Optimization principles for discrete choice theory

Fabian Bastin
Department of Computing Science
and Operational Research
Université de Montréal
Quebec, Canada
http://www.iro.umontreal.ca/~bastin

April 29th, 2008

Motivation

We consider the problem

$$\min_{x \in \mathbb{R}^n} f(x).$$

We assume in this tutorial that $f : \mathbb{R}^n \to \mathbb{R} \in C^2$ (i.e. f is twice continuously differentiable), and we consider only minimization since maximization can be achieved by minimizing the opposite :

$$\max_{x \in \mathbb{R}^n} = - \min_{x \in \mathbb{R}^n} f(x).$$

Example : maximum log-likelihood.

$$\max_{\theta} \mathcal{LL}(\theta) = \frac{1}{I} \sum_{I=1}^{I} \log f_\theta(x_i|\theta).$$

Mixed logit models estimation

Probability choice of A_j by individual i :

$$P_{ij}(\theta) = E_{\gamma}\left[L_{ij}(\gamma,\theta)\right] = \int L_{ij}(\gamma,\theta) f(\gamma) d\gamma.$$

Since the choice probabilities are now expectations, they have to be estimated.

Analogy to stochastic programming : sample average approximation (SAA). We sample over the random parameters, and maximize the approximate log-likelihood that we obtain :

$$\max_{\theta} SLL(\theta) = \max_{\theta} \frac{1}{I} \sum_{i=1}^{I} \ln SP^{R}_{ij_i}.$$

The resulting problem is nonlinear nonconvex, but unconstrained.

Some notations...

Recall that the gradient vector and the Hessian are

$$\nabla_x f(x) = \left(\frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2} \quad \cdots \quad \frac{\partial f}{\partial x_n} \right)^T,$$

$$\nabla^2_{xx} f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial^2 x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial^2 x_n} \end{pmatrix}$$

Ideally, we would like to find a global minimizer.

Definition (Global minimizer)

Let $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$. A point x^* is a global minimizer of $f(\cdot)$ on X if

$$f(x^*) \leq f(x) \qquad \forall\, x \in X.$$

Local minimization

In practice, we can only find local minimizers.

Definition (Local minimizer)

Let $f : X \to \mathbb{R}$. A point $x^* \in X$ is a local minimizer of $f(\cdot)$ on X if there exists a neighborhood $B_\epsilon(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| < \epsilon\}$ such that

$$f(x^*) \leq f(x) \qquad \forall\, x \in B_\epsilon(x^*).$$

Lemma

Let $X \subset \mathbb{R}^n$, $f \in C^1$ on X, and $x \in X$. If $d \in \mathbb{R}^n$ is a feasible direction at x, i.e. there exists some scalar $\lambda(x, d) > 0$ such that

$$(x + \lambda d) \in X, \ \forall \lambda \in [0, \lambda(x, d)],$$

and $\nabla_x f(x)^T d < 0$, then there exists a scalar $\delta > 0$ such that for all $0 < \tau \leq \delta$, $f(x + \tau d) < f(x)$.

In such a case, we say that d is a descent direction at x.

Steepest descent

If x in the interior of X, we see that a particular descent direction if $-\nabla_x f(x)$, as long as $\nabla_x f(x) \neq 0$. Consider the linear expansion of f around x :

$$f(x+d) \approx f(x_0) + \nabla_x f(x)^T d,$$

for "small" d, i.e. if $\|d\|$ is small. If the approximation is good, it seems reasonable to try to make $\nabla f(x)^T d$ as small as possible (i.e. as negative as possible). From the Cauchy-Schwartz inequality, we have

$$\nabla_x f(x)^T d \geq -\|\nabla_x f(x)\| \|d\|,$$

and the equality is achieved if $d = -\alpha \nabla_x f(x)$, with $\alpha > 0$.

The direction $-\nabla_x f(x)$ is called the steepest descent direction of f at x. If $\nabla_x f(x) = 0$, there is no descent direction !

First-order criticality

From the previous observations, we also have the following result.

Theorem (Necessary condition)

Let $X \subset \mathbb{R}^n$ an open set, $f \in C^1$ on X. Is x is a local minimizer of f on X, then $\nabla_x f(x) = 0$.

From the previous theorem, we introduce the definition of first-order criticality.

Definition (First-order criticality)

A point x^* is said to be be first-order critical for $f : \mathbb{R}^n \to \mathbb{R}$ if $\nabla_x f(x^*) = 0$.

First-order criticality (2)

A first-order critical point is however not always a local minimizer. Consider for instance the function $f(x) = x^3$. Then $f'(0) = 0$, but it is clear that 0 is not a local minimizer for $f(\cdot)$.

Special case : convex functions. If f is convex, a local minimizer is also a global minimizer, and so a first-order critical point is a global minimizer.

For nonconvex functions, we need more, and so we introduce additional definitions.

Definition

Let D be a real matrix of dimension $n \times n$. The associated quadratic form is the function $\psi : \mathbb{R}^n \to \mathbb{R}$ defined as

$$\psi(x) = x^T D x.$$

Preliminary steps towards second-order criticality

Definition

Let D be a real matrix of dimension $n \times n$. D is said to be positive semi-definite (definite) if $\psi(x) \geq 0$, for all $x \in \mathbb{R}^n$ ($\psi(x) > 0$, for all $x \in \mathbb{R}^n$, $x \neq 0$).

Definition

Let A be a matrix of dimension $n \times n$. λ is an eigenvalue of A if there exists some vector $u \in \mathbb{R}_0^n$ such that $Au = \lambda u$. u is the eigenvector of A associated to λ.

Theorem

A real symmetric matrix D of dimension $n \times n$ is positive semi-definite (definite) iff all its eigenvalues are non-negative (strictly positive).

Second-order criticality conditions

Theorem (Second-order necessary condition)

Let $X \subset \mathbb{R}^n$ an open set and $f \in C^2$ on X. If x^* is a local minimizer of f on X, then $\nabla f(x) = 0$ and $\nabla^2_{xx} f(x)$ is positive semi-definite.

This condition is not sufficient ! Consider for instance

$$f(x, y) = x^3 + y^3.$$

Then

$$\nabla f(x, y) = \begin{pmatrix} 3x^2 \\ 3y^2 \end{pmatrix}, \qquad \nabla^2 f(x, y) = \begin{pmatrix} 6x & 0 \\ 0 & 6x \end{pmatrix}.$$

The vector $(0, 0)^T$ is not a (local) minimizer, while it satisfies the necessary conditions.

Second-order criticality sufficient conditions

The positive definition condition however allows us to set up a sufficient condition.

Theorem

Let $X \subset \mathbb{R}^n$ an open set and $f \in C^2$ on X. If $\nabla_x f(x^*) = 0$ and $\nabla^2_{xx} f(x^*)$ is a positive definite matrix, then there exists some $\epsilon > 0$ such that $f(x^*) < f(x)$ for all $x \in B_\epsilon(x^*) \setminus \{x^*\}$.

Corollary (Second-order sufficient condition)

Let $X \subset \mathbb{R}^n$ an open set and $f \in C^2$ on X. If $\nabla_x f(x^*) = 0$ and $\nabla^2_{xx} f(x^*)$ is a positive definite matrix, then x^* is a local minimizer of f on X.

Algorithms

Definition (Descent algorithm)

An algorithm A is a descent algorithm with respect to a continuous function $z : X \to \mathbb{R}$ is

1. if $x \notin X^*$ and $y \in A(x)$, then $z(y) < z(x)$,
2. if $x \in X^*$ and $y \in A(x)$, then $z(y) \leq z(x)$.

All the algorithms that we will consider are descent algorithms, at least locally, and are iterative. In other terms, they will construct a (possibly infinite) sequence of point x_n $(n = 0, \ldots)$ and we hope that this sequence converges to a minimizer of the objective function.

The question is therefore how to construct such a sequence ?

Steepest descent algorithm

The simplest algorithm is the steepest descent method.

Step 0. Given a starting point x_0, set $k = 0$.

Step 1. Set $d_k = -\nabla_x f(x_k)$. If $d_k = 0$, stop.

Step 2. Solve (possibly approximately) $\min_\alpha f(x_k + \alpha d_k)$ for the stepsize α_k.

Step 3. Set $x_k \leftarrow x_k + \alpha_k d_k$, $k \leftarrow k + 1$. Go to Step 1.

Theorem (Convergence of the steepest descent algorithm)

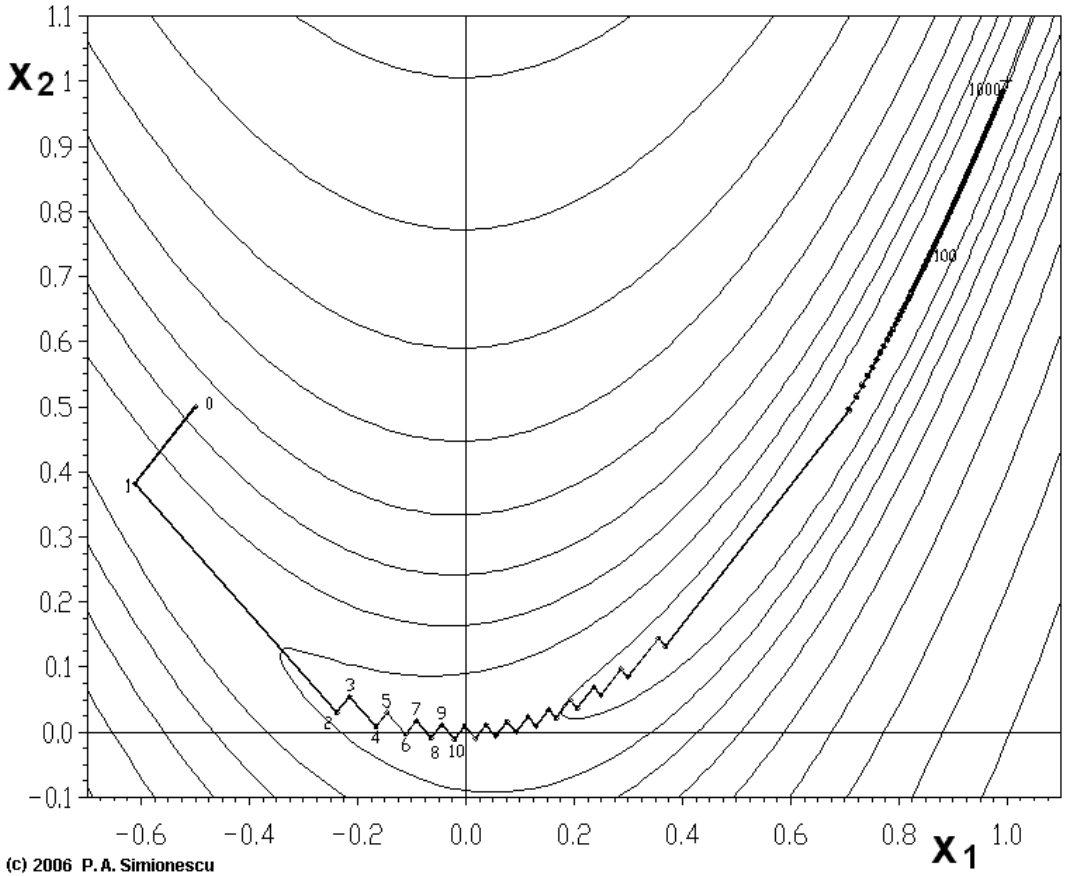
Suppose that $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on the set $S = \{x \in \mathbb{R}^n \,|\, f(x) \leq f(x_0)\}$, and that S is a closed and bounded set. then every point x^* that is a cluster point of the sequence $\{x_k\}$ satisfies $\nabla_x f(x^*) = 0$.

But proving convergence is not sufficient ! Is it efficient ?

The Rosenbrock function

Apply the steepest descent algorithm to the function

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$$



"Zig-zag" behavior. In other terms, the convergence is very slow ! Is it possible to improve it ?

Newton method : introduction

Can we converge faster ? If f is C^2, we can write the Taylor expansion of order 2 :

$$f(x+d) \approx f(x) + \nabla_x f(x)^T d + \frac{1}{2} d^T \nabla^2_{xx} f(x) d.$$

Note : from the Taylor expansion, we see that if $\frac{1}{2} d^T \nabla^2_{xx} f(x^*) d > 0$, x^* is a local minimizer iff $\nabla_x f(x^*) = 0$. Otherwise, we can choose d such that $\nabla_x f(x)^T d < 0$. Moreover, if $\nabla_x f(x^*) \neq 0$, we can neglect the term $\frac{1}{2} d^T \nabla^2_{xx} f(x^*) d$ for x sufficiently close to x^*, and again choose d such that $\nabla_x f(x^*)^T d < 0$.

This again justifies our definition of first-order criticality.

Newton method

At iteration k, define (around the current iterate x_k)

$$m_k(d) = f(x_k) + \nabla_x f(x_k)^T d + \frac{1}{2} d^T \nabla^2_{xx} f(x_k) d.$$

If $\nabla^2_{xx} f(x_k)$ is positive definite, $m_k(d)$ is convex in a neighborhood of x_k and we can minimize $m_k(d)$ by computing d such that $\nabla_d m_k(d) = 0$. In other terms, we compute d_k as

$$d_k = -[\nabla^2_{xx} f(x_k)]^{-1} \nabla_x f(x_k),$$

and we set

$$x_{k+1} = x_k + d_k.$$

If the starting point x^0 is close to the solution, the convergance is quadratic, but if the starting point is not good enough, the method can diverge !

Quasi-Newton method

It is often difficult to obtain an analytical expression for the derivatives, and they may be costly to compute at each iteration, so we prefer to turn to approximations.

First-order derivatives can be computed by finite differences :

$$\frac{\partial f}{\partial x_{[i]}} \approx \frac{f(x + \epsilon e_i) - f(x)}{\epsilon},$$

for small ϵ, and $e_i = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 & 0 & \dots 0 \end{pmatrix}^T$ is the ith canonical vector, or central differences :

$$\frac{\partial f}{\partial x_{[i]}} \approx \frac{f(x + \epsilon e_i) - f(x - \epsilon e_i)}{2\epsilon}.$$

This however requires $O(n)$ evaluations of the objective.

Hessian approximation : BFGS

We can approximate the Hessian with a similar approach, but the computation cost is then of $O(n^2)$, wich is usually too expensive.

An alternative is to construct an approximation of the Hessian that we will improve at each iteration, using the gained information. We then speak of quasi-Newton method.

A popular approximation is the BFGS (Broyden, Fletcher, Goldfarb and Shanno) :

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^t d_k} + \frac{B_k d_k (B_k d_k)^T}{d_k^T B_k d_k},$$

where $y_k = \nabla_x f(x_{k+1}) - \nabla_x f(x_k)$.

BHHH

Various alternatives to BFGS exist, as the Symmetric-Rank 1 (SR1) that is popular for nonconvex optimzation with trust-region. The BHHH, proposed by Berndt, Hall, Hall, Hausman in 1974, approximation is well-known in discrete choice estimation, and is derived from the information identity :

$$\hat{B}_k = \frac{1}{I} \sum_{i=1}^{I} \nabla_\theta \ln SP_{ij_i} \nabla_\theta \ln SP_{ij_i}^T .$$

BHHH exploits the structure of the problem, and only the information available at the current iteration is used.

The information identity however relies on the assumption is correctly specified, but for complex models, this assumption can only be partially satisfied, and is clearly violated during the model development process. In such cases, the convergence can be very slow.

Globalization of the Newton method

Global convergence : the algorithm must converge for any starting point. BUT it still converges to a local mimimum, not a global minimum !

So how to ensure global convergence ? Globalization of the Newton method :

- linesearch methods ;
- trust-region methods.

Line-search methods generate the iterates by setting

$$x_{k_1} = x_k + \alpha_k d_k .$$

where d_k is a search direction and $\alpha_K d_k$ is chosen so that

$$f(x_{k+1}) < f(x_k).$$

Linesearch methods are therefore descent methods, and a special case is the steepest descent method.

Most line-search versions of the basic Newton method generate the direction d_k by modifying the Hessian matrix $\nabla_{xx} f(x_k)$ to ensure that the quadratic model m_k of the function has a unique minimizer.

The modified Cholesky decomposition approach adds positive quantities to the diagonal of $\nabla_{xx} f(x_k)$ during the Cholesky factorization. As a result, a diagonal matrix, E_k, with nonnegative diagonal entries is generated such that

$$\nabla_{xx} f(x_k) + E_k$$

is positive definite.

Linesearch methods (2)

Given this decomposition, the search direction d_k is obtained by solving

$$(\nabla_{xx} f(x_k) + E_k) d_k = -\nabla_x f(x_k).$$

After d_k is found, a line-search procedure is used to choose an $\alpha_k > 0$ that approximately minimizes f along the ray

$$\{x_k + \alpha d_k \mid \alpha > 0\}.$$

The algorithms for determining α_k usually rely on quadratic or cubic interpolation of the univariate function

$$\phi(\alpha) = f(x_k + \alpha d_k)$$

in their search for a suitable α_k.

Determination of α_k

An elegant and practical criterion for a suitable α_k is to require α_k to satisfy the sufficient decrease condition :

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \mu \alpha_k d_k^T \nabla_x f(x_k).$$

and the curvature condition :

$$|d_k^T \nabla_x f(x_k + \alpha_k d_K)| \leq \eta |d_k^T \nabla_x f(x_k)|.$$

where μ and η are two constants with $0 < \mu < \eta < 1$. The sufficient decrease condition guarantees, in particular, that $f(x_{k+1}) < f(x_k)$, while the curvature condition requires that α_k be not too far from a minimizer of ϕ.

Requiring an accurate minimizer is generally wasteful of function and gradient evaluations, so codes typically use $\mu = 0.001$ and $\eta = 0.9$ in these conditions.

Trust region methods

Principe : at iteration k, approximately minimize a model m_k of the objective inside a region $\mathcal{B}_k$. A typical choice for m_k is

$$m_k(x_k + s) = f(x_k) + s^T \nabla f(x_k) + \frac{1}{2} s^T H_k s.$$

H_k : approximation of $\nabla^2_{xx} f(x_k)$.

We therefore have to solve the subproblem

$$\min_s m_k(x_k + s), \text{ such that } x_k + s \in \mathcal{B}_k.$$

The solution is the candidate iterate with candidate step s_k.

Computing the following ratio :

$$by\, \rho_k = \frac{f(x_k + s_k) - f(x_k)}{m_k(x_k + s_k) - m_k(x_k)}.$$

Trust region methods (2)

Let η_1 and η_2 be constants such that $0 < \eta_1 \leq \eta_2 < 1$ (for instance, $\eta_1 = 0.01$ and $\eta_2 = 0.75$).

- If $\rho_k \geq \eta_1$, accept the candidate.
- If $\rho_k \geq \eta_2$, enlarge $\mathcal{B}_k$, otherwise reduce it or keep it the same.
- If $\rho_k < \eta_1$, reject the candidate and reduce $\mathcal{B}_k$.

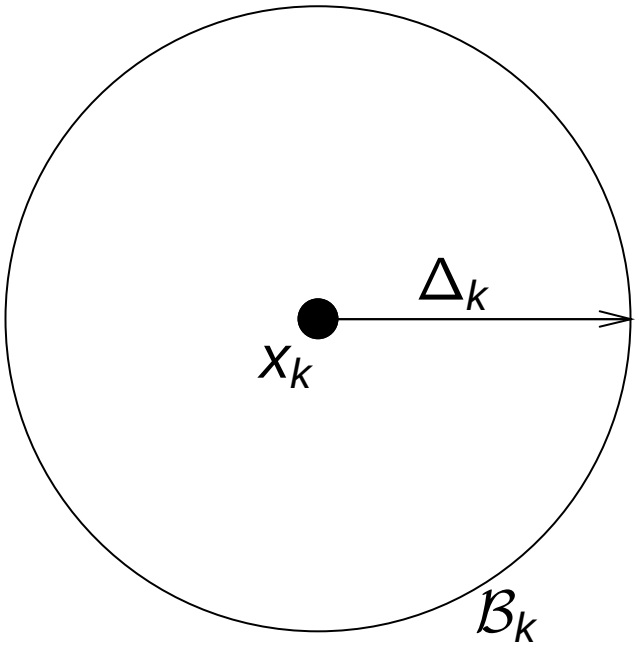
Stop when some criteria are met (e.g. norm of the relative gradient must be small enough).

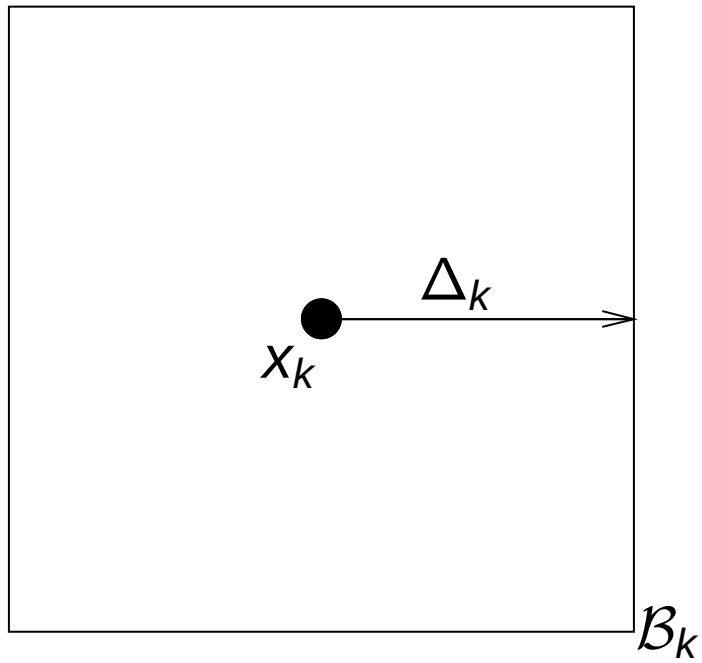
The neighborhood where we considere the model as valid is mathematically defined by a ball centered at x_k, and with a radius Δ_k :

$$\mathcal{B}_k = \{x \mid \|x - x_k\|_k \leq \Delta_k\}.$$

Norms

The norm choice may depend of the iteration k. We often use the 2- or ∞-norm.





Let x be a vector of $\mathbb{R}^n$.

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_{(i)}^2}, \quad \|x\|_\infty = \max_{i=1,\ldots,n} \{|x_{(i)}|\}, \quad \|x\|_1 = \sum_{i=1}^{n} |x_{(i)}|.$$

Trust region update

At the end of each iteration k, we update the trust-region radius as following :

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \infty) & \text{si } \rho_k \geq \eta_2, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{si } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{si } \rho_k < \eta_1, \end{cases}$$

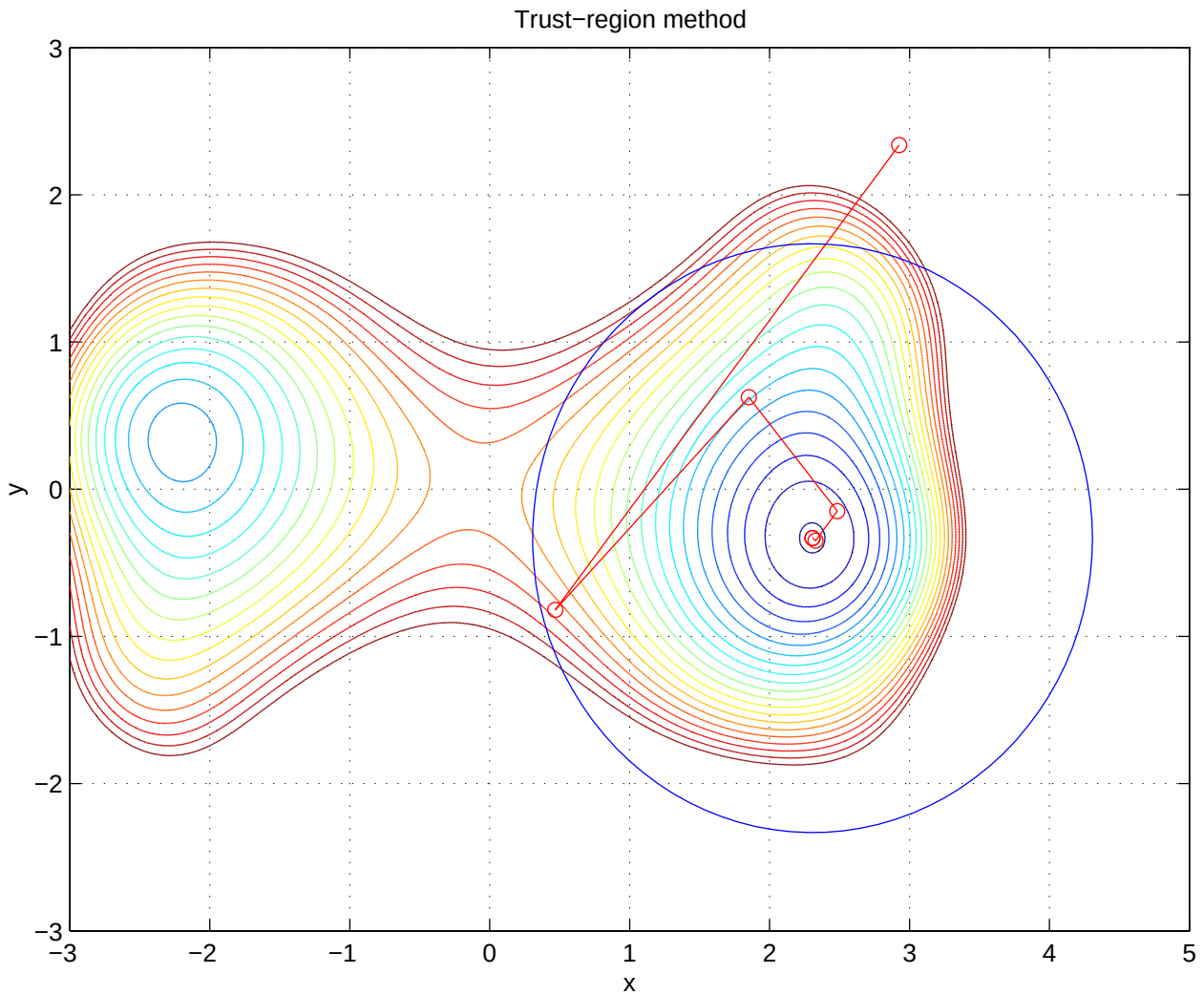
with $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \gamma_1 \leq \gamma_2 < 1$.

Remark : there exist variants, both for theoretical or pratical concerns.

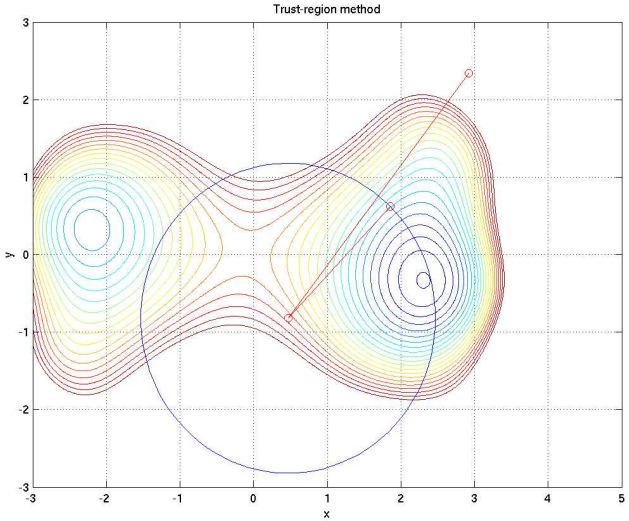
The choice of Δ_0 remains difficult, and various heuristics have been proposed. See for instance A. Sartenaer, *Automatic Determination Of An Initial Trust Region In Nonlinear Programming*, SIAM Journal of Scientific Computing 18(6), pp. 1788-1803, 1997.

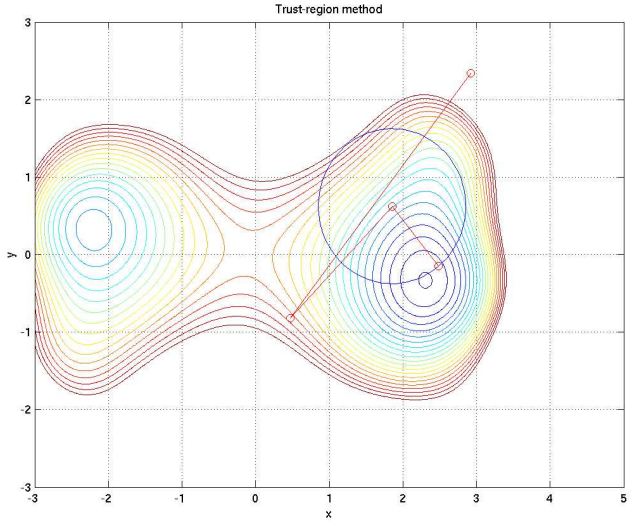
Trust region : example

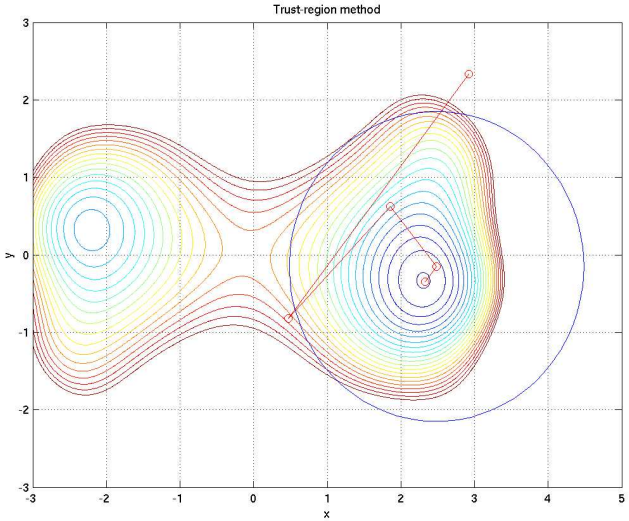
$$\min_{x,y} -10x^2 + 10y^2 + 4\sin(xy) - 2x + x^4$$

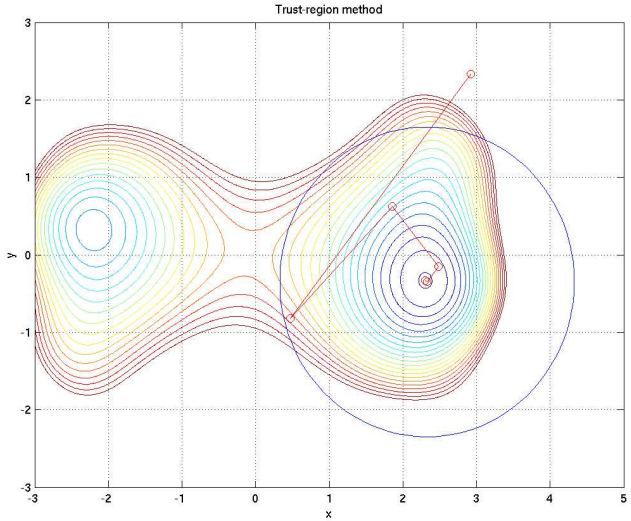


Trust region : example (2)









Adaptive scheme

At iteration k, use R_k draws, with $R^k_{\min} \leq R_k \leq R_{\max}$. (Bastin, Cirillo and Toint, 2006).

Main ingredients : statistical inference to assess SAA accuracy. Estimate required R to equalize accuracy of the SAA objective and model decrease :

$$R^s := \max \left\{ R^k_{\min}, \left\lceil \frac{\alpha \hat{\sigma}^2_{R_k}(x)}{(\Delta m_k^{R_k})^2} \right\rceil \right\}.$$

α is a constant ; $\hat{\sigma}^2_{R_k}(z)$ is the "empirical variance" of the SAA.

Define the new sample size R^+ on the basis of

1. ratio between model improvement and accuracy : $\tau_1^k = \sqrt{R_k} \Delta m_k^{R_k} / \alpha \hat{\sigma}^2_{R_k}(x)$;
2. ratio between current and suggested sizes : $\tau_2^k = R_k / \min\{R_{\max}, R^s\}$.

Convergence

Non-monotone algorithm $\longrightarrow$ safeguards :

- if poor model prediction, adapt the sample size used at x_k or at x_{k+1} to $\max\{R_k, R^+\}$;
- increase the minimum sample size if no sufficient decrease has been observed since the last evaluation of $\hat{g}_{R_{k+1}}$.

Under mild regularity conditions, the algorithm can be proved to converge to a first-order and, under some usual additional conditions, second-order critical points of the SAA (with $R_{\max}$) draws.

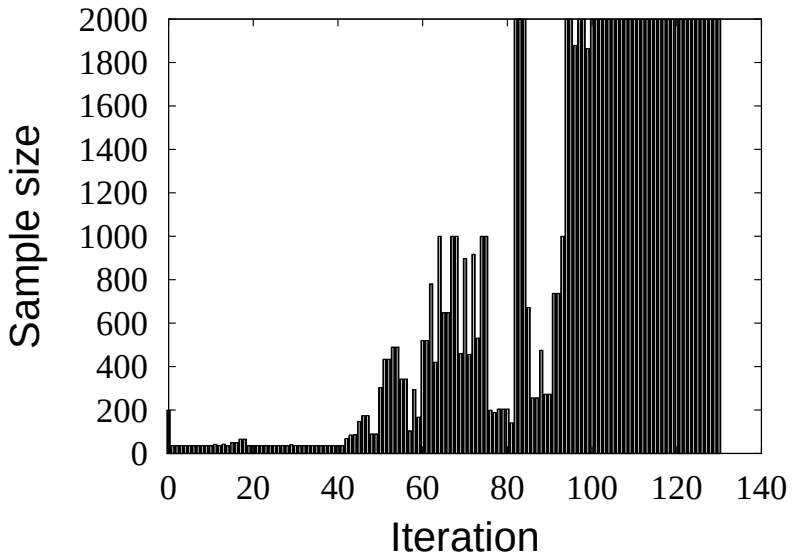
Adapted stopping criterium : stops when the gradient norm is less then a fraction of the estimated accuracy.

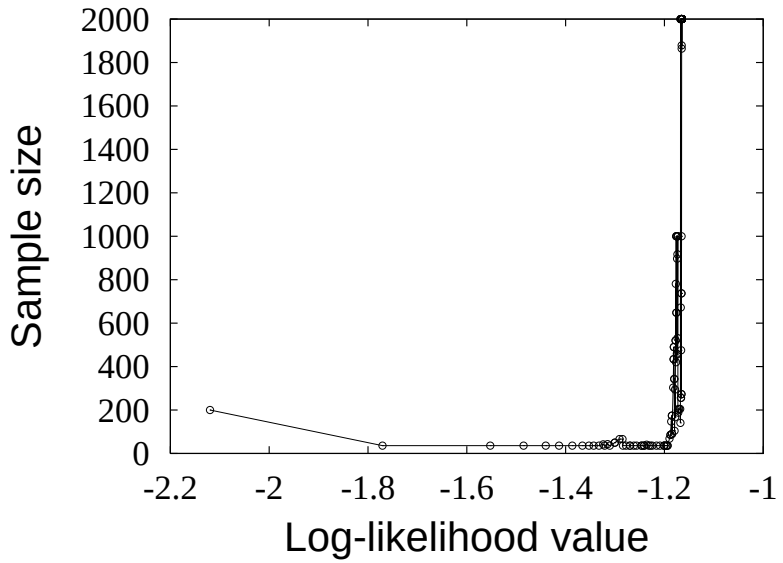
Example

Mixed logit models : stochastic log-likelihood maximization

$$\max_\theta LL(\theta) = \max_\theta \frac{1}{N} \sum_{n=1}^{N} \ln E[P_{ij_i}(x, \theta, \xi)].$$

Mode choice model : Mobidrive data (Axhausen and al., 2002) $N = 5799$ observations, $R_{\max} = 2000$ draws per individual, 14 parameters (integral dimension : 3 normal variates).





↓ Starting value

↓ Maximum value