Mehrkriterielle sequentielle Parameteroptimierung für Anwendungs-Probleme mit stark limitiertem Budget

Martin Zaefferer, Boris Naujoks, Thomas Bartz-Beielstein, Martina Friese, Olaf Mersmann, Oliver Flasch

Fakultät für Informatik und Ingenieurwissenschaften, FH Köln
Steinmüllerallee 1, 51643 Gummersbach
Tel.: +49 2261 8196-0
Fax: +49 2261 8196-15
E-Mail: {Vorname}.{Nachname}@fh-koeln.de

1 Einleitung

Sequentielle Parameteroptimierung (SPO, [1]) wurde bisher hauptsächlich für einkriterielle Anwendungen eingesetzt. Die sequentielle Optimierung mit Surrogatmodellen eignet sich, um bei kosten- und zeitintensiven Problemen reale Funktionsauswertungen einzusparen. In der Praxis ergeben sich häufig Problemstellungen, die nur mit wenigen Zielfunktionsauswertungen (einem geringen Budget) zu lösen sind und zusätzlich mehrere Zielkriterien aufweisen. Für diese Anwendungen wurde die SPO Toolbox SPOT für die mehrkriterielle Optimierung erweitert. Beispiele für diese Anwendungsfälle sind die Optimierung von Staubabscheidern in Kohlekraftwerken oder die Optimierung von Fehlalarmraten und Erkennungsraten zur Detektion von Anomalien in Trinkwasserdaten. Die vorliegende Fallstudie befasst sich mit bekannten Testfunktionen, um die Anwendbarkeit verschiedener Ansätze zu prüfen.

Die Integration von Verfahren zur Surrogatmodellierung in Evolutionäre Algorithmen ist bereits bekannt. Eine Übersicht für den einkriteriellen Fall liefert Jin [2], Arbeiten für den mehrkriteriellen Fall fassen Knowles und Nakayama zusammen [3]. Ein bekannter Ansatz mit Surrogatmodellen für MCO ist zum Beispiel ParEGO von Knowles [4]. Verschiedene Ansätze setzen zudem auf Expected Improvement oder andere Kriterien, die die Varianz berücksichtigen, um den Suchraum zu explorieren. Dazu gehört zum Beispiel S-Metric Expected Improvement [5], Efficient Global Optimization oder S-Metric Selection Efficient Global Optimization [6]. Ein

Original Publication:
Zaefferer, Martin, Boris Naujoks, Thomas Bartz-Beielstein, Martina Friese, Olaf Mersmann, and Oliver Flasch. "Mehrkriterielle sequentielle Parameteroptimierung für Anwendungs-Probleme mit stark limitiertem Budget." In Proceedings. 22. Workshop Computational Intelligence, Dortmund, 6.-7. Dezember 2012, p. 385. KIT Scientific Publishing, 2014.

Überblick und Vergleich diese Ansätze wird von Wagner et al. gegeben [7]. Der hier vorgestellte Ansatz unterscheidet sich in so fern, dass

i) die Toolbox zur sequentiellen Parameteroptimierung (SPOT) zum Einsatz kommt, die bereits eine Vielzahl von Modellen zur Verfügung stellt, so dass die Kombination verschiedener Methoden auf eine einfache Art und Weise möglich ist,

ii) die einzelnen Ziele separat optimiert werden (nicht aggregiert),

iii) keine Varianz berücksichtigt wird (greedy, wenig exploration),

iv) neben üblichen Varianten wie Kriging verschiedene Modelltypen einsetzbar sind und

v) besonders wenige Zielfunktionsauswertungen erlaubt werden. In der Regel werden bei den in der Literatur beschriebenen Ansätzen deutlich mehr als die hier maximalen 250 Auswertungen erlaubt.

Der in SPOT derzeit standardmäßig eingesetzte Ansatz für mehrkriterielle Probleme wird im Folgenden als MSPOT bezeichnet.

Die genauen Ziele und Fragestellungen, die hier bearbeitet werden, sind in Abschnitt 2 beschrieben. Daran anschließend werden die Methoden bzw. die Optimierer in Abschnitt 3 vorgestellt. Der Aufbau der hier betriebenen Experimente ist in Abschnitt 4 dargestellt. Die Ergebnisse dieser Experimente werden schließlich in Abschnitt 5 ausgewertet, während schließlich im Abschnitt 6 eine Zusammenfassung und ein Ausblick auf weitere Arbeiten gegeben werden.

2 Fragen und Ziele

Für diese Arbeit sind verschiedene Fragestellungen von Bedeutung.

I **Lassen sich auch bei stark beschränktem Budget mit einem Surrogatmodell Verbesserungen gegenüber einem klassischen Ansatz erreichen?**
Aus der Literatur sind zahlreiche Ergebnisse bekannt, die die Überlegenheit von Surrogatmodellen gegenüber modellfreien Verfahren belegen. Allerdings sollten die Zusatzkosten für die Berechnung der Modelle nicht vernachlässigt werden. Um diese Überlegenheit auch bei

stark beschränkten Budgets und bei relativ hohen Suchraumdimensionen nachzuweisen, wird in Experimenten MSPOT mit S-Metric Selection Evolutionary Multi-objective Optimization Algorithm (SMS-EMOA) verglichen. Da diese beiden Ansätze auf sehr unterschiedliche Weise neue Lösungen generieren, wird zur besseren Vergleichbarkeit auch KRIG-SMS-EMOA getestet. KRIG-SMS-EMOA ist ein Ansatz, der wie SMS-EMOA aufgebaut ist, aber ein Kriging Surrogatmodell verwendet. Da durch das Surrogatmodell kostspielige Lösungen heraus gefiltert werden, ist ein Vorteil für KRIG-SMS-EMOA und MSPOT zu erwarten. Dies wird auch durch bereits vorliegende Ergebnisse bestätigt [8]. Des Weiteren untersuchen wir, ob der zielgerichtete Einsatz von Surrogatmodellen in MSPOT den vergleichsweise einfachen KRIG-SMS-EMOA-Ansatz schlägt. Falls das Modell aber die Zielfunktion schlecht approximiert, könnte die stärkere Randomisierung in KRIG-SMS-EMOA diesen Vorteil wettmachen. Diese Vermutungen sollen durch den Vergleich der Ansätze auf verschiedenen Testfunktionen untersucht werden.

II **Gibt es bestimmte Umstände, die den Einsatz eines Surrogatmodells unprofitabel machen?**
Sollten die angesetzten Budgets zu klein sein, um hinreichend Daten für die Surrogatmodelle zu sammeln, wäre es möglich, dass Ansätze mit entsprechend unzureichenden Modellen vergleichbar oder sogar schlechter wie der Vergleichsansatz ohne Surrogatmodell arbeiten. Hier ist von Interesse, in welchen Fällen dies auftritt.

III **Welchen Einfluss haben unterschiedliche Typen von Surrogatmodellen?**
Für den hier getesteten Satz von Zielfunktionen sollte zudem untersucht werden, welchen Einfluss der Einsatz von unterschiedlichen Modelltypen hat. Dazu werden Kriging, Multivariate Adaptive Regression Splines (MARS) [9] und Random Forest (RF) [10] heran gezogen.

IV **Welchen Einfluss hat die Dimension des Suchraumes?**
Da die Zielfunktionen bzgl. der Suchraumdimension skalierbar ist, wird auch untersucht, welchen Einfluss dieser Parameter besitzt. Dies wird in einem ausgewählten Fall (ZDT1) mit zusätzlichen Experimenten durchgeführt.

V **Wie unterscheidet sich der Rechenaufwand der verschiedenen Ansätze?**
Um für den Anwendungsfall eine sinnvolle Empfehlung geben zu

können, ist es notwendig, neben der erreichten Güte auch den Rechenaufwand und damit die benötigte Zeit der verschiedenen Ansätze zu vergleichen.

3 Methodenbeschreibung

3.1 MSPOT

Um die Unterschiede zum einkriteriellen SPOT Ansatz zu erläutern, wird dieser zunächst vorgestellt. In einem ersten Schritt erzeugt SPOT ein initiales Design, das aus mehreren Punkten im Suchraum besteht. In diesen Punkten wird die Zielfunktion (Probleminstanz) ausgewertet. Die Ergebnisse dieser Auswertung werden verwendet, um ein Surrogatmodell zu generieren. Hierfür kann beispielsweise ein lineares Regressionsmodell oder ein Krigingmodell gewählt werden.

Dieses Surrogatmodell kann auf zwei unterschiedliche Arten benutzt werden. (i) Der naive Ansatz (`naive`) erstellt ein neues Design aus vielen Punkten und wertet sie auf dem Surrogatmodell aus. Diese werden sortiert, und der Beste (oder die Besten) werden auf der Zielfunktion ausgewertet. (ii) Ein anspruchsvollerer Ansatz (`elaborate`) nutzt state-of-the-art Optimierer, um das Optimum des Surrogatmodells zu finden. Dieses Optimum wird für weitere Auswertungen auf dem Surrogatmodell vorgeschlagen. Die sequentiellen Schritte der Auswertung auf der Zielfunktion und der Generierung und Verwendung des Surrogatmodells werden solange wiederholt bis die vorgegebene Anzahl von Zielfunktionsauswertungen (Budget) erreicht ist.

Im Pseudocode von MSPOT (cf. Alg. 1) werden zwei Phasen unterschieden. Phase 1 dient der Berechnung des initialen Designs (Zeile 1–4) und Phase 2 der sequentiellen Verbesserung (Zeile 6–16). In Phase 1 wird das initiale Design erstellt und k_0-mal auf der Zielfunktion ausgewertet. Im Gegensatz zum einkriteriellen Ansatz in SPOT ergibt sich damit nicht jeweils ein skalarer Wert, sondern ein Vektor von Zielfunktionswerten, dessen Länge der Anzahl der optimierten Ziele entspricht.

Phase 2 besteht aus der folgenden Schleife

1. Erstelle ein Modell M für jedes der n Ziele basierend auf den evaluierten Design-Punkten.
2. `Naive:` Erstelle ein (großes) Design $\vec{X}'$ aus l Designpunkten mit Latin Hypercube Sampling und berechne deren Güte $\vec{Y}'$ durch Auswertung auf den Modellen.

Algorithmus 1: MSPOT

`// Phase 1: Initialdesign:`

O ist das Optimierungsproblem mit d Entscheidungsvariablen und n zu optimierenden Zielen;

erstelle ein Initialdesign $\vec{X} := \{\vec{x}^1, \ldots, \vec{x}^m\}$ mit $\vec{x} \in \mathbb{R}^d$ bestehend aus m Punkten;

setze $k := k_0$ für die Anzahl der Auswertungen jedes Punktes;

foreach $\vec{x} \in \vec{X}$ **do**

    O mit $\vec{x}$ k-mal auswerten um Güte $\vec{y}$ von $\vec{x}$ zu bestimmen, mit $\vec{y} \in \mathbb{R}^n$;

`// Phase 2: Erstellen, Nutzen und Verbessern der Modelle:`

while Abbruchkriterium nicht wahr **do**

    Erstelle Surrogatemodelle $M_i, i \in \{1, \ldots, n\}$, basierend auf $\vec{X}$ und $\vec{Y} := \{\vec{y}_1, \ldots, \vec{y}_m\}$;

    `// Naive:`

    erstelle ein Set $\vec{X'}$ von l neuen Designpunkten mit Latin Hypercube Sampling;

    `// Elaborate:`

    erweitere $\vec{X'}$ um Pareto optimale Punkte aus Optimierer (z.B. SMS-EMOA)

    foreach $\vec{x'} \in \vec{X'}$ **do**

        bestimme vorhergesagte Güte des neuen Designs auf jedem Modell mit $\vec{y'} := M(\vec{x'})$;

    foreach $\vec{x} \in \vec{X}$ **do**

        bestimme vorhergesagte Güte des bekannten Designs $\hat{y} := M(\vec{x})$;

    wähle a Designpunkte $\vec{X''}$ aus $\vec{X'}$ basierend auf Non-Dominated Sorting Rank und Hypervolumenbeitrag ($a \ll l$) unter Berücksichtigung von $\hat{y}$;

    foreach $\vec{x''} \in \vec{X''}$ **do**

        O mit $\vec{x''}$ k-mal auswerten um geschätzte Güte $\vec{y''}$ von $\vec{x''}$ zu bestimmen;

    Erweitere das Design mit $\vec{X} := \vec{X} \cup \vec{X''}$ und $\vec{Y} := \vec{Y} \cup \vec{Y''}$;

Bestimme aus $\vec{Y}$ die Pareto-Front $\vec{P}$;

3. `Elaborate`: Wende einen state-of-the-art Optimierungsalgorithmus (z.B. SMS-EMOA) an, mit $M_i, i \in \{1, ..., n\}$ als mehrkriterielle Zielfunktion.
4. Wähle neue Lösungskandidaten $\vec{X''}$ (mit $a = |\vec{X''}|, a \ll l$) aus den Ergebnissen einer oder beider der vorigen Schritte. Hierfür werden die Kandidaten zuerst nach ihrem Rang aus dem *Non-dominated Sorting* [11] sortiert. Als Tie-Breaker wird der Hypervolumen-Beitrag verwendet. Um sicher zu stellen, dass neue Lösungen nicht zu nah an bereits bekannten Punkten im Lösungsraum liegen, werden die bereits bekannten Punkte auch auf dem Modell evaluiert und bei der Bestimmung des Hypervolumen-Beitrags berücksichtigt.

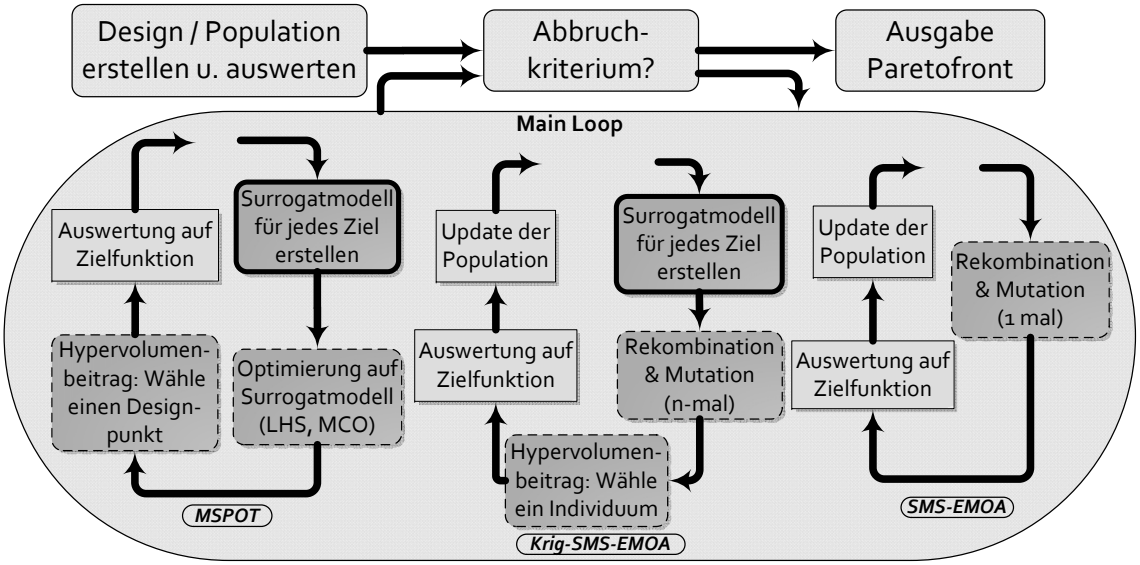

Bild 1: Diagramm der verschiedenen Optimierungsverfahren

5. Die ausgewählten Punkte $\vec{X}''$ werden zum Design hinzugefügt und auf dem Problem $\mathcal{O}$ ausgewertet. Dies wird bis zum Abbruch des Algorithmus wiederholt.

Der MSPOT Zyklus wird in Abb. 1 visualisiert.

Der Ansatz von MSPOT ist keine grundsätzlich neue Idee. Wie bereits in Abschnitt 1 erwähnt, verwenden mehrere andere Verfahren Surrogatmodelle in der Mehrzieloptimierung. Voutchkov und Keane [12] setzen beispielsweise NSGA-II ein um, ähnlich wie MSPOT, neue, vielversprechende Lösungen mit verschiedenen Surrogatmodellen zu generieren. Dabei wird, anders als in MSPOT, der euklidische Abstand verwendet, um eine gleichmäßige Verteilung der Punkte auf der Pareto-Front zu garantieren. Jeong and Obayashi [13] betrachten zwar auch alle Ziele getrennt voneinander, nutzen aber nicht die vorhergesagten Erwartungswerte des Modells, sondern die Expected Improvement (EI) jedes einzelnen Modells als Kriterium und verwenden ausschließlich Kriging.

3.2 SMS-EMOA

Die verwendete SMS-EMOA [14] Variante ist eine Implementierung in `R` von Mersmann[1]. Hier wird in jeder Iteration ein neues Individuum durch Rekombination und Mutation erstellt und ausgewertet. Wenn es einen entsprechend höheren Hypervolumen-Beitrag hat, ersetzt das Individuum das bisherige Mitglied der Population mit dem geringsten Beitrag.

[1] Verfügbar unter: `https://git.p-value.net/emoa.git/plain/examples/sms_emoa.r`

3.3 KRIG-SMS-EMOA

Basierend auf dem zuvor beschriebenen SMS-EMOA unterscheidet sich der KRIG-SMS-EMOA Optimierer dahin gehend, dass hier mehrere Individuen in jeder Iteration erstellt werden. Durch Auswertung der Fitnesswerte des Kriging-Modells wird von diesen Individuen dasjenige mit der größten erwarteten Verbesserung bestimmt. Das ausgewählte Individuum wird auf der realen Zielfunktion ausgewertet und bei entsprechender Güte in die Population aufgenommen. Im Gegensatz zum MSPOT Ansatz ist hier vor allem zu beachten, wie die Modelle ausgenutzt werden. Während in MSPOT Latin Hypercube Sampling oder MCO Optimierer genutzt werden, verwendet KRIG-SMS-EMOA die Variationsoperatoren des SMS-EMOA.

4 Versuchsaufbau

Die Optimieralgorithmen werden auf 10 verschiedenen Testfunktionen verglichen, ZDT1 bis ZDT6 und DTLZ1 bis DTLZ4. [15, 16] In vielen Tests wird ZDT5 ausgelassen, da es kein kontinuierliches Problem ist. Da hier allerdings auch Surrogatmodelle wie Random Forest ihre Anwendung finden, sollte auch ZDT5 von Interesse sein.

Die Testfunktionen und ihre Konfiguration sind in Tabelle 1 zusammengefasst. Vor allem bei ZDT1 bis ZDT3 sollte die hohe Dimensionalität des Entscheidungsraums in Zusammenhang mit den sehr kleinen benutzten Budgets zu einem höheren Schwierigkeitsgrad führen. Dabei ist nicht zu erwarten, dass die Optimierer die tatsächliche Pareto-Front innerhalb der erlaubten Zahl von Funktionsauswertungen approximieren können.

Die Einstellungen der verschiedenen Läufe mit MSPOT sind in Tabelle 2 aufgelistet. Da die Zielfunktionen kein Rauschen aufweisen, wurde die Anzahl der Wiederholungen für jeden Designpunkt auf $k = 1$ festgelegt.

Die drei verschiedenen Surrogatmodelle, die eingesetzt werden, sind:

1. **spotPredictForrester** (MSPOT-Krig): Eine Kriging Implementierung in R basierend auf MATLAB Code von Forrester et al. [17]. Ein Nelder-Mead [18] Algorithmus mit bound constraints [19] aus dem `nloptr` R-Paket[2] wird zur internen Parameterbestimmung verwendet.

[2] Alle hier verwendeten R-Pakete (z.B. `SPOT` oder `nloptr`) sind auf der CRAN Homepage frei verfügbar, i.e. `http://cran.r-project.org`

Tabelle 1: Auflistung der Testfunktionen. Angegeben sind Such- (d) und Zielraumdimensionen (n). Der Referenzpunkt wird für die Auswertung (Hypervolumenvergleich) und von den Optimierern selbst benutzt.

Function	Dimension (d,n)	Reference Point	Grenzen (low,high)
ZDT1	(30,2)	(11,11)	x_i:(0,1)
ZDT2	(30,2)	(11,11)	x_i:(0,1)
ZDT3	(30,2)	(11,11)	x_1:(0,1) x_i:(-5,5)
ZDT4	(30,2)	(11,200)	x_1:(0,2^{30}) x_i:(0,2^5)
ZDT5	(30,2)	(50,50)	(0,1)
ZDT6	(30,2)	(11,11)	(0,1)
DTLZ1	(7,3)	(1000,1000,1000)	(0,1)
DTLZ2	(12,3)	(11,11,11)	(0,1)
DTLZ3	(12,3)	(1000,1000,1000)	(0,1)
DTLZ4	(12,3)	(11,11,11)	(0,1)

2. **spotPredictRandomForest** (MSPOT-RF): Random Forest Modell aus dem R-Paket `randomForest`, das auf Breiman and Cutler's Fortran Code für Klassifikation und Regression [10] basiert.
3. **spotPredictEarth** (MSPOT-MARS): Multivariate adaptive regression splines (MARS) [9] aus dem R-Paket `earth`.

Die drei Modelle werden mit Standardwerten genutzt. Mit den festgelegten Einstellungen in Tabelle 2 bedeutet dies, dass keine Anpassungen an die Dimensionalität vorgenommen werden. Ausnahmen sind übliche Anpassung in den Prozeduren zur Erstellung der Surrogatmodelle selbst (Parameterschätzung, etc.).

Für das Kriging Modell in KRIG-SMS-EMOA gelten die gleichen Einstellungen wie in MSPOT. Es wird die gleiche Anzahl von Auswertungen auf dem Modell benutzt, wie auch in MSPOT. Alle hier verwendeten Optimierer speichern ein Archiv an Pareto-optimalen Lösungen, das für die finale Auswertung berücksichtigt wird.

Da die Zahl der Funktionsauswertungen in Anlehnung an reale industrielle Optimierungsprobleme sehr klein gehalten werden soll, werden diese auf maximal 250 festgelegt. Es werden also nicht die üblichen mehreren tausend Auswertungen erlaubt.

Tabelle 2: Parameter Setup für SPOT

Parameter	Wert
auto.loop.nevals	250
spot.ocba	FALSE
init.design.size m	50
init.design.repeats k_0	1
seq.design.maxRepeats k	1
seq.design.new.size a	1
seq.design.size a	500
seq.design.oldBest.size	0
seq.predictionModel.func M	`"spotPredictForrester"` `"spotPredictRandomForest"` `"spotPredictEarth"`
seq.predictionOpt.budget	1000
seq.predictionOpt.method	`"sms-emoa"`

5 Ergebnisanalyse

Die Ergebnisse der Experimente werden in Abb. 2 dargestellt. Wie bereite eingangs vermutet wurde, werden die niedrig dimensionalen Instanzen von ZDT1 annähernd gelöst, während bei fast allen anderen der Abstand zum optimalen Hypervolumen recht groß ist. Generell lässt sich beobachten, dass SMS-EMOA ohne Surrogatmodell, wie zu erwarten, am schlechtesten abschneidet. Fast immer ähnliche Ergebnisse wie SMS-EMOA liefert MSPOT-RF.

Die Ausnahme von dieser Regel bildet vor allem DTLZ2, wo MSPOT-RF nicht nur deutlich besser als SMS-EMOA ist, sondern auch alle anderen Ansätze hinter sich lässt. Für diesen Fall lässt sich an der Verteilung der gefundenen Pareto-Front erkennen, dass MSPOT-RF vor allem Randpunkte findet und damit die Ränder der Front bereits recht gut approximiert. Es werden aber nur selten Punkte in der Mitte der Pareto-Front gefunden.

Des Weiteren zeigen sowohl MSPOT-Krig als auch MSPOT-MARS oft ähnlich gute Ergebnisse, wobei MSPOT-Krig fast immer am besten oder an zweiter Stelle abschneidet, während MSPOT-MARS in einigen Fällen (z.B. DTLZ1 und DTLZ3) sogar die schlechtesten Ergebnisse liefert. Über alle Funktionen zeigt MSPOT-Krig damit die robustesten Ergebnisse und sollte somit als Methode der Wahl gelten. Typischerweise scheint KRIG-SMS-EMOA zwischen MSPOT-Krig und SMS-EMOA zu liegen. Dabei scheint die Art der Modellausnutzung in MSPOT zuverlässiger gu-

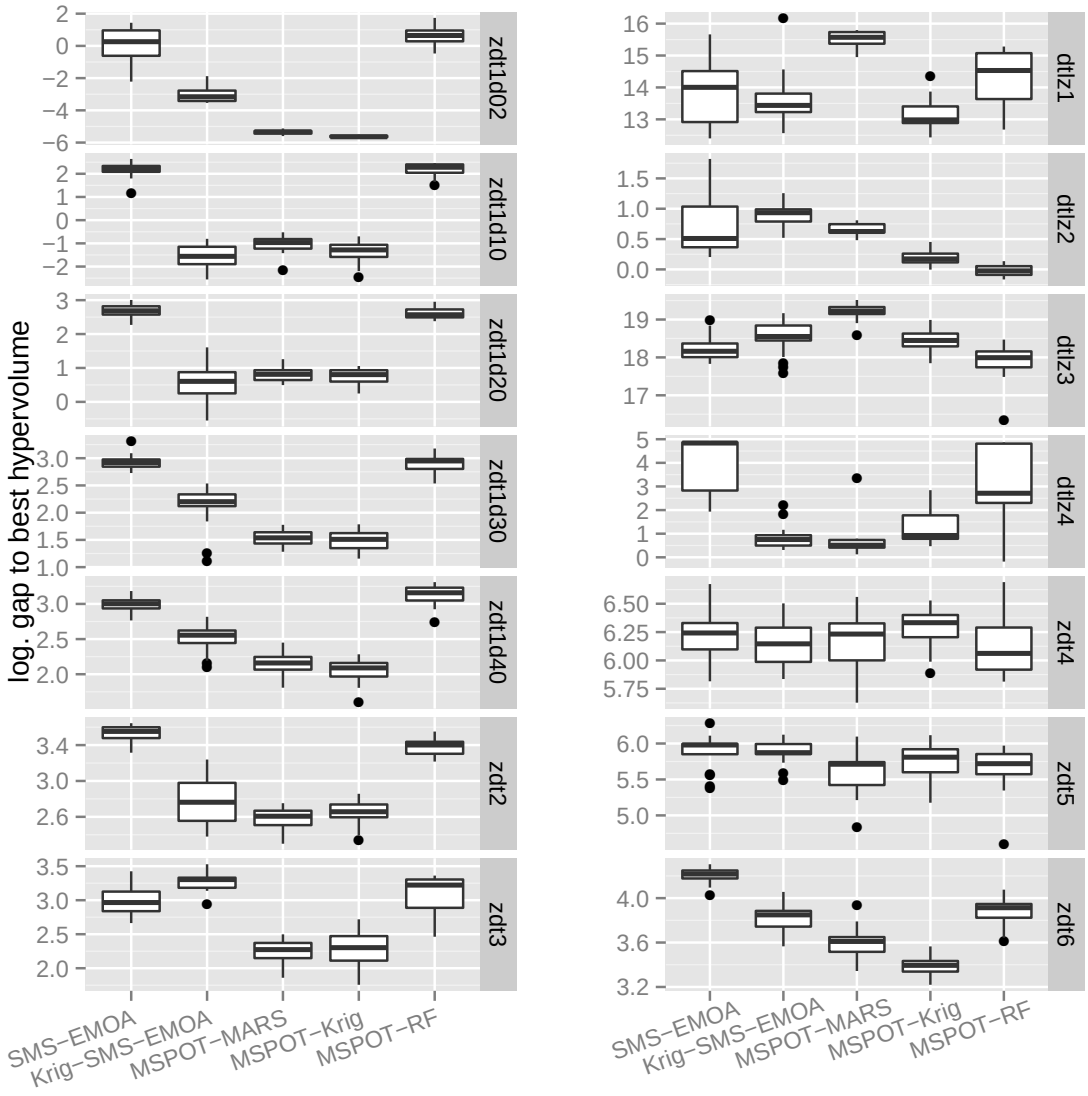

Bild 2: Boxplots: Log. des Abstandes zum optimalen Hypervolumen. Kleinere Werte sind besser.

te Ergebnisse zu liefern, da KRIG-SMS-EMOA in einigen Fällen auch schlechtere Ergebnisse liefert (z.B. ZDT3, DTLZ2).

Weiter ist zu erkennen, dass die verschiedenen Dimensionen bei ZDT1 zumindest ähnliches Verhalten der benutzten Verfahren zeigen. Dabei ergeben sich für die Dimensionen 2, 30 und 40 recht ähnliche Muster, bei denen MSPOT-Krig vorne liegt. Signifikanz zeigt sich aber vor allem bei der zweidimensionalen Instanz. Eine starke Änderung des Verhaltens bei höheren Dimensionen lässt sich nicht erkennen.

Zudem zeigt Abbildung 3 auszugsweise die Entwicklung der Gütewerte über den Optimierungsverlauf. Für ZDT1 kann MSPOT-MARS nach anfänglich deutlichem Rückstand ähnlich gute Werte erreichen wie MSPOT-Krig. Für ZDT6 hingegen ist zu erkennen, dass die Relationen kaum verändert werden, während auch der Gesamtfortschritt im abgebildeten Be-

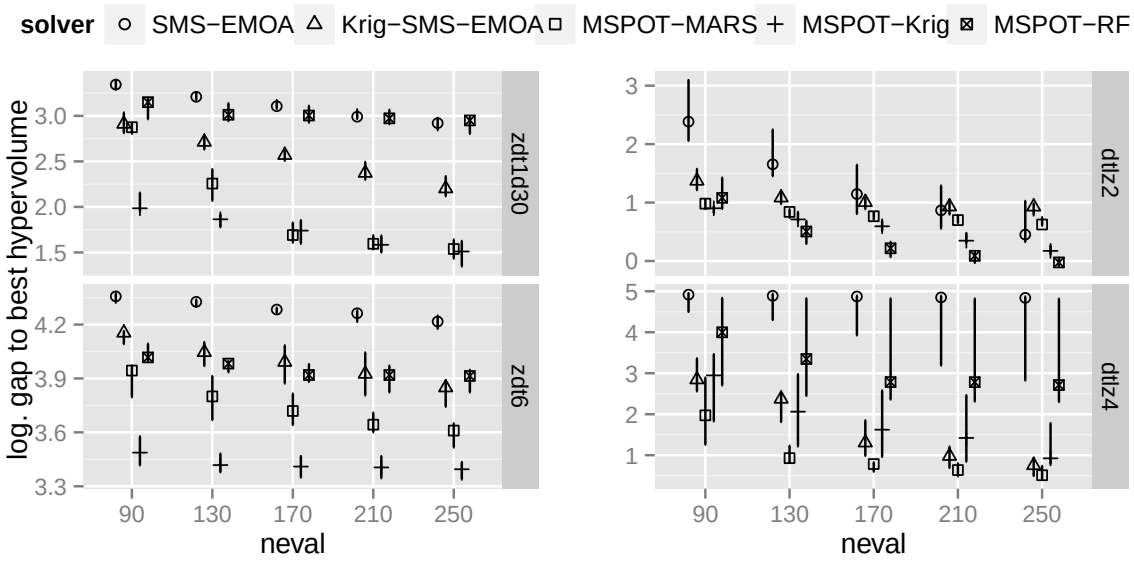

Bild 3: Log. des Abstandes zum optimalen Hypervolumen für ausgewählte Zielfunktionen, zu verschiedenen Zeitpunkten während der Optimierung. Angegeben sind jeweils die erreichten Werte nach `neval` Funktionsauswertungen. Die Markierungen zeigen den Median, die Balken den Quartilsabstand.

reich nicht sehr groß ist. In anderen Fällen (DTLZ4) hingegen kann vor allem KRIG-SMS-EMOA die Lücke zu dem jeweils besten Ansatz langsam schließen. SMS-EMOA kann zwar auch in einem Fall nach anfänglichem Rückstand wieder aufholen (DTLZ2). Andererseits zeigt SMS-EMOA auf DTLZ4 kaum Verbesserung des Median, obwohl einzelne Läufe durchaus in bessere Regionen vorstoßen können.

Neben der erzielten Lösungsgüte ist auch die Zeit für einen Optimierlauf zu beachten. Die durchschnittlich gemessenen Zeiten für einen Durchlauf sind in Tabelle 3 zu finden. Dabei ist klar, dass die Laufzeit des surrogatmodell-freien SMS-EMOA keine Rolle spielt und deshalb auch nicht in der Tabelle gelistet ist. Bei realen, zeitintensiven Anwendungen wäre dieser Wert allerdings, aufgrund der in Relation zu den anderen Optimierern häufigen und dann sehr langen Auswertungszeiten der Zielfunktion, am größten.

Von den anderen Ansätzen ist MSPOT-MARS am schnellsten, MSPOT-RF etwas langsamer und beide Ansätze mit Kriging die langsamsten. Dabei steigt auch der Zeitbedarf der Kriging-Ansätze mit höherer Dimensionalität des Suchraumes am stärksten.

Eine offensichtliche Abweichung des Musters ist, dass auf DTLZ1 MSPOT-MARS im Schnitt langsamer als MSPOT-RF ist. Gleichzeitig ist in diesem Fall auch die Güte von MSPOT-MARS besonders schlecht. Diese Zielfunktion scheint für MARS schwerer und auch langwieriger zu modellie-

Tabelle 3: Zeitbedarf der verschiedenen Optimierer. Zeit in Stunden angegeben. Die Zeitmessung der SMS-EMOA wird in dieser Tabelle ausgelassen, da sich diese außer Konkurrenz in einem Bereich von wenigen Sekunden bewegt. In Klammern ist die Standardabweichung angegeben.

	KRIG-SMS-EMOA	MSPOT-MARS	MSPOT-Krig	MSPOT-RF
ZDT1d02	3.70 (0.13)	**0.49 (0.04)**	2.59 (0.11)	2.62 (0.12)
ZDT1d10	8.26 (0.40)	**0.79 (0.041)**	7.16 (0.41)	2.38 (0.30)
ZDT1d20	15.26 (0.94)	**1.29 (0.06)**	14.41 (0.88)	2.54 (0.16)
ZDT1d30	24.68 (0.74)	**1.80 (0.10)**	23.03 (0.68)	2.53 (0.20)
ZDT1d40	32.57 (4.03)	**2.23 (0.10)**	30.96 (3.12)	2.64 (0.17)
ZDT2d30	24.75 (0.55)	**2.27 (0.13)**	23.14 (0.55)	2.55 (0.25)
ZDT3d30	25.05 (0.50)	**1.57 (0.09)**	22.66 (0.86)	2.60 (0.22)
ZDT4d10	8.32 (0.35)	**0.70 (0.05)**	6.89 (0.26)	2.56 (0.29)
ZDT5d11	8.67 (0.36)	**0.64 (0.07)**	7.18 (0.39)	2.53 (0.26)
ZDT6d10	8.47 (0.27)	**0.73 (0.04)**	7.32 (0.18)	2.56 (0.29)
DTLZ1d7	8.29 (0.36)	4.03 (1.38)	6.94 (0.17)	**3.33 (0.25)**
DTLZ2d12	14.35 (0.38)	**1.93 (0.73)**	12.51 (0.42)	3.47 (0.15)
DTLZ3d12	13.88 (0.56)	**2.30 (0.90)**	12.15 (0.30)	3.38 (0.22)
DTLZ4d12	14.59 (0.29)	**1.60 (0.89)**	12.35 (0.36)	3.58 (0.33)

ren. Auch die Güte auf DTLZ3 ist mit MSPOT-MARS sehr schlecht. Dort ist MSPOT-MARS zwar nicht langsamer als MSPOT-RF, zeigt aber die zweit langsamste Zeit für MSPOT-MARS insgesamt. Lange Laufzeiten scheinen für MARS mit schlechter Güte zu korrelieren.

Für die Wahl der Methode ist also hier auch der Zeitbedarf entscheidend. Für sehr teure bzw. zeitaufwendige Zielfunktionen ist MSPOT-Krig zu empfehlen, während nicht ganz so kostenintensive Zielfunktionen möglicherweise auch mit MSPOT-MARS gut zu lösen sind, vor allem wenn der Zeitbedarf das aufwendigere MSPOT-Krig nicht zulässt. Dabei lässt sich der Zeitbedarf nicht nur durch die Wahl des Modells beeinflussen. Unter anderem kann die Anzahl der Auswertungen auf dem Modell (hier jeweils 1500) den Zeitbedarf verändern. Für den Kriging Ansatz lässt sich des Weiteren auch das Budget für die interne Parameterschätzung anpassen, um den Zeitbedarf zu verändern. Derartige Änderungen hätten auch einen entsprechenden Einfluss auf die Güte der Ergebnisse.

6 Zusammenfassung und Ausblick

Als Zusammenfassung der Ergebnisse werden im folgenden die in Abschnitt 2 gestellten Fragen beantwortet.

I **Lassen sich auch bei stark beschränktem Budget Verbesserungen mit einem Surrogatmodell gegenüber einem klassischen Ansatz erreichen?**
Es zeigt sich, dass von Surrogatmodellen unterstützte Ansätze für die mehrkriterielle Optimierung (MSPOT) wie erwartet von Vorteil sind. Die Pareto-Fronten der meisten getesteten Funktionen lassen sich zwar auf Grund des beschränkten Budgets und der hohen Suchraumdimension nicht gut approximieren, trotzdem erlauben die Surrogatmodelle zumindest eine Verbesserung der Güte.

II **Gibt es bestimmte Umstände die den Einsatz eines Surrogatmodelles unprofitabel machen?**
In einem Fall (ZDT4) zeigt keines der getesteten Ansätze mit Surrogatmodell Vorteile gegenüber der einfachen SMS-EMOA. Hier ist also zumindest bei dem vorliegenden Budget nicht zu empfehlen, die rechnerisch aufwendigeren Modelle einzusetzen. Dies gilt aber nur für den Einsatz auf Zielfunktionen mit geringen Auswertungszeiten.

III **Welchen Einfluss haben unterschiedliche Typen von Surrogatmodellen?**
Die Wahl des Modells scheint meist auf Kriging zu fallen, obwohl einzelne Funktionen mit Random Forest (DTLZ2, DTLZ3) oder MARS (ZDT2, DTLZ4) besser zu lösen sind.

IV **Welchen Einfluss hat die Dimension des Suchraumes?**
Die Dimension des Suchraumes hat für ZDT1 einen klaren Einfluss auf die Güte der Ergebnisse. Es lässt sich in allen Instanzen von ZDT1 erkennen, dass MSPOT-MARS, KRIG-SMS-EMOA und MSPOT-Krig die besten Resultate liefern. Die Varianz der gefundenen Ergebnisse ist dabei für $d = 2$ am geringsten.

V **Wie unterscheidet sich der Rechenaufwand der verschiedenen Ansätze?**
Die Unterschiede im Rechenaufwand sind signifikant und sollten bei der Wahl des Modells berücksichtigt werden. Besonders zu beachten ist, dass der Rechenaufwand für höhere Dimensionen des Suchraumes schlechter skaliert als bei den anderen Ansätzen, dafür aber die

im Schnitt bessere Güte erzielt. Nur wenn die Zielfunktion kostengünstig und schnell auszuwerten ist, ist keines der Modelle zu empfehlen, da hier ein SMS-EMOA Lauf mit größerem Budget vermutlich bessere Ergebnisse in gleicher Zeit oder ähnliche Ergebnisse in weniger Zeit erzielen kann.

Die Ergebnisse zeigen, dass die Performanz der Modelle von den zugrundeliegenden Zielfunktionen abhängt. Daher werden wir bei den weiteren Versuchen Ensemble-Ansätze heran zu ziehen. So kann beispielsweise der MSPOT Ansatz soweit abgeändert werden, dass mehrere Modelltypen verwendet werden, wobei jedem erlaubt wird einen neuen Punkt für die nächste Auswertung vorzuschlagen.

Auch sollte der hier präsentierte MSPOT Ansatz mit anderen relevanten Einfügekriterien auf diesen Problemen verglichen werden. So könnten in SPOT auch Kriterien wie Hypervolume Expected Improvement [5], oder ähnliche Ansätze implementiert werden, die die verschiedenen Ziele aggregieren und/oder die geschätzte Varianz berücksichtigen. Hier ist aber auch eine weitere Steigerung des rechnerischen Aufwandes zu erwarten.

In weiteren Untersuchungen sollen diese Methoden vor allem auf realen Problemen angewendet werden, um die praktische Relevanz der Ergebnisse zu überprüfen. Da reale Probleme oft mehr als nur zwei Ziele haben, ist auch eine Erhöhung der Suchraumdimension von weiterem Interesse. Für die einleitend erwähnte Optimierung eines Staubabscheiders sind zum Beispiel vier bis zehn Ziele von unterschiedlicher Relevanz zu beachten.

Literatur

[1] Bartz-Beielstein, T.; Parsopoulos, K. E.; Vrahatis, M. N.: Design and analysis of optimization algorithms using computational statistics. *Applied Numerical Analysis and Computational Mathematics (ANACM)* 1 (2004) 2, S. 413–433.

[2] Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing* 9 (2005) 1, S. 3–12.

[3] Knowles, J. D.; Nakayama, H.: Meta-Modeling in Multiobjective Optimization. In: *Multiobjective Optimization*, S. 245–284. Springer. 2008.

[4] Knowles, J.: ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation* 10 (2006) 1, S. 50–66.

[5] Emmerich, M.: *Single- and Multi-objective Evolutionary Design Optimization: Assisted by Gaussian Random Field Metamodels*. Dissertation, Universität Dortmund, Germany. 2005.

[6] Ponweiser, W.; Wagner, T.; Biermann, D.; Vincze, M.: Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted -Metric Selection. In: *PPSN*, S. 784–794. 2008.

[7] Wagner, T.; Emmerich, M.; Deutz, A.; Ponweiser, W.: On expected-improvement criteria for model-based multi-objective optimization. *Parallel Problem Solving from Nature–PPSN XI* (2010), S. 718–727.

[8] Zaefferer, M.; Bartz-Beielstein, T.; Friese, M.; Naujoks, B.; Flasch, O.: Multi-Criteria Optimization for Hard Problems under Limited Budgets. In: *GECCO Companion '12: Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion* (Soule, T.; et al., Hg.), S. 1451–1452. Philadelphia, Pennsylvania, USA: ACM. ISBN 978-1-4503-1178-6. 2012.

[9] Friedman, J. H.: Multivariate adaptive regression splines. *Ann. Stat.* 19 (1991) 1, S. 1–141.

[10] Breiman, L.: Random Forests. *Machine Learning* 45 (2001) 1, S. 5 –32.

[11] Deb, K.: *Multi-Objective Optimization using Evolutionary Algorithms*. New York NY: Wiley. 2001.

[12] Voutchkov, I.; Keane, A.: Multiobjective Optimization Using Surrogates. In: *Adaptive Computing in Design and Manufacture ACDM*, S. 167–175. ISBN 0-9552885-0-9. 2006.

[13] Jeong, S.; Obayashi, S.: Efficient global optimization (EGO) for multi-objective problem and data mining. In: *IEEE Congress on Evolutionary Computation* (Corne, D.; et al., Hg.), S. 2138–2145. IEEE. ISBN 0-7803-9363-5. 2005.

[14] Beume, N.; Naujoks, B.; Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* 181 (2007) 3, S. 1653–1669.

[15] Zitzler, E.; Deb, K.; Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* 8 (2000) 2, S. 173–195.

[16] K. Deb, L. Thiele, M. L.; Zitzler, E.: Scalable Test Problems for Evolutionary Multi-Objective Optimization. Technical Report 112, Institut für Technische Informatik und Kommunikationsnetze, ETH Zürich, Zürich. 2001.

[17] Forrester, A.; Sobester, A.; Keane, A.: *Engineering Design via Surrogate Modelling*. Wiley. 2008.

[18] Nelder, J.; Mead, R.: A simplex method for function minimization. *Computer Journal* 7 (1965) 4, S. 308–313.

[19] Box, M. J.: A New Method of Constrained Optimization and a Comparison With Other Methods. *The Computer Journal* 8 (1965) 1, S. 42–52.