Optimization principles for discrete choice theory

Fabian Bastin
fabian.bastin@cirrelt.ca
University of Montréal – CIRRELT

Summer School 2015

Likelihood approach

Assume that we face a population of size I, K alternatives, individual choices as well as attributes values for all the alternatives, in every choice situation.

The explanatory variables are supposed to be exogeneous to the choice situation.

The classical approach aims to find the parameters β that maximize the probability of the observed choices:

$$\max_{\beta} \mathcal{L}(\beta) = \prod_{i=1}^{I} \prod_{k=1}^{K} \left(P_{i,k}(\beta)\right)^{Y_{i,k}},$$

where

$$Y_{i,k} = \begin{cases} 1 & \text{if } i \text{ has chosen } k; \\ 0 & \text{otherwise.} \end{cases}$$

Likelihood function

Equivalently, we can write

$$\max_{\beta} \mathcal{L}(\beta) = \prod_{i=1}^{I} P_i(\beta),$$

where we have omitted the index of the chosen alternative in $P_i(\beta)$.

Note: the previous formulation assume that the observations are indépendent.

$\forall\, i,\ 0 < P_i(\beta) < 1$. Numerically: not stable.

Solution: transform the product to a sum by using the logarithm operator:

$$\max_{\beta} \mathcal{LL}(\beta) = \sum_{i=1}^{I} \ln P_i(\beta).$$

Optimization problem

We face an unconstrained optimization problem:

$$\max_{\beta} LL(\beta) = \frac{1}{I} \sum_{i=1}^{I} \ln P_i(\beta),$$

where we have introduced the factor $\frac{1}{I}$ in order to control the function behavior when I rises to infinity. Many authors do not introduce this factor, but it allows to maintain a value that is not sensitive to the population size.

In many cases, the function will even be concave, so we have a convex optimization problem.

Optimization problem

More generally, we consider the problem

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

where we assume 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:

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

Maximum log-likelihood:

$$\max_{\beta} \mathcal{LL}(\beta) = \frac{1}{I} \sum_{i=1}^{I} \log f(x_i | \beta).$$

Some notations...

Recall that the gradient vector and the Hessian are

$$\nabla_x f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{pmatrix}^T,$$

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

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 $\lambda > 0$ such that

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

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 is $-\nabla_x f(x)$, 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. If 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.

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)

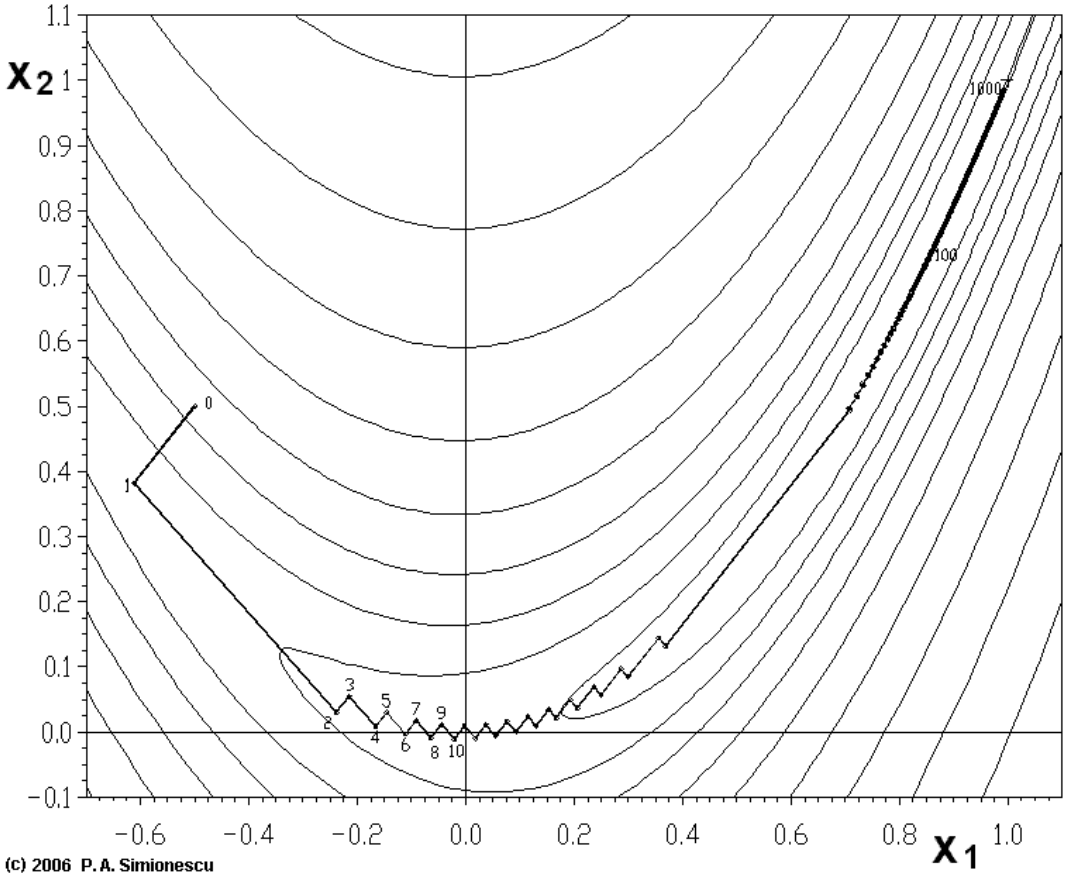
Suppose that $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on the set $S = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$, and that S is a closed and bounded set. then every point x^* that is a cluster point of the sequence $\{x_k\}$ satisfies $\nabla_x f(x^*) = 0$.

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: assume $\frac{1}{2} d^T \nabla^2_{xx} f(x^*) d > 0$ if $\|d\| < \epsilon$ (sufficient second-order criticality condition - local strong convexity).

If $\nabla_x f(x^*) = 0$, then x^* is a local minimizer.

Otherwise, we can choose d such that $\nabla_x f(x^*)^T d < 0$ and x^* is not a local minimizer.

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 convergence 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 i^{th} 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.

Particular case: maximum likelihood.

$$\max \mathcal{LL}(\beta) = \sum_i P_i(\beta).$$

Information identity: when $I \to \infty$, and under some conditions (basically, the model has to be correctly specified), we have

$$H(\beta^*) = -B(\beta^*),$$

where B is the information matrix.

The BHHH method, proposed by Berndt, Hall, Hall, Hausman in 1974, simply replaces the Hessian by the opposite of the information matrix in the quasi-Newton method.

Information matrix

The information matrix is

$$B = E\left[\frac{\nabla P(\beta^*)}{P(\beta^*)}\frac{\nabla P(\beta^*)^T}{P(\beta^*)}\right].$$

In practice, we compute it as

$$B = \frac{1}{I}\sum_{i}^{I}\frac{\nabla P_i(\beta^*)}{P_i(\beta^*)}\frac{\nabla P_i(\beta^*)^T}{P_i(\beta^*)}.$$

Advantage: cheap to compute!
Drawback: only valid if the information identity holds. In practice, rarely the case! Example: multinomial logit on panel data.

Suggestion: start with the BHHH and then switch to BFGS or SR1.

Asymptotics

It can be shown that

$$\sqrt{l}(\beta_l^* - \beta^*) \overset{\mathcal{D}}{\Rightarrow} \mathcal{N}(0, H^{-1}BH^{-1}).$$

so the information matrix will be needed when computing some confidence intervals on the parameters.

Under the information identity, this can be simplified as

$$\sqrt{l}(\beta_l^* - \beta^*) \overset{\mathcal{D}}{\Rightarrow} \mathcal{N}(0, -B),$$

but since the identity often do not hold, care must be exercised.

Note: a model can be statistically consistent while not correctly formulated (i.e. one obtains the right parameters)!
Example: multinomial logit on panel data.

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

$$\{x_k + \alpha d_k \,|\, \alpha > 0\}.$$

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

Principle: 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:

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

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:

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

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

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

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

Trust region update

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

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

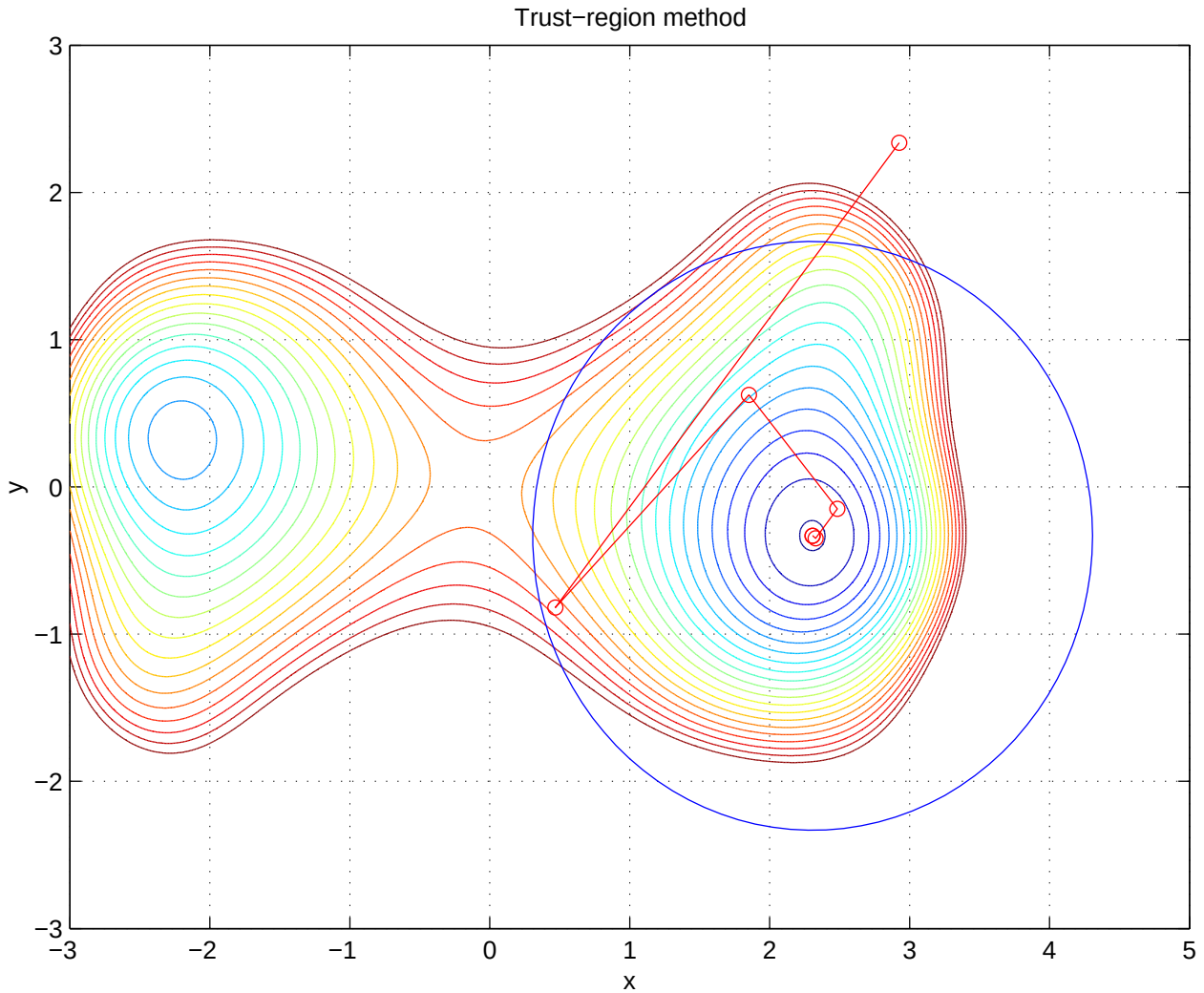
with $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \gamma_1 \leq \gamma_2 < 1$.

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)