Optimal Design Using Sampling Methods

C. T. Kelley

Department of Mathematics
Center for Research in Scientific Computation
North Carolina State University
Raleigh, North Carolina, USA

UNC OR Seminar, Oct 20, 2004

Supported by NSF, ARO, DOEd.

Outline

- Sources and Collaborators
- The problem
  - Optimization Landscapes
  - Difficulties
  - Deterministic Sampling
- Coordinate search and convergence
- Generalizations, elaborations, and descriptive results
- Implict filtering
- Example
- Constraints?

Sources

- Nelder+Mead, Hooke+Jeeves
- Dennis (+) Audet (+) Torczon (+) Lewis
- Richard Carter
- Margaret Wright
- Conn, Toint, Scheinberg
- Don Jones
- 10^6 others

Collaborators

- IFFCO developers from NCSU Math:
  Tony Choi, Owen Eslinger, Paul Gilmore,
  Alton Patrick, Dan Finkel

Collaborators

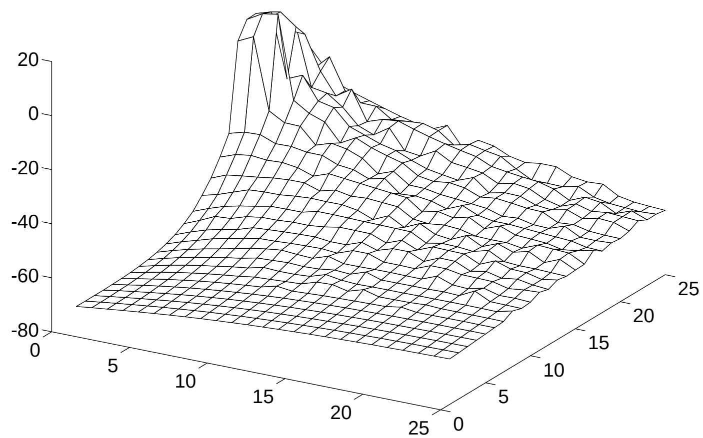
- IFFCO developers from NCSU Math:
  Tony Choi, Owen Eslinger, Paul Gilmore,
  Alton Patrick, Dan Finkel
- NCSU Math: David Bortz, Jörg Gablonsky,
  Katie Kavanagh, Jill Reese, Todd Coffey, Dan Finkel

Collaborators

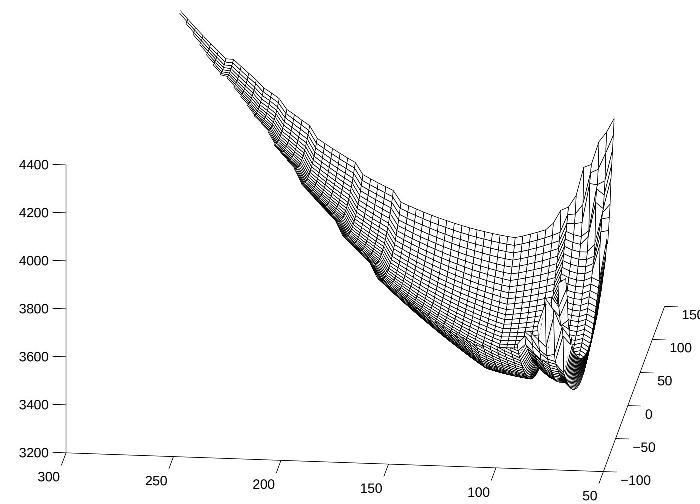
- IFFCO developers from NCSU Math:
  Tony Choi, Owen Eslinger, Paul Gilmore,
  Alton Patrick, Dan Finkel
- NCSU Math: David Bortz, Jörg Gablonsky,
  Katie Kavanagh, Jill Reese, Todd Coffey, Dan Finkel
- Clients:
  - NCSU ECE: Bob Trew, Griff Bilbro, Dan Stoneking
  - NCSU MAE: Joe David, C. Y. Cheng
  - UNC: Casey Miller, Chris Kees, Glenn Williams
  - Stoner and Asso: Richard Carter
  - Univ. Trier: Astrid Battermann

Optimization Landscapes

Semiconductors



Gas Pipelines



Applications: semiconductor, automotive, aeronautical, environmental, energy, . . .

Objectives:

- Useful decrease in the function
- Capture smooth part; avoid entrapment by local minima

What are the problems?

- Optimization/parameter identification

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators
  - No source code or too much source code

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators
  - No source code or too much source code
  - Non-differentiable control structures
    eg `if-then`

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators
  - No source code or too much source code
  - Non-differentiable control structures
    eg `if-then`
  - Discontinuities
    inner iterations, adaptivity

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators
  - No source code or too much source code
  - Non-differentiable control structures
    eg `if-then`
  - Discontinuities
    inner iterations, adaptivity
  - Non-smooth/discontinuous physics
    shocks, phase transitions, . . .

What are the problems?

- Optimization/parameter identification
- Goals: useful improvement for a few calls to the objective/constraint functions
- Black-box simulators
  - No source code or too much source code
  - Non-differentiable control structures
    eg `if-then`
  - Discontinuities
    inner iterations, adaptivity
  - Non-smooth/discontinuous physics
    shocks, phase transitions, . . .
- Non-deterministic simulators

Optimization Problem

$$\min_{x \in \Omega} f(x)$$

- Conventional Newton-based methods can fail if f is
  - multi-modal,
  - non-convex,
  - discontinuous,
  - non-deterministic, or if
- Ω is not determined by smooth inequalities.

Sampling methods attempt to address these problems.

Stencil-based sampling methods

- Begin with a base point x.
- Examine points on a stencil;
  reject or fix points not in Ω.
- Determine location and/or shape of new stencil.
- If $f(x)$ is smallest, perhaps shrink the stencil.

Stencil-based sampling methods

- Begin with a base point x.
- Examine points on a stencil;
  reject or fix points not in Ω.
- Determine location and/or shape of new stencil.
- If $f(x)$ is smallest, perhaps shrink the stencil.

Examples: Grid-based: Coordinate Search, Hooke-Jeeves, (P)MDS, GPS
Grid-free: Nelder-Mead, Implicit Filtering
But not: DIRECT, GA, SA

Stencil-based sampling methods

- Begin with a base point x.
- Examine points on a stencil;
  reject or fix points not in Ω.
- Determine location and/or shape of new stencil.
- If $f(x)$ is smallest, perhaps shrink the stencil.

Examples: Grid-based: Coordinate Search, Hooke-Jeeves, (P)MDS, GPS
Grid-free: Nelder-Mead, Implicit Filtering
But not: DIRECT, GA, SA

This is not global optimization.

Brothers and sisters, beware!

Learn from the world's easiest problem.

Minimize $x^T x$ with a sampling method and see that ...

- Sampling methods are not substitutes for Newton's method.

Brothers and sisters, beware!

Learn from the world's easiest problem.

Minimize $x^T x$ with a sampling method and see that ...

- Sampling methods are not substitutes for Newton's method.
  - Many do poorly for very easy problems.
    We can fix some of that.

Brothers and sisters, beware!

Learn from the world's easiest problem.

Minimize $x^T x$ with a sampling method and see that ...

- Sampling methods are not substitutes for Newton's method.
  - Many do poorly for very easy problems.
    We can fix some of that.
  - They need many calls to f even when doing well.
    Very expensive, especially for large problems.

Brothers and sisters, beware!

Learn from the world's easiest problem.

Minimize $x^T x$ with a sampling method and see that ...

- Sampling methods are not substitutes for Newton's method.
  - Many do poorly for very easy problems.
    We can fix some of that.
  - They need many calls to f even when doing well.
    Very expensive, especially for large problems.
- The theory is descriptive rather than predictive.

Brothers and sisters, beware!

Learn from the world's easiest problem.

Minimize $x^T x$ with a sampling method and see that . . .

- Sampling methods are not substitutes for Newton's method.
  - Many do poorly for very easy problems.
    We can fix some of that.
  - They need many calls to f even when doing well.
    Very expensive, especially for large problems.
- The theory is descriptive rather than predictive.
- The iteration will stagnate if you use smaller stencils that the quality of f will support.

Example: coordinate search

Sample f at x on a stencil centered at x, scale=h

$$S(x,h) = \{x \pm he_i\}$$

- Move to the best point.
- If x is the best point, reduce h.
- Break ties any way you like.

Necessary Conditions: no legal downhill direction (which is why you reduce h).

What if x is the best point?

If f is smooth and

$f(x) \leq \min_{z \in S(x,h)} f(z)$ **(stencil failure)**
then
$\|\nabla f(x)\| = O(h)$ which leads to …

What if x is the best point?

If f is smooth and

$f(x) \leq \min_{z \in S(x,h)} f(z)$ **(stencil failure)**
then
$\|\nabla f(x)\| = O(h)$ which leads to . . .

Theory: If (x_n, h_n) are the points/scales generated by coordinate search and f has bounded level sets, then

- $h_n \to 0$ (finitely many grid points/level) and therefore
- any limit point of $\{x_n\}$ is a critical point of f.

What if x is the best point?

If f is smooth and

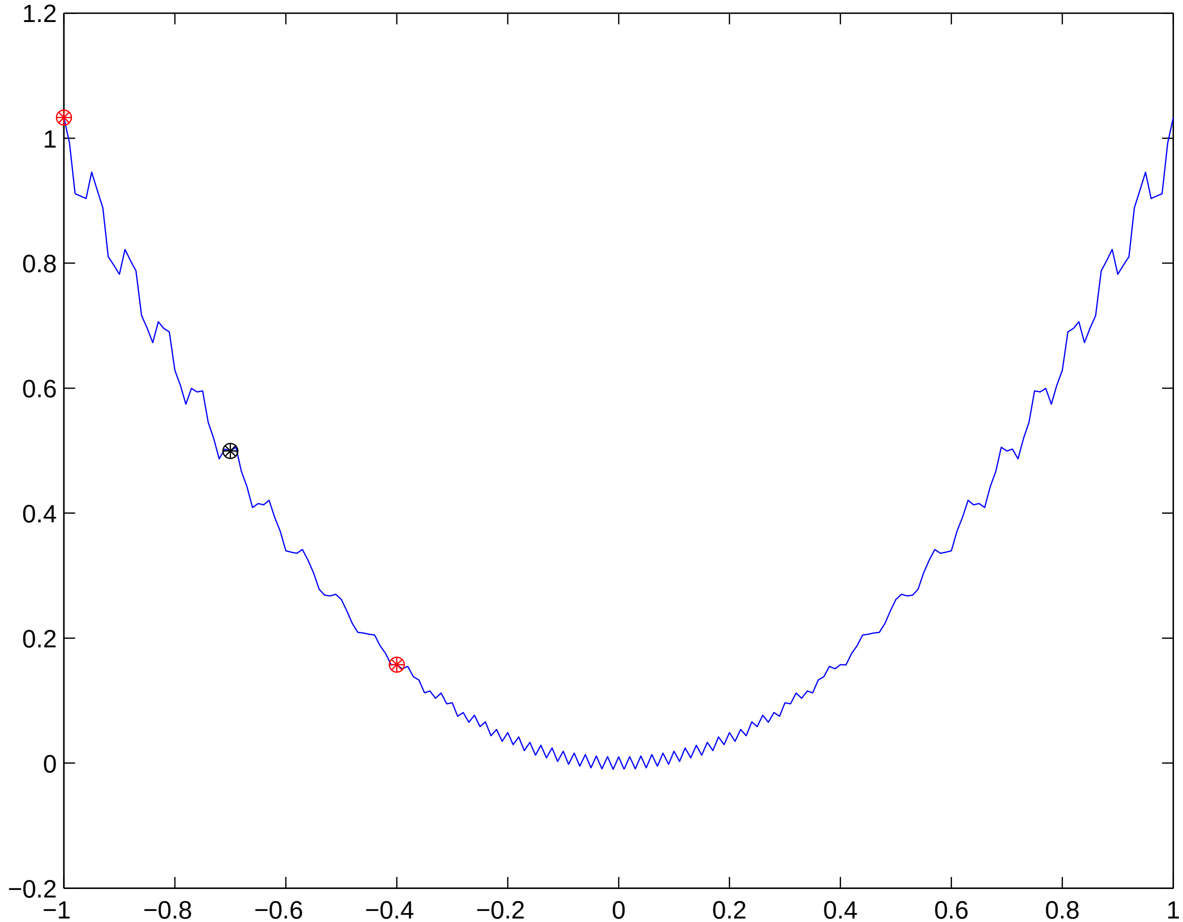
$f(x) \leq \min_{z \in S(x,h)} f(z)$ **(stencil failure)**
then
$\|\nabla f(x)\| = O(h)$ which leads to . . .

Theory: If (x_n, h_n) are the points/scales generated by coordinate search and f has bounded level sets, then

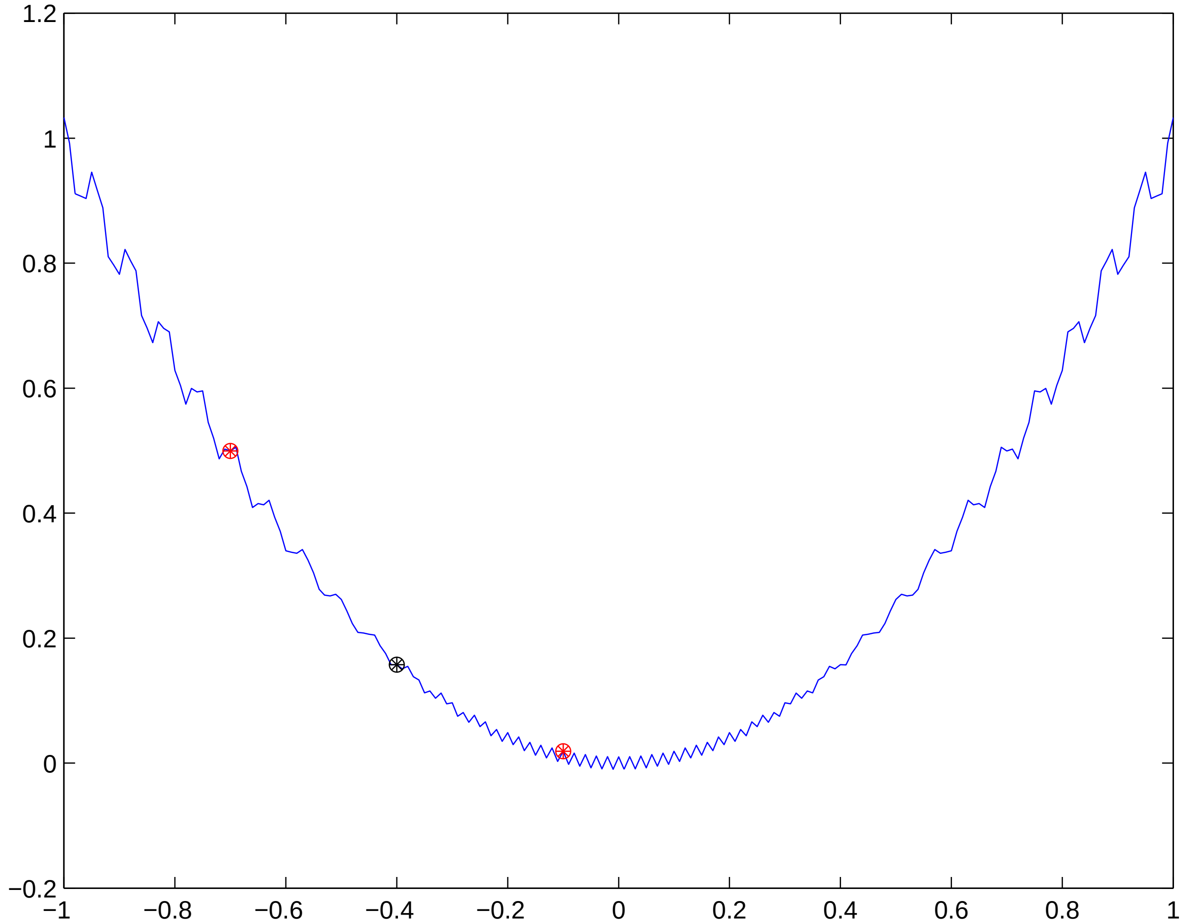
- $h_n \to 0$ (finitely many grid points/level) and therefore
- any limit point of $\{x_n\}$ is a critical point of f.

But these are not methods for smooth problems.

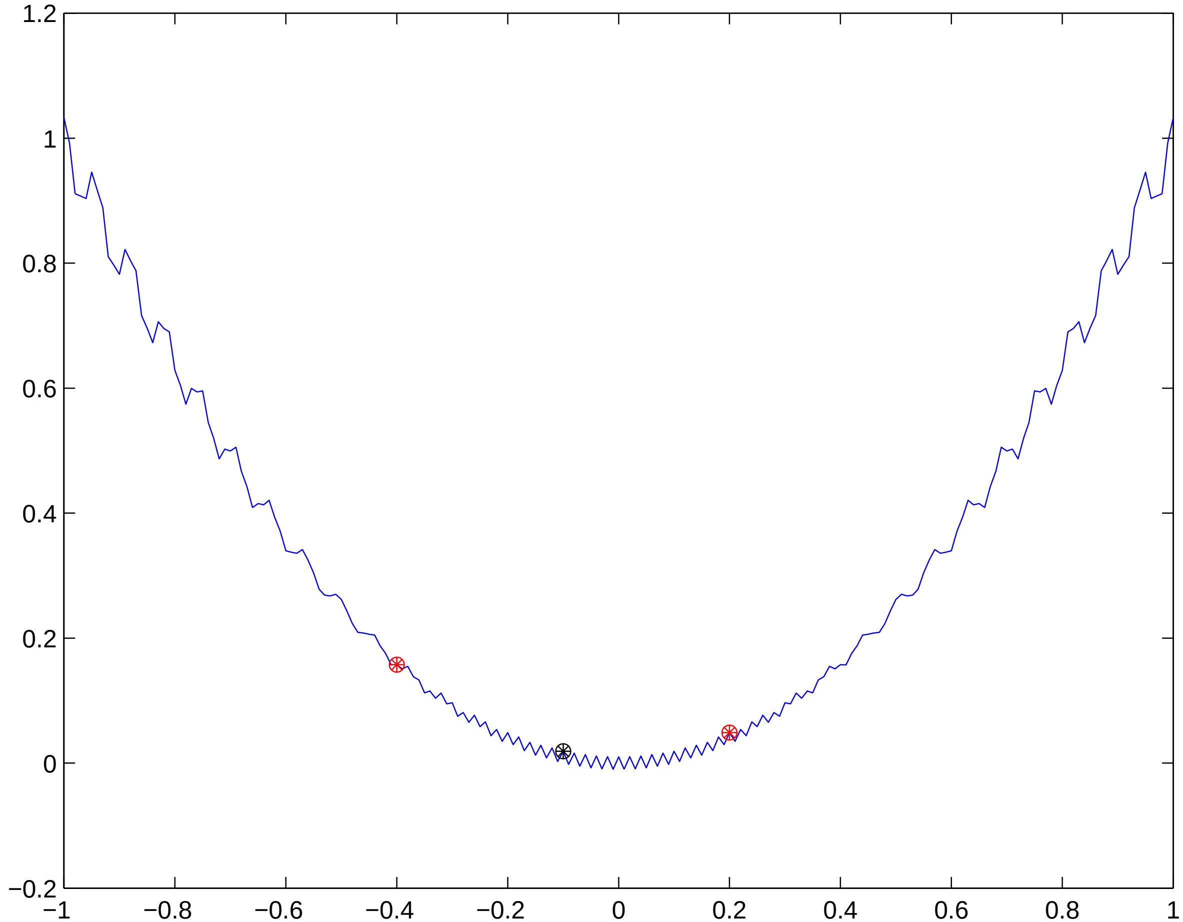
Coordinate Search: Start



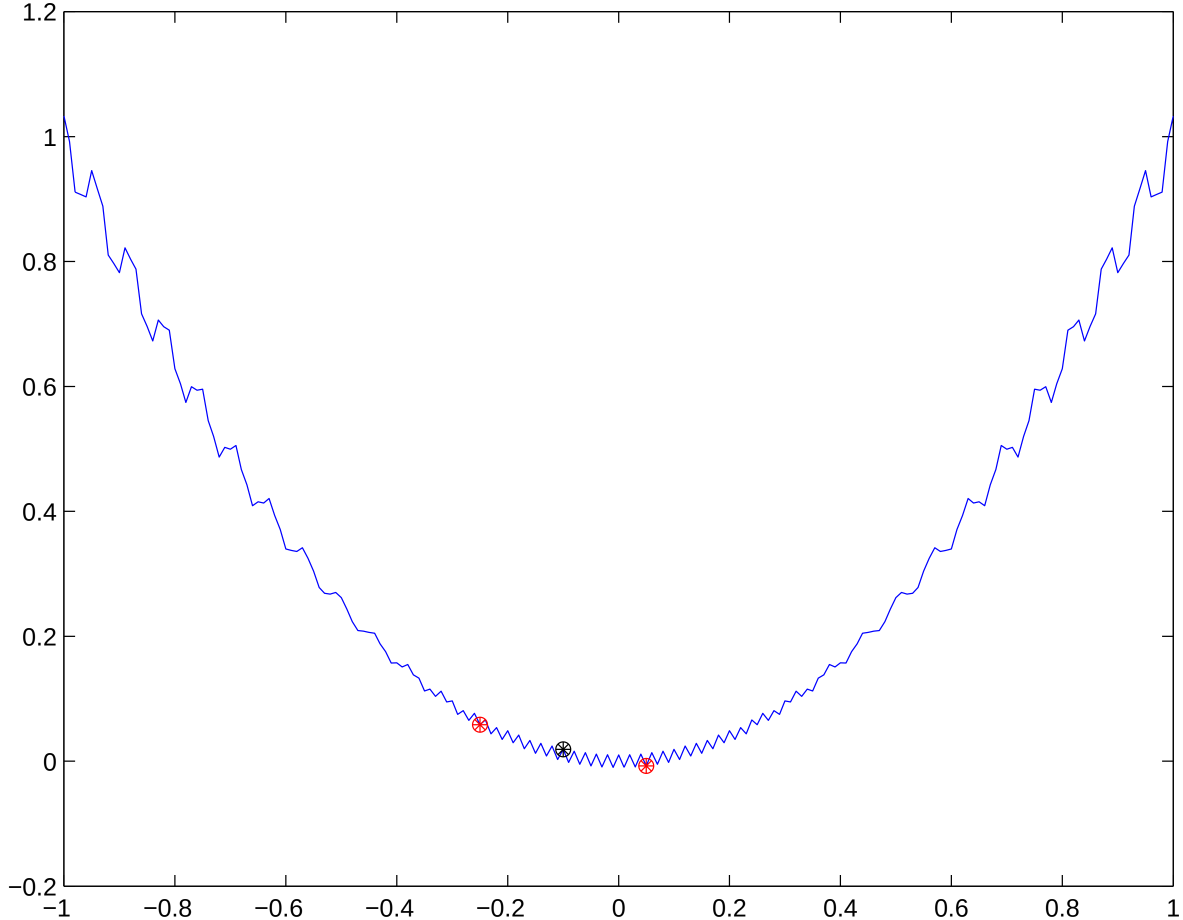
Coordinate Search: Move



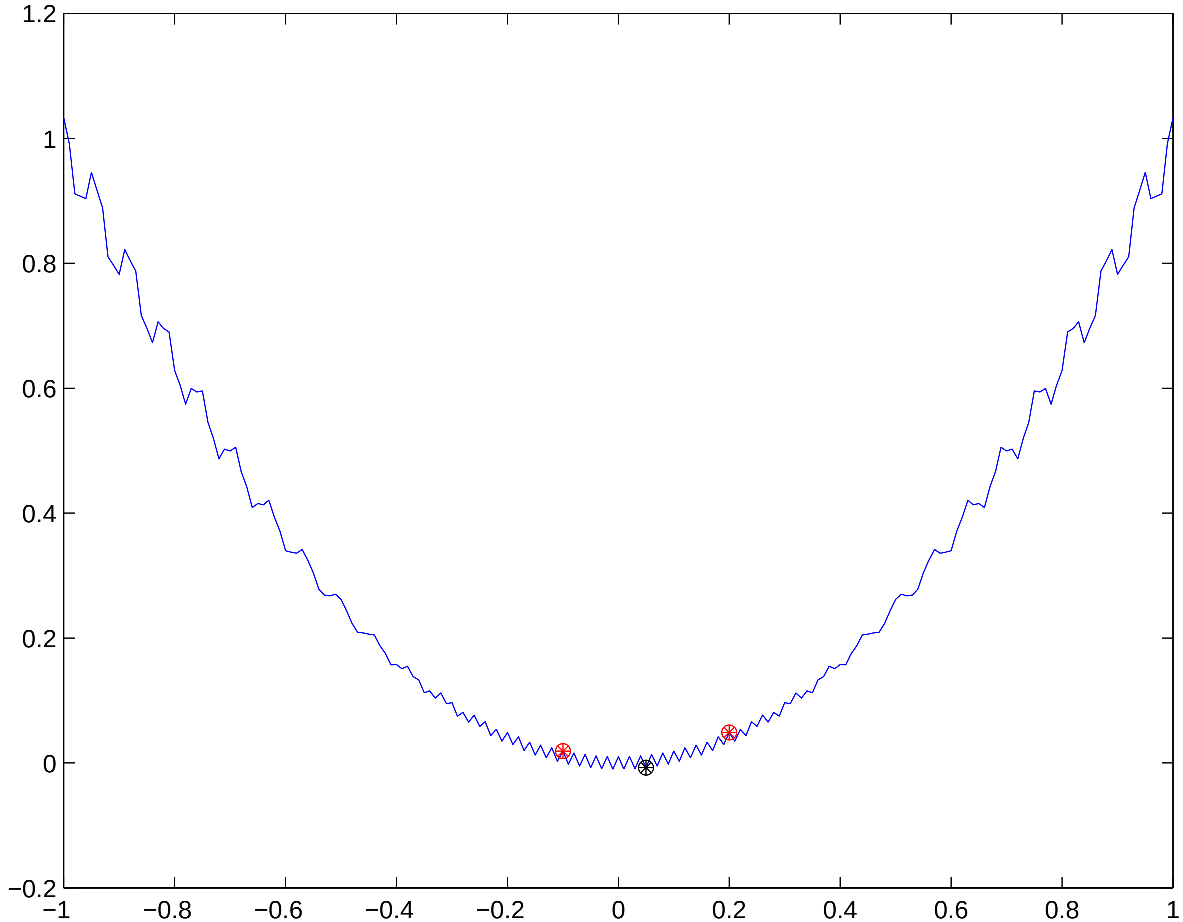
Coordinate Search: Stencil Failure



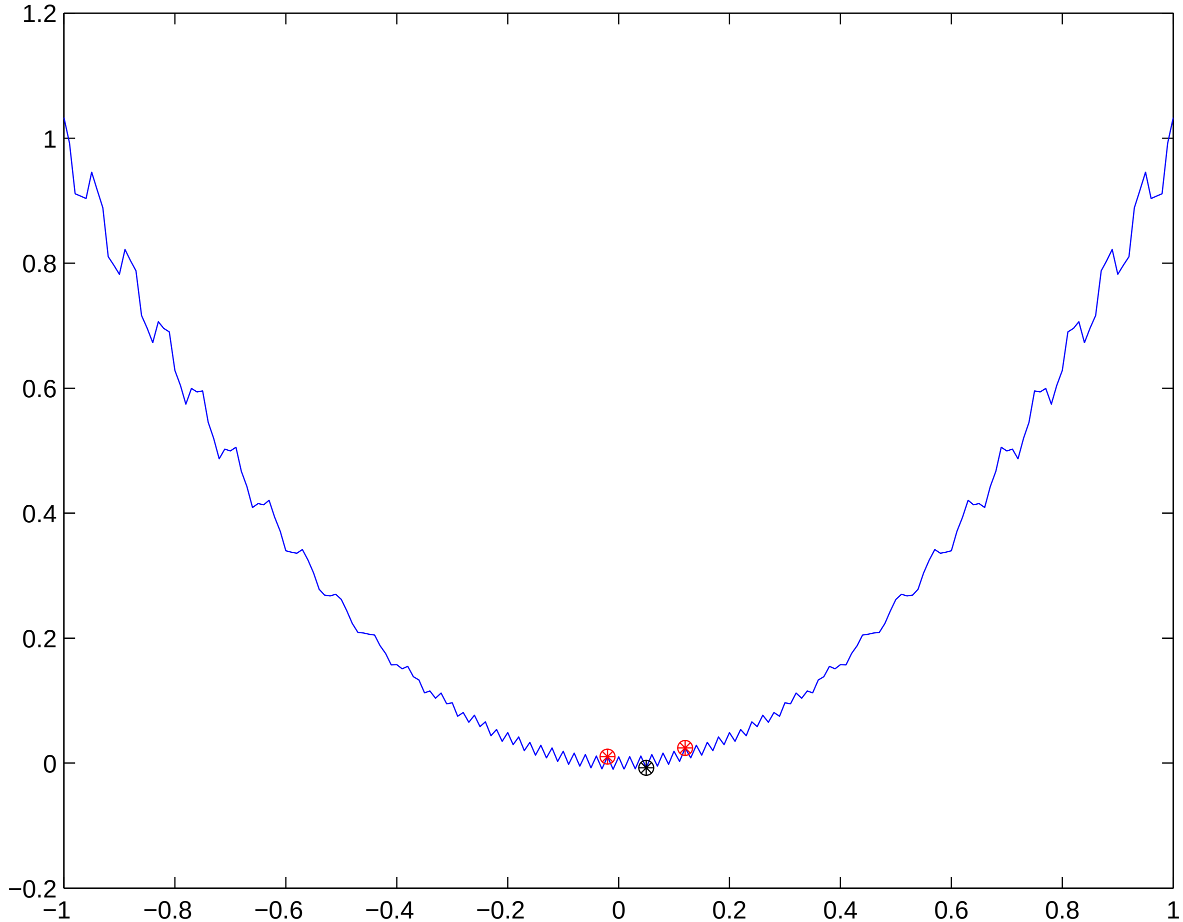
Coordinate Search: Shrink/Move



Coordinate Search: Stencil Failure



Coordinate Search: Termination



Elaborations used in practice

- Take the first better point and move (HJ, MDS, GPS)
- Adapt the shape of the stencil (NM)
- Use a stencil with fewer points (NM,IF,GPS)
- Build a model gradient (IF,DFO)
- Build a model Hessian (IF,DFO)
- Parallel evaluation of f (IF,PDS,GPS)
- Bound constraints built in (HJ,IF,GPS)
- Restarts (almost everybody)
- Categorical variables (GPS)

If you wind up with coordinate search if the elaborations fail, then you get a convergence result.

Model Problem
motivated by the pictures

$$\min_{R^N} f$$

$$f = f_s + \phi$$

- f_s smooth, easy to minimize; ϕ noise
- N is small, f is typically costly to evaluate.
- f has multiple local minima
  which trap most gradient-based algorithms.

How much should this cost?

- Suppose f is a convex quadratic.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N+1$ calls to f.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N+1$ calls to f.
  - CG terminates after N gradients.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N + 1$ calls to f.
  - CG terminates after N gradients.
- So, for a very easy problem,

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N+1$ calls to f.
  - CG terminates after N gradients.
- So, for a very easy problem,
  - Best possible is $2N^2+O(N)$ calls to f.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N+1$ calls to f.
  - CG terminates after N gradients.
- So, for a very easy problem,
  - Best possible is $2N^2+O(N)$ calls to f.
  - Be happy with $O(N^2)$.

How much should this cost?

- Suppose f is a convex quadratic.
  - Centered differences are exact.
  - One gradient evaluation costs $2N+1$ calls to f.
  - CG terminates after N gradients.
- So, for a very easy problem,
  - Best possible is $2N^2+O(N)$ calls to f.
  - Be happy with $O(N^2)$.
- But, Newton needs one gradient+Hessian.
  Don't discard your conventional codes.

Convergence?

Stencil failure implies that

$$\|\nabla f_s(x_n)\| = O\left(h_n + \frac{\|\phi\|_{S(x_n,h_n)}}{h_n}\right)$$

where

$$\|\phi\|_{S(x,h)} = \max_{z\in S} |\phi(z)|.$$

Bottom line

So, if (x_n, h_n) are the points/scales generated by coordinate search, f has bounded level sets, **and**

$$\lim_{n \to \infty} (h_n + h_n^{-1} \|\phi\|_{S(x,h_n)}) = 0$$

then

- $h_n \to 0$ (finitely many grid points) and therefore
- any limit point of $\{x_n\}$ is a critical point of f.

Limits

- Theory is descriptive, not predictive.

Limits

- Theory is descriptive, not predictive.
    - Most problems you care about don’t satisfy assumptions.

Limits

- Theory is descriptive, not predictive.
    - Most problems you care about don’t satisfy assumptions.
    - The sequence of useful scales is finite.
      The iteration will stagnate.

Limits

- Theory is descriptive, not predictive.
  - Most problems you care about don't satisfy assumptions.
  - The sequence of useful scales is finite. The iteration will stagnate.
- Theory is behind practice.

Limits

- Theory is descriptive, not predictive.
  - Most problems you care about don't satisfy assumptions.
  - The sequence of useful scales is finite.
    The iteration will stagnate.
- Theory is behind practice.
- Practice is behind applications.

Limits

- Theory is descriptive, not predictive.
  - Most problems you care about don't satisfy assumptions.
  - The sequence of useful scales is finite. The iteration will stagnate.
- Theory is behind practice.
- Practice is behind applications.

Even so, the theory provides useful guidance.

Implicit Filtering

Accelerate coordinate search with a quasi-Newton method.

imfilter$(x, f, pmax, \tau, \{h_n\}, amax)$

    for $k = 0, \dots$ **do**

        `fdquasi`$(x, f, pmax, \tau, h_n, amax)$

    end for

$pmax$, τ, $amax$ are termination parameters

fdquasi = finite difference quasi-Newton method using a central difference gradient $\nabla_h f$.

fdquasi$(x, f, pmax, \tau, h, amax)$

$p = 1$

while $p \leq pmax$ and $\|\nabla_h f(x)\| \geq \tau h$ **do**

compute f and $\nabla_h f$

terminate with success on stencil failure

update the model Hessian H if appropriate; solve

$Hd = -\nabla_h f(x)$

use a backtracking line search, with at most $amax$ backtracks,

to find a step length λ

Failure: leave x unchanged if $> amax$ backtracks

$x \leftarrow x + \lambda d$; $p \leftarrow p + 1$

end while

Failure: if $p > pmax$ leave x unchanged

Termination

Calculus implies that

$$\nabla_h f(x) = \nabla f_s(x) + O(h^2 + \|\phi\|_{S(x,h)}/h).$$

So if fdquasi terminates with success:

- $\|\nabla_h f(x)\| \leq \tau h$ **(small gradient condition)**
- Stencil failure

(*i. e.* there is no line search or outer iteration failure) then

$$\|\nabla f_s(x)\| = O(h + \|\phi\|_{S(x,h)}/h)$$

leading to . . .

Basic Convergence Theorem

Let (x_n, h_n) be the sequence from implicit filtering.
If

- ∇f_s is Lipschitz continuous.
- $\lim_{n\to\infty}(h_n + h_n^{-1}\|\phi\|_{S(x,h_n)}) = 0$
- fdquasi terminates with success for infinitely many n.

then any limit point of $\{x_n\}$ is a critical point of f_s.

Basic Convergence Theorem

Let (x_n, h_n) be the sequence from implicit filtering.
If

- ∇f_s is Lipschitz continuous.
- $\lim_{n\to\infty}(h_n + h_n^{-1}\|\phi\|_{S(x,h_n)}) = 0$
- fdquasi terminates with success for infinitely many n.

then any limit point of $\{x_n\}$ is a critical point of f_s.

- Same theory as coordinate search,
  unless you use a clever $\{h_n\}$
- Very different in practice.

Model Hessians

- BFGS for unconstrained
- Projected SR1 for bound constraints
- Gauss-Newton for least squares

Model Hessians

- BFGS for unconstrained
- Projected SR1 for bound constraints
- Gauss-Newton for least squares

Theory: If

- ϕ decays sufficiently rapidly near optimality
- $h_n \to 0$ fast enough

then (BFGS, GN) you get superlinear convergence.

Model Hessians

- BFGS for unconstrained
- Projected SR1 for bound constraints
- Gauss-Newton for least squares

Theory: If

- ϕ decays sufficiently rapidly near optimality
- $h_n \to 0$ fast enough

then (BFGS, GN) you get superlinear convergence.

Practice: the method performs poorly without the quasi-Newton Hessian.

How to get the software

- IFFCO: Implicit Filtering For Constrained Optimization
- New version released May, 2001
  MPI/PVM/Serial
- ftp to ftp.math.ncsu.edu in
  FTP/kelley/iffco/IFFCO.tar.gz or email to
  Tim_Kelley@ncsu.edu
  http://www4.ncsu.edu/˜ctk
  http://www4.ncsu.edu/˜ctk/iffco.html
- Next major version, 2005.

Example: Hydraulic Capture

- Control flow of contaminants in groundwater.
  - Keep plume on site.
  - Keep concentrations at acceptable levels.
  - Minimize cost, volume of contaminant, contaminant concentration . . .
- Control flow and pressure.
  - Municipal water supplies.
  - Agriculture.

Many approaches

- Tightly coupled simulation/optimization (Shoemaker)
- GAs (Mayer, Pinder, Minkser, Yeh . . .)
- Surrogates: response surface, neural nets

Our objectives:

- Examine many formulation, simulator, optimizer combinations in a portable way.
- Build testbed for both groundwater and optimization communities.
- Design new approaches.

Today: one problem/simulator/optimizer triple.

What we do.

- Black-box optimization:
  Use accepted, widely-used, production 3D simulators.
  - Improved portability/documentation relative to research codes.
  - No guarantee of differentiability wrt design variables.
- Put problems/solutions on the web.
  http://www4.ncsu.edu/~ctk/community.html

Flow in the saturated zone

$$S_s \frac{\partial h}{\partial t} = \nabla \cdot (K \nabla h) + \mathscr{S},$$

Data:

- BC, IC, spatial domain Ω
- S_s (specific storage coefficient)
- K (hydraulic conductivity)
- $\mathscr{S}$ is the souce/sink term,
  computed from the design variables.

Output: h (hydraulic head)
Typical simulators: ADH, FEMWATER, MODFLOW.

Species Transport

$$\frac{\partial \theta C}{\partial t} = \nabla \cdot (\theta \mathbf{D} \cdot \nabla C) - \nabla \cdot (\theta \mathbf{v} C) + \mathscr{S}^C.$$

Data: porosity θ, interphase
Design: $\mathscr{S}^C$ mass sources/sinks

- C is concentration, solution of PDE;
- v is velocity, computed from h;
- $\mathbf{D}$ is the dispersion tensor, computed from h.

Computing the fluid velocity, v

Darcy's law says

$$\theta v = \frac{k}{\mu}(\nabla p + \rho g \nabla z)$$

- $p = \rho g(h - z)$: fluid pressure
- k: intrinsic permeability; μ: dynamic viscosity
- ρ: density; g: gravitational acceleration
- ∇z: vector in vertical direction

What's $\mathbf{D}$

$$\mathbf{D}_{ij} = \delta_{ij}\alpha_t|v| + (\alpha_l - \alpha_t)\frac{v_i v_j}{|v|} + \delta_{ij}\tau D^*$$

- α_l, α_t: longitudinal/transverse dispersivities
- τ: tortuosity of the porous medium
- D^*: free liquid diffusivity.

Design variables

Number and location of wells, pumping rates.
Pumping rates and well locations go in the source term for flow

$$\int_\Omega \mathscr{S}(t)\,d\Omega = \sum_{i=1}^{n} Q_i$$

and for concentration

$$\int_\Omega \mathscr{S}^C(t)\,d\Omega = \sum_{i=1}^{n} C(x_i) Q_i.$$

Examples:

- Sum of δ functions at well locations.
- Well model with well diameter, well type, ...

Example: Hydraulic Capture

Minimize total cost:

$$f^T(Q) = \underbrace{\sum_{i=1}^{n} c_0 d_i^{b0} + \sum_{Q_i < -10^{-6}} c_1 |Q_i{}^m|^{b_1} (z_{gs} - h^{min})^{b_2}}_{f^c} +$$

$$\underbrace{\int_0^{t_f} \left(\sum_{i, Q_i < -10^{-6}} c_2 Q_i (h_i - z_{gs}) + \sum_{i, Q_i > 10^{-6}} c_3 Q_i \right) dt}_{f^o},$$

to keep a contaminant inside a “capture zone”.
$\Omega = [0, 1000] \times [0, 1000]$

Notation

- $\{(x_i, y_i)\}$ are well locations.
- Q_i is pumping rate
  (> 0 for injection, < 0 for extraction.
- d_i is depth of well i
- h_i is head at well i (MODFLOW)
- z_{gs} is elevation of ground surface
- Q^m is design pumping rate.
- h^{min} is minimum allowable pumping rate.

Boundary conditions: Unconfined aquifer

$$\left.\frac{\partial h}{\partial x}\right|_{x=0} = \left.\frac{\partial h}{\partial y}\right|_{y=0} = \left.\frac{\partial h}{\partial z}\right|_{z=0} = 0, t > 0$$

$$K\frac{\partial h}{\partial z}(x,y,z=h,t>0) = -1.903 \times 10^{-8} \text{ (m/s)}.$$

$$h(1000,y,z,t>0) = 20 - 0.001y\text{(m)},$$
$$h(x,1000,z,t>0) = 20 - 0.001x\text{(m)},$$
$$h(x,y,z,0) = h_s.$$

Constraints I

Simple bounds:

$$Q^{emax} \leq Q_i \leq Q^{imax}, \; i = 1, \ldots, n$$

Limits on the pumps.
Simple linear inequality:

$$\sum_i Q_i \geq Q_T^{max},$$

limit on total net extraction rate.

Constraints II

Keep wells away from Dirichlet boundary

$$0 \le x_i, y_i \le 800.$$

Bounds on h

$$h^{min} \le h_i \le h^{max}, \; i = 1, \ldots, n$$

No dry holes.
Velocity Highly nonlinear function of well locations.
$50 \times 50 \times 10$ grid.

Formulation Decisions I

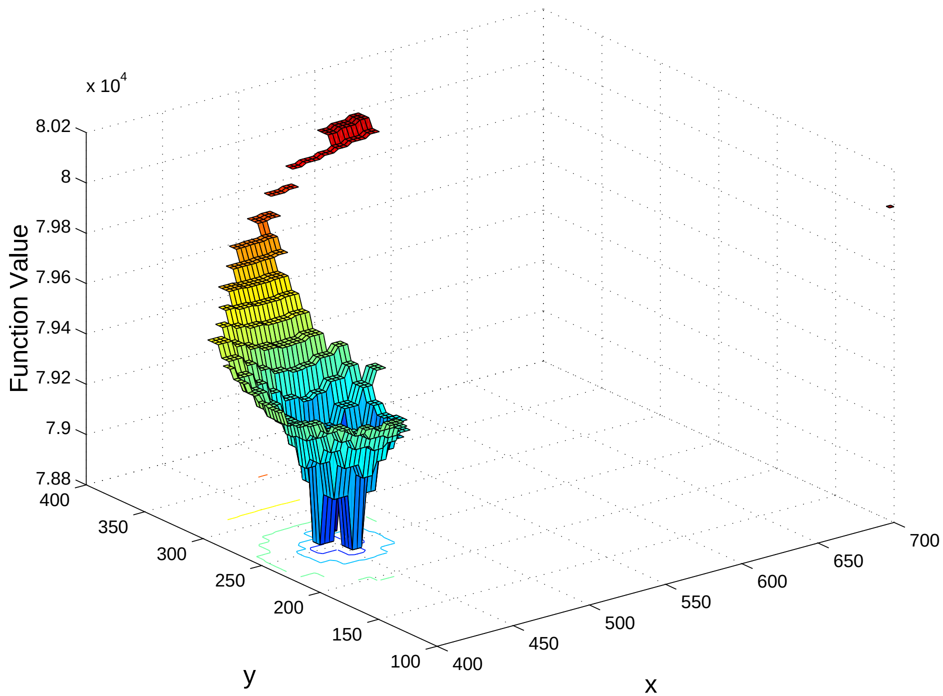
- Contain plume: constrain velocity at zone boundary.
  Test velocity at five downstream locations.
  Approximate velocity with difference of h.
  Five new constraints.
  Need only flow code. Better simulations in progress.
- Implicit filtering deals with bounds naturally.
- Treat constraints as yes/no for sampling method
  - Stratify by cost.
  - Avoid simulator if infeasible wrt cheap (linear) constraints.
- Well is de-installed if pumping rate is suff small.

Formulation Decisions II

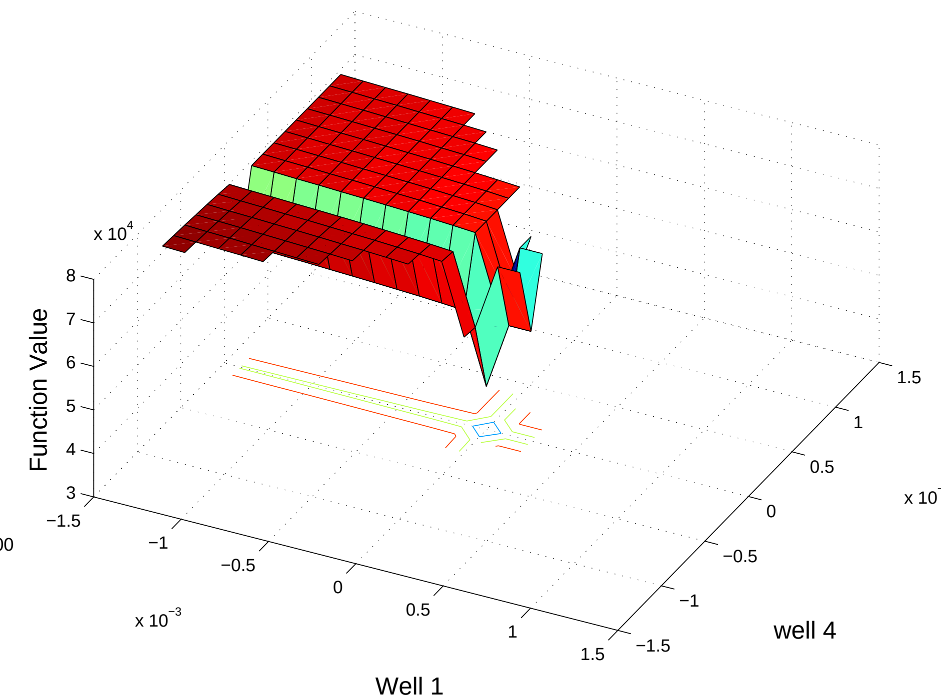
- Discontinuous objective.
  - $50 \times 50 \times 10$ grid. Wells must be on grid nodes. Move to nearest.
  - Remove well from array ($d_i = 0$) if pumping rate is too small.
- Treat head constraint and linear constraints as *hidden* or *yes-no*.
- Initial iterate: two extraction, two injection

Landscapes

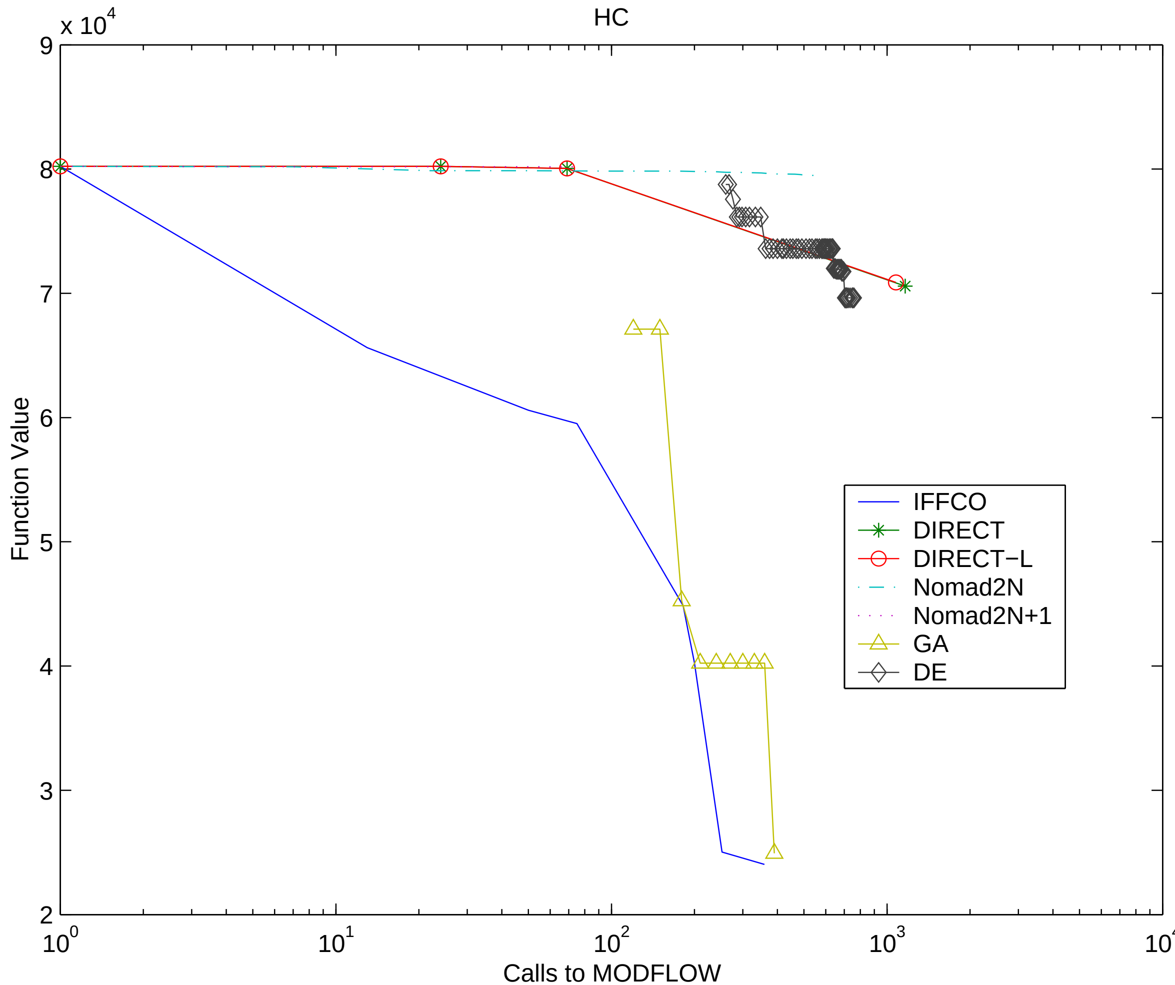
Vary (x_1, y_1) near initial iterate



Vary pumping rate initial iterate



Other Approaches



Community Problems

- Suite of problems in groundwater remediation
  3D, flow+transport, varying difficulty.
- We provide or point to simulators/optimization codes that will produce a formulation and a solution.
- No pretense that formulation or solution is best possible.
- Portable, good testbed for optimization codes.

How to get the Community Problems

- Constantly updated on
  http://www4.ncsu.edu/~ctk/community.html
- Packages include problems, makefiles, IFFCO example.
  You need to get the simulators; we tell you how.
- Tested on
  - g77: Solaris, Red Hat 7.3,8.0, MAC OSX, IBM-SP
  - MPI: IBM-SP, Dell+Red Hat 8.0
- Three problems in place (only MODFLOW).
- New problems under construction.
- Massive comparison in progress
  GA, NOMAD, Boeing DE, DIRECT, APPS

Conclusions

- Deterministic Stencil-Based Sampling Methods

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like

  are there if Newton and sampling methods fail.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like
  - DIRECT (no stencil),

  are there if Newton and sampling methods fail.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like
  - DIRECT (no stencil),
  - SA or GA (no stencil, random),

  are there if Newton and sampling methods fail.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like
  - DIRECT (no stencil),
  - SA or GA (no stencil, random),

  are there if Newton and sampling methods fail.
- We give away software and test problems.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like
  - DIRECT (no stencil),
  - SA or GA (no stencil, random),

  are there if Newton and sampling methods fail.
- We give away software and test problems.

Conclusions

- Deterministic Stencil-Based Sampling Methods
  - can solve some of your problems.
  - will not solve all of your problems.
- Newton-based codes are much better when they work.
- More aggressive methods like
  - DIRECT (no stencil),
  - SA or GA (no stencil, random),

  are there if Newton and sampling methods fail.
- We give away software and test problems.