Optimal Design of Groundwater Remediation Systems with Sampling Methods

C. T. Kelley

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

Chinese University of Hong Kong
Hong Kong, January 25, 2006

Supported by NSF, ARO, DOEd.

Outline

- Collaborators
- General formulation and example problem
  - Formulation
  - Optimization Landscapes
  - Results
- Implicit Filtering
- Community problems
- Conclusions

Collaborators

- IFFCO developers from NCSU Math:
  Tony Choi, Owen Eslinger, Paul Gilmore,
  Alton Patrick, Vincent Bannister
- NCSU Math: Corey Winton, Dan Finkel, Jörg Gablonsky,
  Katie Fowler , Chris Kees, Jill Reese, Todd Coffey
- Other Places:
  - Boeing: Andrew Booker, John Dennis
  - UNC: Casey Miller, Matt Farthing, Glenn Williams
  - ERDC: Stacy Howington
  - Univ. Trier: Astrid Battermann
  - Mich Tech: Alex Mayer

What's the problem.

- Control flow of contaminants in groundwater.
  - Keep plume on site.
  - Keep concentrations at acceptable levels.
  - Minimize cost, mass 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 long-term 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.
  - Community might listen to us.
  - 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.

Typical simulator: MT3D

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

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

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

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

Constraints I

Simple bounds:

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

Limits on the pumps.
Simple linear inequality:

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

limit on total net extraction rate.

Constraints II

Keep wells away from Dirichlet boundary

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

Bounds on h

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

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

Formulation Decisions I

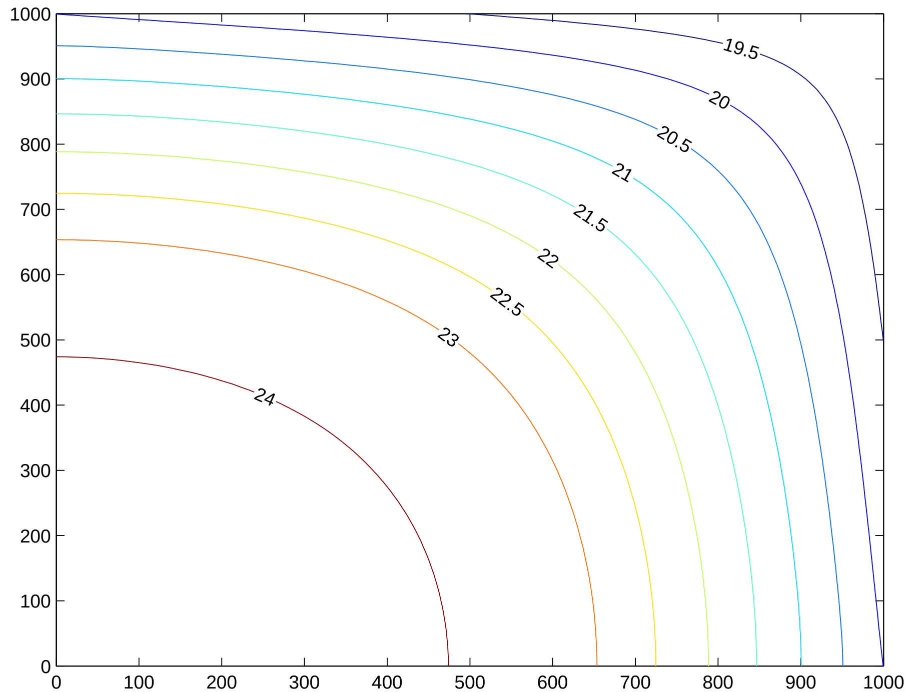
- Contain plume: constrain velocity at zone boundary.
  Test velocity at five downstream locations.
  Approximate velocity with difference of h.
  Five new constraints.
  Need only flow code. Better simulations in progress.
- Implicit filtering deals with bounds naturally.
- Treat constraints as yes/no 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

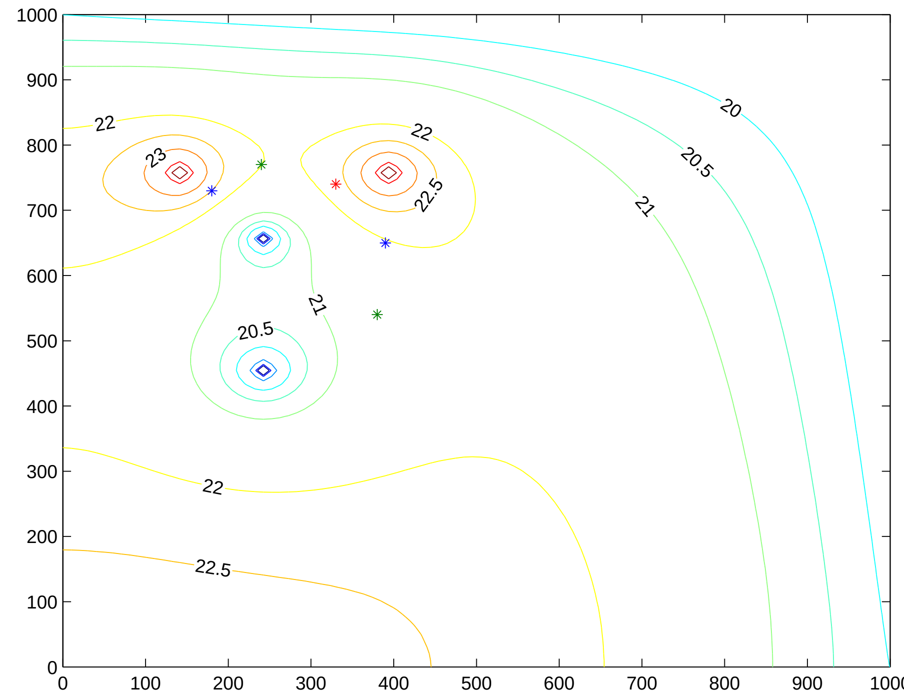
- Discontinuous objective.
  - $50 \times 50 \times 10$ grid. Wells must be on grid nodes. Move to nearest.
  - Remove well from array ($d_i = 0$) if pumping rate is too small.
- Treat head constraint and linear constraints as *hidden* or *yes-no*.
- Initial iterate: two extraction, two injection

Initial iterate

No wells

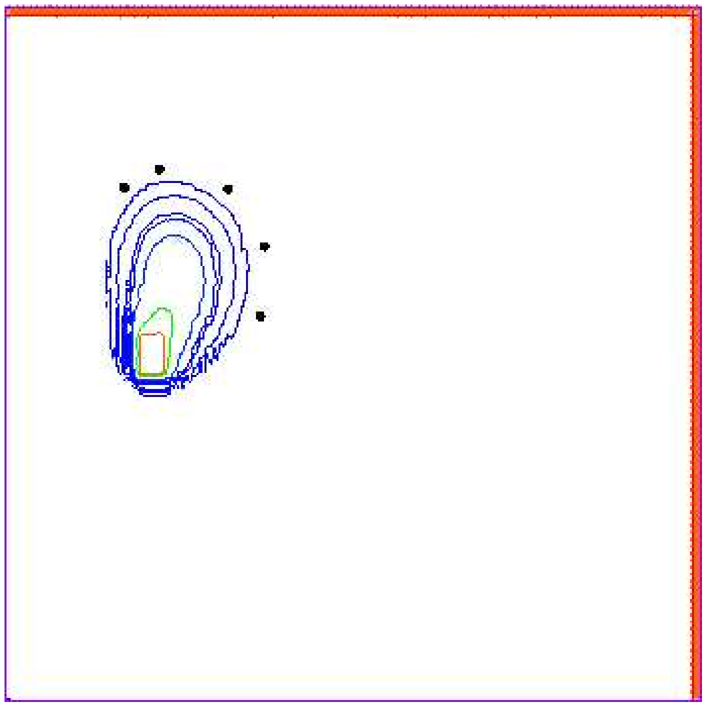


Four well configuration

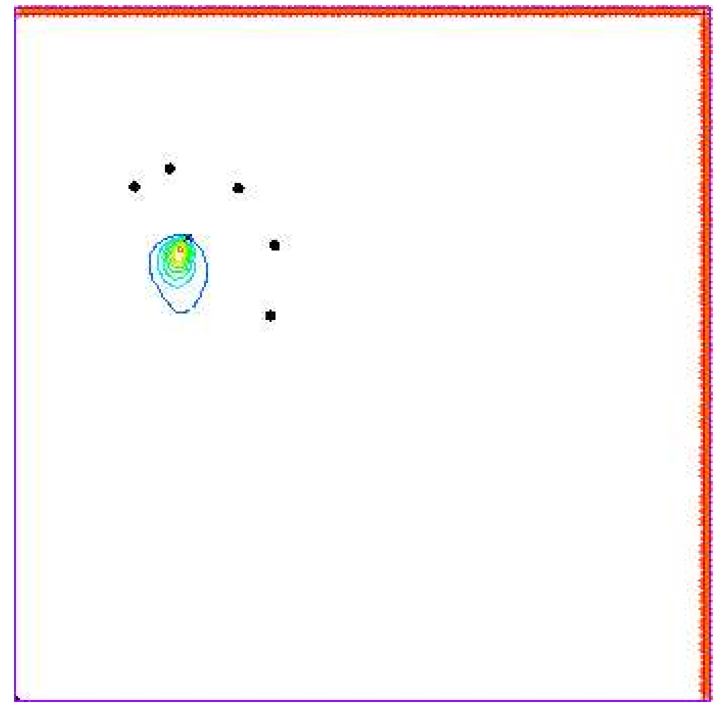


Initial/Final Plumes

Initial Plume

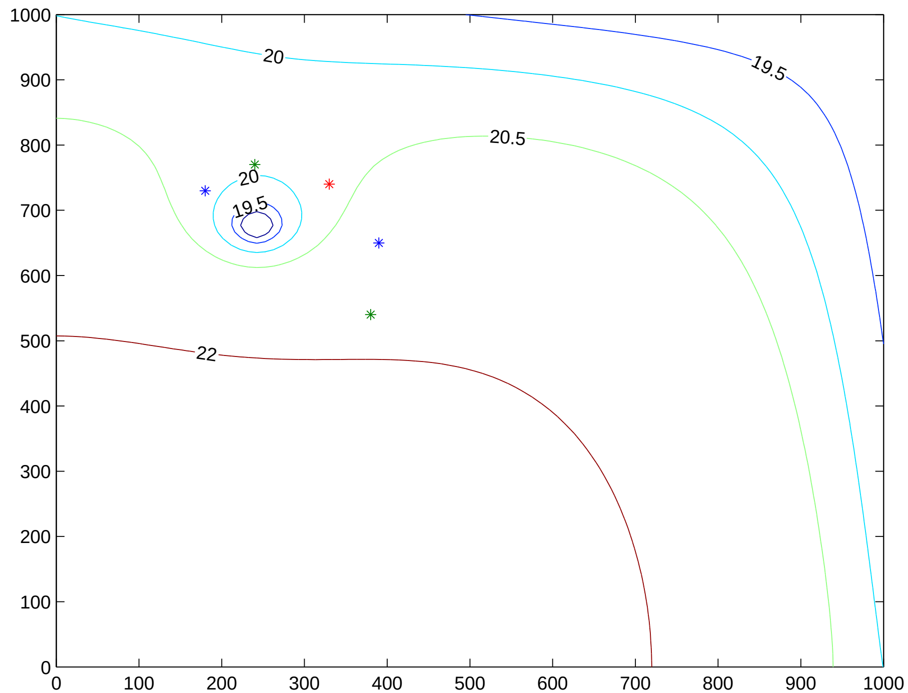


Plume after 5 years

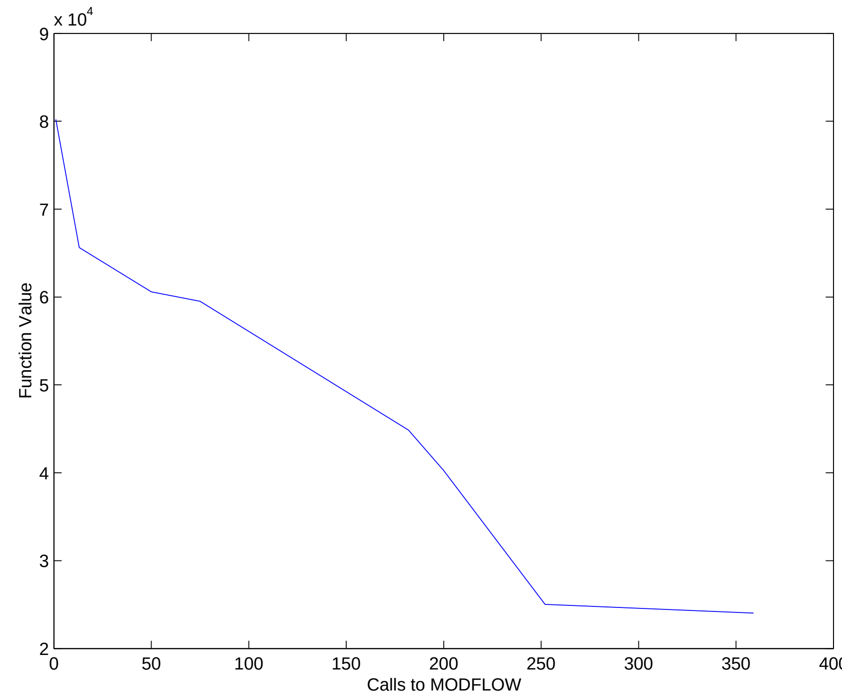


Results

Optimal configuration

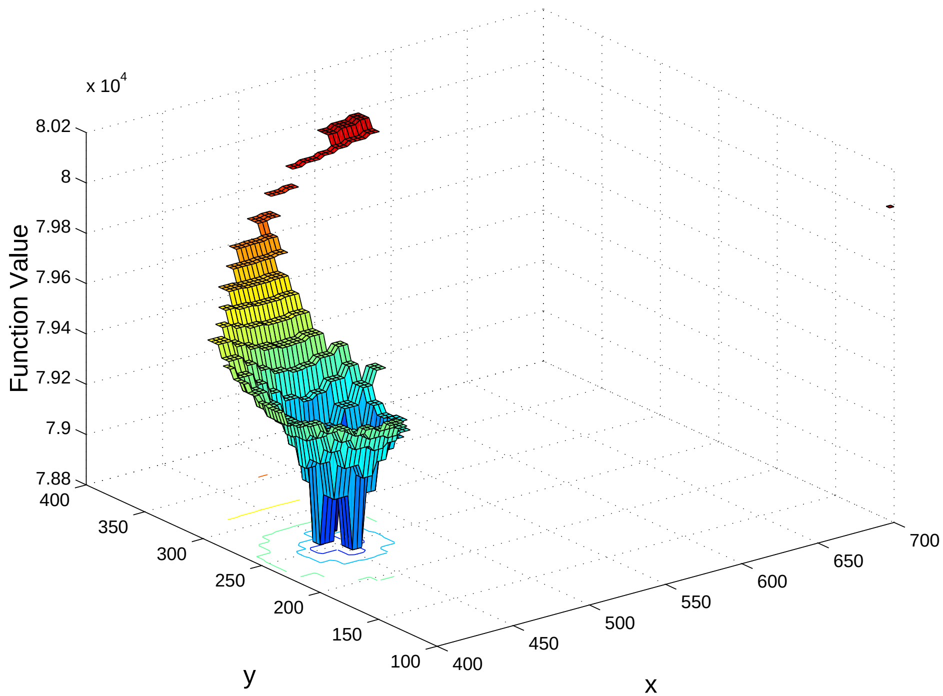


Cost of Optimization

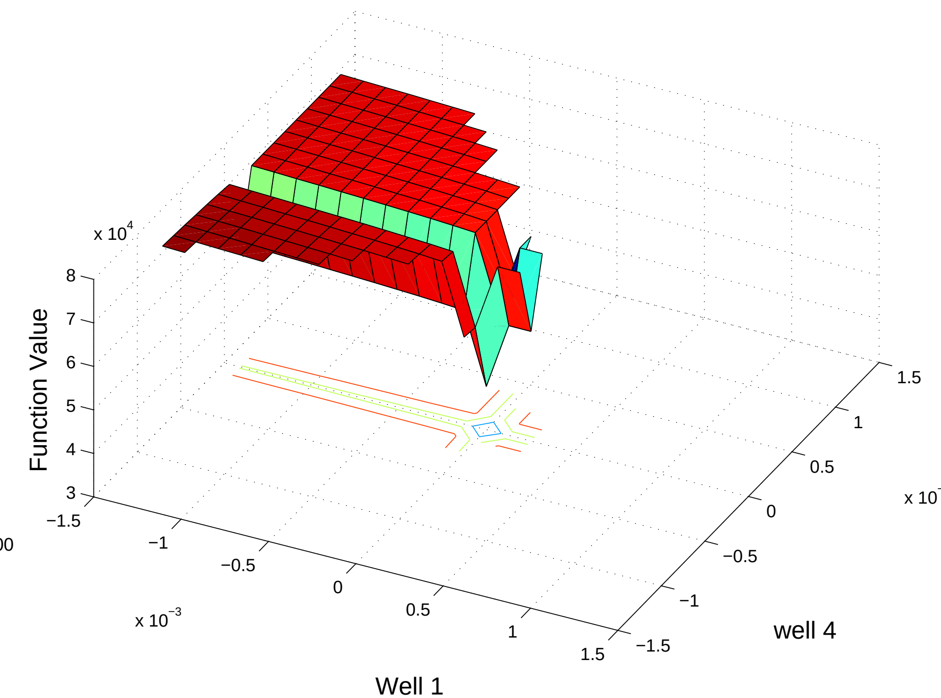


Landscapes

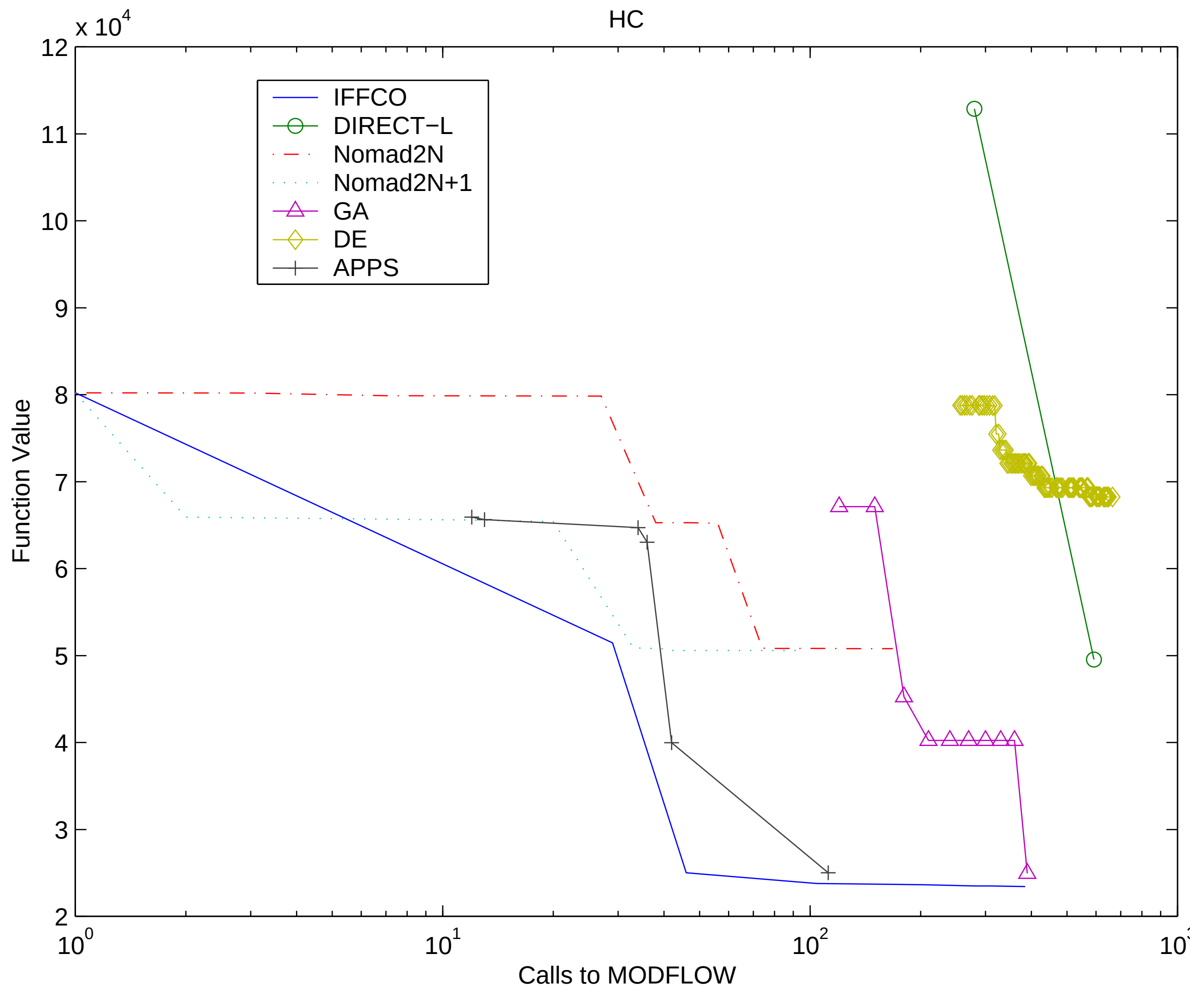
Vary (x_1, y_1) near initial iterate



Vary pumping rate initial iterate



Other Approaches



Wait a minute!

- Optimal point has one well, we start with four.
  Was this fair?
- How does performance depend on initial iterate?
- Do some methods benefit from special choices?
- How can you construct a “rich” set of initial iterates for testing?
- We’re trying:
  - Use DIRECT to find feasible points.
  - Use statistics to identify clusters.
  - Sample wisely within the clusters.

Optimization strategy

$$\min_{x \in \mathscr{D}} f(x)$$

- Conventional gradient-based methods can fail if f is
  - multi-modal,
  - non-convex,
  - discontinuous,
  - non-deterministic, or if
- $\mathscr{D}$ 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 adjust points not in $\mathscr{D}$.
- Determine location of next stencil.
- If $f(x)$ is smallest, shrink the stencil.

Examples: Coordinate Search, Nelder-Mead, Hooke-Jeeves, (P)MDS, GPS, Implicit Filtering

This is not global optimization.

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.

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

What if x is the best point?

Smooth Objective

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

So, 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.

Not a method for smooth problems.

Model Problem

motivated by the landscapes.

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

$$f = f_s + \phi$$

- 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.

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/level) and therefore
- any limit point of $\{x_n\}$ is a critical point of f.

Analysis for Hooke-Jeeves, MPS, GPS is similar.
Nelder-Mead is different.

Implicit Filtering

Accelerate coordinate search with a quasi-Newton method.

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

for $k = 0, \ldots$ **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 fi nd a step length λ

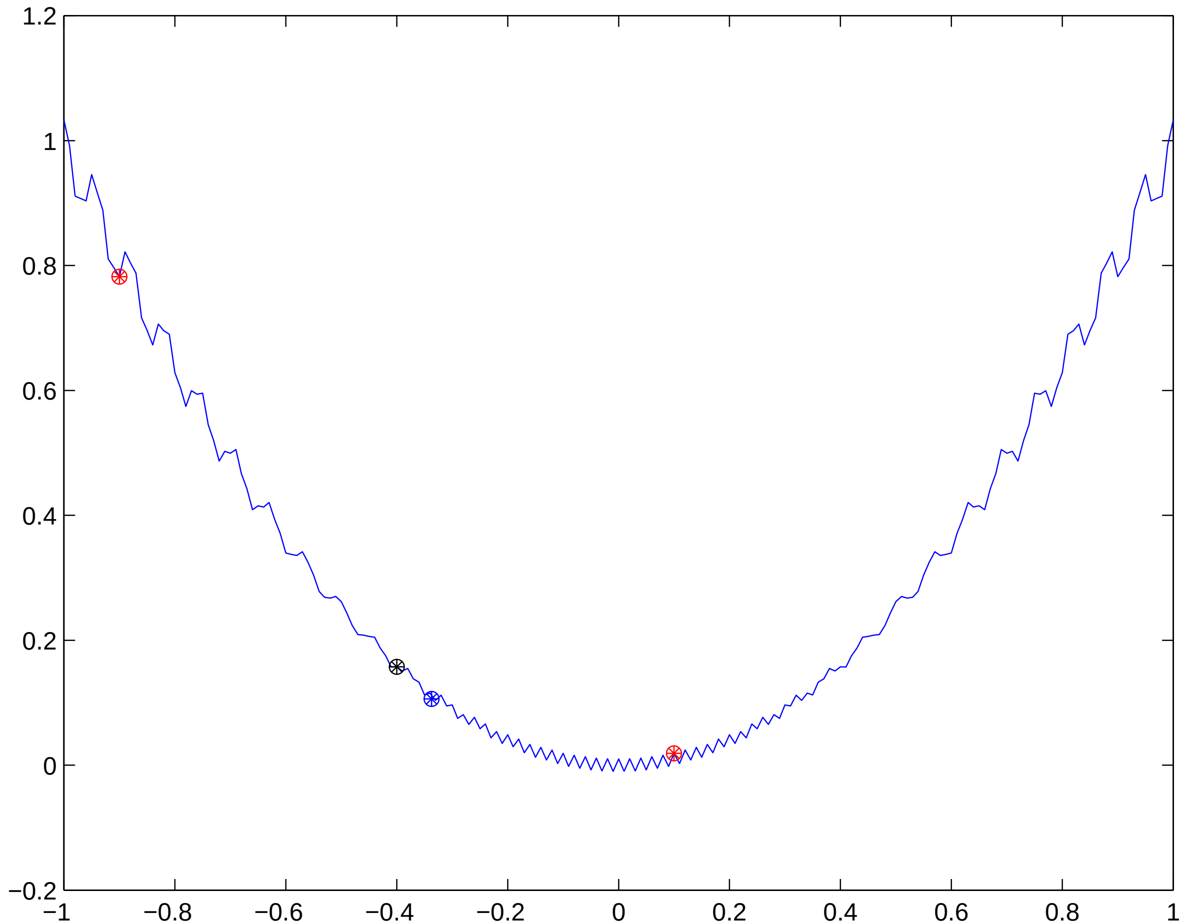
terminate with failure on $> amax$ backtracks

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

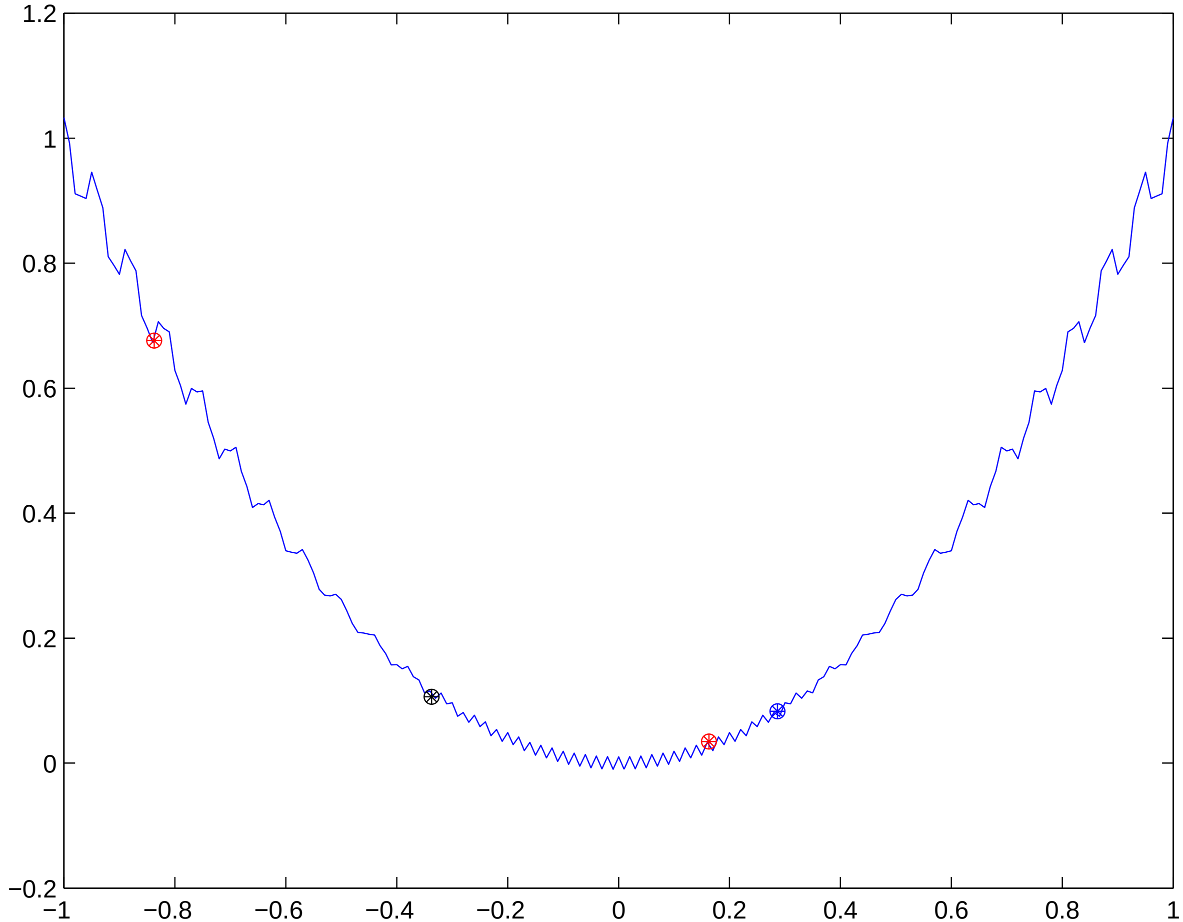
end while

if $p > pmax$ report iteration count failure

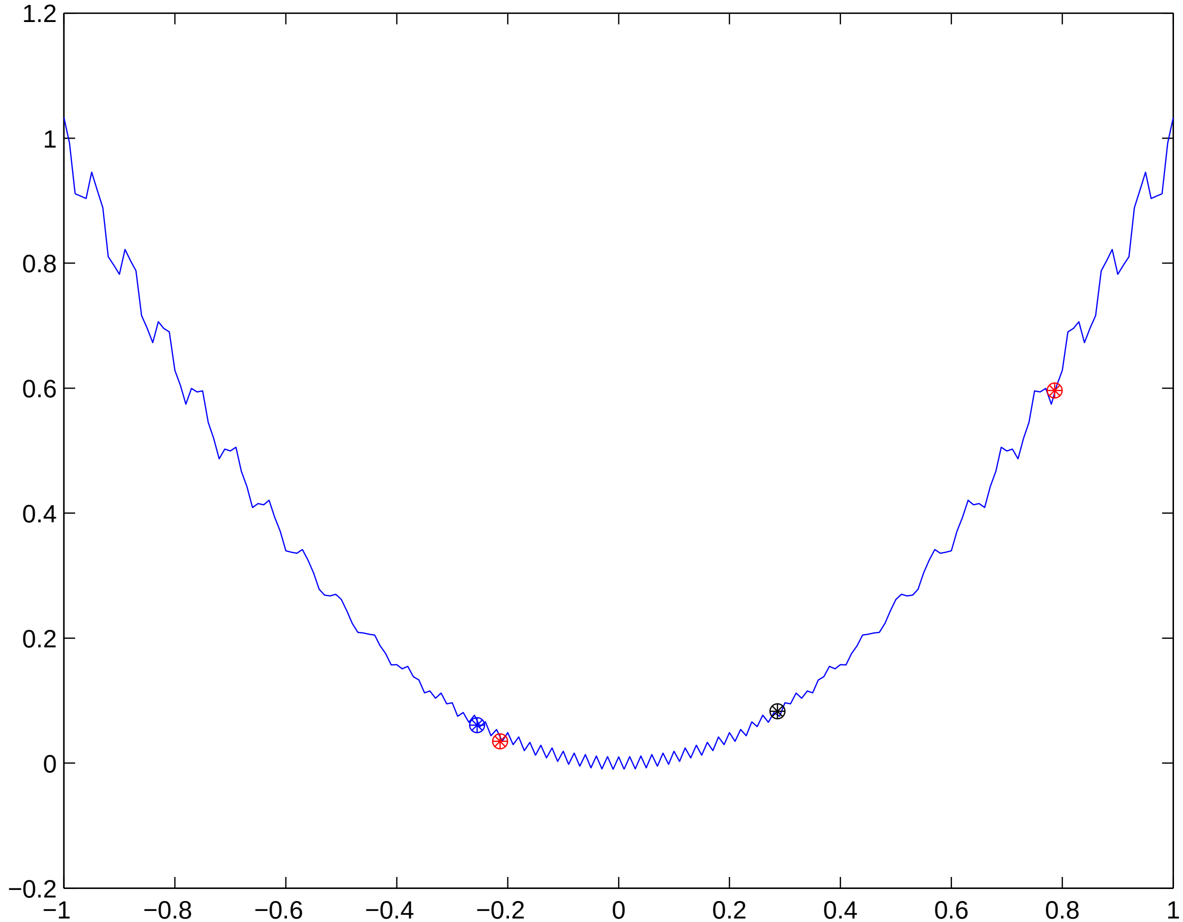
Implicit Filtering: Start



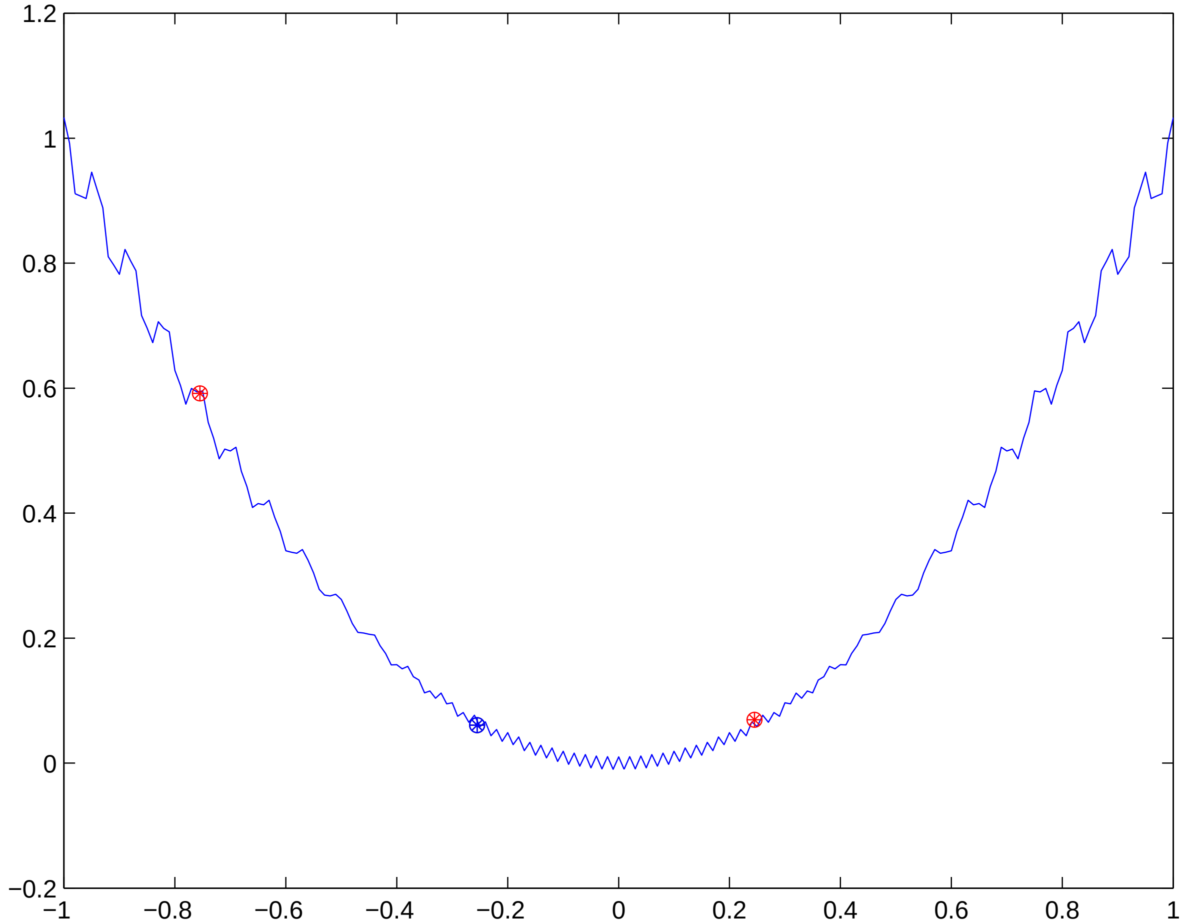
Implicit Filtering: Move



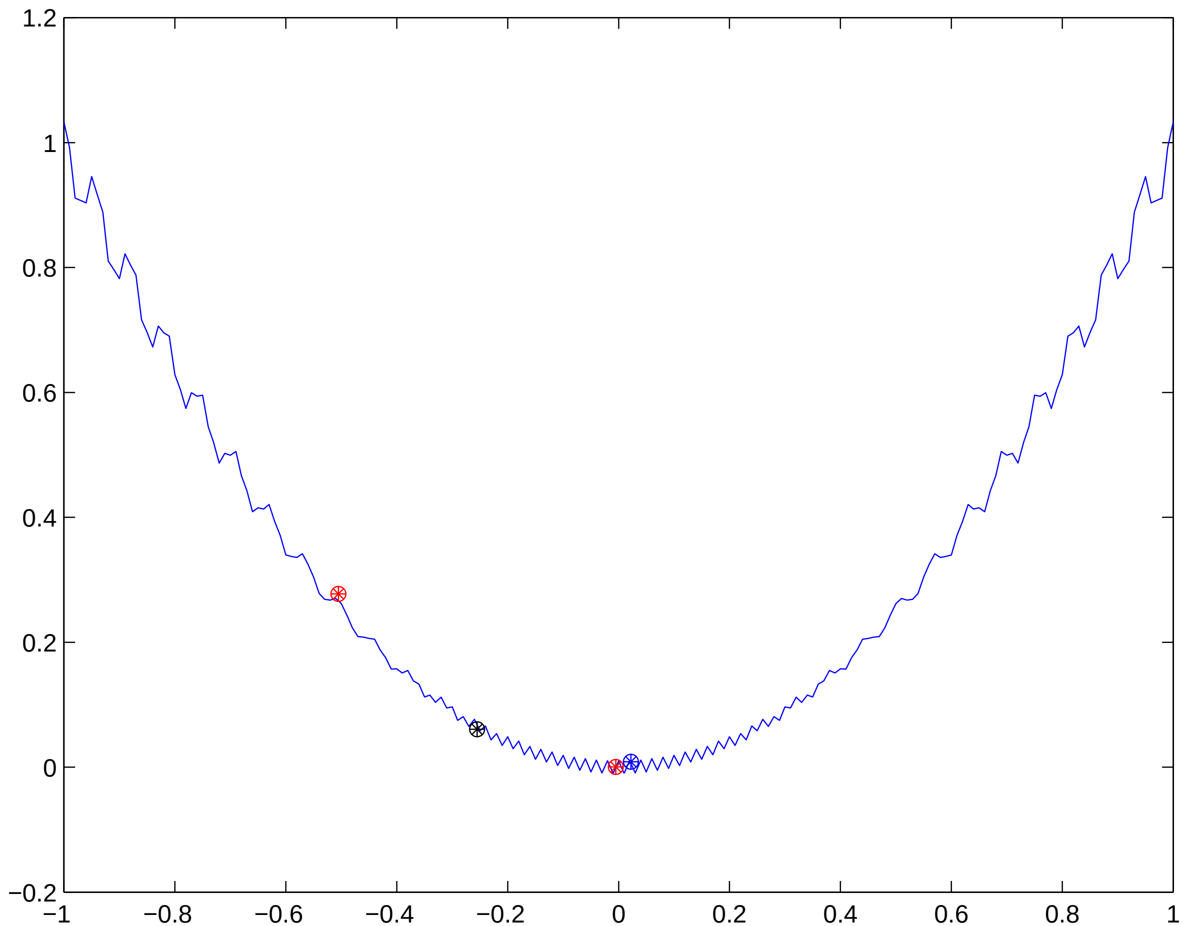
Implicit Filtering: Move



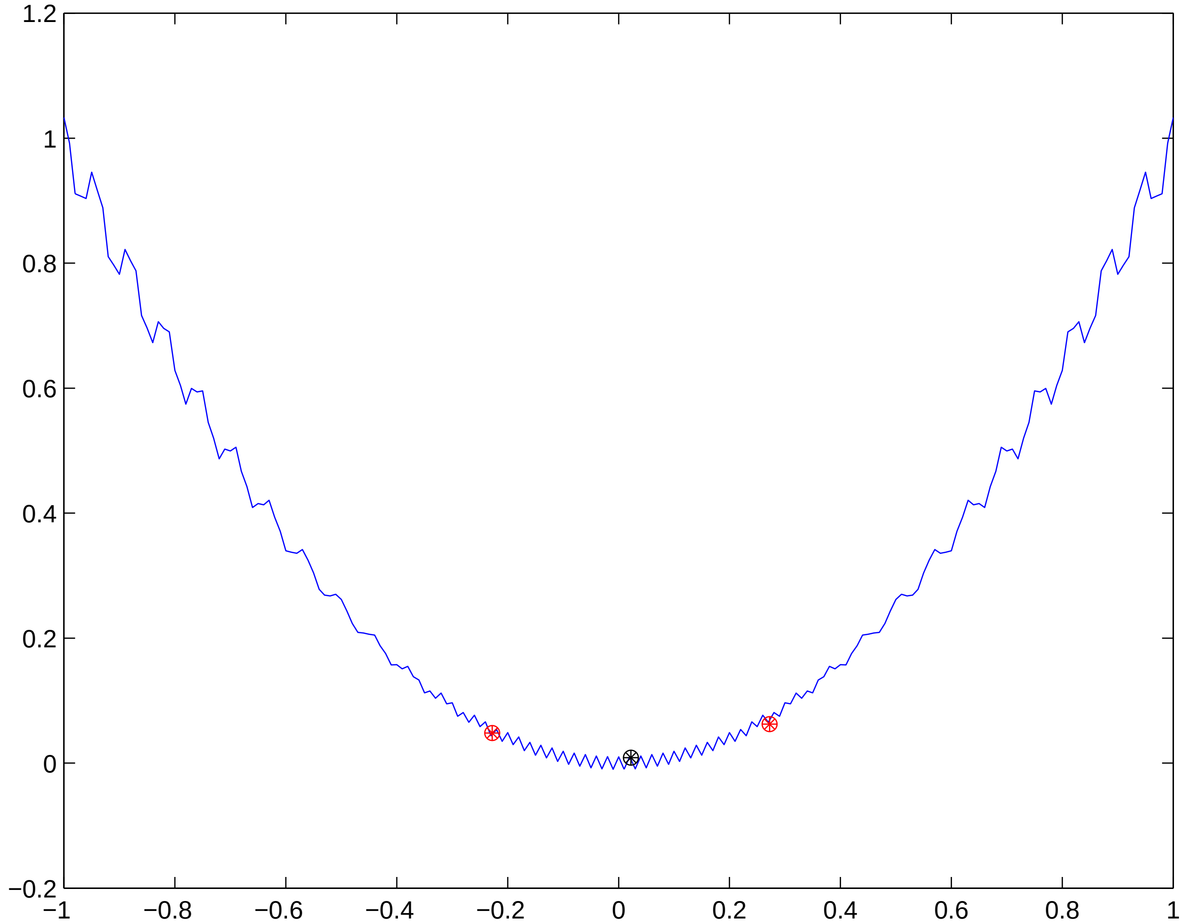
Implicit Filtering: Stencil Failure



Implicit Filtering: Shrink/Move



Implicit Filtering: Termination



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.
Convergence rates need more.

Hidden Constraints

A hidden constraint is violated if the call to f fails.
One can (we do) treat all but bound constraints as hidden.
What to do?

- Assign a large value.
- Assign a value of infinity and reject the sample. OK for HJ/MDS, bad for IF.
- Assign a value a bit higher than the nearby points.
- Reject and use least squares to compute $\nabla_h f$.
  - Coming in new version.

NCSU fortran implementation: IFFCO

- Naturally parallel; but watch out for load balancing.
- Use best value in stencil + quasi-Newton search.
- Quasi-Newton model Hessian essential in practice.
- Termination
  - fdquasi: stencil failure, small gradient, amax, pmax
  - overall: list of scales, budget, target
- parameters: IFFCO has reasonable defaults
- hidden constraints: f does not return a value
  IFFCO is prepared
- MATLAB: imfil.m new version coming soon

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

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, several linux clusters
- Three problems in place (only MODFLOW).
- New problems under construction.
- Massive comparison in progress
  GA, NOMAD, Boeing DE, DIRECT, APPS

Conclusions

- Optimal design of groundwater remediation problems
  - Formulation: constraints
    specification of problem
    choice of simulators
  - Community problems
  - Solution: we like sampling methods
- Sampling methods
  - Variants of coordinate search
  - Implicit filtering