Direct Multisearch for Multiobjective Optimization

Ana Luísa Custódio[1] José F. Aguilar Madeira[2]

A. Ismael F. Vaz[3] Luís Nunes Vicente[4]

[1]Universidade Nova de Lisboa [2]IDMEC-IST, ISEL

[3]Universidade do Minho [4]Universidade de Coimbra

Optimization 2011

July 24-27, 2011

Outline

1 Introduction and motivation

2 Direct MultiSearch

3 Numerical results

4 Further improvements on DMS

5 Conclusions and references

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Derivative-free multiobjective optimization

MOO problem

$$\min_{x \in \Omega} F(x) \equiv (f_1(x), f_2(x), \ldots, f_m(x))^\top$$

where

$$\Omega = \{x \in \mathbb{R}^n : \quad \ell \leq x \leq u\}$$

$f_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}, {}_{j=1,\ldots,m}$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$ and $u \in (\mathbb{R} \cup \{+\infty\})^n$

- Several objectives, often conflicting.
- Functions with unknown derivatives.
- Expensive function evaluations, possibly subject to noise.
- Impractical to compute approximations to derivatives.

Derivative-free multiobjective optimization

MOO problem

$$\min_{x \in \Omega} F(x) \equiv (f_1(x), f_2(x), \ldots, f_m(x))^\top$$

where

$$\Omega = \{x \in \mathbb{R}^n : \quad \ell \leq x \leq u\}$$

$f_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}, j = 1, \ldots, m$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$ and $u \in (\mathbb{R} \cup \{+\infty\})^n$

- Several objectives, often conflicting.
- Functions with unknown derivatives.
- Expensive function evaluations, possibly subject to noise.
- Impractical to compute approximations to derivatives.

Derivative-free multiobjective optimization

MOO problem

$$\min_{x \in \Omega} F(x) \equiv (f_1(x), f_2(x), \ldots, f_m(x))^\top$$

where

$$\Omega = \{x \in \mathbb{R}^n : \quad \ell \leq x \leq u\}$$

$f_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}, j = 1, \ldots, m$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$ and $u \in (\mathbb{R} \cup \{+\infty\})^n$

- Several objectives, often conflicting.
- Functions with unknown derivatives.
- Expensive function evaluations, possibly subject to noise.
- Impractical to compute approximations to derivatives.

Derivative-free multiobjective optimization

MOO problem

$$\min_{x \in \Omega} F(x) \equiv (f_1(x), f_2(x), \ldots, f_m(x))^\top$$

where

$$\Omega = \{x \in \mathbb{R}^n : \quad \ell \leq x \leq u\}$$

$f_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}, {}_{j=1,\ldots,m}$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$ and $u \in (\mathbb{R} \cup \{+\infty\})^n$

- Several objectives, often conflicting.
- Functions with unknown derivatives.
- Expensive function evaluations, possibly subject to noise.
- Impractical to compute approximations to derivatives.

Derivative-free multiobjective optimization

MOO problem

$$\min_{x \in \Omega} F(x) \equiv (f_1(x), f_2(x), \ldots, f_m(x))^\top$$

where

$$\Omega = \{x \in \mathbb{R}^n : \quad \ell \leq x \leq u\}$$

$f_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}, j = 1, \ldots, m$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$ and $u \in (\mathbb{R} \cup \{+\infty\})^n$

- Several objectives, often conflicting.
- Functions with unknown derivatives.
- Expensive function evaluations, possibly subject to noise.
- Impractical to compute approximations to derivatives.

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

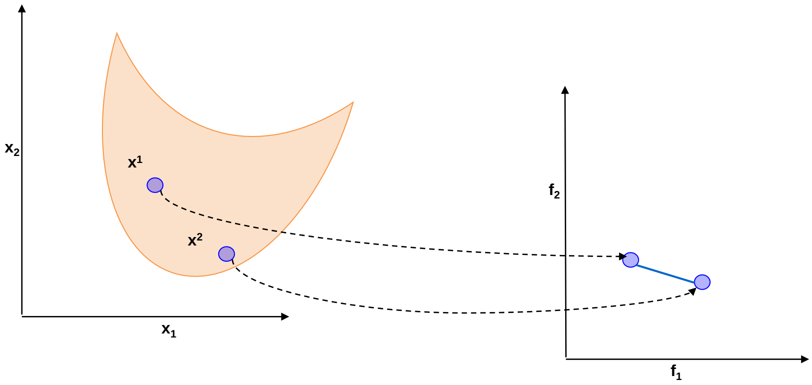
DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

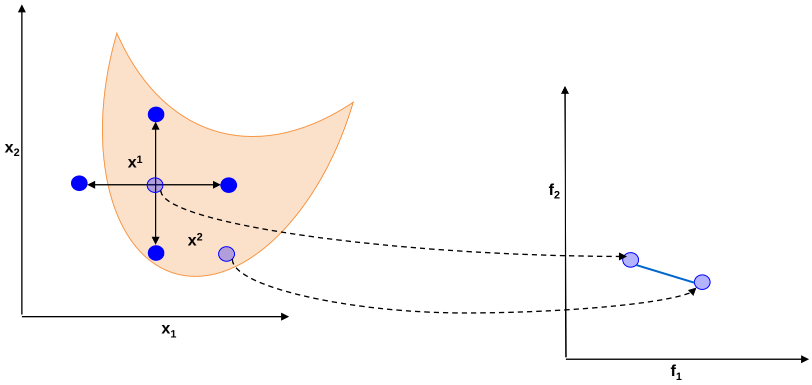
DMS algorithmic main lines

- Does not aggregate any of the objective functions.
- Generalizes ALL direct-search methods of directional type to MOO.
- Makes use of search/poll paradigm.
- Implements an optional search step (only to disseminate the search).
- Tries to capture the whole Pareto front from the polling procedure.
- Keeps a list of feasible nondominated points.
- Poll centers are chosen from the list.
- Successful iterations correspond to list changes.

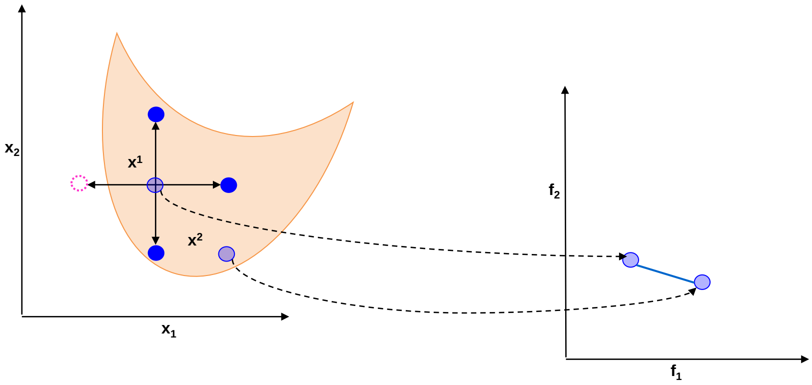
DMS example



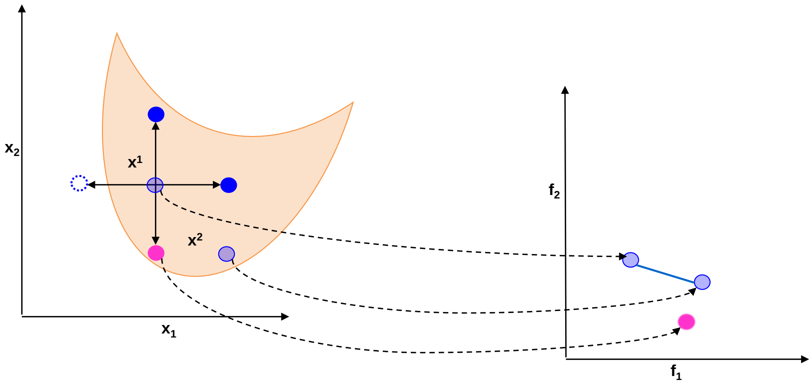
DMS example



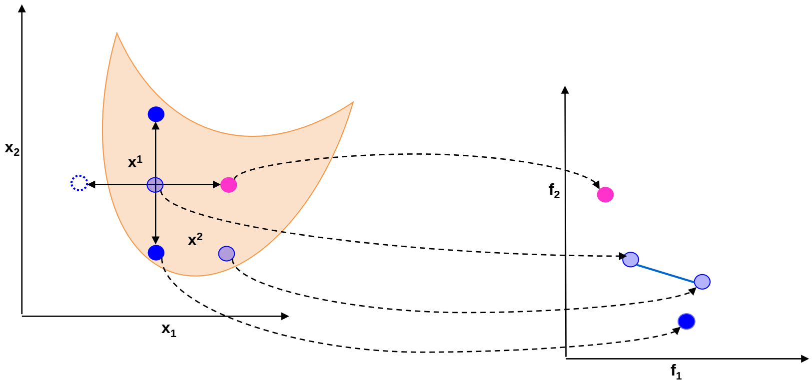
DMS example



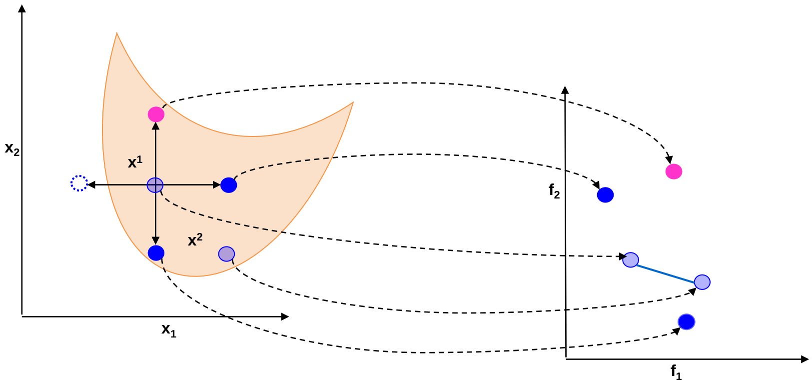
DMS example



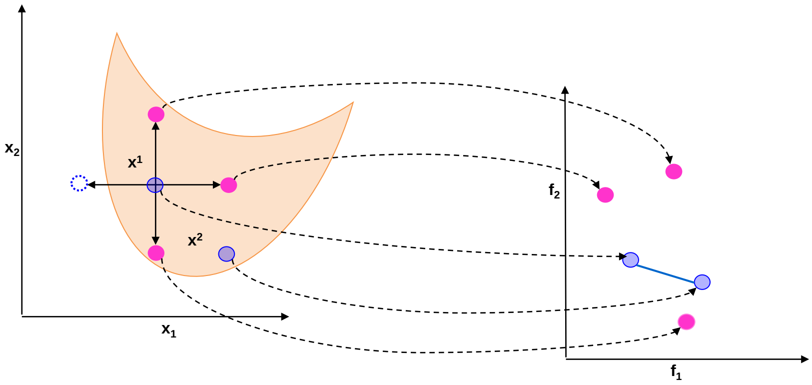
DMS example



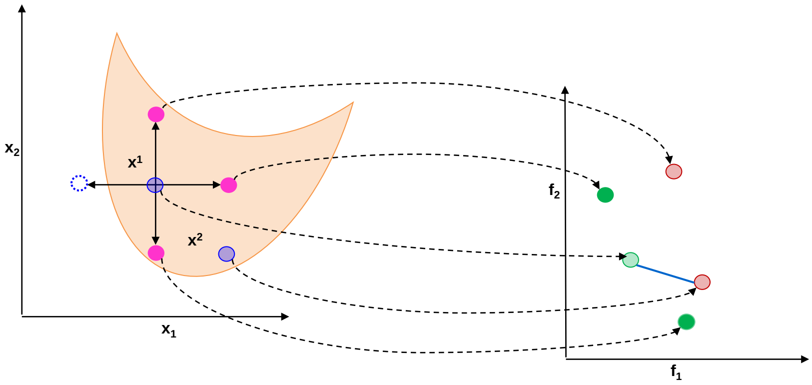
DMS example



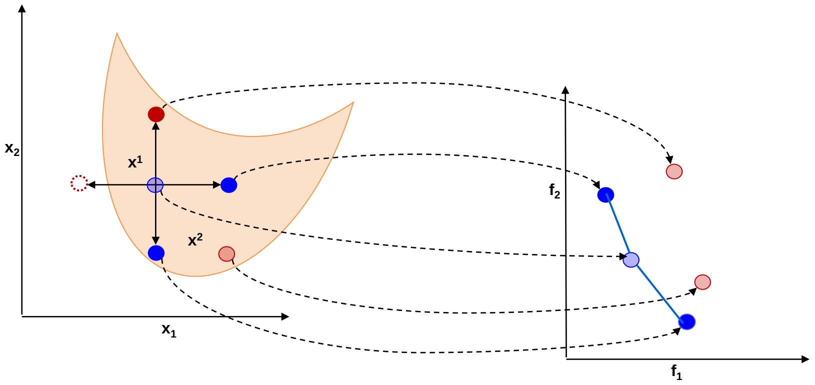
DMS example



DMS example



DMS example



DMS search & poll steps

- At each iteration considers a list of feasible nondominated points $\hookrightarrow L_k$
- Evaluate a finite set of feasible points $\hookrightarrow L_{add}$.
- Remove dominated points from $L_k \cup L_{add} \hookrightarrow L_{filtered}$.
- Select list of feasible nondominated points $\hookrightarrow L_{trial}$.
- Compare L_{trial} to L_k (success if $L_{trial} \neq L_k$, unsuccess otherwise).

DMS search & poll steps

- At each iteration considers a list of feasible nondominated points $\hookrightarrow L_k$
- Evaluate a finite set of feasible points $\hookrightarrow L_{add}$.
- Remove dominated points from $L_k \cup L_{add} \hookrightarrow L_{filtered}$.
- Select list of feasible nondominated points $\hookrightarrow L_{trial}$.
- Compare L_{trial} to L_k (success if $L_{trial} \neq L_k$, unsuccess otherwise).

DMS search & poll steps

- At each iteration considers a list of feasible nondominated points $\hookrightarrow L_k$
- Evaluate a finite set of feasible points $\hookrightarrow L_{add}$.
- Remove dominated points from $L_k \cup L_{add} \hookrightarrow L_{filtered}$.
- Select list of feasible nondominated points $\hookrightarrow L_{trial}$.
- Compare L_{trial} to L_k (success if $L_{trial} \neq L_k$, unsuccess otherwise).

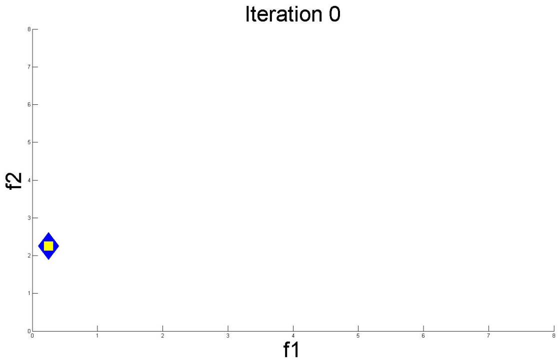
DMS search & poll steps

- At each iteration considers a list of feasible nondominated points $\hookrightarrow L_k$
- Evaluate a finite set of feasible points $\hookrightarrow L_{add}$.
- Remove dominated points from $L_k \cup L_{add} \hookrightarrow L_{filtered}$.
- Select list of feasible nondominated points $\hookrightarrow L_{trial}$.
- Compare L_{trial} to L_k (success if $L_{trial} \neq L_k$, unsuccess otherwise).

DMS search & poll steps

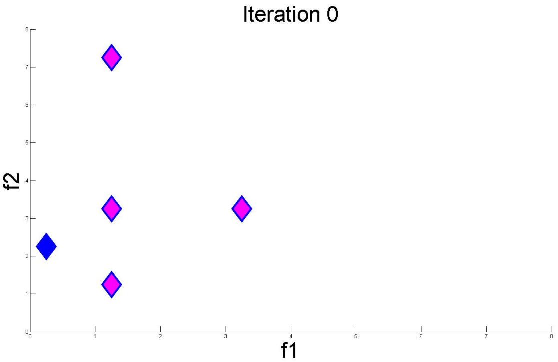
- At each iteration considers a list of feasible nondominated points $\hookrightarrow L_k$
- Evaluate a finite set of feasible points $\hookrightarrow L_{add}$.
- Remove dominated points from $L_k \cup L_{add} \hookrightarrow L_{filtered}$.
- Select list of feasible nondominated points $\hookrightarrow L_{trial}$.
- Compare L_{trial} to L_k (success if $L_{trial} \neq L_k$, unsuccess otherwise).

Numerical Example — Problem SP1 [Huband et al.]



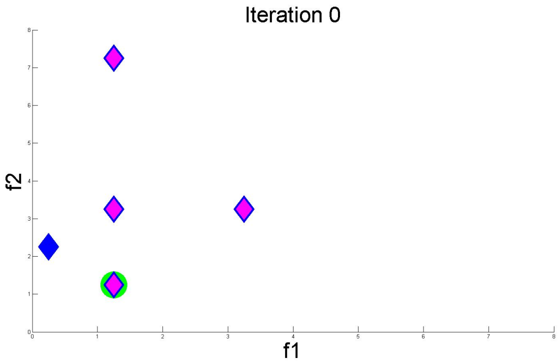
- ⧫ Evaluated points since beginning.
- ■ Current iterate list.

Numerical example — problem SP1 [Huband et al.]



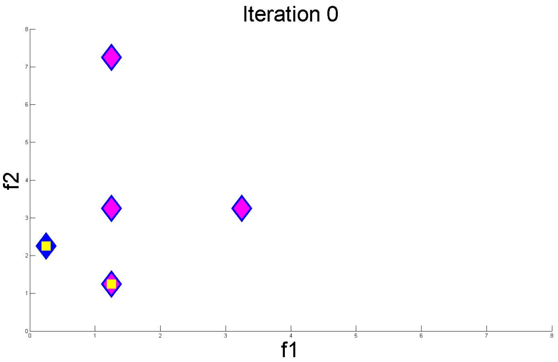
- Evaluated poll points.
- Evaluated points since beginning.

Numerical example — problem SP1 [Huband et al.]



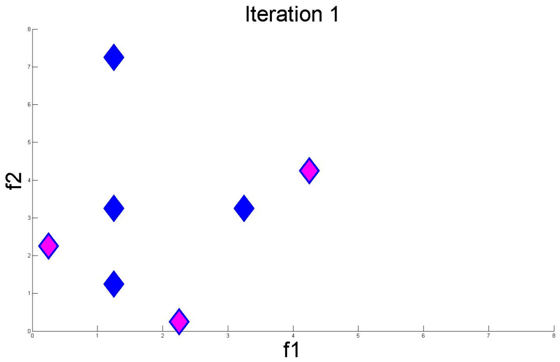
- Nondominated evaluated poll points.

Numerical example — problem SP1 [Huband et al.]



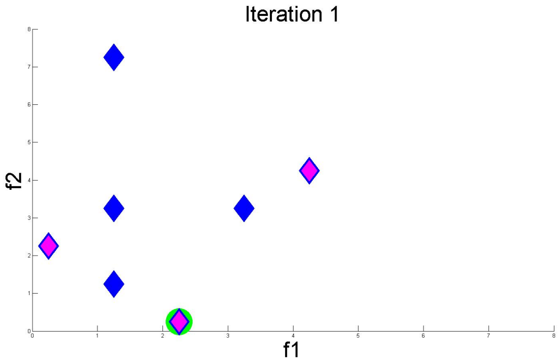
- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Numerical example — problem SP1 [Huband et al.]



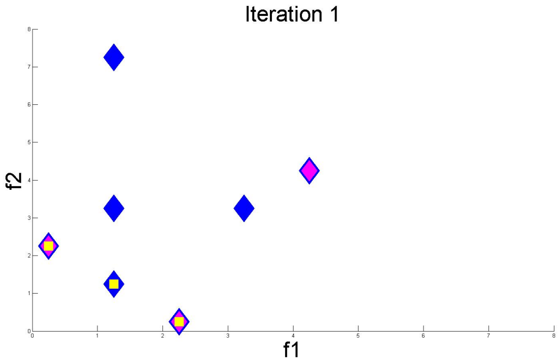
- ♦ Evaluated poll points.
- ♦ Evaluated points since beginning.

Numerical example — problem SP1 [Huband et al.]



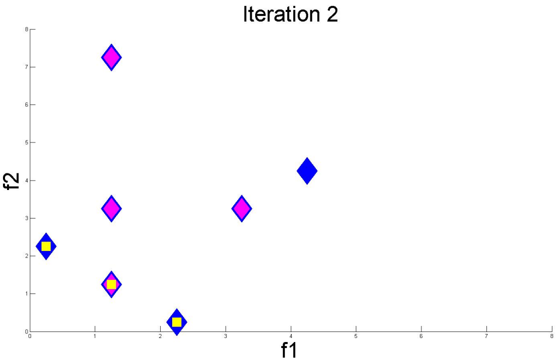
- Nondominated evaluated poll points.

Numerical example — problem SP1 [Huband et al.]



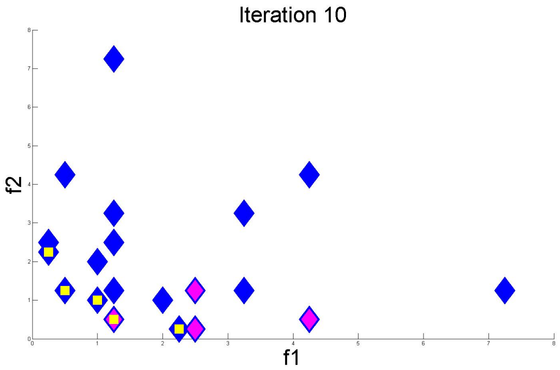
- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Numerical example — problem SP1 [Huband et al.]



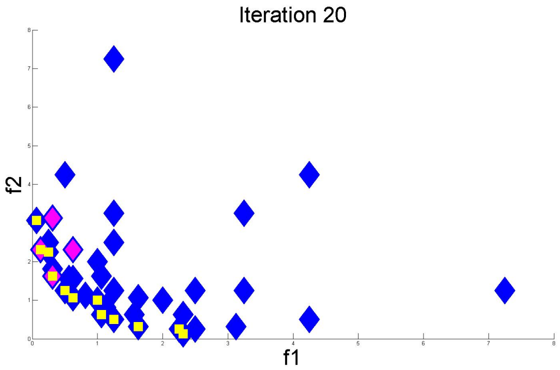
- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Numerical example — problem SP1 [Huband et al.]



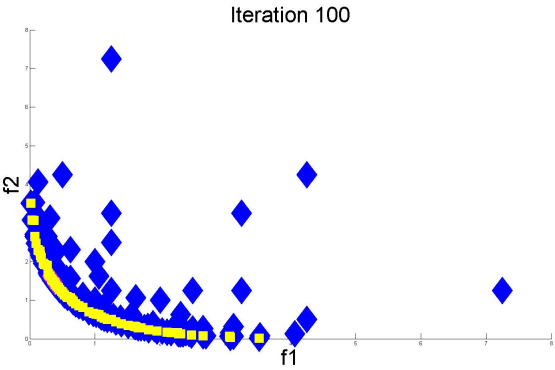
- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Numerical example — problem SP1 [Huband et al.]



- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Numerical example — problem SP1 [Huband et al.]



- Evaluated poll points.
- Evaluated points since beginning.
- Current iterate list.

Refining subsequences and directions

For both globalization strategies (using the mesh or the forcing function in the search step), one also has:

Theorem (existence of refining subsequences)

There is at least a convergent subsequence of iterates $\{x_k\}_{k \in K}$ *corresponding to unsuccessful poll steps, such that* $\alpha_k \longrightarrow 0$ *in* K.

Definition

Let x_* *be the limit point of a convergent refining subsequence.*

Refining directions for x_* *are limit points of* $\{d_k/\|d_k\|\}_{k \in K}$ *where* $d_k \in D_k$ *and* $x_k + \alpha_k d_k \in \Omega$.

Refining subsequences and directions

For both globalization strategies (using the mesh or the forcing function in the search step), one also has:

Theorem (existence of refining subsequences)

There is at least a convergent subsequence of iterates $\{x_k\}_{k \in K}$ *corresponding to unsuccessful poll steps, such that* $\alpha_k \longrightarrow 0$ *in* K.

Definition

Let x_* *be the limit point of a convergent refining subsequence.*

Refining directions for x_* *are limit points of* $\{d_k/\|d_k\|\}_{k \in K}$ *where* $d_k \in D_k$ *and* $x_k + \alpha_k d_k \in \Omega$.

Refining subsequences and directions

For both globalization strategies (using the mesh or the forcing function in the search step), one also has:

Theorem (existence of refining subsequences)

There is at least a convergent subsequence of iterates $\{x_k\}_{k \in K}$ *corresponding to unsuccessful poll steps, such that* $\alpha_k \longrightarrow 0$ *in* K.

Definition

Let x_* *be the limit point of a convergent refining subsequence.*

Refining directions for x_* *are limit points of* $\{d_k/\|d_k\|\}_{k \in K}$ *where* $d_k \in D_k$ *and* $x_k + \alpha_k d_k \in \Omega$.

Pareto-Clarke critical point

Let us focus (for simplicity) on the unconstrained case, $\Omega = \mathbb{R}^n$.

Definition

x_* is a Pareto-Clarke critical point of F (Lipschitz continuous near x_*) if

$$\forall d \in \mathbb{R}^n, \exists j = j(d), f_j^\circ(x_*; d) \geq 0.$$

Analysis of DMS

Assumption

- $\{x_k\}_{k \in K}$ refining subsequence converging to x_*.
- F Lipschitz continuous near x_*.

Theorem

If v is a refining direction for x_* then

$$\exists j = j(v) : f_j^\circ(x_*; v) \geq 0.$$

Analysis of DMS

Assumption

- $\{x_k\}_{k \in K}$ refining subsequence converging to x_*.
- F Lipschitz continuous near x_*.

Theorem

If v is a refining direction for x_* then

$$\exists j = j(v) : f_j^\circ(x_*; v) \geq 0.$$

Analysis of DMS

Assumption

- $\{x_k\}_{k \in K}$ refining subsequence converging to x_*.
- F Lipschitz continuous near x_*.

Theorem

If v is a refining direction for x_* then

$$\exists j = j(v) : f_j^\circ(x_*; v) \geq 0.$$

Analysis of DMS

Assumption

- $\{x_k\}_{k \in K}$ refining subsequence converging to x_*.
- F Lipschitz continuous near x_*.

Theorem

If v is a refining direction for x_* then

$$\exists j = j(v) : f_j^\circ(x_*; v) \geq 0.$$

Convergence analysis of DMS

Theorem

If the set of refining directions for x_* is dense in $\mathbb{R}^n$, then x_* is a Pareto-Clarke critical point.

Notes

- When $m = 1$, the presented results coincide with the ones reported for direct search.
- This convergence analysis is valid for multiobjective problems with general nonlinear constraints.

Convergence analysis of DMS

Theorem

If the set of refining directions for x_* is dense in $\mathbb{R}^n$, then x_* is a Pareto-Clarke critical point.

Notes

- When $m = 1$, the presented results coincide with the ones reported for direct search.
- This convergence analysis is valid for multiobjective problems with general nonlinear constraints.

Convergence analysis of DMS

Theorem

If the set of refining directions for x_* is dense in $\mathbb{R}^n$, then x_* is a Pareto-Clarke critical point.

Notes

- When $m = 1$, the presented results coincide with the ones reported for direct search.
- This convergence analysis is valid for multiobjective problems with general nonlinear constraints.

Convergence analysis of DMS

Theorem

If the set of refining directions for x_* is dense in $\mathbb{R}^n$, then x_* is a Pareto-Clarke critical point.

Notes

- When $m = 1$, the presented results coincide with the ones reported for direct search.
- This convergence analysis is valid for multiobjective problems with general nonlinear constraints.

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

Numerical testing framework

Problems

- 100 bound constrained MOO problems (AMPL models available at http://www.mat.uc.pt/dms).
- Number of variables between 1 and 30.
- Number of objectives between 2 and 4.

Solvers

- DMS tested against 8 different MOO solvers (complete results available at http://www.mat.uc.pt/dms).
- Results reported only for
  AMOSA – simulated annealing code.
  BIMADS – based on mesh adaptive direct search algorithm.
  NSGA-II (C version) – genetic algorithm code.

All solvers tested with default values.

Numerical testing framework

Problems

- 100 bound constrained MOO problems (AMPL models available at http://www.mat.uc.pt/dms).
- Number of variables between 1 and 30.
- Number of objectives between 2 and 4.

Solvers

- DMS tested against 8 different MOO solvers (complete results available at http://www.mat.uc.pt/dms).
- Results reported only for
  AMOSA – simulated annealing code.
  BIMADS – based on mesh adaptive direct search algorithm.
  NSGA-II (C version) – genetic algorithm code.

All solvers tested with default values.

Numerical testing framework

Problems

- 100 bound constrained MOO problems (AMPL models available at http://www.mat.uc.pt/dms).
- Number of variables between 1 and 30.
- Number of objectives between 2 and 4.

Solvers

- DMS tested against 8 different MOO solvers (complete results available at http://www.mat.uc.pt/dms).
- Results reported only for
  AMOSA – simulated annealing code.
  BIMADS – based on mesh adaptive direct search algorithm.
  NSGA-II (C version) – genetic algorithm code.

All solvers tested with default values.

Numerical testing framework

Problems

- 100 bound constrained MOO problems (AMPL models available at http://www.mat.uc.pt/dms).
- Number of variables between 1 and 30.
- Number of objectives between 2 and 4.

Solvers

- DMS tested against 8 different MOO solvers (complete results available at http://www.mat.uc.pt/dms).
- Results reported only for
  AMOSA – simulated annealing code.
  BIMADS – based on mesh adaptive direct search algorithm.
  NSGA-II (C version) – genetic algorithm code.

All solvers tested with default values.

Numerical testing framework

Problems

- 100 bound constrained MOO problems (AMPL models available at http://www.mat.uc.pt/dms).
- Number of variables between 1 and 30.
- Number of objectives between 2 and 4.

Solvers

- DMS tested against 8 different MOO solvers (complete results available at http://www.mat.uc.pt/dms).
- Results reported only for
  AMOSA – simulated annealing code.
  BIMADS – based on mesh adaptive direct search algorithm.
  NSGA-II (C version) – genetic algorithm code.

All solvers tested with default values.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \ \ -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \ -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \;\; -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \ \ -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \ \ -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \;\; -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

DMS numerical options

- No search step.
- List initialization: sample along the line ℓ–u.
- List selection: all current feasible nondominated points.
- List ordering: new points added at the end of the list, poll center moved to the end of the list.
- Positive basis: $[I \ \ -I]$.
- Step size parameter: $\alpha_0 = 1$, halved at unsuccessful iterations.
- Stopping criteria: minimum step size of 10^{-3} or a maximum of 20000 function evaluations.

Performance metrics — Purity

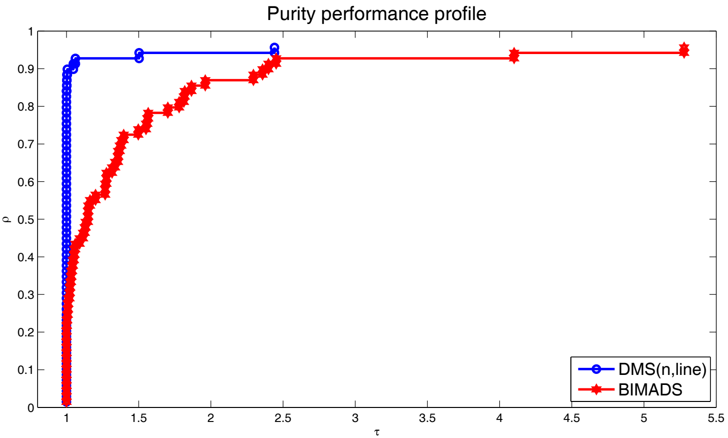
$F_{p,s}$ (approximated Pareto front computed by solver s for problem p).

F_p (approximated Pareto front computed for problem p, using results for all solvers).

Purity value for solver s on problem p:

$$\frac{|F_{p,s} \cap F_p|}{|F_{p,s}|}.$$

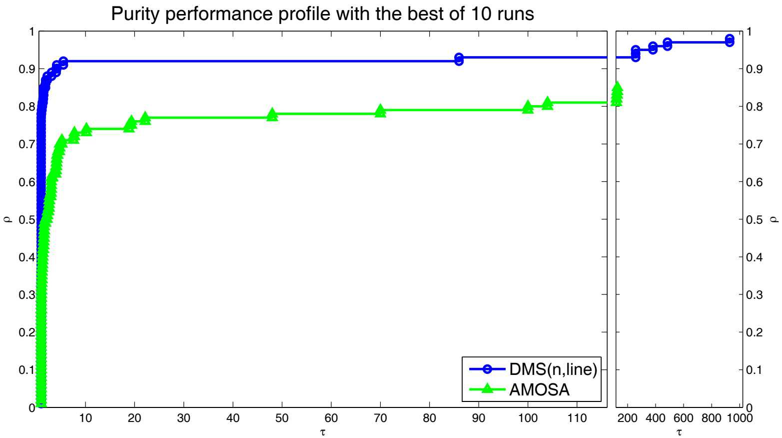
Comparing DMS to other solvers (Purity)



Purity Metric (percentage of points generated in the reference Pareto front)

$$t_{p,s} = \frac{|F_{p,s}|}{|F_{p,s} \cap F_p|}$$

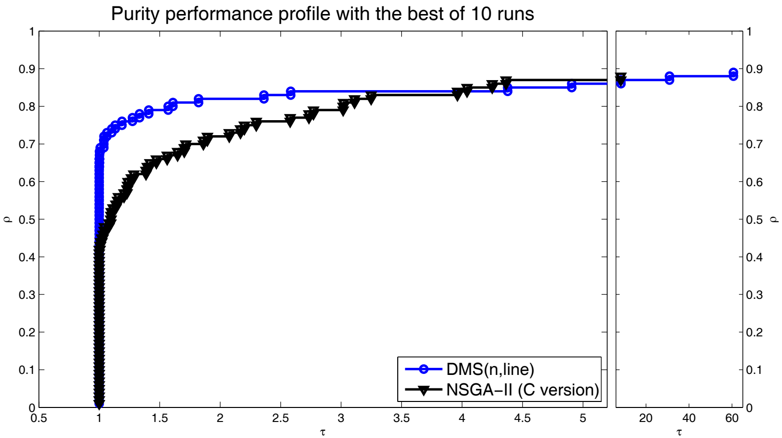
Comparing DMS to other solvers (Purity)



Purity Metric (percentage of points generated in the reference Pareto front)

$$t_{p,s} = \frac{|F_{p,s}|}{|F_{p,s} \cap F_p|}$$

Comparing DMS to other solvers (Purity)



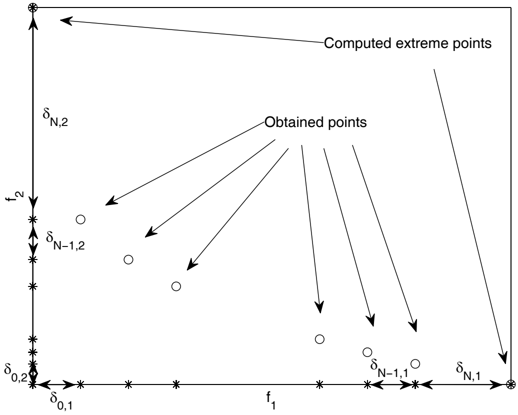
Purity Metric (percentage of points generated in the reference Pareto front)

$$t_{p,s} = \frac{|F_{p,s}|}{|F_{p,s} \cap F_p|}$$

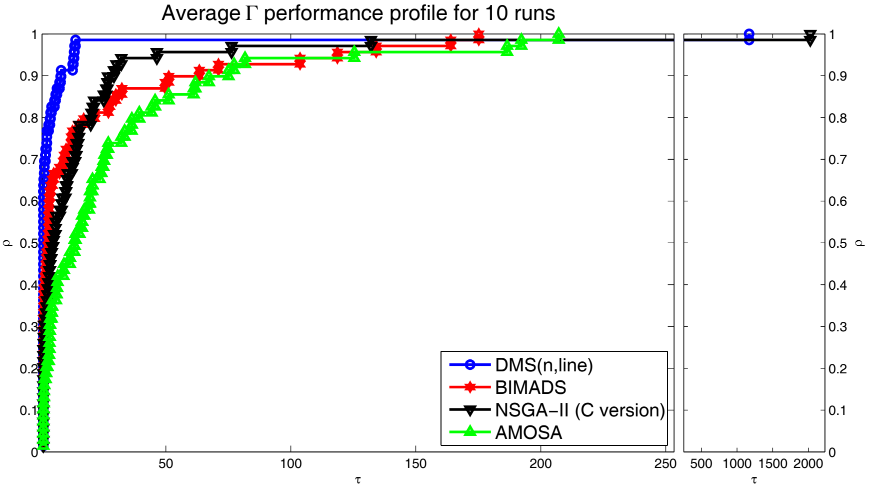
Performance metrics — Spread

Gamma Metric (largest gap in the Pareto front)

$$\Gamma_{p,s} = \max_{j \in \{1,\ldots,m\}} \left(\max_{i \in \{0,\ldots,N\}} \{\delta_{i,j}\} \right)$$



Comparing DMS to other solvers (Spread)



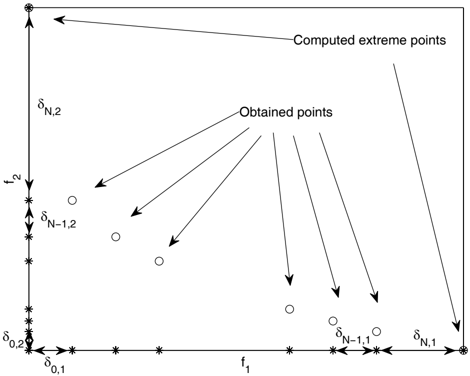
Gamma Metric (largest gap in the Pareto front)

Performance metrics — Spread

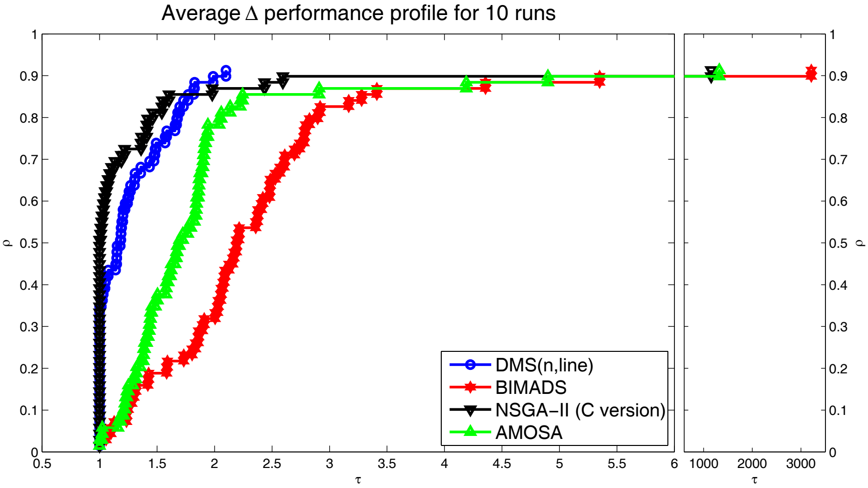
Delta Metric (uniformity of gaps in the Pareto front)

$$\Delta_{p,s} = \max_{j \in \{1,\ldots,m\}} \left(\frac{\delta_{0,j} + \delta_{N,j} + \sum_{i=1}^{N-1} |\delta_{i,j} - \bar{\delta}_j|}{\delta_{0,j} + \delta_{N,j} + (N-1)\bar{\delta}_j} \right)$$

where $\bar{\delta}_j$, for $j = 1, \ldots, m$, is the $\delta_{i,j}$'s average.

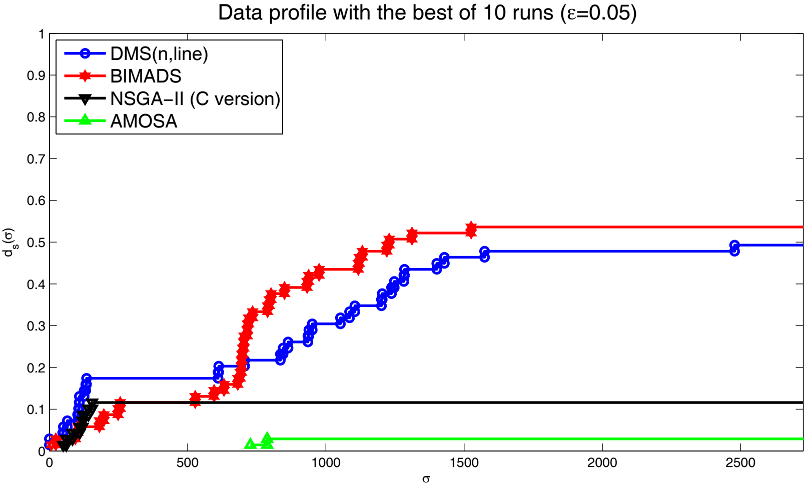


Comparing DMS to other solvers (Spread)



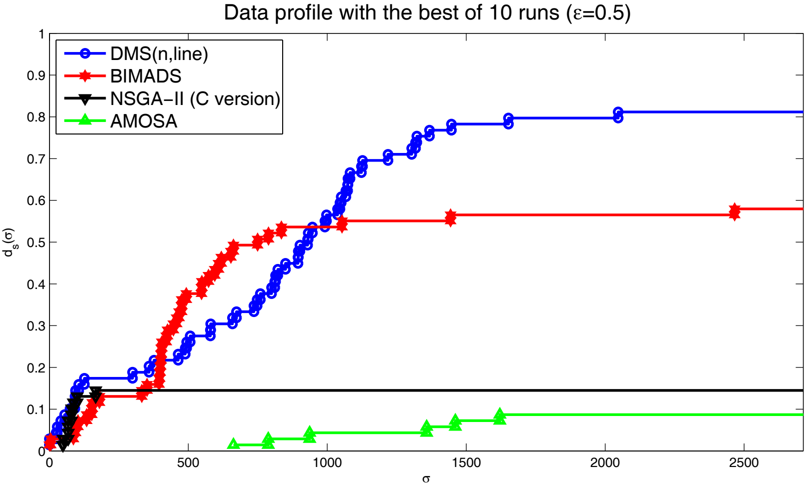
Delta Metric (uniformity of gaps in the Pareto front)

Comparing DMS to other solvers



maximum function evaluations = 5000

Comparing DMS to other solvers

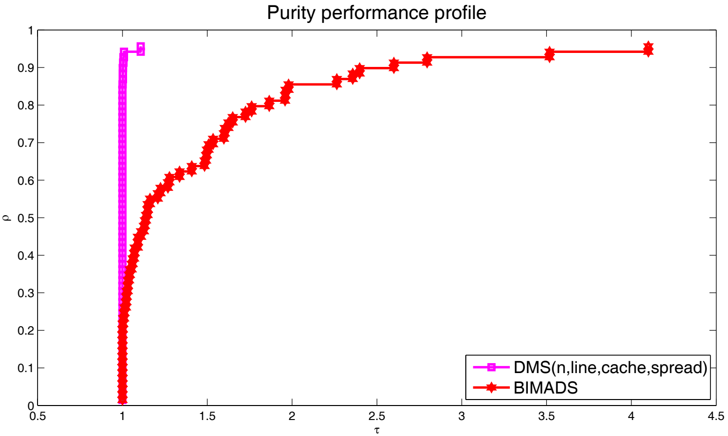


maximum function evaluations = 5000

Outline

1. Introduction and motivation
2. Direct MultiSearch
3. Numerical results
4. Further improvements on DMS
5. Conclusions and references

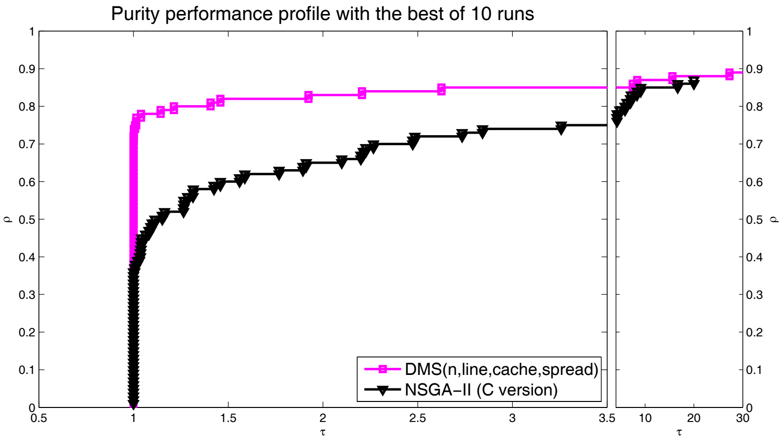
Comparing DMS to other solvers (Purity)



Purity Metric (percentage of points generated in the reference Pareto front)

$$t_{p,s} = \frac{|F_{p,s}|}{|F_{p,s} \cap F_p|}$$

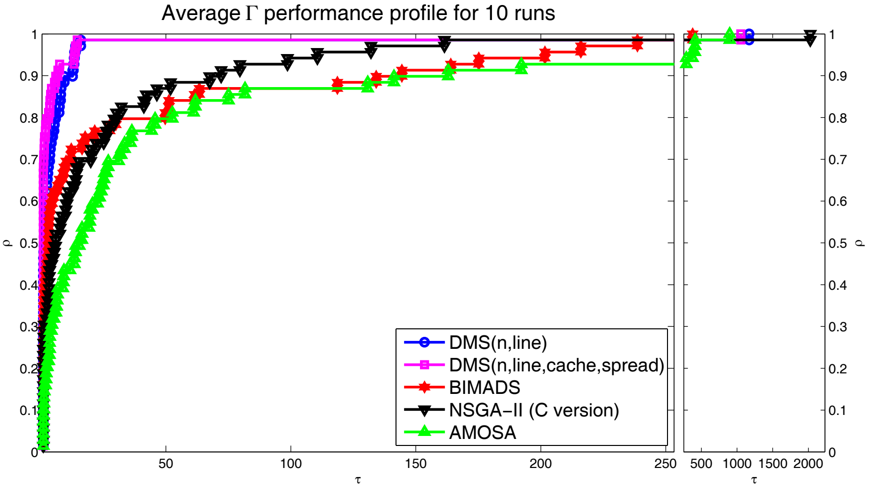
Comparing DMS to other solvers (Purity)



Purity Metric (percentage of points generated in the reference Pareto front)

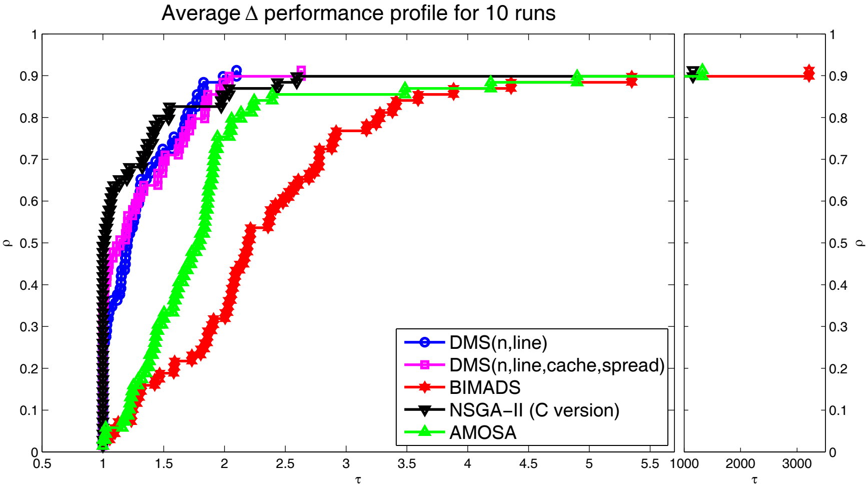
$$t_{p,s} = \frac{|F_{p,s}|}{|F_{p,s} \cap F_p|}$$

Comparing DMS to other solvers (Spread)



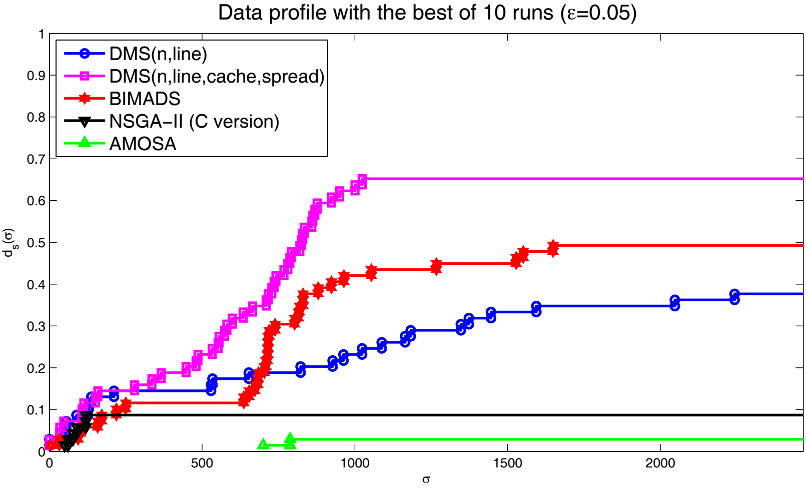
Gamma Metric (largest gap in the Pareto front)

Comparing DMS to other solvers (Spread)



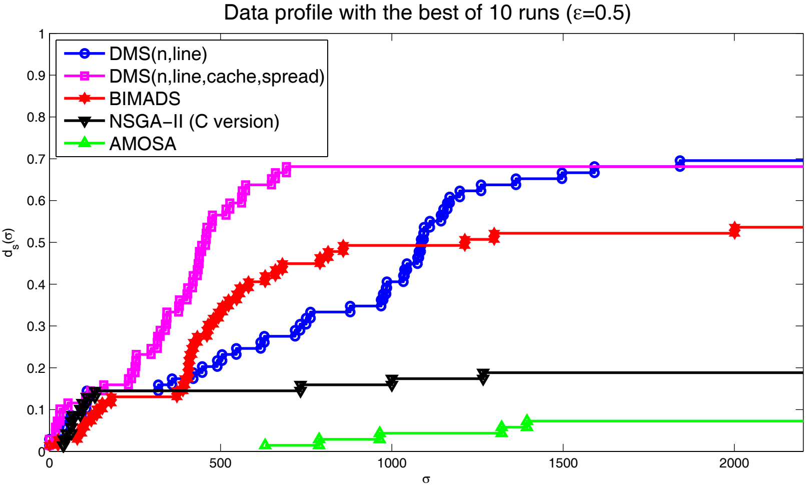
Delta Metric (uniformity of gaps in the Pareto front)

Comparing DMS to other solvers



maximum function evaluations = 5000

Comparing DMS to other solvers



maximum function evaluations = 5000

Outline

1 Introduction and motivation

2 Direct MultiSearch

3 Numerical results

4 Further improvements on DMS

5 Conclusions and references

Conclusions and references

- Development and analysis of a novel approach (Direct MultiSearch) for MOO, generalizing ALL direct-search methods.

- Direct MultiSearch (DMS) exhibits highly competitive numerical results for MOO.

DMS (Matlab implementation) and problems (coded in AMPL) freely available at: `http://www.mat.uc.pt/dms`.

A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente, Direct multisearch for multiobjective optimization, to appear, SIAM Journal on Optimization.

Conclusions and references

- Development and analysis of a novel approach (Direct MultiSearch) for MOO, generalizing ALL direct-search methods.

- Direct MultiSearch (DMS) exhibits highly competitive numerical results for MOO.

DMS (Matlab implementation) and problems (coded in AMPL) freely available at: `http://www.mat.uc.pt/dms`.

A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente, Direct multisearch for multiobjective optimization, to appear, SIAM Journal on Optimization.

Conclusions and references

- Development and analysis of a novel approach (Direct MultiSearch) for MOO, generalizing ALL direct-search methods.

- Direct MultiSearch (DMS) exhibits highly competitive numerical results for MOO.

DMS (Matlab implementation) and problems (coded in AMPL) freely available at: `http://www.mat.uc.pt/dms`.

A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente, Direct multisearch for multiobjective optimization, to appear, SIAM Journal on Optimization.

Conclusions and references

- Development and analysis of a novel approach (Direct MultiSearch) for MOO, generalizing ALL direct-search methods.

- Direct MultiSearch (DMS) exhibits highly competitive numerical results for MOO.

DMS (Matlab implementation) and problems (coded in AMPL) freely available at: `http://www.mat.uc.pt/dms`.

A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente, Direct multisearch for multiobjective optimization, to appear, SIAM Journal on Optimization.