

5th Annual MOPTA Conference
Modeling and Optimization: Theory and Applications
University of Windsor, Ontario, Canada
July 25-27, 2005

Evaluating Optimization Software

Jorge J. Moré

Mathematics and Computer Science Division
Argonne National Laboratory



Motivation

- I. A scientist tells you that they have been using the codes in *Numerical Recipes in C*, and wonders if the optimization community has developed better codes.
- II. NEOS users want to know what solver they should buy for solving problems in . . .
- III. A program manager (or dean) asks about specific instances where optimization algorithms have made an impact in solving realistic problems.

Benchmarking: Why?

- ▶ Developing prototype implementations
- ▶ Evaluation of algorithmic options
- ▶ Comparing optimization solvers
- ▶ Formalizing claims like
 - ◆ *I tried that but it did not work*
 - ◆ *Trust region methods are more robust than line searches*
 - ◆ *Interior points methods are superior to active set methods*

Benchmarking

A **benchmark** is defined by

- ▶ \mathcal{P} – Benchmark problems
- ▶ \mathcal{T} – Convergence test
- ▶ \mathcal{S} – Set of solvers

Choose a performance measure $t_{p,s}$ for each $p \in \mathcal{P}$ and $s \in \mathcal{S}$.

Performance Measures

- ▶ Computing time
- ▶ Number of iterations
- ▶ Number of function evaluations

Benchmarking Proposals

Proposal: Compute average or cumulative totals of metric

Objection: Sensitive to results on a small number of problems

Proposal: Rank solvers by the number of k-th place entries.

Objection: No information on the size of improvement

Proposal: Compute the number of wins by a fixed percentage

Objection: Dependent on the subjective choice of a parameter

Proposal: Compute medians or quartiles of a metric

Objection: Information between quartiles is lost

Performance Ratios

A basic question. Given a solver $s \in \mathcal{S}$, what is the fraction of problems where the performance ratio is at most τ , where the performance ratio is

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : 1 \leq s \leq n_s\}}.$$

Special cases

- ▶ fastest solver: $r_{p,s} = 1$
- ▶ solver fails: $r_{p,s} = \infty$

Performance Profiles

A **performance profile** is the distribution function of a performance metric.

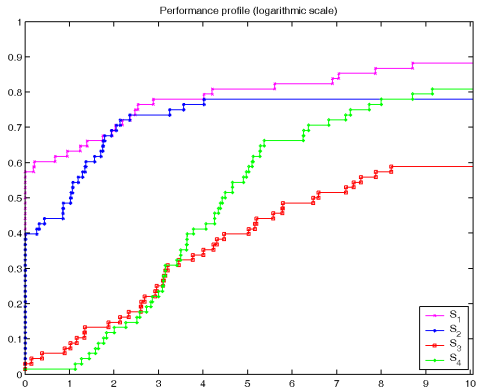
- ▶ Compute the performance ratio

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : 1 \leq s \leq n_s\}}.$$

- ▶ Compute the probability distribution function

$$\rho_s(\tau) = \frac{1}{n_p} \text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\}$$

Performance Profiles



$$\mathcal{P} = \text{COPS}, \quad \mathcal{S} = \{S_1, \dots, S_4\}, \quad \mathcal{T} = \square$$

Properties of Performance Profiles

The performance profile $\rho_s : \mathbb{R} \mapsto [0, 1]$ is a non-decreasing, piecewise constant function, continuous from the right at each breakpoint.

- ▶ Information on the size of improvement is provided
- ▶ Does not depend on the subjective choice of a parameter
- ▶ Can be used to compare more than two solvers.
- ▶ Not sensitive to small changes in the data
- ▶ Not sensitive to the data on a small number of problems

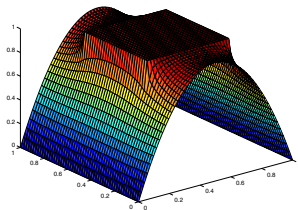
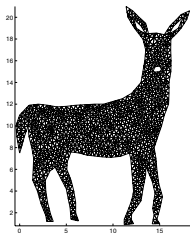
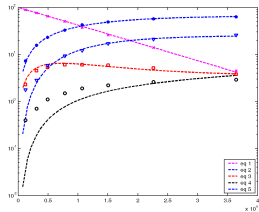
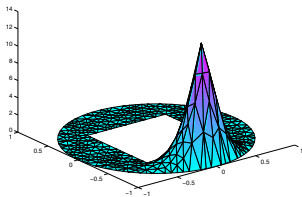
The Standard Caveats of Optimization Benchmarking

- ▶ Conclusions apply only to the benchmark problems \mathcal{P}
- ▶ Results depend on the convergence test \mathcal{T}
- ▶ Results are time-dependent since solvers \mathcal{S} evolve and improve
- ▶ Timings depend on the architecture, compilers, libraries, . . .

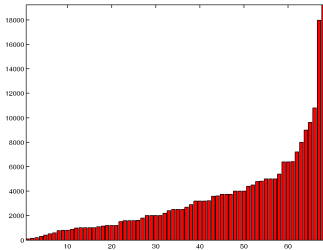
Benchmark Problems

- ▶ AMPL format
 - ◆ COPS
 - ◆ Nonlinear Optimization Models
 - ◆ MacMPEC
- ▶ GAMS format
 - ◆ GAMS Model Library
 - ◆ Handbook of Test Problems in Local and Global Optimization
- ▶ SIF format
 - ◆ CUTEr
- ▶ Fortran
 - ◆ MINPACK-2 Model Problems

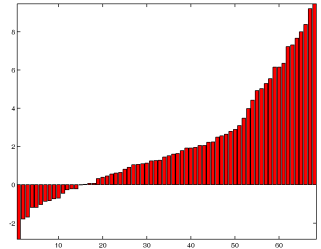
Benchmarking with COPS



COPS Data



Number of variables



Minimal (\log_2) solution times

Trivia question. What is the percentage of CUTeR problems that require more than 1 second of computing time?

Sensitivity of Performance Profiles

What if ...

- ▶ Times are noisy
- ▶ We drop a solver from the benchmark
- ▶ We select a random subset of the problems
- ▶ We only consider hard problems
- ▶ We only consider problems with n large

Claim. Timing data can be obtained with 10% accuracy for reasonable problems

Sensitivity of Performance Profiles

Theorem. Let r_i and \hat{r}_i for $1 \leq i \leq n_p$ be performance ratios for some solver, and let ρ and $\hat{\rho}$ be, respectively, the performance profiles defined by these ratios. If

$$|r_i - \hat{r}_i| \leq \epsilon, \quad 1 \leq i \leq n_p$$

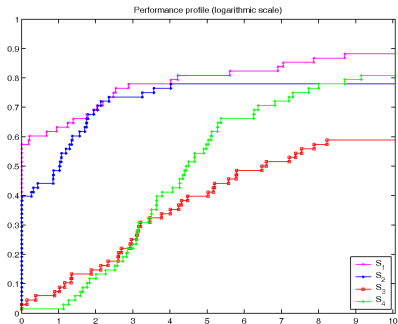
for some $\epsilon > 0$, then

$$\int_1^\infty |\rho(t) - \hat{\rho}(t)| dt \leq \epsilon$$

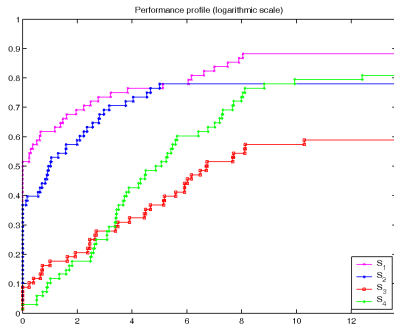
Note. If $\hat{t}_i = t_i(1 + \epsilon_i)$, $|\epsilon_i| < \epsilon$, then $|r_i - \hat{r}_i| \leq \theta|r_i|$, where

$$\theta = \frac{2\epsilon}{(1 - \epsilon)}$$

Performance Profiles: Timings

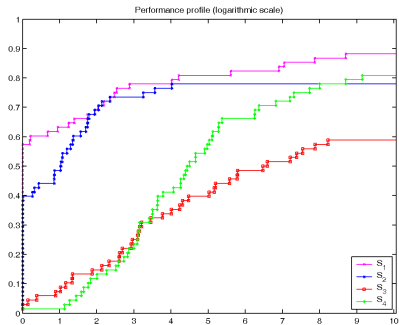


Profiles for S_1, \dots, S_4

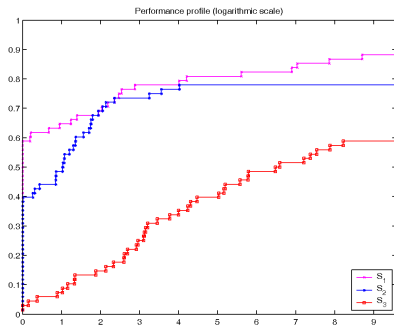


Profiles when $t_i \leftarrow t_i(1 + \varepsilon_i)$
 $|\varepsilon_i| \leq \varepsilon$ random, $\varepsilon = 0.1$

Performance Profiles: Solvers

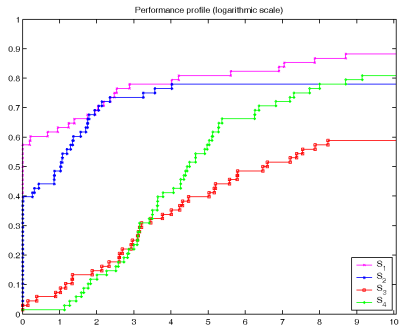


Profiles for S_1, \dots, S_4

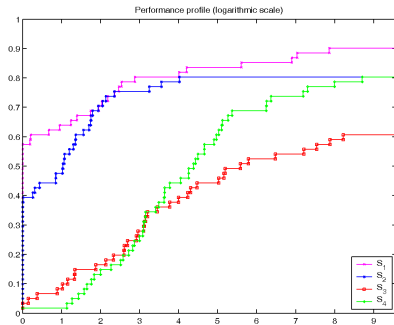


Profiles without solver S_4

Performance Profiles: Random (90%) subsets

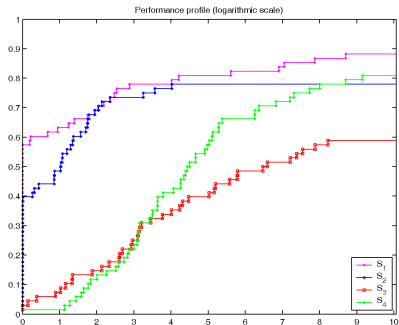


Profiles for S_1, \dots, S_4

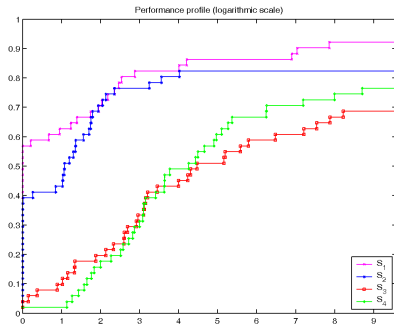


Profiles for random (90%) subset

Performance Profiles: Random (75%) subsets

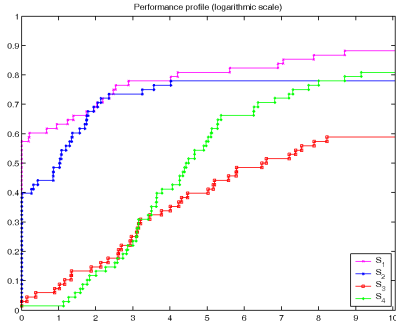


Profiles for S_1, \dots, S_4

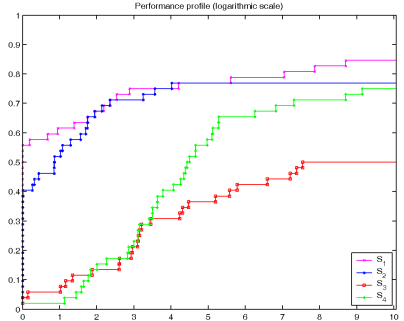


Profiles for random (75%) subset

Performance Profiles: Hard Problems



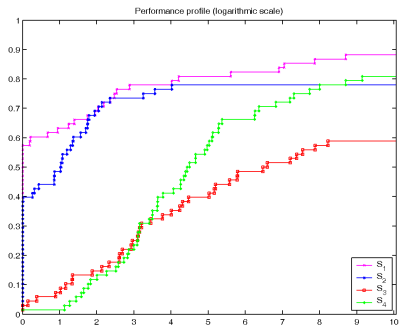
Profiles for S_1, \dots, S_4



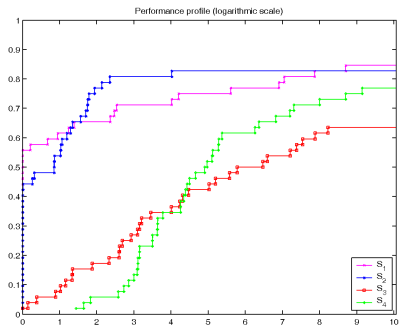
Profiles for hardest (75%) subset

Definition. The *hardness* of $p \in \mathcal{P}$ is $\min \{t_{p,s} : s \in \mathcal{S}\}$.

Performance Profiles: Large Problems



Profiles for S_1, \dots, S_4



Profiles for $n \geq q_1$

Quartiles for n are $(q_1, q_2, q_3) = (1098, 2500, 4398)$

Convergence Tests for Constrained Optimization

Trial Ballon. For the optimization problem

$$\min \{f(x) : l \leq c(x) \leq u\}$$

consider the convergence test

$$\nu(x) = \|\nabla f(x) - \nabla c(x)\lambda\|$$

Issues

- ▶ How do we compute multipliers?
- ▶ What happens if we do not have access to the solver code?
- ▶ Should we modify solver code?

Elements of a Convergence Test

- ▶ Choose a set $\mathcal{A}_\tau(x)$ of τ -active constraints
- ▶ Determine the cone $S_\tau(x)$ associated with $\mathcal{A}_\tau(x)$
- ▶ Choose approximate multipliers $\lambda_\tau(x) \in S_\tau(x)$.

$$\lambda_\tau(x) = \operatorname{argmin} \left\{ \|\nabla f(x) - \nabla c(x)\lambda\| : \lambda \in S_\tau(x) \right\}$$

- ▶ Define a stationarity measure

$$\nu_\tau(x) = \|\nabla f(x) - \nabla c(x)\lambda_\tau(x)\|$$

- ▶ Define a feasibility measure

Properties of the Convergence Test

$$\nu_\tau(x) = \|\nabla f(x) - \nabla c(x)\lambda_\tau(x)\|$$

Properties

- ▶ $\nu_\tau(x) \geq 0$
- ▶ If $\{x_k\} \rightarrow x^*$ and x^* is a KKT point then $\{\nu_\tau(x_k)\} \rightarrow 0$
- ▶ If $\{x_k\} \rightarrow x^*$, $\{\nu_\tau(x_k)\} \rightarrow 0$, and x^* satisfies the MFCQ, then there are multiplier estimates with

$$\nabla f(x^*) = \nabla c(x^*)\lambda^*, \quad \lambda^* \in S_\tau(x^*)$$

Counterexamples

I. If $\{x_k\} \rightarrow x^*$ and x^* is a KKT point, but $\tau = 0$, then $\{\nu_\tau(x_k)\}$ may stay bounded away from zero.

Example

Consider any sequence $\{x_k\}$ with $\mathcal{A}(x_k) = \mathcal{A}_\tau(x_k)$ empty

II. If $\{x_k\} \rightarrow x^*$, but x^* does not satisfy the MFCQ, then we may have $\{\nu_\tau(x_k)\} \rightarrow 0$ at a non-KKT point x^* .

Example

$$\min \left\{ \xi : \frac{1}{2}\xi^2 \geq 0 \right\}$$

Enforcing a Uniform Convergence Test

Let $\nu : \mathbb{R}^n \mapsto \mathbb{R}$ be a convergence test.

Choose a desirable accuracy level τ^* .

Set τ to default tolerance values.

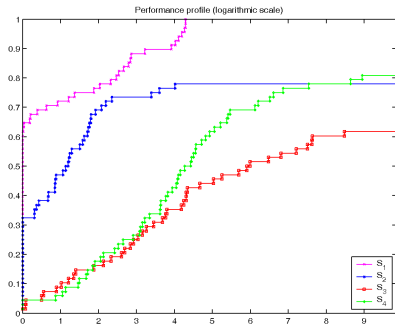
For $k = 1, 2, \dots$

- ▶ Set solver tolerances to τ and solve
- ▶ If $\nu(x) \leq \tau^*$ then **exit**
- ▶ $\tau = \tau/10$

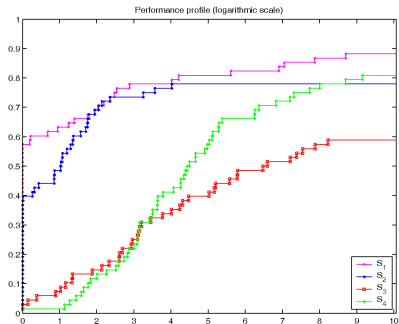
end

Note. We do not assume that solutions times increase as tolerances are decreased

Performance Profiles

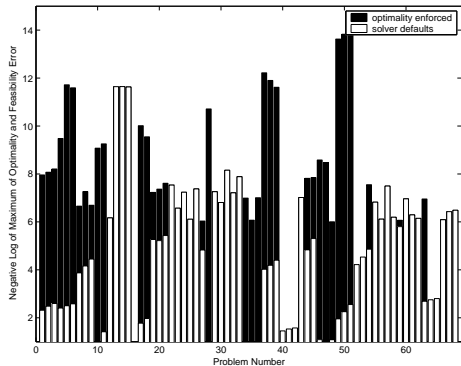


Default convergence test

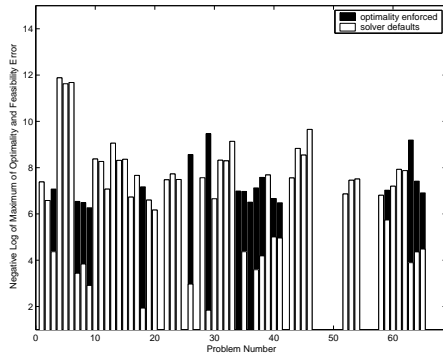


Uniform convergence test

Enforcing a Uniform Convergence Test: Details



Solver S_1



Solver S_2

Convergence Tests and Scale Invariance

Is the stationarity measure

$$\nu_{\tau}(x) = \|\nabla f(x) - \nabla c(x)\lambda_{\tau}(x)\|$$

scale invariant?

Motivation

- ▶ Tolerances should be independent of problem scaling
- ▶ Users want to choose the units in the problem and yet retain the same behavior in the optimization algorithm.

Scale Invariance

Original problem

$$\min \{f(x) : l \leq c(x) \leq u\}$$

Transformed problem

$$\min \{ \hat{f}(x) : \hat{l} \leq \hat{c}(x) \leq \hat{u} \}$$

Scaling transformations

- ▶ $\hat{f} = \alpha f, \quad \alpha > 0$
- ▶ $(\hat{c}, \hat{l}, \hat{u}) = \beta(c, l, u), \quad \beta > 0$
- ▶ $\hat{f}(x) = f(Sx)$ and $\hat{c}(x) = c(Sx),$

where S is a nonsingular and diagonal matrix.

Observation. The vector x^* is a solution of the original problem if and only if $S^{-1}x^*$ is a solution of the transformed problem.

What is Scale Invariance?

Original problem

$$\min \{f(x) : l \leq c(x) \leq u\}$$

Transformed problem

$$\min \left\{ \hat{f}(x) : \hat{l} \leq \hat{c}(x) \leq \hat{u} \right\}$$

The algorithm is **scale invariant** if

- ▶ $\hat{x}_k = x_k$ provided $\hat{x}_0 = x_0$
- ▶ $\hat{x}_k = S^{-1}x_k$ provided $\hat{x}_0 = S^{-1}x_0$

The **invariance group** of an algorithm is the set of scalars α, β and matrices S for which the algorithm is scale invariant.

Scale Invariant Algorithms

Claim

Scale invariance is a desirable attribute of algorithms and software.

Exercise. What is the invariance group for ...

- ▶ Nelder-Mead
- ▶ Steepest descent
- ▶ Fletcher-Reeves
- ▶ BFGS
- ▶ Newton methods
- ▶ Exact penalty methods
- ▶ Augmented Lagrangians
- ▶ SQP methods

Concluding Remarks

What is in the future?

- ▶ Distributed computing: TAO
- ▶ Self-adaptive optimization software
- ▶ Application-specific modeling languages
- ▶ Cyberinfrastructure



