BFO: a simple “brute-force” optimizer and its self optimization

Philippe Toint

Department of Mathematics, University of Namur, Belgium

(philippe.toint@fundp.ac.be)

EWMINLP 2010, Marseille, April 2010

The problem

We consider the unconstrained nonlinear programming problem:

$$\text{minimize} \quad f(x)$$

for $x \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$.

Important special case: the nonlinear least-squares problem

Additional constraints:

- there may be bounds on the variables
- derivatives are unavailable
- objective function possibly nonsmooth
- some variables can only assume discrete values

Motivation: parameter tuning in algorithm design (Audet-Orban), but many other examples. . .

Direct methods for optimization

A long history of direct search methods, including:

- Hookes-Jeeves, Nelder-Mead
- Coope-Price
- Dennis, Torczon, Lewis, Trosset (GPS)
- Dennis, Audet, Abramson (MADS).

This is one more of them

A very simple version:

BFO, a Brute-Force Optimizer

BFO basics

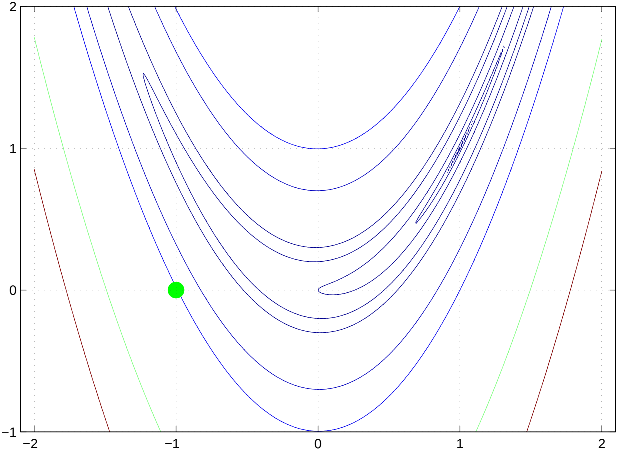
Main features: (not really original)

- Minimizes on progressively finer grids
- Some grid spacing remains fixed to user-specified discrete values
- Only for Local minimization. . .

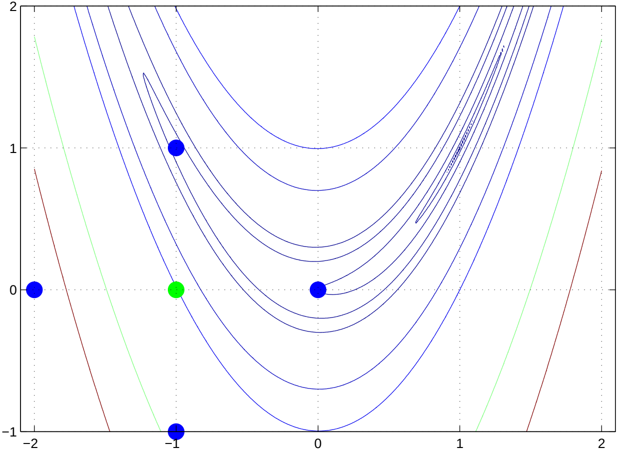
But also

- Attempts to learn from convergence history
- Includes some elements of random search
- User-friendly interface

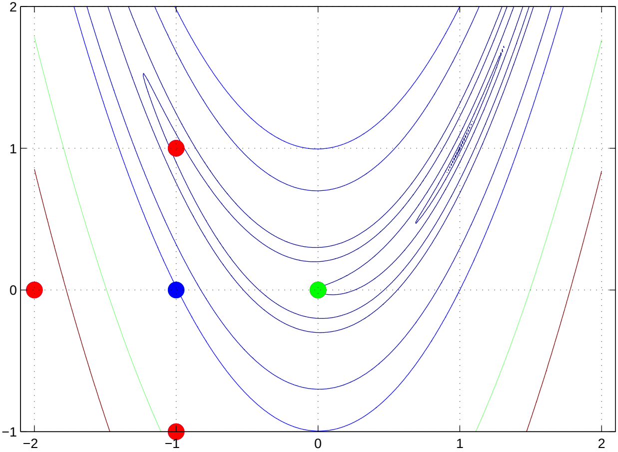
The compass-search on Rosenbrock's function



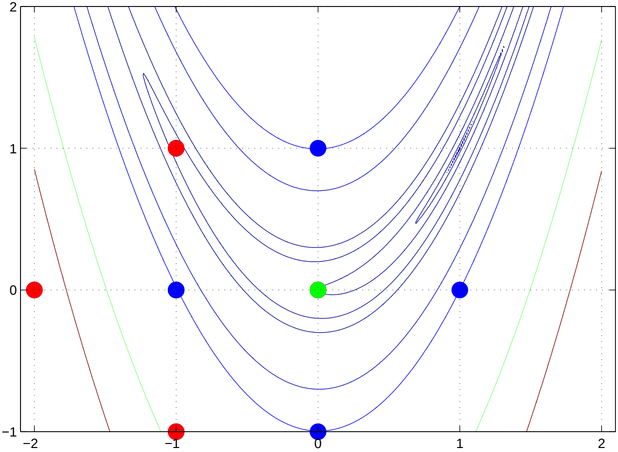
The compass-search on Rosenbrock's function



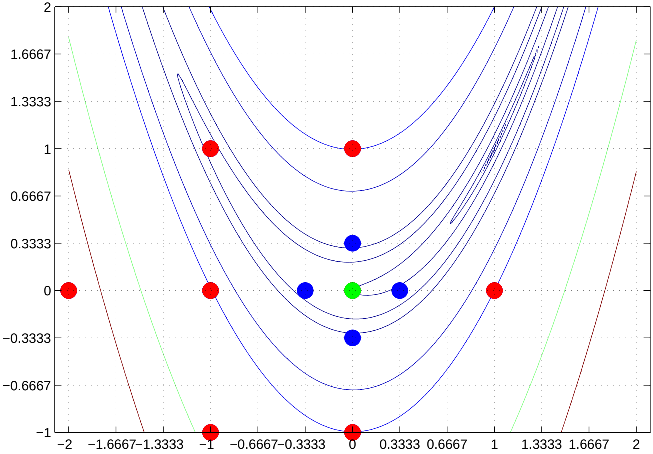
The compass-search on Rosenbrock's function



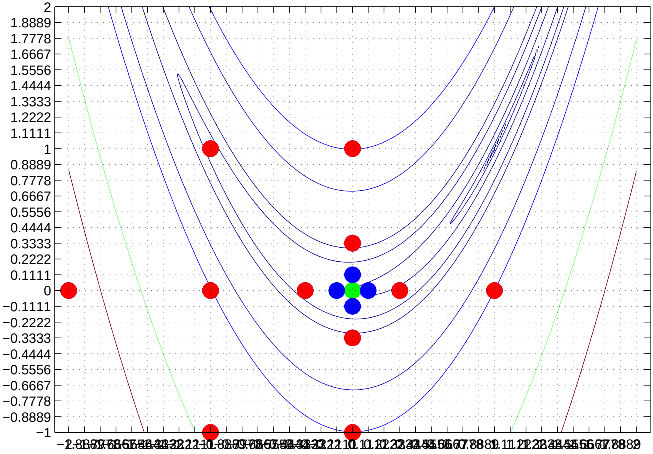
The compass-search on Rosenbrock's function



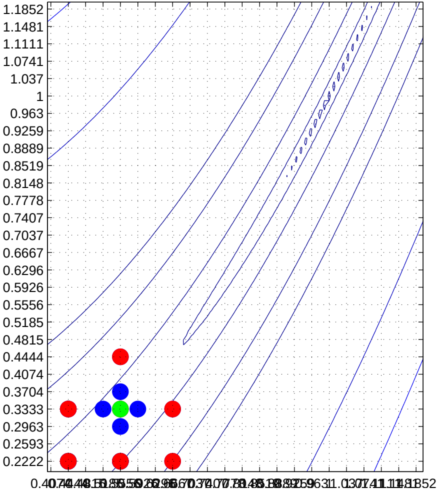
The compass-search on Rosenbrock's function



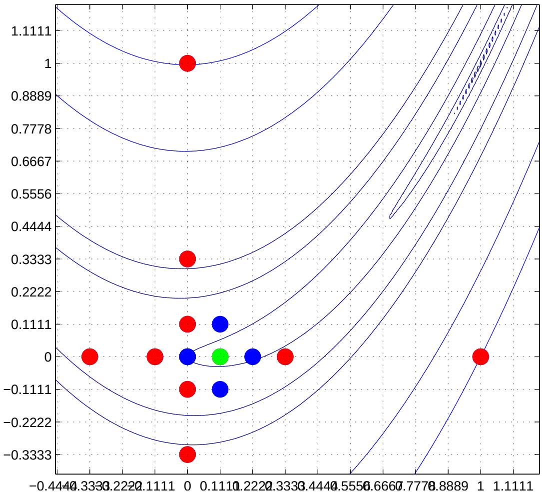
The compass-search on Rosenbrock's function



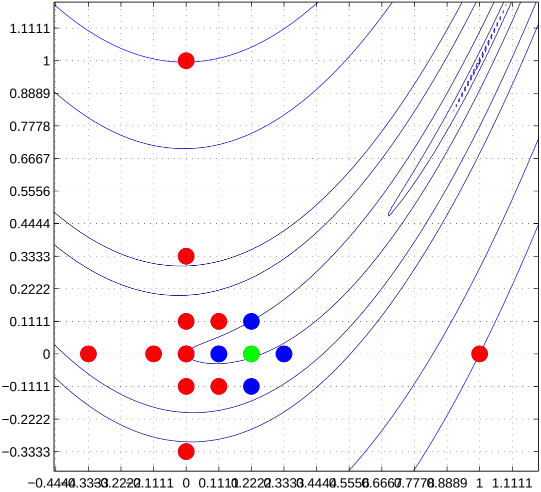
The compass-search on Rosenbrock's function



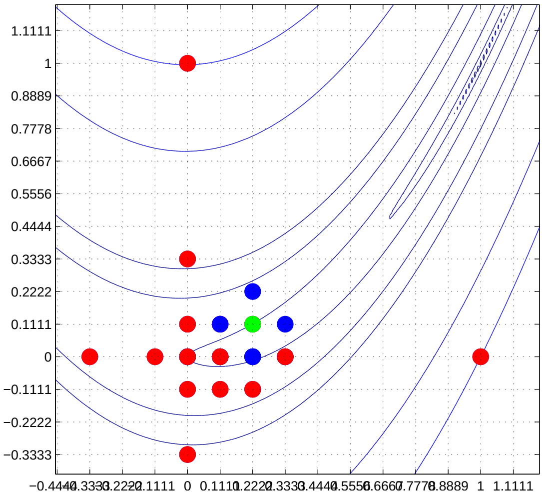
The compass-search on Rosenbrock's function



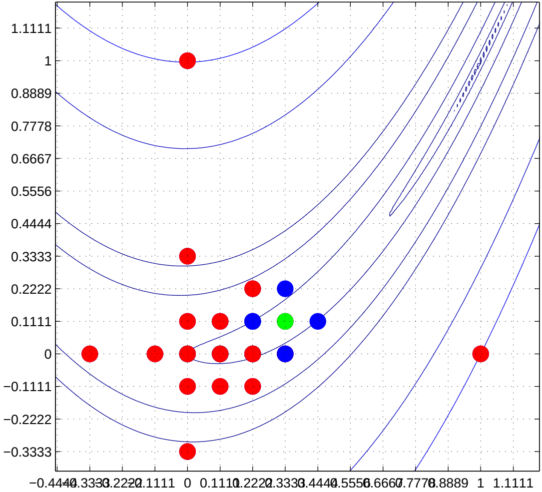
The compass-search on Rosenbrock's function



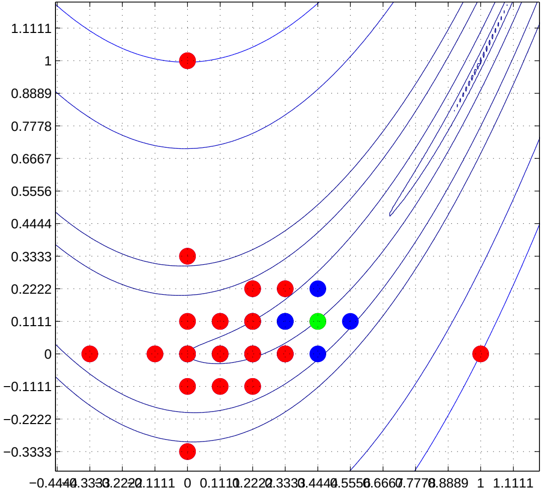
The compass-search on Rosenbrock's function



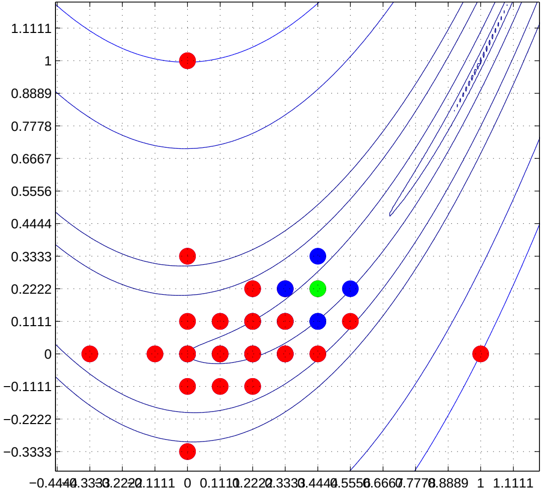
The compass-search on Rosenbrock's function



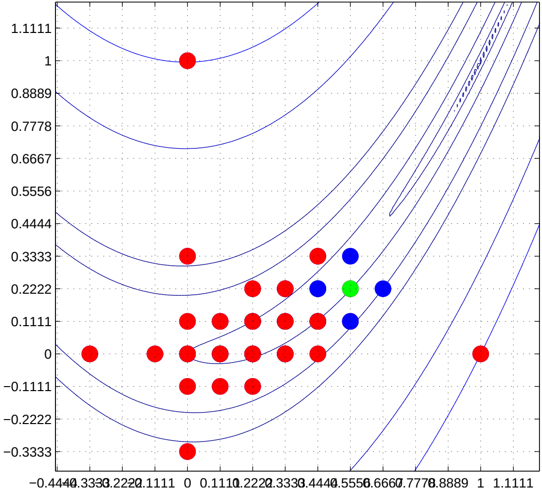
The compass-search on Rosenbrock's function



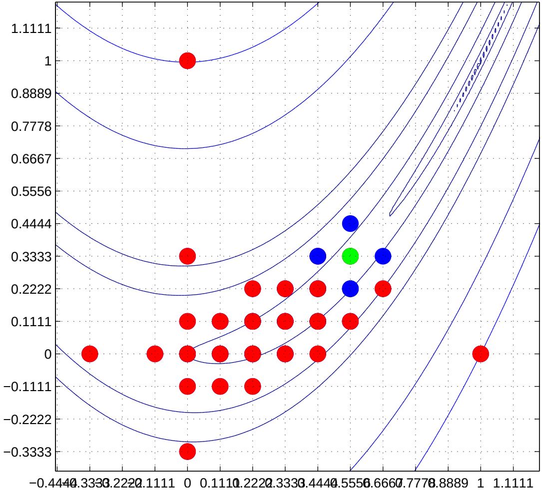
The compass-search on Rosenbrock's function



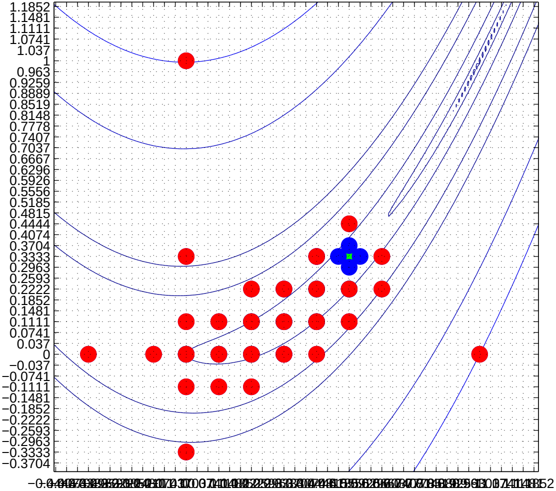
The compass-search on Rosenbrock's function



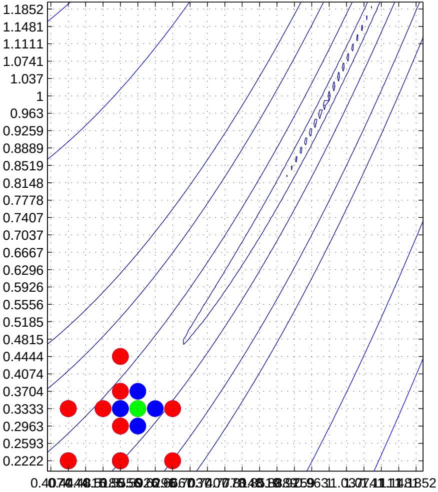
The compass-search on Rosenbrock's function



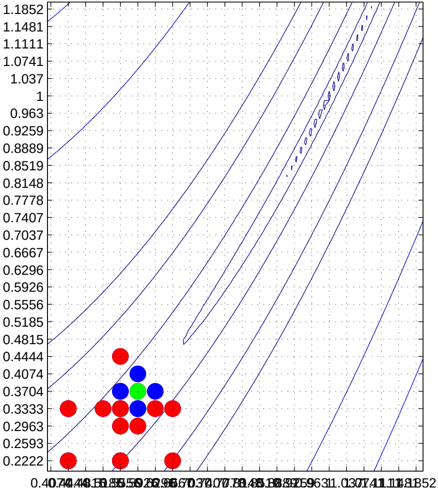
The compass-search on Rosenbrock's function



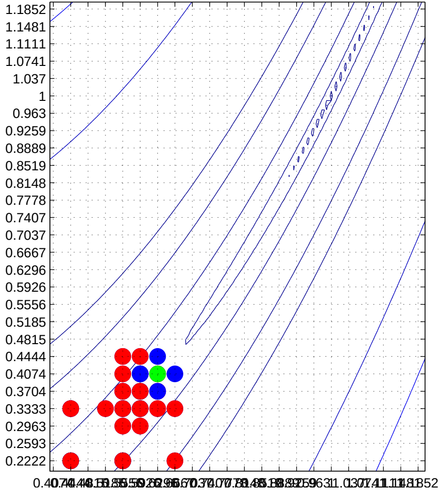
The compass-search on Rosenbrock's function



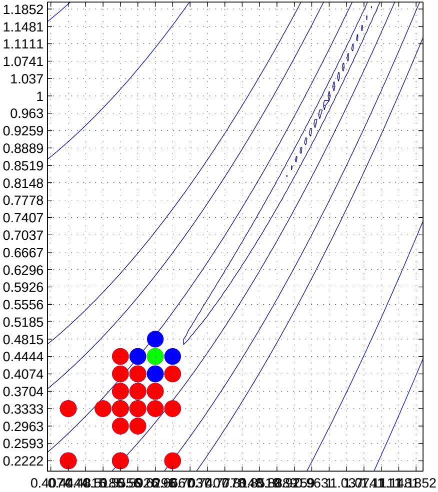
The compass-search on Rosenbrock's function



The compass-search on Rosenbrock's function



The compass-search on Rosenbrock's function



Weaknesses and algorithmic “solutions” (1)

- All compass points evaluated at each iteration
  - stop evaluating as soon as sufficient reduction is obtained
  - (suitable ordering of search directions)
- Painful zigzag crawling when axis not suitable
  - align axis on a grid with progress direction on previous grid (+ random complement)
  - define progress on a user-defined number of past iterations (inertia)
  - (independent scale for each variable)

Weaknesses and algorithmic "solutions" (2)

- Compute function at infeasible points
  - keep track of active and nearly-active bound constraints
- Slow progress on fine grids
  - expand the (continuous) grid on "successful iterations"
- Minima in neighbouring discrete subspaces not close to each other
  - recursively explore a local tree of discrete subspaces
  - keep track of record value in each such subspace to avoid re-exploration

Additional package features

Implementation features:

- check-pointing at user-specified frequency
- norms of vector functions t(f(x))
- allows objective functions with user-defined parameters
- allows partial objective computations
- allows randomized termination test
- provides a FD estimate of the (projected) gradient's norm (in the continuous variables) at termination
- very flexible keyword-based calling sequence

BFO: the Brute-Force Optimizer

User may specify (amongst others):

- grid shrinking/expansion factors
- inertia for defining progress directions
- initial scale in continuous variables
- local tree-search strategy (depth-first vs breadth-first)
- max conditioning for sampling gradient descent

A first setting for algorithm tuning

How best choose the algorithmic parameters?

- a set of 64 (uncs) + 64 (box) + 47 (mixed) test problems
- define objective function value

$$\phi_{BFO}(\text{params}) = \text{total number of evaluations to solve all problems}$$

- choose (reasonable) initial default values
- simple idea: let BFO optimize for finding better values!
  (problem in 6 continuous and 2 discrete variables)

(D. Orban and Ch. Audet)

A robust approach (1)

DANGER: overfitting. . .

Investigate other formulations inspired by robust optimization (in continuous variables)

(Idea also considered by A. Conn?)

- local sample $\mathcal{S}(\text{params})$:
  Consider varying each continuous algorithmic parameter by

  $$-5\% \qquad -1\% \qquad 0\% \qquad +1\% \qquad +5\%$$

  around each tested value (percentage tunable)
- local box

$$\mathcal{B}(\text{params}) = \prod_{1}^{\#\text{params}} [0.95\,\text{params}_i, 1.05\,\text{params}_i]$$

A robust approach (2)

Three "robust" formulations:

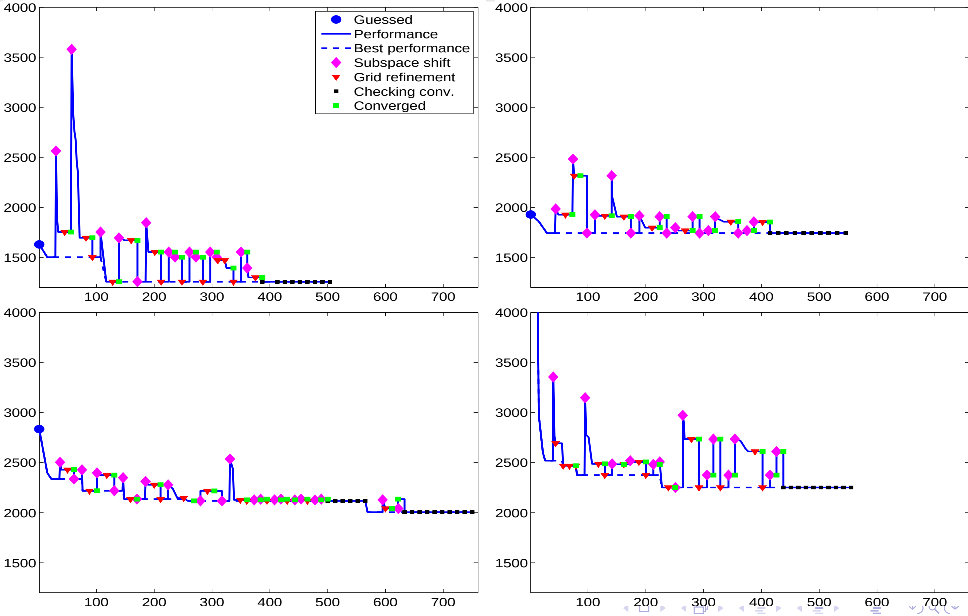
$$\min_{\text{params}} \sum_{\text{pars}\in\mathcal{S}(\text{params})} \phi_{BFO}(\text{pars})$$

$$\min_{\text{params}} \max_{\text{pars}\in\mathcal{S}(\text{params})} \phi_{BFO}(\text{pars})$$

$$\min_{\text{params}} \max_{\text{pars}\in\mathcal{B}(\text{params})} \phi_{BFO}(\text{pars})$$

- progressively (and considerably) more expansive
- BFO may be used for both optimization levels (form. 2 & 3)
- maximization problem possibly approximate (moderately coarse stopping rule)

On 3 test problems only: the optimization history



On 3 test problems only: the optimal parameters

	shrink	decr.	max exp.	in.	scale	exp.	condlim
guess	0.1	0.00001	3	5	1	2.1	1.2
simple	0.05	0.56469	3	6	1	2.1	1.147526
sum	0.1	0.00001	3.200578	5	1.25	2.114963	1.0734592
max ($\mathcal{S}$)	0.05	0.00001	4.0626793	9	1.1787	1.99772	1.4533078
max ($\mathcal{B}$)	0.1	0.000001	3.258	6	1.25	2.1	1.2

Which is the most sensible?

Maybe max ($\mathcal{B}$) ?

More computations are on the way. . .

Conclusions

BFO

- yet another direct method for mixed integer local optimization
- further direct improvements under development (ordering, sample gradients, etc)
- simple compact implementation
- freely-available easy-to-use Matlab package
- useable for algorithm tuning
- more to do (constraints, use of structure, . . .)

Optimization of algorithmic parameters

- four formulations proposed and (nearly) tested
- fully robust formulation most reasonable?

Thank you for your attention!

A new image reconstruction technique using the Linear Sampling Method in inverse-scattering problems

Philippe Toint (with M. Fares and S. Gratton)

Department of Mathematics, University of Namur, Belgium

(philippe.toint@fundp.ac.be)

EWMINLP 2010, Marseille, April 2010

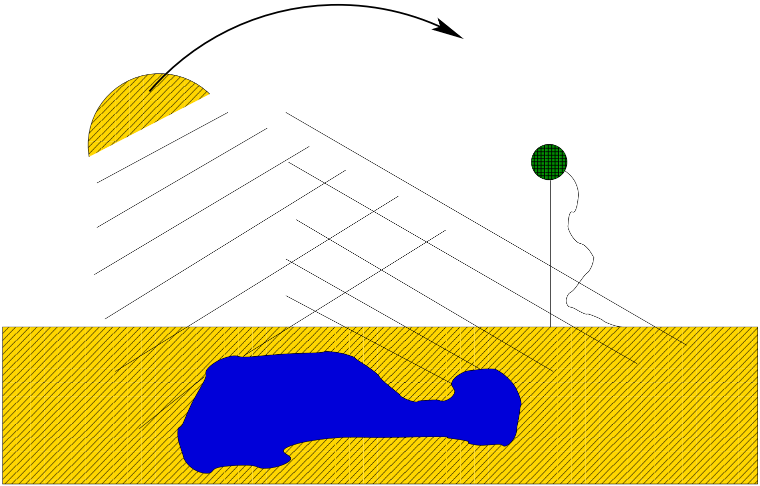
The problem (1)

The name of the game:

Discover the shape of hidden objects

- objects typically hidden in another medium (ground, water, body, . . .)
- illuminate the objects by plane waves from “”all” directions
- measure waves scattered (reflected) by the object in “all” directions
- reconstruct the object

The problem (2)



The Linear Sampling Method (1)

The direct problem:

- the hidden object $\mathcal{O}$ is given
- the incident plane wave

$$u^{inc}(x,d) = e^{ik\langle x,d\rangle} \qquad d \in \mathcal{S}$$

(k is the wave number)

- the PDE model: find $u^s(x,d)$ (the scattered wave) such that

$$\Delta u^s + k^2 u^s = 0 \quad \text{(outside } \mathcal{O}\text{)}$$

with

$$u = u^{ins} + u^s, \qquad u = 0 \text{ on } \partial\mathcal{O}$$

and

$$\lim_{r\to\infty} \frac{\partial u^s}{\partial r} - iku^s = 0.$$

The Linear Sampling Method (2)

The inverse problem problem:

- the hidden object $\mathcal{O}$ is given
- one knows the far-field u^∞ such that

$$u(x,d) = \left[\frac{e^{ik\langle x,d\rangle}}{x} u^\infty\left(\frac{x}{\|x\|}, d\right) + O\left(\frac{1}{x}\right)\right] \quad \text{when} \quad \|x\| \to \infty$$

- assuming the same PDE model, find $\delta\mathcal{O}$.

A 0-1 formulation:

Given u^∞, decide, for each $x \in \mathbb{R}^3$, whether $x \in \mathcal{O}$ or not.

The Linear Sampling Method (3)

For $\hat{x} = x/\|x\|$, define the far-field operator and equation

$$(\mathcal{F}g)(\hat{x}) \stackrel{\text{def}}{=} \int_{\mathcal{S}} u^{\infty}(\hat{x}, d) g(d) \, ds(d) = \frac{1}{4\pi} e^{-ik\langle \hat{x}, d\rangle}$$

(the far-field pattern associated to the plane wave $e^{ik\langle \hat{x}, d\rangle}$)

Finding g is ill-posed!

But (Colton) . . .

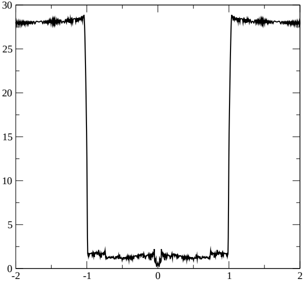
$$\forall \epsilon \quad \exists g_{\epsilon} \quad \|\mathcal{F} g_{\epsilon}(\cdot, d) - \tfrac{1}{4\pi} e^{-ik\langle \cdot, d\rangle}\| \leq \epsilon$$

The Linear Sampling Method (4)

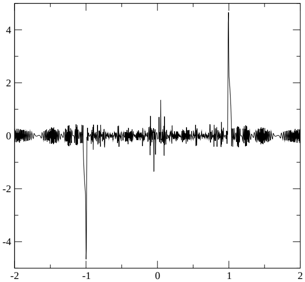
The crucial property:

$$\lim_{z\in\mathcal{O},z\to\partial\mathcal{O}} \|g_\epsilon(\cdot,z)\| = \infty, \quad \text{and} \quad \|g_\epsilon(\cdot,z)\| = \infty \quad \text{for} \quad (z \in \mathbb{R}^3 \setminus \mathcal{O})$$

For $\mathcal{O}$ a sphere: compute the shape of $\mathcal{O}$ from the level curves of $\|g_\epsilon\|$



$\log_{10}\|g_\epsilon\|$



$\log_{10}\|\nabla g_\epsilon\|$

The Linear Sampling Method (5)

Approximate the far-field equation

$$\int_{\mathcal{S}} u^{\infty}(\hat{x}, d) g(d)\, ds(d) = \frac{1}{4\pi} e^{-ik\langle \hat{x}, d\rangle}$$

by its discretized form

$$\sum_{j=1}^{N} F_{\ell,j} g_j(z) = e^{-ik\langle d_\ell, z\rangle}$$

i.e.

$$Fg(z) = b^{\infty}(z) \qquad (z \in \mathcal{B} \subset \mathbb{R}^3)$$

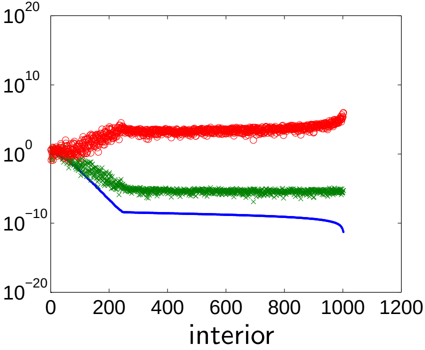
This is an underdetemined system! $\Rightarrow$ regularization (ignore nullspace)

Regularizing the discretized far-field equation

Assume $F = U\Sigma V^*$. Well-known regularization techniques (Tikhonov-Morosov, TSVD, L-curve, GCV, ...) applicable if the Picard coefficients

$$\left\{ \frac{u_\ell^* b^\infty(z)}{\sigma_\ell} \right\} \quad \text{decrease to zero.}$$

However,



Operator perturbation and clustered eigenspace recovery

Use a different approach! Consider

$$A = Q^* \Lambda Q \quad \text{and} \quad \bar{A} = A + tE = \bar{Q}^* \bar{\Lambda} \bar{Q}$$

Choose $b = \sum_{j=1}^{n} \alpha_j q_j$. Then, after some analysis and for t small,

$$\bar{q}_\ell^* b \approx \alpha_\ell + t \sum_{j \neq \ell} \alpha_j q_\ell^* E^* P_\ell (\lambda_\ell I_{n-1} - \Lambda_\ell)^{-1} e_{j\ell}$$

where $P_\ell = Q$ minus its ℓ-th column.

$|\bar{q}_\ell^* b|$ large whenever t is small, ℓ is the index of a clustered eigenvalue and b has a significant component along the cluster's eigenvectors

Note: A, Q or Λ may be unknown!

Application to Linear Sampling

Choose

$$\bar{A} = FF^*, \quad \text{and} \quad tE = FF^* - F^\infty (F^\infty)^*$$

and thus

$$\bar{Q} = U \quad \text{and} \quad |\bar{q}_\ell^* b| = u_\ell^* b^\infty.$$

$|u_\ell^* b^\infty|$ large whenever ℓ is large, $\|FF^* - F^\infty(F^\infty)^*\|$ is small and b^∞ has a significant component along the (unknown) approximate nullspace of the far-field operator

(as observed)

reconstruct nullspace!

The SVD-tail algorithm

1. Select $\mathcal{T}$ a d-dimensional subspace (approximately) spanned by the the d leftmost singular vectors of F
2. Compute $\{w_\ell\}$ a basis of $\mathcal{T}$ and $\vartheta_\ell(z) = w_\ell^* b^\infty(z) \quad (\ell = 1, \ldots, d)$
3. Define

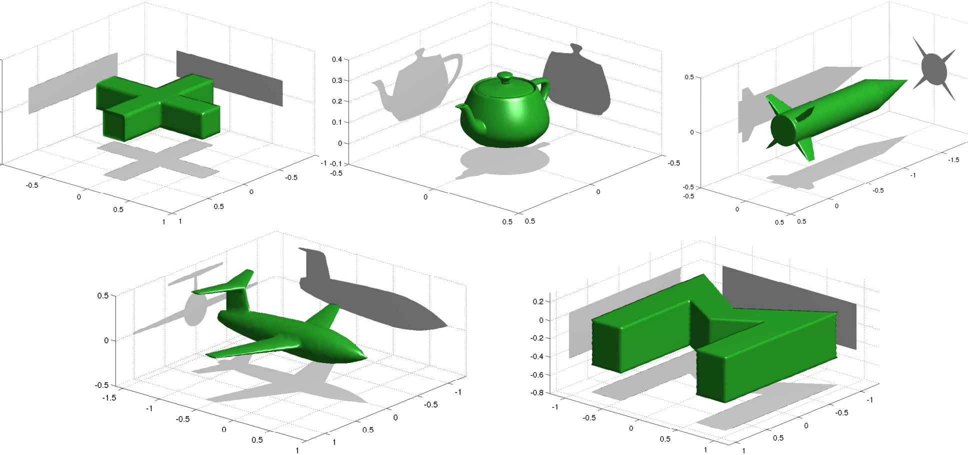
$$\psi_d(z) = \frac{1}{\|\vartheta(z)\|}$$

$\psi_d(z)$ small for z outside $\mathcal{O}$ and large for z inside $\mathcal{O}$

Same as $g_\epsilon(z)$ but MUCH cheaper to evaluate (no full SVD)!

Use level curves of ψ_d to compute $\partial\mathcal{O}$!

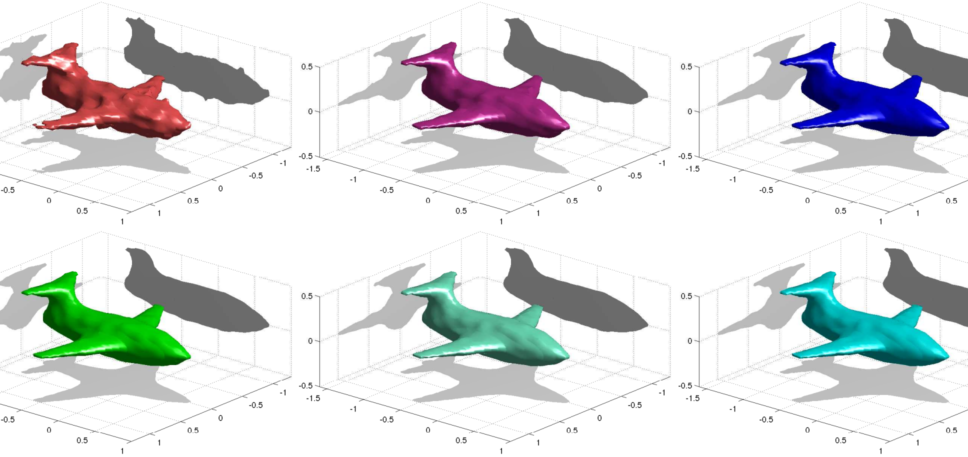
(isovalue heuristic)



-0.5 0 0.5 1
1 0.5 0 -0.5
0.4 0.3 0.2 0.1 0 -0.1
-1 -0.5 0 0.5
0.5 0 -0.5
0.5 0 -0.5 -1 -1.5
0.5 0 -0.5
-1.5 -1 -0.5 0 0.5 1
1 0.5 0 -0.5 -1
0.2 0 -0.2 -0.4 -0.6 -0.8
-1 -0.5 0 0.5 1
1 0.5 0 -0.5 -1

Illustration: the reconstruction of the plane

$N = 1002, d = 5, 20, 35, 50, 65, 80$



Conclusions

- Interesting (to me) 0-1 decision problem in 3D-space
- Efficient computaional scheme
- Uses level curves of a nullspace "indicator"
- Can this be extended to other such problems ?

Thank you for your attention!