BERLINER
GEOWISSENSCHAFTLICHE
ABHANDLUNGEN

Reihe B Band 28

Michael Alvers

Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle

FU · TU · TFH
Berlin
1998

BERLINER GEOWISSENSCHAFTLICHE ABHANDLUNGEN

Reihe A: Geologie und Paläontologie · Reihe B: Geophysik · Reihe C: Kartographie
Reihe D: Geoinformatik · Reihe E: Paläobiologie

Reihe B: Geophysik

D 188

Herausgegeben von geowissenschaftlichen Instituten
der Freien und der Technischen Universität Berlin
sowie von der Technischen Fachhochschule Berlin

Selbstverlag Fachbereich Geowissenschaften
Freie Universität Berlin
1998

Berliner geowiss. Abh.	(B)	28	V+108 S.	88 Abb.	Berlin 1998

Michael Alvers

Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle

Selbstverlag Fachbereich Geowissenschaften, FU Berlin

Gedruckt mit Zuschüssen des Fachbereichs Geowissenschaften der Freien Universität Berlin und des Sonderforschungsbereiches 267 Deformationsprozesse in den Anden.

Druck: Offsetdruckerei G. Weinert, Saalburgstraße 3, 12099 Berlin
Verlag: Selbstverlag Fachbereich Geowissenschaften, Freie Universität Berlin. 1998.

ISBN 3-89582-056-3
ISSN 0772-687X

Meinen Eltern.

Kreativität ist das Ermöglichen neuer Wechselwirkungseinheiten.

Gerd Binnig

Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle [1]

von

Michael R. Alvers [2]

Zusammenfassung

Es wurden verschiedene nichtlineare Optimierungsverfahren auf die Geometrieparameter von Dichtemodellen angewendet. Dabei waren diese mit Randbedingungen versehen. Ein Vergleich zeigte die Vor- und Nachteile der einzelnen Algorithmen.

Anhand von zwei synthetischen Testfunktionen wurde das Verfahren Downhill-Simplex mit der (1,10)-CMA-ES verglichen. Das Simplexverfahren zeigte bei weniger als zehn Parametern ein besseres Konvergenzverhalten als die Evolutionsstrategie. Bei höheren Dimensionen konvergierte die Evolutionsstrategie deutlich besser, während Simplex bei mehr als 40 Parametern nicht mehr konvergierte. Die Evolutionsstrategie zeigt unabhängig von der Anzahl der Parameter logarithmische Konvergenzgeschwindigkeit.

Die Evolutionsstrategie, genetische Algorithmen, Simulated-Annealing, Threshold-Accepting und der Deluge-Algorithm wurden danach anhand eines zweidimensionalen Salzstockmodells untersucht. Dabei wurden die Geometrieparameter optimiert und die Dichten des Modells konstant gehalten. Die 52 zu optimierenden Parameter wurden mit Randbedingungen versehen.

Die (1,10)-CMA-ES benötigte im Durchschnitt die wenigsten Funktionsaufrufe und erreichte die besten Qualitäten. Ausgehend von verschiedenen Startmodellen konnten ähnliche Lösungen, welche alle sehr gute Qualitätswerte hatten, gefunden werden. Rekombination verbesserte die Konvergenzeigenschaften immer. Evolutionsstrategie war stabil gegenüber numerischen Variationen.

Genetische Algorithmen konvergierten zu Beginn des Optimierungsprozesses schneller als alle anderen Verfahren, erreichten aber nie die von der Evolutionsstrategie gefundenen besten durchschnittlichen Qualitätswerte.

Downhill-Simplex, Simulated-Annealing, Threshold-Accepting und der Deluge-Algorithm zeigten insgesamt schlechteres Konvergenzverhalten als die Evolutionsstrategie und die genetischen Algorithmen.

---

[1] Dissertation am Fachbereich Geowissenschaften der Freien Universität Berlin zur Erlangung des Grades eines Doktors der Naturwissenschaften. Tag der Disputation: 13. Februar 1998, Gutachter: Prof. Dr. H.-J. Götze und Prof. Dr. I. Rechenberg.

[2] Anschrift des Verfassers: Dipl.-Geophys. Michael R. Alvers, Institut für Geologie, Geophysik und Geoinformatik, Freie Universität Berlin, Malteserstr. 74-100, D-12249 Berlin.

Eine Anwendung der Evolutionsstrategie auf ein Salzstockmodell zeigte die Probleme der Optimierung in drei Dimensionen. Dennoch konnte das Modell in Hinblick auf die gravimetrische Anpassung deutlich verbessert werden.

Abstract

Various non-linear optimization techniques were applied to constrained geometry parameters of density models. A comparison showed advantages and disadvantages of each method.

By means of synthetical test-functions the downhill-simplex-method has been compared to the (1,10)-CMA-ES. For numbers of parameters smaller ten, better convergence behaviour was achieved with the simplex-method. However, at higher dimensions evolution-strategy converges better than the simplex-method. Simplex does not converge for dimensions greater than 40. Independent of the number of parameters, the evolution-strategy showed a logarithmic convergence speed.

The evolution-strategy, genetic algorithms, simulated annealing, threshold accepting and the deluge algorithm were investigated using a two-dimensional salt dome model. Here only the geometry of the model was optimized, the densities were held constant. The 52 parameters were constrained.

In average the (1,10)-CMA-ES required the less function evaluations and gained the best qualities. Independent of the start configuration evolution-strategy found similar solutions, whose qualities were fairly good. Recombination always improved the results in terms of convergence behaviour. The evolution-strategy was stable against numerical variations.

At the beginning of the optimization process genetic algorithms converge faster than all other methods, but they never reach the average quality values which were gained by evolution-strategy.

In summary the simplex-method, genetic algorithms, simulated annealing, threshold accepting and the deluge algorithm showed worse convergence behaviour than evolution-strategy and genetic algorithms.

An application of evolution-strategy to a three dimensional salt dome model showed the problems encountered in three dimensions. Despite this, in terms of gravity fit, the model could be improved noticeable.

Resumen

En el presente trabajo se aplican diferentes métodos de optimización no lineales, con condiciones de borde, a modelos de densidad. Una comparación muestra las ventajas y desventajas de cada método.

Mediante dos funciones sintéticas se compara el método Downhill- Simplex con el (1,10)-CMA-ES. Para un número de parámetros menores que 10, el método Simplex muestra una

mejor convergencia que el método de estrategias de evolución. Para mayores dimensiones el método de estrategias de evolución converge mucho mejor que Simplex; mientras que para más de 40 parámetros el método Simplex deja de converger. Las estrategias de evolución muestran, independientemente del número de parámetros, una velocidad de convergencia tipo logarítmica.

Utilizando un modelo bidimensional de un domo de sal, se analizan comparativamente los métodos de estrategias de evolución, algoritmos genéticos, Simulated- Annealing, Threshold-Accepting y Deluge-Algorithm. Para esto se optimizan sólo los parámetros geométricos del modelo, mientras que las densidades permanecen constantes. Se aplicaron condiciones de borde a los 52 parámetros optimizados.

El método (1,10)-CMA-ES en promedio necesita el menor número de llamados a la función para alcanzar la mejor calidad. El método encuentra a partir de modelos iniciales diferentes soluciones similares, que tienen todas valores de calidad muy buenas. El método de estrategias de evolución es estable frente a las variaciones numéricas. La recombinación mejora en todos los casos las características de la convergencia.

Los algoritmos genéticos convergen al principio mejor que cualquier otro método, pero no logran alcanzar los mejores valores promedios de calidad encontrados con las estrategias de evolución.

Los métodos Downhill-Simplex, Simulated-Annealing, Threshold-Accepting y Deluge-Algorithm muestran en general una peor convergencia que las estrategias de evolución y los algoritmos genéticos.

La aplicación del método de estrategias de evolución a un modelo tridimensional muestra los problemas de la optimización en el espacio tridimensional. A pesar de ellos, un modelo de un domo de sal puede ser notablemente mejorado en relación a su ajuste a datos gravimétricos.

Inhaltsverzeichnis

1

Einleitung

Das Ziel der Arbeit ist die Anwendung von Optimierungsverfahren, die keine Ableitung der Qualitätsfunktion benötigen, um die interaktive Modellierung geophysikalischer Modelle zu unterstützen. Besonderer Wert wird dabei auf die Berücksichtigung von Randbedingungen gelegt.

Die Entwicklungen im Bereich der elektronischen Datenverarbeitung in den letzten Jahren ermöglichen es heute, immer komplexere Abbilder der Natur mit Hilfe von Computern zu erstellen. Angewandt auf Fragestellungen der Geowissenschaften führt dies zum besseren Verständnis rezenter Strukturen und dadurch zum besseren Verständnis von Prozessen, die zu diesen Strukturen geführt haben. Dies ist von Bedeutung, da ein gutes Verständnis geologischer Abläufe zu einer besseren Rekonstruktion der Entwicklung des Erdkörpers führt. Dadurch wird das Auffinden von Rohstoffen wie Erdgas, Erdöl und Mineralen, welche in diesen Prozessen entstandenen sind, erleicht.

Besonders im Hinblick auf eine in der nahen Zukunft zu erwartende rasante Entwicklung im Bereich der dynamische Modellierung erdgeschichtlicher Abläufe und der *joint inversion* verschiedener geowissenschaftlicher Methoden ist es heute erforderlich, die computertechnischen Grundlagen für diese Entwicklungen zu schaffen.

Ein wesentlicher Beitrag zur Integration unterschiedlicher Informationen in einem Modellierungsprogramm wird an unserem Institut durch die intensive Arbeit mit Geoinformationssystemen (IOGIS) im Rahmen eines Habilitationsvorhabens geleistet. Dieses Vorhaben stellt die konsequente Fortführung der von H.-J. Götze mit eingeleiteten interaktiven gravimetrischen Modellierung dar. Es wird bei diesem Konzept der Tatsache Rechnung getragen, daß Vorstellungskraft zusammen mit Expertenwissen nach wie vor Motor jeder wissenschaftlichen Innovation ist. Eine der „Lücken", die heute noch bestehen, ist die interaktive Konstruktion realitätsnaher 3D-Modelle. Daher soll u.a. ein Werkzeug entwickelt werden, mit dem geologische Interpretationsideen schnell und zuverlässig in zwei- und dreidimensionale Modelle umzusetzen werden konnen. Das im Rahmen dieser Arbeit entwickelten Programm DARWIN++ versucht, diesen Anforderungen gerecht zu werden.

Ein anderer, noch wenig untersuchter Aspekt ist die Optimierung physikalischer und geometrischer Parameter eines Modells mit Hilfe nicht linearer Optimierungsmethoden. Die Frage nach der Plausibilität eines Modells kann nur soweit beantwortet werden, wie geprüft

werden kann, ob es bekannten Randbedingungen genügt. Randbedingungen besonderer Art sind synthetisch berechnete Felder, die mit gemessenen übereinstimmen müssen. Es sollte daher möglich sein, ein Modell im Rahmen gegebener Plausibilitätsgrenzen, in denen sich ein Parameter bewegen darf, unter Berücksichtigung bestimmter Aspekte, wie z.B. der Anpassung berechneter an gemessene Daten, automatisch zu optimieren. Deshalb wird ein besonderer Schwerpunkt dieser Arbeit auf der Anwendung und dem Test nichtlinearer Optimierungstechniken für die Parameteroptimierung liegen.

Die vorliegende Dissertation entstand im Rahmen des u.a. vom Bundesministerium für Forschung und Bildung (BMBF) finanzierten Forschungsverbundprojektes Evolutionäre Algorithmen in der Bioinformatik. Das Teilprojekt EVOTECH des Verbundes hatte das Ziel, Algorithmen aus der Bioinformatik, die in den 60er Jahren von I. Rechenberg entwickelt wurden, theoretisch zu begründen, weiterzuentwickeln und an praktischen Optimierungsproblemen zu testen. Deshalb sind die Evolutionsstrategien in dieser Arbeit von zentraler Bedeutung. Die Zusammenarbeit mit der Wissenschaftlergruppe von I. Rechenberg der Technischen Universität Berlin schaffte ideale Bedingungen für die Arbeit mit diesen Verfahren.

Die in vorangegangenen Untersuchungen gewonnenen Erfahrungen über die Anwendbarkeit der Algorithmen legen die Verwendung dieses Verfahrens nahe. Ein Vergleich der Evolutionsstrategie mit anderen modernen Optimierungstechniken soll die Leistungsfähigkeit der Evolutionsstrategie prüfen und die Anwendbarkeit für die erweiterten Fragestellungen im interaktiven Modellierungsprozeß zeigen.

Die Zusammenarbeit mit der Firma BEB Erdgas und Erdöl im Rahmen des o.g. Forschungsvorhabens machte es möglich, bei der Softwareentwicklung auch Anforderungen der industriellen Praxis zu berücksichtigen. Die untersuchten Optimierungsverfahren wurden an einem gravimetrischen Modell einer Salzstruktur erprobt und analysiert.

Alle Algorithmen zur Modellierung und Optimierung wurden in einem Modellierungs- und Konstruktionswerkzeug implementiert. Eine objektorientierte Programmarchitektur, die mit Hilfe der Programmiersprache C++ realisiert wird, soll spätere Weiterentwicklungen ermöglichen und Betriebssicherheit sicherstellen. Dieses Konzept soll auch die Programmierung interaktiver Modellierungsverfahren mit Echtzeitberechnung und Echtzeitvisualisierung erleichtern. Eine klare Gestaltung der graphischen Benutzerschnittstelle des Programms soll das Erstellen und Optimieren von gravimetrischen Modellen vereinfachen und den Benutzer weitestgehend von Konstruktionsaufgaben auf Papier befreien.

2

Vorwärtsrechnung

Aufgrund der Nichteindeutigkeit der Potentialverfahren kann aus einem gemessenen Schwerefeld nicht direkt auf die Quelle geschlossen werden. Deshalb werden Verfahren benötigt, die es erlauben, das Potential und seine Ableitungen eines geologischen Modells zu berechnen. Im folgenden soll die Herleitung der später verwendeten Formeln zur Berechnung der gravitativen Wirkung von zwei- und dreidimensionalen Objekten mit homogener Dichteverteilung auf einem beliebigen Punkt im Raum beschrieben werden.

2.1 Polygonaler Querschnitt

Die Berechnung der gravitativen Wirkung eines vertikalen Schnitts durch eine Struktur, die in einer Richtung unendlich ausgedehnt ist, wurde zuerst von Talwani et al. (1959) beschrieben. Grant und West (1965) veröffentlichten eine effizientere Formulierung. Algorithmische Verbesserungen wurden von Won und Bevis (1987) vorgeschlagen. Ivanov entwickelte eine zu Won und Bevis identische Formulierung bereits in den 50er Jahren (Ivanov, 1955). Die z-Komponente des Schwerevektors in Bezug auf einen beliebigen Punkt P kann mit diesem Verfahren mit der kompakten Gleichung 2.1 beschrieben werden. Die Abbildung 2.1 zeigt die geometrischen Verhältnisse.

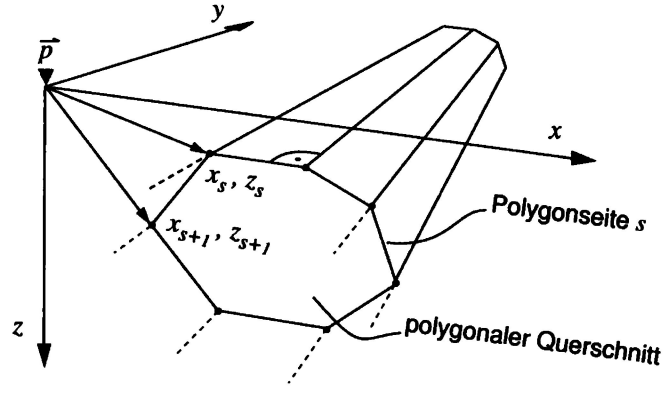


Abbildung 2.1: Die Schwerewirkung eines polygonalen Querschnitts durch ein lang ausgedehntes Objekt wird am Punkt P berechnet. Erläuterung der geometrischen Verhältnisse der Gleichungen 2.1 - 2.5 auf Seite 4.

Die Gleichung 2.1 beschreibt die Lösung ein Linienintegrals um das jeweils betrachtete Polygon.

$$g = \rho\gamma \sum_{i=1}^{l} (A_s B_s - C_s D_s) \tag{2.1}$$

mit :

γ	Gravitationskonstante
ρ	Dichte
l	Anzahl der Polygonseiten
s	laufender Index über Polygoneckpunkte

$$A_s = \ln\left[\frac{x_{s+1}^2 + z_{s+1}^2}{x_s^2 + z_s^2}\right] \tag{2.2}$$

$$B_s = -2\left[\frac{(z_{s+1} - z_s)\, z_s(x_{s+1} - x_s) - x_s(z_{s+1} - z_s)}{(x_{s+1} - x_s)^2 - (z_{s+1} - z_s)^2}\right] \tag{2.3}$$

$$C_s = -2\left[\frac{(x_{s+1} - x_s)\, z_s(x_{s+1} - x_s) - x_s(z_{s+1} - z_s)}{(x_{s+1} - x_s)^2 - (z_{s+1} - z_s)^2}\right] \tag{2.4}$$

$$D_s = \arctan\left[\frac{(x_s z_{s+1}) - (z_s x_{s+1})}{(x_s x_{s+1}) - (z_s z_{s+1})}\right] \tag{2.5}$$

Sie läßt sich sehr einfach programmieren und ihre numerische Berechnung ist mit wenig Rechenaufwand verbunden. Durch algorithmische Formulierungen kann eine Beschleunigung erreicht werden, indem Polygonseiten, die aneinander grenzen, nur einmal berechnet werden.

Zweidimensionale Berechnungen sollten nur angewendet werden, wenn die betrachtete Struktur wesentlich länger als breit ist. Ein Fünffaches zwischen Länge und Breite ist je nach erforderlicher Genauigkeit für viele Fälle ausreichend (siehe z.B. Tsuboi (1983) oder Telford (1990)). Der Schnitt sollte außerdem ungefähr senkrecht zur Längsachse des Objektes verlaufen. Werden diese Anforderungen nicht eingehalten, gilt die zweidimensionale Formel nicht und es treten Ungenauigkeiten auf. Mathematische Formulierungen, die diese Ungenauigkeiten der Länge-Breite-Verhältnisse berücksichtigen (Rasmussen und Pedersen, 1979; Shuey und Pasquale, 1973; Busby, 1987), erlauben die Berechnung von Objekten, die nicht wesentlich länger als breit sind, deren Querschnitt sich in y-Richtung aber nicht ändert (siehe Abbildung 2.1). Diese Formulierungen werden oft als „zweieinhalbdimensional“ bezeichnet. Auch derartige zweieinhalbdimensionale Strukturen erlauben nur eine eher unrealistische Beschreibung der meisten geologischen Verhältnisse. Erst durch eine echte dreidimensionale Repräsentation können Modelle geologischer Strukturen adäquat behandelt werden.

2.2 Polyeder

Erste Arbeiten zur Berechnung der ersten Ableitung des Potentials dreidimensionaler Strukturen lieferten Talwani und Ewing (1960), Bott (1960), Corbato (1965), Cordell und Henderson (1968) sowie Mader (1959). Sie entwickelten z.B. Formeln für die Berechnung von

Prismen mit horizontalen Deckflächen oder Quadern. Auch mit solchen Elementarkörpern können geologische Strukturen nur unbefriedigend genau beschrieben werden, bzw. es müssen sehr viele dieser Elementarkörper zur Approximation der zu beschreibenden Strukturen benutzt werden, was wiederum zu großem Rechenaufwand führt. Auch die numerische Lösung auftretender Integrale ist aufgrund des stark erhöhten Rechenaufwands unbefriedigend. Erst in den 70er Jahren gab es Arbeiten, die eine Berechnung vertikaler Prismen mit Dreiecksgrundflächen, deren Deckflächen beliebig orientiert sein können, beschrieben (Woodward, 1975). Götze (1976), Okabe (1979) und Barnett (1976) leiteten Formeln für die Berechnung der Schwere und ihrer Ableitungen von Polyedern mit homogener Dichteverteilung ab. Hier soll die Formulierung von Götze (1976) vorgestellt werden, mit der das Potential (Geoid), die Komponenten des Schwerevektors, die Richtungsableitungen sowie die magnetische Wirkung mit remanenter und induzierter Magnetisierung eines Polyeders basierend auf einem Integralkern „baukastenartig" berechnet werden kann. Im folgenden werden die wesentlichen Teile der Herleitung beschrieben. Die abgeleiteten Formeln haben für Polyeder Gültigkeit, welche eine homogene Dichteverteilung (oder Suszeptibilitätsverteilung) im Innenraum haben und deren Oberfläche durch eine geschlossene polygonale Vernetzung mit konstanten Flächennormalen beschrieben werden kann. Das Potential $U(P)$ eines solchen Objektes an einem beliebigen Punkt P ergibt sich durch Integration über die Volumenelemente dV, die sich im Abstand $|\vec{r}_2 - \vec{r}_1|$ vom Meßpunkt P befinden (Abbildung 2.2).

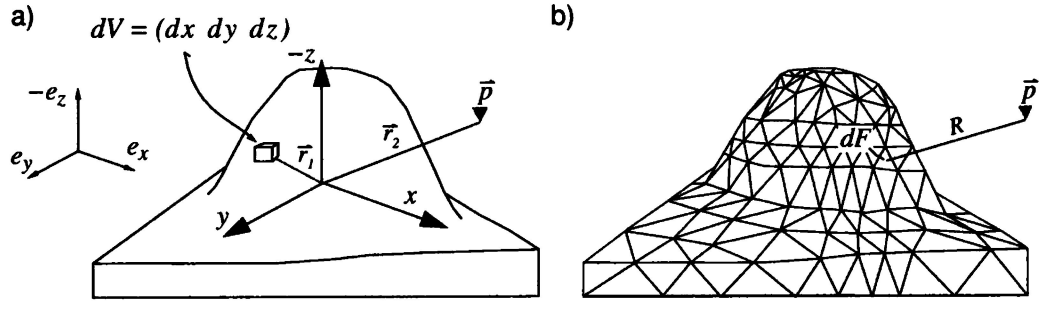


Abbildung 2.2: Ein betrachtetes Objekt wird in Volumenelemente dV zerlegt, die sich im Abstand $|\vec{r}_2 - \vec{r}_1|$ vom Meßpunkt P befinden (a). Die gleiche Struktur wird durch eine Triangulierung der Oberfläche beschrieben (siehe Text).

Das Potential $U(P)$ kann wie folgt formuliert werden:

$$U(P) = \rho\,\gamma \iiint_{Volumen} \frac{dV}{|\vec{r}_2 - \vec{r}_1|} \tag{2.6}$$

mit :

γ	Gravitationskonstante
ρ	Dichte

Die Komponenten des Schwerevektors $\vec{g}$ ergeben sich aus der partiellen Ableitung des Potentials $U(P)$ nach den Richtungen ϵ_x, ϵ_y und ϵ_z:

$$\vec{g} = \left(\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, \frac{\partial U}{\partial z}\right). \tag{2.7}$$

Die z-Komponente des Schwerevektors ist demnach:

$$g_z = \frac{\partial U}{\partial z} = \rho\,\gamma \iiint\limits_{Volumen} \left(\frac{\partial}{\partial z}\right) \frac{1}{|\vec{r}_2 - \vec{r}_1|}\, dV\,. \tag{2.8}$$

Die Lösung dieses Integrals wird durch eine Koordinatentransformation des Bezugssystems in die Ebene des jeweils betrachteten Polygons sowie durch die Anwendung der Integralsätze von Gauß und Green ermöglicht. Durch die Anwendung des Gaußschen Satzes wird das Volumenintegral zunächst in ein Integral über die Oberfläche des Polyeders umgewandelt und man erhält:

$$g_z = \rho\,\gamma \oiint\limits_{Oberfläche} \cos(\vec{n}, \vec{e}_z)\, \frac{1}{R}\, dF \tag{2.9}$$

wobei $\vec{n}$ die Flächennormale des Flächenstücks dF, $\vec{e}_z$ ein Einheitsvektor in z-Richtung und R der Abstand des Flächenelements dF von P ist. Wenn sich das betrachtete Objekt durch eine geschlossene polygonale Vermaschung beschreiben läßt, für dessen Teilflächen dF_i der $\cos(\vec{n}, \vec{e}_z)$ konstant ist, kann dieser Term vor das Integralzeichen gezogen werden. Die Summation aller k Teilflächen ergibt die Gesamtwirkung. Daraus folgt:

$$g_z = \sum_{i=1}^{k} \rho\,\gamma\, \cos(\vec{n}_i, \vec{e}_z) \overbrace{\oiint\limits_{Oberfläche} \frac{1}{R}\, dF_i}^{verbleibendes\ Integral\ I}. \tag{2.10}$$

Das Integral I aus Gleichung 2.10 wird durch die Anwendung des Satzes von Green in ein Linienintegral transformiert, welches geschlossen gelöst werden kann. Das Ergebnis ist eine dem zweidimensionalen Fall ähnliche, einfache Gleichung:

$$g_z = \rho\,\gamma \sum_{i=1}^{k} \cos(\vec{n}_i, \vec{e}_z) \left[\sum_{s=1}^{l} \delta_1\, d_{is}\, LN_{is} + D_i \sum_{s=1}^{l} \delta_1\, ARC_{is} \right] \tag{2.11}$$

mit :

$$LN_{is} = \ln\left(\frac{a_{is} + R_{isa}}{b_{is} + R_{isb}}\right)$$

und

$$ARC_{is} = \arctan\Bigg(\underbrace{\frac{r_{isa}^2 + a_{is}R_{isa}}{d_{is}D_i}}_{A}\Bigg) - \arctan\Bigg(\underbrace{\frac{r_{isb}^2 + b_{is}R_{isb}}{d_{is}D_i}}_{B}\Bigg) = \arctan\left(\frac{A - B}{1 + AB}\right) + \delta_2\pi$$

wobei :

$$R_{isa} = \sqrt{D_i^2 + d_{is}^2 + a_{is}^2}$$

$$R_{isb} = \sqrt{D_i^2 + d_{is}^2 + b_{is}^2}$$

$$r_{isa}^2 = a_{is}^2 + d_{is}^2$$
$$r_{isb}^2 = b_{is}^2 + d_{is}^2$$

$$\delta_1 = \begin{pmatrix} +1 \\ -1 \end{pmatrix} \vec{n}_{is} \text{ zeigt in den Halbraum, der die Projektion von } P \begin{pmatrix} \text{enthält} \\ \text{nicht enthält} \end{pmatrix}$$

$$\delta_2 = \begin{cases} 0 & : \quad AB > -1 \\ +1 & : \quad AB < -1 \; und \; B < 0 \\ -1 & : \quad AB < -1 \; und \; B > 0 \end{cases}$$

k	Anzahl der Polygone,
l	Anzahl der Polygonseiten,
i	Laufindex über die Polygone,
s	Laufindex über die Polygonseiten,
$\vec{n}_i$	Flächennormale des Polygons,
$\vec{e}_z$	Einheitsvektor in z – Richtung.

Die Längen a_{is} und b_{is} erhalten positive Vorzeichen, wenn P^*, P^{**} und $\vec{a_{is}}$ bzw. $\vec{b_{is}}$ die gleiche Orientierung wie das betrachtete Polygon haben, negative andernfalls. Die Abbildung 2.3 zeigt die geometrischen Verhältnisse.

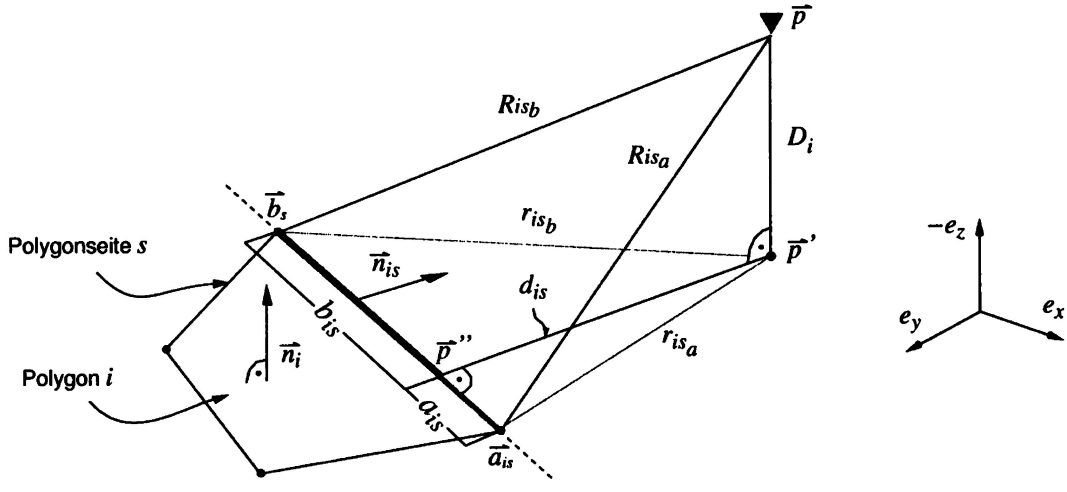


Abbildung 2.3: Zur Erläuterung der Gleichung 2.11. P wird orthogonal in die Ebene des Polygons und der so entstehende Punkt P^* wiederum orthogonal auf die jeweils betrachtete Polygonseite s projiziert. Die Seitennormale zeigt in den Halbraum, in dem sich die Projektion P^* der Station P befindet. Die Größe a_{is} erhält ein positives, b_{is} ein negatives Vorzeichen.

Alle benötigten Größen können durch einfache Vektoroperationen berechnet werden. Der Abstand D_i wird durch das Skalarprodukt der Vektoren $\vec{a}_{is} - P$ und der Flächennormalen $\vec{n}_i$ berechnet (Foley et al., 1993). Die Abbildung 2.4 erläutert die Gegebenheiten für das in Abbildung 2.3 gezeigte Beispiel.

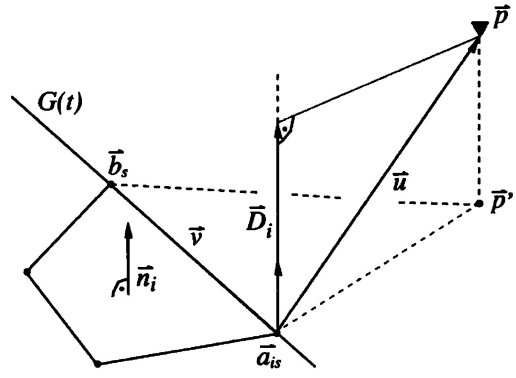


Abbildung 2.4: Der kürzeste Abstand des Punktes P von einer Ebene ergibt sich durch die Projektion des Vektors $\vec{u} = P - \vec{a}_{is}$ auf die Normale $\vec{n}_i$ der betrachteten Fläche. Die Länge des Vektors $\vec{D}_i$ ist die gesuchte Größe.

Der Punkt P^* kann mit $P^* = P - D_i\ \vec{n}_i$ konstruiert werden. Die Projektion des Punktes P^* auf die jeweilige Polygonseite wird mit Hilfe der kürzesten Distanz von P^* zur Geraden

$$G(t) = \vec{a}_{is} + t\ \vec{v}$$

berechnet. Aufgrund der Orthogonalitätsbedingung der Projektion gilt:

$$P^* - (\vec{a}_{is} + t\ \vec{v}) = 0$$

damit folgt für P^{**}

$$\begin{aligned} P^{**} &= \vec{a}_{is} + \frac{(P^* - \vec{a}_{is}) \cdot \vec{v}}{\vec{v} \cdot \vec{v}} \cdot \vec{v} \\ \text{mit :} & \\ \vec{v} &= \vec{b}_{is} - \vec{a}_{is} \end{aligned}$$

Auf die Behandlung von Sonderfällen, wie z.B. $d_{is} = 0$ oder $D_i = 0$, soll hier nicht eingegangen werden. Detailierte Informationen zur effizienten Implementierung können bei Götze und Lahmeyer (1988) und Holstein und Ketteridge (1996) nachgelesen werden.

In der Praxis werden die Oberflächen modellierter Strukturen mit Dreiecken beschrieben. Dies garantiert, daß $\cos(\vec{n}_i, \vec{e}_z)$ auch bei einer späteren interaktiven Bearbeitung (Götze, 1984) immer konstant ist. Außerdem gibt es im Bereich der Computergrafik viele Algorithmen zur Visualisierung und Speicherung von Dreiecksgittern (siehe z.B. Neider et al. (1993)), so daß sich Dreiecksgitter ideal zur Programmierung der Formel eignen.

3

Verwendete Optimierungsmethoden

3.1 Allgemeines Konzept

Eine grundlegende Einführung in Konzepte der Optimierung kann hier nicht gegeben werden. Wichtige Arbeiten, die eine umfassende Beschreibung der Problematik von Optimierungsverfahren geben, sind z.B. bei Rechenberg (1994) oder Goldberg (1989) zu finden. Hier sollen lediglich die wichtigsten Begriffe der gravimetrischen Modelloptimierung erklärt werden, die für das Verständnis der Arbeit erforderlich sind.

Die Abbildung 3.1 zeigt das Flußdiagramm eines Optimierungsprozesses. Parameter, die den Aufbau eines zu optimierenden Objektes bestimmen, werden durch die Anwendung einer Optimierungstechnik derart verändert, daß eine Qualitätsfunktion ein Optimum einnimmt.

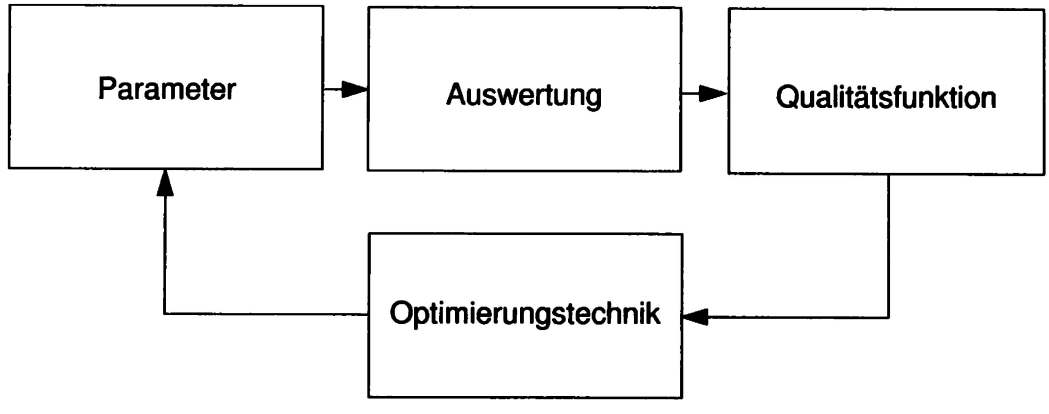


Abbildung 3.1: Flußdiagramm eines Optimierungsprozesses. Modellparameter definieren ein Objekt, welches optimiert werden soll. Eine Optimierungstechnik stellt die Parameter derart ein, daß die Qualitätsfunktion eine Optimum einnimmt.

Einen typischen Optimierungsprozeß, der diesem Schema genügt, stellt das interaktive Modellieren eines geophysikalischen Modells mittels einer geeigneten grafischen Benutzerschnittstelle und eines Rechners dar. Eine auf geowissenschaftliche Fragestellungen bezogene Einführung in Konzepte des interaktiven Modellierens findet man z.B. in Götze (1984). Die Abbildung 3.2 zeigt das Diagramm angewendet auf gravimetrische Modelloptimierung. In 3.2 enthält a) die zu optimierenden Parameter Dichten ρ_i und die Modellkoordinaten, b) enthält die Methoden zur Berechnung der Schwere (siehe Abschnitt 2.1 und 2.2) und c) die Qualitätsfunktion, die hier mit der Summe der Residuenquadrate der gemessenen und berechneten Daten berechnet wird.

Ziel ist es, Optimierungsverfahren zur Unterstützung des interaktiven Modellierungsprozesses einzusetzen, mit denen die zu optimierenden Parameter innerhalb vorgegebener Grenzwerte mit Hilfe eines Algorithmus einstellen, daß die Qualitätsfunktion ein Minimum einnimmt, d.h. die Differenzen zwischen gemessenen und berechneten Werte möglichst gering werden.

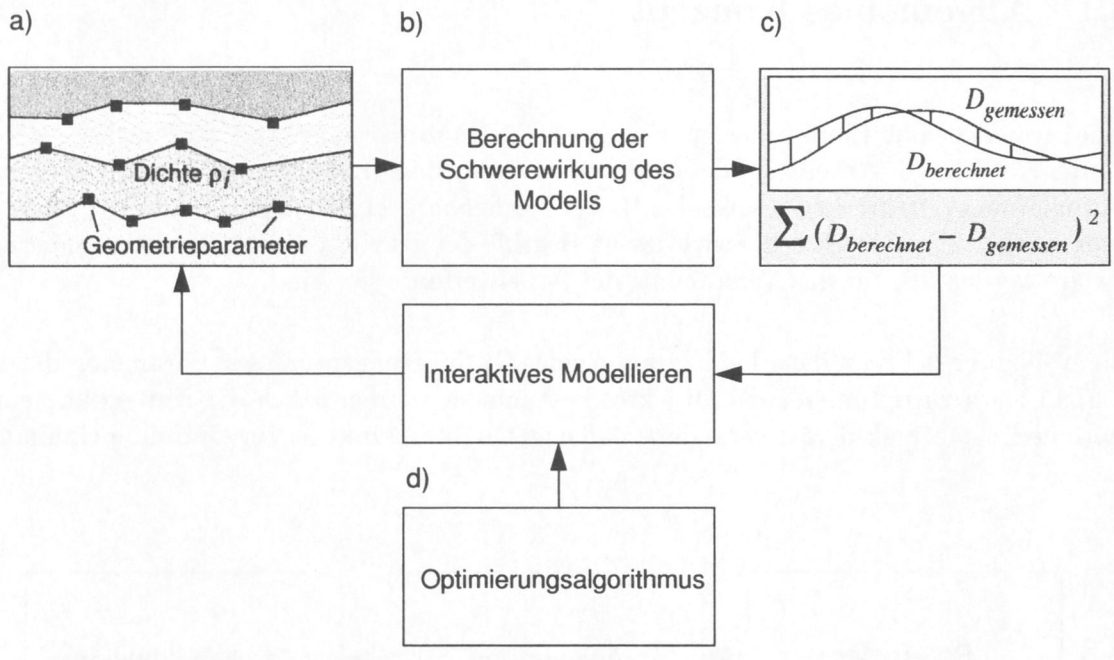


Abbildung 3.2: Das Optimierungskonzept angewendet auf die Optimierung gravimetrischer Modelle. Ein Optimierungsalgorithmus soll interaktives Modellieren unterstützen.

Wird der Prozeß von a) über b) über c) zurück nach a) einmal durchlaufen, spricht man von einer Iteration. Die Qualität des Modells muß dann i.a. einmal durch Auswertung der Qualitätsfunktion berechnet werden. Daher wird im folgenden eine Iteration auch als ein Funktionsaufruf bezeichnet. Daraus ergibt sich die Frage, unter welchen Bedingungen dieser iterative Prozeß abgebrochen werden kann. Die möglichen Kriterien sind

- das Erreichen einer vorher festgelegten Qualität,

- das Erreichen einer vorher festgelegten Anzahl von Iterationen,
- das Erreichen einer vorher festgelegten Qualitätsdifferenz zwischen den Iterationen n und $n-1$ oder
- das interaktive Eingreifen in den Prozeß.

Auch das Auftreten von Fehlern, z.B. Überschneidungen der Modellgeometrie bei der Optimierung von Koordinaten, muß als mögliches Kriterium zum Terminieren des Algorithmus zur Verfügung stehen, falls derartige Fehler nicht korrigierbar sind. In einem Anwenderprogramm sollten dem Benutzer alle diese Möglichkeiten zur Verfügung stehen.

Anforderungen an den Optimierungsalgorithmus

Da Lösungen der Potentialverfahren nicht eindeutig sind, existieren voneinander verschiedene Parameterkonstellationen, die zu gleichen Qualitäten führen. Daher ist die Lösung der inversen Aufgabe nicht eindeutig, und die Qualitätsfunktion hat mehrere gleichberechtigte Optima. Es muß daher mit großer Wahrscheinlichkeit angenommen werden, daß unter diesen Umständen auch suboptimale Lösungen existieren. Daher sollten Optimierungsverfahren, die zur Optimierung von Potentialmodellen benutzt werden, in der Lage sein, eine globale Suche durchzuführen, d.h. den von n Parametern aufgespannten n-dimensionalen Parameterraum „gleichmäßig" abzusuchen. Das wiederum heißt, daß sie nicht auf lokale Optima vorzeitig konvergieren sollten. Algorithmen, die nach einer Parametervariation nur Verbesserungen der Qualität akzeptieren, können Nebenoptima oft nicht verlassen, da der Gradient der Qualitätsfunktion an einem solchen Punkt nicht in Richtung des globalen Optimums zeigt und jede Variation eine Verschlechterung der Qualität zur Folge hat. Die Abbildung 3.3 erläutert die verwendeten Begriffe am eindimensionalen Beispiel.

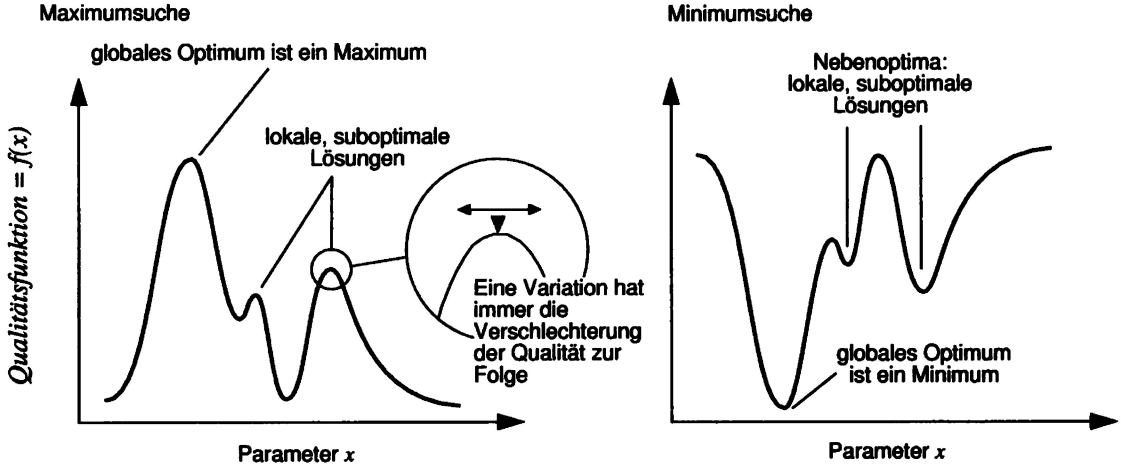


Abbildung 3.3: Zur Erläuterung des Begriffs „lokales Optimum " einer eindimensionalen Qualitätsfunktion. Kleine Variationen des Parameters x in der Nähe eines lokalen Optimums haben eine Verschlechterung der Qualität zur Folge.

Bei Optimierungsverfahren kann man zwischen Methoden, die lokale Informationen der Qualitätsfunktion nutzen und Verfahren, die den Parameterraum stochastisch absuchen, unterscheiden. Während Verfahren, die lokale Informationen der Qualitätsfunktion, z.B. die des Gradienten, nutzen, sehr schnell in Richtung eines möglicherweise lokalen Optimums verlaufen können, besteht aus beschriebenem Grund die Gefahr vorzeitiger Konvergenz auf Nebenoptima. Dies ist plausibel, da der Gradient an einem lokalen Optimum in entgegengesetzter Richtung zum globalen Optimum gerichtet ist. Stochastische Suchverfahren nutzen keine lokalen Informationen der Qualitätsfunktion sondern suchen den Parameterraum auf der Grundlage eines Zufallsprozesses ab. Die Gefahr vorzeitiger Konvergenz auf lokale Optima ist bei diesen Verfahren geringer, da sie prinzipiell lokale Optima verlassen können.

Seit Newton wurden die verschiedensten Optimierungsmethoden entwickelt, die alle ein Ziel verfolgen: die Suche nach dem globalen Optimum (Minimum oder Maximum) einer gegebenen Zielfunktion. Besonders in den letzten dreißig Jahren wurden viele Verfahren publiziert, die den Parameterraum stochastisch absuchen und nach bestimmten Heuristiken gefundene Lösungen akzeptieren bzw. verwerfen und weitersuchen. Es wurden auch Verfahren vorgeschlagen, die den natürlichen Evolutionsprozeß kopieren bzw. Prozesse der statistischen Physik simulieren.

Für diese Arbeit wurden sechs Verfahren ausgewählt, die sie sich für „nichtlineare" Optimierungsprobleme eignen. Von nichtlinearer Optimierung spricht man, wenn es nichtlineare Abhängigkeiten der zu optimierenden Parameter untereinander gibt.

Im folgenden werden die Verfahren

- Monte-Carlo,
- Hill-Climbing,
- Simulated-Annealing,
- Deluge-Algorithmus,
- Threshold-Accepting,
- Downhill-Simplex,
- Genetische Algorithmen und
- Evolutionsstrategien

beschrieben und in ihrem Aufbau verglichen. Die Beschreibung der Algorithmen erfolgt in den folgenden Abschnitten aufgrund der besseren Verständlichkeit in verbaler Form. Die Implementierung der Algorithmen erfolgte in den Programmiersprachen C bzw. C++. Für einen detaillierten Einblick in die Syntax und die Architektur dieser Algorithmen in den Programmiersprachen C und C++ siehe z.B. Lippman (1993); Johnsonbaugh und Kalin (1995); Traister (1993).

3.2 Monte-Carlo-Suche

Das Monte-Carlo-Verfahren (Metropolis et al., 1953) ist ein stochastisches Verfahren, dessen Algorithmus einfach zu programmieren ist. Er liefert für manche Optimierungsprobleme gute Ergebnisse und wird seit der Verfügbarkeit von Computern häufig zur Optimierung technischer Problemstellungen eingesetzt. Der Monte-Carlo-Algorithmus ist die Grundlage für die meisten anderen untersuchten Verfahren. Da die Erklärung der Methode wichtig für das allgemeine Verständnis ist, wird dieser Algorithmus als erster behandelt.

Der Algorithmus des Monte-Carlo-Verfahrens kann wie folgt beschrieben werden:

- Setze die Variable *beste Qualität* zu Beginn der Optimierung auf den Wert 0.
- Führe folgende Iterationsschleife aus:
- Erzeuge eine zufällige Parameterkonstellation, gleichverteilt über den gesamten Parameterraum und werte die Qualitätsfunktion aus. Gleichverteilt heißt hier, daß jeder Punkt im Lösungsraum mit gleicher Wahrscheinlichkeit „gewählt" werden kann.
- Falls die berechnete Qualität **besser** als die *beste Qualität* ist, die bislang erreicht wurde, setze die *beste Qualität* auf den Wert der berechneten Qualität und speichere die Parametereinstellung als die beste bislang erreichte.
- Falls die berechnete Qualität **schlechter** als die *beste Qualität* ist, die bislang erreicht wurde, erzeuge erneut eine Parameterkonstellation.
- Wiederhole die Iterationsschleife, bis ein vorher festgelegtes Kriterium zum Beenden des Prozesses erreicht ist.

In jedem Durchlauf der Iterationsschleife wird ein neuer Parametervektor erzeugt, der vom vorherigen Ergebnis völlig unabhängig ist. Es besteht somit die Möglichkeit vorzeitiger Konvergenz. Jedes erreichte (lokale) Optimum kann prinzipiell verlassen werden, wenn ein Parametervektor erzeugt wird, der eine bessere Qualität hat, als die beste bislang erreichte. Allerdings ist die Wahrscheinlichkeit, eine bestimmte Parameterkonstellation im Lösungsraum (z.B. das globale Optimum) durch zufällige Suche zu finden, sehr gering. Dies verdeutlicht folgende Überlegung: Ein zu untersuchendes Gebiet wird in 1000 Teilbereiche unterteilt. Zum Beispiel kann ein fiktives quaderförmiges Untersuchungsgebiet von $100 \times 100 \times 6\ km$ in $10 \times 10 \times 10 = 1000$ Würfel unterteilt werden. Desweiteren werden zehn (verschiedene) Dichten, z.B. aus dem Intervall von 1.5 - 2.5 g/cm^3, auf die erzeugten Teilbereiche verteilt. Damit ergeben sich 10^{1000} mögliche Anordnungen der zehn Dichten auf die 1000 Teilbereiche. Für die Anzahl der nötigen Versuche ψ, einen bestimmten Zustand mit einer gewissen Wahrscheinlichkeit zu finden, kann gezeigt werden (Woitschach, 1978), daß

$$\psi = \frac{\ln(1-w)}{\ln(m-1/m)}$$

wobei m die Anzahl aller möglichen Zustände und w die Wahrscheinlichkeit ist, einen bestimmten Zustand zu realisieren. Setzt man $w = 95\ \%$, so erhält man $\psi \approx 3 \times m$. Soll ein Zustand mit 95 % Wahrscheinlichkeit gefunden werden, müssen daher 3×10^{1000}

verschiedene Anordnungen realisiert werden. Es wäre daher dreimal günstiger, alle möglichen Zustände zu realisieren, also den gesamten Parameterraum abzusuchen, als den Monte-Carlo-Algorithmus anzuwenden. Die Realisierung und Berechnung der 10^{1000} möglichen Zustände ist aufgrund der enormen Rechenzeit, die man benötigen würde, allerdings praktisch nicht durchführbar. Daraus folgt, daß selbst bei dieser groben Diskretisierung des Untersuchungsgebietes der Iterationsaufwand mit einem Monte-Carlo-Verfahren eine gesuchte Parameterverteilung mit 95 % Wahrscheinlichkeit zu finden, jeden zeitlichen Rahmen sprengen würden. Man spricht hier von np-harten Problemen.

Fazit: die gute Eigenschaft des Monte-Carlo-Verfahrens, weniger der Gefahr der vorzeitigen Konvergenz zu unterliegen, steht der geringen Wahrscheinlichkeit, ein Optimum durch stochastische Suche zu finden, gegenüber. Der Algorithmus kann daher für geophysikalische Aufgaben nur einen globalen Überblick der Qualitätsfunktion geben und eventuell zur Initialisierung für andere Verfahren benutzt werden.

3.3 Hill-Climbing

Der Algorithmus für die Hill-Climbing Methode, häufig auch als *Trial and Error* bezeichnet, ist dem Monte-Carlo-Verfahren prinzipiell ähnlich und kann wie folgt formuliert werden:

- Setze die Variable *beste Qualität* zu Beginn der Optimierung auf den Wert 0.
- Erzeuge eine zufällige initiale Parameterkonstellation gleichverteilt über den gesamten Parameterraum .
- Führe folgende Iterationsschleife aus:
- Variiere die aktuelle Parameterkonstellation durch eine stochastische Veränderung und werte die Qualitätsfunktion aus.
- Falls die berechnete Qualität **besser** als die *beste Qualität* ist, die bislang erreicht wurde, setze die *beste Qualität* auf den Wert der berechneten Qualität und speichere die Parametereinstellung als die beste bislang erreichte.
- Falls die berechnete Qualität **schlechter** als die *beste Qualität* ist, die bislang erreicht wurde, variiere die Parameterkonstellation erneut.
- Wiederhole die Iterationsschleife, bis ein vorher festgelegtes Abbruchkriterium erreicht ist.

Der einzige und entscheidende Unterschied zwischen den beiden Algorithmen ist, daß Hill-Climbing auf den Ergebnissen vorangegangener Iterationen aufbaut, während beim Monte-Carlo-Verfahren jede Parametereinstellung unabhängig von bereits erzeugten Einstellungen realisiert wird. Im Hill-Climbing Algorithmus wird eine Position mit einer zufälligen Veränderung variiert. Ist diese Variation der Parameter richtig auf die Eigenschaften der Qualitätsfunktion abgestimmt, kann das Verfahren gut konvergieren. Die Schwierigkeit besteht in der praktischen Anwendung darin, eine Variationsbreite zu finden, die für das jeweilige Optimierungsproblem und die lokalen Gegebenheiten angemessen ist. Ist die Variationsbreite zu klein, kann sich der Algorithmus auf einem lokalen Optimum

„verfangen", da nur die nähere Umgebung berücksichtigt wird. Ist die Variationsbreite zu groß, kann das globale Optimum übersprungen werden. Für größere Variationsbreiten geht der Algorithmus in den des Monte-Carlo-Verfahrens über, da der Parameterraum „großräumiger" abgesucht wird.

Strategien, die von diesem Algorithmus abgeleitet wurden, verwenden Heuristiken, nach denen die Variationsbreite automatisch den lokalen Gegebenheiten angepaßt wird (Rechenberg, 1973). Hill-Climbing arbeitet ohne derartige Verbesserungen für die meisten praktischen Probleme schlecht.

Die Abbildung 3.4 zeigt einen Vergleich der beschriebenen Verfahren.

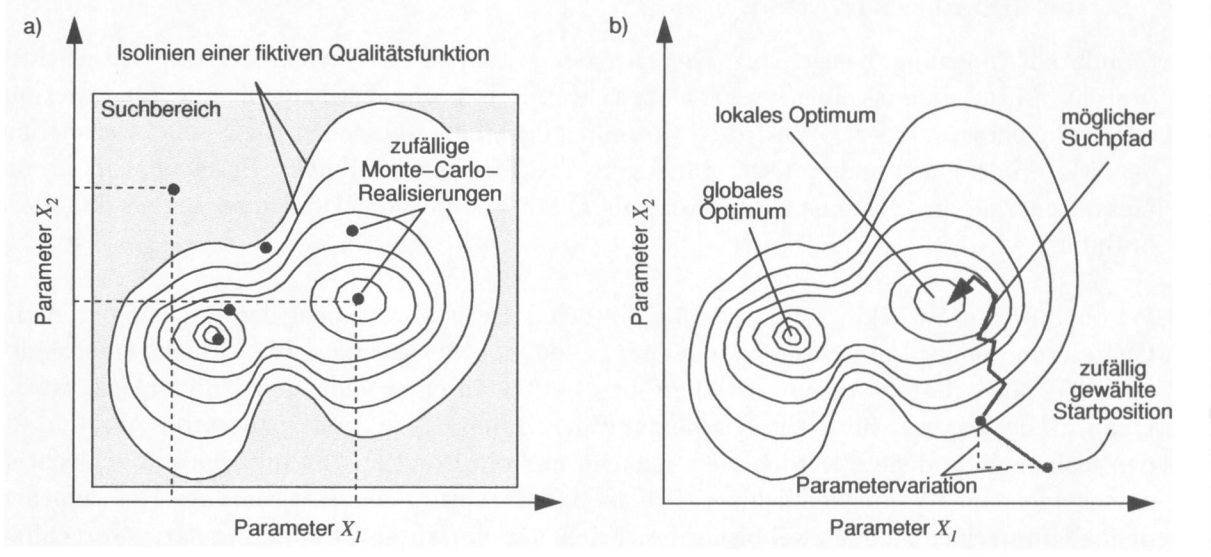


Abbildung 3.4: Während die Monte-Carlo-Methode (a) den gesamten Parameterraum gleichverteilt absucht, baut das Hill-Climbing Verfahren (b) auf erreichten Positionen auf. Gezeigt ist eine fiktive zweidimensionale Qualitätsfunktion, welche durch die Parameter x_1 und x_2 bestimmt wird.

3.4 Simulated-Annealing

Der Simulated-Annealing-Algorithmus (Kirkpatrick et al., 1983) ist den Algorithmen der Monte-Carlo-Methode und des Hill-Climbing ähnlich:

- Setze die Variable *beste Qualität* zu Beginn der Optimierung auf den Wert 0.
- Setze die Variable *Temperatur* zu Beginn der Optimierung auf einen (problemabhängigen) initialen Wert.
- Erzeuge eine initiale Parameterkonstellation und werte die Qualitätsfunktion aus.
- Führe folgende Iterationsschleife aus:
- Variiere die aktuelle Parameterkonstellation durch eine stochastische Veränderung und werte die Qualitätsfunktion aus.

- Falls die berechnete Qualität **besser** als die *beste Qualität* ist, die bislang erreicht wurde, setze die *beste Qualität* auf den Wert der berechneten Qualität und übernimm die Parametereinstellung als Ausgangspunkt für die nächste Iteration. Erniedrige die Variable um einen (problemabhängigen) vorher festgelegten Wert.
- Falls die berechnete Qualität **schlechter** als die *beste Qualität* ist, die bislang erreicht wurde, berechne eine Akzeptanzwahrscheinlichkeit und übernimm die (schlechtere) Realisierung mit dieser Wahrscheinlichkeit als Ausgangspunkt für die nächste Iteration. Ist die berechnete Wahrscheinlichkeit so klein, daß keine Übernahme erfolgt, erzeuge eine neue Parameterkonstellation durch Variation der bestehenden.
- Wiederhole die Iterationsschleife, bis ein vorher festgelegtes Kriterium zum Beenden des Algorithmus erreicht ist.

Simulated-Annealing basiert auf Theorien der statistischen Physik. Die abgeleitete Idee besteht darin, daß in einem sehr heißen Material, große Freiheit für die Orientierung der Kernspinns existiert. Wird das Material abgekühlt, versuchen die Spinns der Atome des Materials einen energetisch günstigen Zustand einzunehmen. Bezeichnet man die Gesamtenergie des Systems (Materials) als Qualitätsfunktion, kann man sagen, daß diese optimiert wird.

Wie beim Monte-Carlo Verfahren sorgen stochastische Variationen der Parameter für die Generierung neuer Parametereinstellungen. Die Qualität des erzeugten Modells entscheidet auch hier über dessen Akzeptanz, allerdings mit einem entscheidenden Unterschied: Ist die Qualität besser, d.h. führt die Auswertung der Qualitätsfunktion zu besseren Werten, als den bislang gefundenen, werden die Lösungen immer akzeptiert. Ist die Qualität schlechter, entscheidet eine Wahrscheinlichkeit P über die Akzeptanz der Realisierung. Der entscheidende Unterschied zu den zwei bisher beschriebenen Verfahren ist, daß Simulated-Annealing damit prinzipiell auch Parametereinstellungen schlechterer Qualität akzeptieren kann. Die Abbildung 3.5 verdeutlicht das Konzept.

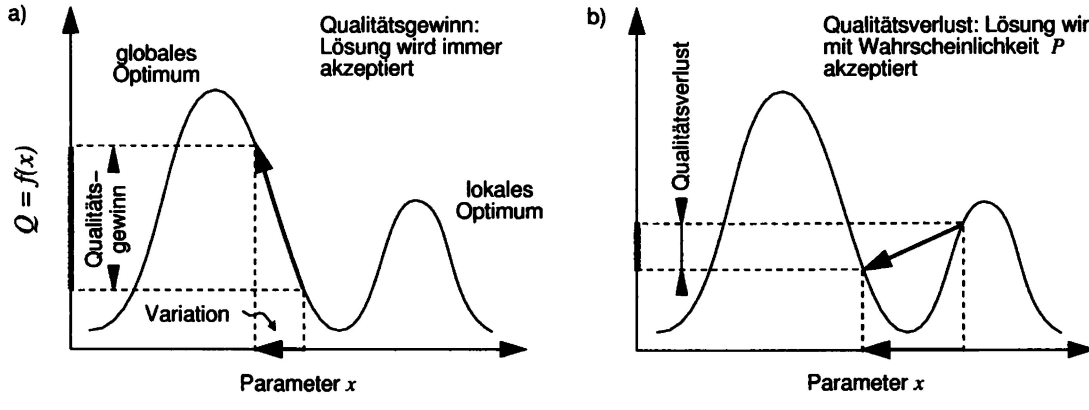


Abbildung 3.5: Variationen der Parameter, die zu besseren Qualitäten führen, werden immer akzeptiert (a). Variationen, die schlechtere Qualitäten zur Folge haben, werden mit Wahrscheinlichkeit P akzeptiert (b), die im Laufe der Optimierung kleiner wird.

Die Wahrscheinlichkeit P, mit der Modelle schlechterer Qualität akzeptiert werden, hängt von zwei Strategieparametern ab:

- vom Qualitätsverlust zwischen Iteration n zu $n-1$ und
- vom algorithmischen Parameter *Temperatur*.

P wird nach folgender Formel berechnet:

$$P = \epsilon^{\frac{Verlust\ an\ Qualität}{Temperatur}}.$$

Die Temperatur wird im Laufe der Optimierung nach einem vorzugebenden Abkühlungsplan (*annealing schedule*) erniedrigt. Ist die Temperatur hoch, so daß P im Durchschnitt gegen 1 geht, wird ein Zufallspfad (*random walk*) im Optimierungsraum beschrieben. Alle Modelle, sowohl besserer als auch schlechterer Qualität, werden akzeptiert, daher ist keine Verbesserung der Qualität zu erwarten. Ist die Temperatur 0, ist auch $P = 0$ und der Algorithmus arbeitet wie Hill-Climbing. Schlechtere als bislang erreichte Qualitäten werden nie akzeptiert. Der *annealing schedule* sollte so aufgebaut sein, daß am Anfang der Optimierung P relativ groß ist, z.B. derart, daß 2/3 aller schlechteren Modelle akzeptiert werden. Im Verlauf der Optimierung sollte P kleiner werden, um damit den Prozeß zu Hill-Climbing „einzufrieren". Mit diesem Konzept kann prinzipiell erreicht werden, daß lokale Optima mit einer gewissen Wahrscheinlichkeit $P > 0$ verlassen werden können.

3.5 Great-Delug-Algorithmus

Ein weiterer stochastischer Suchalgorithmus ist der sogenannte Great-Delug-Algorithmus (Sintflutalgorithmus) nach Dueck (1993). Wie beim Simulated-Annealing-Algorithmus gibt es auch hier eine gewisse Wahrscheinlichkeit dafür, daß ein schlechteres als das beste bisher gefundene Modell als Ausgangspunkt für die nächste Iteration akzeptiert wird. Es kann damit auch bei diesem Algorithmus zu, Rückschritte in der Qualitätsfunktion kommen, um somit eventuell existierende lokale Optima verlassen zu können. Der Autor des oben zitierten Artikels zeigt, daß der Sintflutalgorithmus bei der Optimierung des Problems des Handlungsreisenden dem Simulated-Annealing-Verfahren deutlich überlegen ist.

Der Great-Delug-Algorithmus basiert auf folgender Idee: Während der Variation der Parameter wird der Raum der Qualitätsfunktion (Landschaft) theoretisch langsam mit „Wasser geflutet". Akzeptiert werden nur solche Parametervektoren, deren berechnete Qualitäten auf „Inseln", damit also über dem theoretischen „Wasserspiegel" liegen. Rückschritte sind möglich, solange die Qualität der erzeugten Parametervektoren über dem Wasserspiegel liegt. Die Variationen der Parameter erfolgen auch bei diesem Verfahren durch stochastische Variation. Der Algorithmus kann wie folgt beschrieben werden:

- Setze die Variable *beste Qualität* zu Beginn der Optimierung auf den Wert 0.
- Setze die Variable *Regen* zu Beginn der Optimierung auf einen problemabhängigen initialen Wert größer Null.
- Setze die Variable *Wasserstand* zu Beginn der Optimierung auf einen problemabhängigen initialen Wert größer Null.
- Erzeuge ein initiale Parameterkonstellation und werte die Qualitätsfunktion aus.

- Führe folgende Iterationsschleife aus:
- Variiere die aktuelle Parameterkonstellation durch eine stochastische Veränderung und werte die Qualitätsfunktion aus.
- Falls die berechnete Qualität **über** dem Wasserspiegel liegt, übernimm diese Realisierung als Ausgangspunkt für die nächste Iteration. Erhöhe den Wasserspiegel um einen (problemabhängigen) Wert.
- Falls die berechnete Qualität **unter** dem Wasserspiegel liegt, erzeuge eine neue Parameterkonstellation durch Variation der bestehenden.
- Wiederhole die Iterationsschleife, bis ein vorher festgelegtes Kriterium zum Beenden des Algorithmus erreicht ist.

Dieser Algorithmus hat gegenüber Simulated-Annealing den Vorteil, daß nur ein einziger Strategieparameter (**Regen**) den Verlauf der Optimierung kontrolliert und daher kein vorher festgelegter Plan (annealing schedule) benötigt wird. Es ist lediglich darauf zu achten, daß der Wasserspiegel nicht über den aktuellen Qualitätswert angehoben wird, da damit das Prinzip des Algorithmus verletzt würde. Dies kann durch eine Beschränkung erreicht werden, die eine Korrektur des zu hohen Wasserspiegels vornimmt und ihn auf den aktuellen Qualitätswert oder wenig unterhalb setzt. Tritt dieser Fall oft ein, bedeutet dies, daß der Parameter **Regen** zu groß gewählt wurde. Da dann nur noch Realisierungen zugelassen werden, die über dem Wasserspiegel liegen und damit besser als der momentane Qualitätswert sind, degeneriert der Algorithmus zu Hill-Climbing. Für **Regen** $\rightarrow 0$ geht der Algorithmus in einen „*random walk*" über, da der Wasserspiegel nicht erhöht wird und alle Realisierungen über dem Wasserspiegel akzeptiert werden. Die Abbildung 3.6 verdeutlicht das Prinzip des Algorithmus.

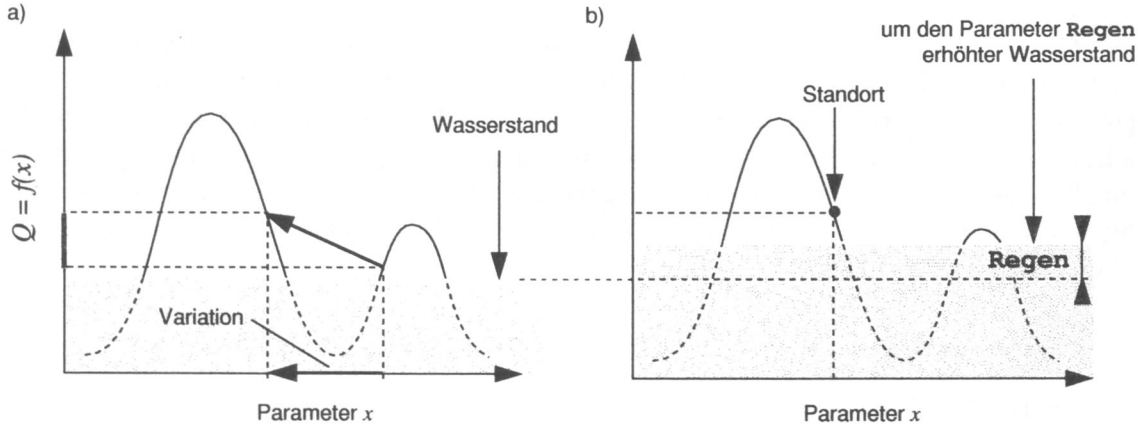


Abbildung 3.6: Bei erfolgreicher Parametervariation (a) wird beim Sintflutalgorithmus der Wasserstand um *Regen* erhöht (b). Variationen, für die gilt: $Q = f(x) < Wasserspiegel$, werden nicht akzeptiert. Vom Standpunkt in (b) ausgehend können Rückschritte gemacht werden.

Die Analogie, daß Wasser eine Landschaft überflutet und sich so Inseln herausbilden, ist auf eine Maximumsuche bezogen. Man sucht den „Gipfel" der Qualitätsfunktion. Für gravimetrische Optimierungen, bei der ein Minimum gesucht wird, muß die Analogie daher negiert werden. Man sucht dann den tiefsten Punkt der Qualitätsfunktion. Auch das Akzeptanzkriterium muß dann entsprechend geändert werden.

3.6 Threshold-Accepting

Dieser Algorithmus von Dueck und Scheuer (1990) basiert ebenfalls auf der grundlegenden Idee, Rückschritte in der Qualitätslandschaft zuzulassen. Hier wird ein Schwellwert benutzt, um zu entscheiden, ob eine Parameterkonstellation bestimmter Qualität akzeptiert wird oder nicht. Wie bei Simulated-Annealing werden Verbesserungen der Qualität immer akzeptiert, Verschlechterungen nur dann, wenn der Verlust an Qualität kleiner als ein bestimmter Schwellwert *threshold* ist. Diese Schwelle wird bei erfolgreichen Schritten verkleinert. Die Abbildung verdeutlicht daß Prinzip.

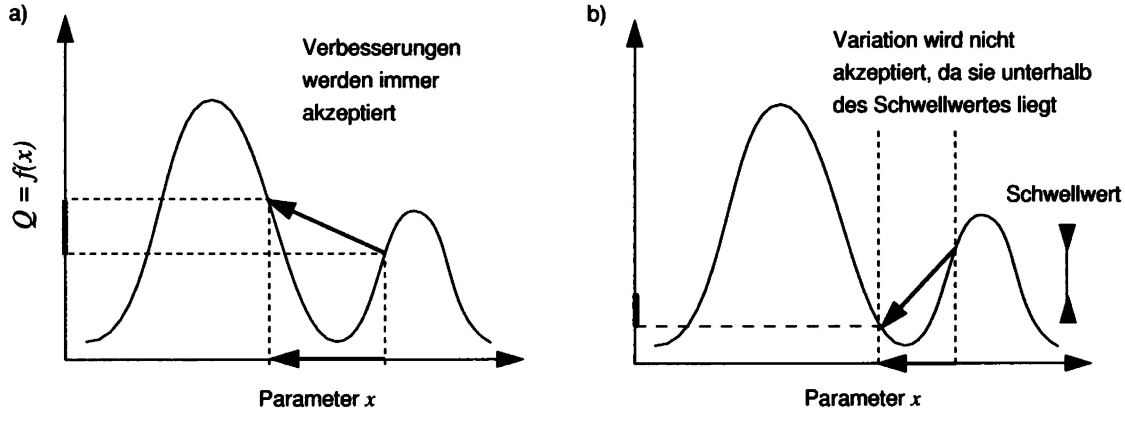


Abbildung 3.7: Beim Threshold-Accepting-Verfahren werden Parametervariationen, die zu besseren Qualitäten führen (a) immer akzeptiert. Solche, die Verschlechterungen zur Folge haben (b), werden nur akzeptiert, wenn sie nicht schlechter sind, als eine vorgegebene Schwelle erlaubt.

Der Algorithmus wird wie folgt formuliert:

- Setze die Variable *beste Qualität* zu Beginn der Optimierung auf den Wert 0.
- Setze die Variable *Schwelle* zu Beginn der Optimierung auf einen (problemabhängigen) initialen Wert größer Null.
- Erzeuge eine initiale Parameterkonstellation und werte die Qualitätsfunktion aus.
- Führe folgende Iterationsschleife aus:
- Variiere die aktuelle Parameterkonstellation durch eine stochastische Veränderung und werte die Qualitätsfunktion aus.

- berechne die Differenz zwischen dieser Qualität und der Qualität der vorangegangenen Iteration.
- Falls die berechnete Differenz **größer** als Null ist, übernimm diese Realisierung als Ausgangspunkt für die nächste Iteration. Erniedrige den Parameter Schwelle um einen (problemabhängigen) Wert.
- Falls die berechnete Differenz **kleiner** als Null ist, übernimm diese Realisierung nur dann als Ausgangspunkt für die nächste Iteration, wenn die Differenz kleiner als der Parameter Schwelle ist.
- Wiederhole die Iterationsschleife, bis ein vorher festgelegtes Kriterium zum Beenden des Algorithmus erreicht ist.

Der Algorithmus ist in seinem Aufbau dem des Simulated-Annealing ähnlich. Er hat aber den Vorteil, daß keine Wahrscheinlichkeit berechnet werden muß. Im Unterschied zum Sintflutalgorithmus bezieht sich die Schwelle auf die lokale Position im Parameterraum und nicht wie der Wasserstand auf die gesamte Qualitätsfunktion.

Ist der Schwellwert sehr groß, z.B. größer oder gleich der aktuellen Qualität, wird ein Zufallspfad beschritten, denn alle Realisationen werden akzeptiert. Ist der Schwellwert für mögliche Rückschritte Null, sind keine Rückschritte möglich und der Algorithmus arbeitet wie Hill-Climbing. Die Schwierigkeit besteht hier zum einen darin, einen für das jeweilige Optimierungsproblem angepaßten Startwert zu finden und zum anderen eine Heuristik zu finden, nach der der Schwellwert verkleinert wird.

3.7 Downhill-Simplex

Eine weitere Methode, die ebenfalls ohne Ableitung der Qualitätsfunktion arbeitet, ist der Downhill-Simplex-Algorithmus (Nelder und Mead, 1965). Dieser Algorithmus hat gegenüber den bisher beschriebenen Verfahren den Vorteil, daß er die Variationen der Parameter den lokalen Gegebenheiten der Qualitätsfunktion anpassen kann. Simplex durchsucht den Raum allerdings nicht global und birgt somit die Gefahr vorzeitiger Konvergenz auf lokalen Optima.

Die Implementierung der Methode ist im Vergleich zu den bisher beschriebenen Verfahren aufwendiger (Press et al., 1992). Hier sollen nur die wesentlichen Strukturen des Algorithmus skizziert werden.

Ausgehend von einem beliebig gewählten Startvektor S, werden parallel zu den Koordinatenachsen n Vektoren P_i nach der Vorschrift

$$P_i = S + \lambda \epsilon_i \ \forall i$$

erzeugt. Darin sind

S *der Startpunkt,*
λ *ein skalarer Parameter und*
ϵ_i *die i Einheitsvektoren der jeweiligen Dimension.*

Die Größe n ist hierbei die Dimension des Parameterraumes. Der Parameter λ bestimmt die Größe des Simplex zu Beginn des Optimierungsprozesses. Die $n + 1$ Vektoren werden nun folgender Prozedur unterzogen:

- Zunächst wird der Vektor $\vec{W}$ schlechtester Qualität der $n + 1$ Vektoren ermittelt: Startvektor $\vec{S}$ und seine Modifikationen $\vec{P}_i$.
- **Reflexion:** $\vec{W}$ wird in Richtung des Vektordurchschnitts $\vec{A}$ der n verbleibenden Vektoren ($\vec{S} - \vec{P}_i \ \forall i$) gespiegelt. Der Vektor wird mit einem vorher festgelegten Reflexionsfaktor F_r multipliziert (Abbildung 3.8 b).
- **Expansion:** Bei erfolgreicher Spiegelung wird $\vec{W}$ einer zusätzlichen Expansion unterzogen. Expandiert wird mit einem vorher festgelegten Expansionsfaktor F_e (Abbildung 3.9 a).
- **Kontraktion:** Ist die Reflexion nicht erfolgreich, wird $\vec{W}$ in Richtung des Vektordurchschnitts bewegt. Wie weit, hängt auch hier von einem vorher festgelegten Faktor, dem Kontraktionsfaktor F_c, ab.

Die Abbildung 3.8 a) zeigt eine beliebige zweidimensionale Qualitätsfunktion in Isoliniendarstellung mit einer zufälligen Startkonfiguration.

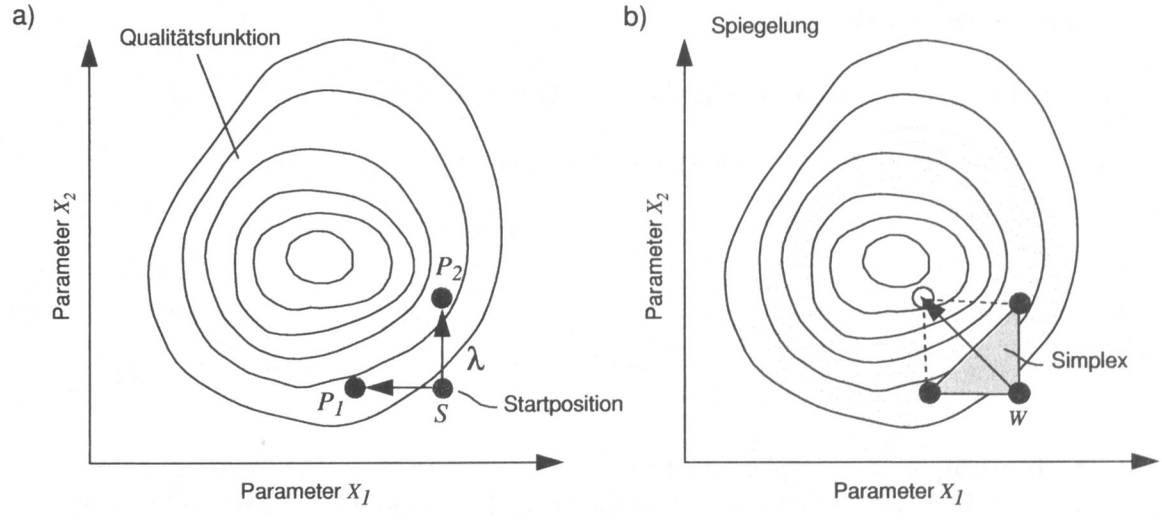


Abbildung 3.8: Zum Simplex-Algorithmus: a) Aus dem Startpunkt S werden die n Vektoren $\vec{P}_i$ erzeugt. Der skalare Faktor λ bestimmt die Größe des Simplex zu Beginn der Optimierung. Die schlechteste Realisierung $W = S$ wird gespiegelt (b).

Die Abbildung 3.9 a) zeigt am zweidimensionalen Beispiel die Operation „Expansion". Die Expansion ist nicht erfolgreich, da die erreichte Qualität im Vergleich zu der Qualität der Reflexion schlechter ist. Die expandierte Realisierung wird verworfen. Ausgehend von der

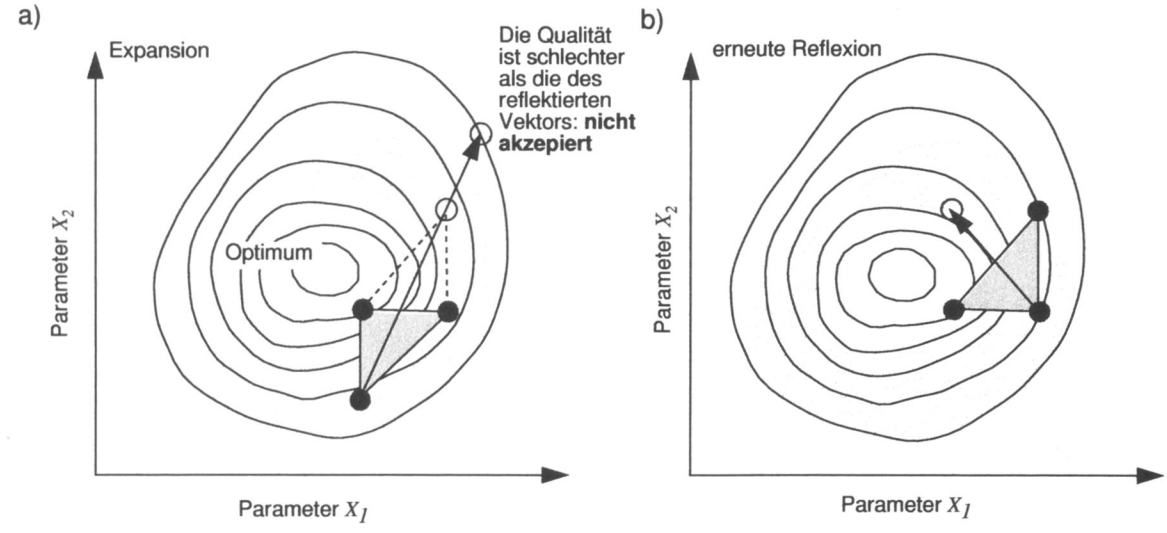


Abbildung 3.9: Weitere Operationen des Simplex-Algorithmus: a) ist eine Reflexion erfolgreich, wird eine zusätzliche Expansion getestet. Hier ist diese Expansion nicht erfolgreich, so daß der neue Simplex mit dem reflektierten Punkt gebildet wird (b).

neuen Situation wird wiederum reflektiert (Abbildung 3.9 b).

Für den Fall, daß alle beschriebenen Aktionen nicht zu einer besseren Qualität führen, sind zwei weitere Operationen möglich:

- Kontraktion des gesamten Simplex (Abbildung 3.10 d) und
- stochastische Variation des gesamten Simplex (Abbildung 3.10 e).

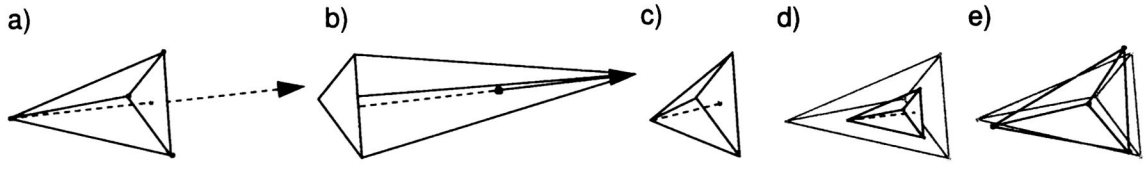


Abbildung 3.10: Simplex-Operationen bei drei freien Parametern: a) Ermittlung des Durchschnittsvektors, b) Reflexion, c) Kontraktion, d) Kontraktion des gesamten Simplex und e) Variation des gesamten Simplex.

Führen auch diese Operationen nicht zu besseren Qualitäten, bzw. es wird nach einigen Iterationen wieder die gleiche Position erreicht, ist ein möglicherweise lokales Optimum erreicht und der Simplex degeneriert, d.h. sein Volumen geht gegen Null. Dann kann der Algorithmus beendet oder von einer neuen Position aus gestartet werden. Mit den beschriebenen Operationen ist der Simplex-Algorithmus in der Lage, die räumliche Verteilung seiner Ecken der „Qualitätslandschaft" anzupassen. In einem „langgestreckten Tal" beispielsweise wird er sich selbst zur Richtung der Achse des Tals hin ausrichten. In einer Kurve wird er seine Ecken so ausrichten, daß der Simplex der dort benötigten Form angepaßt ist.

3.8 Genetische Algorithmen

Genetische Algorithmen (Holland, 1992; Goldberg, 1989; Schöneburg et al., 1994) verfolgen das gleiche Ziel wie die bisher beschriebenen Verfahren: die Optimierung einer vorgegebenen Zielfunktion. Diese Methode ist den natürlichen Mechanismen der biologischen Evolution nachempfunden. Im Gegensatz zu den bisher beschriebenen Verfahren steht bei den genetischen Algorithmen nicht die Variation der Parameter, sondern die „Rekombination" verschiedener, im Optimierungsprozeß erzeugter Parametervektoren im Vordergrund. Um ein Optimierungsproblem mit genetischen Algorithmen zu bearbeiten, werden folgende Vorbereitungen durchgeführt:

- Festlegen eines Bereiches, in dessen Grenzen sich die Parameter bewegen dürfen (Minimum und Maximum),
- Normierung dieses Bereiches auf das Intervall [0,1],
- Initialisierung einer Startpopulation mit gleichverteilten Zufallszahlen über dem Intervall [0,1],
- Transformation in „*Bitstrings*" einer vorgegebenen Länge.

Unter *Bitstrings* versteht man binär kodierte Zahlen. Die numerische Auflösung, die mit solchen Bitstrings erreicht werden kann, wird dabei durch ihre Länge bestimmt. Die Abbildung 3.11 erklärt die beschriebenen Vorbereitungen.

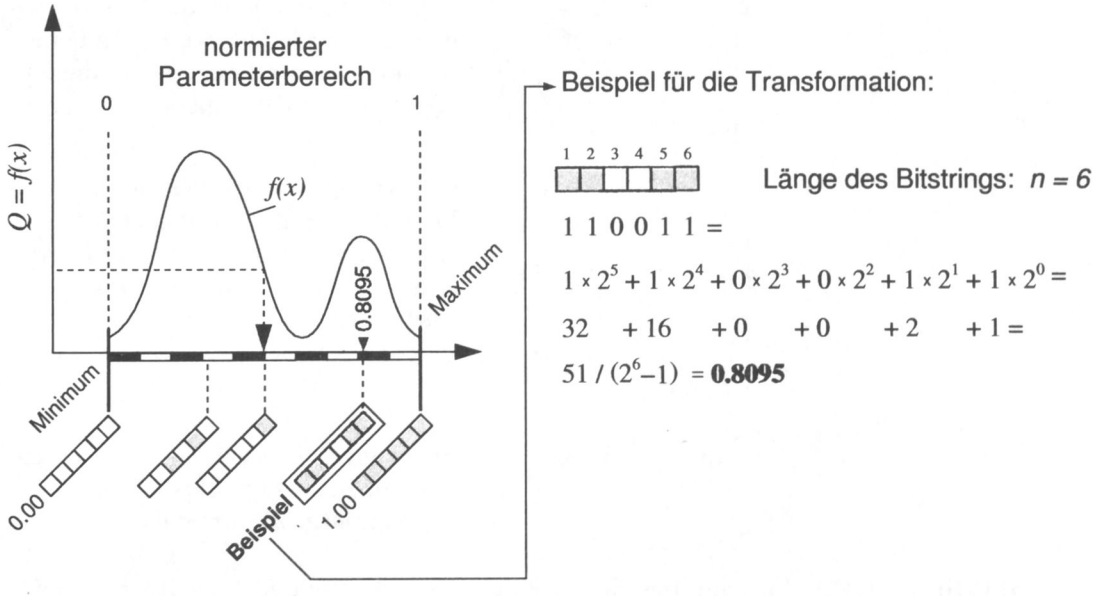


Abbildung 3.11: Vorbereitungen zum Arbeiten mit genetischen Algorithmen. Die Länge des Bitstrings bestimmt die numerische Auflösung: für n=6 ergibt sich der kleinste mögliche Abstand zwischen den einzelnen Strings: $1/2^6 - 1 = 0.01587$.

Nach diesen Vorbereitungen wird ein Algorithmus ausgeführt, der durch die mit der Qualität gewichteten Auswahl von Modellindividuen einen „Selektionsdruck" (Hoff und Miram,

1987; Hofbauer und Sigmund, 1984) erzeugt. Wegen der biologischen Analogie des Algorithmus wird im folgenden eine Parametereinstellung als Modellindividuum oder Individuum bezeichnet. Individuen hoher Qualität werden dabei mit größerer Wahrscheinlichkeit selektiert. Dadurch unterscheiden sich genetische Algorithmen von den bisher beschriebenen Verfahren stark. Die eigentliche „Evolution", die Optimierung, wird durch neue Kombinationen der Parameter vorhandener Modellindividuen erzeugt. Mutationen, also Variationen der Parameter, die bei den bisher beschriebenen Algorithmen eine entscheidende Rolle spielten, sind hier von untergeordneter Bedeutung. Eine sehr kleine Mutationswahrscheinlichkeit entscheidet, ob ein Parameter der transformierten Bitstrings verändert wird oder nicht. In der hier beschriebenen Form wird dann ein Bit des Bitstrings negiert. Die Abbildung 3.12 zeigt den eigentlichen Optimierungsablauf.

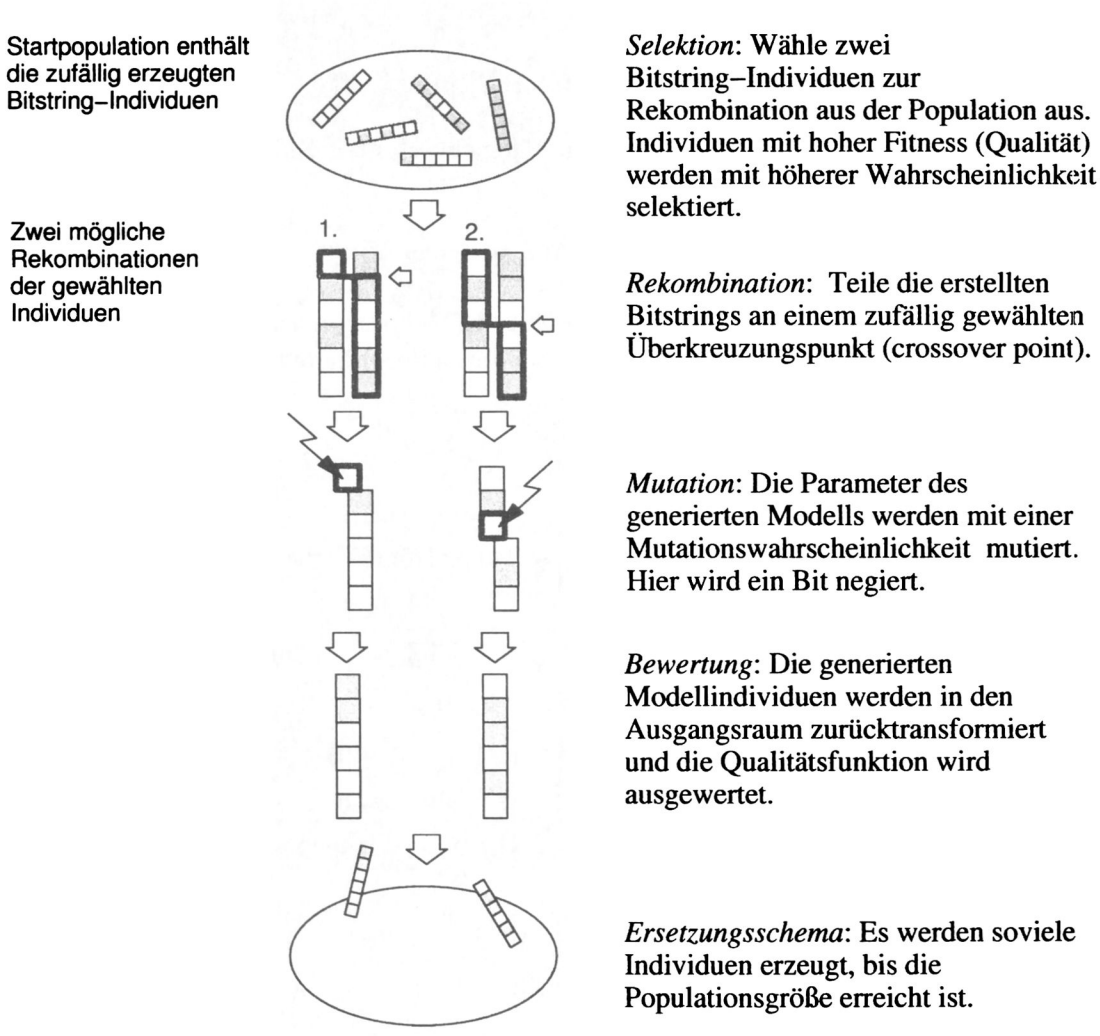


Abbildung 3.12: Anschauung des Ablaufs bei genetischen Algorithmen. Die gesamte Population wird gegen eine neue ausgetauscht. Der Antrieb der Optimierung ist die Rekombination der Individuen mit guten Qualitätswerten. Mutationen sorgen im wesentlichen für Variabilität in der Population.

Sind mehrere Parameter zu optimieren (allgemeiner Fall), werden die einzelnen Bitstrings hintereinander geschrieben und als ein Individuum betrachtet. Die Verwendung des binären Zahlensystems ist für das Arbeiten des Algorithmus nicht zwingend. Auch die

verwendete Transformationsvorschrift vom dezimalen in das binäre Zahlensystem kann mit verschiedenen Codierungen erfolgen. Der sog. *Cray-Code* wird hier von einigen Autoren vorgeschlagen (siehe z.B. Goldberg (1989)).

Im Algorithmus in Abbildung 3.12 werden solange Individuen rekombiniert, bis die neue Population die gewünschte Größe hat. Andere, einfachere Ersetzungsschemata benutzen z.B. nur die zwei besten Individuen, rekombinieren sie und führen sie der Ausgangspopulation zu. Auch Konzepte mit „Kindergarten" oder „Altersheim" wurden vorgeschlagen (Schöneburg et al., 1994; Goldberg, 1989). Man versucht mit diesen Ansätzen, Individuen guter Qualität, die eine Parameterkonstellation repräsentieren, die für den Optimierungsprozeß zu früh (z.B. durch Mutation) entstanden sind, aufzubewahren, um sie zu einem späteren Zeitpunkt wieder in den Evolutionsprozeß einfließen zu lassen. Dies ist sinnvoll, da es Situationen gibt, in denen es für gute Individuen keine „Partner" gibt, mit denen diese rekombiniert werden können, so daß Individuen entstehen, die diesen ähnlich sind, also gute Qualität haben. Auch gibt es die Möglichkeit mehrerer *crossover points*. Die Vielfalt an Varianten der genetischen Algorithmen ist so groß, daß hier nur die wichtigsten Züge der Verfahren beschrieben werden können.

Auch bei genetische Algorithmen kann es zu Rückschritten in bezug auf die Qualitätsfunktion kommen, da die gesamte Population durch die Rekombinationen der Individuen der Elternpopulation ersetzt wird, **ohne** daß die Qualität eine direkte Rolle spielt. Der einzige „Antrieb" ist die Selektion, die gewichtet mit der Qualität geschieht.

3.9 Evolutionsstrategien (ES)

Die Algorithmen der Evolutionsstrategien, welche etwa zur gleichen Zeit wie die genetischen Algorithmen entwickelt wurden (Rechenberg, 1973), basieren ebenfalls auf der Idee der Adaptierung der Prinzipien der natürlichen Evolution: Rekombination, Mutation und Selektion. Der wesentliche Unterschied zu den genetischen Algorithmen ist, daß die Evolutionsstrategien keine Transformation der zu optimierenden Parameter in eine binäre (oder andere) Repräsentation benötigen. Während genetische Algorithmen Begrenzungen, in denen sich die Parameter bewegen können benötigen, ist dies bei Evolutionsstrategien nicht zwingend erforderlich. Für Genetische Algorithmen gibt es bis heute wenig fundierte theoretische Grundlagen. Das sog. Schema Theorem erlaubt keine Aussagen über die Konvergenzeigenschaften der Algorithmen. Rechenberg konnte hingegen für das Konvergenzverhalten der Evolutionsstrategien für bestimmte synthetische Testfunktionen eine mathematische Grundlage ableiten (Rechenberg, 1994), die Aussagen über das Konvergenzverhalten zuläßt. Auch Schwefel (1995, 1977) erarbeitete mit wesentlichen theoretischen Grundlagen, die die Basis der heutigen Evolutionsstrategien.

Die einzige Voraussetzung zum Arbeiten mit Evolutionsstrategien, ein „stark kausaler Zusammenhang" zwischen den Modellparametern und der Qualitätsfunktion, ist für die meisten Optimierungsprobleme gegeben. Stark kausal bedeutet, daß kleine Änderungen an den Parametern zu verhältnismäßig kleinen Änderungen der Qualität führen. Es werden keine Ableitungen der Qualitätsfunktion benötigt und die Funktion muß nicht stetig bzw. stetig differenzierbar sein. Für nichtkausale Situationen können dem Problem angepaßte Mutationsoperatoren kausale Bedingungen schaffen (Herdy, 1996). Die Anwendbarkeit der Evolutionsstrategien zur Optimierung geophysikalischer Modelle wurde von Barleben (1989),

Blümecke (1991), Dirks (1995) und Alvers (1990); Alvers et al. (1995) gezeigt. Im folgenden sollen die Algorithmen der Evolutionsstrategien beschrieben werden.

3.9.1 Die $(\mu, +\lambda)$-ES mit adaptiver Schrittweitenregelung

Die Schreibweise „$(\mu, +\lambda)$-ES" wurde von Schwefel (1977) eingeführt. Die Strategieparameter μ und λ bedeuten Anzahl der Eltern(modelle) bzw. Anzahl der Nachkommen(modelle). Der Algorithmus der $(\mu + \lambda)$-ES kann wie folgt beschrieben werden:

Schritt 1 Erzeuge μ verschiedene Startmodelle: Elternpopulation,

Schritt 2 erzeuge λ Kopien durch zufällige Wahl der μ Startmodelle (Nachkommenpopulation),

Schritt 3 mutiere die Parameter der Individuen der Nachkommenpopulation durch Addition normalverteilter Zufallszahlen,

Schritt 4 bewerte die so erzeugten Individuen durch die Auswertung der Qualitätsfunktion und führe sie einer Selektion zu,

Schritt 5 selektiere die μ besten Individuen aus Eltern- und Nachkommenpopulation als Ausgang für die nächste Generation.

Die Abbildung 3.13 zeigt das Flußdiagramm einer $(3 + 6)$-ES mit der von Rechenberg eingeführten Symbolik.

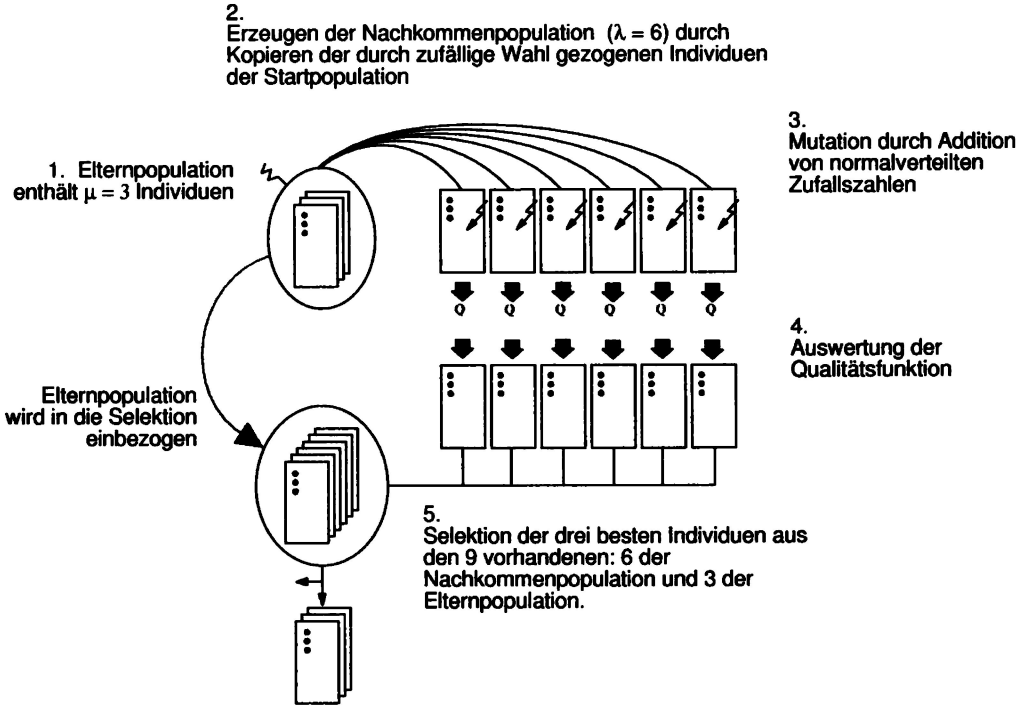


Abbildung 3.13: Beispiel einer $(3 + 6)$−Evolutionsstrategie. Mit drei Elternmodellen werden durch zufällige Wahl sechs Nachkommenmodelle erzeugt. Diese werden mutiert und einer Selektion zugeführt. Die drei besten werden Ausgang für die nächste Generation.

Entscheidend ist, daß die Elternpopulation mit in die Selektion einbezogen wird. Dies wird in Abbildung 3.13 durch den Pfeil zwischen Eltern- und Nachkommenpopulation zum Ausdruck gebracht. In der Schwefelschen Notation symbolisiert das „+"-Zeichen die Einbeziehung der Elternpopulation in die Selektion. Werden die Individuen der Elternpopulation nicht in die Selektion einbezogen und nur die Individuen der Nachkommenpopulation berücksichtigt, sind Rückschritte in der Qualitätslandschaft prinzipiell möglich. Es werden dann lediglich die μ besten Nachkommen selektiert ohne zu berücksichtigen, ob sie besser oder schlechter sind als die Individuen der Startpopulation. Die $(\mu + \lambda)$-ES kann als paralleles Hill-Climbing bezeichnet werden.

Eine konzeptionelle Erweiterung stellt erst die (μ, λ)-ES dar, da das Verlassen von lokalen Optima prinzipiell möglich wird. Die Elternpopulation wird dann nicht in den Selektionsprozeß einbezogen (siehe auch die Abschnitte 3.3, 3.4, 3.5 und 3.6). Für $\mu = \lambda = 1 \rightarrow (1 + 1)$-ES geht die Evolutionsstrategie in Hill-Climbing über - der mutierte Nachkomme wird nur akzeptiert, wenn er eine bessere Qualität hat als der Elternvektor. Die $(1, 1)$-ES beschreibt einen Zufallsweg (random walk), da jede Variation akzeptiert wird, egal ob sie eine schlechtere oder bessere Qualität hat.

Die Mutation der Parameter erfolgt durch die Addition normalverteilter Zufallszahlen. Dies ist plausibel, da Variationen in der Biologie häufig kleine Unterschiede der Ausprägung verschiedener Merkmale einer Art sind, die Normalverteilungen genügen (Sedlag und Weinert, 1987). Aufgrund der nicht vorhandenen Codierung des zu optimierenden Objektes in Evolutionsstrategien werden im Unterschied zur Biologie Mutationen am Phänotyp - dem Modell - und nicht am Genotyp durchgeführt. In der Natur ist die genetische Information durch vier Grundbausteine der DNS quaternal codiert. In genetischen Algorithmen ist die Codierung des zu optimierenden Objektes oft binär. Eine geometrische Interpretation kann gegeben werden: Im $\mathbb{R}^n$ liegen die λ Nachkommen auf einer Hyperkugelschale gleichverteilt um den Elternvektor.

Die Einführung eines Konzepts zur Steuerung der Variationsbreite (Schrittweite) ermöglicht eine Adaption des Algorithmus an lokale Gegebenheiten der Qualitätsfunktion. Die Idee ist, jedem Individuum eine eigene „Schrittweite" zuzuordnen, mit der die Mutationsverteilung skaliert wird. Die Nachkommenindividuen liegen dann auf einer Hyperkugel, deren Radius mit dem jeweiligen Faktor skaliert wurde. In der Selektion werden Individuen guter Qualität ausgewählt und der Durchschnitt der Schrittweiten dieser Individuen als Schrittweite für die nächste Generation zugrunde gelegt. Dadurch ist es möglich, in der Nähe eines Optimums die Variationen adaptiv derart zu verkleinern, daß ein Optimum mit großer Genauigkeit erreicht werden kann. Allerdings ist diese Verkleinerung der Schrittweite unabhängig von der Art des Optimums (lokal oder global) und birgt die Gefahr vorzeitiger Konvergenz. Auch bei der Behandlung von Randbedingungen ist dieses Konzept hinderlich, da Schrittweiten an Rändern schnell infinitesimal klein werden und der Optimierungsprozeß stagniert (Hansen, pers. Mitt.). Auch eine Dämpfung der Adaption der Schrittweiten behebt dieses Problem nicht prinzipiell (Hansen et al., 1995).

3.9.2 Die (μ, λ)-ES mit Kovarianzmatrixadaption

Die jüngsten Entwicklungen von Hansen und Ostermeier (1997) erweitern Evolutionsstrategien derart, daß die Adaption des Erzeugendensystems, d.h. eine adaptive Drehung und Skalierung der Mutationsverteilung, möglich wird. Diese Algorithmen werden als ES-CMA Algorithmen bezeichnet (*covariance matrix adaption*). Sie ermöglichen die Optimierung von skalierten Qualitätsfunktionen. Der Algorithmus ist sehr aufwendig zu implementieren, daher sollen hier lediglich die Prinzipien erläutert werden. Im folgenden werden Erklärungen anhand der $(1, \lambda)$-ES gegeben. Der Algorithmus kann problemlos auf die (μ, λ)-ES erweitert werden. Ein Nachkommenvektor $\vec{x}_N^k$ der $k = 1, \ldots, \lambda$ Nachkommen wird wie folgt erzeugt:

$$\vec{x}_N^k = \vec{x}_E + \delta_E \, \mathbf{B}_E \, \vec{z}_k$$

wobei:

n	=	Anzahl der Parameter (Dimension) des Vektors,
$\vec{x}$	=	$(x_1, \ldots, x_n) \in I\!R^n$ Objektvariablenvektor, der optimiert werden soll,
E		Index für Elternvektor,
N		Index für Nachkommenvektor,
k		$(1, \ldots, \lambda)$ Index für Nachkommen,
δ	>	0 globale Schrittweite,
$\vec{z}$	=	$(x_1, \ldots, x_n) \in I\!R^n$ Vektor normalverteilter Zufallszahlen und
$\mathbf{B}$	$\in$	$I\!R^{n \times n}$ Transformationsmatrix, die $\vec{z}$ linaer transformiert.

Das Hauptproblem besteht bei diesem Algorithmus in der Adaption der Matrix $\mathbf{B}$ entsprechend des Differenzvektors zwischen Elternvektor und selektiertem Nachkommenvektor. Die Idee, die hierbei zugrunde gelegt wird, ist die Auswertung erfolgreicher Schritte im Parameterraum. Es wird jedoch nicht nur der Betrag des Mutationsvektors, also die globale Schrittweite, betrachtet, sondern der Vektor insgesamt. Es werden damit auch Informationen über die Richtung der Bewegung im Parameterraum in Betracht gezogen. Die Informationen erfolgreicher Mutationen werden summiert und zur Adaption der Transformationsmatrix $\mathbf{B}$ benutzt. Die Transformation der Mutationsverteilung $\vec{z}$ durch $\mathbf{B}$ erfolgt unabhängig von der Adaption der globalen Schrittweite δ. Der Grund dafür ist, daß die globale Schrittweite in einer wesentlich kürzeren Zeit als $\vec{z}$ adaptiert werden kann. Zur Adaption von $\vec{z}$ werden bei n Parametern $n(n+1)/2$ Iterationen benötigt, während δ in n Iterationen adaptiert werden kann. Der Mechanismus, nachdem die Strategieparameter δ und $\mathbf{B}$ adaptiert werden, ist sehr komplex und soll hier nicht beschrieben werden. Eine ausführliche Beschreibung einschließlich Hinweise zur Implementierung und Tests der Algorithmen an synthetischen Funktionen findet man im oben zitierten Artikel.

3.9.3 Die $(\mu/\rho, \lambda)$-ES

In den bisher beschriebenen Evolutionsstrategien wurden jeweils die μ besten Individuen als Ausgangspunkt für die nächste Generation herangezogen. Das Prinzip der Rekombination,

welches bei den genetischen Algorithmen eine wesentliche Rolle spielt, wurde bislang nicht simuliert.

Bei der $(\mu/\rho, \lambda)$-ES wird die Rekombination der ρ besten Individuen eingeführt. Der Strategieparameter ρ bestimmt dabei, wieviele Individuen in die Rekombination einbezogen werden. Rechenberg unterscheidet zwei verschiedene Arten der Rekombination: die intermediäre und die dominante (Rechenberg, 1973). Bei intermediärer Rekombination wird der Durchschnitt der Parameter der rekombinierten Individuen zum Bau des neuen benutzt. Da jedes Individuum einen Vektor im $\mathbb{R}^n$ darstellt, kann das so erzeugte Individuum mathematisch betrachtet als Vektordurchschnitt der rekombinierten Vektoren interpretiert werden. Bei dominanter Rekombination wird per Zufall entschieden, von welchem Individuum der jeweils betrachtete Parameter gewählt wird. Im Gegensatz zu den genetischen Algorithmen haben hier alle an der Rekombination beteiligten Individuen die gleiche Wahrscheinlichkeit, selektiert zu werden. Die Abbildung 3.14 zeigt den Ablaufplan einer (3/3, 6)-ES.

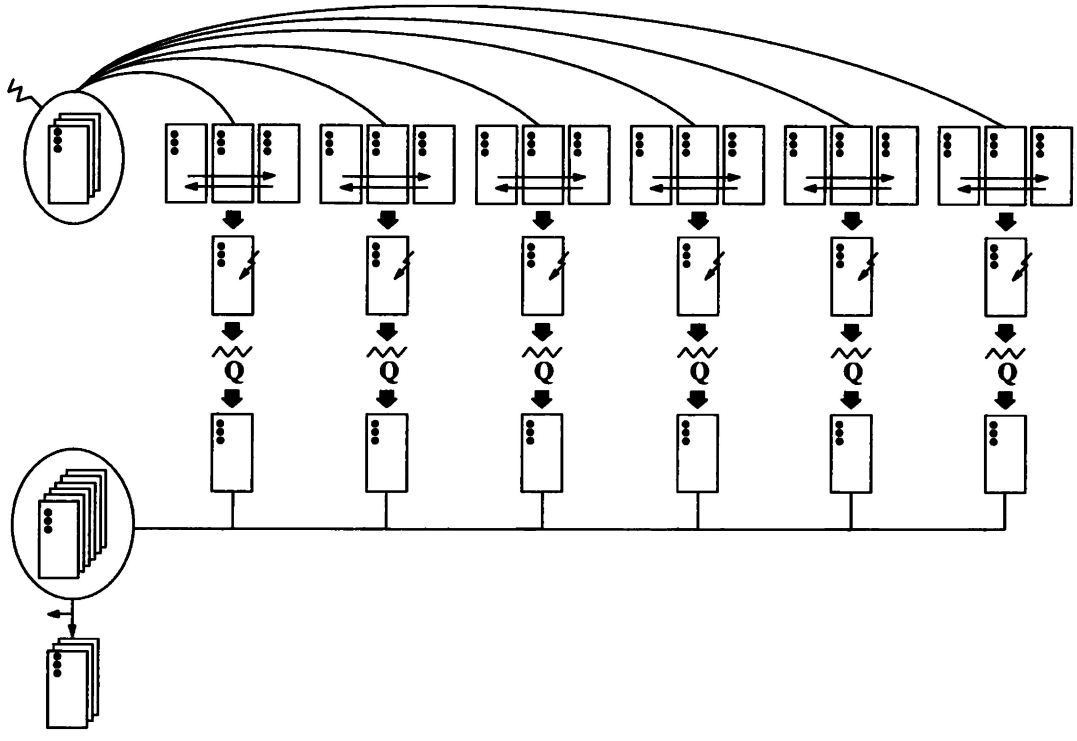


Abbildung 3.14: Beispiel einer (3/3, 6)−Evolutionsstrategie. Drei Elternmodelle erzeugen durch zufällige Wahl und Rekombination sechs Nachkommenmodelle. Diese werden mutiert und der Selektion zugeführt. Die drei besten werden Ausgang für die nächste Generation.

3.10 Wahrscheinlichkeit im $\mathbb{R}^n$

Parameteroptimierungen finden fast immer in hochdimensionalen Parameterräumen statt. Sobald mehr als zwei freie Parameter die Qualitätsfunktion bestimmen, ist diese nicht mehr direkt visualisierbar. Die Effekte, die bei der Erhöhung der Dimension des Parameterraumes

auftreten, können daher nicht problemlos zugänglich gemacht werden. Für das Verständnis der beschriebenen Algorithmen werden daher einige grundlegende Phänomene erläutert.

Im folgenden werden zwei Experimente durchgeführt, die Einblick in das Verhalten von Optimierungsstrategien bzw. stochastischen Suchprozessen geben.

3.10.1 Ein Monte-Carlo-Experiment

Es soll gezeigt werden, wie sich die a priori Wahrscheinlichkeit, einen beliebigen Punkt des Raumes durch stochastische Suche zu finden, bei Erhöhung der Dimension des Raumes verändert. Als synthetische Qualitätsfunktion wird die Gleichung

$$y(x_i) = \frac{1}{1+\sum_{i=1}^{n} x_i^2} \text{ mit } x_i > 0 \tag{3.1}$$

benutzt. Die Abbildung 3.15 zeigt die Funktion für $n = 1$, also mit einem freien Parameter. Das Maximum $y(x_i) = 1$ nimmt die Funktion für $x_i = (0, 0, ..., 0)$ ein.

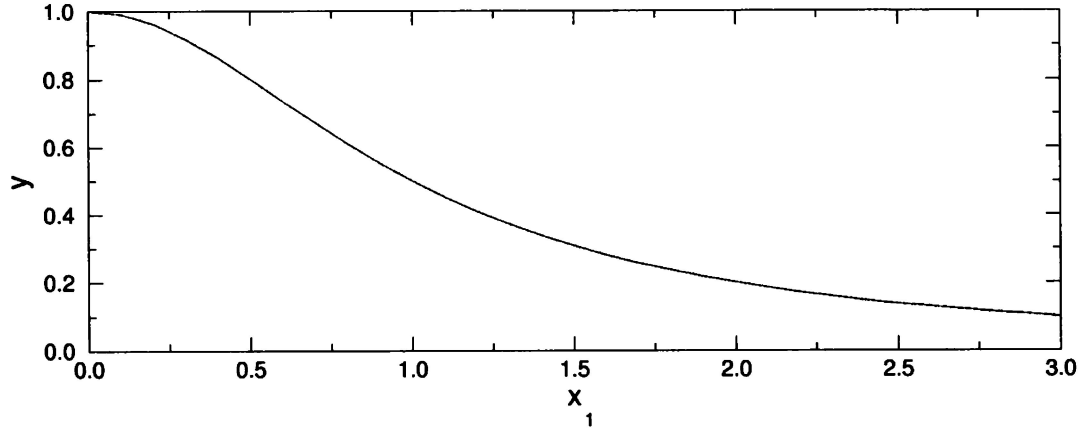


Abbildung 3.15: Der Graph der Gleichung 3.1 für $n = 1$ im Intervall [0,3].

In einer Monte-Carlo-Simulation soll eine Zufallssuche nach dem Optimum durchgeführt werden. Für verschiedene Dimensionen wird folgender Algorithmus ausgeführt:

- Die Parameter x_i der Gleichung 3.1 werden mit gleichverteilten Zufallszahlen aus dem Intervall $[0, 1)$ belegt.
- Die Gleichung wird ausgewertet und die Werte $y(x_i)$ werden entsprechend ihres Funktionswertes farblich codiert und dargestellt.
- Diese Prozedur wird 100,000 Mal wiederholt, so daß man von statistisch relevanten Ergebnissen ausgehen kann.

Für Dimensionen größer als zwei können die Werte $y(x_i)$ nicht direkt dargestellt werden. Es wird deshalb eine Projektion des $\mathbb{R}^n$ in den $\mathbb{R}^2$ eingeführt:

$$A = \sqrt{\frac{2}{n}\sum_{i=1}^{n} x_{i_{gerade}}^2} \quad \text{und} \quad B = \sqrt{\frac{2}{n}\sum_{i=1}^{n} x_{i_{ungerade}}^2} \tag{3.2}$$

Die Projektion hat die Eigenschaft, den Raum derart abzubilden, daß keine Verzerrungen auftreten. Es ist beispielsweise leicht ersichtlich, daß die Vektoren $(1,1,1,...,1)$ und $(1,1)$ oder $(2,0,2,0,...,2,0)$ und $(2,0)$ an die gleiche Position im $\mathbb{R}^2$ projiziert werden. Der oben beschriebene Algorithmus wurde für einzelne Dimensionen zwischen zwei und 500 durchgeführt. Die Abbildung 3.16 zeigt die Resultate der Simulationen.

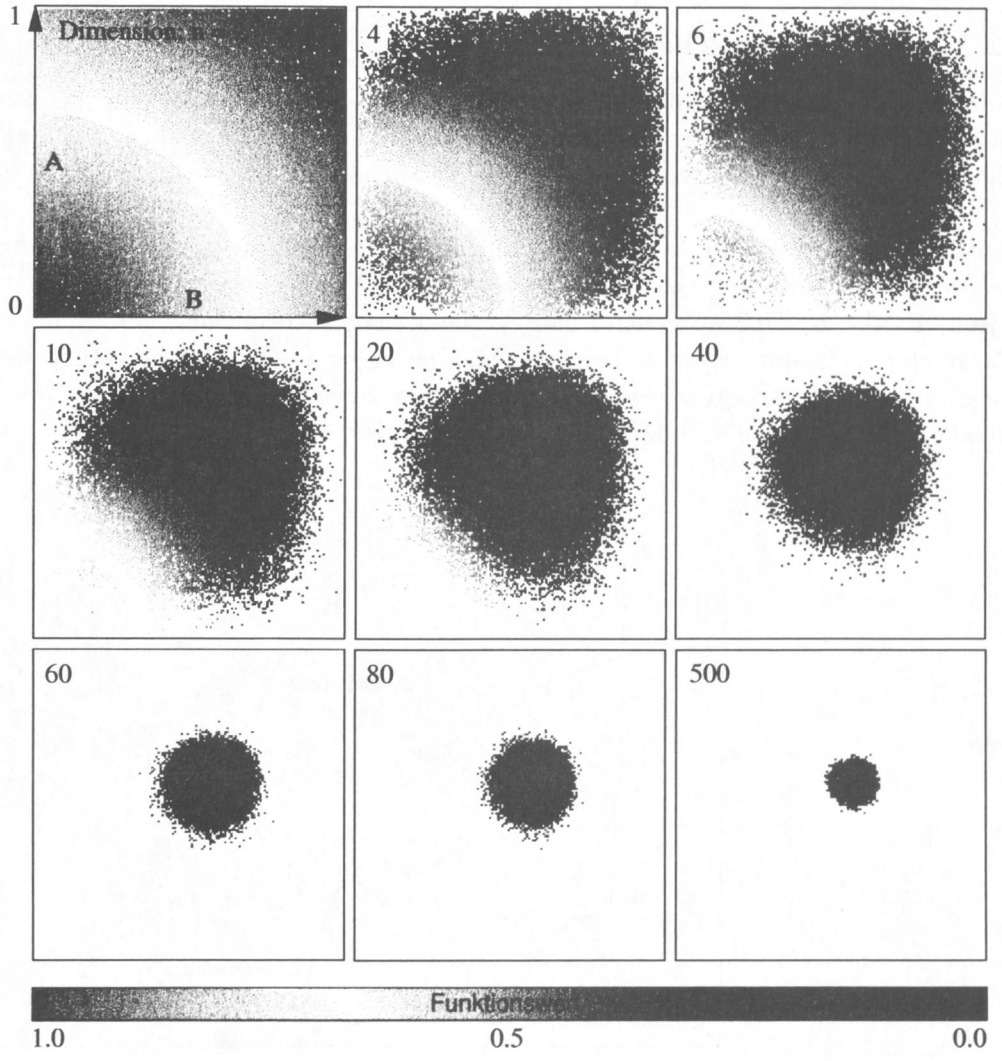


Abbildung 3.16: Ergebnisse einer Monte-Carlo-Simulation für eine synthetische Qualitätsfunktion. Bei Erhöhung der Dimension des Parameterraumes nimmt bei einer Zufallssuche die durchschnittlich erreichte Qualität stark ab. Die Grauskala codiert den Funktionswert der Gleichung 3.1 (siehe Text).

Bewertung der Ergebnisse

Die Abbildung 3.16 zeigt deutlich, daß sich die **gleichverteilten** Zufallsrealisierungen über dem Intervall [0,1) bei Erhöhung der Dimension vom Maximum der Gleichung, welches sich jeweils in der linken unteren Ecke der Teilbilder befindet, entfernen. Auch

die Farbcodierung läßt erkennen, daß die Funktionswerte immer kleiner werden. Bei 300 Parametern ist der durchschnittlich erreichte Funktionswert $< 10^{-7}$.

Dieses Experiment zeigt, welchen gravierenden Einfluß allein die Anzahl der freien Parameter auf ein Optimierungsproblem hat. Aufgaben, die mit einer rein stochastischen Suche für wenige Parameter durchaus bearbeitet werden können, bedürfen bei höheren Dimensionen besser angepaßter Verfahren.

3.10.2 A priori Wahrscheinlichkeit

Folgende Frage soll untersucht werden: Wie verhält sich die Wahrscheinlichkeit, eine Verbesserung der Qualität in Abhängigkeit von der Dimension zu erreichen, wenn an einem beliebigen Punkt des Parameterraumes eine Zufallssuche initiiert wird?

Als synthetische Qualitätsfunktion dient wiederum die Gleichung 3.1. Betrachtet man beispielsweise die Funktion für $n = 1$, so ist die Wahrscheinlichkeit, vom (beliebig gewählten) Punkt $x = 0.5$ ausgehend eine Verbesserung zu erreichen, gerade $0.5 = 50\%$. Der Parameter kann entweder in Richtung Maximum ($x \to 0$) oder in entgegengesetzte Richtung ($x \to +\infty$) bewegt werden. Schon für zwei freie Parameter ist die Wahrscheinlichkeit kleiner als 0.5. Die Abbildung 3.17 erläutert dies.

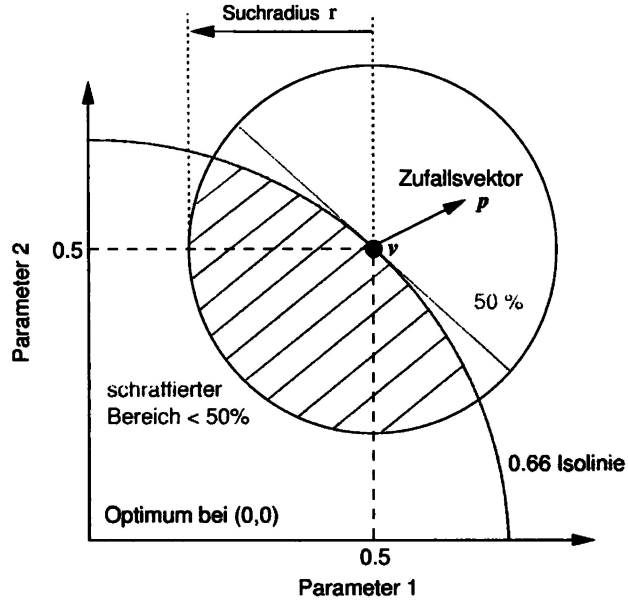


Abbildung 3.17: Im $\mathbb{R}^2$ ist die Wahrscheinlichkeit w, daß ein Zufallsvektor einer bestimmten maximalen Länge (Suchradius) in eine Richtung zeigt, in der sich bessere Funktionswerte befinden, als am Punkt selbst, kleiner 50% (schraffierter Bereich). Gilt für den Suchradius $r \to 0$, folgt für die Wahrscheinlichkeit $w \to 0.5$.

Es wurde folgende Simulation ausgeführt:

- Addiere zum (beliebig gewählten) Punkt $v = (0.5, 0.5, ..., 0.5)$ einen Vektor gleichverteilter Zufallszahlen einer festgelegten Länge. Nenne den konstruierten Punkt p.

- Berechne die Gleichung 3.1 und vergleiche den Funktionswert von p mit dem von v.
- Falls p einen besseren Funktionswert als v hat, wird dieser schwarz, andernfalls rot codiert. Mit Hilfe der Gleichung 3.2 wird p in den zweidimensionalen Raum projiziert und dargestellt.

Die Abbildung 3.18 zeigt die Ergebnisse der Simulationen.

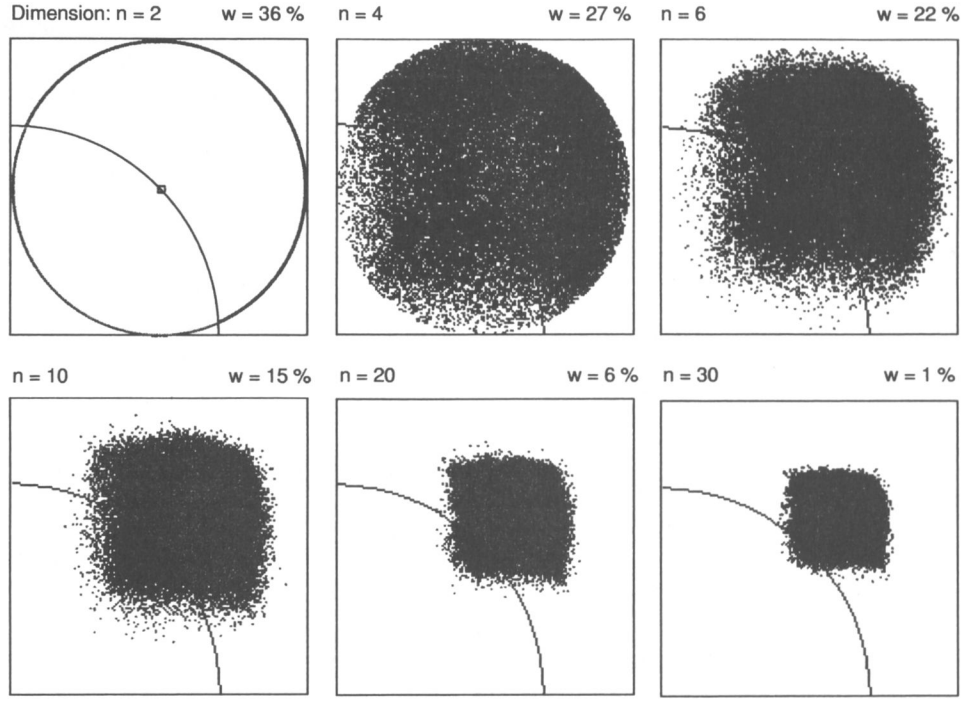


Abbildung 3.18: Ergebnisse einer Monte-Carlo-Simulation für eine synthetische Qualitätsfunktion (siehe Text).

Bewertung der Ergebnisse

Die Simulation zeigt, daß die Wahrscheinlichkeit, daß ein Vektor gleichverteilter Zufallszahlen in die Richtung eines Gebietes der Qualitätsfunktion zeigt, in der sich bessere Funktionswerte als an der aktuellen Position im Parameterraum befinden, mit der Erhöhung der Dimension des Parameterraumes drastisch abnimmt. Bei der Bewertung dieses Ergebnisses muß außerdem beachtet werden, daß es sich bei Gleichung 3.1 um eine symmetrische Funktion handelt. Betrachtet man beispielsweise einen Parabelgrat oder andere komplexere Testfunktionen (siehe z.B. (Whitley et al., 1995)), verringert sich die Wahrscheinlichkeit zusätzlich. Bei praktischen Optimierungsproblemen kann i. allg. von einer Skalierung der einzelnen Koordinatenachsen zueinander ausgegangen werden, so daß die Gleichung 3.1 eher als „einfacher Sonderfall" bezeichnet werden muß.

Die beiden Simulationen haben verdeutlicht, daß allein die Anzahl der freien Parameter eines zu optimierenden Systems die Wahrscheinlichkeit, einen Punkt im Parameterraum durch Zufallssuche zu finden, erheblich beeinflußt.

3.11 Die Randbedingungen

Für die gravimetrische Optimierung sind Randbedingungen für Modellparameter unerläßlich. Dies hat zwei Gründe: Zum einen kann durch Parametergrenzen die kombinatorische Vielfalt des Modells eingeschränkt werden, zum anderen werden physikalisch sinnvolle Bereiche „abgesteckt". Außerdem wird das Äquivalenzprinzip, welches den Potentialverfahren zugrunde liegt, durch sinnvolle Randwerte eingeengt.

Weil die Freiheit für die Bewegung im Parameterraum dadurch eingeschränkt ist, muß bei der Vergabe solcher Bedingungen beachtet werden, daß das Erreichen eines Optimums unter bestimmten Umständen ganz verhindert werden kann. Die Abbildung 3.19 zeigt eine Situation, in der das Optimum prinzipiell nicht erreicht werden kann, da die Randbedingungen für die beiden Parameter zu „eng" gesetzt wurden.

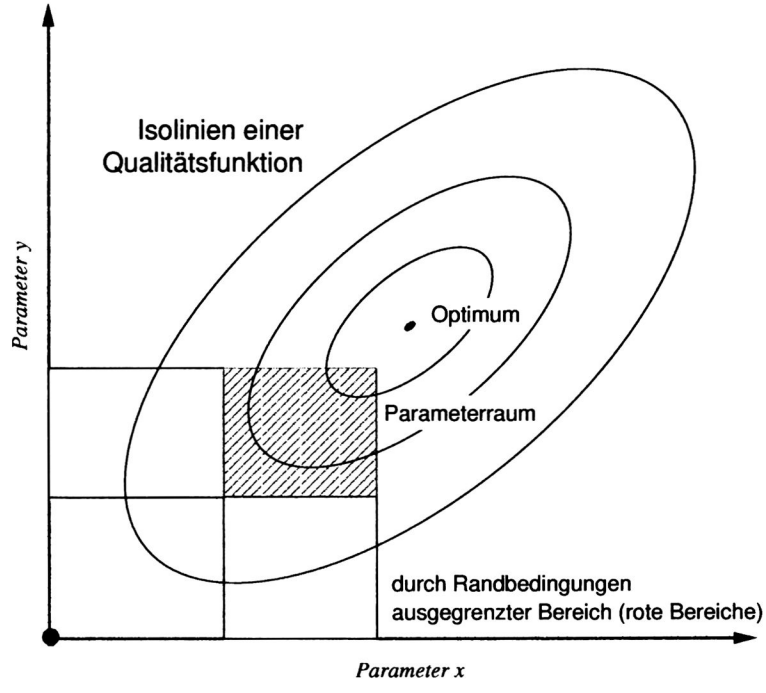


Abbildung 3.19: Randbedingungen können das Erreichen eines Optimums verhindern. Gezeigt sind hier die Isolinien einer fiktiven Qualitätsfunktion. Das Optimum befindet sich außerhalb der zulässigen Bereiche der Parameter x und y.

In einem Optimierungsprozeß würden sich in diesem Beispiel die Parameter x und y in die rechte obere Ecke des grün schraffierten Bereichs bewegen. Die Optimierung würde dort stagnieren, da sich die Parameter nicht weiter nach rechts bzw. nach oben bewegen könnten. Tritt während einer Modelloptimierung der Fall auf, daß sich die Mehrzahl der Parameter an vorgegebenen Randwerten befindet und keine Verbesserung der Qualität zu erkennen ist, liegt sehr wahrscheinlich die hier konstruierte Situation vor. Dann sollte der Optimierungsprozeß unterbrochen werden und die gewählten Randbedingungen und/oder die Topologie des gesamten Modells überprüft werden.

4

Vergleich der Optimierungsverfahren

Im folgenden werden die vorgestellten Algorithmen getestet. Zunächst sollen einige Parameter erläutert werden, die bei den durchgeführten Tests verwendet werden und für das allgemeine Verständnis der Simulationen wichtig sind.

Verwendete Pseudo-Zufallszahlen

Alle Algorithmen, außer dem Simplex-Verfahren, benötigen für stochastische Variationen der Modellparameter Zufallszahlen. Da sich mit Hilfe eines Computers Sequenzen echter Zufallszahlen nicht erzeugen lassen, wird hier auf einen Algorithmus zurückgegriffen, der sog. Pseudozufallszahlen erzeugt (siehe z.B. Kennedy und Gentle (1989)). Die Programmbibliothek des verwendeten C++ Compilers bietet hier eine Implementierung des Verfahrens von Rotenberg (1960), die gut getestet ist und statistischen Tests genügende Ergebnisse liefert. Auf die Implementierung komplizierterer Verfahren (Tausworth, 1965) kann daher verzichtet werden.

Wiederholungen

Um statistisch relevante Aussagen treffen zu können, sollten Optimierungen, die auf pseudostochastischen Prozessen basieren, möglichst oft wiederholt werden. Zur Beurteilung der Ergebnisse wird der Durchschnitt und die Varianz betrachtet. Deshalb werden in allen folgenden Simulationen mindestens 90 Wiederholungen durchgeführt. Aus statistischer Sicht wären mehr Wiederholungen wünschenswert, der Rechenaufwand dafür ist allerdings leider immer noch zu groß.

Abbruchkriterium

Das Kriterium, nachdem Optimierungen abgebrochen bzw. mit neuen Startbedingungen wiederholt werden, wird nach vorangegangenen Tests bei den meisten folgenden Simulationen auf 30,000 Funktionsaufrufe festgelegt. Falls ein anderes Abbruchkriterium verwendet wurde, wird dies explizit erwähnt. Dies hat zum einen den Grund in beschränkter Rechnerkapazität, zum anderen, daß nach 30000 Funktionsauswertungen i. allg. sehr wenige Strukturänderungen beobachtet werden.

4.1 Vergleich des Simplex-Verfahrens mit der Evolutionsstrategie

Nach früheren Untersuchungen (Alvers et al., 1995) erscheinen die Evolutionsstrategie und der Downhill-Simplex-Algorithmus für Problemstellungen aus der Gravimetrie und Magnetik als besonders geeignet. Daher soll ein Vergleich beider Verfahren anhand einer synthetischen Qualitätsfunktion Einblick in das Konvergenzverhalten in Abhängigkeit von der Dimension und der Skalierung des jeweiligen Problems geben. Eine synthetische Funktion wurde gewählt, um Simulationen oft genug wiederholen zu können, so daß statistisch relevante Ergebnisse erreicht werden. Bei geophysikalischen Modellen, deren Berechnung i. allg. wesentlich länger dauert, wäre ein solches Vorgehen aufgrund der zu erwartenden Rechenzeiten nicht möglich.

Im folgenden werden die Algorithmen (1,10)-ES mit Kovarianzmatrixadaption nach Hansen und Ostermeier (1997) sowie Simplex nach Nelder und Mead (1965) benutzt. Als Qualitätsfunktion dient die Gleichung des Hyperellipsoides:

$$Q = f(x_i) = \sum_{i=1}^{dim} a_i(\vec{x}_i^* - \vec{x}_i)^2 \rightarrow MIN \tag{4.1}$$

mit :

$\vec{x}_i^*$ = vorgegebener Zielvektor, hier der Nullvektor,

$\vec{x}_i$ = der zu optimierende Vektor,

a_i = Skalierungsfaktor für die Achsen des Hyperellipsoides und

i = laufender Index von 1 bis dim = Anzahl der Parameter.

Die Funktion ist für Testzwecke geeignet, da das Optimum bekannt ($x_i^* = \vec{0}$) und die numerische Berechnung nicht aufwendig ist. Gesucht wird also ein Minimum. Mit dem Faktor a_i können die Achsenverhältnisse des Hyperellipsoides verändert und somit die Abhängigkeit der getesteten Verfahren von der Skalierung eines Optimierungsproblems untersucht werden. Anhand von zwei verschiedenen Skalierungen wird im folgenden die Abhängigkeit von der Dimension untersucht:

(1) $a_i = 1 \; \forall \; i$ - symmetrisches Problem: Hyperkugel; alle Parameter haben den gleichen Einfluß auf Q = $f(x)$,

(2) $a_i = \{1/1^2, 1/2^2, \ldots, 1/i^2\}$ - ein in alle Richtungen skaliertes Problem: Hyperellipsoid. Das Achsenverhältnis der kürzesten zur längsten Achse beträgt $1 : 1/i^2$.

Die Skalierung des Hyperellipsoides wurde derart gewählt, um die gravimetrische Situation, in der jeder Geometrieparameter etwa mit dem inversen Quadrat des Abstands in die Qualitätsfunktion eingeht, zu simulieren. Die Komponenten der Startvektoren $\vec{x}_i$ wurden für alle folgenden Simulationen gleichverteilt aus dem Intervall $[0, 1)$ gewählt. Die Zufallszahlensequenzen wurden für alle Simulationen gleich gewählt. Um statistisch relevante Aussagen treffen zu können, werden die Durchschnittsverläufe aus 500 Simulationen gezeigt. Das Abbruchkriterium wurde auf 10^{-10} festgelegt. Die durchschnittliche Startqualität ist abhängig von der Anzahl der Parameter und aufgrund der Verwendung von Zufallszahlen für alle Simulationen leicht unterschiedlich.

Symmetrisches Problem (1)

Zunächst wurden die Simulationen für den symmetrischen Fall durchgeführt. Die Abbildung 4.1 zeigt die erreichte Qualität für 6 und 10 Parameter als Funktion der benötigten Funktionsaufrufe (Iterationen) für das Simplex-Verfahren und die Evolutionsstrategie mit konstanten a_i. In beiden Fällen konvergiert die Simplex-Methode schneller als die Evo-

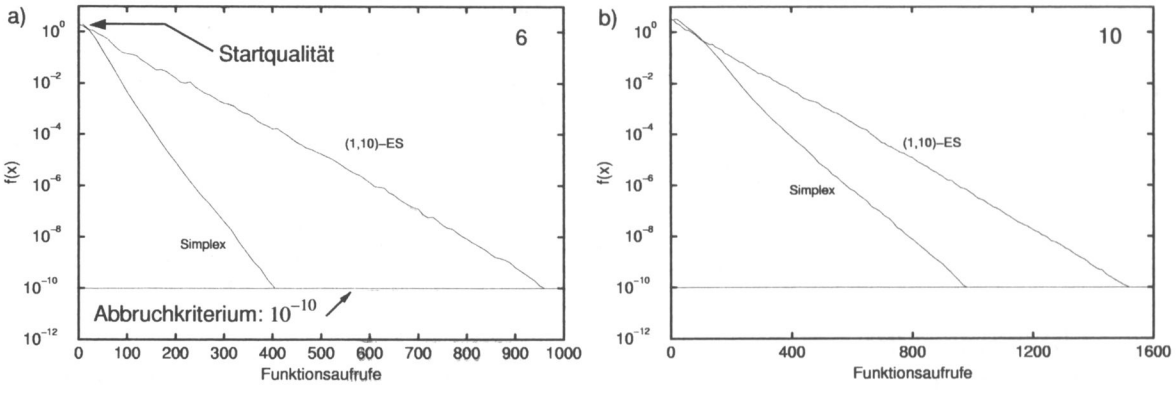


Abbildung 4.1: Vergleich der Evolutionsstrategie mit Simplex bei 6 (a) und 10 (b) freien Parametern.

lutionsstrategie. Bei 15 Parametern verhalten sich beide Verfahren ungefähr gleich. Die Situation verändert sich ab 20 Parameter zugunsten der ES (Abbildung 4.2).

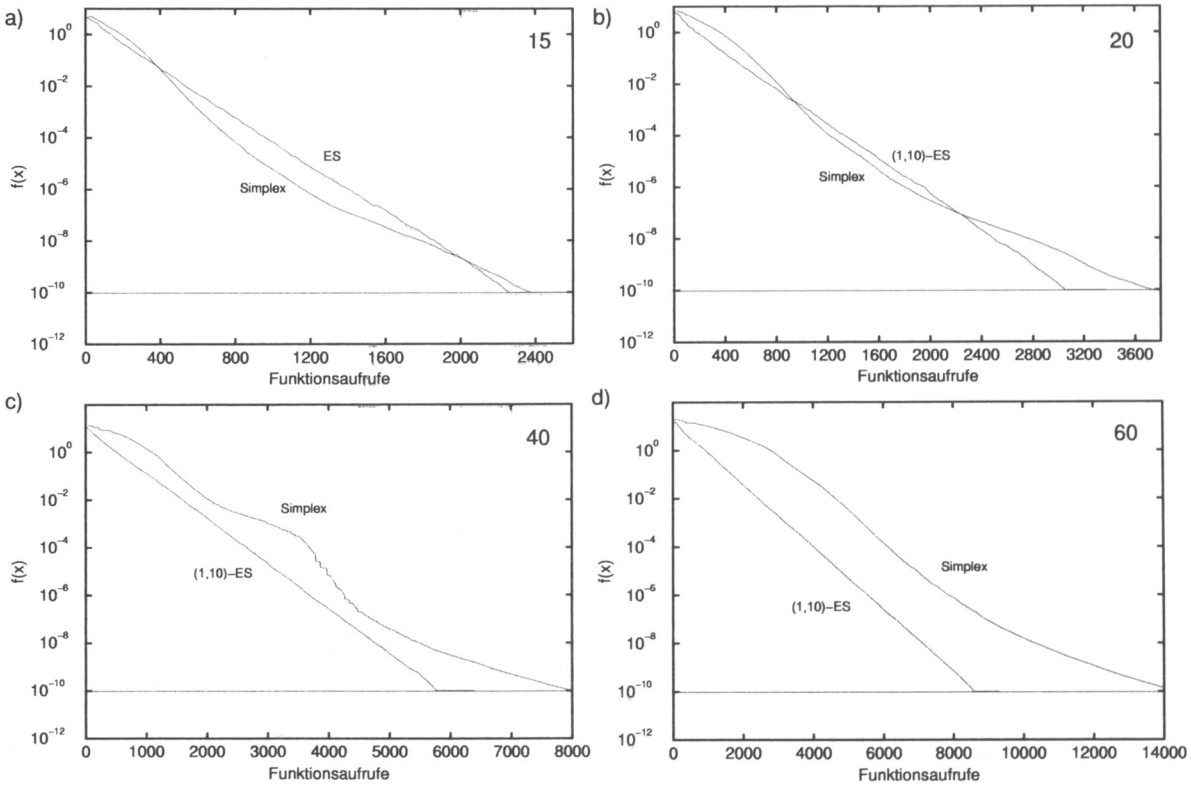


Abbildung 4.2: Vergleich der Evolutionsstrategie mit dem Simplex-Algorithmus. Die Anzahl der Parameter beträgt 15 (a), 20 (b), 40 (c) und 60 (d).

Wird die Dimension weiter erhöht, verschlechtert sich das Konvergenzverhalten des Simplex-Algorithmus stark, während die Evolutionsstrategie weiterhin logarithmisches Verhalten zeigt. Die Abbildung 4.3 zeigt die Simulationen für 100 und 200 Parameter.

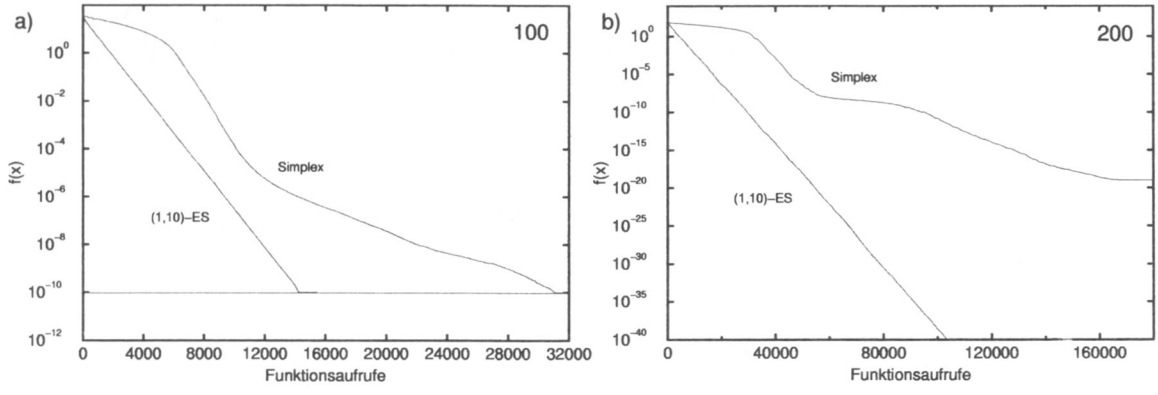


Abbildung 4.3: Ein Vergleich der Evolutionsstrategie mit Simplex für 100 (a) und 200 (b) Parameter. Die Evolutionsstrategie zeigt unabhängig von der Genauigkeit logarithmisches Verhalten. Zu beachten ist die unterschiedliche Skalierung der Abszisse in a) und b).

Aus den bisher gezeigten Durchschnittsverläufen können keine Aussagen über die Varianz, also die Abhängigkeit vom gewählten Startpunkt der Simulationen gemacht werden. Ideal wäre ein Verfahren, welches nicht vom gewählten Startpunkt abhängig ist, da es besser zu reproduzierende Ergebnisse liefern würde. Daher ist es wichtig, nicht nur das durchschnittliche Verhalten, sondern auch die Streuung der einzelnen Optimierungen zu untersuchen.

Die Abbildungen 4.4 zeigt 20 zufällig ausgewählte sowie den besten und den schlechtesten Verlauf für die Simplex-Methode und die Evolutionsstrategie. Die Anzahl der Parameter beträgt 20. Der Simplex-Algorithmus zeigt hier eine deutlich größere Varianz als die ES. Die schlechteste Simplex-Optimierung benötigt fast doppelt soviel Iterationen, um den Wert $f(x) = 10^{-10}$ zu erreichen, wie die Evolutionsstrategie.

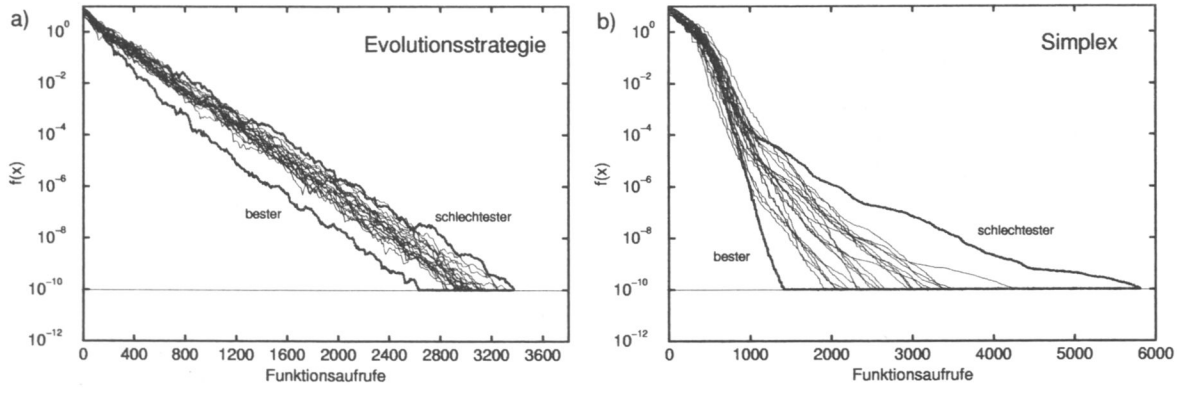


Abbildung 4.4: Vergleich der Evolutionsstrategie mit dem Simplex-Algorithmus für 20 Parameter. Dargestellt sind 20 zufällig aus 500 ausgewählten Simulationen sowie der beste und der schlechteste Verlauf der Simulation.

Die Varianz der Simplex-Optimierungen ist wesentlich größer als die der Evolutionsstrategie. Das Simplex-Verfahren ist damit sensitiver in bezug auf die Wahl des Startpunktes im Parameterraum. Dies ist bei aufwendig zu berechnenden Qualitätsfunktionen ein entscheidender Nachteil, da nicht ausgeschlossen werden kann, daß die Startposition derart gewählt wurde, daß die Optimierung sehr lange dauert.

Der Zusammenhang zwischen durchschnittlich benötigten Funktionsauswertungen und der Dimension ist in der Abbildung 4.5 dargestellt. Gezeigt ist die Anzahl der Funktionsaufrufe, die nötig waren, um die $f(x) = 10^{-10}$ zu erreichen. Die Simplex-Verläufe haben deutlich größere Varianzen als die der Evolutionsstrategie. Besonders negativ ist die Verschlechterung dieses Verhaltens bei steigender Anzahl von Parametern zu bewerten. Eine Optimierungsstrategie, deren Eigenschaften linear mit der Anzahl der zu optimierenden Parameter korreliert ist, ist dem Simplex-Algorithmus vorzuziehen.

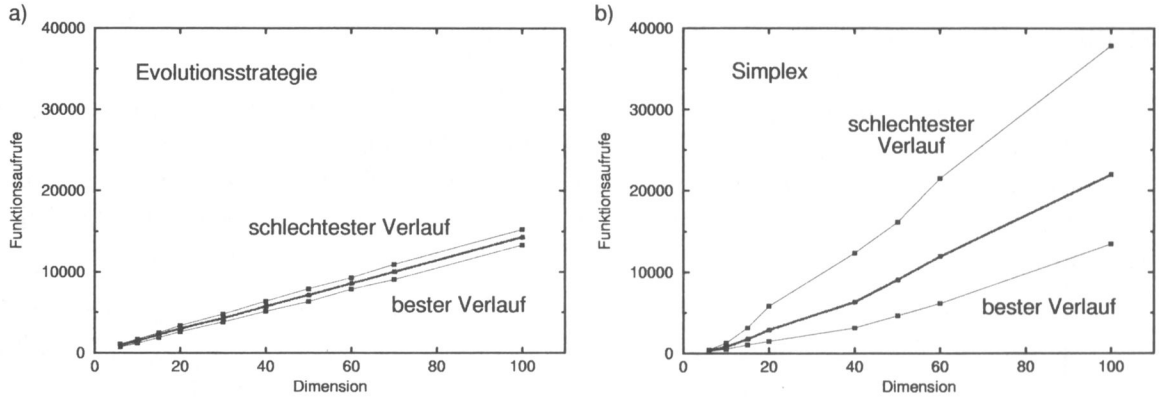


Abbildung 4.5: Die durchschnittliche Anzahl von Funktionsaufrufen als Funktion der Dimension des Parameterraumes. Gezeigt sind die Verläufe, die zur besten und schlechtesten Qualität geführt haben, sowie der Durchschnitt (mittlere Kurve) der Verläufe.

Zusammenfassung für konstante a_i (Hyperkugel)

Die Untersuchungen für das symmetrische Problem (konstante a_i) können wie folgt zusammengefaßt werden:

- Betrachtet man die durchschnittlichen Verläufe, konvergiert der Simplex-Algorithmus bis ungefähr 10 Parameter schneller als die Evolutionsstrategie.
- Bei Problemen mit mehr Parametern zeigt die Evolutionsstrategie höhere Konvergenzgeschwindigkeiten als der Simplex-Algorithmus.
- Die Konvergenzgeschwindigkeit der Evolutionsstrategie verbessert sich mit dem Logarithmus der erreichten Qualität.
- Mit der Evolutionsstrategie kann im Rahmen der Rechengenauigkeit des Computers jede beliebige Genauigkeit erreicht werden.

- Der Simplex-Algorithmus konvergiert bei Dimensionen größer 100 schlecht oder gar nicht.
- Das Konvergenzverhalten des Simplex-Algorithmus hängt stark vom gewählten Startvektor ab. Die Varianz ist größer als bei der Evolutionsstrategie, die praktisch unabhängig von den Startbedingungen ist.

Der Simplex-Algorithmus ist somit für nicht skalierte Probleme mit weniger als 10 Parametern durchaus geeignet. Leider erfüllen wenige Optimierungsprobleme diese Bedingung. Alle zu optimierenden Parameter haben i. allg. einen unterschiedlichen Einfluß auf die Qualitätsfunktion. Dies trifft speziell für gravimetrische und magnetische Probleme zu, da jeder Parameter ungefähr mit dem Inversen der zweiten und dritten Potenz seines Abstands vom Meßpunkt gewichtet in die Qualitätsfunktion eingeht. Daher ist eine gravimetrische Qualitätsfunktion praktisch immer skaliert.

Skaliertes Problem (2)

Um zu untersuchen, wie sich die beiden Verfahren bei skalierten Funktionen verhalten, wird $a_i = \{1/1^2, 1/2^2, \ldots, 1/i^2\}$ gewählt. Diese Skalierung soll näherungsweise die Verhältnisse einer gravimetrischen Qualitätsfunktion simulieren. Die Abbildung 4.6 zeigt wiederum die Konvergenzverhalten des Simplex-Algorithmus und der Evolutionsstrategie bei verschiedenen Dimensionen.

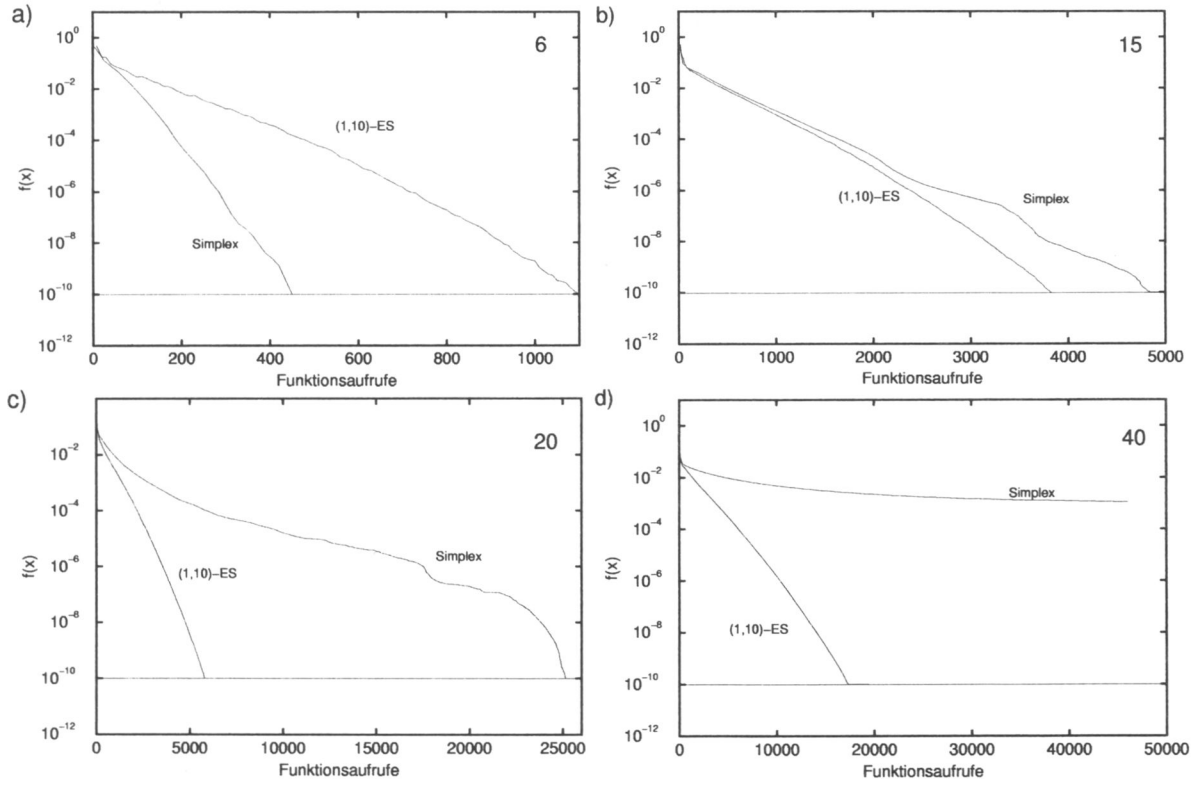


Abbildung 4.6: Vergleich des Simplex-Algorithmus mit der Evolutionsstrategie bei skalierter Qualitätsfunktion. Die Anzahl der Parameter beträgt 6 (a), 15 (b), 20 (c) und 40 (d). Gezeigt ist der Durchschnitt aus 500 Optimierungsläufen.

Die Evolutionsstrategie konvergiert logarithmisch. Bei 40 Parametern konvergiert Simplex im Mittel nicht mehr innerhalb von 45,000 Iterationen. Im Vergleich mit den in Abbildung 4.2 gezeigten Resultaten für konstante $a_i = 1$ wird deutlich, daß die Evolutionsstrategie bei dieser Skalierung im Vergleich zu Simplex ebenfalls deutlich bessere Ergebnisse liefert. Wesentlich ist aber die starke Zunahme der Varianz bei den Simplex-Optimierungen der skalierten Funktion.

Die Abbildung 4.7 zeigt das Verhalten beider Verfahren im Vergleich. Wiederum ist der Durchschnitt aus 500 Simulationen sowie der beste und der schlechteste Verlauf dargestellt.

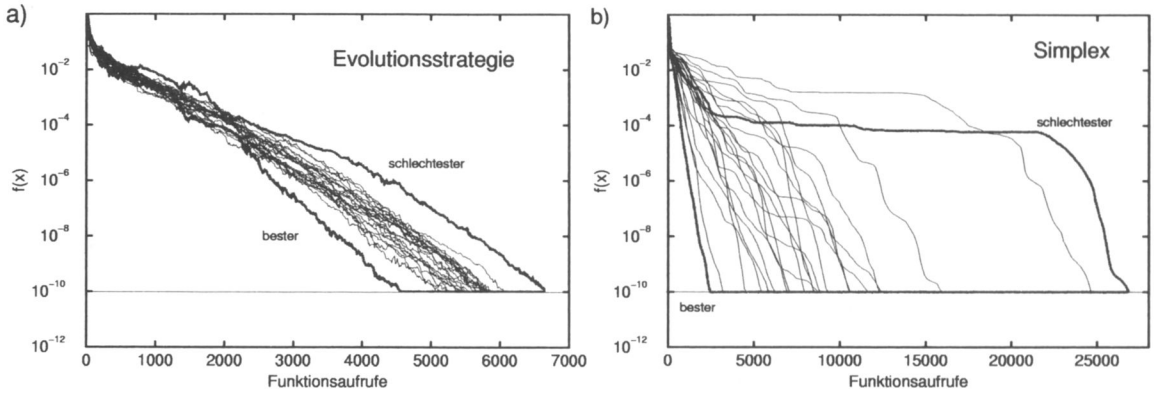


Abbildung 4.7: Vergleich des Simplex-Algorithmus und der Evolutionsstrategie. Die Dimension beträgt 20. Dargestellt sind einige aus 500 Simulationen zufällig ausgewählte sowie der beste und der schlechteste Verlauf. Gegenüber Abbildung 4.4 ist eine deutliche Zunahme der Varianz der Simplex-Simulationen zu erkennen.

Die Simplex-Methode verschlechtert ihre Konvergenzeigenschaften und konvergiert bei Dimensionen größer als 40 nicht mehr. Dieses Verhalten macht Simplex für den praktischen Einsatz bei Problemen mit mehr als 10 Parametern unbrauchbar. Eine derart starke Abhängigkeit vom gewählten Startpunkt kann bei aufwendig zu berechnenden Qualitätsfunktionen zu sehr langen Rechenzeiten führen bzw. dazu, daß der Algorithmus gar nicht konvergiert.

Zusammenfassung für skalierte a_i (Hyperellipsoid)

Nach den vorangegangenen Untersuchungen kann die Simplex-Methode auch bei skalierten Funktionen immer dann eingesetzt werden, wenn weniger als 10 Parameter zu optimieren sind. Je weniger Parameter, desto besser ist der Simplex-Algorithmus geeignet. Da Evolutionsstrategie weniger von gegebenen Startbedingungen abhängig ist, ist für Probleme mit mehr als 10 Parametern dieses Verfahren zu bevorzugen.

Die Abbildung 4.8 zeigt nochmals die durchschnittlichen, maximal und minimal benötigten Funktionsaufrufe in Abhängigkeit von der Dimension. Betrachtet man die benötigten Funktionsaufrufe, kann auch hier ein vorteilhaftes Verhalten von Evolutionsstrategie gezeigt werden.

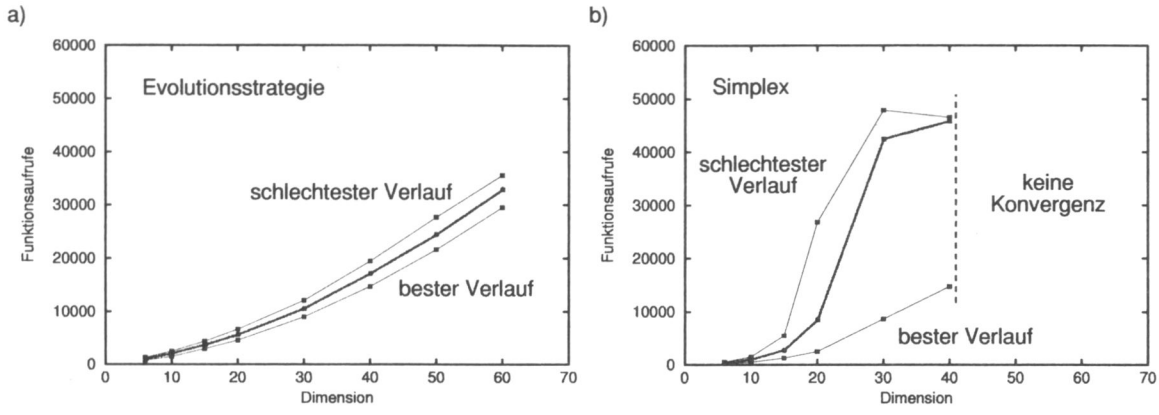


Abbildung 4.8: Aufgetragen sind die durchschnittlich benötigten Funktionsaufrufe (rot) sowie der beste und der schlechteste Verlauf für die Evolutionsstrategie (a) und Simplex (b) als Funktion der Dimension bei skalierter Qualitätsfunktion. Simplex konvergiert bei mehr als 40 Parametern nicht mehr.

4.2 Test der Algorithmen an einem gravimetrischen Modell

4.2.1 Das Testmodell

Um Aussagen über die Konvergenzeigenschaften der vorgestellten Algorithmen bei der Optimierung von geologisch interessanten Modellen treffen zu können, wird ein zweidimensionales Testmodell konstruiert. Das Modell ist ein vertikaler Schnitt durch einen Salzstock in Norddeutschland, welcher im Rahmen seismischer Untersuchungen erstellt wurde (Abbildung 4.9).

Die durchgeführten seismischen und gravimetrischen Modellierungen zeigten eine deutliche Aufteilung des Salzstocks in ein Salzkissen und einen Salzfuß (Schweitzer, pers. Mitt.). Daher soll hier untersucht werden, ob die zu testenden Algorithmen in der Lage sind, die bereits modellierten Strukturen nachzubilden oder andere, den Randbedingungen genügende Lösungen zu finden. In Abbildung 4.9 ist zu erkennen, daß die berechnete Schwereanomalie zu „negativ" ist, was offenbar durch den räumlich zu weit ausgedehnten Salzstock verursacht wird. Die Optimierungsalgorithmen sollten daher den zu großen Salzkörper verkleinern. Die gemessene Anomalie dieses Modells, welche auf den Meßwerten von insgesamt 325 Stationen basiert, ist mit ca. 4 mGal über dem ausgewählten Schnitt gering. Es soll untersucht werden, ob die implementierten Algorithmen auf der Basis einer so geringen Anomalie das Modell optimieren können. Erschwerend kommt bei diesem Testmodell hinzu, daß die Tertiärschicht mit einer geringen Dichte von 2.21 g/cm^3 die gravimetrische Wirkung des Salzstocks abschirmt.

Zur Optimierung wurden 26 Punkte, die den Salzstock des Startmodells in seiner Geometrie bestimmen, ausgewählt. Da jeder Punkt eine x- und eine z-Koordinate hat, wird der Raum, in dem die Optimierung stattfindet, von 52 Parametern (Dimensionen) aufgespannt. Die Dichten der Modellkörper wurden gemäß bekannten, für Norddeutschland typischen Werten (Schweitzer, pers. Mitt.) gewählt und während der Optimierung nicht verändert.
Ziel ist es, die Form des Salzkörpers zu optimieren. Die Randbedingungen der Geometrieparameter wurden derart gewählt, daß die Geometrieparameter größtmögliche Freiheit in

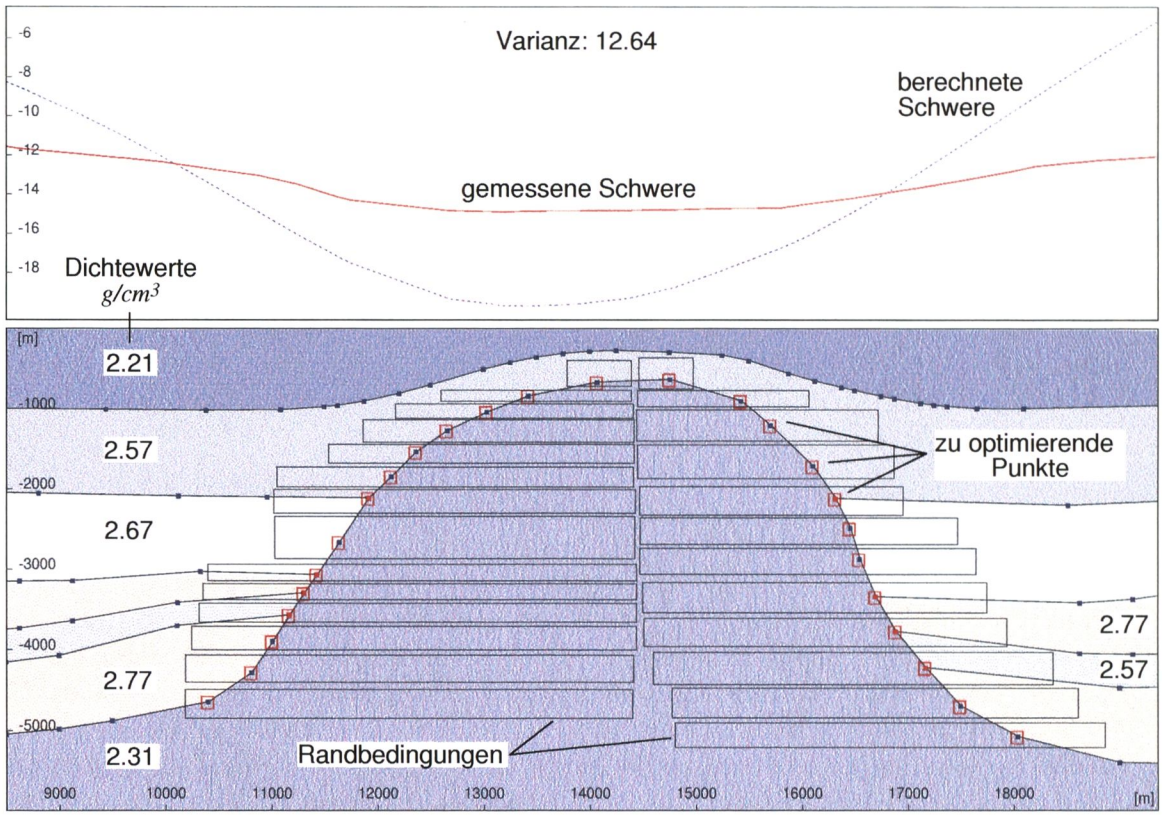


Abbildung 4.9: Startmodell zum Test der implementierten Optimierungsalgorithmen. Zu optimieren sind 26 geometrische Modellpunkte, deren Freiheit durch Randbedingungen eingeschränkt ist.

beiden Richtungen haben. Es soll damit dem im Abschnitt 3.11 beschriebenen Problem der zu eng gewählten Grenzen Rechnung getragen werden. Daher können im Sinne der Genese von Salzstöcken unrealistische Modelle nicht ausgeschlossen werden. Deshalb wird im folgenden Abschnitt ein Strafterm eingeführt, der derartige Modellbildungen verhindern soll.

4.2.2 Die Randbedingungen

Wie aus Abbildung 4.9 ersichtlich ist, sind die zu optimierenden Parameter in ihrer Bewegungsfreiheit durch Randbedingungen eingeschränkt (rechte, linke, obere und untere Grenze - schwarze Kästen). Diese wurden interaktiv, auf geologischen Vorstellungen basierend konstruiert. Da zu erwarten ist, daß sich der Salzstock während der Optimierung verkleinert, sind die Geometriegrenzen sinnvoll gewählt. Da sich die Kästen nicht überschneiden, wird die Wahrscheinlichkeit für die Überschneidung von Polygonseiten stark verringert bzw. ausgeschlossen.

4.2.3 Die Qualitätsfunktion

Die Qualitätsfunktion des Modells, für die das Minimum gesucht wird, setzt sich aus zwei Termen zusammen: der Varianz gemessener sowie berechneter Daten und einem Strafterm für spitze Winkel. Die Gesamtqualität des Modells wird mit folgender Gleichung berechnet:

$$Q_{gesamt} = \underbrace{\left[\sum_{i=1}^{n_s} (x_i^m - x_i^c)^2\right] n_s^{-1}}_{Varianz} + \underbrace{\left[\sum_{k=1}^{n_p} s\,\delta\right] n_p^{-1}}_{Strafterm} \qquad (4.2)$$

mit :

n_s Anzahl der Meßstationen,

n_p Anzahl der zu optimierenden Punkte,

x_i^m gemessene Daten,

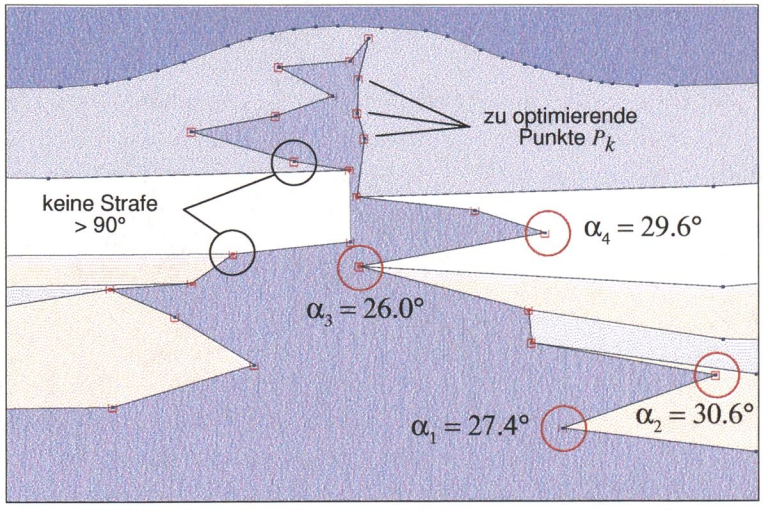
x_i^c berechnete Daten,

$\vec{P}_k$ zu optimierende Punkte,

s $< (\vec{P}_{k-1} - \vec{P}_k)\ ,\ (\vec{P}_k - \vec{P}_{k+1}) >$ (Skalarprodukt),

δ $\begin{cases} 0 & : \ s < 0 \\ 1 & : \ s > 0 \end{cases}$.

Vorangegangene Untersuchungen haben gezeigt, daß ohne einen solchen Strafterm zwar schnell eine gute Übereinstimmung von berechneten und gemessenen Daten gefunden wird, die dabei entstehenden Modelle aus geologischer Sicht aufgrund der unrealistischen Geometrie aber unbrauchbar sind. Leider reicht die Beschränkung der Modellparameter durch Randbedingungen nicht immer aus, dies völlig zu verhindern. Innerhalb der Ränder kann es Kombinationen der Parameter geben, die zu Modellen führen, wie sie in der Abbildung 4.10 gezeigt sind. Eine zu starke Einschränkung der Parameter führt aber unter Umständen dazu, daß das Erreichen eines Optimums ganz und gar ausgeschlossen wird.



Strafe für alle Winkel, die kleiner als 90° sind. Beispiele:

Winkel	Skalarprodukt
α1 = 27.4°	0.8876
α2 = 30.6°	0.8603
α3 = 26.0°	0.8984
α4 = 29.6°	0.8694
.	.
.	.

Die Summe der Skalarprodukte aller Winkel < 90° dividiert durch die Anzahl der Punkte in diesem Beispiel beträgt:

21.0778 / 26 = 0.8107.

Abbildung 4.10: Geologisch unrealistisches Modell zur Erläuterung des Strafterms. Die Strafe für spitze Winkel wird mit der Summe der Skalarprodukte, der durch die Punkte aufgespannten Winkel, berechnet.

Das in der Abbildung 4.10 gezeigte Beispielmodell hat eine gravimetrische Varianz von 0.52. Der durch die beschriebene Formulierung des Strafterms berechnete Wert von $21.0778/26 = 0.8107$ liegt somit im Bereich der Varianz des Modells. Dies hat den Vorteil,

daß der Strafterm unabhängig von der Anzahl der zu optimierenden Modellpunkte ist, da die Summe aller Skalarprodukte durch die Anzahl der Punkte dividiert wird. Es muß daher der Wert nicht in den Bereich der Varianz, also des ersten Qualitätsterms, skaliert werden. Am Anfang der Optimierung hat er somit ein kleineres Gewicht.

Die Varianz des in Abbildung 4.9 gezeigten Startmodells beträgt 12.64, der Strafterm des Startmodells beträgt 0.0. Die Optimierung wird daher eher im Sinne der Anpassung der Schwere verlaufen, und somit wird die Bedingung eines glatten Modells eine untergeordnete Rolle spielen.

Andere Gewichtungen dieses Terms resultierten in einer Erniedrigung der Konvergenzgeschwindigkeit und hatten keine Verbesserung im Hinblick auf die im Laufe der Optimierung entstehenden Modelle zur Folge (siehe dazu auch 4.2.4).

4.2.4 Evolutionsstrategie

Die Evolutionsstrategie hat sich bei den durchgeführten Tests gegenüber den anderen Verfahren als überlegen erwiesen. Daher sollen diese Ergebnisse zuerst vorgestellt werden. Wie im Abschnitt 4 wird hier der Algorithmus der (1,10)-ES-CMA angewendet. Folgende Punkte werden untersucht:

- Einfluß der Anzahl der Nachkommen (Populationsgröße),
- numerische Stabilität,
- Konvergenz von verschiedenen Ausgangsmodellen,
- Schrittweite zu Beginn der Optimierung,
- Strategien mit Rekombination,
- Verhalten ohne Kovarianzmatrixadaption und
- Einfluß der Gewichtung des Strafterms.

Optimiert wird das in Abschnitt 4.2.1 beschriebene Testmodell. Im folgenden wird der Optimierungslauf beschrieben, der zur besten je erreichten Qualität geführt hat, und dessen Verhalten analysiert.

4.2.4.1 Bester Optimierungslauf

Die Initialisierung des Algorithmus erfolgte zunächst mit einer im Vergleich zu den Ausdehnungen des Testmodells geringen Schrittweite von ca. 30 m. Der durchschnittliche euklidische Abstand vom Elternpunkt zu den mit dieser Schrittweite erzeugten Nachkommen ist somit gering, und der Algorithmus untersucht am Anfang des Optimierungsprozesses die nähere Umgebung des Startmodells. Im Laufe der Optimierung adaptiert der Algorithmus aus dem Differenzvektor zwischen Elternvektor und selektiertem Nachkommenvektor die erfolgreiche Richtung und die erfolgreiche Schrittweite (zum Vergleich siehe auch Abschnitt 3.9).

Die Abbildung 4.11 zeigt den Verlauf der Qualität der besten Optimierung. Während die Qualität in den ersten 10,000 Funktionsaufrufen deutlich in Richtung Minimum verläuft, findet in den weiteren 140,000 Funktionsaufrufen nur eine geringfügige Qualitätsverbesserung statt. Es sind deutliche Rückschritte zu erkennen, die teilweise zu Qualitätsverbesserungen führen. Dies beweist, daß der Algorithmus Verschlechterungen der Qualität akzeptieren kann, um Modelle zu erzeugen, die bessere Qualitäten als die bislang gefundenen haben. In Abbildung 4.11 b) ist bis ca. 500 Funktionsauswertungen deutlich die Adaptionsphase des Algorithmus zu erkennen.

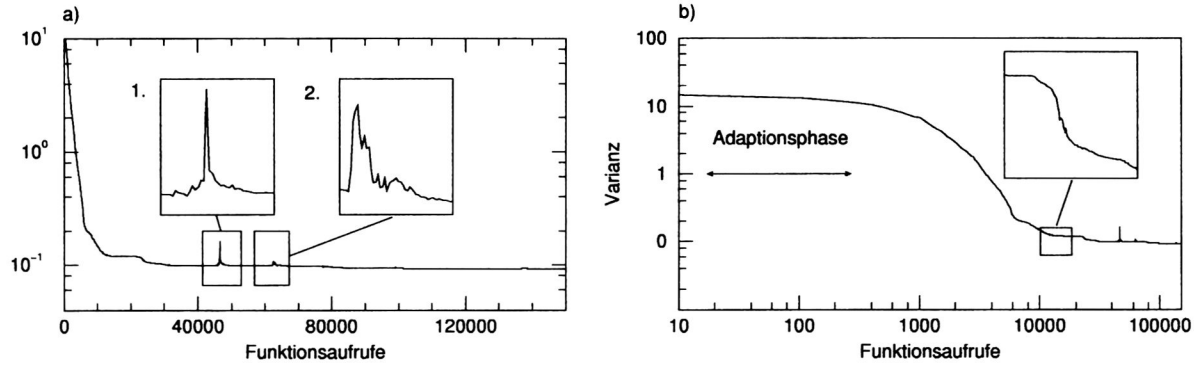


Abbildung 4.11: Verlauf der Varianz (Abszisse) einer Optimierung mit der (1,10)-ES-CMA in einfach logarithmischer (a) und in doppelt logarithmischer Darstellung (b). Es sind deutlich Verluste an Qualität zu erkennen: a) 1 und 2.

Die Abbildung 4.12 zeigt ein Beispiel für einen Rückschritt zwischen 60,000 und 70,000 Funktionsaufrufen (siehe auch Abbildung 4.11 a) Kasten 2).

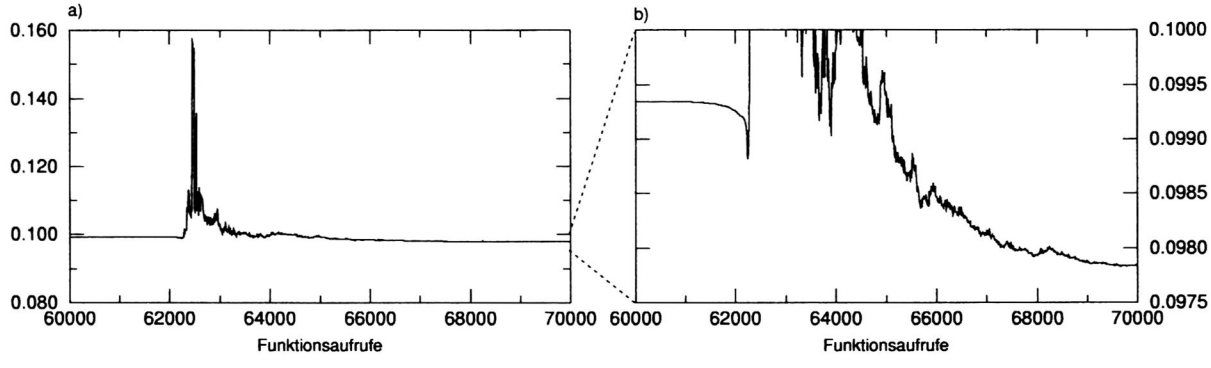


Abbildung 4.12: Verlust an Qualität. Diese Rückschritte führen in manchen Fällen zu Verbesserungen.

Interessant sind auch die Verläufe der durchschnittlichen Achsenlängen (Schrittweite) und der Achsenverhältnisse von größter zu kleinster Achse. Aus Abbildung 4.13 ist eine deutliche Zunahme der Achsenverhältnisse (a) und eine über mehrere Zehnerpotenzen schwankende Schrittweite ersichtlich. Gegen Ende der Optimierung nimmt die Schrittweite Werte vom Anfang des Prozesses an. Eine geometrische Interpretation ist ein sich in seiner Form, Größe und Richtung änderndes Mutationsellipsoid, welches sich durch die Qualitätslandschaft bewegt.

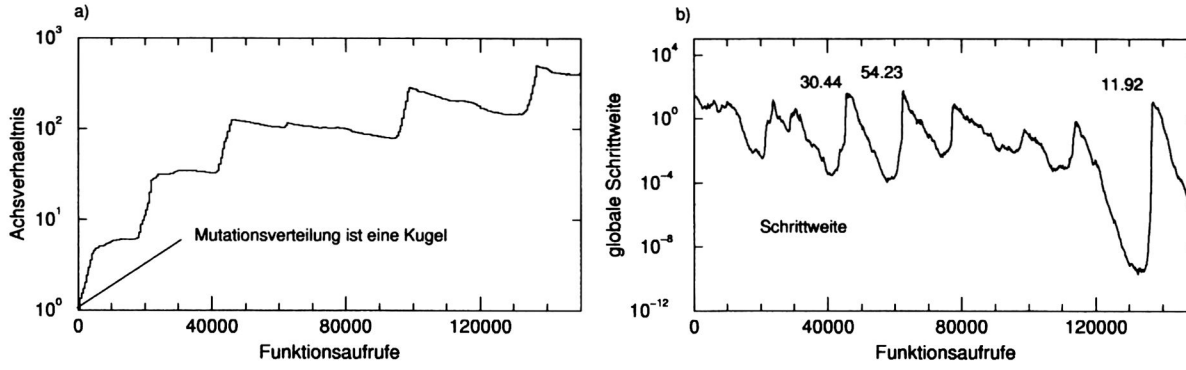


Abbildung 4.13: Die Entwicklung der Achsenverhältnisse (a) und der Größe des Mutationsellipsoides (b) bei der (1,10)-ES-CMA.

Die bisherigen Erläuterungen beziehen sich ausschließlich auf den numerischen Wert der Qualität des Modells, welcher mit der Qualitätsfunktion berechnet wird. Diese „Zahl" erlaubt aber keine Aussagen über die geologische Plausibilität des Modells. Die Abbildung 4.14 zeigt die Entwicklung des Modells im Laufe der Optimierung mit Evolutionsstrategie.

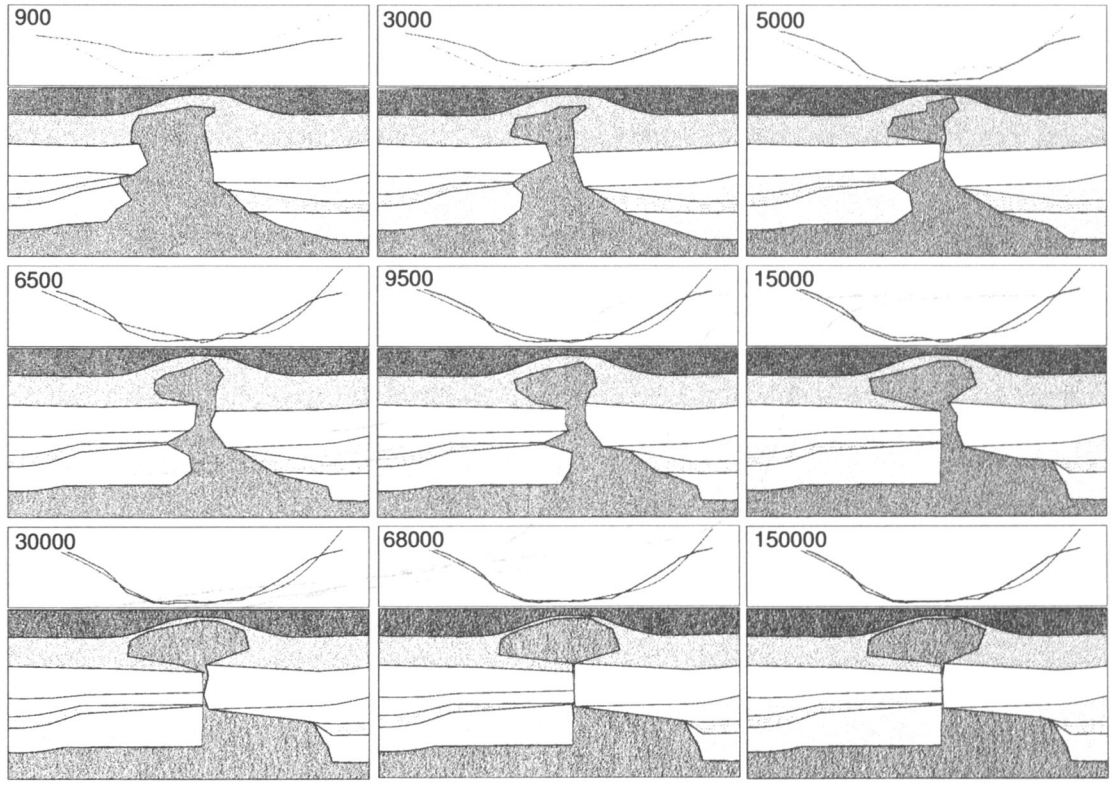


Abbildung 4.14: Momentaufnahmen der Optimierung des Testmodells. In der linken oberen Ecke jedes Teilbildes sind die jeweils bis zu diesem Stadium benötigten Funktionsaufrufe gezeigt. Zwischen 30,000 und 150,000 findet keine wesentliche Änderung der Geometrie der Salzstruktur statt.

Während der ersten 30,000 Funktionsaufrufe ändert sich die Geometrie signifikant. An der Übereinstimmung der Kurven der gemessenen und berechneten Daten ist eine deutliche Verbesserung der Qualität zu erkennen. In den verbleibenden Iterationen werden trotz

der geringfügigen Verbesserung der Qualität dennoch detaillierte Strukturinformationen gewonnen. Das Ergebnis nach 150,000 Funktionsaufrufen ist eine deutliche Teilung des Salzstocks in ein Salzkissen und einen Salzfuß. Der Algorithmus findet allerdings innerhalb der vorgegebenen Grenzen keine Lösung, die zu symmetrischen Schichten links und rechts des Salzstocks führt.

Die beste gravimetrische Varianz (der erste Term von Gleichung 4.2), die erreicht wurde, ist 0.0926. Fast alle Punkte befinden sich an ihren Randbedingungen und die Optimierung stagniert. In einem nächsten Schritt kann die Topologie des erzeugten Modells in der vom Algorithmus vorgeschlagenen Weise (Trennung des Salzkörpers) geändert und weiter optimiert werden. Hier sollen aber die Konvergenzeigenschaften der Evolutionsalgorithmen untersucht werden.

4.2.4.2 Anzahl der Nachkommen (Populationsgröße)

Im folgenden wird eine Untersuchung des Einflusses der Populationsgröße auf die Konvergenzeigenschaften des Algorithmus durchgeführt. Ziel ist es, besonders günstige Werte für die Anzahl der Nachkommen zu ermitteln bzw. den von Hansen und Ostermeier (1996) empfohlenen Wert von 10 Nachkommen für eine gravimetrische Modelloptimierung zu bestätigen. Es wurden Testläufe für mehrere Nachkommenzahlen durchgeführt. Das Abbruchkriterium wurde aufgrund der in 4.14 gezeigten Resultate und der begrenzten Rechenzeit auf 30,000 Funktionsaufrufe festgelegt. Die Abbildung 4.15 zeigt den Durchschnitt aus 30 Simulationen für jede Populationsgröße.

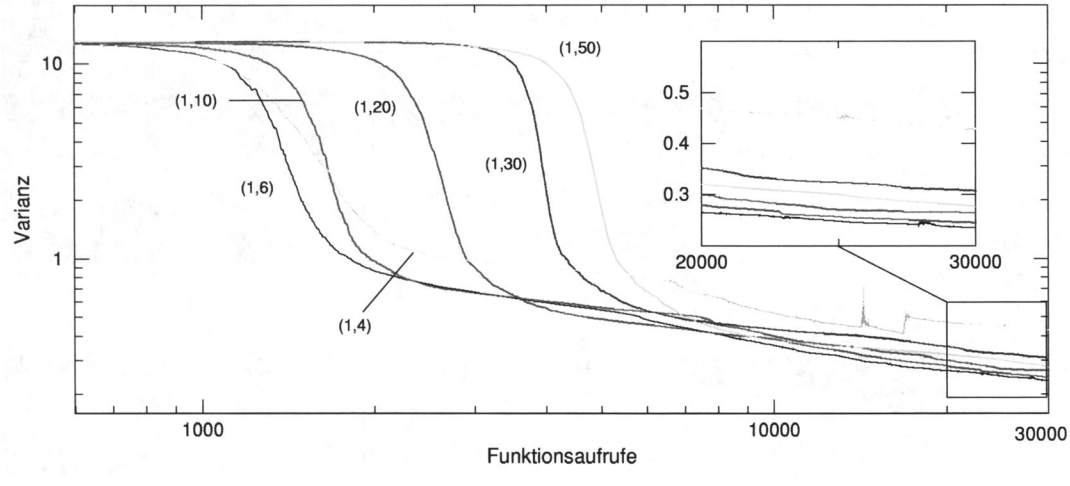


Abbildung 4.15: Vergleich verschiedener Populationsgrößen bei ES-CMA Algorithmen. Die Adaptionsphase der Kovarianzmatrix setzt für verschiedene Nachkommenzahlen zu unterschiedlichen Zeitpunkten ein.

Während Algorithmen mit weniger Nachkommen in der Anfangsphase schneller konvergieren, benötigt der Algorithmus für größere Populationen mehr Funktionsauswertungen, um die Struktur der Qualitätslandschaft zu adaptieren. Das beste mittlere Ergebnis wurde mit sechs Nachkommen erreicht. Strategien mit weniger als sechs Nachkommen konvergieren schlecht.

4.2.4.3 Numerische Stabilität

Die in 4.15 gezeigten Verläufe sind lediglich Beispiele für speziell ausgewählte Gegebenheiten und lassen keine statistischen Aussagen über das durchschnittliche Verhalten der Algorithmen zu. Für den praktischen Einsatz ist es aber von außerordentlicher Bedeutung, die numerischen Eigenschaften der Algorithmen zu kennen. Um beurteilen zu können, wie stark die Strategien von leicht veränderten Startbedingungen abhängen, müssen die Varianzen der einzelnen Verläufe untereinander betrachtet werden. Die Abbildung 4.16 zeigt die Verläufe der 30 Wiederholungen für 6 und 10 Nachkommen.

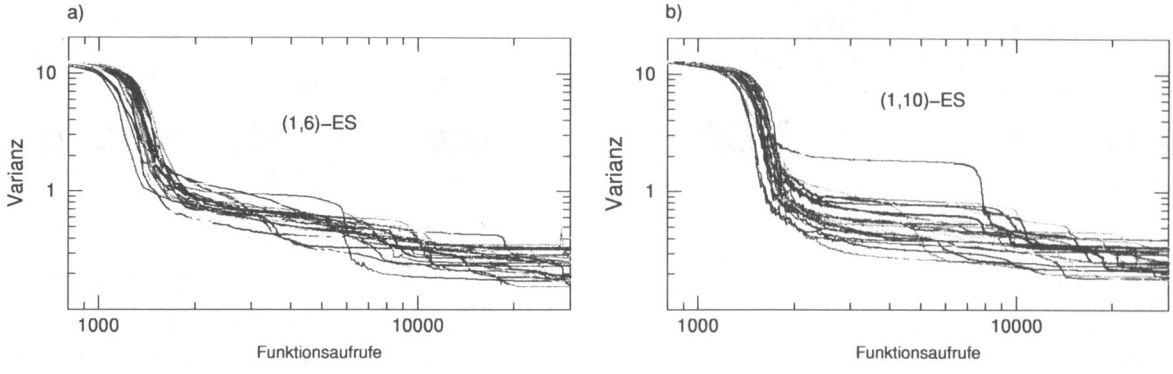


Abbildung 4.16: 30 Optimierungen mit 6 (a) und 10 (b) Nachkommen in doppelt logarithmischer Darstellung.

Es ist zu erkennen, daß die Algorithmen gering von der Variation des Startmodells abhängen und die einzelnen Optimierungen ähnlich verlaufen. Der Durchschnitt dieser drei Simulationen (Abbildung 4.17) zeigt, daß bis ca. 3,000 Funktionsaufrufe die Strategien mit weniger Nachkommen schneller konvergieren. Diese frühere Konvergenz ist für den späteren Verlauf unerheblich.

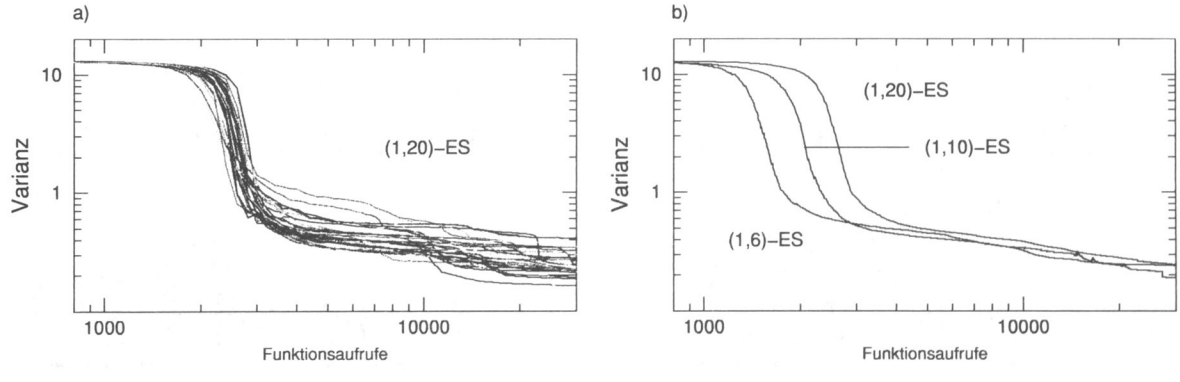


Abbildung 4.17: 30 Optimierungen mit 20 Nachkommen (a) und die Durchschnittsverläufe von 6, 10 und 20 Nachkommen (b). Alle Verfahren konvergieren nach der Adaption der Kovarianzmatrix sehr ähnlich.

Insgesamt verhält sich der Algorithmus numerisch sehr stabil. Bei Simulationen, die nach 30,000 Funktionsaufrufen relativ schlechte Qualitäten haben, sorgt der Adaptionsmechanismus des Verfahrens zu einem späteren Zeitpunkt für eine Verbesserung, so daß praktisch alle Verläufe schließlich konvergieren und das beste Ergebnis finden.

4.2.4.4 Konvergenz von verschiedenen Ausgangsmodellen

Nach der Analyse der numerischen Stabilität soll nun die Frage behandelt werden, wie verschiedene Startpositionen die Algorithmen beeinflussen. Dazu wurde das Testmodell aus Abbildung 4.9 modifiziert (Abbildung 4.18).

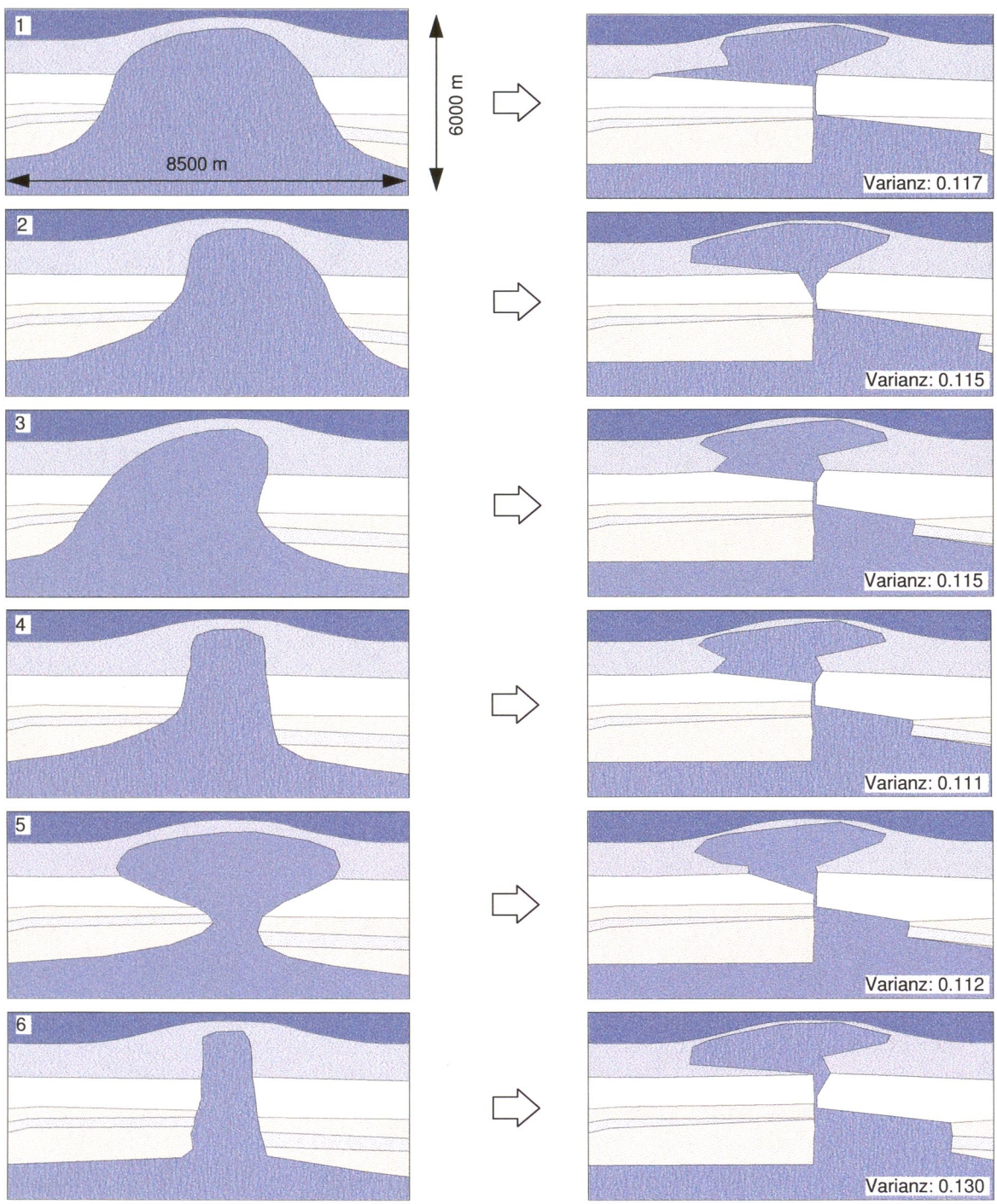


Abbildung 4.18: Von verschiedenen Startmodellen (links) gefundene Lösungen (rechts). In allen Fällen werden Modelle mit guten Qualitäten erzeugt. Auch die prinzipielle Struktur wird recht gut reproduziert.

Die Abbildung 4.18 zeigt links die generierten Startmodelle und rechts die entsprechenden optimierten Modelle. Es ist ersichtlich, daß nicht immer die gleiche Lösung gefunden wird und daß sich die gravimetrischen Varianzen nur gering unterscheiden. In allen Fällen wird annähernd die gleiche Struktur des Salzstocks gefunden. Von sehr verschiedenen Startmodellen ausgehend verlaufen die Qualitätskurven der Optimierungen sehr ähnlich (Abbildung 4.19).

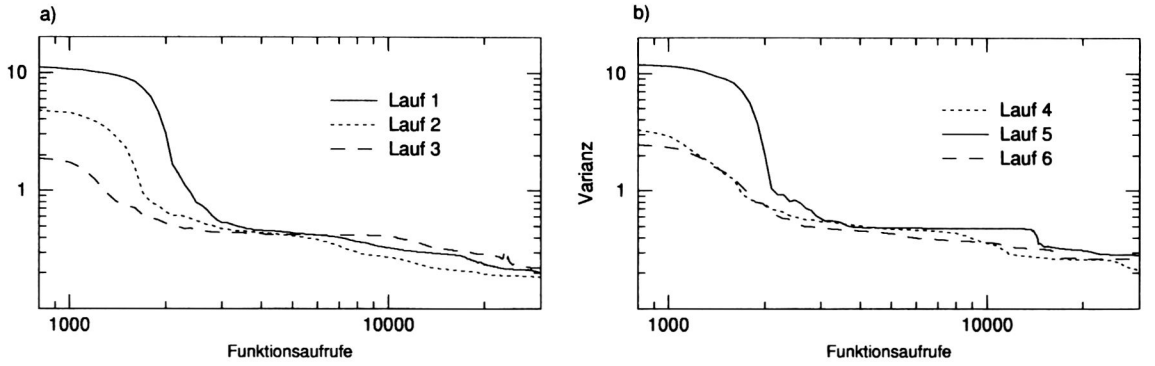


Abbildung 4.19: Verlauf der Qualitäten der Optimierungen aus der Abbildung 4.18.

Die Abbildung 4.19 zeigt die Verläufe der Qualitäten der in der Abbildung 4.18 gezeigten Optimierungen bis 30,000 Funktionsaufrufe. Es konnte gezeigt werden, daß der Algorithmus in der Lage ist, von jedem Punkt des Parameterraumes ausgehend, Lösungen zu finden, die eine gute Qualität haben und geologisch akzeptabel sind.

4.2.4.5 Startschrittweite

Ein weiterer wichtiger Punkt, der untersucht werden soll, ist der Einfluß der initialen Größe des Mutationsellipsoides am Beginn der Optimierung (Startschrittweite). Offensichtlich ist eine zu kleine Schrittweite uneffektiv, da die erzeugten Nachkommen dann sehr dicht am Elternpunkt liegen. Der Algorithmus kann zwar die Größe und die Form des Mutationsellipsoides im Laufe der Optimierung der Qualitätslandschaft anpassen, aber auch dieser Prozeß kostet Funktionsaufrufe und trägt nicht zur eigentlichen Optimierung des Modells bei. Es soll daher versucht werden, einen günstigen Wert zu finden, mit dem die Startschrittweite zu Beginn der Optimierung eingestellt werden kann. Dazu wird wiederum eine (1,10)-ES-CMA mit verschiedenen Startwerten auf das Testmodell angewendet. Um statistisch relevante Aussagen machen zu können, werden auch hier jeweils 30 Optimierungen betrachtet, die von leicht veränderten Startmodellen (siehe Abbildung 4.9) ausgehen. Die Abbildung 4.20 zeigt die Ergebnisse der Testrechnungen.

Optimierungen für Schrittweiten kleiner 1 zeigen nur sehr geringe Unterschiede. Die größeren Startwerte führen zu den besten Resultaten. Allerdings gehen alle Optimierungen nach mehr als 30,000 Funktionsaufrufen ineinander über. Betrachtet man die durchschnittlichen

Varianzen bzw. Standardabweichungen über alle Funktionsaufrufe, so ergibt sich auch hier ein Vorteil für größere Startschrittweiten (siehe Tabelle 4.1).

Da alle gefundenen Lösungen auch plausibel sind, kann nach diesen Untersuchungen die Größe des Mutationsellipsoides zu Beginn der Optimierung als relativ unkritisch angesehen werden. Der Algorithmus ist offenbar in der Lage, aus jeder beliebigen Startkonfiguration heraus das Mutationsellipsoid an die gegebene Qualitätslandschaft anzupassen. In einem Anwenderprogramm sollte für die Geometrieoptimierung bei Verwendung der Einheit Meter eine Startschrittweite zwischen 500 und 1000 und bei der Verwendung der Einheit Kilometer zwischen 0.5 und 1.0 angegeben werden.

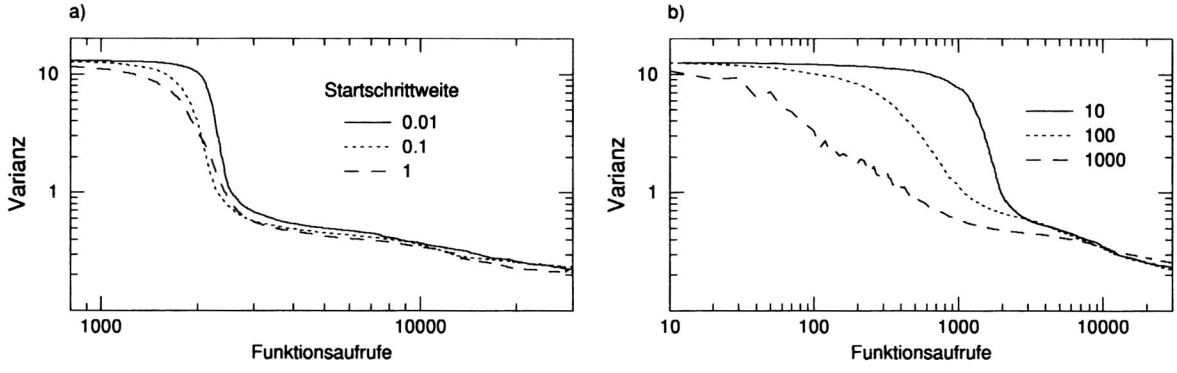


Abbildung 4.20: Durchschnittliche Verläufe für verschiedene Startschrittweiten. Die Optimierungen in a) unterscheiden sich nur sehr gering. In b) ist eine schnellere Konvergenz für größere Werte ersichtlich.

Startschrittweite	Varianz	Standardabweichung
0.01	0.0287898	0.169676
0.1	0.0209504	0.144743
1	0.0630968	0.251191
10	0.0213485	0.146111
100	0.0413185	0.203270
500	0.0044910	0.067015
1000	0.0022348 ←	0.047273 ←

Tabelle 4.1: Vergleich der durchschnittlichen Varianzen und der Standardabweichungen für die in Abbildung 4.20 gezeigten Optimierungen. Größere Startschrittweiten haben kleinere mittlere Varianzen.

4.2.4.6 Rekombination

Hier soll der Einfluß der Rekombination untersucht werden. Rekombination der μ besten Nachkommen ist immer dann sinnvoll, wenn die Auswertung der Qualitätsfunktion nicht fehlerfrei ist. Dies ist z.B. bei der Optimierung von realen Objekten, bei der die Bestimmung der Qualität des Objektes mit Meßfehlern behaftet sein kann, der Fall (Rechenberg,

1994). Die Rekombination sollte den Optimierungsprozeß auch dann positiv beeinflussen, wenn das Optimierungsproblem multimodal ist und die Nebenoptima, bezogen auf den gesamten Parameterraum, dicht beieinander liegen. Meßfehler werden hier vernachlässigt, da sie deutlich kleiner sind, als die in einer Optimierung erreichte Qualität. Der zweite Punkt kann nicht beurteilt werden. Es ist lediglich bekannt, daß aufgrund der Nichteindeutigkeit der Potentialverfahren mehrere gleichberechtigte Lösungen existieren. Über die Existenz mehrerer *Neben*optima kann daher nur spekuliert werden. Über deren eventuellen Abstand kann keine Aussage gemacht werden. Aus diesem Grund soll empirisch gezeigt werden, ob eine Strategie mit Rekombination eine Verbesserung der Konvergenzeigenschaften mit sich bringt oder nicht. In der Abbildung 4.21 sind die Ergebnisse einiger Experimente zusammengefaßt. Gezeigt sind wiederum Durchschnittsverläufe aus 30 Wiederholungen, die von leicht unterschiedlichen Startmodellen ausgingen. Es ist zu sehen, daß die Rekombination die Konvergenzgeschwindigkeit erhöht. Daher ist es empfehlenswert, mit Rekombinationen der Parameter zu arbeiten.

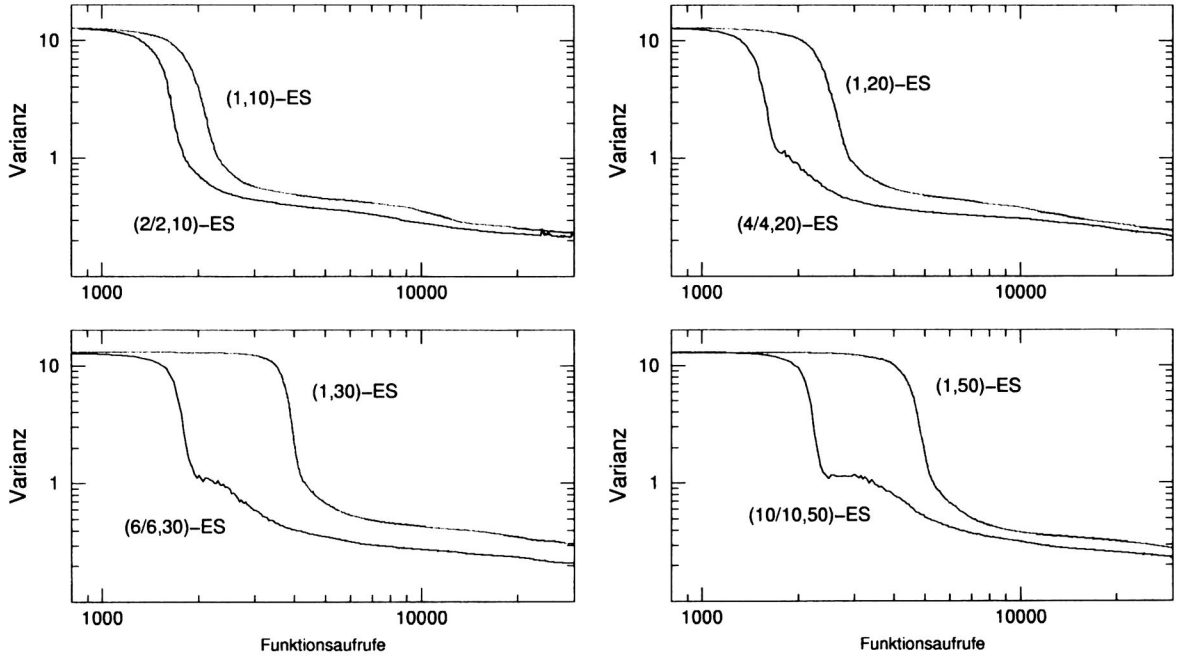


Abbildung 4.21: Die $(\mu/\rho, \lambda)$-ES-CMA mit $\rho = \mu$. Die Konvergenzgeschwindigkeiten der Algorithmen verbessern sich durch Rekombination. Die besten Resultate erreichte die (6/6, 30)-ES.

4.2.4.7 Evolutionsstrategien ohne Kovarianzmatrixadaption

Desweiteren soll der Algorithmus der (1,10)-ES-CMA mit den Evolutionsstrategien ohne Kovarianzmatrixadaption verglichen werden. Diese Algorithmen sind für Problemem ohne starke Korrelationen zwischen den Parametern anwendbar und haben sich bei vielen Optimierungsproblemen bewährt. Für Probleme der gravimetrischen Optimierung mit Randbedingungen führen sie allerdings nicht zu Modellen guter Qualität und konvergieren schlechter als die Algorithmen mit Kovarianzmatrixadaption. Zum Test wurden die Untersuchungen, die für verschiedene Startschrittweiten durchgeführt wurden, ohne Adaption wiederholt. Die Abbildung 4.22 zeigt die drei besten Modelle, die mit einer (1,10)-ES ohne

Kovarianzmatrixadaption nach ca. 5.4 Millionen Funktionsauswertungen gefunden wurden. Die Startschrittweiten betrugen 0.1, 10 und 500.

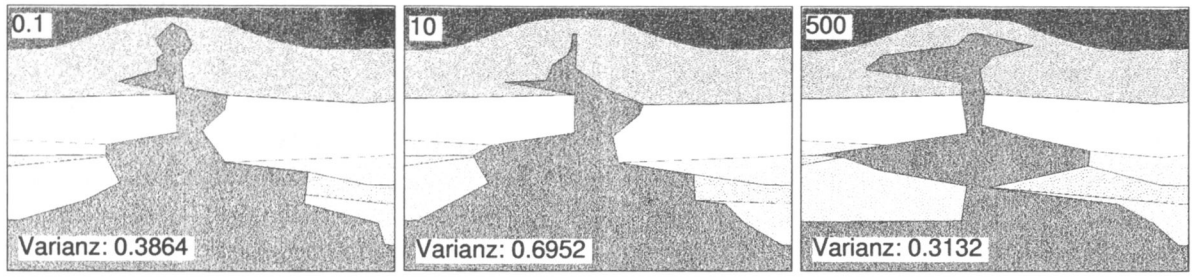


Abbildung 4.22: Die besten Modelle, die ohne Kovarianzmatrixadaption gefunden wurden. Die Varianzen sind größer als bei CMA-Strategien.

Die besten Modelle, die ohne Kovarianzmatrixadaption gefunden wurden, sind im Hinblick auf die Genese von Salzstöcken geologisch wenig realistisch (Guillemont, 1971). Auch der in Abbildung 4.23 gezeigte Vergleich der Algorithmen mit und ohne Adaption zeigt die Überlegenheit des CMA Algorithmus.

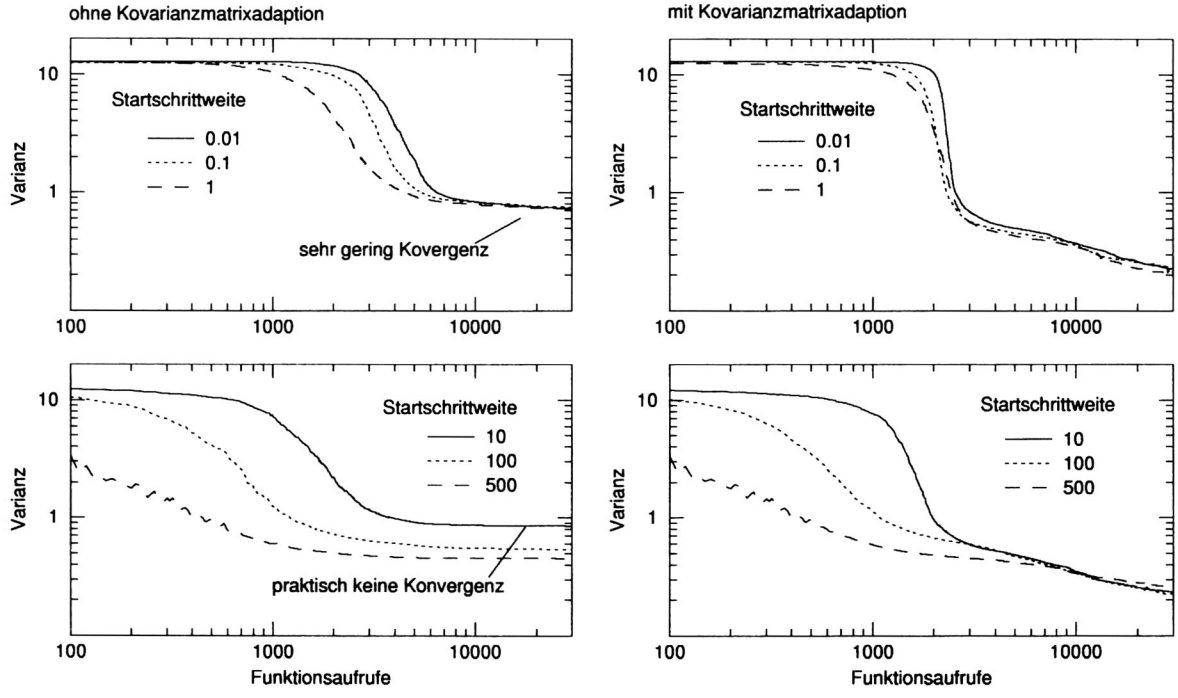


Abbildung 4.23: Vergleich der (1,10)-ES und der (1,10)-ES-CMA für verschiedene Startschrittweiten. Der CMA Algorithmus hat bessere Konvergenzeigenschaften.

Da die erste Ableitung der Kurven der (1,10)-ES ohne CMA fast 0 ist, liegt eine Interpretation lokaler Optima nahe. Es kann nicht gesagt werden, ob es sich um suboptimale Lösungen handelt. Wenn ja, so ist der Kovarianzmatrixadaptionsalgorithmus in der Lage, diese zu verlassen.

4.2.4.8 Der Einfluß des Strafterms

Abschließend soll der Einfluß des Strafterms auf die Konvergenzeigenschaften der Algorithmen untersucht werden. Dazu werden wiederum jeweils 30 Optimierungen für verschiedene Gewichtungen des Strafterms durchgeführt. Die Abbildung 4.24 zeigt einige durchschnittliche Verläufe für die Gewichtungen 0 (kein Strafterm), 1 (ohne Gewichtung: die Summe der Skalarprodukte für Winkel kleiner 90 Grad), 10 und 100. Im Gegensatz zu den vorangegangenen Abbildungen ist in Abbildung 4.24 nur die gravimetrische Varianz dargestellt und nicht die Gesamtqualität nach Gleichung 4.2. Diese Darstellung zeigt, daß die vorgeschlagene Berechnung des Strafterms (siehe Gleichung 4.2) ohne zusätzliche Gewichtung nicht die besten Qualitäten liefert. Obwohl der Verlauf ohne Strafterm zu guten Qualitäten führt, sind die Ergebnisse instabil, d.h. die durchschnittliche Varianz der Verläufe ist groß. Die Gewichtung mit 0.1 (nicht dargestellt) verbessert zwar die erreichte gravimetrische Qualität gegenüber der Gewichtung mit dem Faktor 1. Die dabei entstehenden Modelle enthalten allerdings häufig unrealistisch spitze Winkel, welche durch den Strafterm gerade vermieden werden sollen.

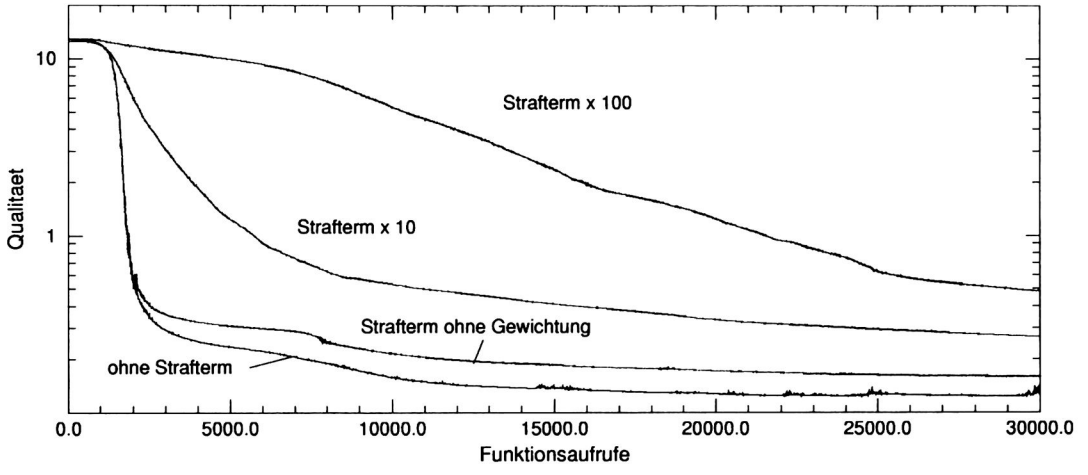


Abbildung 4.24: Verschiedene Gewichtungen des Strafterms.

Die folgende Abbildung zeigt die zwei besten gefundenen Modelle für die Optimierung ohne Strafterm und mit Strafterm (gewichtet mit 0.1).

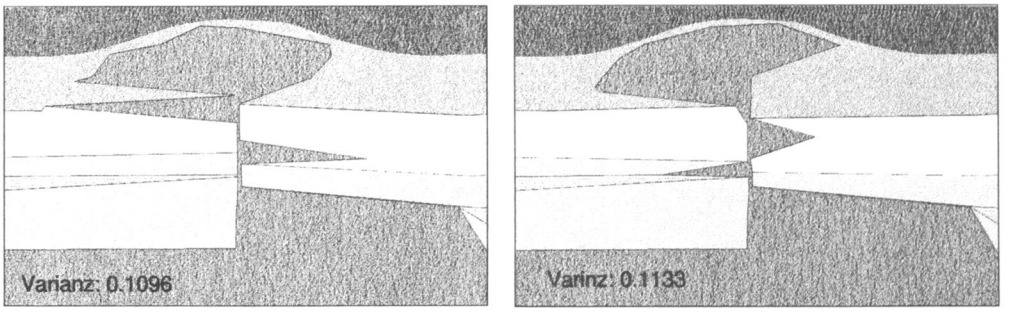


Abbildung 4.25: Modelle entstanden ohne (links) und mit 0.1 gewichtetem Strafterm (rechts).

Es sei betont, daß die Beispiele aus Abbildung 4.25 insgesamt betrachtet als die besten zu beurteilen sind. Die Mehrzahl der Modelle, besonders bei der Optimierung ohne Strafterm, müssen als geologisch unrealistisch bezeichnet werden. Fazit ist, daß der Strafterm ohne Gewichtung den besten Kompromiß zwischen geologischer Plausibilität und gefundener gravimetrischer Varianz ist.

4.2.4.9 Zusammenfassung der Ergebnisse für die Anwendung der Evolutionsstrategie

Es konnte anhand eines Modellbeispiels gezeigt werden, daß die Evolutionsstrategie mit Kovarianzmatrixadaption ein Optimierungsverfahren ist, welches unter verschiedenen Randbedingungen (Strafterm und Geometriegrenzen) eine beträchtliche Anzahl von freien Parametern (im Beispiel: 52) in angemessener Größenordnung von Funktionsaufrufen und somit in angemessener Rechenzeit optimieren kann.

Die Algorithmen sind weitestgehend unabhängig von strategieinternen Parametern und verhalten sich numerisch stabil gegenüber kleinen Variationen der Ausgangsmodelle und der Verwendung von Sequenzen anderer Zufallszahlen. Sie finden, von sehr unterschiedlichen Ausgangssituationen ausgehend, Lösungen, die der besten gefundenen sehr ähnlich sind und sich in der Standardabweichung geringfügig unterscheiden.

Die Evolutionsstrategie zeigt ein sehr gutes Verhalten, wenn es um die „Feineinstellung" der Parameter geht. Auch globale Aspekte der Qualitätsfunktion werden vom Adaptionsmechanismus gut erkannt, was praktisch immer zu einer schnellen und sicheren Konvergenz des Algorithmus führt.

Es wurde ein Strafterm für die „Glattheit" des Modells formuliert, der bei der Optimierung zu geologisch plausiblen Modellen führt. Ebenso konnte gezeigt werden, daß andere Gewichtungen dieses Terms nicht sinnvoll sind.

Für den praktischen Einsatz der Methode werden folgende Strategieparameter vorgeschlagen:

- Größe der Population (λ): 6 bis 10 Individuen,
- Anzahl der Eltern (μ): 1/5 λ mit Rekombination, sonst 1,
- Anzahl der Individuen für die Rekombination: μ,
- Startschrittweite: 1000 [m] bzw. 1 [km].
- Kovarianzmatrixadaption ist immer zu empfehlen.

4.2.5 Genetische Algorithmen

Die Idee, die den genetischen Algorithmen zugrunde liegt, ist der der Evolutionsstrategien ähnlich. Daher soll nach der Anwendung von Evolutionsstrategien auch die genetischen Algorithmen untersucht werden. Genetische Algorithmen wurden bereits vielfach

zur Optimierung von geophysikalischen Problemen eingesetzt. Hier sind z.B. Boschetti et al. (1996, 1997), Stoffa und Sen (1994) und Sambridg und Drijkoningen (1992) zu nennen.

Im folgenden wird wiederum das in Abschnitt 4.9 beschriebene Testmodell optimiert. Auch der dort beschriebene Strafterm wird wie bei den Evolutionsstrategien (siehe Abschnitt 3.9) benutzt. Der Einfluß folgender algorithmischer Parameter wird untersucht:

- Größe der Population,
- Länge der Bitstrings (Diskretisierung),
- Mutationswahrscheinlichkeit,
- Rekombinationswahrscheinlichkeit und Rekombinationsschemen,
- Basis des Codierungssystems und
- Verwendung verschiedener Ersetzungsschemen.

4.2.5.1 Größe der Population

Der „Motor" der Optimierung bei genetischen Algorithmen ist die Rekombination der Individuen guter Qualität einer Population. Die Größe der Population sollte neben dem verwendeten Selektionsschema Einfluß auf das Konvergenzverhalten des Algorithmus haben. Daher werden Optimierungen des Testmodells mit verschiedenen Populationsgrößen durchgeführt. Die Abbildung 4.26 zeigt die Durchschnittsverläufe der Qualitäten für verschiedene Populationsgrößen.

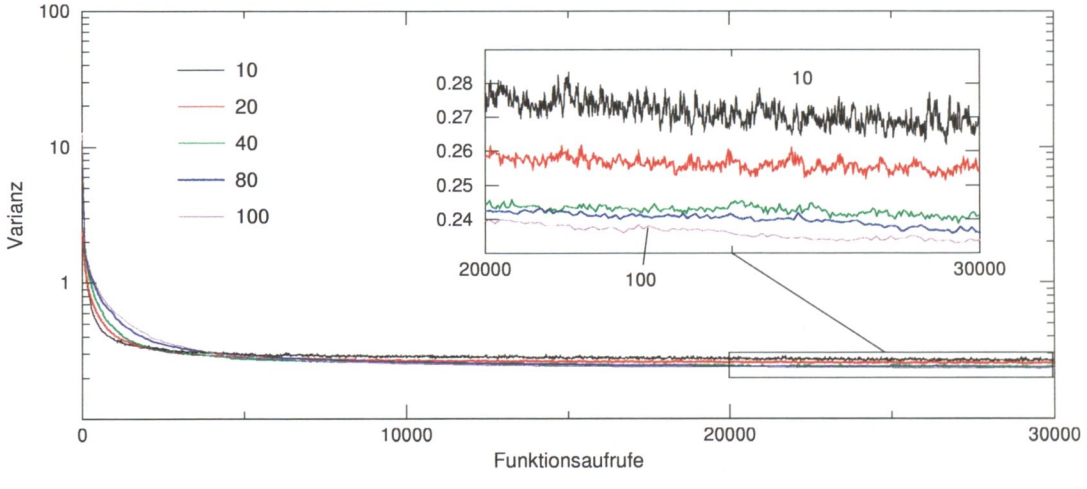


Abbildung 4.26: Vergleich verschiedener Populationsgrößen bei genetischen Algorithmen. Größere Populationen erreichen im Durchschnitt bessere Qualitäten.

Die nach 30,000 Funktionsaufrufen durchschnittlich erreichten Qualitäten unterscheiden sich geringfügig. Der Verlauf mit 100 Individuen hat die größte Konvergenzgeschwindigkeit. Die Standardabweichung der einzelnen Verläufe untereinander ist am geringsten. Noch größere Populationen führen zu keinen Verbesserungen der erreichten Qualität bzw. des Konvergenzverhaltens. Die Abbildung 4.27 zeigt die Verläufe für die Populationsgrößen 100, 200, 300 und 500 in doppelt logarithmischer Darstellung. Ein Vorteil ist für 200 Individuen

zu erkennen, während 500 Individuen eine deutliche Verschlechterung der Konvergenzeigenschaften zur Folge haben.

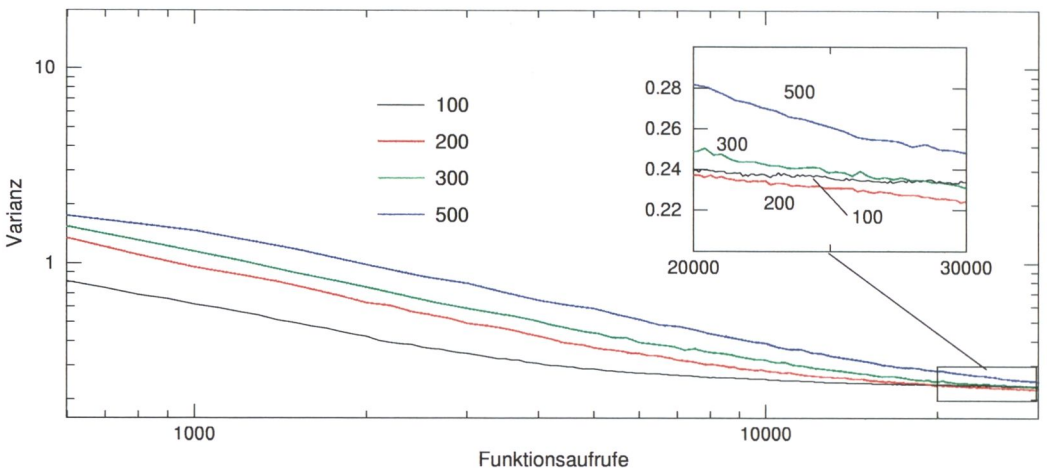


Abbildung 4.27: Vergleich der Populationsgrößen 100, 200, 300 und 500 bei genetischen Algorithmen.

In der Tabelle 4.2 sind einige charakteristische numerische Werte für verschiedene Populationsgrößen zusammengefaßt. Die beste erreichte Qualität bezieht sich auf insgesamt ca. eine Million Funktionsaufrufe pro Populationsgröße.

Populationsgröße	beste Qualität	durchschnittliche Qualität	durchschnittliche Standardabweichung
10	0.1295	0.2681	0.0040
40	0.2412	0.2280	0.0022
60	0.1710	0.2383	0.0017
80	0.1713	0.2360	0.0021
100	0.1271 ←	0.2337	0.0017
200	0.1889	0.2244 ←	0.0041
500	0.2003	0.2481	0.0035

Tabelle 4.2: Zusammenfassung der Ergebnisse der Simulationen für verschiedene Populationsgrößen bei genetischen Algorithmen.

Die Populationsgröße kann nach diesen Resultaten als unkritisch bezeichnet werden. Das beste Ergebnis wurde mit 100 Individuen gefunden; die besten mittleren Ergebnisse lieferten die Simulationen mit 200 Individuen, so daß Populationsgrößen zwischen 100 und 200 als günstig bezeichnet werden können.

4.2.5.2 Länge der Bitstrings

Die Länge der verwendeten Bitstrings für jeden Parameter (siehe auch Abschnitt 3.8) ist direkt mit der möglichen Auflösung der Modellparameter korreliert. Beträgt die Länge z.B. 1, können nur zwei Zustände realisiert werden: 0 und 1. Daher können sich die Parameter nur an ihren Extrema, Minimum (0 - das Bit ist nicht gesetzt) und Maximum (1 - das Bit ist gesetzt), befinden. Werden die Strings verlängert, sind feinere „Abstufungen“ zwischen

Minimum und Maximum möglich. Im allgemeinen gilt: je länger die Bitstrings, desto besser die numerische Modellauflösung.

Es soll der Einfluß dieses Strategieparameters auf die Konvergenzgeschwindigkeit und die Geometrie der gefundenen Modelle untersucht werden. Basierend auf den vorangegangenen Untersuchungen wurde für die folgenden Simulationen die Populationsgröße auf 100 Individuen festgelegt. Um statistische Aussagen treffen zu können, wurden alle Simulationen 30 mal mit unterschiedlichen Zufallszahlensequenzen und damit unterschiedlich initialisierten Startpopulationen wiederholt. Die Abbildung 4.28 zeigt die Resultate.

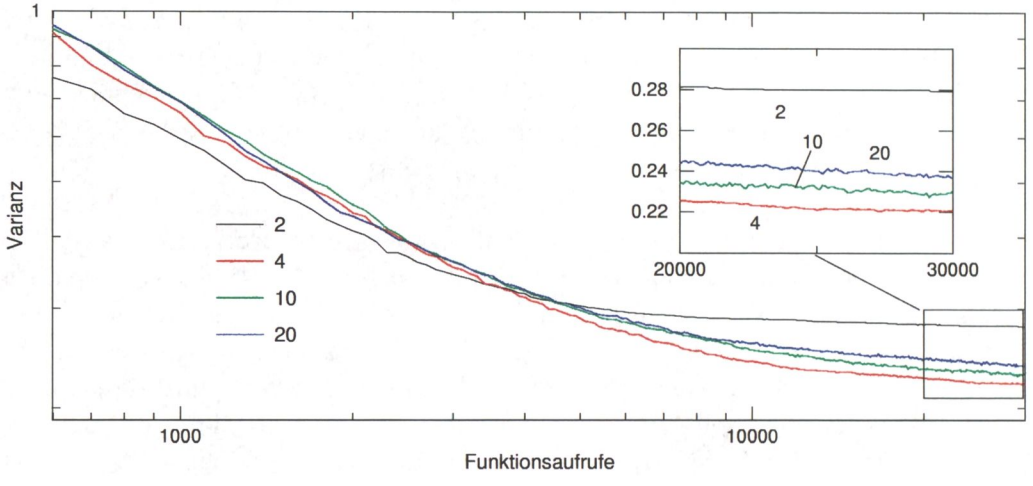


Abbildung 4.28: Vergleich verschiedener Bitstringlängen bei genetischen Algorithmen. Die Simulationen mit vier Bits zeigen bestes mittleres Verhalten.

Erstaunlicherweise gibt es einen Vorteil für eine Länge von vier Bits pro Parameter. Auch die mit kurzen Bitstrings in einer Million Funktionsaufrufen gefundenen besten Modelle haben sehr gute Qualitäten (siehe Abbildung 4.29), so daß nach diesen Untersuchungen Strings mit weniger als sechs Bits zu bevorzugen sind.

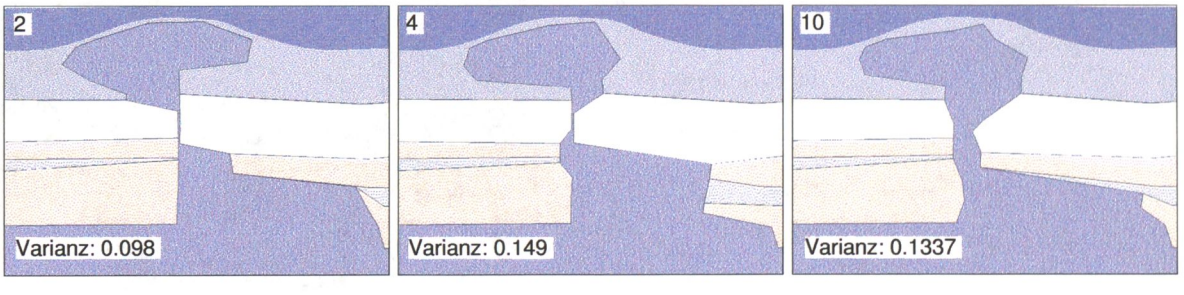


Abbildung 4.29: Die besten je gefundenen Modelle für Bitstringlängen zwei, vier und 10. Kurze Strings erzeugen „numerisch" gute Modelle.

Genau betrachtet, ist es durchaus sinnvoll, den Parameterraum zunächst mit „einem gröberen Raster" abzusuchen. Folgende kurze Überlegung soll dies verdeutlichen: Im Testmodell werden 26 Geometriepunkte optimiert; so ergeben sich wegen der zwei Ortskoordinaten jedes Punktes 52 Parameter. Beträgt die Bitlänge z.B. „zwei" erhält man $2^{104} = 2.0210^{31}$ mögliche Zustände. Die Anzahl der möglichen Konstellationen für Bitstringlänge 10 beträgt hingegen $2^{520} = 3.4310^{156}$. Die Differenz zwischen den möglichen Realisierungen beträgt

beträgt somit ca. 125 Zehnerpotenzen. Dieses Zahl verdeutlicht, wie sinnvoll ein derartiges Vorgehen sein kann.

4.2.5.3 Mutationswahrscheinlichkeit

Im Laufe der Optimierung mit genetischen Algorithmen tritt leicht der Fall auf, daß die Population mehrere gleiche Individuen enthält. Diese entstehen meist durch Rekombination von Individuen, deren Bitstrings ähnliche Muster haben. Sind zu viele gleichartige Individuen in einer Population und haben diese gute Qualitätswerte, so daß sie häufig selektiert werden, kommt es zu einem Zustand, in dem es nur noch einen einzigen Typ von Bitstrings gibt. Rekombiniert man zwei (gleiche) Bitstrings wird das Produkt immer der gleiche Bitstring sein und die Optimierung stagniert völlig. Die Wahrscheinlichkeit, daß dieser Fall auftritt, ist mit der Länge der Bitstrings korreliert. Je länger die Strings, desto geringer ist die Wahrscheinlichkeit.

Durch die Einführung einer Mutationswahrscheinlichkeit wird für Variabilität in der Population gesorgt. Die Bits werden vom Algorithmus mit dieser Mutationswahrscheinlichkeit verändert. Ist das benutzte Codierungssystem binär, werden die ausgewählten Bits negiert. Ist die Mutationswahrscheinlichkeit sehr hoch, wird die Optimierung hauptsächlich von Mutationen bestimmt und die Eigenschaft der Rekombinationssuche wird überdeckt. Die Mutationswahrscheinlichkeit sollte demnach so eingestellt werden, daß der Algorithmus nicht die beschriebenen Nachteile der Reproduktion gleicher Individuen aufweist und die Rekombinationssuche wenig gestört wird.

Um günstige Werte für Mutationswahrscheinlichkeiten zu finden, wurden Simulationen für verschiedene Werte unter den oben beschriebenen Bedingungen durchgeführt. Die Abbildung 4.30 zeigt die Ergebnisse.

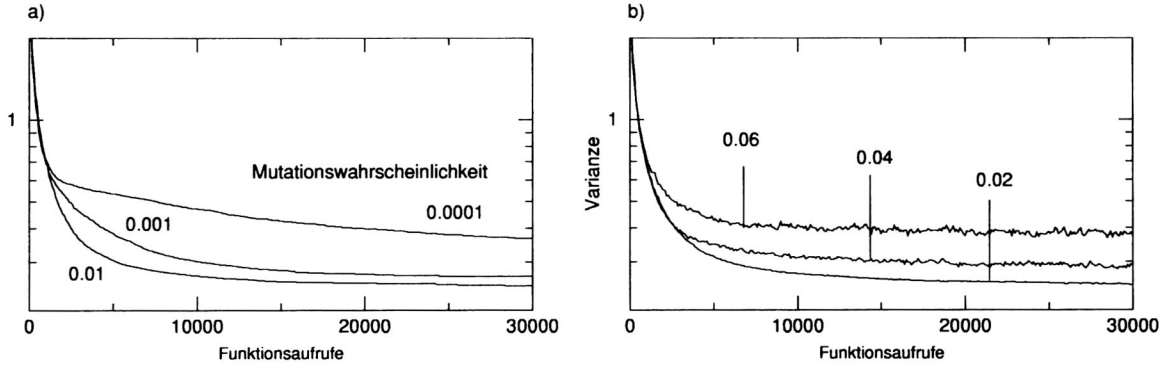


Abbildung 4.30: Vergleich verschiedener Mutationswahrscheinlichkeiten bei genetischen Algorithmen.

Beste mittlere Resultate wurden mit der Mutationswahrscheinlichkeit 0.01 erreicht. Da die Tests mit 100 Individuen durchgeführt wurden, gilt:

$$Mutationswahrscheinlichkeit = \frac{1}{Populationsgröße}$$

Dieses Ergebnis bestätigt den in der Literatur empfohlenen Wert (Goldberg, 1989). In Abbildung 4.30 ist deutlich zu sehen, daß größere Mutationswahrscheinlichkeiten für eine

größere Varianz der Verläufe sorgen. Erstaunlich ist die Tatsache, daß sehr kleine Mutationswahrscheinlichkeiten (0.0001 in Abbildung 4.30) zu sehr schlechter Konvergenz führen. Dies bedeutet, daß der Algorithmus mit der Rekombination allein nur schlecht arbeitet. Die mittleren Verläufe sind in Tabelle 4.3 zusammengefaßt. Der Wert 0.01 hat auch hier das beste Resultat.

Mutationswahrscheinlichkeit	durchschnittliche Varianz
0.0001	0.0311
0.001	0.0048
0.01	0.0034 ←
0.02	0.0042
0.04	0.0036
0.06	0.0041

Tabelle 4.3: Zusammenfassung der mittleren Varianzen der 30 Simulationen für verschiedene Mutationswahrscheinlichkeiten bei genetischen Algorithmen.

4.2.5.4 Rekombinationswahrscheinlichkeit

Neben der Mutationswahrscheinlichkeit kann auch die Wahrscheinlichkeit, mit der zwei selektierte Individuen rekombiniert werden, variiert werden. Wenn die Rekombinationswahrscheinlichkeit auf den Wert 0 gesetzt wird, werden selektierte Individuen nie rekombiniert, und ausschließlich die sehr geringe Mutationsrate sorgt für Variationen. Setzt man den Wert auf 1 werden die selektierten Individuen immer rekombiniert. Die Frage, die sich daher auch hier ergibt, ist die, ob es einen besonders günstigen Wert für die gravimetrische Modelloptimierung gibt. Hierzu werden Simulationen für die Werte zwischen 0.0, 0.3 und 1.0 durchgeführt. Die anderen Strategieparameter werden wie bei den vorangegangenen Untersuchungen gesetzt. Die Abbildung 4.31 zeigt die Resultate der Optimierungen.

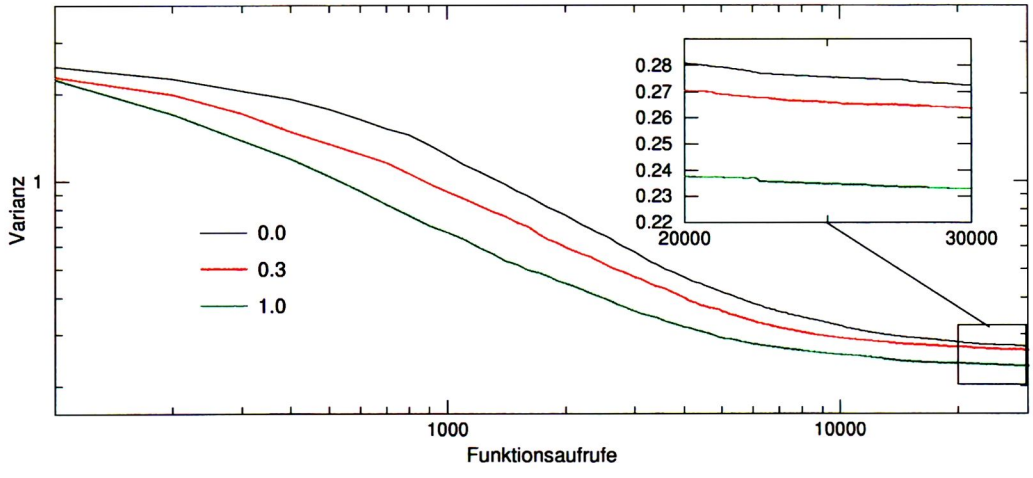


Abbildung 4.31: Vergleich verschiedener Rekombinationswahrscheinlichkeiten bei genetischen Algorithmen.

Die Kurven verlaufen ähnlich, was bedeutet, daß die Rekombinationswahrscheinlichkeit das Verhalten des Algorithmus sehr gering beeinflußt. Dies ist bemerkenswert, da ohne die Re-

kombination vergleichbar gute Ergebnisse erzielt werden. Dies bestätigt noch einmal, daß die Optimierung im wesentlichen auf Mutation und Selektion basiert. Da dieses Ergebnis von außerordentlicher Bedeutung ist, wurde die Programmimplementierung mehrmals sorgfältig überprüft und auf einfache, sehr gut überschaubare Probleme mit wenigen Parametern angewendet. Programmierfehler können daher mit großer Wahrscheinlichkeit ausgeschlossen werden. Die Ergebnisse sind auch auf andere gravimetrische Probleme übertragbar. Die Rekombination hat bei der Optimierung der getesteten Modelle sehr geringen Einfluß auf das Konvergenzverhalten. Die Mutationswahrscheinlichkeit zu verringern, um der Rekombination mehr Gewicht zu geben, kann nicht als sinnvoll erachtet werden, da die vorangegangenen Untersuchungen gezeigt haben, daß eine Verkleinerung dieses Wertes eine Verlangsamung des Konvergenzverhaltens zur Folge hat.

4.2.5.5 Rekombinationsschemata

In der Natur gibt es Rekombinationen von Chromosomen mit ein und zwei Kreuzungspunkten (*one- and two-point-crossover*). Auch in genetischen Algorithmen kann dieses Phänomen simuliert werden. Mathematisch gesehen erreicht man dies mit *two-point-crossover*, so daß auch Hyperebenen konstruiert werden können, die sich mit nur einem crossover-point nicht erzeugen lassen (Schöneburg et al., 1994). Auch beim *two-point-crossover* ist nicht garantiert, daß alle Punkte des Raumes konstruiert werden können. Um sicherzustellen, daß jeder Ort des Raumes prinzipiell erreichbar ist, kann ein Schema gewählt werden, nach dem die Parameter, die das rekombinierte Individuum bilden, per Zufall vom einen oder anderen selektierten Individuum genommen werden. Von dem hier vorliegenden Optimierungsproblem kann a priori nicht gesagt werden, ob mehr als ein Kreuzungspunkt erforderlich ist. Deshalb werden im folgenden Simulationen für Ein-, Zwei- und Mehrpunkt-Rekombinationen durchgeführt, um empirisch zu untersuchen, inwiefern ein Einfluß auf das Konvergenzverhalten für dieses gravimetrische Problem besteht. Basierend auf den vorangegangenen Untersuchungen wird die Populationsgröße in den folgenden Berechnungen wieder auf 100 festgelegt. Die Abbildung 4.32 zeigt die Ergebnisse der Simulationen.

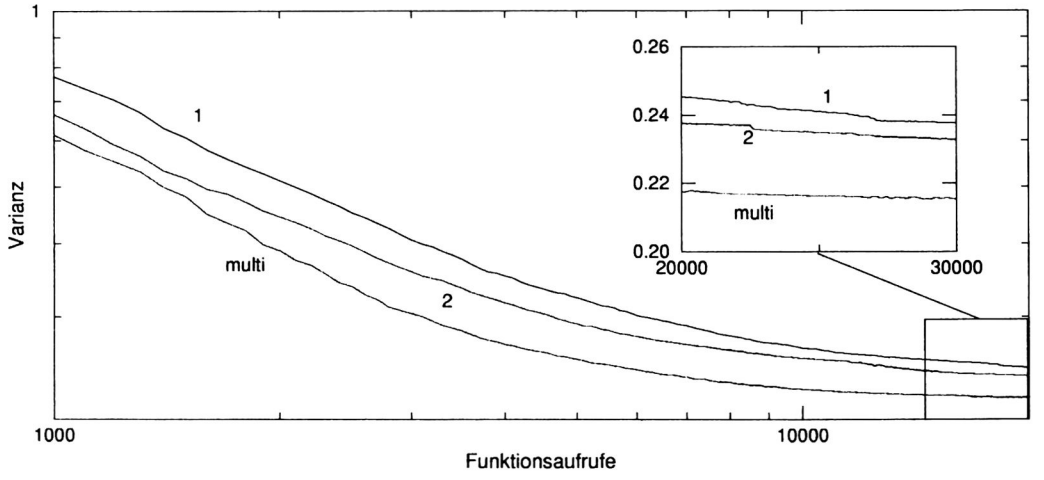


Abbildung 4.32: Vergleich verschiedener Rekombinationsschemata bei genetischen Algorithmen.

Die Mutationswahrscheinlichkeit, die bei diesen Rechnungen verwendet wurde, beträgt 0.01, und die Bitstringlänge wurde auf 6 Bits pro Parameter gesetzt. Die Rekombinationspunkte wurden gleichverteilt über die Länge des Bitstrings gewählt. Dabei dürfen sie nicht identisch sein und ihr Abstand muß mindestens zwei Bits betragen. Diese Einschränkungen sollten sehr ähnliche bzw. gleiche Individuen von vornherein vermeiden. Beste Ergebnisse wurden mit dem Mehrpunkt-Schema erreicht.

4.2.5.6 Basis des Codierungssystems

Ein weiterer Strategieparameter, der bei genetischen Algorithmen eingestellt werden kann, ist die Basis des zugrunde gelegten Codierungssystems. Obgleich sehr viele Implementierungen die binäre Repräsentation benutzen, soll hier auch der Einfluß dieses Parameters untersucht werden. Nach Goldberg (pers. Mitt.) gibt es keine Anhaltspunkte dafür, daß andere Codierungsformen die Konvergenzeigenschaften der Algorithmen verbessern. Die sog. *real-coded genetischen Algorithmen* verwenden das dezimale System und sind den Evolutionsstrategien sehr ähnlich.

Wie bereits beschrieben, werden Mutationen im binären Fall durch Negation eines Bits realisiert. Wird eine andere Basis als 2 verwendet, ist dieses Verfahren nicht mehr anwendbar. Für die folgenden Simulationen werden daher die Mutationen durch Zufallszahlen, welche einer Gleichverteilung genügen, realisiert. [1] Die Umrechnung des jeweils verwendeten Zahlensystems in das Dezimalsystem und umgekehrt wird unter der Verwendung der jeweiligen Basis so durchführt, wie in Abschnitt 3.8 beschrieben. Weiterführende Informationen zur Transformation von Zahlensystemen findet man beispielsweise bei Niemeyer (1989).

Die anderen Strategieparameter werden nach den Erfahrungen der vorangegangenen Tests gesetzt. Um die Vergleichbarkeit zu den meisten bisher durchgeführten Simulationen zu gewährleisten, wird das Rekombinationsschema auf *two-point-crossover* festgelegt. Die Abbildung 4.33 zeigt die durchschnittlich erreichten Werte der Simulationen nach 30,000 Funktionsaufrufen.

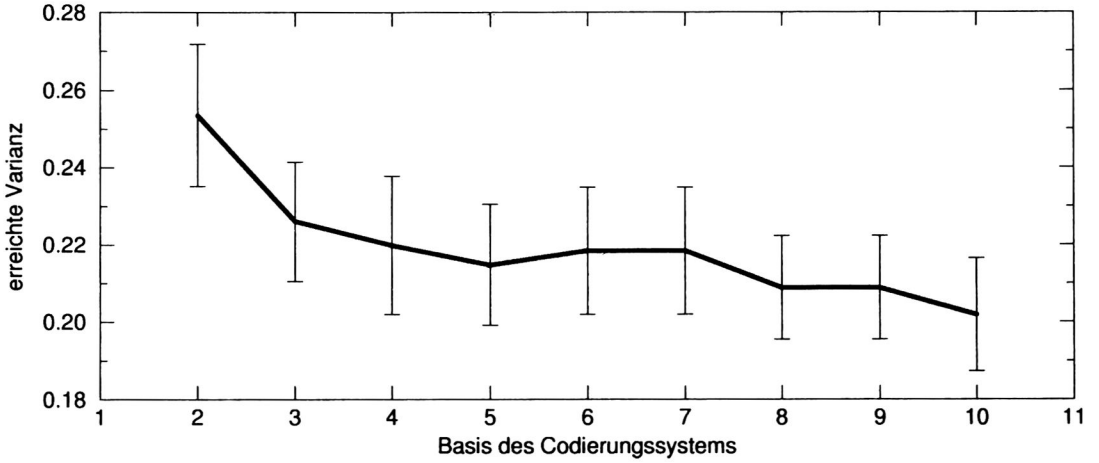


Abbildung 4.33: Vergleich verschiedener Codierungsformen bei genetischen Algorithmen.

[1] Ist die verwendete Basis z.B. 6, könnte zur Anschauung ein Spielwürfel verwendet werden.

Trotz der großen Fehler ist ein Trend zu erkennen, nach dem die dezimale Codierung bessere Ergebnisse liefert. Einzelne Simulationen wurden bei diesem Experiment 100fach wiederholt. Die Fehlerintervalle konnten dadurch nicht verkleinert werden. Die schlechtesten durchschnittlichen Qualitätswerte wurden mit der binären Codierungsform erreicht.

4.2.5.7 Selektionsschema

Bei den bisher durchgeführten Tests wurde die Population sortiert und die besten 50 % der Individuen zum Bau der neuen Generation benutzt. Dabei wurde die Selektionswahrscheinlichkeit vom besten zum schlechtesten zwischen 1 und 0 linear gewichtet. Zusätzlich wurde der jeweils Beste prinzipiell in die nächste Generation übernommen. Im folgenden werden verschiedene Selektionsschemata untersucht. Variiert wird dabei der Anteil der benutzten Individuen. Außerdem wird untersucht, welchen Einfluß die Übernahme des Besten hat.

Die Tabelle 4.4 zeigt die Ergebnisse der Simulationen. Die Optimierungen wurden bei allen Experimenten 100 mal wiederholt. Die Ergebnisse besitzen daher statistisch gesehen eine recht gute Aussagekraft.

Anteil der besten Individuen [%]	durchschnittliche Qualität	Standardabweichung
100	0.249344	0.0203905
50	0.243857	0.0177530
33	0.241983	0.0217880
25	0.243223	0.0185397
20	0.243659	0.0182278
ohne Besten	0.250954	0.0209921

Tabelle 4.4: Zusammenfassung der Ergebnisse der Simulationen für verschiedene Selektionsschemata bei genetischen Algorithmen.

Das verwendete Selektionsschema hat somit sehr wenig bzw. keinen Einfluß auf die durchschnittlich erreichte Qualität. Wird das beste Individuum nicht zwingend in die nächste Generation übernommen, verschlechtern sich die Resultate marginal (letzte Zeile der Tabelle 4.4).

4.2.5.8 Zusammenfassung der Ergebnisse für genetische Algorithmen

Genetische Algorithmen liefern gute durchschnittliche Resultate in Hinblick auf die erzeugten Modelle und ihre Konvergenzgeschwindigkeit. Die untersuchten Strategieparameter haben wenig Einfluß auf das Verhalten des Verfahrens.

Das Verfahren kann sehr schnell zu bemerkenswert guten Resultaten führen, besitzt aber bei der „Feineinstellung“ der Parameter ein sehr schlechtes Verhalten. Dies ist allerdings

wenig verwunderlich, da die Methode keinen Adaptionsmechanismus hat, der dafür sorgt, daß Variationen den lokalen Gegebenheiten der Qualitätsfunktion angepaßt werden.

Es konnte gezeigt werden, daß die Rekombinationswahrscheinlichkeit, die i. allg. als die „treibende Kraft" bei genetischen Algorithmen bezeichnet wird, bei den durchgeführten Tests praktisch keine Rolle spielt. Die Mutationswahrscheinlichkeit hat hingegen starken Einfluß auf das Konvergenzverhalten.

Für den praktischen Einsatz des Verfahrens werden folgende Strategieparameter vorgeschlagen:

- Größe der Population: 100 bis 200 Individuen,
- Länge der Bitstrings: vier bis acht,
- Mutationswahrscheinlichkeit: reziproke Populationsgröße,
- Rekombinationswahrscheinlichkeit 1.0,
- Rekombinationsschema: Zufallswahl,
- Codierungssystem: dezimal.

4.2.6 Simulated-Annealing

Im folgenden wird wiederum das in Abschnitt 4.9 beschriebene Testmodell optimiert. Auch der dort beschriebene Strafterm wird wie bei den Evolutionsstrategien (siehe Abschnitt 3.9) bei den folgenden Simulationen benutzt. Der Einfluß folgender Parameter wird untersucht:

- Temperatur,
- Abkühlungsplan,
- Schrittweite.

4.2.6.1 Temperatur und Abkühlungsplan

Wie bereits beschrieben, kann beim Simulated-Annealing mit dem Strategieparameter „Temperatur" die Wahrscheinlichkeit gesteuert werden, mit der ein Modell akzeptiert wird, welches nach einer Mutation schlechtere Qualität hat als sein Vorgänger. Dies ermöglicht generell das Verlassen von Nebenoptima. Setzt man die Temperatur auf Null, so ist die Wahrscheinlichkeit, Rückschritte zu akzeptieren, ebenfalls gleich Null. Der Algorithmus verhält sich dann wie der Hill-Climbing-Algorithmus. Zunächst soll daher mit einer festen Schrittweite untersucht werden, welche Ergebnisse der Algorithmus erzeugt. Eine Temperatur größer Null sollte eine Verbesserung der Konvergenzeigenschaften zur Folge haben. Es sollte dann möglich sein, eventuell existierende lokale Optima zu verlassen.

Die Abbildung 4.34 zeigt 20 zufällig von insgesamt 100 ausgewählte Verläufe der Optimierung des Testmodells mit dem Simulated-Annealing-Algorithmus. Die Schrittweite wurde dabei auf der Basis der vorangegangenen Untersuchungen mit der Evolutionsstrategie auf 200 Meter festgesetzt und konstant gehalten.

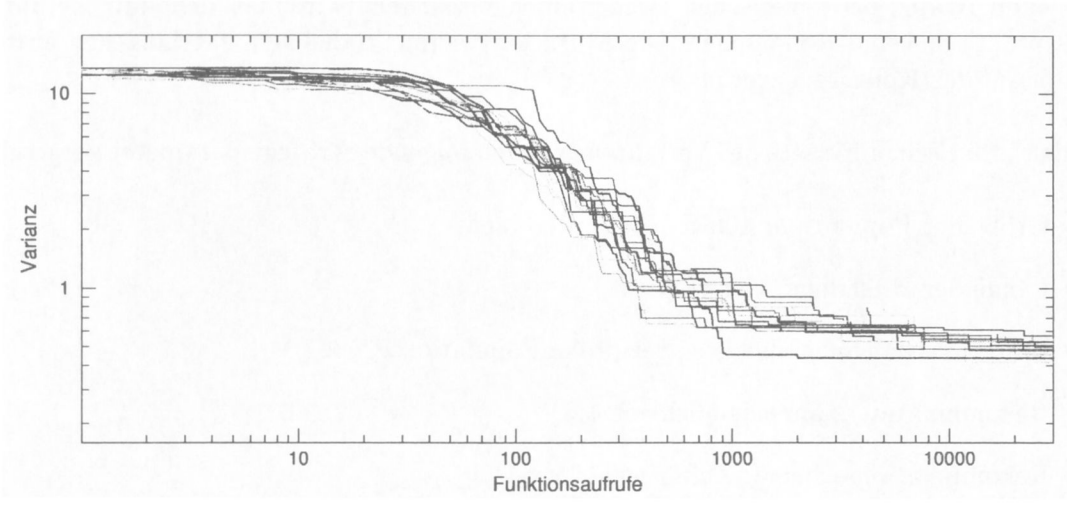


Abbildung 4.34: Gezeigt sind 20 zufällig aus 100 ausgewählte Hill-Climbing-Optimierungen. Das Verhalten wurde simuliert, indem die Temperatur beim Simulated-Annealing-Verfahren auf Null gesetzt wurde.

Der Durchschnitt aus den 100 durchgeführten Optimierungsläufen ist in Abbildung 4.35 gezeigt. Die beste erreichte Varianz liegt bei ca. 0.5 und ist damit deutlich schlechter als die bisher mit der Evolutionsstrategie und den genetischen Algorithmen erreichten Werte (vergl. Abschnitte 3.9 und 3.8).

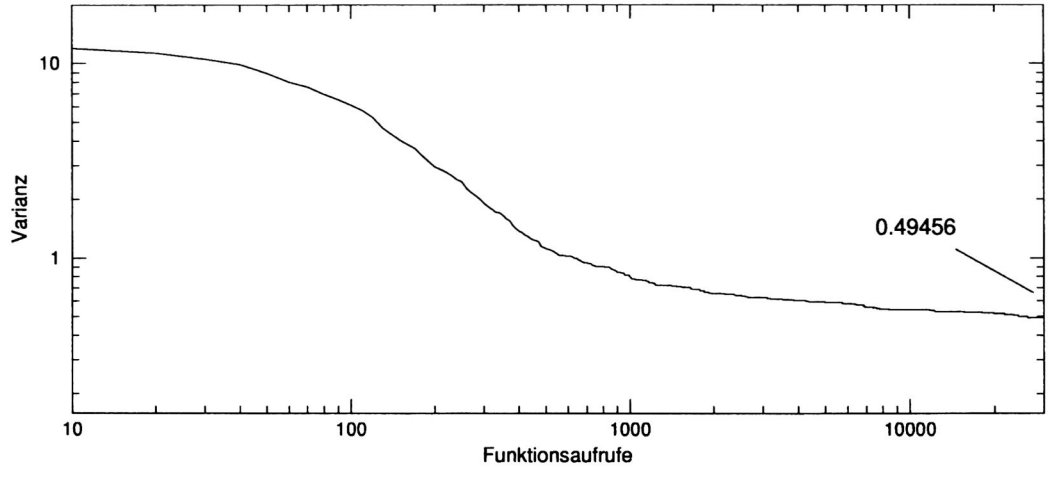


Abbildung 4.35: Durchschnitt der 100 Simulated-Annealing Optimierungen mit einer Temperatur von Null.

Im folgenden soll nun die Temperatur größer Null gesetzt werden und ein Abkühlungsplan soll die Temperatur langsam gegen Null abkühlen. Zur Verringerung der Temperatur wurde daher die Temperatur bei erfolgreichen Schritten mit dem Faktor 0.99 bzw. 0.999 multipiziert. Daraus ergibt sich ein Verlauf der Temperatur, der langsam gegen Null konvergiert,

die Null aber nie erreicht. Die Abbildung 4.36 zeigt den Verlauf der Temperatur und der Wahrscheinlichkeit für einen repräsentativen Optimierungsverlauf.

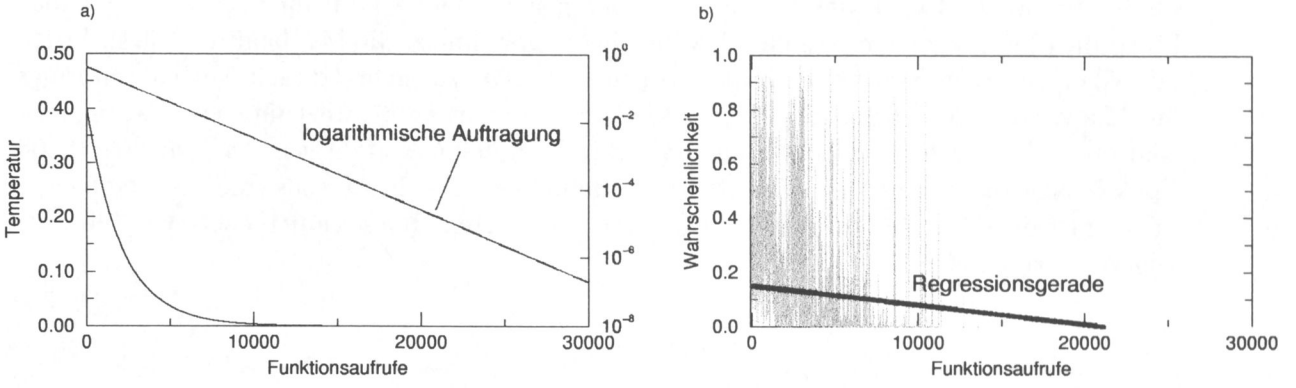


Abbildung 4.36: Die Temperatur und die Wahrscheinlichkeit für die Akzeptanz für Rückschritte aufgetragen über den Funktionsaufrufen.

Die Abbildung 4.37 zeigt die durchschnittlichen Ergebnisse der Simulationen mit den gezeigten Strategieparametereinstellungen.

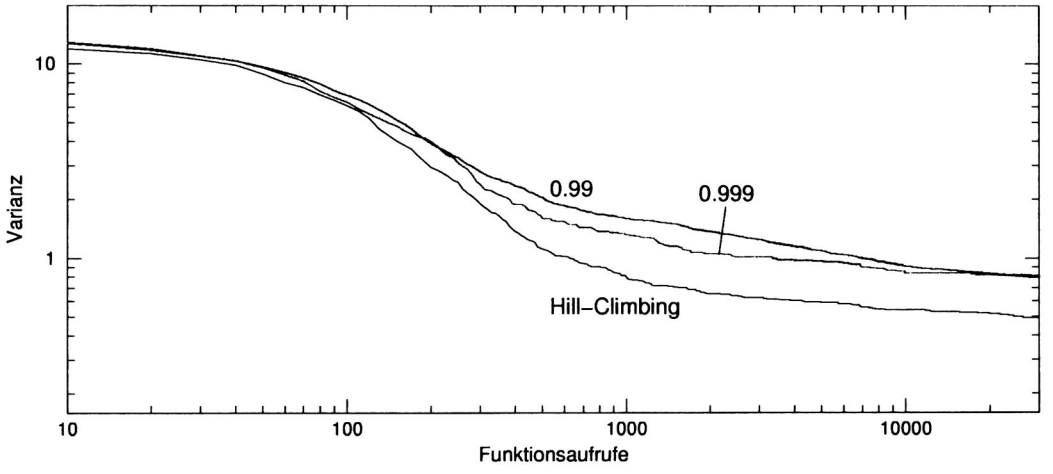


Abbildung 4.37: Der Durchschnitt von 100 Simulated-Annealing Optimierungen mit einer Temperatur größer Null. Die Ergebnisse sind deutlich schlechter als die des Hill-Climbing und damit unbrauchbar.

4.2.6.2 Schrittweite

Die Veränderung der Schrittweite, die bei diesem Experiment auf 200 Meter festgelegt war, brachte keine signifikante Veränderung der Konvergenzeigenschaften.

Basierend auf diesen Versuchen kann erwartet werden, daß bei weiter erhöhter Temperatur die erreichten Varianzen noch schlechter werden, da dann die Wahrscheinlichkeit, Rückschritte zu akzeptieren, steigt. Dies gilt gleichfalls für die Erniedrigung des Faktors zur Verringerung der Temperatur.

4.2.7 Great-Deluge-Algorithmus und Threshold-Accepting

Die Ergebnisse, die mit diesen beiden Algorithmen erreicht wurden, bleiben ebenfalls hinter denen der Evolutionsstrategie und der genetischen Algorithmen zurück. Offenbar führt auch hier, wie beim Simulated-Annealing-Algorithmus, die Möglichkeit, Rückschritte mit einer gewissen Wahrscheinlichkeit zu akzeptieren, zu einer starken Verschlechterung der Konvergenz. Die Resultate für ein breites Spektrum an strategieinternen Parametern sind ebenfalls deutlich schlechter als die des Hill-Climbing-Verfahrens. Auch nach 100,000 Funktionsaufrufen wurden keine Modelle gefunden, die besser als die der einfachen Suchstrategie sind. Die Abbildung 4.38 zeigt die besten durchschnittlichen Verläufe aus 100 Wiederholungen.

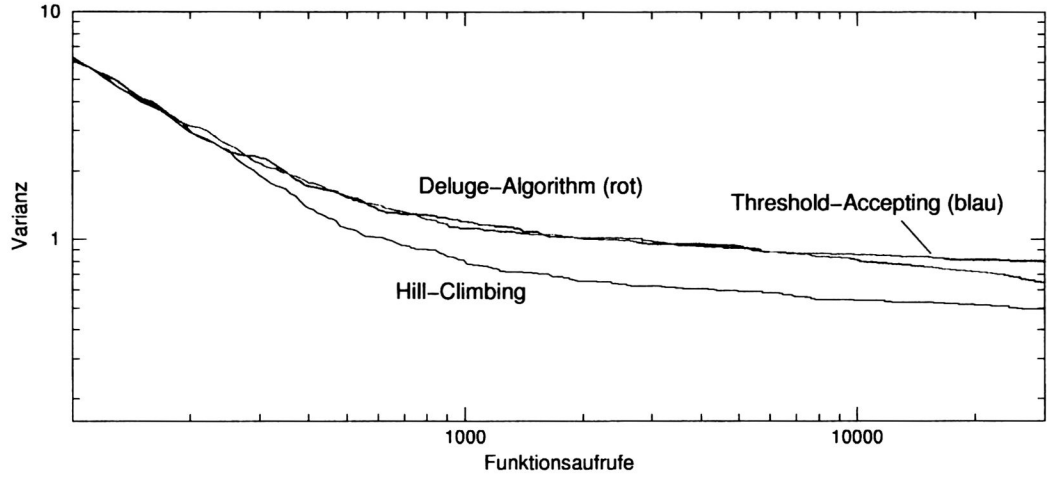


Abbildung 4.38: Der Deluge-Algorithm und das Threshold-Accepting haben schlechtere Konvergenzeigenschaften als die einfache Suchstrategie des Hill-Climbing.

Die von den Algorithmen nach 30,000 Funktionsaufrufen gefundenen Lösungen zeigten ebenfalls keinen „Neugewinn an Information", wie zum Beispiel einen Trend zu einer prinzipiell anderen Struktur des Salzstocks, gegenüber den anderen Verfahren.

4.2.8 Diskussion der Ergebnisse

Die Ergebnisse der Verfahren Simulated-Annealing, Deluge-Algorithm und Threshold-Accepting zeigten insgesamt schlechtere Konvergenzeigenschaften als die Evolutionsstrategie und der genetischen Algorithmen. Die Ursache dafür liegt aus meiner Sicht im wesentlichen darin, daß in den Algorithmen, wie sie hier implementiert wurden, keine Heuristik zum Anpassen der Variation der Parameter vorgesehen ist. Würde man ein einfaches Adaptionsverfahren, wie die von Rechenberg vorgeschlagene „$\frac{1}{5}$ Erfolgsregel" (Rechenberg, 1973) einbauen, so ist sicherlich eine Verbesserung der Konvergenz zu erwarten. Dann aber stellt sich die Frage, warum nicht direkt mit einer (1,x)-Evolutionsstrategie gearbeitet werden soll, die ja die Möglichkeit, Rückschritte zu akzeptieren, ebenfalls inhärent in sich trägt.

Der zweite Punkt, der hier diskutiert werden muß, ist das Optimierungsproblem an sich. Die mit diesem Testmodell und den Verfahren Simulated-Annealing, Deluge-Algorithm und Threshold-Accepting erzielten Ergebnisse stehen im Widerspruch zu den Resultaten eines Vergleichs der Verfahren Deluge-Algorithm und Threshold-Accepting mit dem des Simulated-Annealing (Dueck, 1993; Dueck und Scheuer, 1990) wo gezeigt wird, daß der Deluge-Algorithm und das Threshold-Accepting besser konvergieren als Simulated-Annealing. Allerdings ist darin kein unlösbarer Konflikt zu sehen, sondern dies ist lediglich ein Beispiel dafür, daß sich verschiedene Verfahren für verschiedene Probleme unterschiedlich gut eignen. Offensichtlich eignen sich *diese* Verfahren besser für kombinatorische Optimierungsaufgaben (wie z.B. der des Handlungsreisenden). Die „Natur" der gravimetrischen Optimierung ist offenbar weniger kombinatorisch als kontinuierlich. Bemerkenswert ist aber, daß die genetischen Algorithmen trotz des Fehlens eines Adaptionsmechanismus der lokalen Eigenschaften der Qualitätsfunktion wesentlich bessere Resultate als das Hill-Climbing-Verfahren liefern.

4.3 Optimierung in 3D

Es ist heute schon abzusehen, daß die Entwicklung der geowissenschaftlichen Modellierungen in Zukunft ausschließlich dreidimensional sein wird. Deshalb wird geprüft, in welchem Rahmen sich automatische Optimierungstechniken auf 3D-Modelle der Gravimetrie anwenden lassen. Im folgenden wird die Evolutionsstrategie, die sich bei zweidimensionalen Modelloptimierungen als leistungsfähiges Verfahren bewährt hat, auf eine dreidimensionale Struktur angewendet werden.

Das bereits beschriebene Salzstockmodell, welches hier optimiert werden soll, besteht aus ca. 36,000 Dreiecken und 16,000 Modellpunkten. Die Geometrie basiert auf seismischen Modellierungen, so daß von den lateralen seismischen Horizonten angenommen werden kann, daß sie im Rahmen der Auflösung der Seismik „richtig" sind. Deshalb wird wiederum nur die Form des Salzstocks optimiert, so daß sich die Anzahl der Geometrieparameter deutlich verringert. Die Optimierung aller Parameter wäre zum heutigen Zeitpunkt praktisch nicht durchführbar, da dies zuviel Rechenzeit für die Auswertung der Qualitätsfunktion in Anspruch nehmen würde. Desweiteren haben Voruntersuchungen gezeigt, daß es bei der Mutation aller Geometrieparameter des Horizontes nach wenigen Generationen zu Überschneidungen einzelner Dreiecke kommt und die Bedingungen, die der Formel zur Berechnung der Schwerewirkung von Polyedern (siehe Abschnitt 2.2) zugrunde liegen, verletzt werden. Randbedingungen, welche die Freiheit der Parameter von vornherein einschränken und daher Überkreuzungen a priori verbieten, lassen sich bei derartig komplexen dreidimensionalen Dreiecksgittern schlecht formulieren, da sie interaktiv vorgegeben werden müssen.

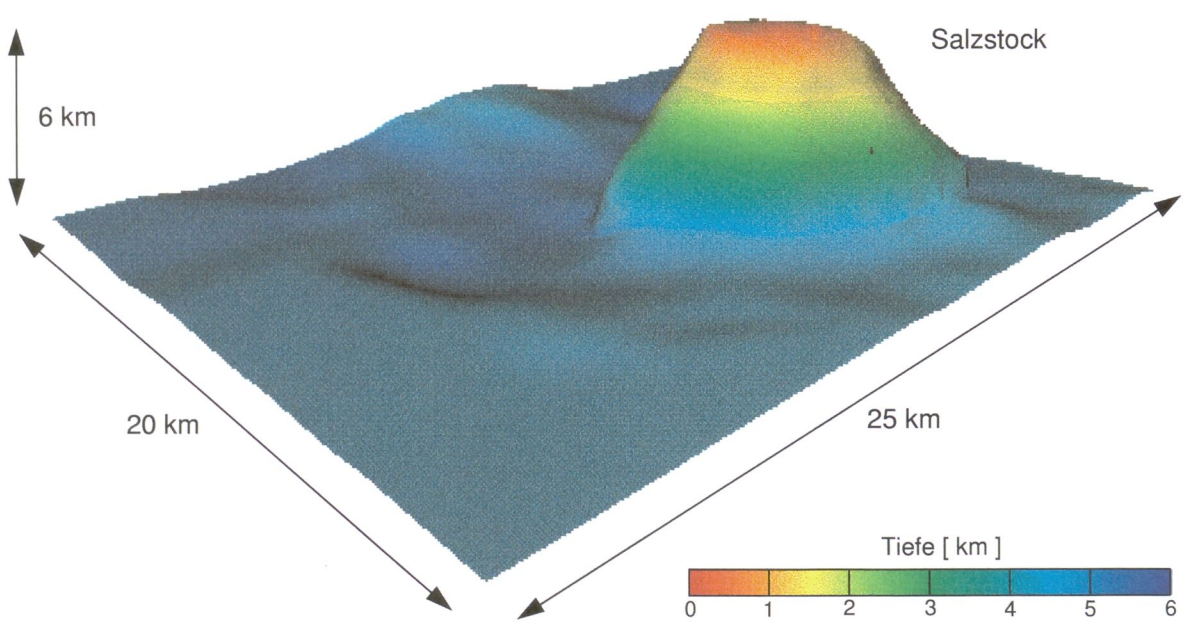
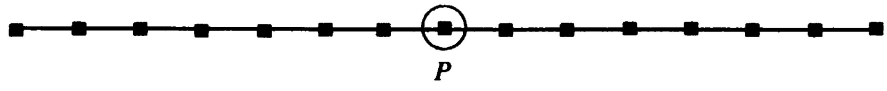


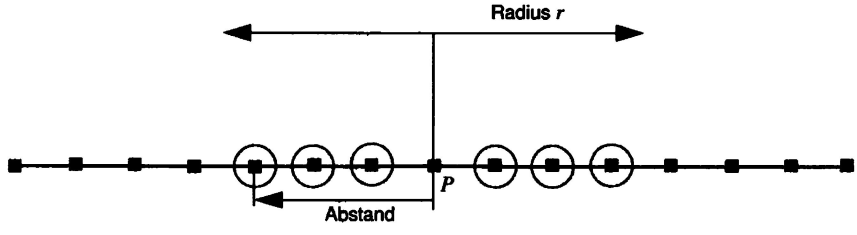
Abbildung 4.39: Der Zechsteinhorizont aus dem zu optimierenden seismischen Modell. Alle anderen Horizonte wurden ausgeblendet.

Dies führte zur Entwicklung eines „generalisierten“ Mutationsverfahrens. So werden nicht alle vom Benutzer gewählten Parameter einzeln mutiert, sondern es wird nur ein Teil der gewählten Parameter betrachtet. Diese werden zufällig ausgewählt und variiert, die verbleibenden Parameter werden dann mit einer Sinusfunktion interpoliert. Der Algorithmus kann anhand eines eindimensionalen Beispiels wie folgt beschrieben werden:

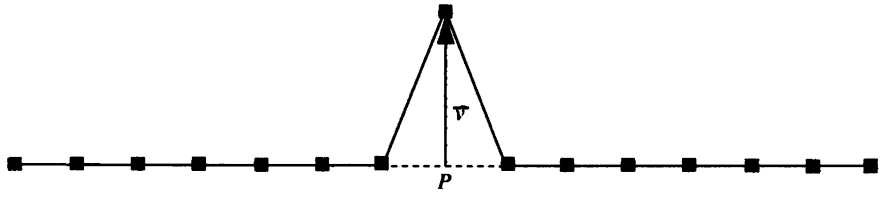
Schritt 1 Wähle zufällig einen Punkt P von allen zu optimierenden Punkten aus,



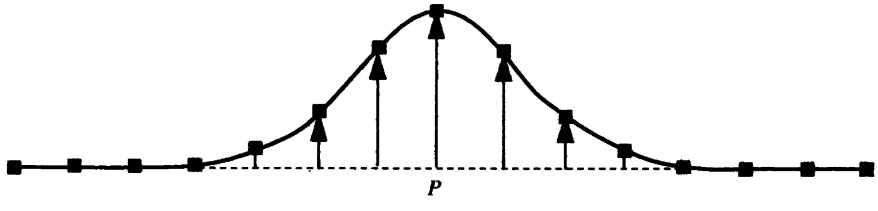
Schritt 2 finde alle Punkte in einem vorgegebenen Bereich mit dem Radius r und berechne den Abstand jedes Punktes innerhalb des Gebietes zum ausgewählten Punkt P,



Schritt 3 addiere zum Punkt P einen Verschiebungsvektor $\vec{v}$, der orthogonal auf der in P konstruierten Tangente (Tangentialebene) steht,



Schritt 4 interpoliere, gewichtet mit dem Abstand von P und der Sinusfunktion, die Verschiebung der Punkte, die sich im Gebiet befinden.



Nach der Variation des ersten Punktes P entsteht bei linearer Interpolation auf einer Ebene ein Kegel. Eine Interpolation mit der Sinusfunktion ergibt glatte Übergänge, wenn der beschriebene Algorithmus mehrmals hintereinander angewendet wird. Nach der Interpolation ergibt sich so ein rotationssymmetrischer „Hügel". Durch geeignete Wahl von r kann der Einflußradius um P bestimmt werden, so daß sowohl langwellige als auch kurzwellige Strukturen erzeugt werden. Die Abbildung 4.40 zeigt die einmalige Anwendung des Algorithmus im dreidimensionalen Fall mit zwei verschiedenen Einflußradien.

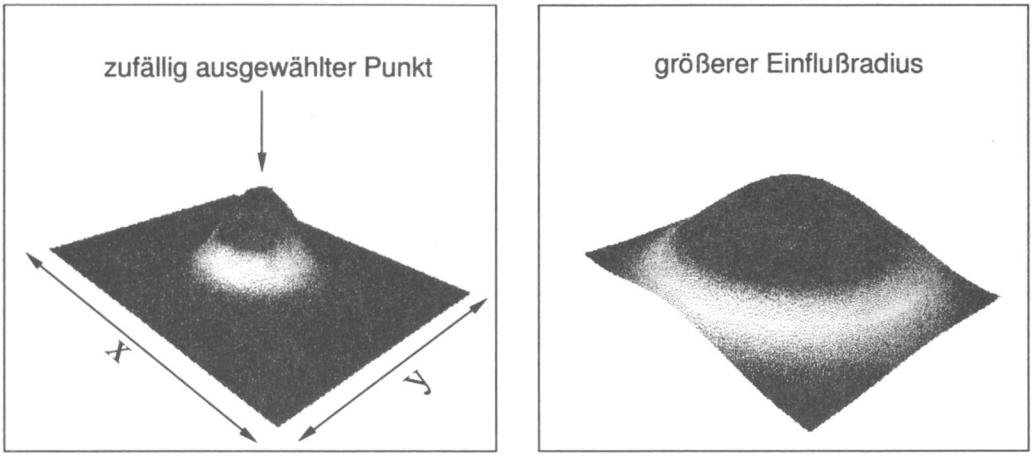


Abbildung 4.40: Unterschiedliche Einflußradien ermöglichen die Modellierung von Strukturen unterschiedlicher Wellenlänge. Durch Überlagerung mehrerer Mutationen können beliebige Strukturen erzeugt werden. Interpoliert wurde hier mit der Sinusfunktion.

Durch die Überlagerung von Sinus- bzw. Cosinusfunktionen verschiedener Wellenlängen und/oder Amplituden können beliebige bijektive Funktionen erzeugt werden (Smirnow, 1965; Kreyszig, 1993). Der Vorteil des oben beschriebenen Verfahrens besteht darin, daß bei mehrfacher Anwendung des Algorithmus mit verändertem Radius r beliebige Oberflächen erzeugt werden können, die nicht der Bedingung der Bijektivität genügen müssen. Daher können beispielsweise „Überhänge", ausgehend von einer Ebene, prinzipiell modelliert werden. Die Triangulierung des Dreiecksgitters muß dann häufiger erneuert werden, da die Anzahl der Dreiecke, welche die Fläche der Ebene aufspannen, nicht ausreicht, um die nun wesentlich vergrößerte Fläche zu beschreiben. Dies kann hier allerdings aufgrund der Komplexität des Problems nicht berücksichtigt werden. Für die gravimetrische Optimierung des Testmodells (siehe unten) reichte die Auflösung des Startmodells aus, so daß eine erneute Triangulierung nicht erforderlich war.

4.3.1 Mutationsrichtung

Variationen eines Punktes innerhalb eines ebenen Horizontes haben keine Änderung der Geometrie des Objektes zur Folge, daher bleibt auch die gravimetrische Wirkung des Objektes die gleiche. Eine Verschiebungsrichtung, die zur Veränderung der Geometrie des Objektes führt, muß also senkrecht zur Tangentialebene, die an einem Punkt konstruiert

werden kann, verlaufen. Diese Richtung kann z.B. aus den Flächennormalen der umliegenden Dreiecke berechnet werden (Abbildung 4.41).

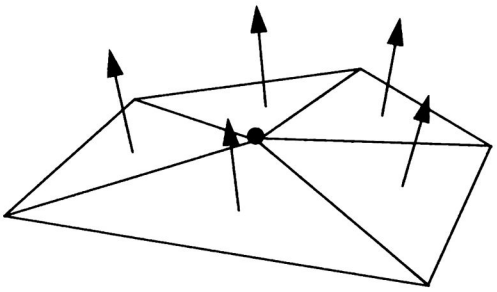


Abbildung 4.41: Die Verschiebungsrichtung für einen gewählten Punkt kann mit Hilfe des Vektordurchschnitts der an den Punkt grenzenden Dreiecke berechnet werden. Im einfachsten Fall werden nur die direkt angrenzenden Dreiecke berücksichtigt.

Allerdings ist der Vektordurchschnitt der Normalenvektoren nicht immer senkrecht zur Tangentialebene. Befindet sich ein Punkt beispielsweise auf einer Kante und es grenzen mehr Dreiecke links als rechts von der Kante an diesen Punkt an, hat der Vektordurchschnitt nicht die gewünschte Richtung (siehe Abbildung 4.42).

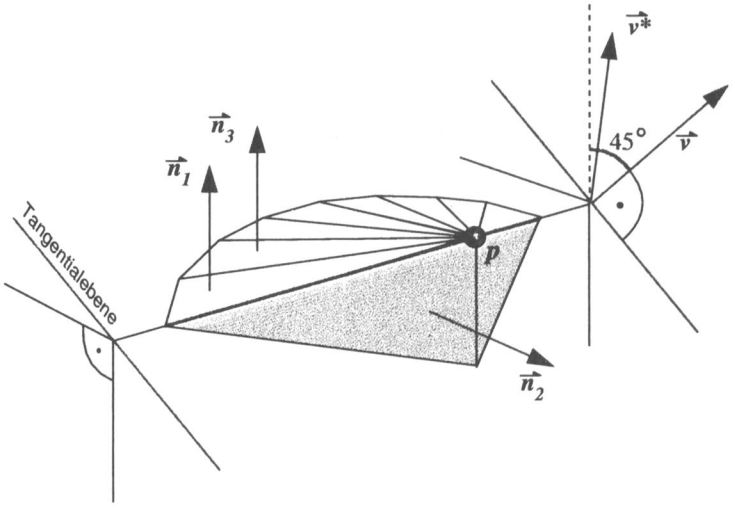


Abbildung 4.42: Die Richtung des Vektordurchschnitts ($\vec{v}^*$) ist abhängig von der Anzahl der rechts und links von der Kante angrenzenden Dreiecke. Die gesuchte Richtung $\vec{v}$ wird aus den *Dreiecksübergängen* konstruiert (siehe Text).

Daher muß eine Richtung errechnet werden, die sich aus den *Unterschieden* der Richtungen der Flächennormalen ergibt. Im Beispiel in Abbildung 4.42 hätten so nur die Übergänge zwischen gelben und blauen Dreiecken Einfluß. Die Gewichtung, wie stark ein Übergang in den Gesamtdurchschnitt eingeht, wird mit dem Skalarprodukt der jeweiligen Flächennormalen berechnet. Im Beispiel von Abbildung 4.42 ist $< n_1, n_2 >= 1$. Damit geht die Kante mit der Gewichtung 1 in den Gesamtdurchschnitt ein. Desweiteren ist z.B. $< n_1, n_3 >= 0$, daher geht dieser Übergang nicht in den Gesamtdurchschnitt ein.

Um den gewünschten Verschiebungsvektor $\vec{v}$, zu berechnen, wird folgender Algorithmus ausgeführt:

Schritt 1	Finde die an den zu verschiebenden Punkt angrenzenden Dreiecke (Anzahl k),
Schritt 2	berechne die Flächennormalen dieser Dreiecke,
Schritt 3	addiere für alle Dreiecksübergänge $j = k - 1$ die Normalen $n_j = n_i + n_{i+1}$,
Schritt 4	Multipliziere n_j mit $\sin(< n_i, n_{i+1} >)$,
Schritt 5	Berechne aus den so erzeugten $k - 1$ Vektoren den Vektordurchschnitt.

Der so errechnete Vektor $\vec{v}$ steht senkrecht auf der Tangentialebene und ist unabhängig von der Anzahl der angrenzenden Dreiecke.

Zur Erhöhung der Variabilität wird der Vektor $\vec{v}$, wie in Abbildung 4.43 gezeigt, in einem vorher festgelegten Gebiet mit dem Radius R zufällig variiert. Überhänge können so besser erzeugt werden.

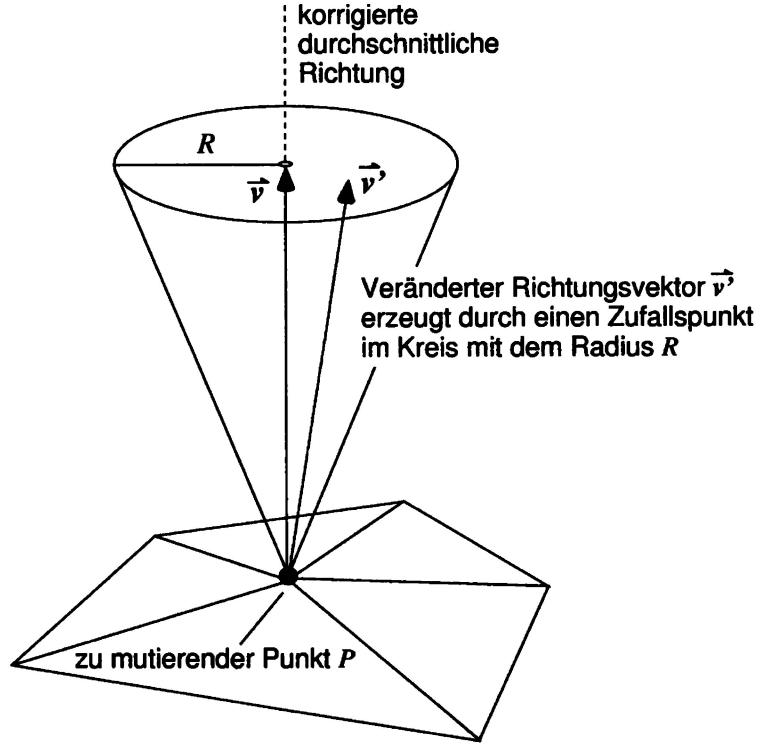


Abbildung 4.43: Zur Erhöhung der Variabilität wird eine Störung des Vektors $\vec{v}$ eingeführt. Der so konstruierte Vektor $\vec{v}'$ ist der eigentliche Verschiebungsvektor.

4.3.2 Ergebnisse der Optimierung

Bei der Optimierung des Salzstocks wurde eine (1,10)-ES mit dem beschriebenen Mutationsverfahren angewendet. Zur Optimierung wurden alle Modellpunkte des Salzstocks in einem vorgegebenen Fenster gewählt. Von diesen insgesamt 3600 Punkten wurden jeweils 30 % mutiert. Die anderen Parameter wurden interpoliert. Der Einflußradius r wurde zu Beginn auf drei Kilometer festgelegt und während der Optimierung als Strategieparameter behandelt. Die Untergrenze wurde auf einen Kilometer festgelegt. Die Abbildung 4.44 zeigt das Modell nach insgesamt 2000 Generationen, also 20,000 Funktionsaufrufen.

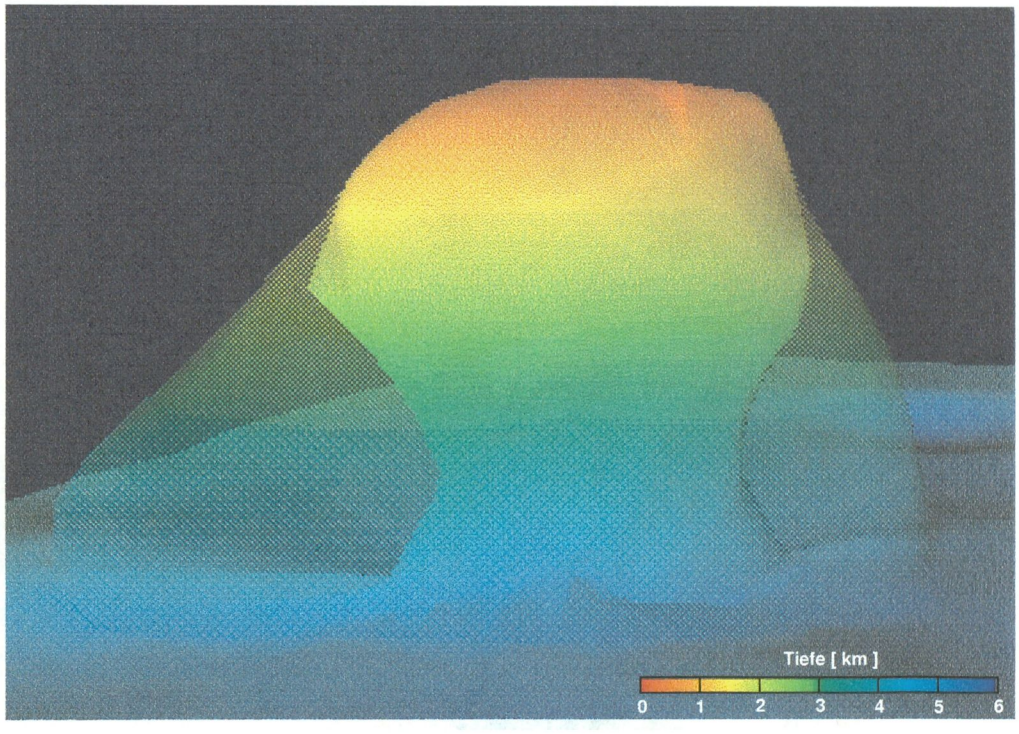


Abbildung 4.44: Das Ausgangsmodell des Salzstocks (transparent) und das optimierte Modell sind gleichzeitig dargestellt. Es ist deutlich eine Verjüngung des Salzkörpers zu erkennen.

Die Optimierung wurde hier abgebrochen, da sich trotz der interpolierten Mutationen der Geometrieparameter Überschneidungen einiger Dreiecke ergaben. Das Modell ist so nicht mehr gültig im Sinne der im Abschnitt 2.2 beschriebenen Gleichung. Dies war bei der großen Anzahl der Dreiecke allerdings zu erwarten und kann bei der Anwendung des vorgestellten Konzeptes nur durch einen Überschneidungstest vermieden werden. Wie bei zweidimensionalen Optimierungen kann nicht ausgeschlossen werden, daß der Algorithmus in bestimmten Stadien der Optimierung nur noch mit der Suche gültiger Modelle beschäftigt ist und die eigentliche Modelloptimierung stagniert. Im schlimmsten Fall liegen Geometriepunkte so dicht beieinander, daß kein Modell ohne Überschneidung gefunden wird. Hier wurde das Kriterium zum Abbruch auf 1,000 Modellgenerierungen festgelegt. Mutationen, die zu Überschneidungen führen, a priori zu verbieten ist bei einer großen Anzahl von Geometrieparametern äußerst schwierig, da derartige Randbedingungen, wie sie bei den beschriebenen 2D-Optimierungen Verwendung fanden, manuell vorgegeben werden müssen. Die erreichte gravimetrische Varianz ist mit denen der zweidimensionalen Modellrechnungen nicht vergleichbar. Die bis zum Abbruch der Optimierung erreichten Ergebnisse können nicht als

optimal angesehen werden. Es kann gesagt werden, daß es sich bei dem gefundenen Modell sehr wahrscheinlich nicht um ein Optimum handelt. Nur der generelle Trend der Verkleinerung des Salzkörpers kann reproduziert werden. Allerdings muß beachtet werden, daß das untersuchte Modell aufgrund der Geometrie des Salzkörpers nur dreidimensional berechnet werden darf und die gefundenen zweidimensionalen Strukturen nur bedingt richtig sind. Die Abbildung 4.45 zeigt den Verlauf der Qualität der Optimierung.

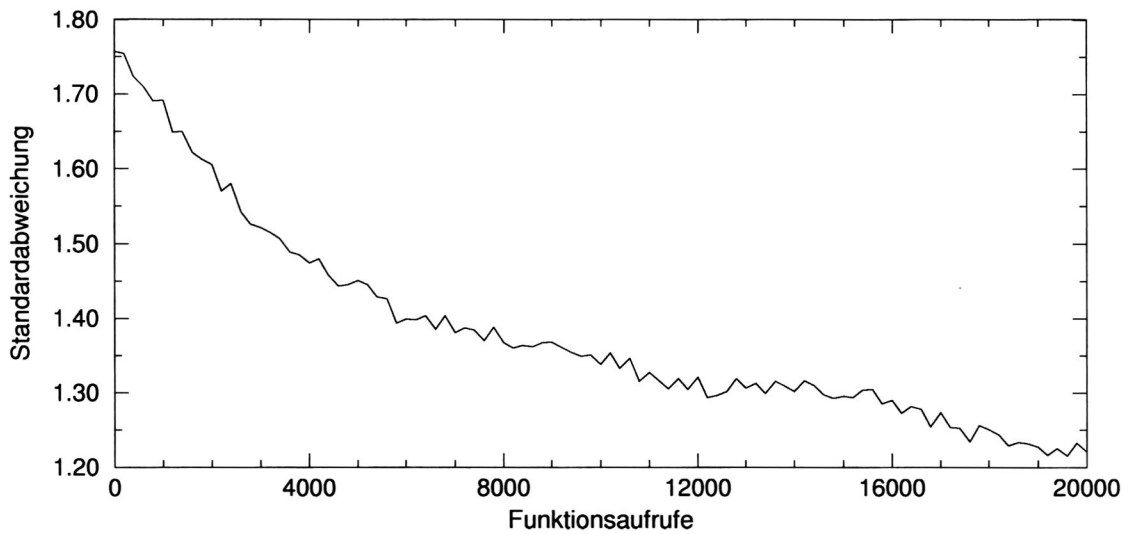


Abbildung 4.45: Verlauf der Standardabweichung bei der Optimierung des Salzstocks. Dargestellt ist der Durchschnitt aus zehn Optimierungen.

Dargestellt sind Durchschnittswerte aus zehn Optimierungsläufen. Das Residualfeld (Abbildung 4.46) zeigt auch nach der Optimierung deutliche Abweichungen zwischen den gemessenen und berechneten Daten.

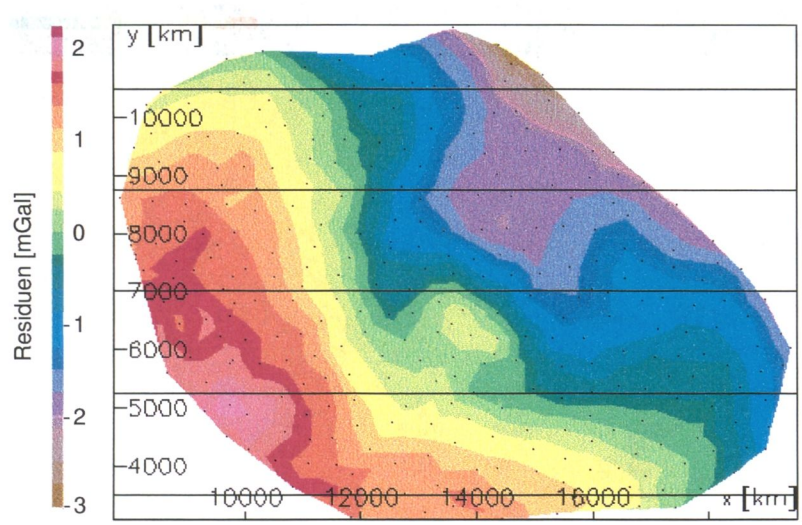


Abbildung 4.46: Residualfeld gemessener und berechneter Daten nach der Optimierung. Es sind deutliche Abweichungen zu erkennen.

Betrachtet man die Anomalien vor und nach der Optimierung, erkennt man aber eine klare Verbesserung gegenüber dem Ausgangszustand (Abbildung 4.47).

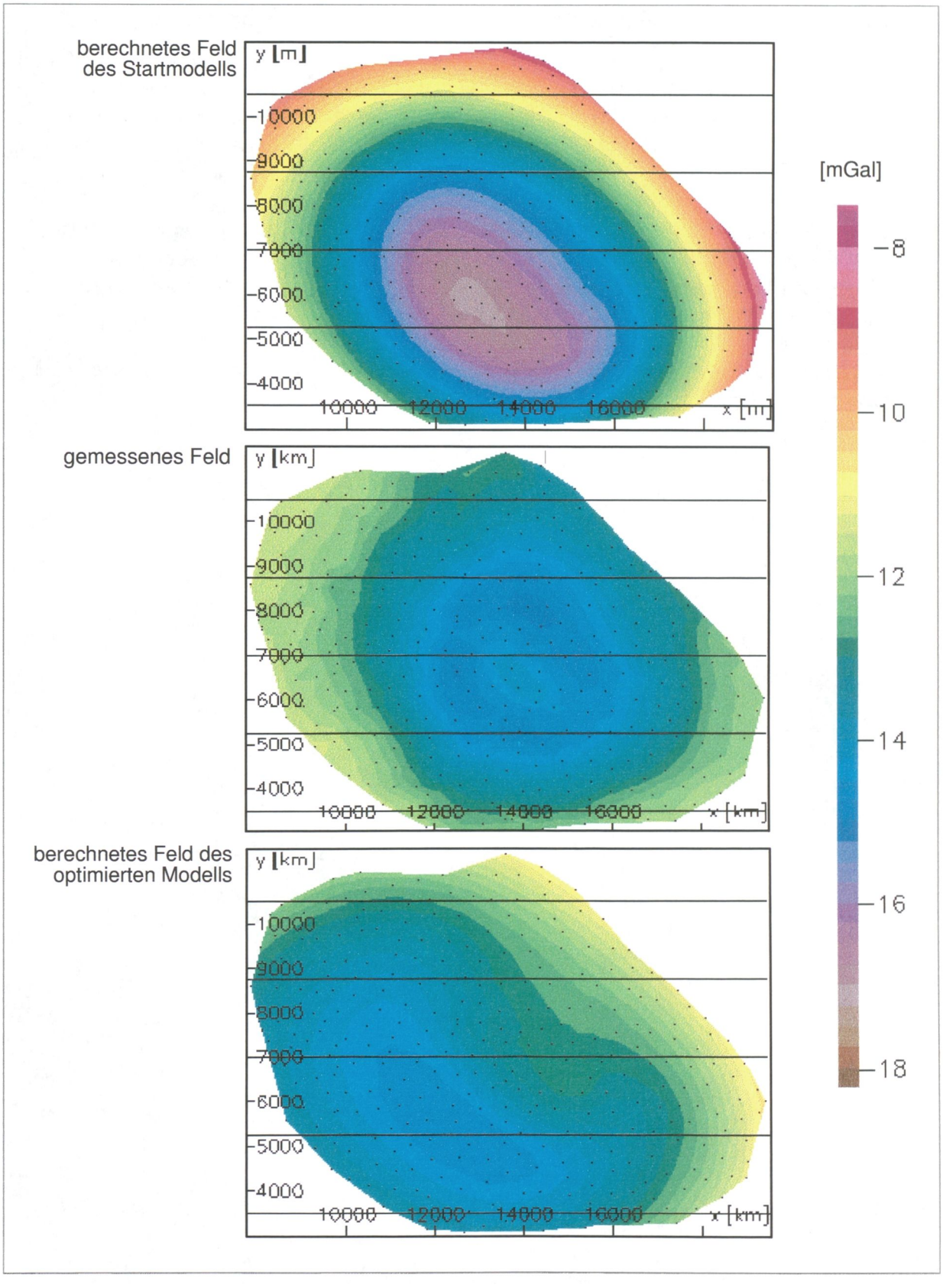


Abbildung 4.47: Die berechnete Anomalie des optimierten Modells (unten) ist den gemessenen Daten ähnlicher als die des Ausgangsmodells.

4.3.3 Zusammenfassung der Ergebnisse für 3D-Optimierung

Der vorgestellte Algorithmus zur Mutation der Modellgeometrie arbeitet in Verbindung mit der (1,10)-Evolutionsstrategie recht gut. Da die Optimierung aufgrund sich überschneidender Dreiecke abgebrochen werden mußte, ist die erreichte Übereinstimmung von gemessenen und berechneten Daten nicht zufriedenstellend. Hier sollte ein geeigneteres Mutationsverfahren geschaffen werden, das Überschneidungen a priori ausschließt.

Das verwendete Mutationsverfahren eignet sich nicht für die CMA-Evolutionsstrategien, da in jeder Generation eine Inititialisierung der Strategie stattfinden muß und somit die adaptierten Richtungen verlorengehen. Ein neues Mutationsverfahren sollte deshalb derart angelegt sein, daß es die Anwendung der CMA-Verfahren ermöglicht.

Im folgenden soll eine Idee vorgestellt werden, welche die beschriebenen Probleme beheben könnte.

5

Ausblick

Wie bereits im Abschnitt 4.3 beschrieben wurde, sind Überschneidungen einzelner Polygonseiten bzw. Dreiecke, welche aufgrund von Mutationen entstehen können, das Hauptproblem bei der Optimierung zwei- und dreidimensionaler geometrischer Strukturen. Prinzipiell kann durch die Einführung von Randbedingungen verhindert werden, daß es zu derartigen Überschneidungen kommt. Randbedingungen schränken den Parameterraum aber möglicherweise unzulässig ein und müssen in aufwendiger „Handarbeit" für jeden einzelnen Parameter vergeben werden. Bei Problemstellungen, bei denen die Anzahl der Parameter nicht zu groß ist, ist dies mit vertretbarem Aufwand möglich. Bei realitätsnäheren 3D-Modellen ist ein derartiges Herangehen allerdings nicht praktikabel. Außerdem ist die Behandlung von verletzten Randbedingungen keineswegs trivial. Bis heute kann kein allgemeingültiges Konzept zur Korrektur von überschrittenen Rändern gegeben werden, welches sich für alle Optimierungsverfahren gleichermaßen eignet. Die bislang verwendete Methode, das Zurücksetzen des Parameters auf den Rand bzw. die Einführung einer Straffunktion beim Überschreiten der Randbedingungen, beeinflußt den Optimierungsprozeß aufgrund der Einschränkung des Parameterraums durchaus. Auch die automatische Generierung von Randbedingungen muß als schwierig bezeichnet werden. Falls möglich, würde dies sicherlich zu einer dramatischen Einschränkung der Bewegungsfreiheit der einzelnen Modellpunkte führen, besonders wenn die Auflösung des Modells sehr „fein" ist, also der Abstand zwischen den Modellpunkten sehr gering ist (siehe Abbildung 5.1).

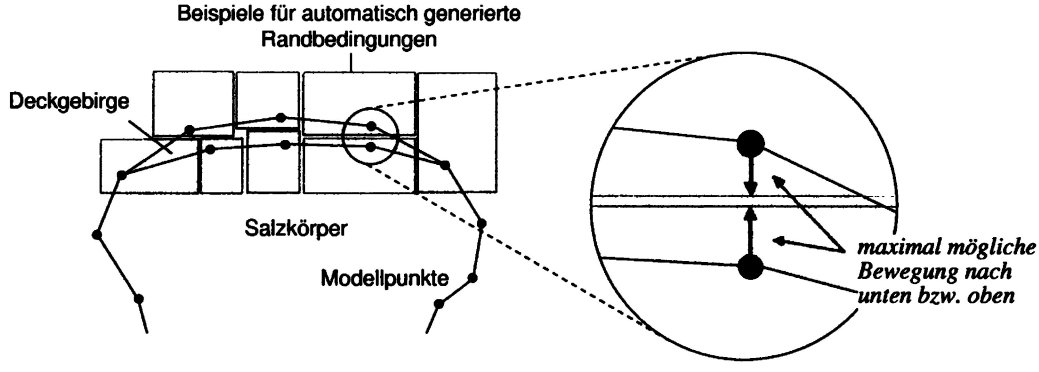


Abbildung 5.1: Beispiel für „zu enge" Randbedingungen. Liegen die Modellpunkte dicht beieinander, schränken Randbedingungen die Bewegungsfreiheit stark ein.

Die Abbildung verdeutlicht, daß eine Generalisierung der Strukturen häufig nicht möglich ist, da auch kleine Körper - z.B. ein Deckgebirge hoher Dichte über einem Salzstock - eine große gravimetrische Wirkung haben können und nicht vernachlässigt bzw. mit anderen Körpern zusammengefaßt werden können.

Ein möglicher Ausweg könnte sein, Mutationsoperatoren zu definieren, bei denen es a priori keine Überschneidungen geben kann. Es ist allerdings klar, daß dies durch die direkte Mutation der Geometrieparameter (ohne Randbedingungen, deren Problem bereits erörtert wurde) nicht erreicht werden kann. Daher besteht der Bedarf nach einem Konzept, welches die beschriebenen Probleme umgeht. Bei einem neuen Verfahren sollte versucht werden, nicht das Modell selbst zu optimieren, sondern der (fiktive) Raum, in dem es sich befindet sollte derart „verzerrt" werden, daß die Geometrie des Modells (welches im verzerrten Raum seine Geometrie mit ändert) optimale Form im Sinne der definierten Qualitätsfunktion annimmt. Diese „Raumtransformation" sollte dabei derart gewählt werden, daß die Topologie einer Struktur, z.B. eines aufgespannten Gitters, erhalten bleibt. Optimiert werden dann die „Koeffizienten" der Transformationsvorschrift.

5.1 Konforme Abbildungen

Die erste Idee, mit der dies möglicherweise realisiert werden könnte, beruht auf konformen Abbildungen (Greul und Kander, 1990; Kreyszig, 1993). Konforme Abbildungen basieren auf Transformationen in der komplexen Zahlenebene und können z.B. Phänomene der Elektrostatik und der Aerodynamik adäquat beschreiben. Sie werden auch in der Bildverarbeitung zur Erzeugung von Verzerrungseffekten benutzt. Leider sind sie nicht immer bijektiv und garantieren daher nicht den Erhalt der Topologie. Außerdem existieren konforme Abbildungen nur für zweidimensionale Probleme, da die komplexe Zahlenebene benötigt wird. Ein vergleichbares Konzept existiert für drei Dimensionen nicht (Kreyszig, 1993). Die Abbildung 5.2 zeigt ein Beispiel einer konformen Abbildung zweiter Art, die die Punktmenge eines Kreises in ein Tragflügelprofil transformiert (aus Greul und Kander (1990), S. 40).

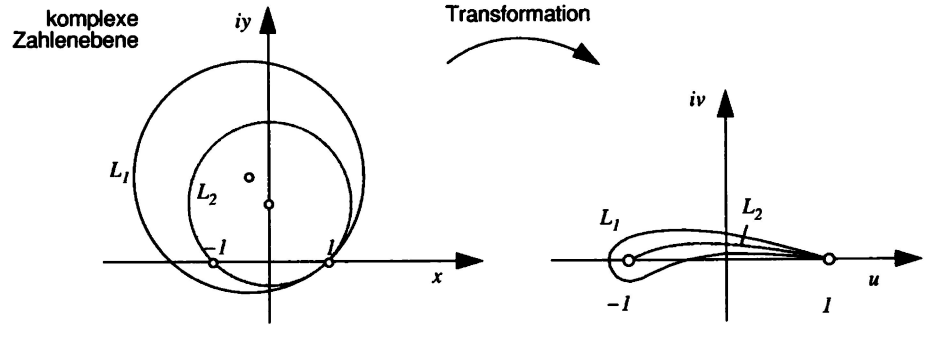


Abbildung 5.2: Beispiel einer konformen Abbildung zweier Kreise in ein Jukowskiprofil mit der Transformationsvorschrift $f(z) = z + \frac{1}{z}$ wobei $z \in C$.

Konforme Abbildungen wurden nicht weiterverfolgt, da kein anschauliches und elegantes Konzept zur Optimierung der Koeffizienten der Transformationsvorschrift abgeleitet werden konnte. Es konnte keine Methode entwickelt werden, die die Bijektivität der verwendeten Funktionen garantiert. Allerdings ist nicht auszuschließen, daß durch intensivere Arbeit

bzw. Literaturstudium ein derartiges Konzept gefunden werden kann. Daher sollten Konforme Abbildungen auch in zukünftigen Arbeiten nicht unberücksichtigt bleiben.

5.2 Manipulatoren

Ein zweiter Ansatz, der die beschriebenen Probleme der Randbedingungen umgehen soll, sind sog. Manipulatoren. Manipulatoren haben „Kraftfelder", welche „Anziehungskräfte" auf die Eckpunkte des Modells haben (Abbildung 5.3).

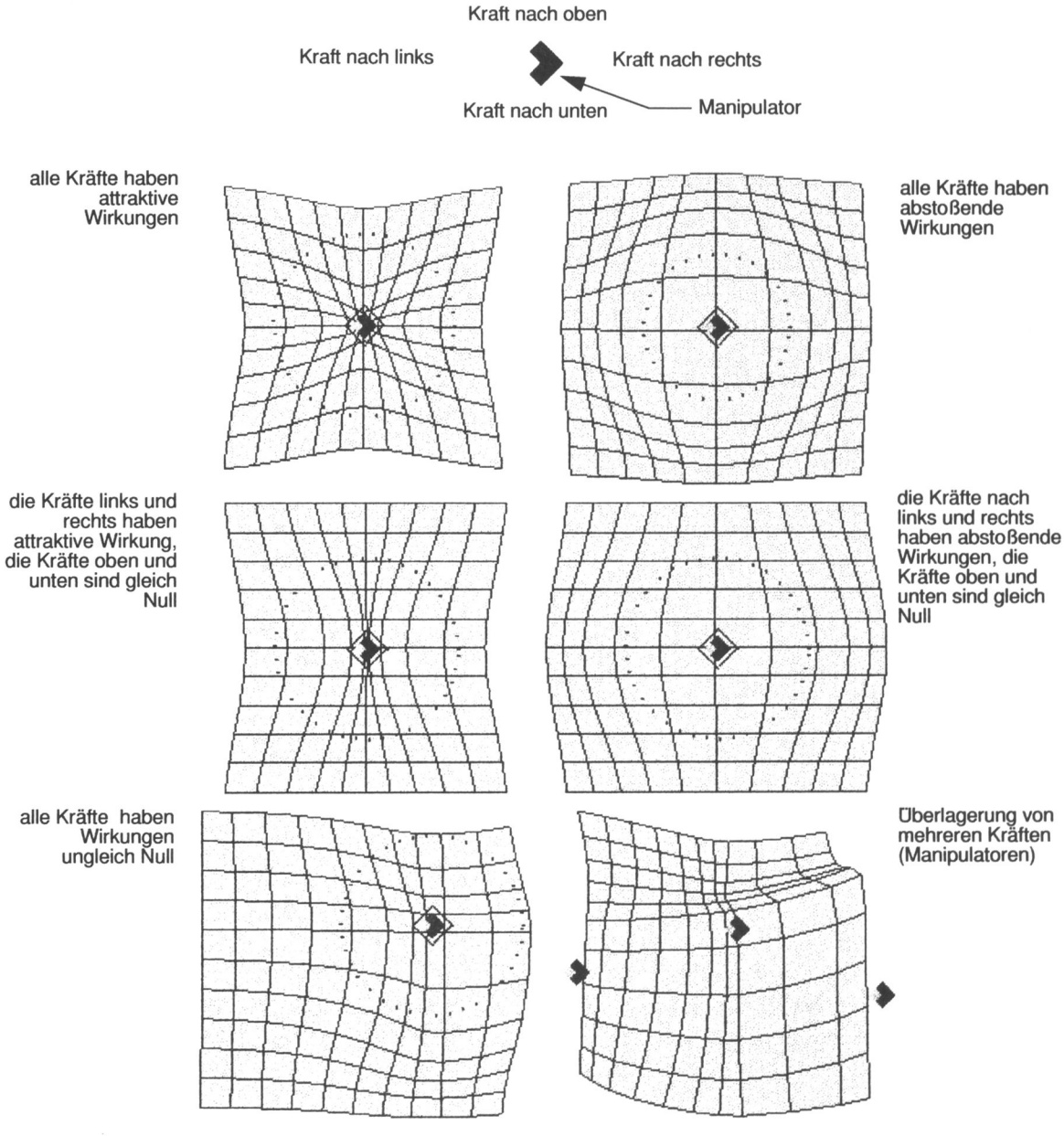


Abbildung 5.3: Beispiele der Verzerrung des Raumes durch „Manipulatoren". Bei der Überlagerung von mehreren Manipulatoren ist der Erhalt der Topologie nicht zwingend gewährleistet.

Diese „Kräfte“ können nach oben, unten, rechts und links verschieden sein und nehmen mit dem Abstand zum Punkt entsprechend einer vorgegebenen Funktion, z.B. einer Exponetialfunktion, ab. Das Konzept läßt sich problemlos auf die 3D-Optimierung erweitern, indem eine „*vor*“- und eine „*zurück*“- Kraft eingeführt wird. Die Abbildung 5.3 zeigt einige zweidimensionale Beispiele für mögliche Veränderungen eines regelmäßigen Gitters mit Manipulatoren. Bei der Verwendung mehrerer Manipulatoren überlagern sich die „Kräfte“ allerdings und es kann auch hier nicht garantiert werden, daß die ursprüngliche Topologie erhalten bleibt. Es kann zu Überschneidungen der Gitterlinien kommen. Daher führte auch dieser Ansatz nicht zu dem gewünschten Erfolg. Allerdings eignen sich Manipulatoren aufgrund der geringen Anzahl von Parametern (vier bzw. sechs Kräfte pro Manipulator) sehr gut, um durch interaktives Verändern der Kräfte einen sehr schnellen Überblick zu erhalten, wie das Modell modifiziert werden muß, so daß es optimal im Sinne der Qualitätsfunktion wird.

5.3 „Raumkrümmung“

Ein Verfahren, welches aus der Analyse der konformen Abbildungen und der Manipulatoren entstanden ist, umgeht die beschriebenen Probleme und führt nur wenig neue ein. Es entstand auch in Anlehnung an Methoden der Entzerrung von Bildern (z.B. Satellitenbildern, siehe Lillesand und Kiefer (1994)) durch Paßpunkte.

Um es einzusetzen, wird der „Raum“, in dem sich der Teil des Modells befindet, dessen Geometrie optimiert werden soll, zunächst mit einem regelmäßigen Gitter „überzogen“. Jeder Geometriepunkt des Modells hat in diesem Gitter eine eindeutige Position relativ zu einem bestimmten Gitterelement (vgl. Abbildung 5.4 a).

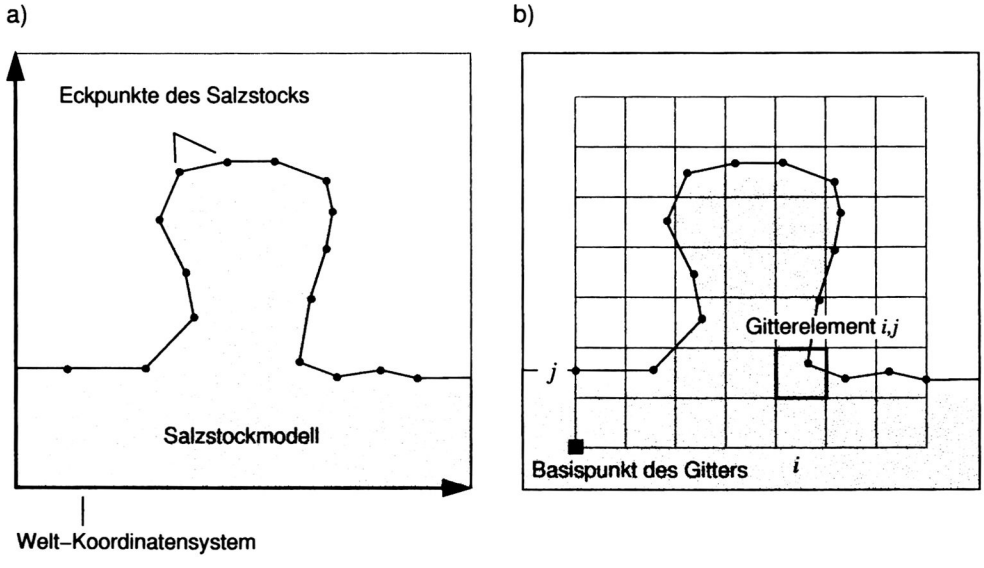


Abbildung 5.4: Dem zu optimierenden Modell (a) wird ein regelmäßiges Gitter überlagert (b). Jedem Eckpunkt kann eindeutig ein Gitterelement i, j zugeordnet werden.

Die Idee besteht darin, das Gitter derart zu verändern–„wie“ bleibt dabei zunächst offen–daß die Form des in diesem „Raum“ befindlichen Modells optimal im Sinne der Qualitätsfunktion wird (Abbildung 5.5 a). Nach der Verzerrung des Gitters kann jeder Punkt innerhalb des jeweiligen Gitterelements linear interpoliert werden (Abbildung 5.5 b). Daraus ergibt sich die Frage, wie es möglich ist, das Gitter derart zu verändern, daß es sich nicht selbst schneidet. Eine Möglichkeit besteht darin, die Gitterabstände d_{ij_x} und d_{ij_z} (im dreidimensionalen Fall auch d_{ij_y}) zu variieren. Dabei dürfen die Gitterabstände nicht kleiner Null werden, da sich das Gitter sonst selbst schneiden und die ursprüngliche Topologie zerstört würde. Eine vorher festgelegte maximale Gitterweite max_g ist sinnvoll, da so a priori bekannt ist, in welchem Intervall $[0, max_g]$ sich eine „Gitterbreite“ d_{ij_k} *mit* $k = x, z, y$ bewegen darf.

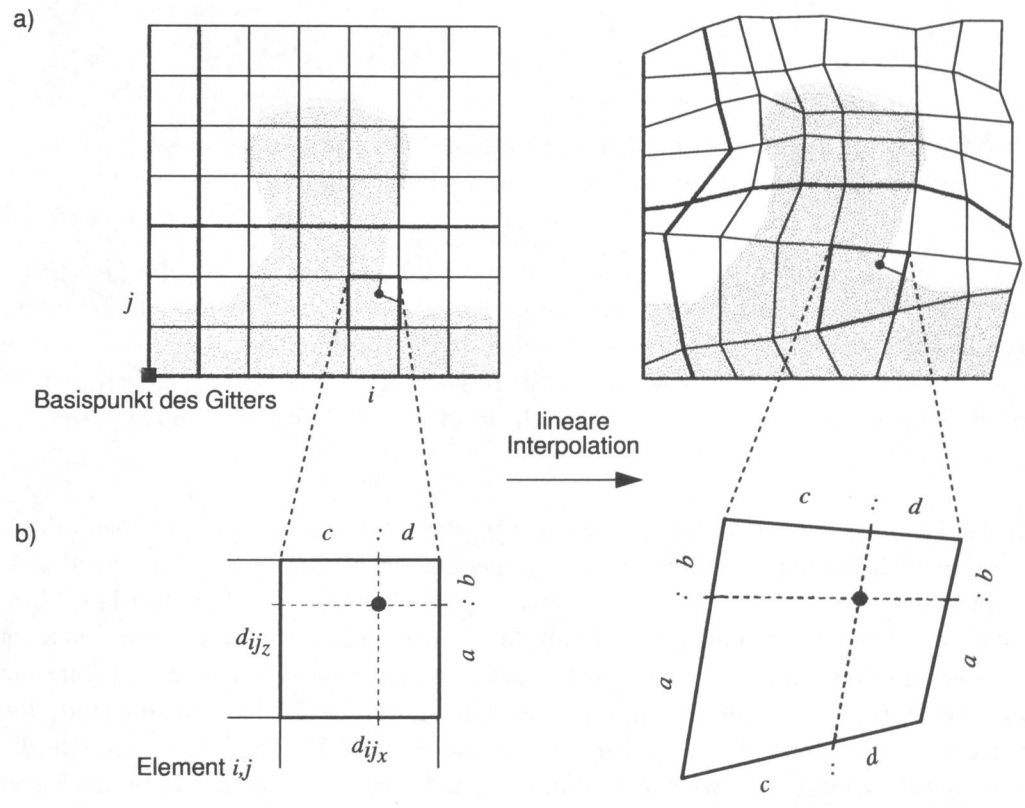


Abbildung 5.5: Durch die Mutationen der Gitterabstände d_{ij_x} und d_{ij_z} wird das Gitter verzerrt. Die Eckpunkte des Modells können linear nach den neuen Abmessungen des entsprechenden Gitterelements interpoliert werden.

Die Optimierung der Modellparameter wurde somit auf die Optimierung der Gitterabstände verlagert. Gewonnen wurde dabei die sichere Erhaltung der Topologie des Gitters und damit auch die Erhaltung der Topologie des Modells. Die Abbildung 5.6 zeigt das Ergebnis der Transformation. Es ist zu beachten, daß diese Veränderungen *relativ* vom einem Element zum nächsten erfolgen müssen. Daher ist auch die *Reihenfolge*, mit der die einzelnen Gitterabstände bearbeitet werden, von Bedeutung. Folgerichtig ergibt sich die Frage nach einem „Bezugspunkt“ im Modell. Dieser Bezugspunkt, von dem aus alle Variationen durchgeführt werden, muß bei diesem Verfahren vom Benutzer interaktiv vorgegeben werden.
In der Abbildung 5.4 b) wurde er willkürlich in die linke untere Ecke des Gitters gelegt. Ein derartiges Bezugssystem schränkt die Freiheit der Optimierung a priori ein, da es

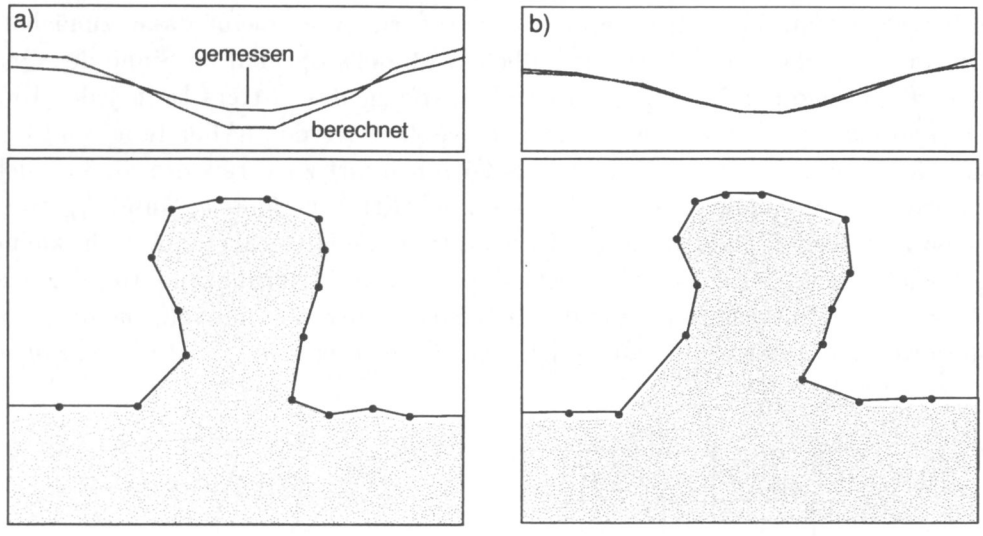


Abbildung 5.6: Nach der Optimierung ist die Anpassung gemessener und berechneter Felder besser (schematische Darstellung).

sein könnte, daß von einem falsch gewählten Bezugspunkt ausgehend, die gesuchte Form prinzipiell nicht konstruierbar ist. Allerdings könnte die Position dieses Punktes, also die Lage des Gitters relativ zum Modell, selbst *mit optimiert* werden. Alles in allem ist dies aber der einzige offensichtliche Nachteil dieses Verfahrens. Er ist zu vernachläßigen, betrachtet man den enormen Vorteil, der sich durch den Erhalt der Topologie des Modells ergibt.

Als günstig ist auch die Möglichkeit zu bewerten, das Modell mit einem groben oder einem feinen Gitter zu überlagern. So kann die numerische Auflösung gesteuert und die Anzahl der zu optimierenden Parameter je nach gewünschter Präzision gewählt werden. Man kann sich auch vorstellen, die Auflösung im Laufe der Optimierung zu „verfeinern", also zu Gittern engerer Maschen überzugehen und die gefundenen groben Strukturen „festzuhalten" bzw. zu speichern. Es ist auch denkbar, das Gitter im Laufe der Optimierung *lokal* zu verfeinern und an anderer Stelle zu vergröbern. Auch diese Fragen (welches Gitterelement soll feiner strukturieert und welche zusammengefaßt werden sollen) könnten „Parameter einer Optimierung" werden.

Im letzten Abschnitt wurden Ideen formuliert, deren Realisierung noch nicht untersucht worden ist. Es muß zunächst geklärt werden, inwiefern sie praktikabel für den Einsatz bei der Modellierung sind. Für zukünftige Entwicklungen im Rahmen der Geometrieoptimierung ist daher vorzuschlagen, diesen Ansatz weiterzuverfolgen und zu prüfen. Die Eleganz besteht nicht zuletzt in der fast trivialen Übertragbarkeit auf dreidimensionale Optimierungsprobleme, die - wie gezeigt - sehr schwierig zu behandeln sind. Nachteile dieser Methode sehe ich in den eingeschränkten bzw. nicht vorhandenen Möglichkeiten der Modifikation der Modelltopologie. Allerdings kann dies zunächst vernachlässigt werden, da es wohl bei der Optimierung von Potentialfeldmodellen eher auf eine Modifikation von Modellideen ankommt und die eigentliche geistige Leistung, die Erstellung der Starttopologie, beim interpretierenden Anwender liegt. Ohne ein plausibles Startmodell ist eine Optimierung ohnehin nicht sinnvoll - oder besser gesagt, unmöglich.

6

Zusammenfassung

In dieser Arbeit wurden verschiedene Optimierungsverfahren auf die Geometrie- und Parameteroptimierung von Potentialfeldmodellen unter Randbedingungen angewendet. Ein Vergleich zeigte die Vor- und Nachteile der einzelnen Verfahren.

Anhand von zwei synthetischen Testfunktionen, welche symmetrisch bzw. in den einzelnen Koordinaten logarithmisch skaliert waren, wurde das Verfahren „Downhill-Simplex" mit der „(1,10)-CMA-Evolutionsstrategie" verglichen. Das Simplexverfahren zeigte für Dimensionen kleiner zehn ein besseres Konvergenzverhalten als die Evolutionsstrategie. Wurde die Dimension darüberhinaus erhöht, konvergierte die Evolutionsstrategie deutlich besser. Simplex konvergiert bei mehr als 40 Parametern nicht mehr. Die Evolutionsstrategie zeigt unabhängig von der Dimension logarithmische Konvergenz, wobei bis zu 200 Parametern getestet wurde.

Die Evolutionsstrategie, genetische Algorithmen, Simulated-Annealing, Threshold-Accepting und der Delug-Algorithm wurden danach anhand eines zweidimensionalen Salzstockmodells getestet. Dabei wurden wiederum nur Geometrieparameter optimiert und die Dichten des Modells konstant gehalten. Die 52 zu optimierenden Geometrieparameter wurden mit Randbedingungen versehen. Eine Straffunktion, die bei Winkeln kleiner 90 Grad einsetzte, sorgte für die Glattheit des Modells. Bei Überkreuzungen einzelner Polygone wurde das erzeugte Modell verworfen. Die Ergebnisse der Tests wurden am Ende der jeweiligen Abschnitte ausführlich zusammengefaßt. Hier sollen die wesentlichen Erkenntnisse wiederholt werden:

- Die Evolutionsstrategie mit Kovarianzmatrixadaption benötigt im Durchschnitt die wenigsten Funktionsaufrufe und erreicht die besten Qualitäten. Sie findet von verschiedenen Startmodellen ausgehend ähnliche Lösungen, die alle sehr gute Qualitätswerte haben. Evolutionsstrategie ist stabil gegenüber numerischen Variationen. Rekombination verbessert die Konvergenzeigenschaften immer.

- Genetische Algorithmen konvergieren anfangs besser als alle anderen Verfahren, erreichen aber nie die von der Evolutionsstrategie gefundenen besten durchschnittlichen Werte, wobei der Durchschnitt jeweils aus 100 Optimierungsläufen gebildet wurde. Dies ist wahrscheinlich auf das Fehlen eines Adaptionsmechanismus der lokalen Gege-

benheiten der Qualitätsfunktion zurückzuführen. Es konnte gezeigt werden, daß bei dieser Optimierung die Mutationsrate einen sehr großen und die Rekombinationswahrscheinlichkeit einen eher geringen Einfluß hat. Das binäre System zur Codierung hatte schlechteste, eine dezimale Codierung hatte beste Konvergenzeigenschaften zur Folge. Getestet wurde bis zur Basis 20. Die Ergebnisse unterscheiden sich kaum von denen des dezimalen Systems.

- Downhill-Simplex, Simulated-Annealing, Threshold-Accepting und der Delug-Algorithm zeigten insgesamt schlechteres Konvergenzverhalten als die Evolutionsstrategie und genetische Algorithmen. Auch hier ist dies im wesentlichen auf das Fehlen eines Adaptionsmechanismus für die Variationsverteilung zurückzuführen.

Da die Evolutionsstrategie die besten Ergebnisse zeigte, wurde sie zur 3D-Optimierung verwendet. Als Testmodell diente wiederum ein auf seismischer Modellierung basierendes Salzstockmodell. Zur Mutation der Geometrie wurden Mutationsoperatoren entwickelt, die eine gewisse Glattheit der Geometrie gewährleisten sollen. Leider macht dieses Verfahren die Anwendung der CMA-Strategien aufgrund von in jeder Generation nötigen Initialisierungen unmöglich, so daß die Konvergenzeigenschaften durch die Anwendung dieses adaptiven Verfahrens deutlich verbessert werden könnten.

Die erreichten Qualitäten sind mit denen der 2D-Optimierung nicht vergleichbar. Aufgrund der hohen Dreiecksdichte liegen die Geometrieparameter dicht beieinander. Dies hatte zur Folge, daß die Optimierung trotz des eigens dafür konzipierten Mutationskonzeptes zu einem zu frühen Zeitpunkt abgebrochen werden mußte, da es zu vielen Überschneidungen einzelner Dreiecke kam. Es konnte lediglich der Trend der Verjüngung des Salzkörpers, der bei den 2D-Optimierungen deutlich zu erkennen war, reproduziert werden. Auch die Varianz der gemessenen und berechneten Daten konnte deutlich verbessert werden. Aus dem Verlauf der gemittelten Qualitätskurven ist aber zu erkennen, daß die Optimierung noch nicht stagniert.

Basierend auf diesen Erfahrungen wurde ein Methode diskutiert, welche störende Überschneidungen a priori unmöglich macht. Die Implementierung, die Anwendung und der Test dieser Methode stehen noch aus und sollten Gegenstand weiterer Forschungen sein.

Anhang

Alle in dieser Arbeit durchgeführten Tests der Optimierungsalgorithmen wurden mit Hilfe eines eigens für diese Aufgabenstellungen konzipierten 2D-Modellierungs- und Konstruktionsprogramms durchgeführt. Dieses Programm heißt DARWIN++ und wurde in der objektorientierten Programmiersprache C++ geschrieben. Es erlaubt die Optimierung von zweidimensionalen gravimetrischen und/oder magnetischen Modellen unter Randbedingungen mit sieben verschiedenen Optimierungsalgorithmen. Ein anderer wesentlicher Aufgabenbereich des Programms ist die interaktive Konstruktion von 3D-Modellen, basierend auf 2D-Schnitten.

Bei der Entwicklung der Software wurde besonderer Wert auf folgende Punkte gelegt:

- intuitive Bedienung,
- Echtzeit-Berechnungen, [1]
- Modularität der Programmarchitektur, die eine Erweiterung des Programms für 3D-Berechnungen sicherstellt und
- eine *nonblocking event handeling* (nicht-blockierende Ereignisverarbeitung).

Nach dem Lesen der folgenden Abschnitte sollte ein Anwender mit folgenden Punkten vertraut sein:

- Modelle in DARWIN++ laden und deren Struktur sowie deren physikalischen Parameter (Dichte und/oder Suszeptibilität) variieren,
- alle Zoom- und Bewegungsfunktionen anwenden,
- veränderte Modelle abspeichern,
- vergeben von Randbedingungen für Geometriepunkte und physikalische Parameter sowie dem
- Arbeiten mit den zur Verfügung stehenden Optimierungsalgorithmen.

Für alle anderen Funktionalitäten, wie z.B. dem Drucken von Modellen, bietet DARWIN++ eine *online*-Dokumentation.

[1] Der Eindruck, daß eine Berechnung in „realer Zeit" abläuft, entsteht dann, wenn sie mindestens 10 mal pro Sekunde durchgeführt werden kann (Animationseffekt). Werden z.B. viele Punkte selektiert und bewegt und sind sehr viele Stationen (online) zu berechnen, ist dieses Verhalten auch bei heute üblichen Rechenleistungen mittlerer *workstations* nicht immer gewährleistet.

1

The Program

DARWIN++ is a 2D-modelling program for gravity and/or magnetic models with construction capability for 3D models. Here a description how models may be displayed and how the main window can be customized will be given. Getting started:

- start darwin by typing **`darwin example.darwin`** in the directory where **`example.darwin`** is located. [1]
- After the program start a model should be seen in the main window with some vertices selected, indicated by small red boxes around the vertices.
- Move the mouse over the model (lower part) and observe status line (see Figure 1.1 on page 89).
- Unselect vertices by pushing the escape key on your keyboard.
- Place mouse over a vertex. If the cursor shape has changed to a little cross, shortly click left mouse button to select the vertex. Deselect vertex by pushing the escape key.
- Place mouse over a vertex. If the cursor shape has changed to a little cross, click left mouse button **and hold button down**. Now move the mouse to move vertex. Deselect vertex.
- Place mouse somewhere over the model but not in the vicinity of a vertex and press left mouse button **and hold button down**. Sweep out a box to select vertices within the box. Deselect selected vertices.
- Place mouse over a body but not in the vicinity of a vertex. Click shortly left mouse button to select the body, indicated by a thicker outline around the polygon. Deselect body by pushing the escape key.
- Perform the actions described in the two items above with the shift key held down to select more than one vertex/body.
- Hold down middle mouse key and move up and down to zoom in and out.
- Hold down right mouse key to move model.

[1] It is recommended to make a backup copy of **`example.darwin`** before loading it into DARWIN++. On UNIX systems type for example **`cp example.darwin example.darwin.bak`**

1.1 The main window

Figure 1.1 shows the DARWIN++ main window.

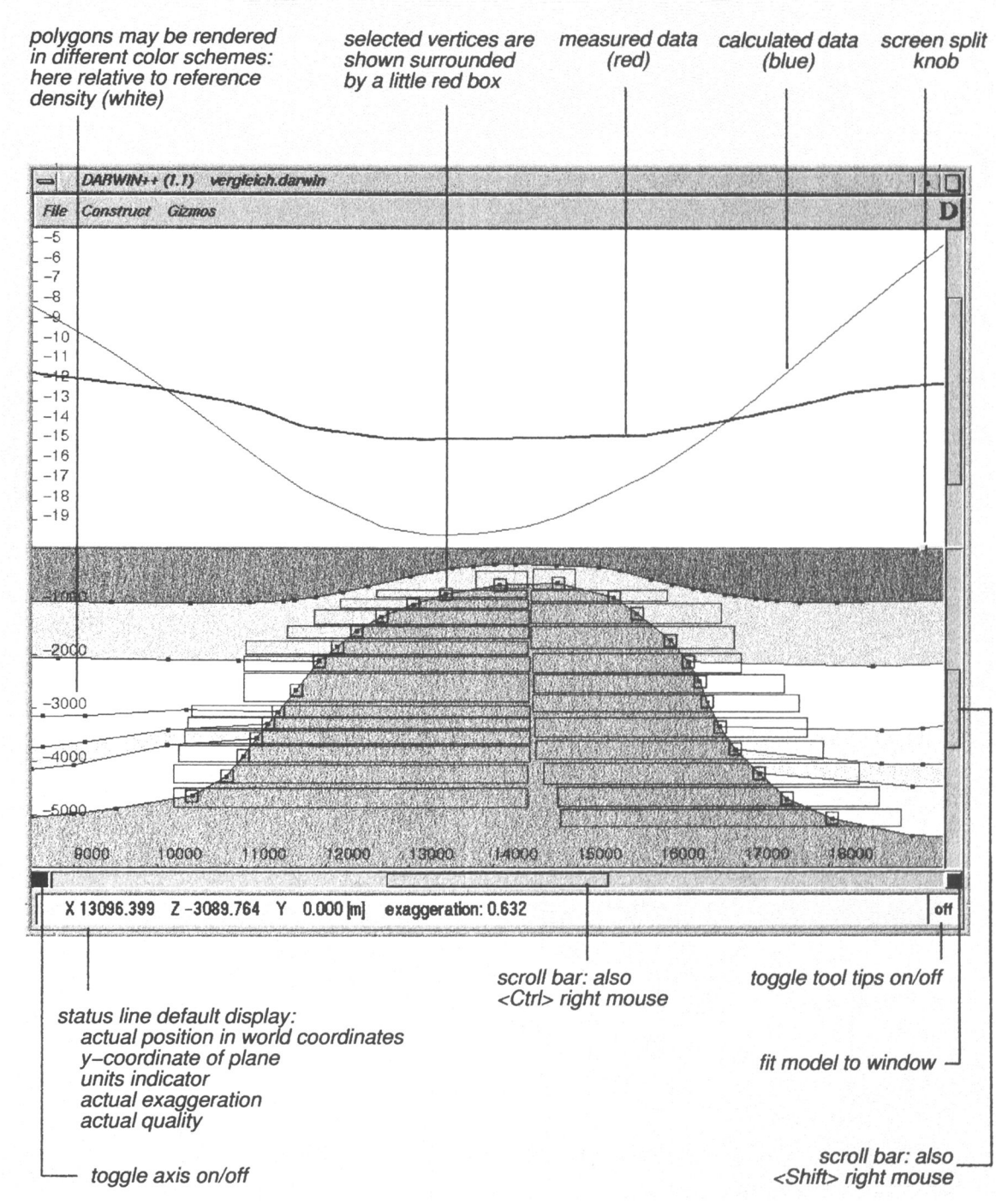


Figure 1.1: The main window.

In the lower half, the model is shown and in the upper half, the measured and calculated data are displayed. The status line shows helpfull values for modelling.

2

Optimization

2.1 Getting started

The program DARWIN++ is able to optimize model parameters like geometry and/or densities/susceptibilities. Parameters to be optimized can be constrained. To get started do this:

- start the program, by typing **darwin simple-opti.darwin** in the directory where **simple.darwin** is located [1]
- After the program start, a simple model should be seen in the main window with two vertices selected, indicated by little red boxes around the vertices.
- If the optimization gizmo is not visible, choose *optimization* from the gizmo's pull down menu (see figure 2.1).
- Start optimization by clicking on the start button (see figure 2.2 on page 91).

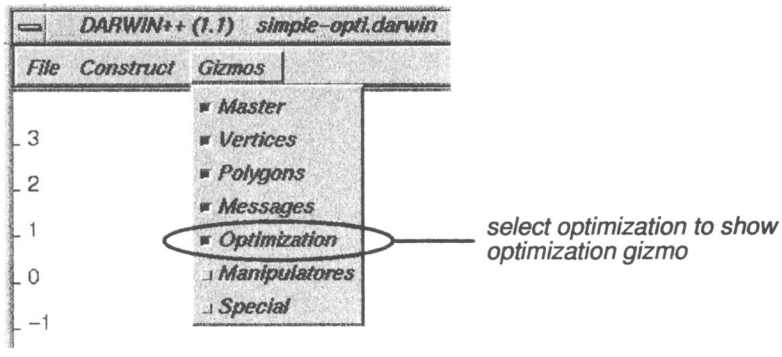


Figure 2.1: *Gizmos* pull down menu.

Curves of measured and calculated data should merge with increasing number of function calls. The optimization can be stopped at any time by selecting the stop button (see figure 2.2). The optimization will be stopped after a few seconds. You can then restore the original model and/or select different parameters (vertices, bodies) and run the program again.

[1] It is recommended to make a backup copy of **simple.darwin** before loading it into DARWIN++. On UNIX systems type for example **cp simple.darwin simple.darwin.bak**.

You may also continue with the optimization already started. To explore the optimization capabilities, it is suggested to change initial configurations by changing the position and/or the number of selected vertices (and/or bodies) of the model **simple-opti.darwin** and run optimization again. Make sure there are selected vertices (and/or bodies) before continuing or running optimization again. If no parameters are selected, you will be asked to select some.

2.2 Selection of algorithms

DARWIN++ provides seven algorithms for parameter optimization that can be chosen on the optimization gizmo:

- evolution strategy (ES)
- genetic algorithms (GA)
- great deluge algorithm (GD)
- simulated annealing (SA)
- downhill simplex (SX)
- threshold accepting (TA)
- monte carlo (MC)

Figure 2.2 shows the gizmo.

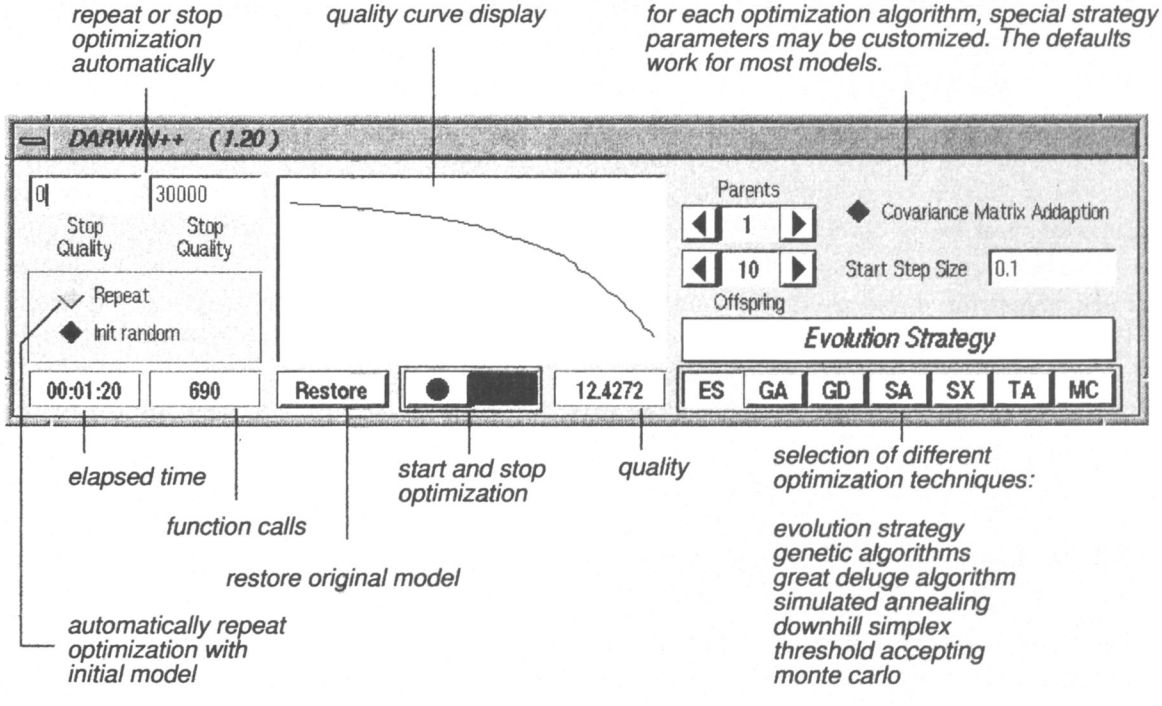


Figure 2.2: The optimization gizmo.

To apply optimization to model parameters, select vertices or bodies. It is highly recommended to constrain parameters prior to using automatic optimization techniques. To set constraints for parameters, refer to section 2.3. If constraints are not given, only an intersection-check will be performed and most of the generated models are rather unrealistic due to their geometry. Solutions (models) for potential field data are not unique and therefore many possible solutions exist. The used algorithms will find one of these models, but not necessarily the right one. Although the un-uniqueness is not totally avoided by constraints, solutions are fairly stable.

During optimization you can toggle between different techniques. However, this is not recommended because most of the algorithms adopt themselves to the local topography of the quality function (in the following the terms "quality" and "variance" are used vice versa). Toggling between techniques causes initialisation of algorithmic parameters and the learned data will be lost. While the program is optimizing, the best model (ever) found is stored and automatically written to a file named `<autowrite.darwin>`. For statistical purposes, optimization can be repeated automatically after a given number of generations or a given quality is reached.

Note: optimization may need hours if not even days to find *the* solution. Although, most often it is faster.

One of the most frequently asked questions about DARWIN++'s optimization capabilities is, which algorithm is appropriate for a specific problem. Unfortunately, no general suggestion can be given. As a rule, it is always good to start with *evolution strategy* because it works best for most problems. If ES does not work, *genetic algorithms* can be tried. For *fine tuning* of parameters, ES may be applied again. For very simple problems (e.g. only one or two vertices or densities/susceptibilities have to be optimized) *downhill simplex* is often sufficient. If all fail–which is the case some times– continue with *great deluge, threshold accepting, simulated annealing* and *monte carlo*. Because most scientists are familiar with *monte carlo*-like techniques, it has been implemented as well. You may also compare other algorithms to this greedy search method. The *monte carlo* technique is very slow but has one most important advantage: it searches the whole parameter space and does not focus on certain subspaces like sub-optima.

Each algorithm implemented has some strategy parameters which **may be** customized. **Without knowledge** about the techniques **it is recommended to use defaults** given by the program. However, if the algorithm does not find any plausible solution with default settings, the strategy parameters have to be changed. A brief description of these parameters for each algorithm follows.

2.2.1 Evolution Strategy (ES)

Evolution strategy adopts the natural principles of evolution: recombination, mutation and selection for parameter optimization. ES was developed by Rechenberg (1973) and significantly improved by Hansen und Ostermeier (1997). ES applies to a wide variety of problems in natural and engineering sciences. ES works in parallel and uses a population of model individuals that undergo the process of evolution. For a detailed explanation, refer to

Rechenberg (1994), Hansen und Ostermeier (1997), Schwefel (1977, 1995). In DARWIN++ the following parameters may be set:

- number of parents,
- number of offspring,
- step size at the beginning and
- covariance matrix adaption (CMA) can be toggled on/off.

Figure 2.3 shows where to change these parameters.

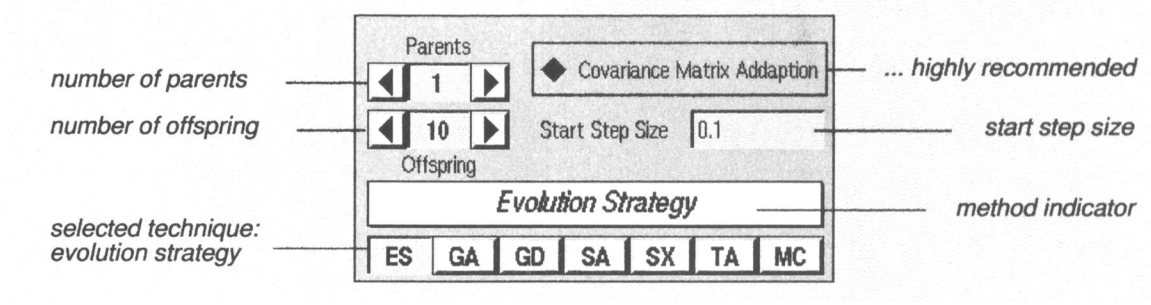


Figure 2.3: Evolution strategies parameter settings.

Strategy parameters for ES are noncritical. The algorithm's ability to adopt the variation distribution to local characteristics of a given optimization problem is most delightful. In general, if CPU time is "unlimited", large numbers of offspring give good results in terms of convergence speed and stability. The number of parents may be set to a fifth of the number of offspring. If the number of parents is greater than one, individuals are recombined by default.

In practice, a (6/6,30)-ES-CMA has been successfully applied to some problems. Here, based on the initial model given by the user, the program creates six different parents, which are recombined until they have generated twelve offspring models.

If the CMA-flag is set, adaption of mutation-distribution to the given quality function is performed. Only for very simple problems (few parameters to optimize), CMA lowers convergence speed slightly and may be switched off.

2.2.2 Genetic Algorithms (GA)

The principle idea of *genetic algorithms* is also recombination, mutation and selection. They where first suggested by Holland (1992) and have often been applied to geophysical model optimization. Algorithmic parameter settings, suggested in a variety of publications, have been implemented in the program.

While ES emphasizes the mutation of parameters, GA's focus more on recombination. ES shows good performance in finding an optimum very precisely, whereas ES has (probably) less ability for global optimization than GA's. However, hybrid applications, where ES and GA are applied to a problem simultaneously, are possible and have been successfully applied to geophysical models. In addition, most GA implementations transform the problem to a binary representation internally, whereas ES's does not. Most ES's take *the* best solution found in a generation and generate offspring base on this position in parameter space whereas GA's work with a rather large "group" of model individuals. For detailed descriptions of many different forms of GA see Goldberg (1989) and Schöneburg et al. (1994).

Customizable parameters are:

- population size,
- probability of mutations (Mp),
- probability of crossover (Cp),
- length of bitstrings (responsible resolution) and
- base of coding system (default: binary).

Figure 2.4 shows where to change these parameters.

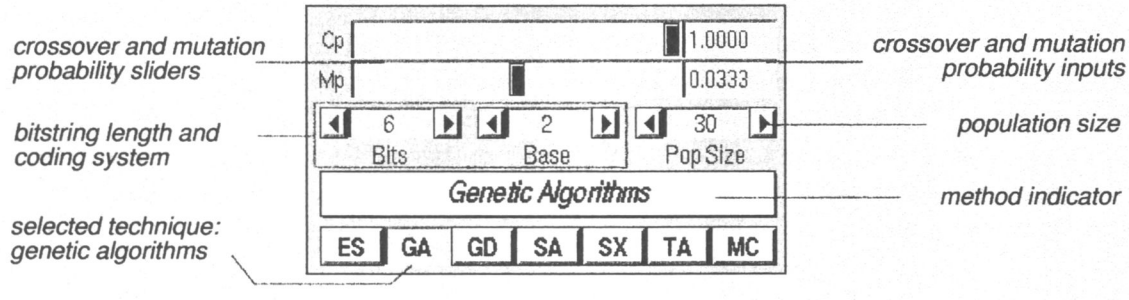


Figure 2.4: Parameter settings for genetic algorithms .

Note that constraints have to be defined prior to running genetic algorithms! Without constraints, the method does not work.

Some considerations for choosing parameters can be given. In general, the larger the population, the more individuals have to be calculated per generation. If CPU power is "unlimited", population sizes > 30 can be used.

Because mutation contributes to the heterogeneity of the population, do not choose mutation probability < 0.001. As a rule, the smaller the mutation probability, the higher the danger of premature convergence. If mutation probability is one, every bit is flipped. If mutation probability is zero, progress is gained by recombination only. This may also cause premature convergence. Mutation probability is a sensitive algorithmic parameter in terms of convergence characteristic of GA and should be changed with care.

The crossover-probability decides if two randomly chosen [2] individuals are recombined or not. Probability for crossover is a relatively noncritical parameter. Evidently, if both, mutation and crossover probability, are set to zero, no progress at all can be expected. If mutation probability is set to one and crossover probability is set to zero, variation is generated by mutation only.

The numerical resolution depends on the length of the bitstrings. For example, with a bitstring length of one, only two states can be realized: minimum (0) or maximum (1) of constraints. In general, the longer the string, the better the resolution and the longer optimization takes.

The numerative system used for the encoding can be customized as well. As default, most GA applications use the binary system (base 2). For gravimetrical problems, the binary system often gives not the best results in terms of convergence speed. The septal (base 7) and quartal (base 4) systems improve convergence speed.

2.2.3 Simulated annealing (SA)

Simulated annealing is an optimization technique based on ideas of statistical physics and was first suggested by Kirkpatrick et al. (1983). Since then, SA has become a general purpose technique for most of all combinatorial optimization problems.

In difference to the *hill climbing* algorithms (HC), SA accepts models of worse quality with a given probability. HC accept only improvements of quality and underlies therefore the danger of premature convergence on local optima. Therefore, SA is capable of escaping from sub-optima and can perform a (more or less) global search. The probability, of a worse model beeing accepted, depends on two algorithmic parameters:

- a temporary temperature given by a so called *annealing schedule* and
- the quality regression made in the last iteration.

The *annealing schedule* has to be defined and adjusted prior to optimization ibeeing started. It "freezes" the probability if worse models are accepting while the process is running. Because results depend strongly on this *schedule*, it has to be carefully "crafted". For a detailed explanation, see e.g. Kirkpatrick et al. (1983). In DARWIN++ the necessary adjustments are done automatically.

The user has control over the following parameters:

- initial temperature and
- initial step-size (variation of model parameters).

Because it is quite difficult to automatically set these parameters, they should be separately adjusted for each optimization. No general suggestion of how to set those parameters can

[2] Individuals are not chosen solely by random but also by their fitness (quality refering to e.g. gravimetrical fit). Individuals with good quality are more likely to be selected.

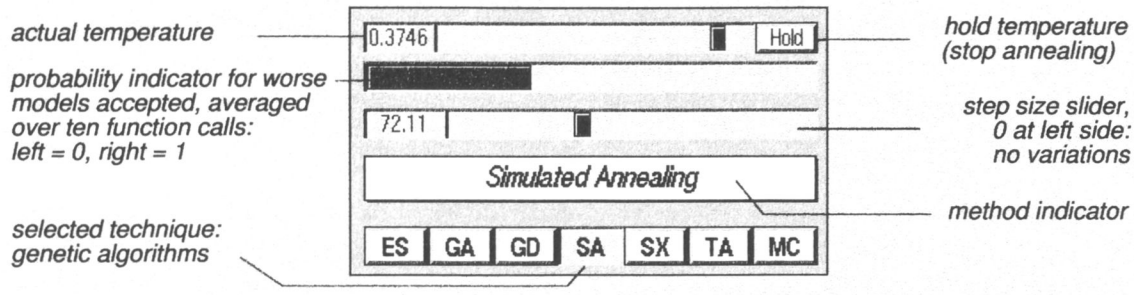


Figure 2.5: Simulated annealing parameter settings.

be given and they have to be set by intuition and experience. As a rule: if step-size control is set to zero, no variations are performed. Very small values in step-sizes cause very slow progress. If set to maximum, the program calculates variations over the entire range of constraints. As stated earlier, program defaults work for most applications.

If temperature is set too high, worse models are accepted too often and the algorithm tends to *random walk*. If temperature settings are too low, the algorithm performs more or less *hill climbing*, since only better solutions are accepted. In general, if CPU power is "unlimited", initial temperature may be chosen higher than the automatically calculated default.

2.2.4 Downhill Simplex (SX)

Simplex is a robust optimization algorithm which gives good results if few parameters (in the order of ten) have to be optimized. SX is based on the idea of a *simplex moving through parameter space* by reflecting the worst corner, and was first suggested by Nelder und Mead (1965). In two dimensions a simplex is a triangle, and in three dimensions it is a tetrahedron. The simplex adopts it's shape to the local topography of the quality function. It is important to note that simplex may go downhill only (if searching a minimum) and therefore has no capability to escape from local optima.

Simplex is not suggested for optimizing more than ten parameters at the same time. Note that from the viewpoint of optimization each vertex in two dimensions has two parameters: it's x and z coordinates. Therefore, not more than five vertices should be optimized at once with this method. For a detailed description of this algorithm refer to Press et al. (1992).

In DARWIN++ the following parameters can be changed:

- reflection coefficient,
- expansion coefficient and
- contraction coefficient.

Figure 2.6 shows where to change those parameters.

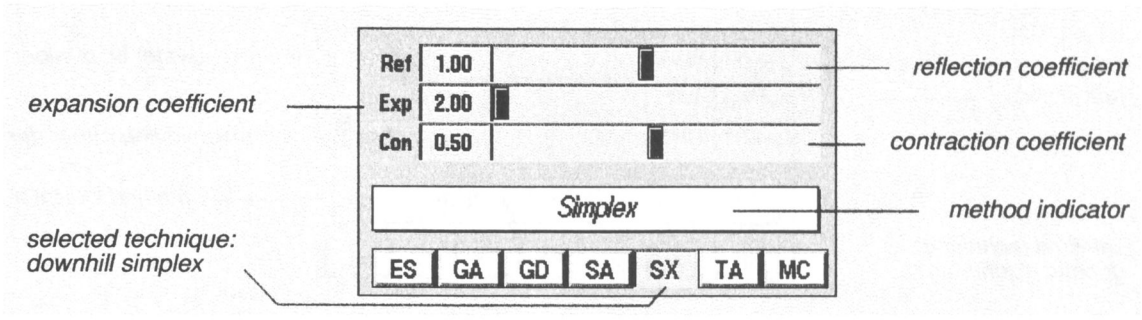


Figure 2.6: Downhill Simplex parameter settings.

All customizable parameters are rather noncritical and may be set within the limits provided by the program. For some problems convergence speed is improved if the reflection coefficient is set greater than the default.

2.2.5 Great Deluge Algorithm (GD)

The idea behind the *great deluge algorithm* for maximum search is as follows: imagine that water floods a landscape that may be interpreted as a quality function. While the water level rises, more and more islands will be left. If it does not stop raining, the water level keeps rising and at one point only *one* island will be left over: the global maximum.

GD varies model parameters and accepts the models, if their qualities lie above the (theoretical) water level. The water level is raised by the algorithmic parameter *rain* if an improvement of quality is encountered. Therefore, accepted model qualities lie on "islands" above the water-level. For a detailed description refer to Dueck (1993).

Note that for gravimetrical optimization, we are looking for a minimum: the best fit of measured and calculated data. The analogy of a rising water level has to be converted into "lowering a ceiling". Then only solutions below this (theoretical) ceiling are accepted. To convertion of the image is acceptable, because a maximum search can easily be changed to a minimum search by multiplying the quality function with the factor -1.

In the following, GD's settings are described for a **minimum search**. In DARWIN++ the following algorithmic parameters may be varied:

- initial water level,
- parameter rain and
- rate of precipitation.

Figure 2.7 shows where to change those parameters. general, if the *water level* is set to the right side of the water level slider, all solutions below this (quality) value are accepted. The default is set to the initial quality plus ten percent. With this setting, almost a

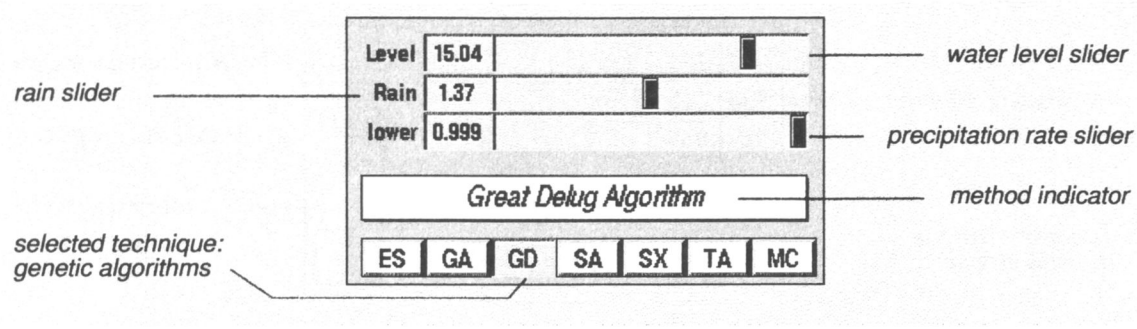


Figure 2.7: Great deluge algorithm parameter settings.

random walk through the parameter space is performed and therefore little optimization takes place. It is impossible by definition and blocked by the program to set the *water level* to a value lower than the actual quality is.

If the parameter *rain* is set to zero, the *water level* is not lowered [3] at all and probably very small progress will be observed. If *rain* is set too large (e.g. so that *water level* is lower than the actual quality), the *water level* is automatically set *to* the value of the actual quality. In this case the algorithm works like *hill climbing* because only solutions better (lower) than the *water level* are accepted.

The rate of precipitation controls *how much it rains* as a function of time (function calls). For successful steps, rain is multiplied with this factor. In general: the more function calls, the less it rains. This causes a "fine tuning" of the water level towards the end. The parameter may be set between zero and 0.999.

As a rule, if CPU time is "unlimited", starting with a high *water level* and a small value of *rain* gives reliable results.

2.2.6 Threshold Accepting (TA)

The *threshold accepting* algorithm is based on ideas of *simulated annealing*, but the parameter settings are easier to handle. TA also accepts regression in quality and is therefore capable of escaping from sub-optima. TA also accepts solutions of inferior quality which are "not much worse" than the best found so far. The parameter of possible regression is decreasing while optimization is running. DARWIN++ handles this diminishment automatically based on experience. For a detailed description of the algorithm refer to Dueck und Scheuer (1990).

The following strategy parameters can be customized in DARWIN++:

- initial value of possible quality regression,
- rate to lower threshold.

Figure 2.8 shows where to change those parameters.

[3] in case of searching a maximum, *raised*

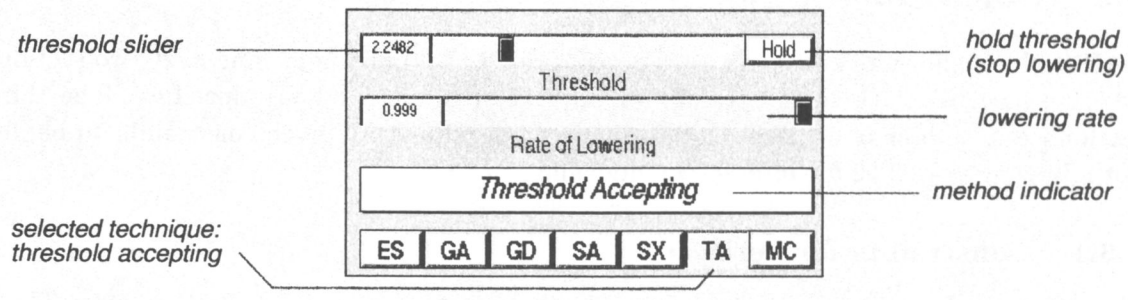


Figure 2.8: Threshold Accepting parameter settings.

The user may also stop the automated lowering of threshold and/or set the value interactively. For example, a higher value allows more regression and therefore (much) worse models will be accepted. This allows to "melt" models created by the algorithm. It can be tested, if the same solution is found again.

The algorithms' performance strongly depends on parameter settings. It is not easy to set these parameters and therefore no general suggestion can be given. It is important to note, if *threshold* is set to zero, the algorithm performs exactly like *hill climbing* because only improvements in quality are accepted. This does not indicate that optimization is terminated, but the capability of escaping from sub-optimal solutions is lost.

As a rule, if CPU time is "unlimited" the lowering rate may be set to 0.999 and the initial threshold may be set to larger values than the program's default (which is 10% of the quality at the beginning). With these parameter settings, almost a *random walk* through parameter space is performed. The "cooling" to *hill climbing* is done very slowly. That way, best results can be expected.

2.2.7 Monte Carlo (MC)

Last but not least, the well known *monte carlo* algorithm is available. MC generates realizations over the entire range of constraints. There is no mechanism to lower or raise step sizes and the parameter space sampling is equally distributed. If a model is found that is better than the one found so far, it is stored and the search continues. It is important to note that each realization is absolutely independent from the one before. Therefore, there is practically no danger of premature convergence.

In high dimensional parameter space the probability to find a solution better than the one found so far, diminishes dramatically with increasing number of function calls. This results in an extreme slowing down of convergence after some hundert iterations. Therefore the use of the MC algorithm is suggested only to get an overview of optimization behaviour. It might be used to "hunt for trends or tendencies" in parameter space.

Note that the *monte carlo* algorithm runs only if constraints are given. MC has no customizable parameters.

2.3 Constraints

DARWIN++ allows to constrain parameters prior to optimization. The next two sections describe how to set these constraints for both vertices and/or body densities. The third section explains how to achieve relations between densities. All given constraints, apply for optimization as well as for interactive modelling.

2.3.1 Constraints for vertices

The freedom of vertices can be constrained by upper, lower, left and right bounds. These bounds can be set in two ways:

(1) by mouse action and

(2) by using the vertices constraints gizmo.

The Figure 2.9 shows the vertices constraints gizmo.

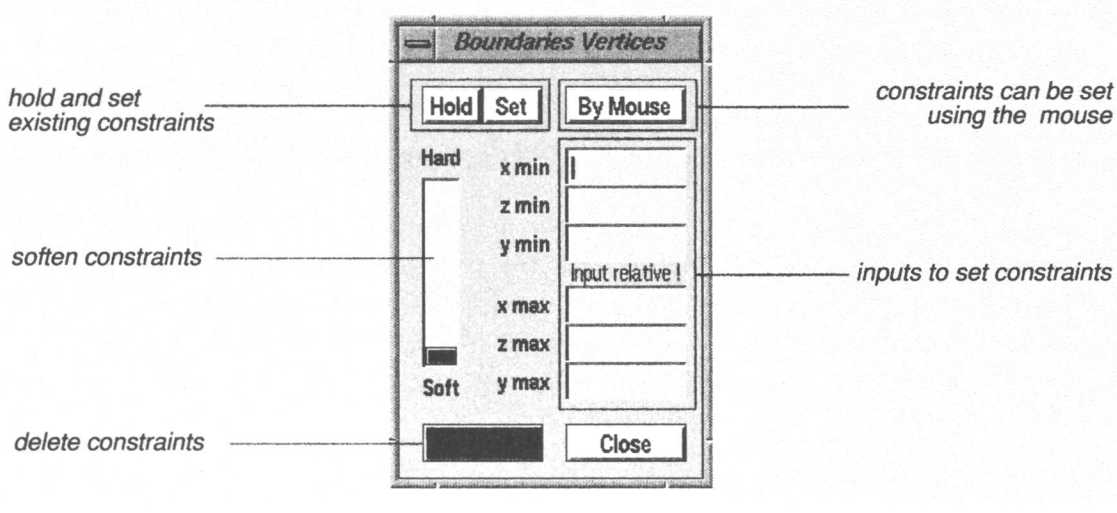


Figure 2.9: Constraints for vertices.

Set constraints by mouse

(1) Click the "By Mouse" button on the vertices constraints gizmo. The button is now set and appears in red.

(2) Hold down left mouse button on the main window and sweep out a box around the vertex you want to constrain. A blue box appears. The "By Mouse" button will be set back to normal.

(3) To correct inappropriate constraints, repeat the procedure or use the numerical input fields in the gizmo.

Set constraints by numerical values

To set constraints by using the numerical input fields, select the vertex by clicking on it with the left mouse button. Existing constraints appear now on the gizmo. If no constraints exist, all inputs are set to zero. You may now enter preferred values which are interpreted in relation to the vertex. For example, if a vertex has the x-coordinate of 5 and xmin is set to -1, xmax is set to $+1$ the vertex may move between 4 and 6.

Fuzzy constraints

You may change the "hardness" of given constraints with a slider. If set to "hard", constraints are always respected and the vertex may not leave the box. If set to "soft", the vertex may leave the box with a probability value of 0.5. This enables you to make bounds fuzzy if no precise information is available.

Delete constraints

To delete constraints, select the vertex whose constraints are to be deleted and click the "Delete" button on the vertices constraints gizmo.

Hold and set geometrical constraints

For convenience it is possible to copy existing constraints to other vertices. To copy do this:

(1) Select the vertex whose constraints you want to copy.

(2) Click the "Hold" button on the vertices constraints gizmo.

(3) Select now the vertices you want to give the stored constraints and click the "Set" button.

All chosen vertices should now have the same boxes. To change given constraints for certain vertices, use the input fields described above.

2.3.2 Constraints for physical parameters

Also physical parameters (e.g. densities) can be constrained. To constrain a body's parameter range do this:

(1) Select the body to be constrained by clicking on it with the left mouse button. Existing bounds appear now on the corresponding input fields on the bodies constraints gizmo (see figure 2.10).

(2) Use now inputs and/or sliders to set or change the upper and lower limit.

Fuzzy constraints

You may change the "hardness" of a given parameter range with a slider. If set to "hard", constraints are always respected and the parameter may not leave the given range. If set

to "soft" the parameter may leave the range with probability 0.5. This enables you to make bounds fuzzy if no certain information is available.

Delete parameter constraints

To delete constraints, select the body whose constraints are to be deleted and click the "Delete" button on the bodies constraints gizmo.

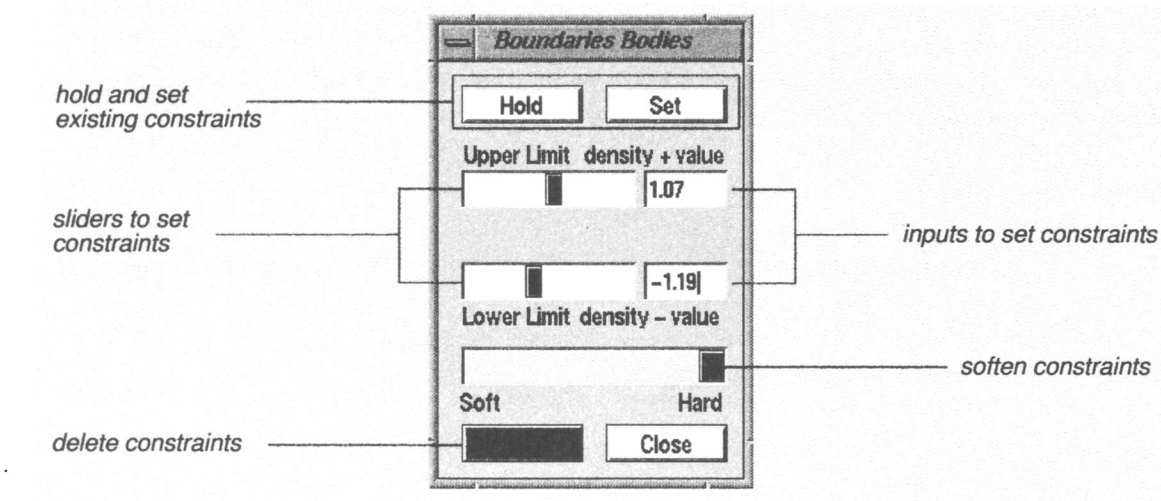


Figure 2.10: Constraints for densities.

Hold and set parameter constraints

For convenience, it is possible to copy existing constraints to other bodies. To copy do this:

(1) Select the body whose constraints you want to copy.

(2) Click the "Hold" button on the bodies constraints gizmo.

(3) Select now the body you want to give the stored constraints and click the "Set" button.

All chosen bodies should now have the same parameter ranges. To modify given constraints for certain bodies, use the input fields and/or sliders on the bodies constraints gizmo. Parameter constraints are respected for optimization as well as for interactive modelling.

2.3.3 Relations between the body's parameters

An important type of constraints are relations between the body's parameters. For example, in a sedimentary basin, almost certainly the density increases with increasing depth. With the bodies relations gizmo (see Figure 2.11) this kind of relations can be set. To set relations do this:

(1) Hold down the shift key and select two bodies by clicking them with the left mouse button. The bodie's ids are displayed on the bodies relations gizmo (on figure 2.11 7 and 2).

(2) Choose the relation type between these bodies by selecting $<$, $=$, or $>$.

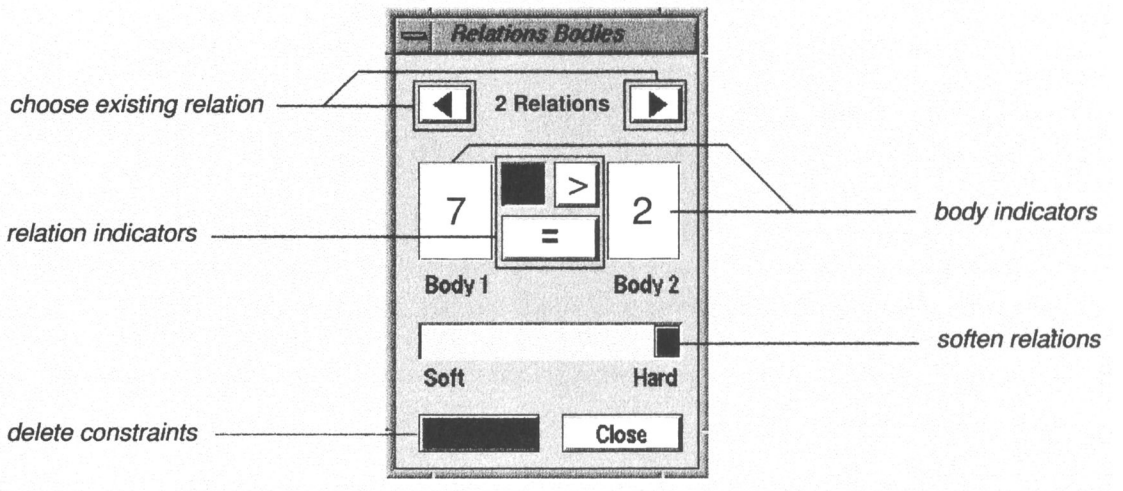


Figure 2.11: Constraints for densities.

Fuzzy relations

You may change the "hardness" of a given relation with a slider. If set to "hard", relations are always respected. If set to "soft" the relation is respected with a probability value of 0.5. This enables you to make relations fuzzy if no certain information is available.

Delete relations

To delete constraints, select a relation with the relations choice buttons on the bodies relations gizmo and click the "Delete" button.

Given relations are respected for both optimization and interactive modelling.

Literaturverzeichnis

ALVERS, M. R. (1990). *Optimierung gravimetrischer Modelle mit der Evolutionsstrategie.* Diplomarbeit, Freie Universität Berlin.

ALVERS, M. R., GÖTZE, H.-J. UND BARRIO, L. (1995). *Joint Inversion of Gravimetric and Magnetic Models Under Fuzzy Constraints.* In: Proceedings of the Conference of the International Union of Geodesy and Geophysics (IUGG), Boulder, U.S.A.

BARLEBEN, T. (1989). *Anwendung von Evolutionsstrategien zur Inversion geoelektrischer Daten.* Diplomarbeit, Technische Universität Berlin.

BARNETT, C. T. (1976). *Theoretical Modeling of the Magnetic and Gravitational Fields of an Arbitrarily Shaped Three-Dimensional Body.* Geophysics, **41 No 6**, pp. 1353–1364.

BLÜMECKE, T. (1991). *Wunder der Evolution.* c't Magazin für Computertechnik, **12**, pp. 228–239.

BOSCHETTI, F., DENTITH, M. UND LIST, R. (1996). *Inversion of Seismic Refraction Data Using Genetic Algorithms.* Geophysics, **61 No 6**, pp. 1715–1727.

BOSCHETTI, F., DENTITH, M. UND LIST, R. (1997). *Inversion of Gravity and Magnetic Data by Genetic Algorithm.* Geophysical Prospecting, **45 No 3**, pp. 461–479.

BOTT, M. H. P. (1960). *The use of Digital Computing Methods for Gravity Interpretation of Sedimentary Basins.* Geophysical Journal of the Royal Astronomical Society, **3 No1.**

BUSBY, J. P. (1987). *An Interactive Fortran 77 Program using GKS Graphics for 2.5 D Modelling of Gravity and Magnetic Data.* Computers & Geosciences, **13 No 6**, pp. 639–644.

CORBATO, C. E. (1965). *A Least Squares Procedure for Gravity Interpretation.* Geophysics, **30**, pp. 63–70.

CORDELL, L. UND HENDERSON, R. G. (1968). *Interactive Three-dimensional Solution of Gravity Anomaly Data Using a Digital Computer.* Geophysics, **33 No 4**, pp. 596–601.

DIRKS, V. (1995). *Travel-Time Curve Inversion for Transversely Isotropic Media with the Help of an Evolution Strategy.* In: Proceedings of the EAGE 57th Conference an Technical Exhibition, P024.

DUECK, G. (1993). *New Optimization Heuristics: The Great Deluge Algorithm and Record-to-Record Travel.* Journal of Computational Physics, **104**, pp. 86–92.

DUECK, G. UND SCHEUER, T. (1990). *Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing.* Journal of Computational Physics, **90**, pp. 161–175.

FOLEY, J. D., VAN DAM, A., FEINER, S. K. UND HUGHES, J. F. (1993). *Computer Graphics.* Addison-Wesley, Reading, Massachusetts, U.S.A., second edition.

GOLDBERG, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, Massachusetts, U.S.A.

GÖTZE, H.-J. (1976). *Ein numerisches Verfahren zur Berechnung der gravimetrischen und magnetischen Feldgrößen für dreidimensionale Modellkörper.* Dissertation, Technische Universität Clausthal.

GÖTZE, H.-J. (1984). *Über den Einsatz interaktiver Computergraphik im Rahmen 3-dimensionaler Interpretationstechniken in Gravimetrie und Magnetic.* Habilitationsschrift, Technische Universität Clausthal.

GÖTZE, H.-J. UND LAHMEYER, B. (1988). *Application of Three-Dimensional Interactive Modeling in Gravity and Magnetics.* Geophysics, **53 No 8**, pp. 1096–1108.

GRANT, F. S. UND WEST, G. F. (1965). *Interpretation Theory in Applied Geophysics.* McGraw Book Co., New York.

GREUL, O. UND KANDER, H. (1990). *Komplexe Funktionen und konforme Abbildungen.* BSB B. G. Teubner Verlagsgesellschaft, Leipzig.

GUILLEMONT, J. (1971). *Geologia del Petroleo.* Parafino, Madrid.

HANSEN, N. UND OSTERMEIER, A. (1996). *Adapting Arbitrary Normal Mutation Distributions in Evolution strategies: The Covariance Matrix Adaption.* In: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC '96), pp. 312–317.

HANSEN, N. UND OSTERMEIER, A. (1997). *Convergence Properties of Evolution Strategies with the Derandomized Covariance Matrix Adaption: The* $(\mu/\mu_I, \lambda) - CMA - ES$. In: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC '97).

HANSEN, N., OSTERMEIER, A. UND GAWELCZYK, A. (1995). *On the Adaption of Arbitrary Normal Mutation Distribution in Evolution Startegies: The Generation Set Adaption.* In: Proceedings of the Sixth Internationa Conference on Genetic Algorithms, pp. 57–64. Morgan Kaufman.

HERDY, M. (1996). *Evolutionsstrategie I - Grundlegende Methodik und Anwendung.* In: Konnektismus und neuronale Netze, Beiträge zur Herbstschule HeKoNN 96, pp. 140–156. GMD-Studien No 300.

HOFBAUER, J. UND SIGMUND, K. (1984). *Evolutionstheorie und dynamische Systeme - Mathematische Aspekte der Selektion.* P. Parey, Berlin und Hamburg.

HOFF, P. UND MIRAM, W. (1987). *Evolution.* Schroedel, Hannover, fifth edition.

HOLLAND, J. H. (1992). *Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence.* MIT Press / Bradford Books, Cambridge, Massachusetts U.S.A., second edition. first ed. 1975.

HOLSTEIN, H. UND KETTERIDGE, B. (1996). *Gravimetric Analysis of Uniform Polyhedra.* Geophysics, **61 No 2**, pp. 357–364.

IVANOV, B. K. (1955). *Integralgleichung für die Inversion für das logarithmische Potential.* Doklady Akademi Nauk, **105**.

JOHNSONBAUGH, R. UND KALIN, M. (1995). *Object-oriented Programming in C++.* Prentice Hall, Englewood Cliffs.

KENNEDY, W. J. UND GENTLE, J. E. (1989). *Statistical Computing.* Marcel Dekker, Inc., New York, Basel.

KIRKPATRICK, S., GELATT, C. D. UND VECCHI, M. P. (1983). *Optimization by Simulated Annealing.* Science, **220**, pp. 671–680.

KREYSZIG, E. (1993). *Advanced Engeneering Mathematics.* John Wiley & Sons, Inc., New York.

LILLESAND, T. M. UND KIEFER, R. W. (1994). *Remote Sensing and Image Interpretation.* John Wiley & Sons, Inc., New York, third edition.

LIPPMAN, S. B. (1993). *C++ Primer.* Addison-Wesley, Reading, Massachusetts, U.S.A., second edition.

MADER, F. (1959). *Das Newtonsche Raumpotential prismatischer Körper und seine Ableitungen bis zur dritten Ordnung.* Österreichische Zeitung für Vermessungswesen, **Sonderheft 11.**

METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, M. UND TELLER, A. (1953). *Equations of state calculations by fast computing machines.* Geophysical Chemical Physics, **21**, pp. 1087–1092.

NEIDER, J., DAVIS, T. UND WOO, M. (1993). *OpenGL Programming Guide.* Addison-Wesley, Reading, Massachusetts, U.S.A.

NELDER, J. A. UND MEAD, R. (1965). *A Simplex Method for Function Minimization.* Computer Journal, **7 No 4**, pp. 308–313.

NIEMEYER, G. (1989). *Einführung in das Programmieren in Assembler.* de Gruyter, Berlin, New York, sixth edition.

OKABE, M. (1979). *Analytical Expressions for Gravity Anomalies due to homogeneous polyhedral bodies and translations into magnetic anomalies.* Geophysics, **44**, pp. 730–741.

PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T. UND FLANNERY, B. P. (1992). *Numerical Recipes in C.* Cambridge University Press, Cambridge, second edition.

RASMUSSEN, R. UND PEDERSEN, L. B. (1979). *End Correction in Potential Field Modeling.* Geophysical Prospecting, **27**, pp. 749–760.

RECHENBERG, I. (1973). *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart-Bad Cannstatt.

RECHENBERG, I. (1994). *Evolutionsstrategie '94.* Frommann-Holzboog, Stuttgart-Bad Cannstatt.

ROTENBERG, A. (1960). *A new Pseudo-Random Number Generator.* Geophysical Journal International, **7**, pp. 75–77.

SAMBRIDG, M. UND DRIJKONINGEN, G. (1992). *Genetic Algorithms in Waveform Inversion.* Journal of the Assosiation for Computer Machinery, pp. 323–342.

SCHÖNEBURG, E., HEINZMANN, F. UND FEDDERSEN, S. (1994). *Genetische Algorithmen und Evolutionsstrategie.* Addison-Wesley, Reading, Massachusetts, U.S.A.

SCHWEFEL, H.-P. (1977). *Numerische Optimierung von Computermodellen mit der Evolutionsstrategie.* Birkenhäuser, Basel und Stuttgart.

SCHWEFEL, H.-P. (1995). *Evolution and Optimum seeking.* John Wiley & Sons, New York.

SEDLAG, U. UND WEINERT, E. (1987). *Biogeographie, Artenbildung, Evolution.* Gustav Fischer Verlag, Jena.

SHUEY, R. T. UND PASQUALE, A. S. (1973). *End Correction in Magnetic Profile Interpretation.* Geophysics, **38 No 3**, pp. 507–512.

SMIRNOW, W. I. (1965). *Lehrgang der höheren Mathematik.* Deutscher Verlag der Wissenschaften, Berlin.

STOFFA, P. L. UND SEN, M. K. (1994). *Nonlinear Multiparameter Optimization Using Genetic Algorithms: Inversion of Plane-Wave Seismograms.* Geophysics, **56 No 11**, pp. 1794–1810.

TALWANI, M. UND EWING, M. (1960). *Rapid Computation of Gravitational Attraction of Three-Dimensional Bodies of Arbitrary shape.* Geophysics, **25 No 1**, pp. 203–225.

TALWANI, M., WORZEL, J. L. UND LANDISMAN, M. (1959). *Rapid Gravity Computation for two-dimensional Bodies with Application to the Mendocino submarin fracture zone.* Journal of Geophysical Research, **64**.

TAUSWORTH, R. (1965). *Random Numbers generated by linear recurrence modulo two.* Mathematics of Computation, **19**, pp. 201–209.

TELFORD, W. M. (1990). *Applied Geophysics.* George Allen & Unwin, Cambridge University Press, second edition.

TRAISTER, R. J. (1993). *Going from C to C++.* Academic Press Professional, Boston.

TSUBOI, C. (1983). *Gravity.* George Allen & Unwin, London.

WHITLEY, D., MATHIAS, K. UND DZUBERA, J. (1995). *Building Better Test Funktions.* In: Eshelman (Hrsg.), *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 239–246, San Francisco, U.S.A. Morgan Kaufmann.

WOITSCHACH, M. (1978). *Läst sich der Zufall rechnen? Nutzen und Grenzen der Wahrscheinlichkeitsrechnung.* Kosmos Bibliothek, Stuttgart.

WON, I. J. UND BEVIS, M. (1987). *Computing the Gravitational and Magnetic Anomalies due to a Polygon: Algorithms and Fortran Subroutines.* Geophysics, **52 No 2**, pp. 232–238.

WOODWARD, D. J. (1975). *The Gravitational Attraction of Vertical Triangular Prisms.* Geophysical Prospecting, **23**, pp. 526–532.

Danksagung

Zuerst möchte ich mich herzlich bei meinem Doktorvater H.-J.Götze bedanken, der mir einen Gastaufenthalt an der Stanford University in Kalifornien in der Gruppe von J. Harbaugh ermöglichte. Der einjährige Aufenthalt hat mein geophysikalisches Verständnis stark geprägt und mir viele Ideen für diese Arbeit gebracht.

Außerdem danke ich ihm und S. Schmidt für viele interessante Gespräche über objektorientierte Konzepte und über die Zukunft der Geowissenschaften. Besonders habe ich eine Diskussion in Erinnerung, welche in Argentinien am Rande eines Computerkurses stattfand, die mir klar machte, welche *Objekte* für die Modellierung geophysikalischer Modelle sinnvoll sind. [1]

Auch I. Rechenberg gebührt Dank, da er doch das Projekt EVOTECH initiierte. Außerdem danke ich ihm für die Erstellung des Zweitgutachtens für diese Arbeit.

Der Firma BEB, vertreten durch Ch. Schweitzer, danke ich für die finanzielle Unterstützung des Projektes EVOTECH über zwei Jahre. Ich danke BEB auch für die Möglichkeit, das Programm DARWIN++, welches im Rahmen dieses Projektes entstanden ist, auf ein industrierelevantes Problem anwenden und testen zu können.

Besonderer Dank gilt allen DARWIN++ Nutzern, die durch konstruktive Kritik, Vorschläge und intensives *β-testing* mit dazu beigetragen haben, das Programm zu einem hilfreichen Modellierungs*tool* zu machen. Besonders möchte ich hier meine Frau Liliana, A. Kirchner, J. Döring, M. Kösters und H. Kebbal-Levin hervorheben.

Für die Möglichkeit der Zusammenarbeit in einem außerordentlich angenehmen Arbeitsklima danke ich allen Mitgliedern der Arbeitsgruppe Gravimetrie und im besonderen G. Goltz für seine professionelle Hilfe in allen technischen Fragen.

Herzlichen Dank auch an H.-J. Götze, E. Kösters, F. Eckhardt, K. Brandler, M. Lühl und meine Frau für die überaus aufwendige Durchsicht des Manuskripts dieser Arbeit.

Last but not least gilt mein Dank dem deutschen Steuerzahler, der das Projekt EVOTECH finanzierte und mir dadurch die Möglichkeit freier wissenschaftlicher Arbeit inklusive Dienstreisen und wissenschaftlicher Geräte ermöglichte. Dank auch den verwaltenden Institutionen BMBF und Freie Universität Berlin.

[1] Ich danke den beiden auch, daß sie mich morgens auf mein Zimmer gebracht haben.

BERLINER GEOWISSENSCHAFTLICHE ABHANDLUNGEN
Freie Universität Berlin Technische Universität Berlin Technische Fachhochschule Berlin

Noch erhältlich sind:

Reihe A: Geologie und Paläontologie

Band 177* / Eckart Schrank & Mohamed I. A. Ibrahim: Cretaceous (Aptian-Maastrichtian) palynology of foraminifera-dated wells (KRM-1, AG-18) in northwestern Egypt. 44 pp., 10 text-figs., 3 tabs., 9 pls., 1995. DM 42,–.

Band 178* / Mumin Mohamud God: Climatic conditions deduced from Late Pleistocene deposits at Karin gap (NE-Somalia). 145 pp., 47 text-figs., 20 tabs., 1995. DM 53,–.

Band 179 / Sun Yong Bin: Radiogene Isotopengeochemie des Troodos-Ophioliths. Untersuchungen (Haupt- und Spurenelemente sowie Sr-, Nd- und Pb-Isotope) der zyprischen Ozeankruste. IX + 137 S., 44 Abb., 23 Tab., 1995. DM 54,–.

Band 180 / Sabine Dietrich: Strukturierung Eisen-Mangan-haltiger Schlämme durch komplexe dynamische Prozesse. Ein Beitrag zur frühdiagenetischen Bildung von Mineralgefügen durch Selbstorganisation. 86 S., 25 Abb., 5 Tab., 1996. DM 52,–.

Band 181 / Gabriel González López:** Evolución Tectónica de la Cordillera de la Costa de Antofagasta (Chile): Con especial referencia a las deformaciones sinmagmáticas del Jurásico-Cretácico Inferior. IX + 111 pp., 47 figs., 12 tabs., 1996. DM 53,–.

Band 182 / Stephan Hannappel: Die Beschaffenheit des Grundwassers in den hydrogeologischen Strukturen der neuen Bundesländer. 151 S., 93 Abb., 29 Tab., 1996. DM 54,–.

Band 183 / Jörg Hettler, Bernd Lehmann & Luis LeMarie Ch.: Environmental Problems of Petroleum Production in the Amazon Lowland of Ecuador. 71 pp., 39 figs., 12 tabs., appendices 1–3, 1996. DM 32,–.

Band 184 / Christian-Peter Kisten: Mineralogische und geochemische Untersuchungen zur Beständigkeit tonmineralischer Dichtmassen. Fallbeispiele: Geschiebemergel/hochplastische Tone. VIII + 81 S., 48 Abb., 24 Tab., 1996. DM 49,–.

Band 185 / Jürgen Adam: Kinematik und Dynamik des neogenen Falten- und Deckengürtels in Sizilien. Quantifizierung neotektonischer Deformationsprozesse in der zentralmediterranen Afro-Europäischen Konvergenzzone. 171 S., 56 Abb., 6 Tab., 3 Taf., 1996. DM 62,–.

Band 186* / Petr Vrbka: Hydrogeologische und isotopenhydrologische Untersuchungen zu regionalen Problemen der GW-Neubildung, der GW-Zirkulation und des Wasserhaushaltes im Nordsudan. 158 S., zahlr. Abb. und Tab., 1996. DM 57,–.

Band 187 / Julia-Francisca Schüle: Untersuchungen an deutschen und spanischen Böden zu Sorptions- und Abbauprozessen der Pflanzenschutzmittel Terbuthylazin und Parathion-Ethyl. XI + 77 S., 28 Abb., 31 Tab., 1996. DM 43,–.

Band 188 / Harm Schultz: Analyse der variszisch-apenninischen Deformationsgeschichte des paläozoischen Basements der Apuaner Alpen (Toskana, Italien). 108 S., 50 Abb., 10 Taf., Anhang A–F, 1996. DM 66,–.

Band 189 / Mathias Dörr:** Der Arcas-Fächer und sein Erosionsgebiet in Nordchile: Bilanz einer exogenen Massenverlagerung. 121 S., 35 Abb., 12 Tab., 5 Taf., 1996. DM 66,–.

Band 190 / Andreas Uhlig: Untersuchung der finiten und progressiven Gesteinsdeformation in den nördlichen Apuaner Alpen (Nord-Toskana/Italien). VIII + 104 S., 40 Abb., 34 Tab., 5 Taf. DM 75,–.

Band 191 / Christian Götz Hirschberg: Die Gerölle von Werra und Weser – ihre strukturelle und kompositionelle Reifung in der Natur und im Experiment. VIII + 114 S., 23 Abb., 31 + 13 Tab., 1997. DM 51,–.

Band 192 / Ernst Kiefer: Provenance und Ausbreitung von Silt am aktiven Plattenrand Kalabriens: Anwendung der Diffusions-Theorie auf Petrographie und Transport terrigener Partikel. 237 S., 49 Abb., 6 Tab., 1997. DM 89,–.

Band 193 / Raphael Wittenbrink:** Zeitliche Variationen der Magmengenese miozäner bis quartärer Vulkanite im südlichen Bereich der Zentralen Vulkanischen Zone der Anden (CVZ, 25°–26°S, 67°–69°W). VII + 135 S., 38 Abb., 5 Tab., 1997. DM 68,–.

Band 194 / Jamal Rhrib: Die Störungszonen des Mittleren Atlas (Zentralmarokko) – Strukturelle Entwicklung in einem intrakontinentalen Gebirge. Les zones des failles dans le Moyen Atlas (Maroc) leurs évolutions structurales dans une chaine intracontinentale. 221 S., 94 Abb., 3 Phototafeln, 1997. DM 78,–.

Band 195 / Sabine Thom: Numerische Strömungsmodellierung zur Wasserbilanzierung im Süden und Südosten Berlins sowie angrenzenden Teilen Brandenburgs. VII + 122 S., 55 Abb., 16 Tab., Anlagen 1–3, 1997. DM 51,–.

Band 196* / Robert Bussert: Die Entwicklung intrakratonaler Becken im Nordsudan. 329 S., 76 Abb., 35 Tab., 19 Taf., 1998. DM 119,–.

* Beiträge aus dem Sonderforschungsbereich 69: Geowissenschaftl. Probleme in ariden Gebieten
** Beiträge aus dem Sonderforschungsbereich 267: Deformationsprozesse in den Anden

BERLINER GEOWISSENSCHAFTLICHE ABHANDLUNGEN
Freie Universität Berlin · Technische Universität Berlin Technische Fachhochschule Berlin

Noch erhältlich sind:

Reihe B: Geophysik

Band 7 / Hussein El-Ali: Geothermische Modelle für ein Profil zwischen Südadria und Thyrrhenis. 51 S., 20 Abb., 1978. DM 18,–.

Band 8 / Gerhard Schwarz: Die elektrische Leitfähigkeit in der Toskana und ein daraus abgeleitetes geothermisches Modell – insbesondere für die Anomalie von Travale. 95 S., 52 Abb., 1 Tab., 1984. DM 29,–.

Band 9 / Peter J. Wigger: Die Krustenstruktur des Nordappennins und angrenzender Gebiete mit besonderer Berücksichtigung der geothermischen Anomalie der Toskana. 87 S., 69 Abb., 1984. DM 29,–.

Band 10 / Mohammed Mehdi Mostaanpour: Einheitliche Auswertung krustenseismischer Daten in Westeuropa. Darstellung von Krustenparametern und Laufzeitanomalien. 96 S., 21 Abb., 7 Tab., 1984. DM 45,–.

Band 11 / Nikolaus Klever: Stationäre Konvektion in porösen Medien – numerische Untersuchungen an unterschiedlichen Fragestellungen aus der Hydrothermik und der Schneemetamorphose. 114 S., 78 Abb., 7 Tab., 1984. DM 31,–.

Band 12 / Horst Letz: Seismizität in Irian Jaya (West-Neuguinea), Indonesien, und ihre tektonische Bedeutung. 108 S., 39 Abb., 12 Tab., 2 Kart., 1985. DM 44,–.

Band 13 / Gerhard Jentzsch: Auflastgezeiten in Fennoskandien. 184 S., 77 Abb., 36 Tab., 1986. DM 66,–.

Band 14 / Hans-Peter Plag: A Regional Study of Norwegian Coastal Long-Period Sea-Level Variations and Their Causes. 175 pp., 76 figs., 57 tbls., 1988. DM 69,–.

Band 15 / Günter Asch: Die Registrierung langperiodischer Signale mit geophysikalischen Sensoren hoher Dynamik. 87 S., 81 Abb., 8 Tab., 1988. DM 52,–.

Band 16 / Thomas Jahr: Gezeitengravimetrie in Dänemark. 137 S., 56 Abb., 30 Tab., 1989. DM 39,–.

Band 18 / Hermann Buness: Krustale Kollisionsstrukturen an den Rändern der nordwestlichen Adriaplatte. 221 S., 194 Abb., 5 Tab., 1992. DM 78,–.

Band 19/Mathias Delleske: Zur Schwerefeldseparation mittels 3D-Modellrechnungen im Hochgebirge. 74 S., 43 Abb., 17 Tab., 1993. DM 76,–.

Band 21/Detlef Krüger: Modellierungen zur Struktur elektrisch leitfähiger Zonen in den südlichen zentralen Anden. 91 S., 60 Abb., 1994. DM 67,–.

Band 22/Christian Klesper: Die rechnergestützte Modellierung eines 3D-Flächenverbandes der Erftscholle (Niederrheinische Bucht). 117 S., 51 Abb., 1994. DM 69,–.

Band 23 / Pedro R. Kress: Tectonic Inversion of the Subandean Foreland – a Combined Geophysical and Geological Approach. 127 pp., 52 figs., 5 tabs., 1995. DM 63,–.

Band 24 / Angelika-Maria Wulff: Absorptionsmechanismen bei Ultraschallwellen in fluidhaltigen Sandsteinen unter verschiedenen Druckbedingungen. 172 S., 59 Abb., 3 Tab., 1995. DM 69,–.

Band 25 / Andreas Kirchner:** 3D-Dichtemodellierung zur Anpassung des Schwere- und des Schwerepotentialfeldes der zentralen Anden. 98 S., 57 Abb., 1997. DM 57,–.

Band 26 / Georg M. Partzsch:** Elektrische Leitfähigkeit partiell geschmolzener Gesteine: Experimentelle Untersuchungen, Modellrechnungen und Interpretation einer elektrisch leitfähigen Zone in den zentralen Anden. 117 S., 47 Abb., 15 Tab., 7 Photos, Anhang, 1998. DM 63,–.

Band 27 / Alfredo Horacio Gangui:** A combined Structural Interpretation based on Seismic Data and 3-D Gravity Modeling in the Northern Puna / Eastern Cordillera, Argentina. X + 176 S., 73 figs., 4 tabs., 1998. DM 78,–.

Band 28 / Michael Alvers:** Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle. V + 108 S., 88 Abb., 1998. DM 58,–.

Band 29 / Jörn Döring: Dichteverteilung und Modellierung des isostatischen Verhaltens der Lithosphäre im Südural. IX + 136 S., 81 Abb., 1998. DM 58,–.

** Beiträge aus dem Sonderforschungsbereich 267: Deformationsprozesse in den Anden

Das vollständige Verzeichnis der lieferbaren Titel der Reihen A, B, C, D und E ist erhältlich bei:

Selbstverlag Fachbereich Geowissenschaften, FU Berlin
Malteserstraße 74–100 • D-12249 Berlin
Tel.: (0) 30-779 23 60 • Fax: (0) 30-776 20 34