An Asynchronously Parallel Optimization Solver for Finding Multiple Minima

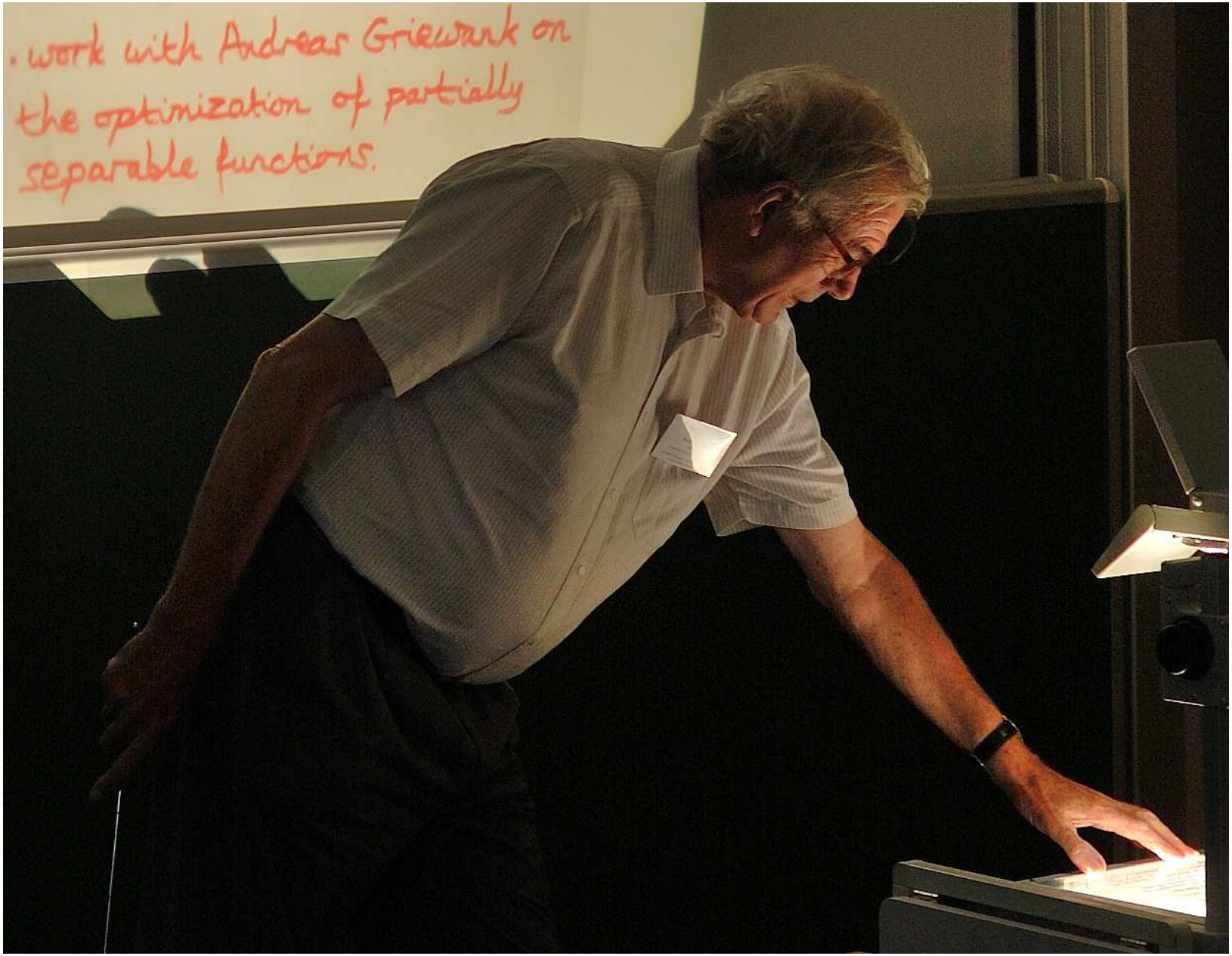
Stefan Wild

Joint work with **Jeff Larson**

Argonne National Laboratory
Mathematics and Computer Science Division

January 4, 2016

Motivation: Michael J.D. Powell



Michal Kocvara, 2011

Use of Software for "Blackbox" Optimization

Google Scholar (1/1/2016)

BOBYQA (2009) **293**

NEWUOA (2006) **281** + **105** (2008)

UOBYQA (2002) **288**

COBYLA (1994) **309**

Use of Software for "Blackbox" Optimization

Google Scholar (1/1/2016)

BOBYQA (2009) **293**

NEWUOA (2006) **281** + **105** (2008)

UOBYQA (2002) **288**

COBYLA (1994) **309**

CMA-ES (2001) **1,857** + **729** (2003) + **572** (2005) + **685** (2006) + ...

NSGA-II (2002) **15,772** + **3,203** (2000) + ...

Others Particle swarm, ant colony, firefly, ...

Use of Software for "Blackbox" Optimization

Google Scholar (1/1/2016)

BOBYQA (2009) **293**

NEWUOA (2006) **281** + **105** (2008)

UOBYQA (2002) **288**

COBYLA (1994) **309**

- *CONDOR, a new parallel, constrained extension of Powell's UOBYQA algorithm* [Vanden Berghen & Bersini (JCAM, 2005)]: **123** (unavailable!)

CMA-ES (2001) **1,857** + **729** (2003) + **572** (2005) + **685** (2006) + ...

NSGA-II (2002) **15,772** + **3,203** (2000) + ...

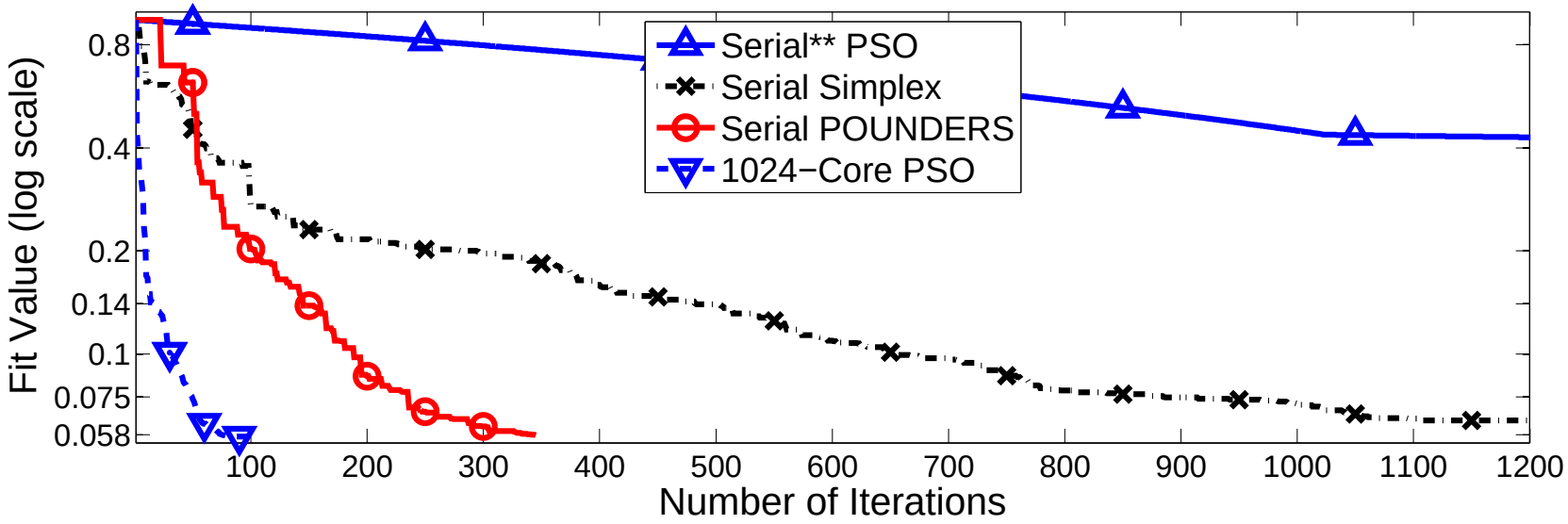
Others Particle swarm, ant colony, firefly, ...

The extensive use of heuristics was a key driver for Powell's later work

Above software: `https://ccpforge.cse.rl.ac.uk/gf/project/powell/`

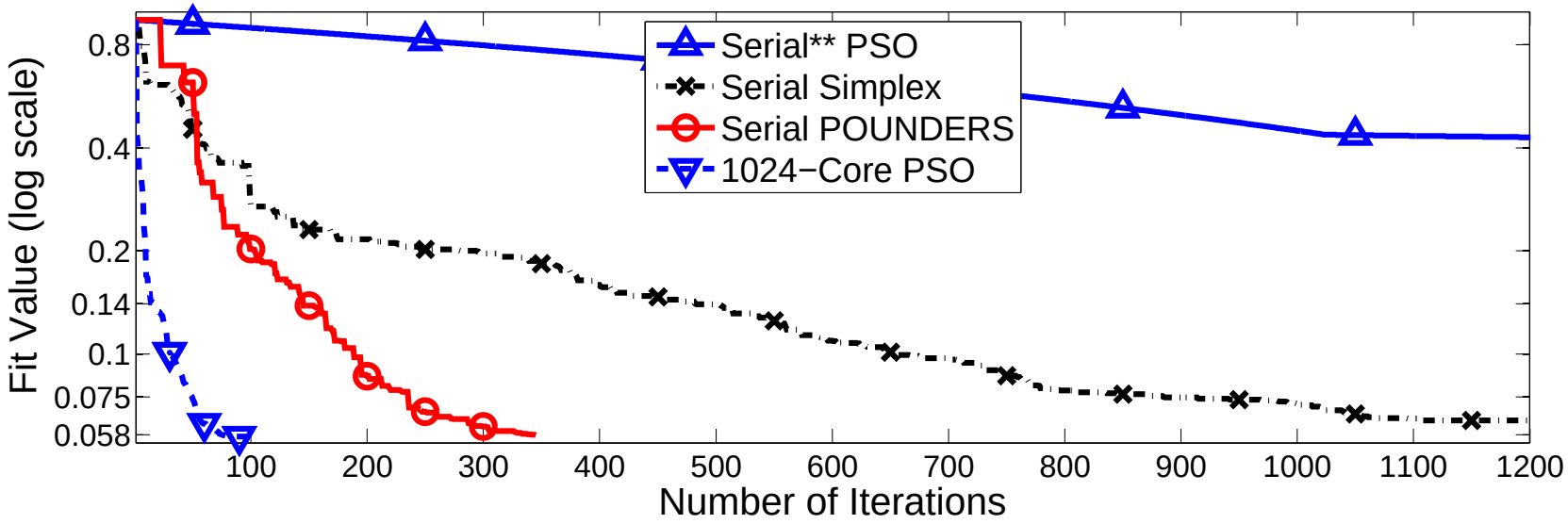
One Reason For Increased Adoption: Concurrent Function Evaluations

Perform p evaluations $f(x^1), \ldots, f(x^p)$ concurrently



One Reason For Increased Adoption: Concurrent Function Evaluations

Perform p evaluations $f(x^1), \ldots, f(x^p)$ concurrently



Poor sequential methods can become attractive as parallelism increases

1. Wall time: Time required to obtain solution
2. Scalability: Efficiency of use of parallel resources

Outline: APOSMM Aims for Increased Concurrency

1. Reframe problem: Multiple local minimizers
2. Multistart: Exploit efficient local solvers
3. Guided by asymptotic convergence
4. Asynchronicity: Beyond batch evaluations
5. Performance metrics
6. Early numerical results

Outline: APOSMM Aims for Increased Concurrency

1. Reframe problem: Multiple local minimizers
2. Multistart: Exploit efficient local solvers
3. Guided by asymptotic convergence
4. Asynchronicity: Beyond batch evaluations
5. Performance metrics
6. Early numerical results

Today: Stay within a Powell-like setting

$$\min_x \{f(x; B(x)) : \ x \in \mathcal{D} \subset \mathbb{R}^n\}$$

- Objective f depends on the output(s) of a computationally expensive blackbox
  - Derivatives unavailable, n small (certainly less than 100)
- Bound constraints $\mathcal{D} = [l, u]$ (compact, independent of blackbox

Henk van der Vorst, 2011

Why Multistart?

Best minimizer(s) approximate global minimizer x^*, $f(x^*) \leq f(x) \, \forall x \in \mathcal{D}$

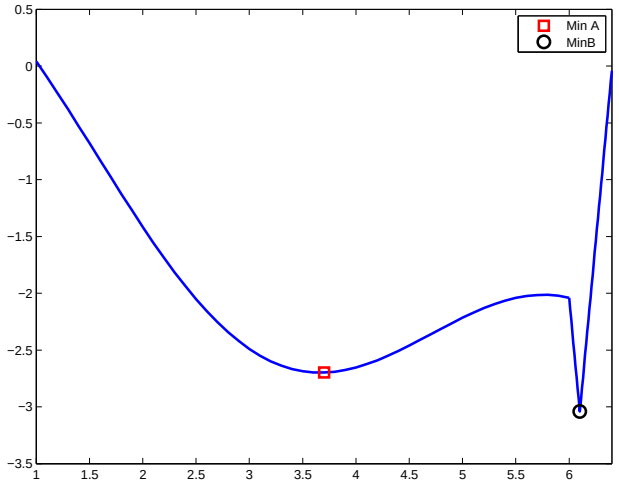
Multiple local minima are often of interest in practice

Design Multiple objectives/constraints might later be of interest

Distinctness j best minimizers have physical meaning

Simulation Errors Spurious local minima from simulator anomalies

Uncertainty Some minima more sensitive to perturbations



Increased opportunity for parallelism

Trilevel simulation/function → local solver → global solver

Efficient local solvers

- (Local) surrogate-based, exploit problem structure
  - least-squares objectives, (un)relaxable constraints, known nonsmoothness, ...

Convergent Methods for Global Optimization, $\min_{x \in \mathcal{D}} f(x)$

either assume more about your problem (e.g., convex f, finite $|\mathcal{D}|$)

or expect to wait forever

Törn and Žilinskas: An algorithm converges to the global minimum for any continuous f if and only if the sequence of points visited by the algorithm is dense in $\mathcal{D}$.

Convergent Methods for Global Optimization, $\min_{x \in \mathcal{D}} f(x)$

either assume more about your problem (e.g., convex f, finite $|\mathcal{D}|$)

or expect to wait forever

Törn and Žilinskas: An algorithm converges to the global minimum for any continuous f if and only if the sequence of points visited by the algorithm is dense in $\mathcal{D}$.

Two-phase iterative methods

1. Global Exploration Sample points from $\mathcal{D}$ $\leftarrow$ Guarantees convergence
2. Local Refinement Ex.- Start a local minimization algorithm $\mathcal{A}$ from some promising subset of (the sample) points
   - ◇ Can require many, sequential evaluations

Convergent Methods for Global Optimization, $\min_{x \in \mathcal{D}} f(x)$

either assume more about your problem (e.g., convex f, finite $|\mathcal{D}|$)

or expect to wait forever

Törn and Žilinskas: An algorithm converges to the global minimum for any continuous f if and only if the sequence of points visited by the algorithm is dense in $\mathcal{D}$.

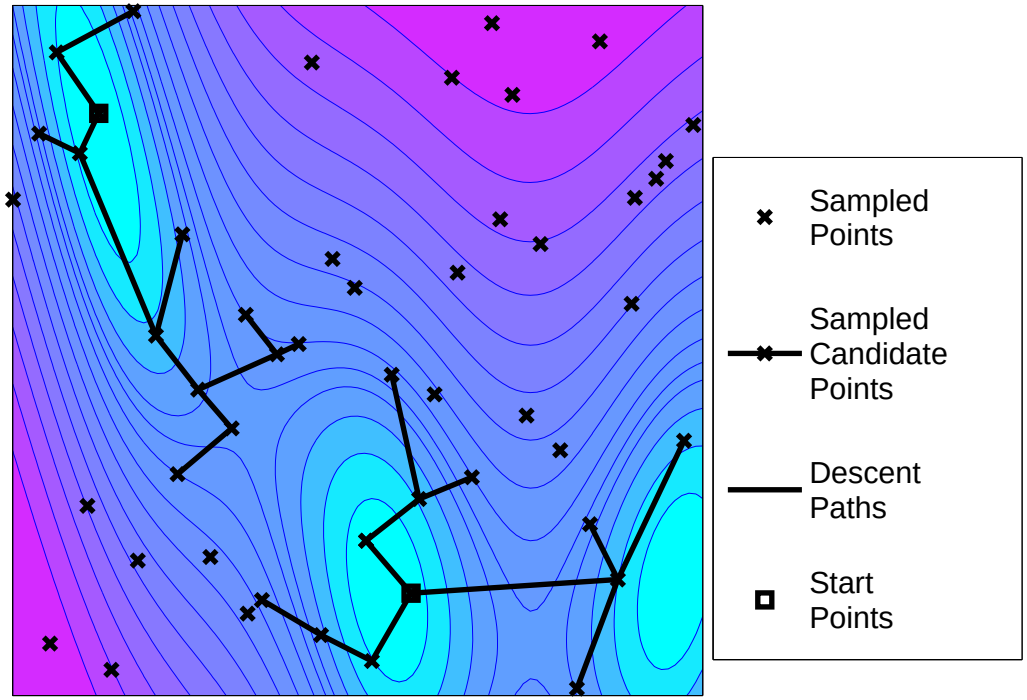
Two-phase iterative methods

1. Global Exploration Sample points from $\mathcal{D}$ $\leftarrow$ Guarantees convergence
2. Local Refinement Ex.- Start a local minimization algorithm $\mathcal{A}$ from some promising subset of (the sample) points
   - ◇ Can require many, sequential evaluations

$\rightarrow$ We want to find many (good) local minima while avoiding repeatedly finding the same local minima . . .
and to do so quickly

Multistart: Multi Level Single Linkage (MLSL) Clustering Procedure

Iteration k [Rinnooy Kan & Timmer (MathProg, 1987)]:



It. 1 Exploration

1. Sample N points from $\mathcal{D}$

- $\mathcal{S}_k = \mathcal{S}_{k-1} \cup \{x^{kN+1-N}, \cdots, x^{kN}\}$ or lower γ quantile of sampled points

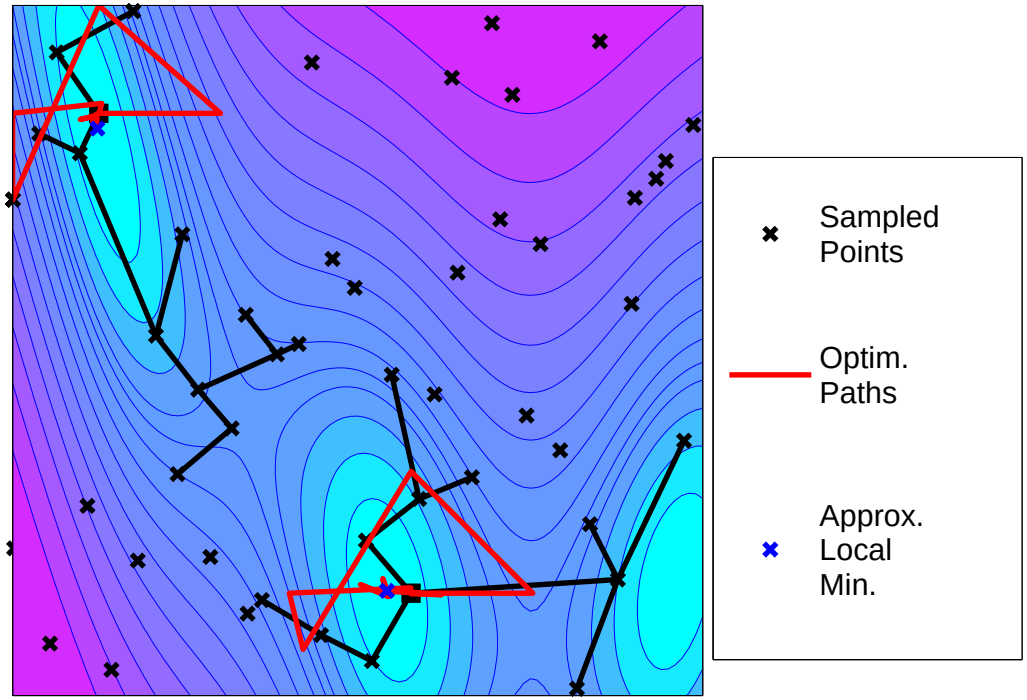
2. Start $\mathcal{A}$ at each sample point $x^i \in \mathcal{S}_k$ provided:

- $\mathcal{A}$ has not been started from x^i, and
- no other sample point $x^j \in \mathcal{S}_k$ with $f(x^j) < f(x^i)$ is within a distance

$$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\text{vol}\,(\mathcal{D})\, \frac{5\Gamma\left(1+\frac{n}{2}\right)\log(kN)}{kN}}$$

Multistart: Multi Level Single Linkage (MLSL) Clustering Procedure

Iteration k [Rinnooy Kan & Timmer (MathProg, 1987)]:



It. 1 Refinement

1. Sample N points from $\mathcal{D}$

- $\mathcal{S}_k = \mathcal{S}_{k-1} \cup \{x^{kN+1-N}, \cdots, x^{kN}\}$ or lower γ quantile of sampled points

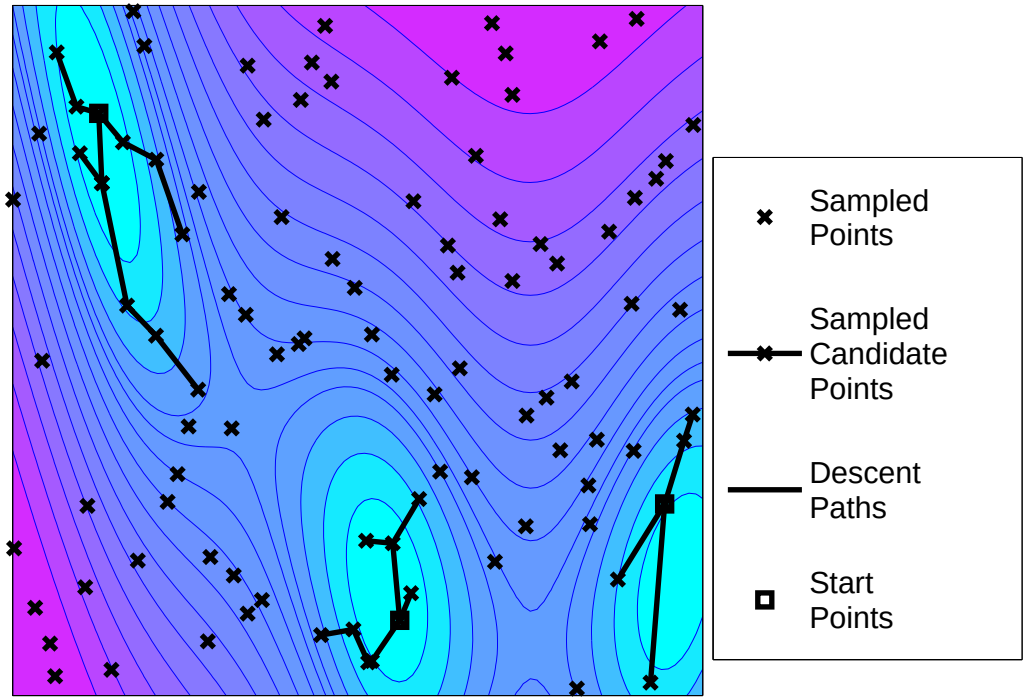
2. Start $\mathcal{A}$ at each sample point $x^i \in \mathcal{S}_k$ provided:

- $\mathcal{A}$ has not been started from x^i, and
- no other sample point $x^j \in \mathcal{S}_k$ with $f(x^j) < f(x^i)$ is within a distance

$$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\mathrm{vol}\,(\mathcal{D})\, \frac{5\Gamma\left(1+\frac{n}{2}\right)\log(kN)}{kN}}$$

Multistart: Multi Level Single Linkage (MLSL) Clustering Procedure

Iteration k [Rinnooy Kan & Timmer (MathProg, 1987)]:



It. 2 Exploration

1. Sample N points from $\mathcal{D}$

- $\mathcal{S}_k = \mathcal{S}_{k-1} \cup \{x^{kN+1-N}, \cdots, x^{kN}\}$ or lower γ quantile of sampled points

2. Start $\mathcal{A}$ at each sample point $x^i \in \mathcal{S}_k$ provided:

- $\mathcal{A}$ has not been started from x^i, and
- no other sample point $x^j \in \mathcal{S}_k$ with $f(x^j) < f(x^i)$ is within a distance

$$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\text{vol}\,(\mathcal{D})\, \frac{5\Gamma\left(1+\frac{n}{2}\right)\log(kN)}{kN}}$$

MLSL Results

Theorem (Rinnooy Kan & Timmer)

Assuming

- Sampling: uniform
- f: Local minimizers uniformly separated *($\Rightarrow$ finite)*
- $\mathcal{A}$: Satisfies (restrictive) descent properties

Then, even if sampling continues forever, with probability 1, MLSL will start finitely many local runs.

MLSL Results

Theorem (Rinnooy Kan & Timmer)

Assuming

Sampling: uniform

f: Local minimizers uniformly separated ($\Rightarrow$ *finite*)

$\mathcal{A}$: Satisfies (restrictive) descent properties

Then, even if sampling continues forever, with probability 1, MLSL will start finitely many local runs.

$\rightarrow$ can refine distance measures based on local curvature knowledge

$\rightarrow$ can update sampling distance for other distributions (e.g., LHS [Larson & W. (2016)])

MLSL Downsides in Practice

Inefficient parallelism

- Batch (size N) sampling
  - time$\left(f(x^{kN+1})\right)$=time$\left(f(x^{kN+i})\right)$, $i = 1, \ldots, N$
- Some number ($< kN - 1$) of $\mathcal{A}$ runs
- Assumes $\mathcal{A}$ "runs to completion" *(oracle)*

Ignores expense of each f evaluation

- Does not consider local optimization points, $\mathcal{L}_k$
- Local algorithm $\mathcal{A}$ run neglects history $\mathcal{H}_k = \mathcal{S}_k \cup \mathcal{L}_k$

Modified Conditions For When to Start a Local Run

MLSL: (S2)–(S4)

$\hat{x} \in \mathcal{S}_k$

(S2) $\nexists x \in \mathcal{S}_k$ with
$[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(S3) $\hat{x}$ has not started a local optimization run

(S4) $\hat{x}$ is at least μ from $\partial\mathcal{D}$ and ν from known local minima

Modified Conditions For When to Start a Local Run

MLSL: (S2)–(S4)
BAMLM [Larson & W. (OptEng, 2015)]: (S1)–(S4), (L1)–(L6)

$\hat{x} \in \mathcal{S}_k$

(S1) $\nexists x \in \mathcal{L}_k$ with $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(S2) $\nexists x \in \mathcal{S}_k$ with $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(S3) $\hat{x}$ has not started a local optimization run

(S4) $\hat{x}$ is at least μ from $\partial\mathcal{D}$ and ν from known local minima

$\hat{x} \in \mathcal{L}_k$

(L1) $\nexists x \in \mathcal{L}_k$ $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(L2) $\nexists x \in \mathcal{S}_k$ with $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(L3) $\hat{x}$ has not started a local optimization run

(L4) $\hat{x}$ is at least μ from $\partial\mathcal{D}$ and ν from known local minima

(L5) $\hat{x}$ is not in an active local optimization run and has not been ruled stationary

(L6) $\exists r_k$-descent path in $\mathcal{H}_k$ from some $x \in \mathcal{S}_k$ satisfying (S2-S4) to $\hat{x}$

Basic Tests on Simulation-Based Problems

Microscopy Configuration of a scanning transmission electron microscope [Rudnaya, Van den Broek, Doornbos, Mattheij, Maubach (Ultramicroscopy, 2011)]

- $n = 3$, at least two local minima (both with values below the average value)

Biometrics Biomechanical control [Easterling, Watson, Madigan, Castle, Trosset (COptA, 2014)]

- $n = 57$, domain scaled to unit cube

Basic Tests on Simulation-Based Problems

Microscopy Configuration of a scanning transmission electron microscope [Rudnaya, Van den Broek, Doornbos, Mattheij, Maubach (Ultramicroscopy, 2011)]

- $n = 3$, at least two local minima (both with values below the average value)

Biometrics Biomechanical control [Easterling, Watson, Madigan, Castle, Trosset (COptA, 2014)]

- $n = 57$, domain scaled to unit cube

Solvers:

GLODS Global & local optimization w/ direct search [Custódio, Madeira (JOGO, 2014)]

Direct Serial Matlab DiRect code [Finkel (2003)]

pVTdirect Parallel DiRect code [He, Watson, Sosonkina (TOMS, 2009)]

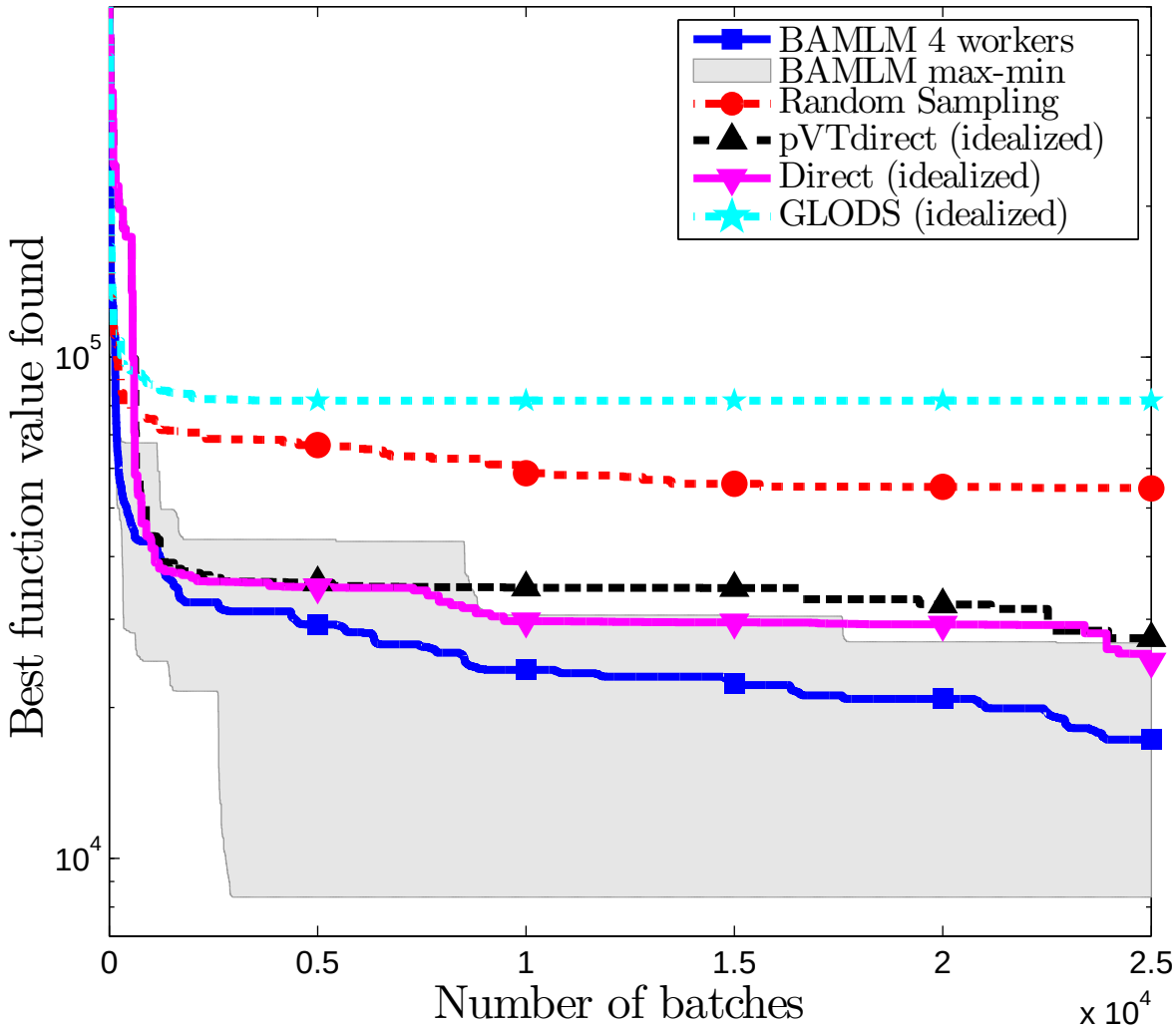
BAMLM [Larson & W. (OptEng, 2015)] with local solver **ORBIT** [W., Regis, Shoemaker (SISC, 2009)]

- Fixed level of concurrency (batch size: 4)
- Idealized performance for non-**BAMLM** solvers

BAMLM Results: Simulation-Based Problems

40 replications

- ◇ *Biometrics* challenging (n=57)

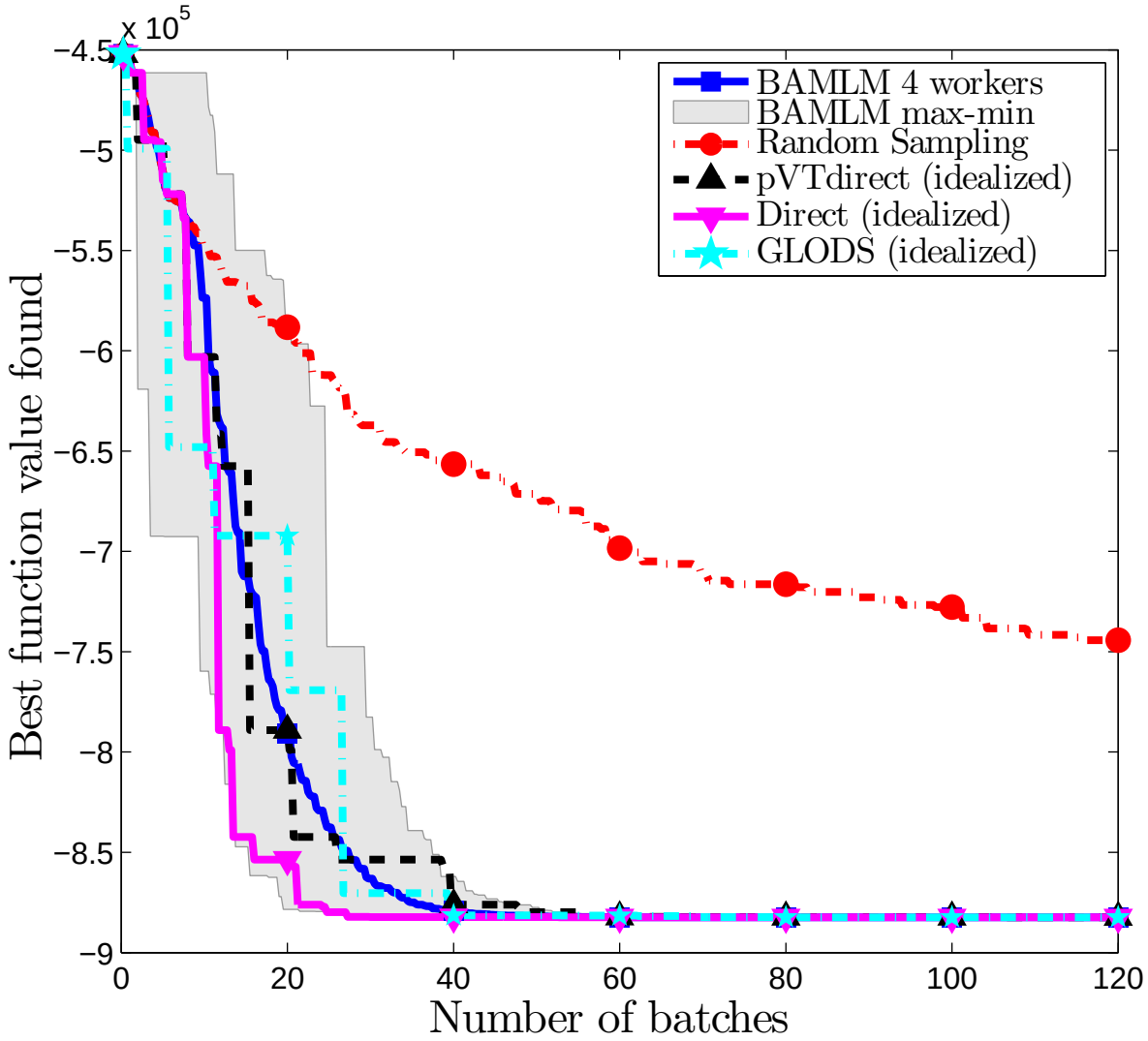


Biometrics, best f value

BAMLM Results: Simulation-Based Problems

40 replications

- *Biometrics* challenging (n=57)
- *Microscopy*: (Idealized) Direct finds global solution

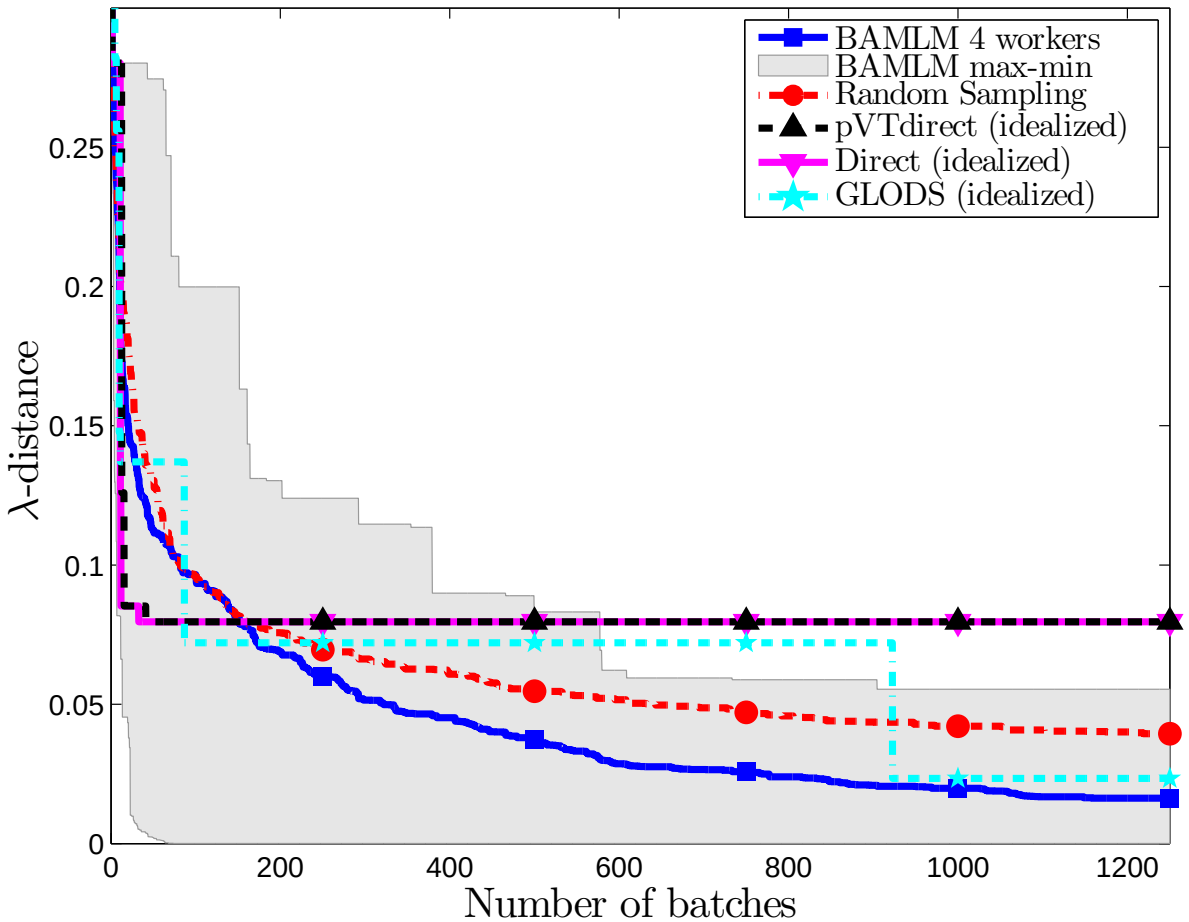


Microscopy, best f value

BAMLM Results: Simulation-Based Problems

40 replications

- *Biometrics* challenging (n=57)
- *Microscopy*: (Idealized) Direct finds global solution
- BAMLM and GLODS find both minimizers

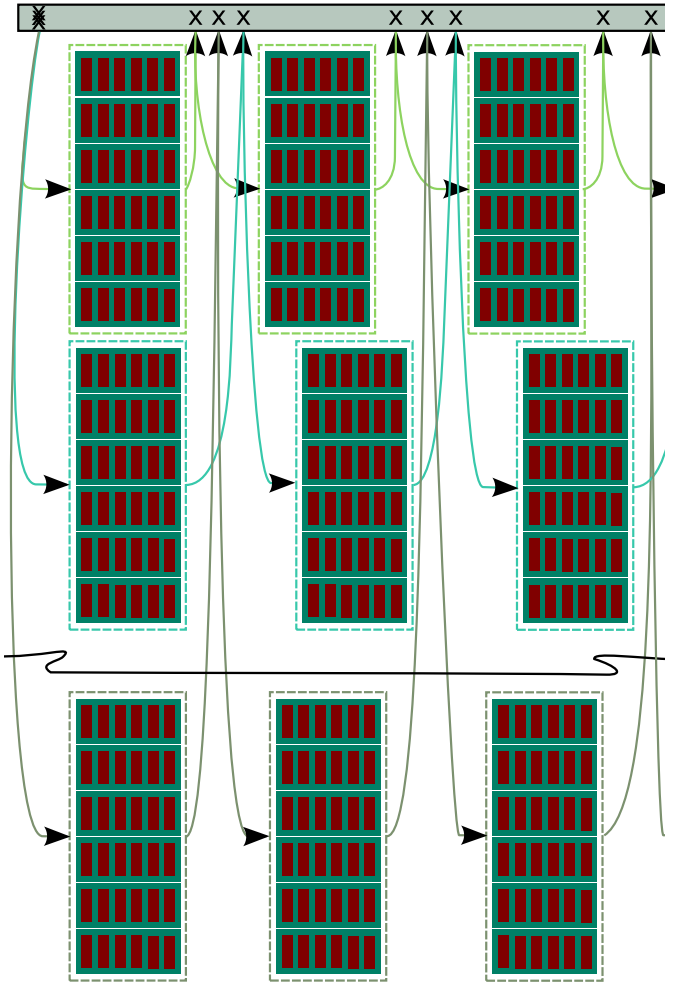


Microscopy, max distance to 2 best minima

Breaking Batch Parallelism

- Better account for dynamic number of local runs
- Decouple local run from fixed resource
- Anticipate nontrivial Var[time $(f(x))$]

Fundamental structural change to alg



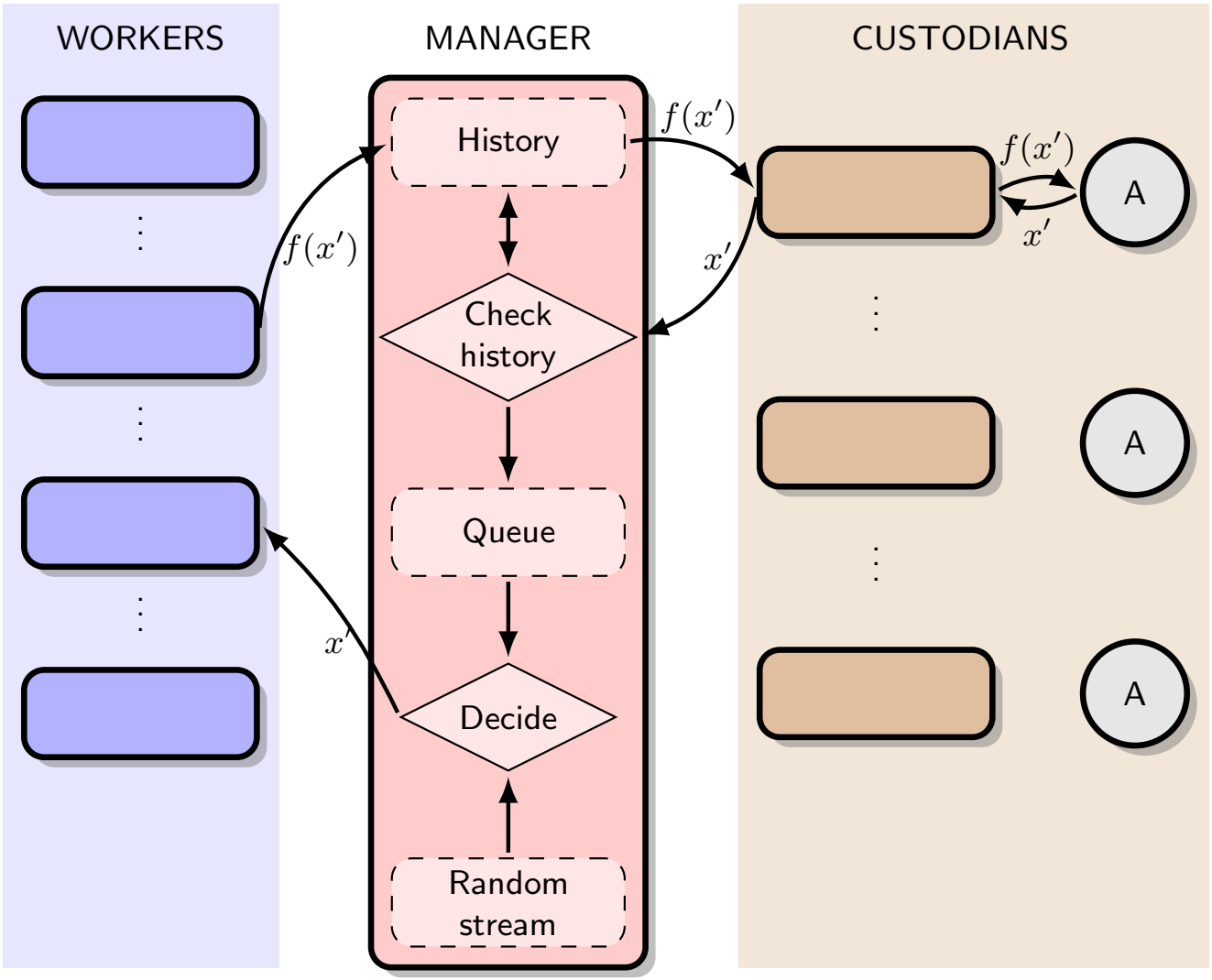
The (A)POSMM Algorithm

Repeat:

- Receive from worker(s) $w_\ell \in W$ that has evaluated its point
- If point was a sample point, update $r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\text{vol}\left(\mathcal{D}\right) \frac{5\Gamma\left(1+\frac{n}{2}\right)\log(|\mathcal{S}_k|)}{|\mathcal{S}_k|}}$
- If point was a local optimization point, add subsequent point in the run (not in $\mathcal{H}_k$) to Q_L if not terminated
- Start run(s) at all point(s) now satisfying conditions, adding subsequent point from each run to Q_L
- Merge/collapse runs within Q_L
- Send point(s) from Q_L and/or $\mathcal{R}$ to worker(s)

W	Set of workers	(level of concurrency $\lvert W \rvert$)
$\mathcal{R}$	Stream of sample points (from $\mathcal{D}$)	
$\mathcal{S}_k$	Sample points after iteration k	
Q_L	Queue of local optimization points (needed by $\mathcal{A}$)	
$\mathcal{H}_k$	History after k evaluations	

(A)POSMM Framework



Two Different Marsupials

POSMM

- ◇ Clears all workers before making decisions for next $|W|$ evaluations
- – Blocking operation, wasteful as Var[time $(f(x))$] grows
- \+ Reproducible runs

Two Different Marsupials

POSMM

- ◇ Clears all workers before making decisions for next $|W|$ evaluations
- − Blocking operation, wasteful as Var[time $(f(x))$] grows
- \+ Reproducible runs

APOSMM

- ◇ Processes single worker $w \in W$ when it frees up
- \+ Fully asynchronous
- − Irreproducible runs (nondeterministic run times)

Lemma

BAMLM ~~(A)POSMM~~ starts no more optimization runs than does MLSL.

Asymptotic Foundations

Lemma

BAMLM ~~(A)POSMM~~ starts no more optimization runs than does MLSL.

Theorem

- Given assumptions on f, $\mathcal{A}$, and sampling, if `(A)POSMM` is run forever, there will almost surely be a finite number of local optimization runs that have a (non-starting) point evaluated.
- Furthermore, if `(A)POSMM` is run forever and the next point given to a worker satisfies Assumption, then with probability 1 each $x^* \in X^*$ will be
  - identified in a finite number of evaluations, or
  - have a single local optimization run that is converging asymptotically to it.

Asymptotic Foundations

Lemma

BAMLM ~~(A)POSMM~~ starts no more optimization runs than does MLSL.

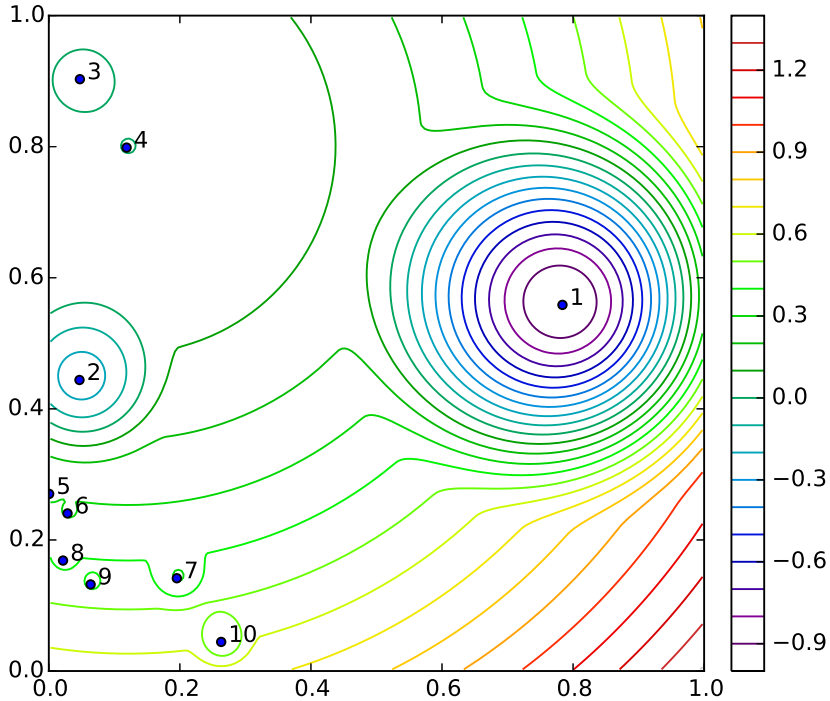
Theorem

- Given assumptions on f, $\mathcal{A}$, and sampling, if `(A)POSMM` is run forever, there will almost surely be a finite number of local optimization runs that have a (non-starting) point evaluated.
- Furthermore, if `(A)POSMM` is run forever and the next point given to a worker satisfies Assumption, then with probability 1 each $x^* \in X^*$ will be
  - identified in a finite number of evaluations, or
  - have a single local optimization run that is converging asymptotically to it.

Assumption:
There exists $K_0 < \infty$ such that for any K_0 consecutive iterations, the probabilities of taking a single point from $\mathcal{R}$ and a point from each of the local optimization runs is bounded away from zero.

Numerical Results: 600 GKLS Functions

- From GKLS problem generator [Gaviano, Kvasov, Lera, Sergeyev (TOMS, 2003)]
- Smooth modifications of a convex quadratic
- $n = 2, \ldots, 7$
- 10 known local minima in the unit cube, unique global min
- 100 problems for each dimension



Two Measures of Success

Time/number of (batches of) evaluations k until success:

j best local minima at a level $\tau \geq 0$

Success if

$$\exists x^i \in \mathcal{H}_k \text{ with } \left\| x^i - x^*_{(i)} \right\| \leq d_n(\tau)$$

for at least j (specially chosen) $x^*_{(i)} \in X^*(j)$

- $X^*(j) = \left\{ x^*_{(1)}, \ldots, x^*_{(j)} \right\}$ = set of minimizers with function values corresponding to the j best local minima
- $d_n(\tau) = \sqrt[n]{\frac{\tau \, \mathrm{vol}(\mathcal{D}) \, \Gamma(\frac{n}{2}+1)}{\pi^{n/2}}}$
- Requires knowledge of $X^*(j)$

Two Measures of Success

Time/number of (batches of) evaluations k until success:

j best local minima at a level $\tau \geq 0$

Success if

$$\exists x^i \in \mathcal{H}_k \text{ with } \left\| x^i - x^*_{(i)} \right\| \leq d_n(\tau)$$

for at least j (specially chosen) $x^*_{(i)} \in X^*(j)$

- $X^*(j) = \left\{ x^*_{(1)}, \ldots, x^*_{(j)} \right\}$ = set of minimizers with function values corresponding to the j best local minima
- $d_n(\tau) = \sqrt[n]{\frac{\tau \, \mathrm{vol}(\mathcal{D}) \, \Gamma(\frac{n}{2}+1)}{\pi^{n/2}}}$
- Requires knowledge of $X^*(j)$

Global function value at a level $\tau \geq 0$

Success if have found $x \in \mathcal{H}_k$ satisfying

$$f(x) - f_G \leq (1-\tau)\left(f(x^0) - f_G\right),$$

- x^0: common starting point
- f_G: (estimate of) value at the global minimum

Numerical Tests

`GLODS`	Global & local optimization using direct search [Custódio, Madeira (JOGO, 2014)]
`Direct`	Serial Matlab DiRect code [Finkel (2003)]
`pVTdirect`	Parallel DiRect code [He, Watson, Sosonkina (TOMS, 2009)]
`CMA-ES`	Parallel Covariance Matrix Adaptation Evolution Strategy code [Hansen & Ostermeier (EvolComp, 2001)]
`Random`	Uniform sampling (as a baseline)
`(A)POSMM`	[Larson, W. (2016)] with local solver `BOBYQA` [Powell (2009)]

Numerical Tests

`GLODS`	Global & local optimization using direct search [Custódio, Madeira (JOGO, 2014)]
`Direct`	Serial Matlab DiRect code [Finkel (2003)]
`pVTdirect`	Parallel DiRect code [He, Watson, Sosonkina (TOMS, 2009)]
`CMA-ES`	Parallel Covariance Matrix Adaptation Evolution Strategy code [Hansen & Ostermeier (EvolComp, 2001)]
`Random`	Uniform sampling (as a baseline)
`(A)POSMM`	[Larson, W. (2016)] with local solver `BOBYQA` [Powell (2009)]

Use data profiles [Moré, W. (SIOPT, 2009)] to aggregate results

$$d_s(\alpha) = \frac{\left|\left\{p \in \mathcal{P} : \frac{t_{p,s}}{n_p+1} \leq \alpha\right\}\right|}{|\mathcal{P}|},$$

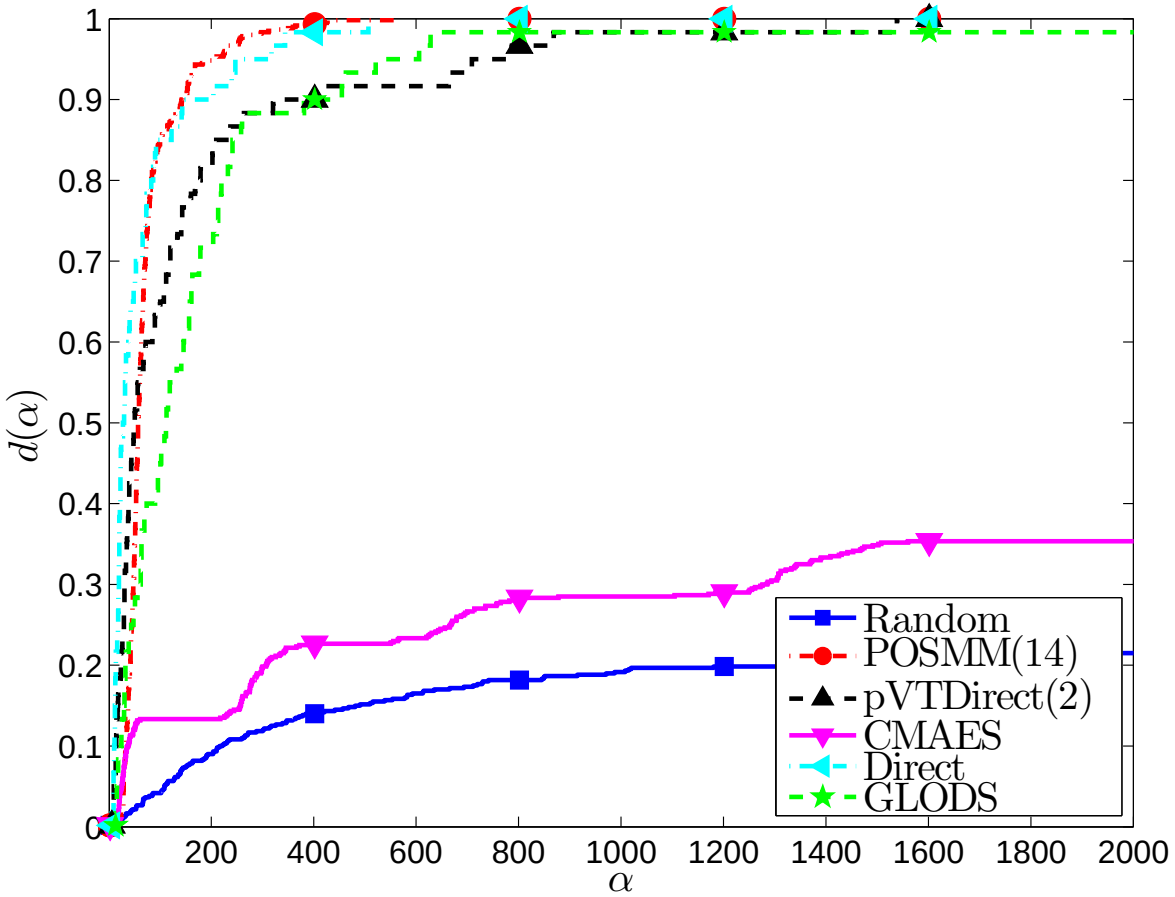
- cdf of successes within α (simplex-equivalent) evals
- $t_{p,s}$: number of evals required to satisfy performance metric

Data Profiles: Ability to Find Approximate Global Minimizer

600 GKLS problems

(A) POSMM

- Makes rapid progress to f_G
- Outperforms other algorithms (even while demanding 14-fold concurrency)



$$\tau = 10^{-2}$$

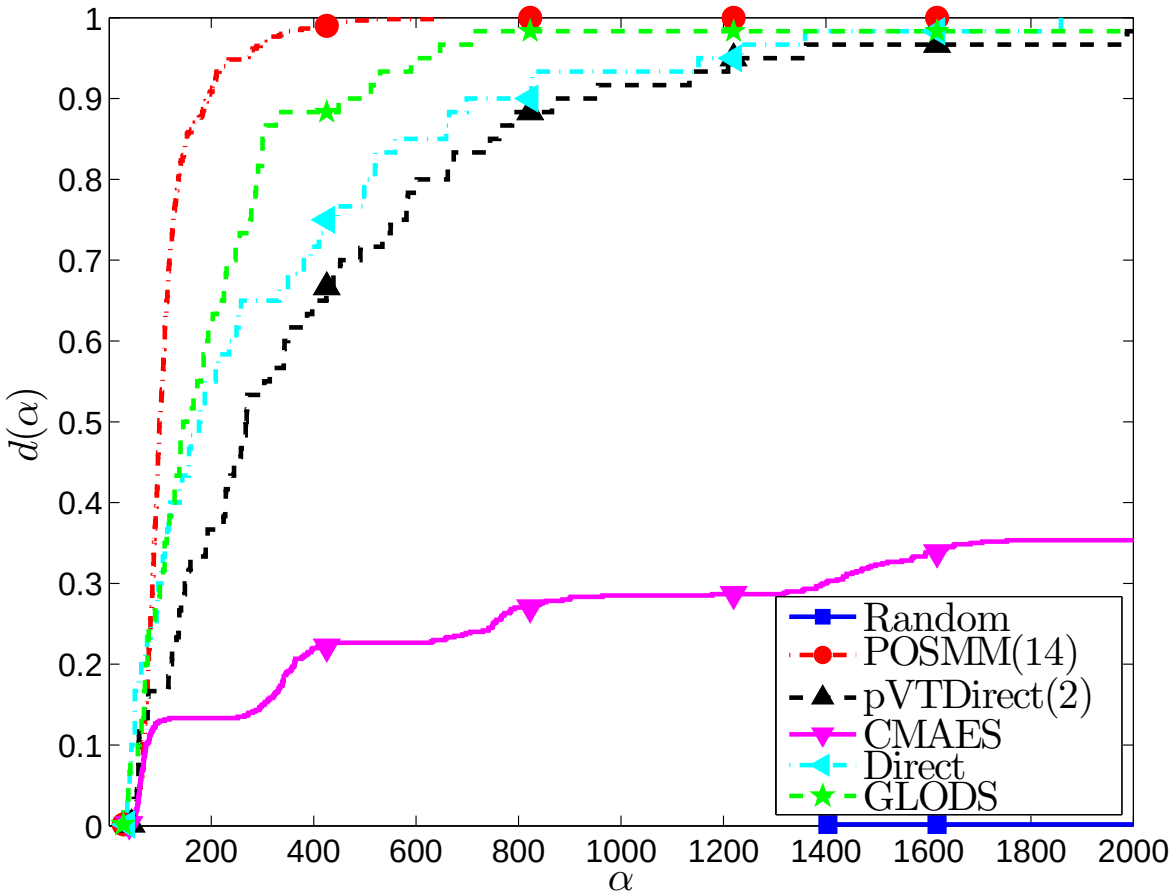
$$f(x) - f_G \leq (1-\tau)\left(f(x^0) - f_G\right)$$

Data Profiles: Ability to Find Approximate Global Minimizer

600 GKLS problems

(A) POSMM

- Makes rapid progress to f_G
- Outperforms other algorithms (even while demanding 14-fold concurrency)



$$\tau = 10^{-5}$$

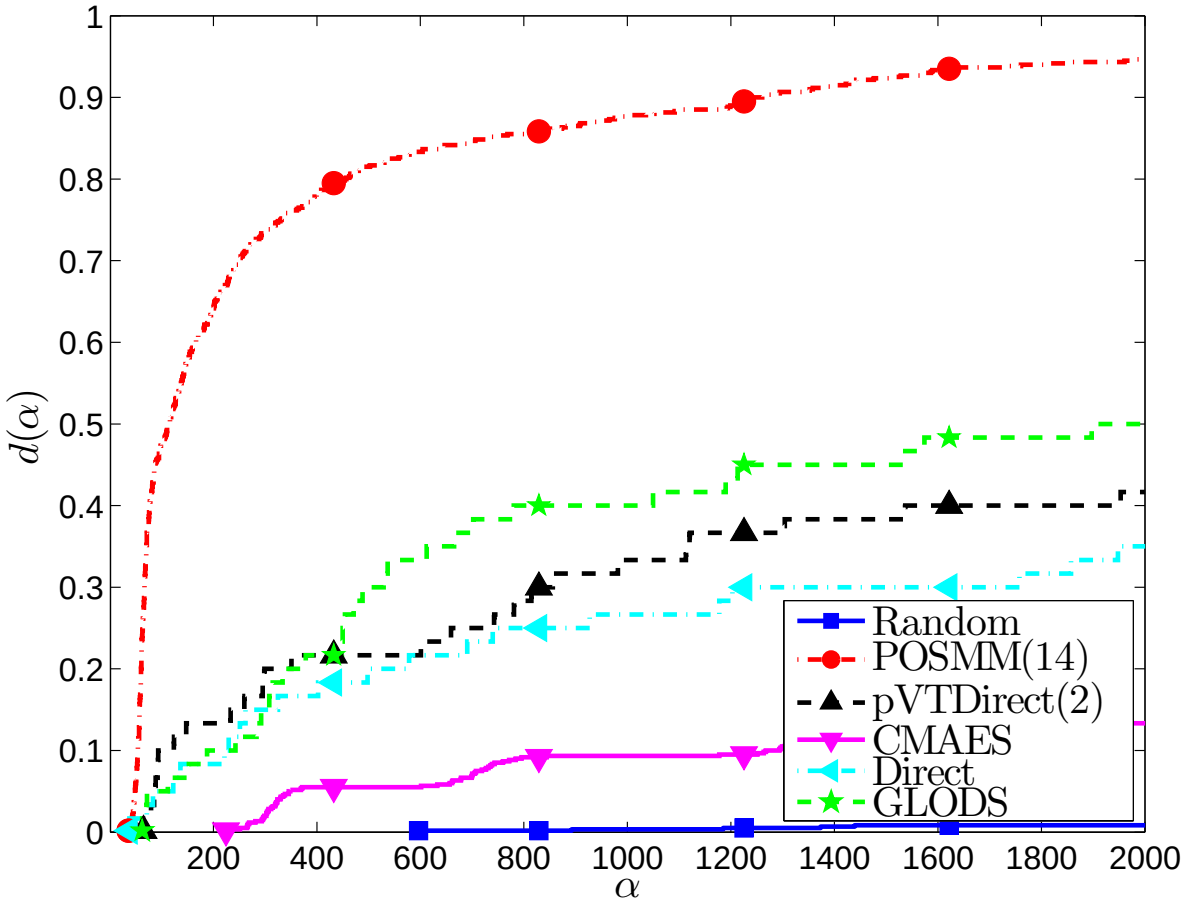
$$f(x) - f_G \leq (1-\tau)\left(f(x^0) - f_G\right)$$

Data Profiles: Ability to Find j Best Minimizers

600 GKLS problems

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances



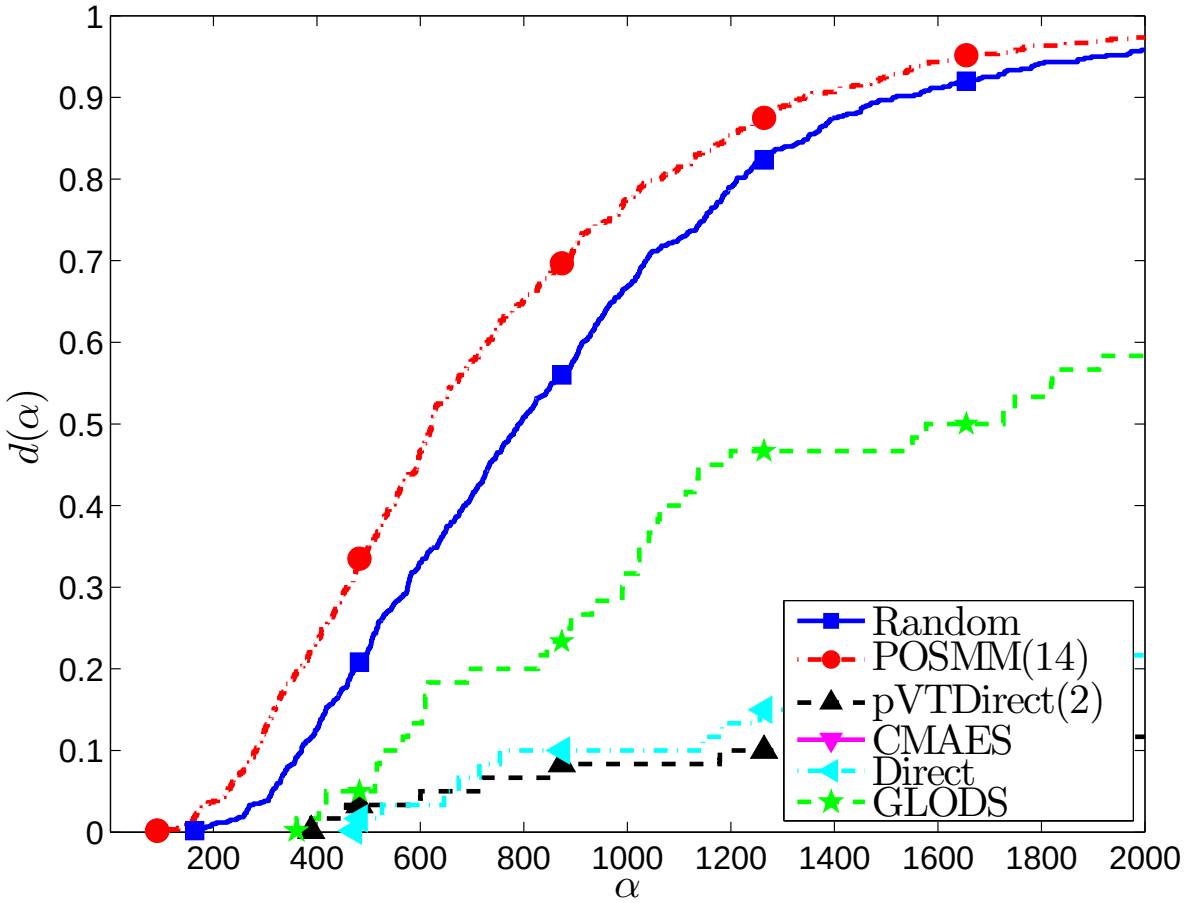
distance $\tau = 10^{-5}$, $j = 2$ minimizers

Data Profiles: Ability to Find j Best Minimizers

600 GKLS problems

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances



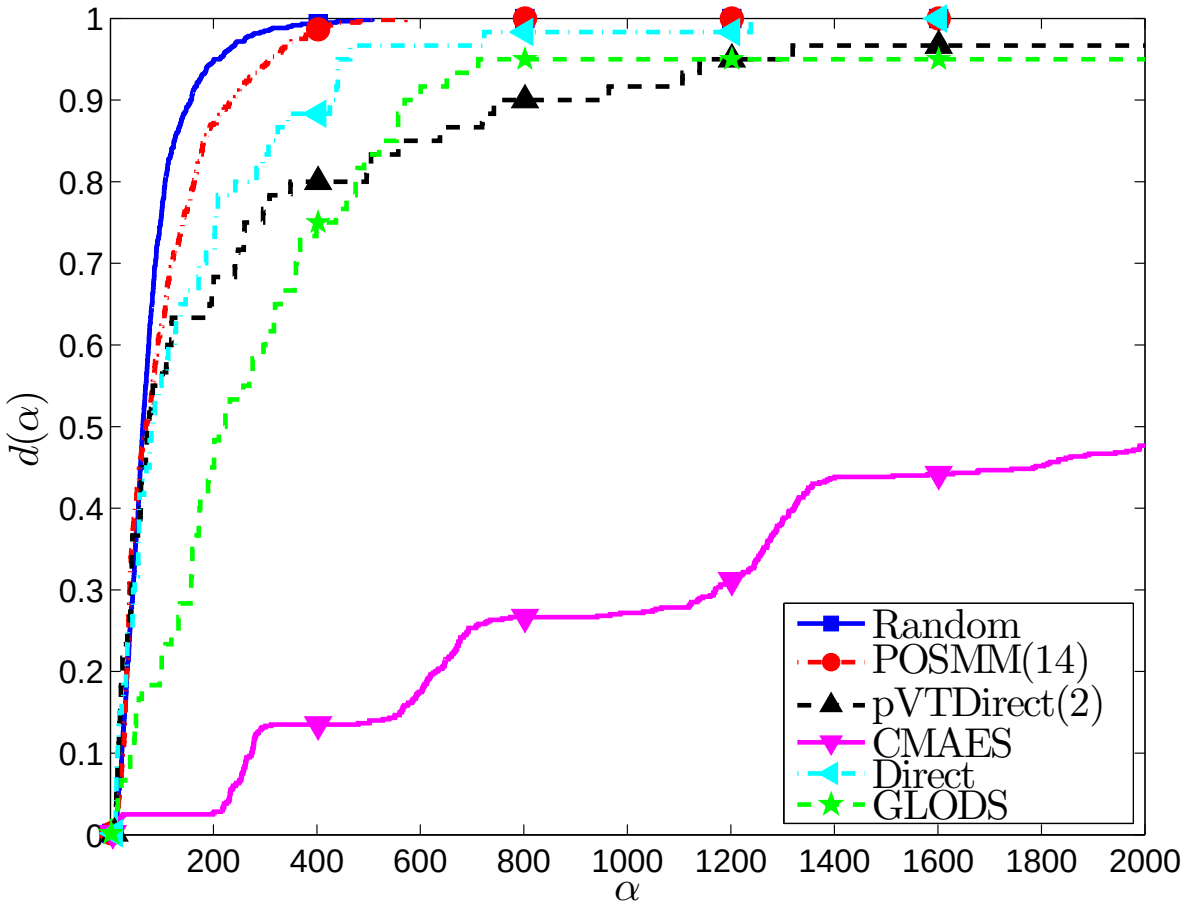
distance $\tau = 10^{-3}$, $j = 7$ minimizers

Data Profiles: Ability to Find j Best Minimizers

600 GKLS problems

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances



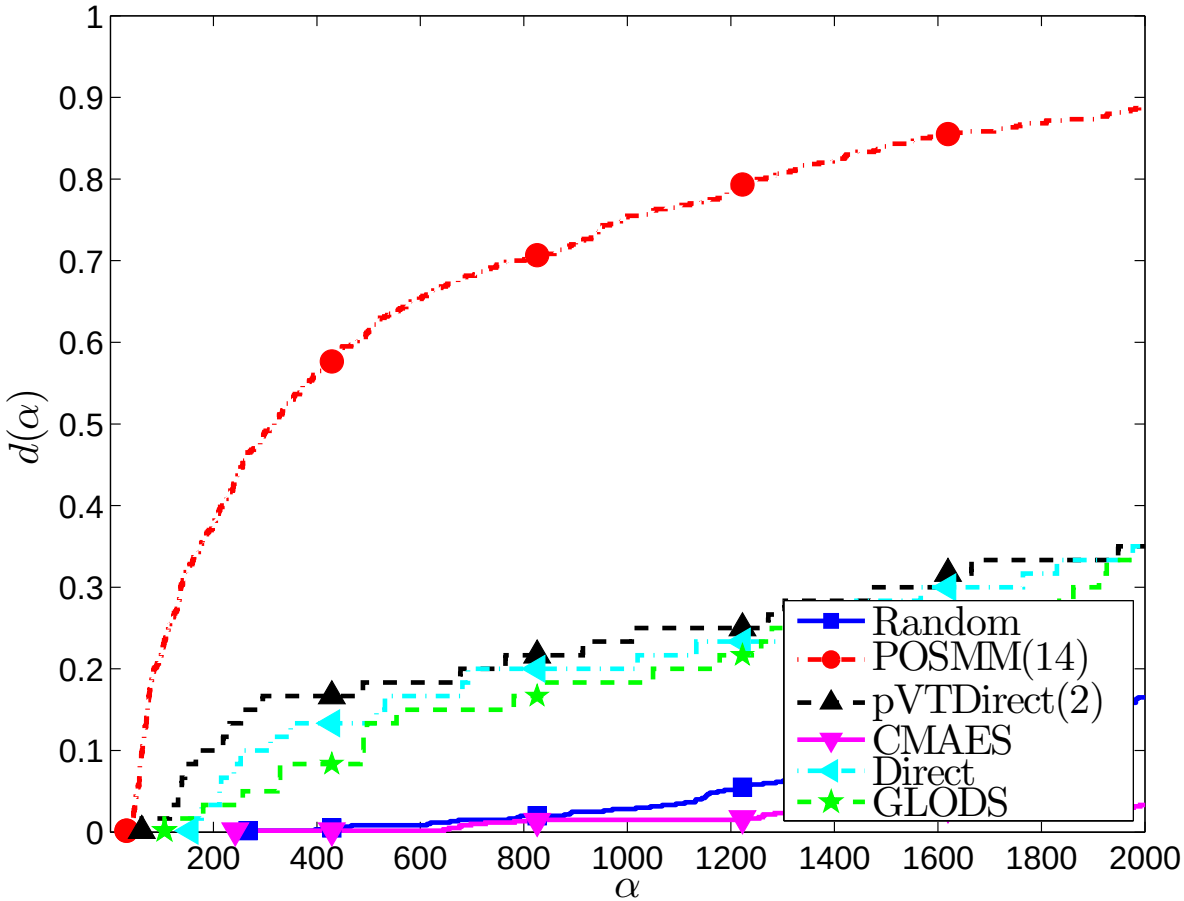
distance $\tau = 10^{-2}$, $j = 3$ minimizers

Data Profiles: Ability to Find j Best Minimizers

600 GKLS problems

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances



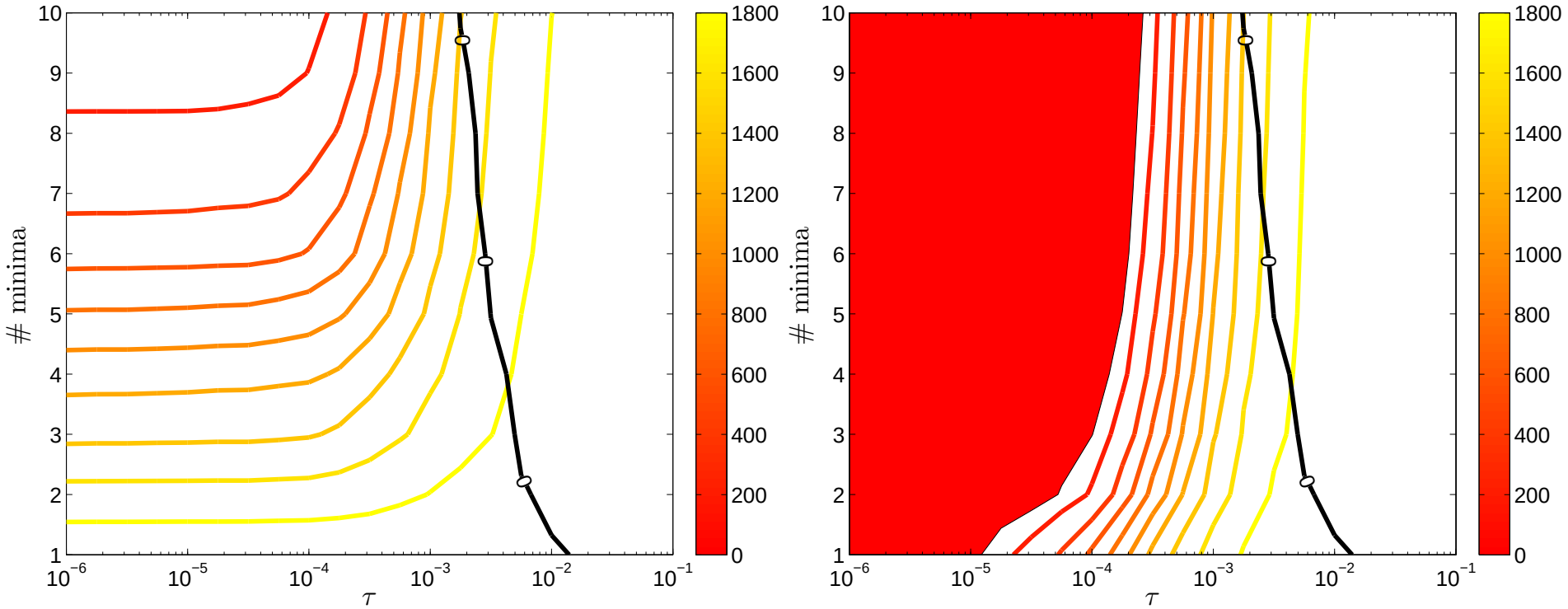
distance $\tau = 10^{-4}$, $j = 3$ minimizers

Sanity Check

- Is this just "advanced" random sampling?
- Is random sampling ever better?

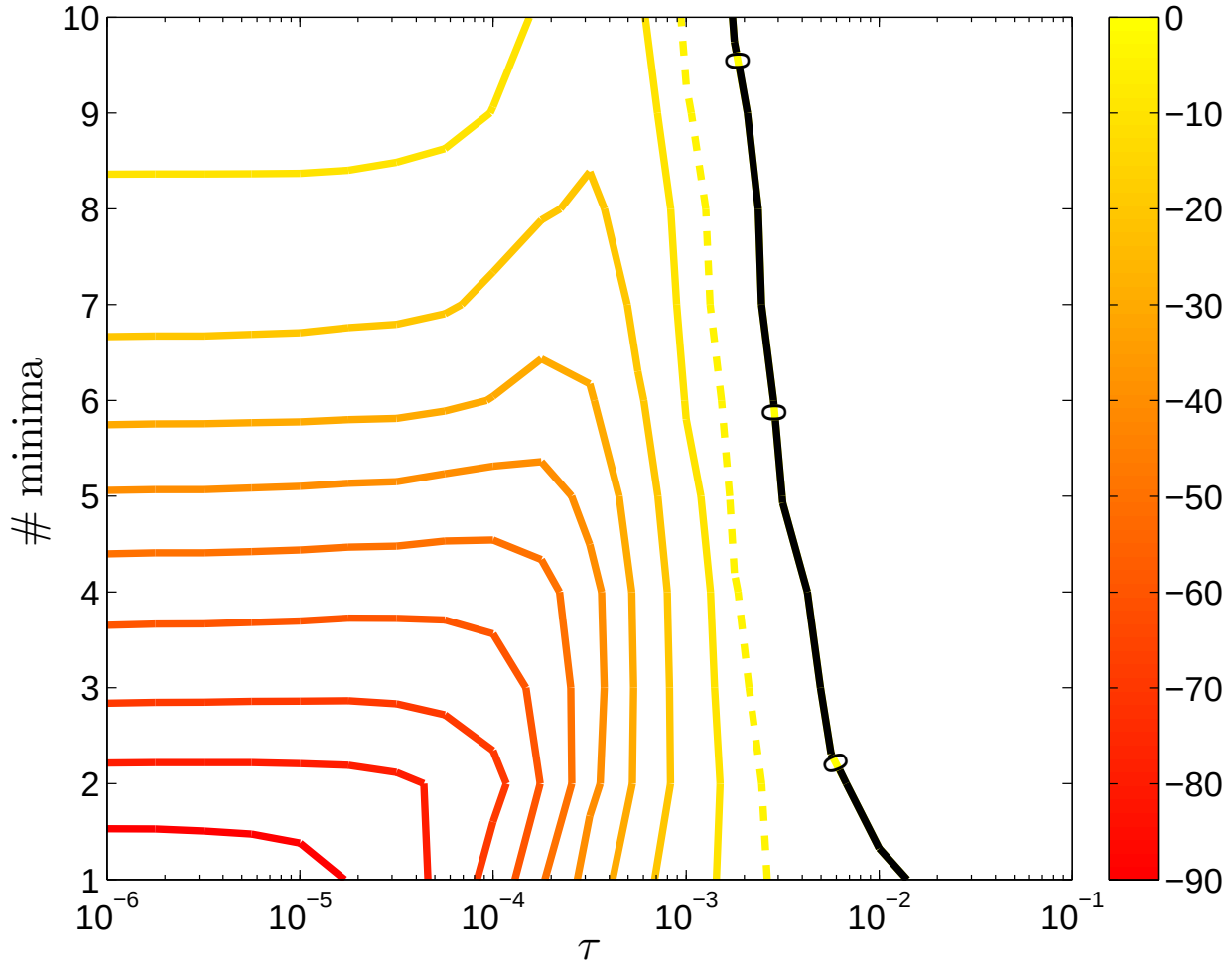
(Tolerance, # Minimizers) Comparison With Random Sampling

Area under data profile: POSMM (L), RS (R)



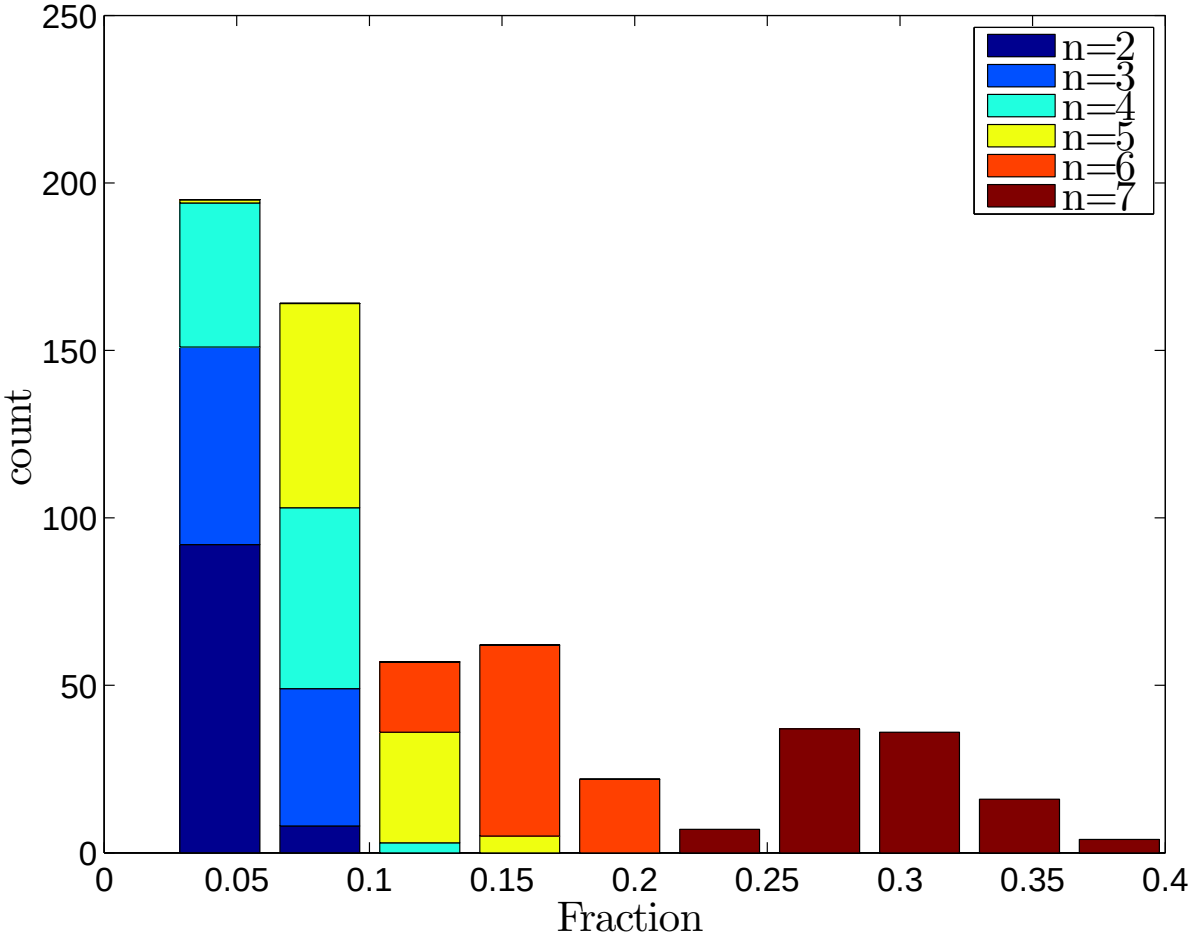
(Tolerance, # Minimizers) Comparison With Random Sampling

Percent difference between POSMM and RS areas



POSMM has clear advantages as tolerances tighten

Fraction of Evaluations Dedicated to Optimization

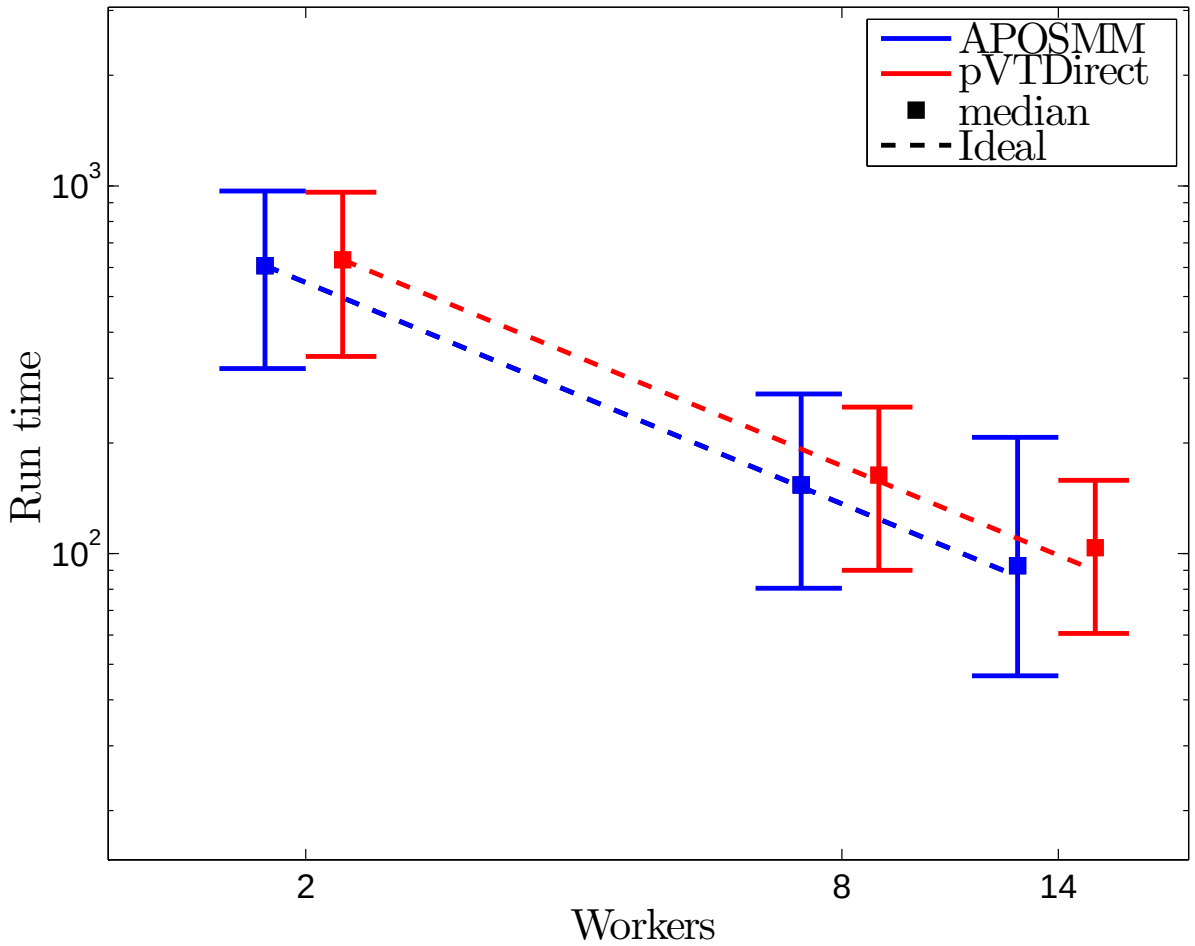


14 workers (10 local minimizers for each n)

Increasing fraction of work assigned to local optimization

Scaling Results For Evaluations: Variable Time($f(x)$)

Mean wall time to exhaust $2000(n+1)$ evals

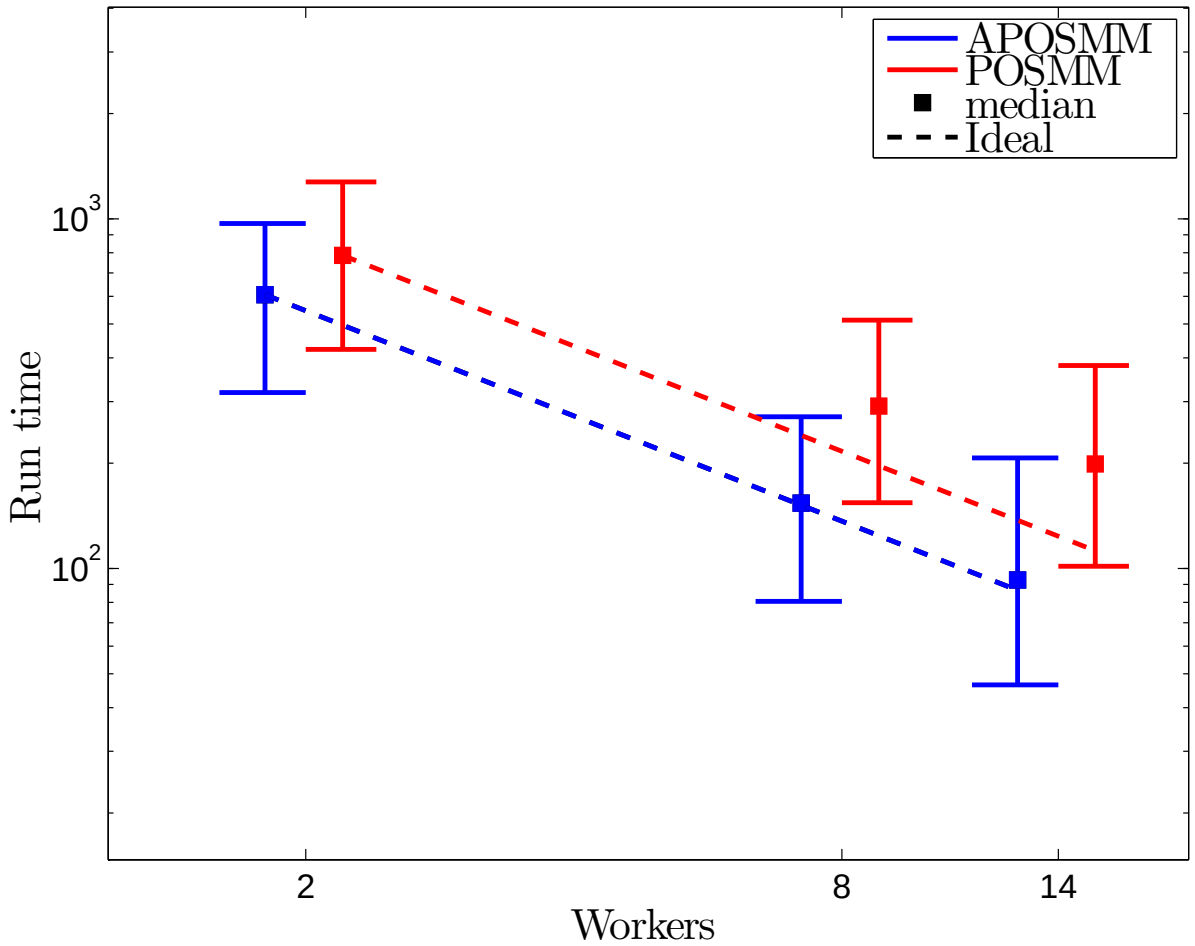


600 GKLS problems, time($f(x)$) ~Unif[0, 0.2] seconds

Asynchronicity of `APOSMM` & `pVTDirect` yields near-perfect execution scaling

Scaling Results For Evaluations: Variable Time($f(x)$)

Mean wall time to exhaust $2000(n+1)$ evals

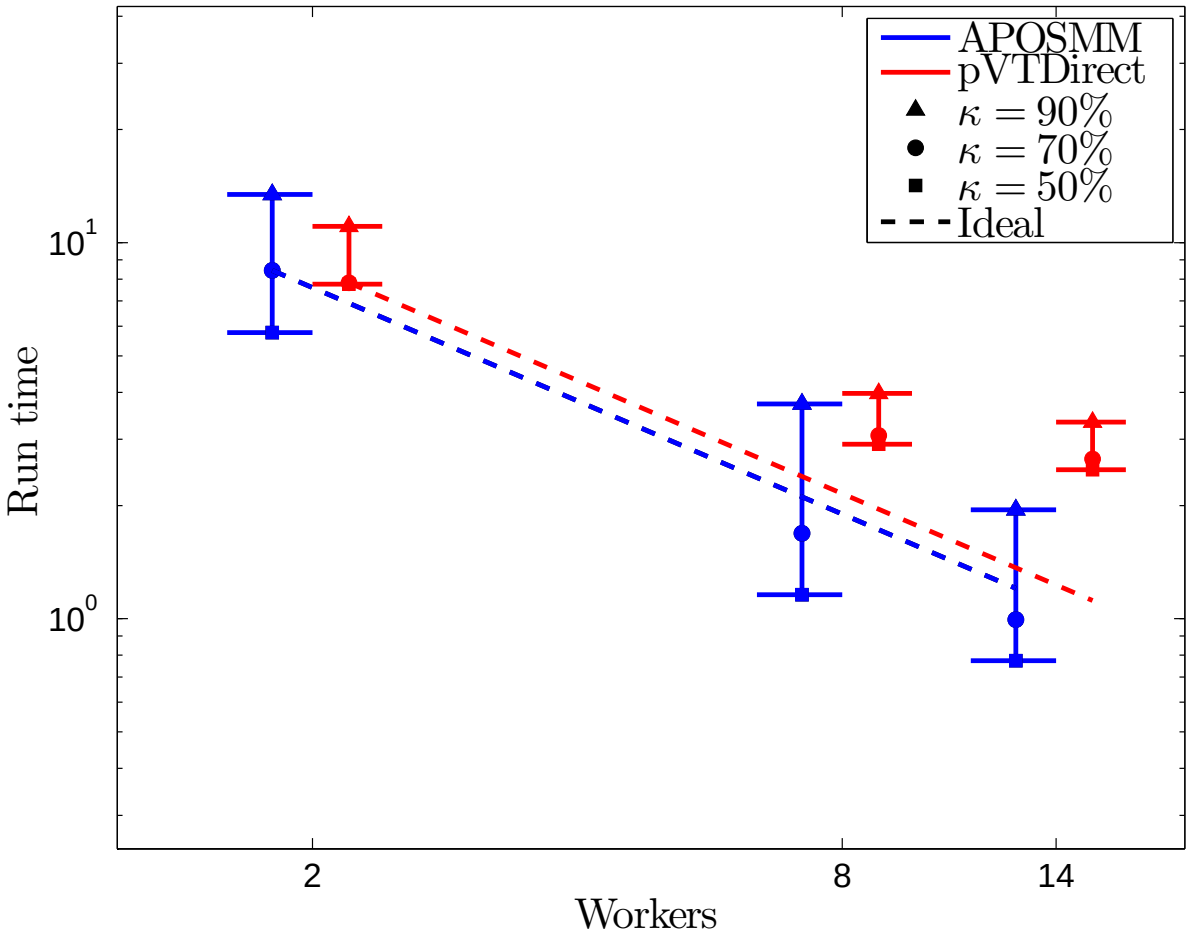


600 GKLS problems, time($f(x)$) ~Unif[0, 0.2] seconds

Asynchronicity of `APOSMM` & `pVTDirect` yields near-perfect execution scaling

Scaling Results For Time To Solution

Mean wall time to find $j = 3$ best minimizers ($\tau = 10^{-2}$ accuracy) on κ probs

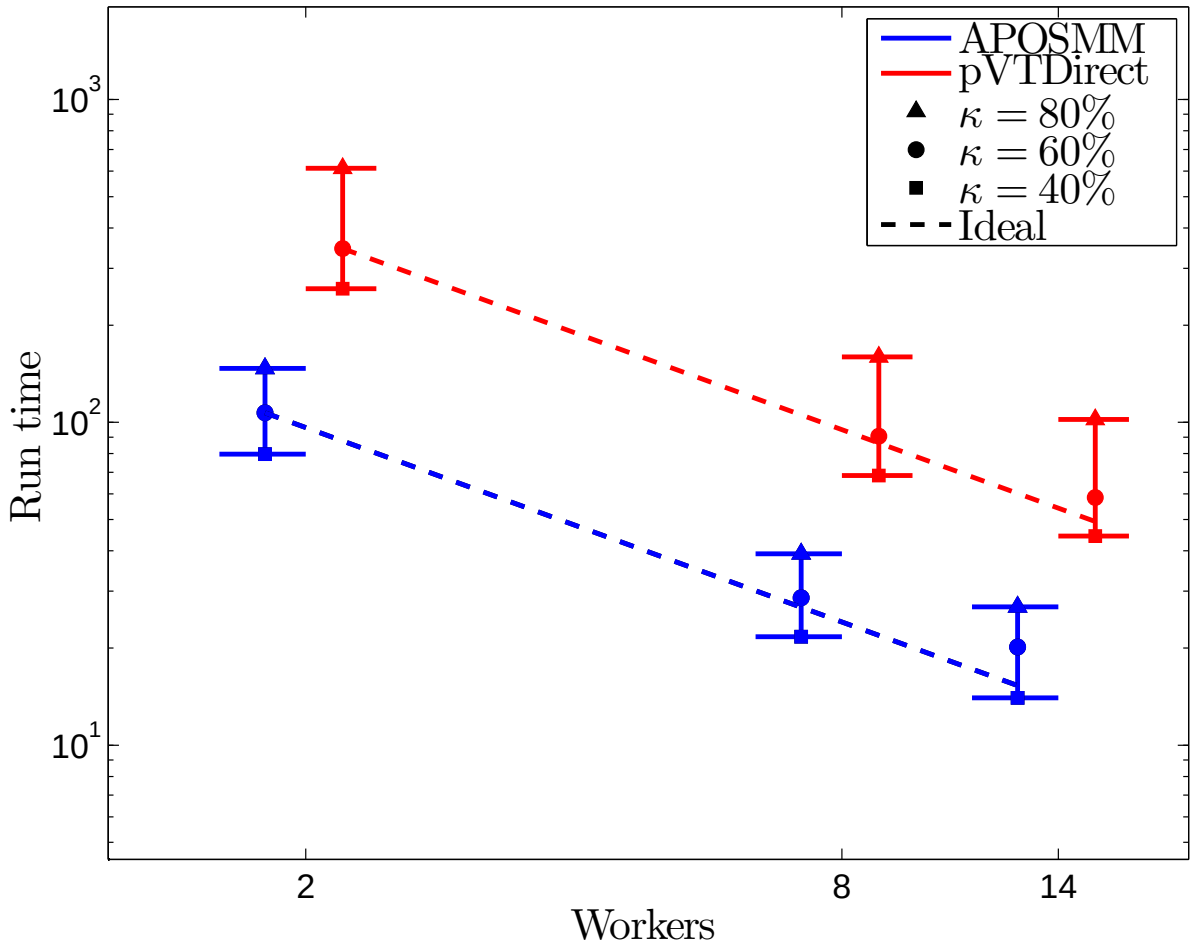


$n = 2$

(A)POSMM time to solution scales well up to (at least) 14-fold concurrency

Scaling Results For Time To Solution

Mean wall time to find $j = 3$ best minimizers ($\tau = 10^{-2}$ accuracy) on κ probs



$n = 7$

(A)POSMM time to solution scales well up to (at least) 14-fold concurrency

Summary & Early Conclusions

- APOSMM+BOBYQA effective at finding multiple minimizers
  ... without sacrificing quality of approximate global soln
- Asymptotic convergence results limit the number of expensive local runs
- POSMM admits additional concurrency, reduces time to solution
- Scales to concurrency above problem dimension, number of minimizers, number of desired minimizers, ...
- Asynchronous logic allows for heterogeneous function evaluation times
- Preprint + python software [Larson, W. (2016)]
  [Larson & W. "A Batch, Derivative-free Algorithm for Finding Multiple Local Minima." (OptEng, 2015)]

Current work:

- Tests for greater concurrency levels
- Interaction with local solver (use of information, concurrent evals, termination, addressing (stochastic) noise, ...)

Summary & Early Conclusions

- APOSMM+BOBYQA effective at finding multiple minimizers
  ... without sacrificing quality of approximate global soln
- Asymptotic convergence results limit the number of expensive local runs
- POSMM admits additional concurrency, reduces time to solution
- Scales to concurrency above problem dimension, number of minimizers, number of desired minimizers, ...
- Asynchronous logic allows for heterogeneous function evaluation times
- Preprint + python software [Larson, W. (2016)]
  [Larson & W. "A Batch, Derivative-free Algorithm for Finding Multiple Local Minima." (OptEng, 2015)]

Current work:

- Tests for greater concurrency levels
- Interaction with local solver (use of information, concurrent evals, termination, addressing (stochastic) noise, ...)

Gracias y Felicidades Jorge!