An Experimental Investigation of Model-Based Parameter Optimization: SPO and Beyond

Frank Hutter, Holger H. Hoos,
Kevin Leyton-Brown, Kevin P. Murphy

Department of Computer Science
University of British Columbia
Canada
{hutter, hoos, kevinlb, murphyk}@cs.ubc.ca

Motivation for Parameter Optimization

Genetic Algorithms & Evolutionary Strategies are

+ Very flexible frameworks

Motivation for Parameter Optimization

Genetic Algorithms & Evolutionary Strategies are

+ Very flexible frameworks
– Tedious to configure for a new domain
    - Population size
    - Mating scheme
    - Mutation rate
    - Search operators
    - Hybridizations, ...

Motivation for Parameter Optimization

Genetic Algorithms & Evolutionary Strategies are

+ Very flexible frameworks
– Tedious to configure for a new domain
  - Population size
  - Mating scheme
  - Mutation rate
  - Search operators
  - Hybridizations, ...

Automated parameter optimization can help

- High-dimensional optimization problem
- Automate $\rightsquigarrow$ saves time & improves results

Parameter Optimization Methods

- Numerical parameters
  - See Blackbox optimization workshop (this GECCO)
  - Algorithm parameters: CALIBRA [Adenso-Diaz & Laguna, '06]

Parameter Optimization Methods

- Numerical parameters
  - See Blackbox optimization workshop (this GECCO)
  - Algorithm parameters: CALIBRA [Adenso-Diaz & Laguna, '06]
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]

Parameter Optimization Methods

- Numerical parameters
  - See Blackbox optimization workshop (this GECCO)
  - Algorithm parameters: CALIBRA [Adenso-Diaz & Laguna, '06]
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  - Genetic algorithms [Terashima-Marín, Ross & Valenzuela-Réndon, '99]

Parameter Optimization Methods

- Numerical parameters
  - See Blackbox optimization workshop (this GECCO)
  - Algorithm parameters: CALIBRA [Adenso-Diaz & Laguna, '06]
- Few categorical parameters: racing algorithms

  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  - Genetic algorithms [Terashima-Marín, Ross & Valenzuela-Réndon, '99]
  - Iterated Local Search

    [Hutter, Hoos, Leyton-Brown & Stützle, '07-'09]
    - ⇝ Dozens of parameters (*e.g.*, CPLEX with 63 parameters)
    - ⇝ For many problems: SAT, MIP, time-tabling, protein folding, MPE, ...

Parameter Optimization Methods

Model-free Parameter Optimization

- Numerical parameters: see BBOB workshop (this GECCO)
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  [*e.g.*, Terashima-Marín, Ross & Valenzuela-Réndon, '99, Hutter, Hoos, Leyton-Brown & Stützle, '07-'09]

Parameter Optimization Methods

Model-free Parameter Optimization

- Numerical parameters: see BBOB workshop (this GECCO)
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  [*e.g.*, Terashima-Marín, Ross & Valenzuela-Réndon, '99, Hutter, Hoos, Leyton-Brown & Stützle, '07-'09]

Model-based Parameter Optimization

Parameter Optimization Methods

Model-free Parameter Optimization

- Numerical parameters: see BBOB workshop (this GECCO)
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  [*e.g.*, Terashima-Marín, Ross & Valenzuela-Réndon, '99, Hutter, Hoos, Leyton-Brown & Stützle, '07-'09]

Model-based Parameter Optimization

- Methods
  - Fractional factorial designs [*e.g.*, Ridge & Kudenko, '07]
  - Sequential Parameter Optimization (SPO)
    [Bartz-Beielstein, Preuss, Lasarczyk, '05-'09]

Parameter Optimization Methods

Model-free Parameter Optimization

- Numerical parameters: see BBOB workshop (this GECCO)
- Few categorical parameters: racing algorithms
  [Birattari, Stützle, Paquete & Varrentrapp, '02]
- Many categorical parameters
  [*e.g.*, Terashima-Marín, Ross & Valenzuela-Réndon, '99, Hutter, Hoos, Leyton-Brown & Stützle, '07-'09]

Model-based Parameter Optimization

- Methods
  - Fractional factorial designs [*e.g.*, Ridge & Kudenko, '07]
  - Sequential Parameter Optimization (SPO)
    [Bartz-Beielstein, Preuss, Lasarczyk, '05-'09]
- Can use model for more than optimization
  - Importance of each parameter
  - Interaction between parameters

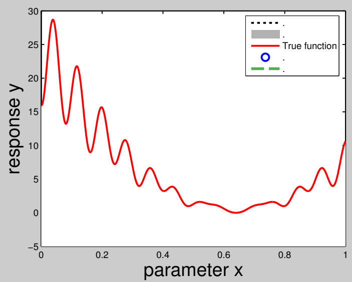
Outline

1. Sequential Model-Based Optimization (SMBO): Introduction
2. Comparing Two SMBO Methods: SPO *vs* SKO
3. Components of SPO: Model Quality
4. Components of SPO: Sequential Experimental Design
5. Conclusions and Future Work

Outline

1. Sequential Model-Based Optimization (SMBO): Introduction
2. Comparing Two SMBO Methods: SPO *vs* SKO
3. Components of SPO: Model Quality
4. Components of SPO: Sequential Experimental Design
5. Conclusions and Future Work

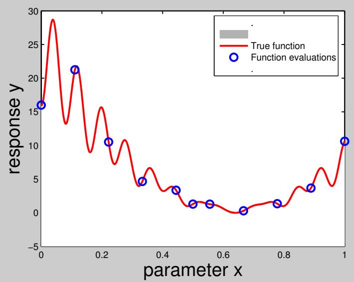
SMBO: Introduction



First step of SMBO

SMBO: Introduction

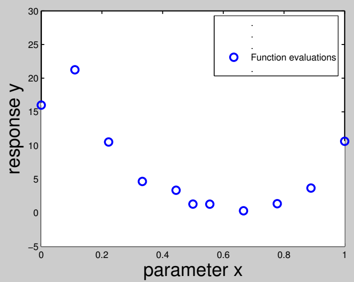
1. Get response values at initial design points



First step of SMBO

SMBO: Introduction

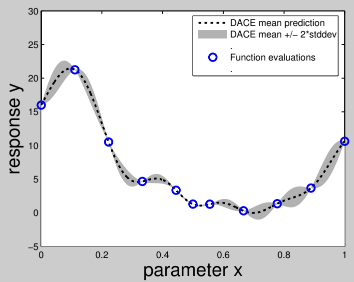
1. Get response values at initial design points



First step of SMBO

SMBO: Introduction

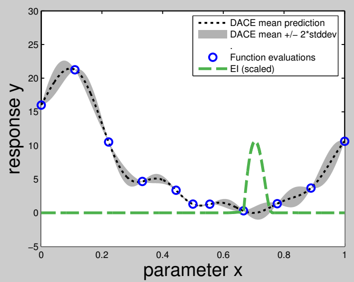
1. Get response values at initial design points
2. Fit a model to the data



First step of SMBO

SMBO: Introduction

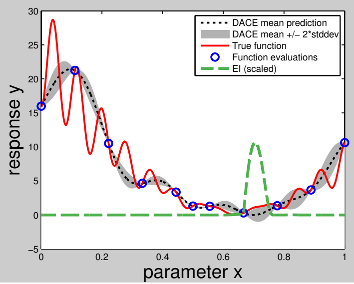
1. Get response values at initial design points
2. Fit a model to the data
3. Use model to pick most promising next design point (based on expected improvement criterion)



First step of SMBO

SMBO: Introduction

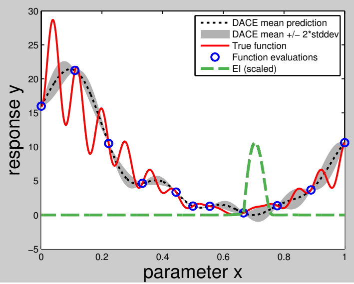
1. Get response values at initial design points
2. Fit a model to the data
3. Use model to pick most promising next design point (based on expected improvement criterion)



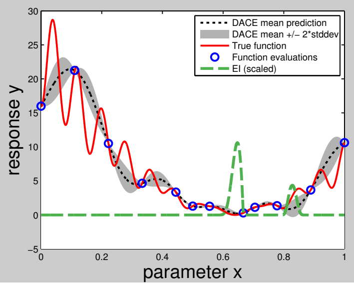
First step of SMBO

SMBO: Introduction

1. Get response values at initial design points
2. Fit a model to the data
3. Use model to pick most promising next design point (based on expected improvement criterion)
4. Repeat 2. and 3. until time is up



First step of SMBO

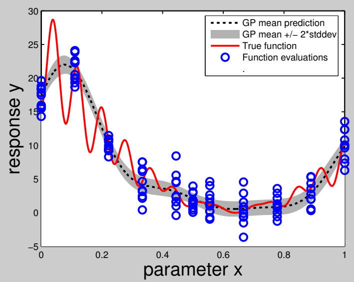


Second step of SMBO

Outline

Dealing with Noise: SKO *vs* SPO

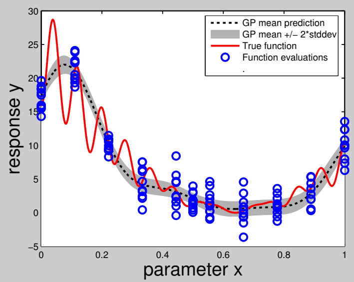
- Method I (used in SKO) [Huang, Allen, Notz & Zeng, '06.]
  - Fit standard GP assuming Gaussian observation noise
  - Can only fit the mean of the responses



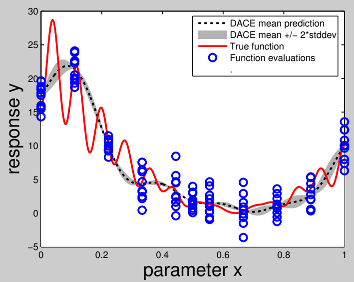
Method I: noisy fit of original response

Dealing with Noise: SKO *vs* SPO

- Method I (used in SKO) [Huang, Allen, Notz & Zeng, '06.]
  - Fit standard GP assuming Gaussian observation noise
  - Can only fit the mean of the responses
- Method II (used in SPO) [Bartz-Beielstein, Preuss, Lasarczyk, '05-'09]
  - Compute *statistic of empirical distribution of responses* at each design point
  - Fit noise-free GP to that



Method I: noisy fit of original response



Method II: noise-free fit of cost statistic

Experiment: SPO *vs* SKO for Tuning CMA-ES

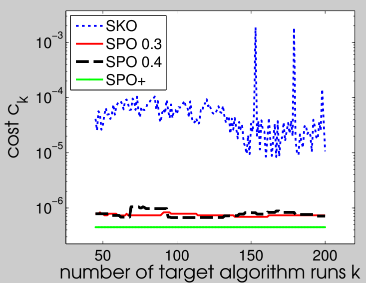
- CMA-ES [Hansen et al., '95-'09]
  - Evolutionary strategy for global optimization
  - State-of-the-art (see BBOB workshop this GECCO)
  - Parameters: population size, number of parents, learning rate, damping parameter

Experiment: SPO *vs* SKO for Tuning CMA-ES

- CMA-ES [Hansen et al., '95-'09]
  - Evolutionary strategy for global optimization
  - State-of-the-art (see BBOB workshop this GECCO)
  - Parameters: population size, number of parents, learning rate, damping parameter
- Tuning objective
  - Solution cost: best function value found in budget
  - Here: Sphere function
  - Minimize mean solution cost across many runs

Experiment: SPO *vs* SKO for Tuning CMA-ES

- CMA-ES [Hansen et al., '95-'09]
  - Evolutionary strategy for global optimization
  - State-of-the-art (see BBOB workshop this GECCO)
  - Parameters: population size, number of parents, learning rate, damping parameter
- Tuning objective
  - Solution cost: best function value found in budget
  - Here: Sphere function
  - Minimize mean solution cost across many runs



Outline

1. Sequential Model-Based Optimization (SMBO): Introduction
2. Comparing Two SMBO Methods: SPO *vs* SKO
3. Components of SPO: Model Quality
4. Components of SPO: Sequential Experimental Design
5. Conclusions and Future Work

Components of SPO: initial design

- Fixed number of initial design points (250) and repeats (2)
  - Size of initial design studied before [Bartz-Beielstein & Preuss, '06]
- Here: studied *which* 250 design points to use

Components of SPO: initial design

- Fixed number of initial design points (250) and repeats (2)
  - Size of initial design studied before [Bartz-Beielstein & Preuss, '06]
- Here: studied *which* 250 design points to use
  - Sampled uniformly at random
  - Random Latin Hypercube
  - Iterated Hypercube Sampling [Beachkofski & Grandhi, '02]
  - SPO's standard LHD

Components of SPO: initial design

- Fixed number of initial design points (250) and repeats (2)
  - Size of initial design studied before [Bartz-Beielstein & Preuss, '06]
- Here: studied *which* 250 design points to use
  - Sampled uniformly at random
  - Random Latin Hypercube
  - Iterated Hypercube Sampling [Beachkofski & Grandhi, '02]
  - SPO's standard LHD
- Result: no significant difference
  - Initial design not very important
  - Using cheap random LHD from here on

Components of SPO: Transformations

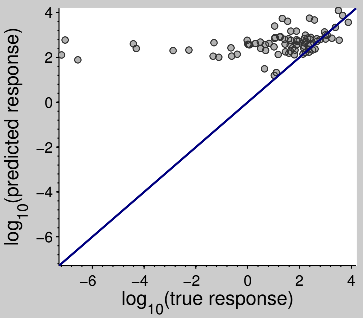
- Compute empirical cost statistics $\hat{c}(\theta)$ first
- Then transform cost statistics: $\log(\hat{c}(\theta))$

Components of SPO: Transformations

- Compute empirical cost statistics $\hat{c}(\theta)$ first
- Then transform cost statistics: $\log(\hat{c}(\theta))$
- Data: solution cost of CMA-ES on sphere
  - Training: $250 \cdot 2$ data points as above
  - Test: 250 new points, sampled uniformly at random

Components of SPO: Transformations

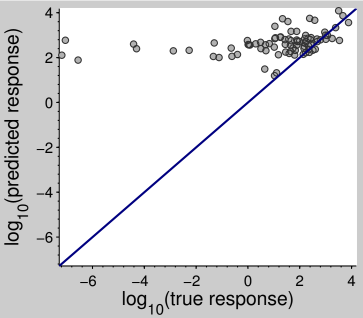
- Compute empirical cost statistics $\hat{c}(\theta)$ first
- Then transform cost statistics: $\log(\hat{c}(\theta))$
- Data: solution cost of CMA-ES on sphere
  - Training: $250 \cdot 2$ data points as above
  - Test: 250 new points, sampled uniformly at random



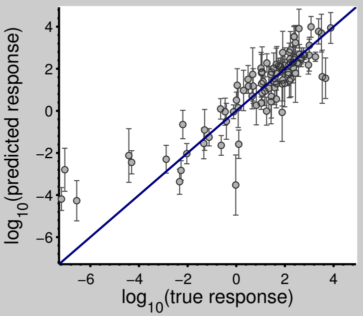
No transformation

Components of SPO: Transformations

- Compute empirical cost statistics $\hat{c}(\theta)$ first
- Then transform cost statistics: $\log(\hat{c}(\theta))$
- Data: solution cost of CMA-ES on sphere
  - Training: $250 \cdot 2$ data points as above
  - Test: 250 new points, sampled uniformly at random



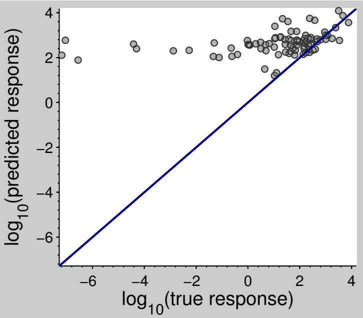
No transformation



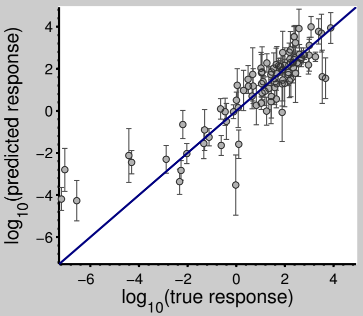
Log transformation

Components of SPO: Transformations

- Compute empirical cost statistics $\hat{c}(\theta)$ first
- Then transform cost statistics: $\log(\hat{c}(\theta))$
- Data: solution cost of CMA-ES on sphere
  - Training: $250 \cdot 2$ data points as above
  - Test: 250 new points, sampled uniformly at random



No transformation



Log transformation

Note: In newer experiments, SKO with log models was competitive

Outline

1. Sequential Model-Based Optimization (SMBO): Introduction
2. Comparing Two SMBO Methods: SPO *vs* SKO
3. Components of SPO: Model Quality
4. Components of SPO: Sequential Experimental Design
5. Conclusions and Future Work

Components of SPO: expected improvement criterion

User wants to optimize some objective c

- We transform c to improve the model
- But that doesn't change the user's objective

$\rightsquigarrow$ Have to adapt expected improvement criterion to handle un-transformed objective

Components of SPO: expected improvement criterion

User wants to optimize some objective c

- We transform c to improve the model
- But that doesn't change the user's objective

$\rightsquigarrow$ Have to adapt expected improvement criterion to handle un-transformed objective

Fix for log-transform: new expected improvement criterion

- Want to optimize $I_{\exp}(\theta) = \max\{0, f_{min} - \exp[f(\theta)]\}$
- There is a closed-form solution (see paper)

Components of SPO: expected improvement criterion

User wants to optimize some objective c

- We transform c to improve the model
- But that doesn't change the user's objective

$\leadsto$ Have to adapt expected improvement criterion to handle un-transformed objective

Fix for log-transform: new expected improvement criterion

- Want to optimize $I_{\exp}(\theta) = \max\{0, f_{min} - \exp[f(\theta)]\}$
- There is a closed-form solution (see paper)
- However: no significant improvement in our experiments

Components of SPO: choosing the incumbent parameter setting in presence of noise

Some algorithm runs can be lucky

- $\rightsquigarrow$ need extra mechanism to ensure incumbent is really good
- $\rightsquigarrow$ SPO increases number of repeats over time

Components of SPO: choosing the incumbent parameter setting in presence of noise

Some algorithm runs can be lucky

$\rightsquigarrow$ need extra mechanism to ensure incumbent is really good

$\rightsquigarrow$ SPO increases number of repeats over time

SPO's mechanism in a nutshell

- Compute cost statistic $\hat{c}(\theta)$ for each configuration θ

Components of SPO: choosing the incumbent parameter setting in presence of noise

Some algorithm runs can be lucky

$\rightsquigarrow$ need extra mechanism to ensure incumbent is really good

$\rightsquigarrow$ SPO increases number of repeats over time

SPO's mechanism in a nutshell

- Compute cost statistic $\hat{c}(\theta)$ for each configuration θ
- $\theta_{inc} \leftarrow$ configuration with lowest $\hat{c}(\theta)$

Components of SPO: choosing the incumbent parameter setting in presence of noise

Some algorithm runs can be lucky

$\rightsquigarrow$ need extra mechanism to ensure incumbent is really good

$\rightsquigarrow$ SPO increases number of repeats over time

SPO's mechanism in a nutshell

- Compute cost statistic $\hat{c}(\theta)$ for each configuration θ
- $\theta_{inc} \leftarrow$ configuration with lowest $\hat{c}(\theta)$
- Perform up to R runs for θ_{inc} to ensure it is good
  - Increase R over time

Components of SPO: choosing the incumbent parameter setting in presence of noise

Some algorithm runs can be lucky

$\rightsquigarrow$ need extra mechanism to ensure incumbent is really good

$\rightsquigarrow$ SPO increases number of repeats over time

SPO's mechanism in a nutshell

- Compute cost statistic $\hat{c}(\theta)$ for each configuration θ
- $\theta_{inc} \leftarrow$ configuration with lowest $\hat{c}(\theta)$
- Perform up to R runs for θ_{inc} to ensure it is good
  - Increase R over time
- But what if it doesn't perform well?
  - Then a different incumbent is picked in the next iteration
  - That might also turn out not to be good...

Components of SPO: choosing the incumbent parameter setting in presence of noise

Simple fix

- Iteratively perform runs for single most promising θ_{new}
  - Compare against current incumbent θ_{inc}
  - Once θ_{new} has as many runs as θ_{inc}: make it new θ_{inc}
- Maintain invariant: θ_{inc} has the most runs of all

Components of SPO: choosing the incumbent parameter setting in presence of noise

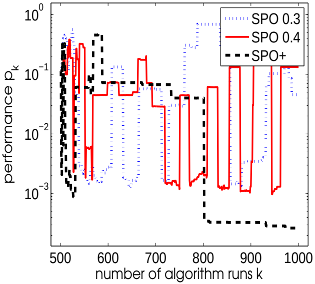
Simple fix

- Iteratively perform runs for single most promising θ_{new}
  - Compare against current incumbent θ_{inc}
  - Once θ_{new} has as many runs as θ_{inc}: make it new θ_{inc}
- Maintain invariant: θ_{inc} has the most runs of all
- Substantially improves robustness $\rightarrow$ new SPO variant: SPO$^+$

Components of SPO: choosing the incumbent parameter setting in presence of noise

Simple fix

- Iteratively perform runs for single most promising θ_{new}
  - Compare against current incumbent θ_{inc}
  - Once θ_{new} has as many runs as θ_{inc}: make it new θ_{inc}
- Maintain invariant: θ_{inc} has the most runs of all
- Substantially improves robustness $\rightarrow$ new SPO variant: SPO$^+$

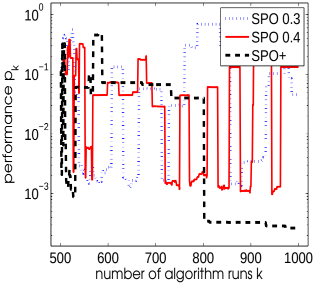


Tuning CMA-ES on Griewangk

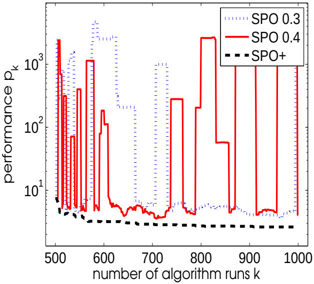
Components of SPO: choosing the incumbent parameter setting in presence of noise

Simple fix

- Iteratively perform runs for single most promising θ_{new}
  - Compare against current incumbent θ_{inc}
  - Once θ_{new} has as many runs as θ_{inc}: make it new θ_{inc}
- Maintain invariant: θ_{inc} has the most runs of all
- Substantially improves robustness $\rightarrow$ new SPO variant: SPO$^+$



Tuning CMA-ES on Griewangk



Tuning CMA-ES on Rastrigin

Summary of Study of SPO components & Definition of SPO$^+$

Model Quality

- Initial design not very important
  - $\rightsquigarrow$ use simple random LHD in SPO$^+$
- Log-transforms sometimes improve model quality a lot
  - $\rightsquigarrow$ use them in SPO$^+$ (for positive functions)

Summary of Study of SPO components & Definition of SPO$^+$

Model Quality

- Initial design not very important
  - $\rightsquigarrow$ use simple random LHD in SPO$^+$
- Log-transforms sometimes improve model quality a lot
  - $\rightsquigarrow$ use them in SPO$^+$ (for positive functions)

Sequential Experimental Design

- Expected improvement criterion
  - $\rightsquigarrow$ New one that's better in theory but not in practice
  - $\rightsquigarrow$ Use original one in SPO$^+$
- New mechanism for increasing #runs & selecting incumbent
  - $\rightsquigarrow$ substantially improves robustness
  - $\rightsquigarrow$ Use it in SPO$^+$

Comparison to State of the Art for tuning SAPS

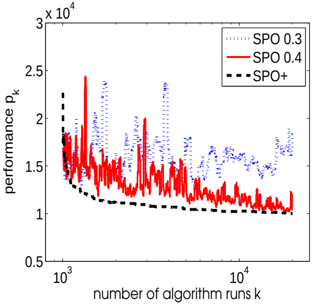
- SAPS
  - Stochastic local search algorithm for SAT
  - 4 continuous parameters
  - Here: min. search steps for single problem instance

Comparison to State of the Art for tuning SAPS

- SAPS
  - Stochastic local search algorithm for SAT
  - 4 continuous parameters
  - Here: min. search steps for single problem instance
- Results known for CALIBRA & ParamILS [Hutter et al, AAAI'07]

Comparison to State of the Art for tuning SAPS

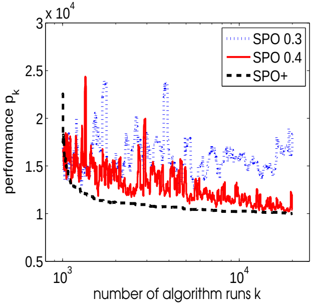
- SAPS
  - Stochastic local search algorithm for SAT
  - 4 continuous parameters
  - Here: min. search steps for single problem instance
- Results known for CALIBRA & ParamILS [Hutter et al, AAAI'07]



Comparison to SPO variants, with varying budget

Comparison to State of the Art for tuning SAPS

- SAPS
  - Stochastic local search algorithm for SAT
  - 4 continuous parameters
  - Here: min. search steps for single problem instance
- Results known for CALIBRA & ParamILS [Hutter et al, AAAI'07]



Comparison to SPO variants, with varying budget

Procedure	SAPS median run-time/10^3
SAPS default	85.5
CALIBRA(100)	10.7 ± 1.1
BasicILS(100)	10.9 ± 0.6
FocusedILS	10.6 ± 0.5
SPO 0.3	18.3 ± 13.7
SPO 0.4	10.4 ± 0.7
SPO$^+$	$\mathbf{10.0} \pm \mathbf{0.4}$

With budget of 20000 runs of SAPS

Outline

Conclusions

- SMBO can help design algorithms
  - More principled, saves development time
  - Can exploit full potential of flexible algorithms

Conclusions

- SMBO can help design algorithms
  - More principled, saves development time
  - Can exploit full potential of flexible algorithms
- Our contribution
  - Insights: what makes a popular SMBO algorithm, SPO, work
  - Improved version, SPO^+, often performs better than SPO

Ongoing & Future Work

Ongoing Extensions of Model-Based Framework

- Use of different models in SPO^+ framework
- Dealing with categorical parameters
- Optimization for Sets/Distributions of Instances

Ongoing & Future Work

Ongoing Extensions of Model-Based Framework

- Use of different models in SPO^+ framework
- Dealing with categorical parameters
- Optimization for Sets/Distributions of Instances

Use of models for scientific understanding

- Interactions of instance features and parameter values
- Can help understand and hopefully improve algorithms

Thanks to

- Thomas Bartz-Beielstein
  - SPO implementation & CMA-ES wrapper
- Theodore Allen
  - SKO implementation