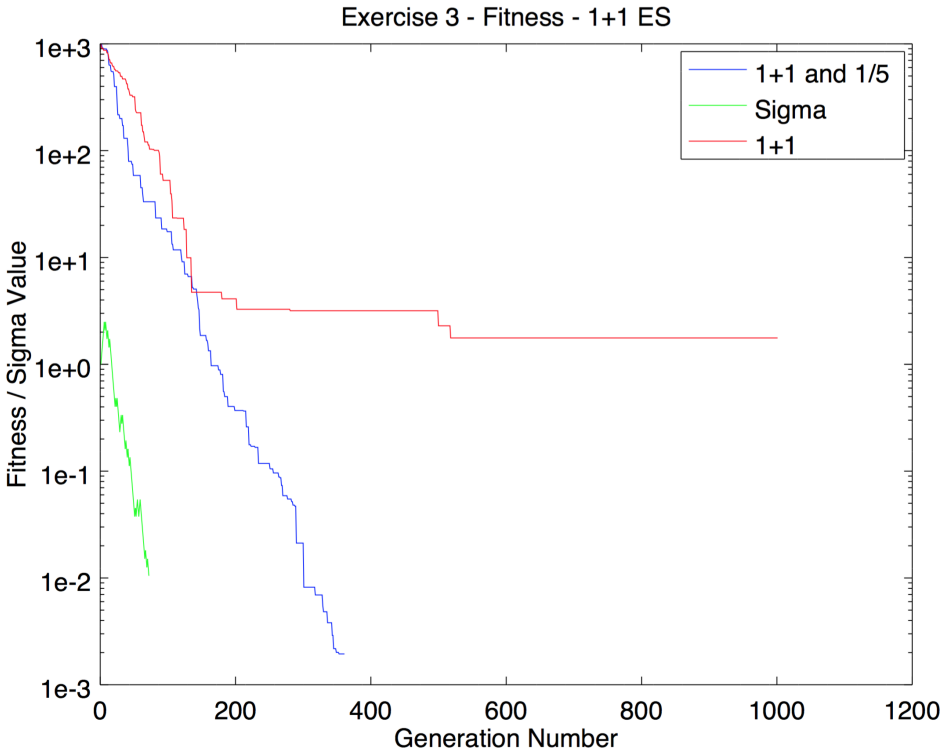

Abbildung 4: Vergleich von (1+1)-ES und (1+1)-ES mit 1/5-Regel

2.0.1 Überprüfung der Korrektheit

Zur absoluten Überprüfung der Korrektheit, wird der Algorithmus mit festgelegten Parametern aufgerufen und der Zufallszahlengenerator mit einem ebenfalls festgelegten Zustand initialisiert. Die Ergebnisse davon sind wie folgt.

- N = 100
- y = $(1, \ldots, 1)^{\mathrm{T}}$
- $\sigma = 1$, $\sigma_{\mathrm{stop}} = 10^{-5}$

Ist-Werte

- Fitnesswert: 2.5113e-07
- Generationen: 6800

Soll-Werte

- Fitnesswert: 2.0220e-07
- Generationen: 7800

Diese Werte liegen außerhalb der Toleranz, der Fehler im Verfahren wurde allerdings nicht gefunden.

3. Evaluation an Sharp- und Parabolic-Ridge Funktion

Als weitere Evaluation des (1+1)-ES Algorithmus mit Erweiterung durch 1/5-Regel wird dieser abseits des Kugelmodells auch noch an den Sharp-Ridge Funktion sowie der Parabolic-Ridge Funktion getestet. Als Parameter werden hierfür folgende Werte verwendet.

- d = 10
- N = 30
- y = $(1, \ldots, 1)^{T}$
- $\sigma = 1$
- $\sigma_{stop} = 3*10^{-3}$

Die Ergebnisse werden linear und halb-logarithmisch dargestellt.

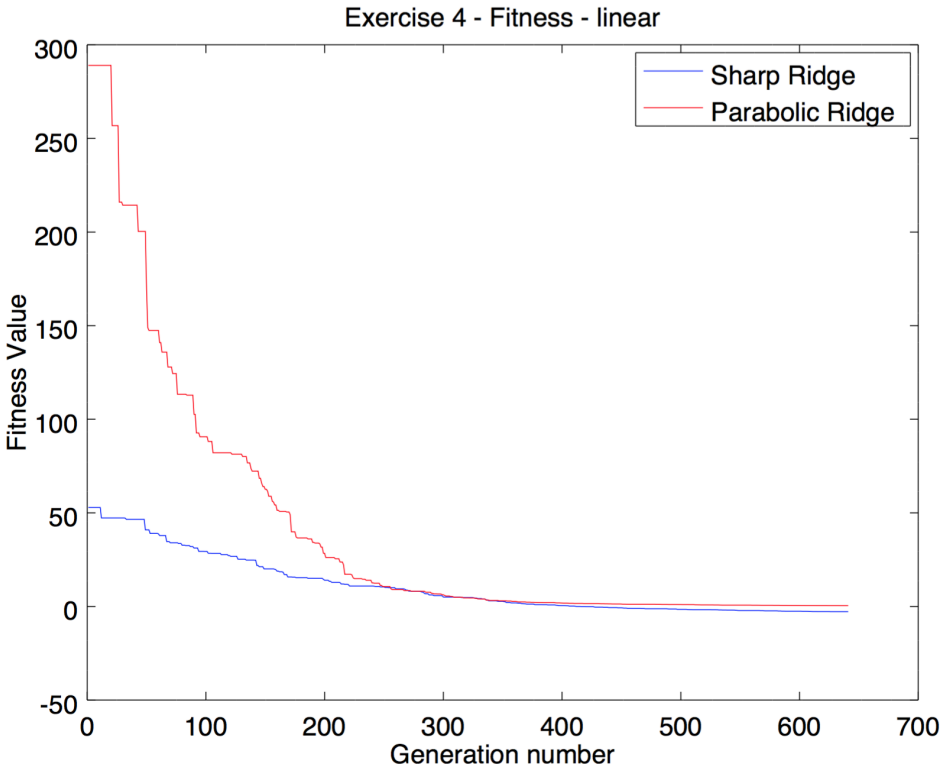

Abbildung 5: Aufgabe 4 - Fitness linear

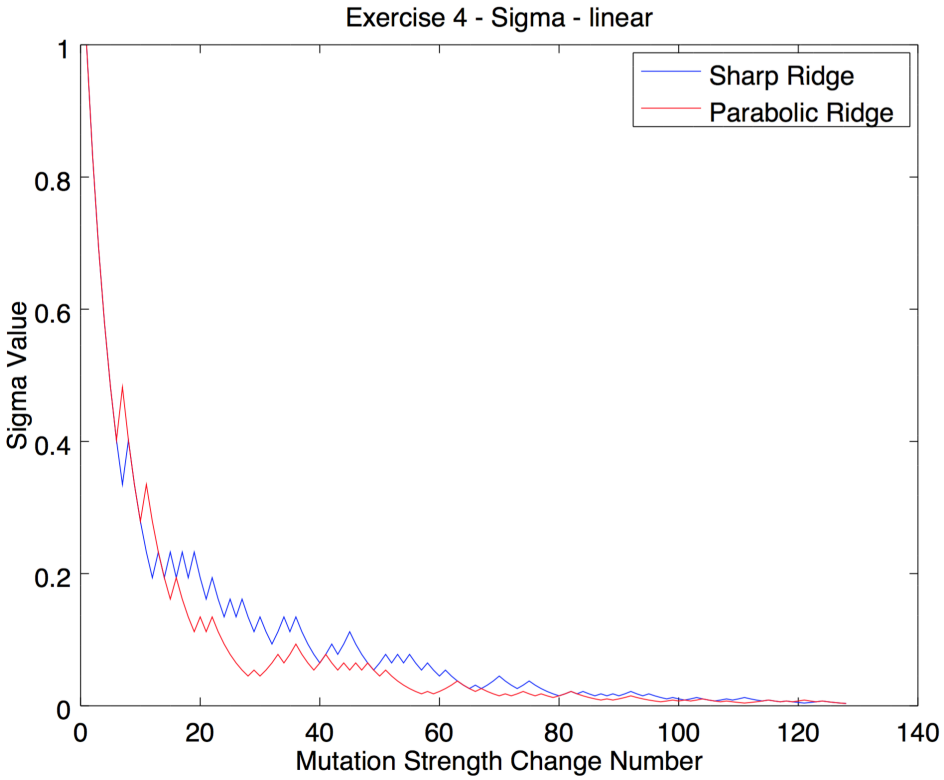

Abbildung 6: Aufgabe 4 - σ linear

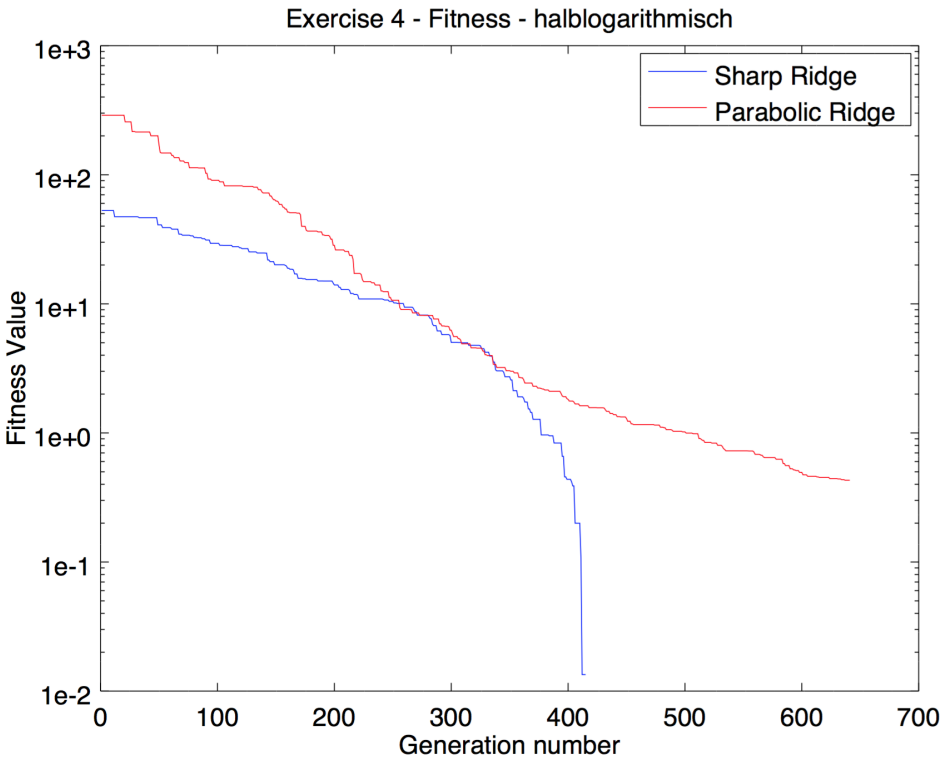

Abbildung 7: Aufgabe 4 - Fitness halb-logarithmisch

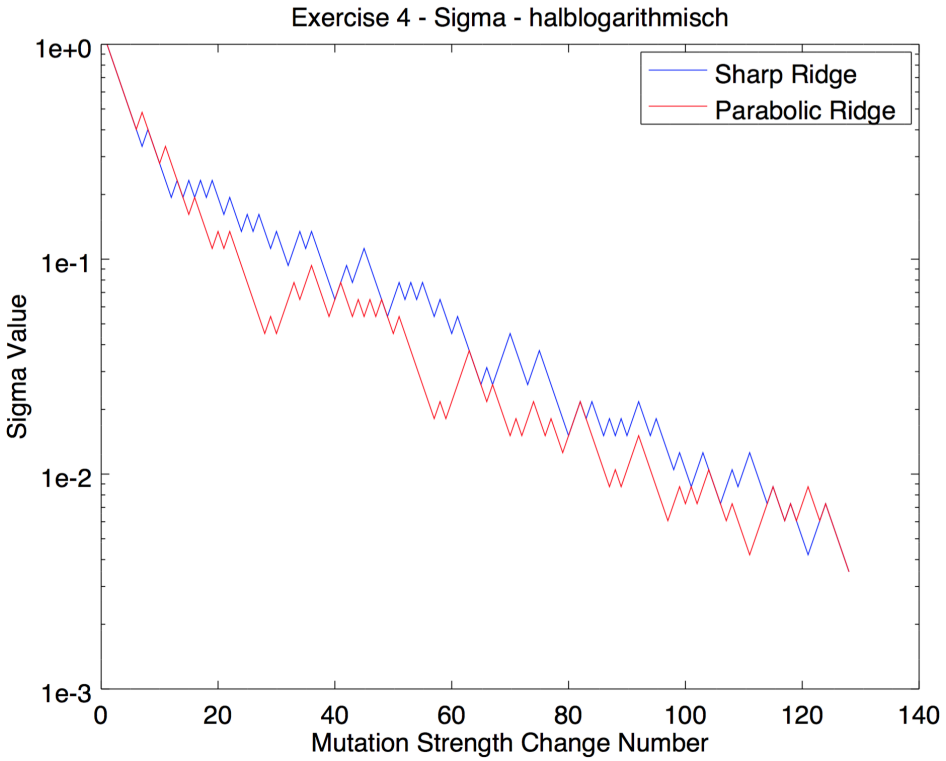

Abbildung 8: Aufgabe 4 - σ halb-logarithmisch

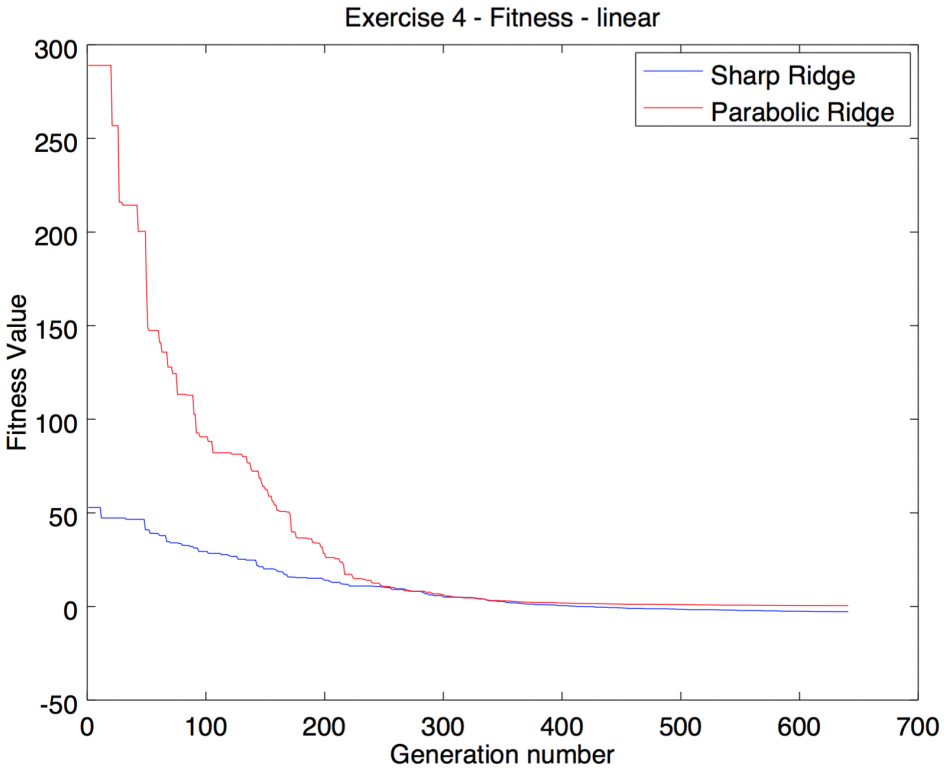

Abbildung 9: Aufgabe 4 - Fitness linear mit d=1

Es ist zu beobachten, dass bei $d = 1$ der Fitnesswert beider Funktionen etwas schneller konvergiert (siehe Abbildung 9).

4. Implementierung der (1, λ)-σ SA-ES

Als weitere Verbesserung der vorangegangen Evolutionsstrategien wird der Algorithmus nun um eine Selbstadaption erweitert. Getestet wird das Ergebnis erneut am Kugelmodell mit Festhaltung der Entwicklung der Fitness sowie des Sigmas. Diese werden auf einem logarithmischen Graphen dargestellt (siehe Abbildung 10) und mit Referenzwerten verglichen. Zu sehen ist in beiden Ergebnissen, dass der Fitnesswert eine gleichbleibende Verbesserung mit jeder Generation durchmacht.

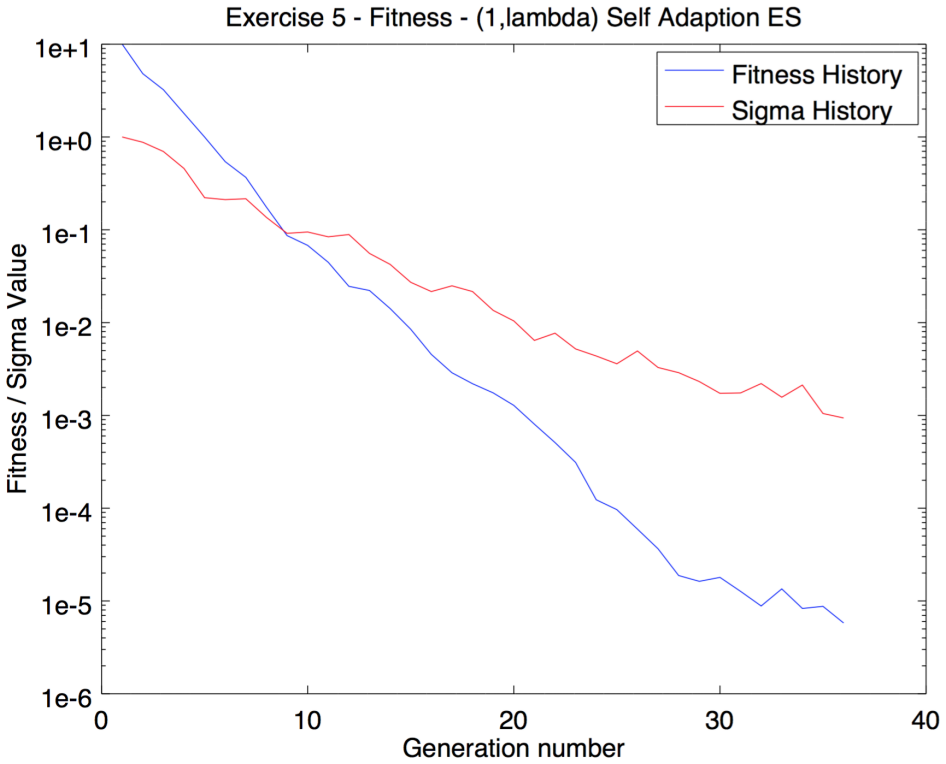

Abbildung 10: Ergebnis des (1, λ)-σ SA-ES am Kugelmodell

5. Erweiterung auf $(\mu/\mu_I, \lambda)$-σ SA-ES

Als finale, hierbei untersuchte Optimierung des Algorithmus, wird dieser nun so angepasst, dass eine Menge an Nachkommen generiert wird und unter den μ besten der Mittelwert für den neuen Elter übernommen wird. Auch für diese Variation wird das Kugelmodell als Testfunktion verwendet. Abbildung 11 zeigt dabei die Entwicklung der Fitness sowie des Sigmas bei $\mu = 3$, $\mu_I = 3$ $\lambda = 10$ sowie $N = 30$.

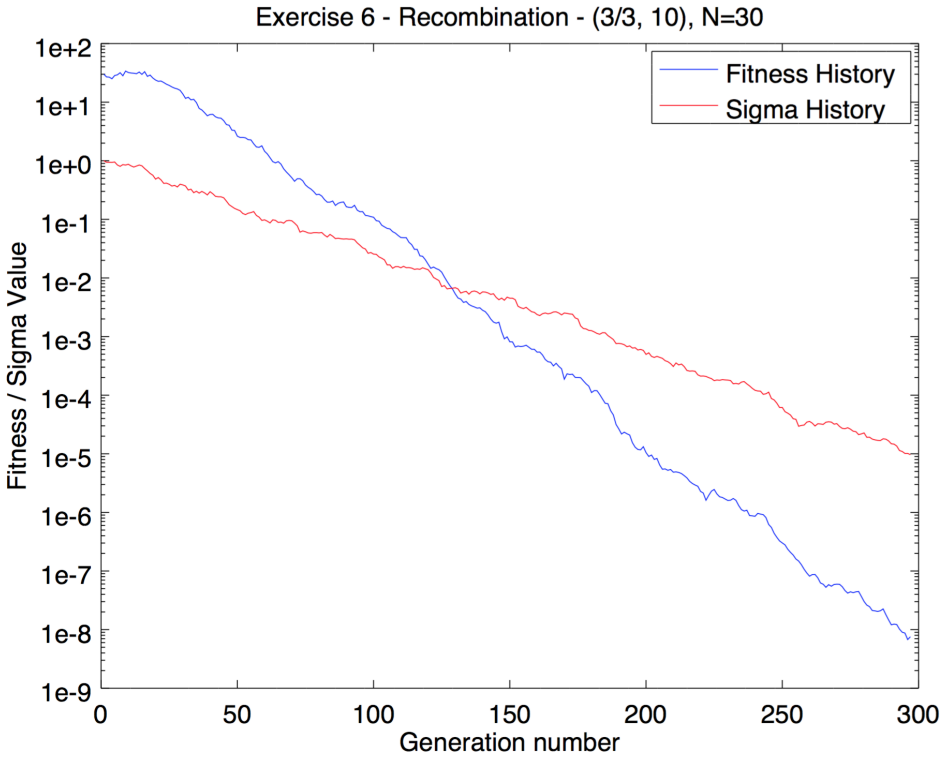

Abbildung 11: Aufgabe 6 - (3/3, 10)-σSA-ES, N = 30

5.1 Überprüfung der Korrektheit

Zur präziseren Bestimmung der Korrektheit, wird der Algorithmus wiederum mit festgelegten Parametern, sowie festgelegten Zufallsgeneratorenzuständen initialisiert und die Ergebnisse verglichen.

5.1.1 (1/1, 10)-σSA-ES

- N = 100
- y = (1, ... , 1)$^{\mathrm{T}}$
- $\sigma = 1$
- $\sigma_{\mathrm{stop}} = 10^{-5}$
- $\lambda = 10$
- $\mu = 1$

Ist-Werte

- Fitnesswert: 3.2429e-12
- Generationen: 163

Soll-Werte

- Fitnesswert: 2.2245e-07
- Generationen: 1314

5.1.2 (3/3, 10)-σSA-ES

- N = 100
- y = (1, ... , 1)$^{\mathrm{T}}$
- $\sigma = 1$
- $\sigma_{\mathrm{stop}} = 10^{-5}$
- $\lambda = 10$
- $\mu = 3$

Ist-Werte

- Fitnesswert: 3.7111e-08
- Generationen: 785

Soll-Werte

- Fitnesswert: 3.7111e-08
- Generationen: 785