INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

A new method for nonlinear optimization

Danillo Roberto Pereira *Jorge Stolfi*
Rafael Felipe Veiga Saracchini

Technical Report - IC-11-19 - Relatório Técnico

November - 2011 - Novembro

A new method for nonlinear optimization

Danillo Roberto Pereira Jorge Stolfi Rafael Felipe Veiga Saracchini *

Abstract

In this work we present a new method for nonlinear optmization based on quadratic interpolation, on the search for stationary point and edge-divided simplex. We show some results of the convergence of our method applied in various functions to one or more guesses.

1 Function optmization by the edge-divided simplex method

1.1 Quadratic interpolation

We consider first the problem of determining a quadratic function $Q : \mathbb{R}^n \to \mathbb{R}$ from given values at certain sample points. The function can be written in matrix form as

$$Q(x) = x^\top A\, x + x^\top B + C \tag{1}$$

where A is a symmetric $n \times n$ matrix, B is a column n-vector and C is a constant. The coefficients A, B and C have $m = (n+1)(n+2)/2$ unknown elements, and therefore they could be obtained in principle from the values of $Q(q_i)$ of Q at m sampling points q_i, $i \in \{0..m-1\}$ by solving a system with m linear equations. However, depending on the placement of the sampling points q_i, the system may be indeterminate or numerically unstable. We show that by proper selection of the sample points, the function can be determined directly by a simple stable formula.

Our method samples Q at the corners and edge midpoints of an arbitrary simplex in $\mathbb{R}^n$. We start with a simplex $S = (p_0, p_1, ..., p_n)$ with $n+1$ points not all in the same hyperplane. Then we compute the midpoints $p_{ij} = (p_i + p_j)/2$ between the vertices p_i and p_j, for $0 \le i, j \le n$. Note that $p_{ii} = p_i$ and $p_{ij} = p_{ji}$, so there are only $n(n-1)/2$ midpoints to be computed. In total we have the minimum required number $m = (n+2)(n+1)/2$ of sample points. See figure 1.

Next we express the function Q in terms of barycentric coordinates relative to the simplex S. Namely, we write

$$Q(x) = \tilde{Q}(z) = z^\top T\, z \tag{2}$$

*Institute of Computing – University of Campinas (UNICAMP). This research is supported by FAPESP (grant 2007/52015-0 and 2009/13744-2) and CAPES.

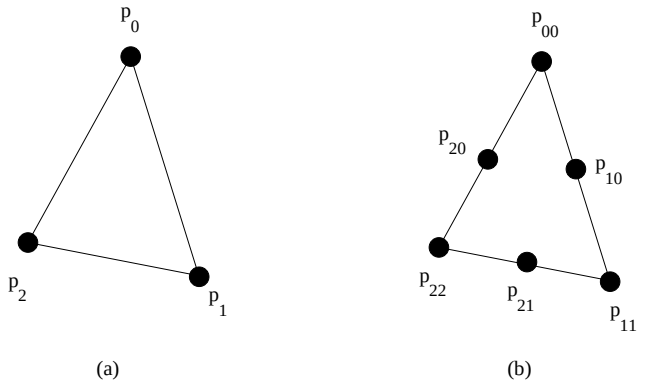

Figure 1: A simplex of $\mathbb{R}^2$ and the derived sampling points p_{ij}.

where z is the column vector of $n+1$ barycentric coordinates of the point x relative to the simplex vertices $p_0, p_1, ..., p_n$. That is, $z = (z_0, z_1, ..., z_n)$ such that

$$\sum_{i=0}^{n} z_i = 1 \qquad \text{and} \qquad \sum_{i=0}^{n} z_i p_i = x \tag{3}$$

The vector z is an element of the *canonical affine n-space* $\mathbb{A}^n$, the set of all vectors of $\mathbb{R}^{n+1}$ that have unit sum.

Let y_{ij} be the given function value at point p_{ij}. Then the interpolating homogeneous quadratic function $\tilde{Q}$ can be obtained by the formula

$$\tilde{Q}(z) = \sum_{i=0}^{n} y_{ii}\, \phi_i(z) + \sum_{\substack{i,j=0 \\ i \neq j}}^{n} y_{ij} \psi_{ij(z)} \tag{4}$$

where ϕ_i and ψ_{ij} are the elements of the quadratic interpolating basis for the sampling points p_{ii} and p_{ij} in barycentric coordinates. Namely,

$$\phi_i(z) = 2\, z_i (z_i - 1/2) \tag{5}$$

$$\psi_{ij}(z) = 2\, z_i\, z_j \tag{6}$$

Observe that

$$\phi_i(p_{rs}) = (i = r)(i = s) \tag{7}$$

for all $i, r, s \in \{0...m\}$, and

$$\psi_{ij}(p_{rs}) = ((i = r)(j = s) + (i = s)(j = r))/2 \tag{8}$$

for all $i, j, r, s \in \{0...m\}$ with $i \neq j$. Therefore the matrix T in formula (2) is given by

$$T_{ij} = \begin{cases} y_i & \text{if } i = j \\ 2y_{ij} - (y_i + y_j)/2 & \text{if } i \neq j \end{cases} \tag{9}$$

1.2 Quadratic optimization

Now consider the problem of finding the stationary point (minimum, maximum, or saddle point) of a quadratic function $Q : \mathbb{R}^n \to \mathbb{R}$. If Q is given by formula (1), the solution is the point $x^* \in \mathbb{R}^n$ such that $\nabla Q(x^*) = 0$ where

$$\nabla Q(x^*) = 2\,A\,x^* + B \tag{10}$$

If Q is expressed in barycentric coordinates by formula (2), the minimum, maximum or saddle point of the corresponding function $\tilde{Q}$ is the point $z^* \in \mathbb{A}^n$ such that $\nabla \tilde{Q}(z^*) = (\lambda, \lambda, ..., \lambda)$ for some $\lambda \in \mathbb{R}$, where $\nabla \tilde{Q}$ is the gradient of $\tilde{Q}$ in $\mathbb{R}^{n+1}$; that is

$$\nabla \tilde{Q}(z^*) = 2\,T\,z^* \tag{11}$$

Therefore the point z^* can be found by solving the linear system $Mz^* = D$ where M is the matrix with $n+2$ lines and $n+2$ columns

$$M = \begin{pmatrix} & & & & & & 1 \\ & & & & & & 1 \\ & & & 2\,T & & & 1 \\ & & & & & & 1 \\ & & & & & & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

D is the column $(n+2)$-vector $(0, 0, .., 0, 1)^\top$, and z^* is the column vector of the $(n+2)$ unknowns $(z_0^*, z_1^*, ..., z_n^*, \lambda)^\top$. Then the point x^* can be computed from z^* by the formula (3).

This method will probably fail if the matrix A of formula (1) has a null or a very small eigenvalue; that is, if there is a principal direction along which the original function Q is approximately affine (first degree polynomial). In those cases the matrix M will be singular or near-singular.

1.3 Iterative quadratic optimization

Suppose now that we want to find a local stationary point (minimum, maximum, or saddle point) x^* of a nonlinear function $f : \mathbb{R}^n \to \mathbb{R}$ (*the goal function*) that is twice differentiable. Suppose also we have an initial guess $x^{(0)}$ for the solution and an upper bound $\delta^{(0)}$ for the distance from $x^{(0)}$ to the true solution x^*. We can do so by an iterating the quadratic optmization algorithm of section 1.2. See algorithm 1. The source code available at [1].

Algorithm 1 *DivSmpMin*$(f, x, \delta, \epsilon, t, \theta, \alpha, \beta, \sigma)$

Requires: A function f; the number n of variables, an initial guess $x \in \mathbb{R}^n$; the initial error estimate δ; the error tolerance ϵ; the maximum number of iterations t; the relative simplex radius θ; the uncertainty adjustment factors α and β; and the optmization direction $\sigma \in \{-1, 0 + 1\}$.

Computes: The local optimum point x^* of f and the new uncertainty estimate δ

while $t \geq 0$ **do**
  $r \leftarrow \theta\,\delta$;
  $(p, y) \leftarrow SimplexSample(f, n, x, r)$;
  $z^* \leftarrow OptQuad(f, n, y)$;
  $x^* \leftarrow \sum_{i=0}^{n+1} z_i^*\, p_{ii}$;
  $d \leftarrow \|x - x^*\|$;
  if $(d > \delta)$ **then** $x^* \leftarrow x + \delta\,(x - x^*)/d$;
  if $(\sigma = +1)$ **then**
    Let y_{ij} the maximum sample function value;
    if $(f(x^*) < y_{ij})$ **then** $x^* \leftarrow p_{ij}$; $d \leftarrow \|x - x^*\|$;
  if $(\sigma = -1)$ **then**
    Let y_{ij} the minimum sample function value;
    if $(f(x^*) > y_{ij})$ **then** $x^* \leftarrow p_{ij}$; $d \leftarrow \|x - x^*\|$;
  $\delta^* = \min\{\alpha\delta, \beta\sqrt{d^2 + n}\}$
  $x \leftarrow x^*$; $\delta \leftarrow \delta^*$; $t \leftarrow t - 1$;
  if $(\delta^* < \epsilon)$ **then** return (x, δ);
end while

At each step, our algorithm first generates a simplex S with circumradius $\theta\delta$ surrounding the current guess x, where θ is some fixed parameter usually smaller than 1. Then we interpolate the goal function f by a quadratic function Q at the vertices and midpoints of S, as described in section 1.1. Next we compute the stationary point x^* of Q as described in section 1.2. If the new guess is adequate, we compute a new uncertainty δ based on the current δ and the distance $\|x^* - x\|$, and we set the next guess x to x^*. This process is repeated until the estimated uncertainty δ is less than a prescribed tolerance ϵ, or a fixed computation budget is exhausted.

The input parameter σ indicates whether the algorithm should search for a maximum ($\sigma = +1$), a minimum ($\sigma = -1$) or any stationary point ($\sigma = 0$). The new guess x^* is considered to be inadequate if the length $\|x^* - x\|$ of the displacement is greater than δ. In that case we truncate to displacement to length δ preserving its direction (step 7). When looking for a minimum ($\sigma = -1$) or a maximum ($\sigma = +1$), the new guess x^* is inadequate also if its function value is not better than the sample values. If so, we set x^* to the point p_{ij} that has the best value of $f(p_{ij})$and adjust x according (steps 8 to 13).

To justify this method we observe that if the vertices of S are sufficiently close to the optimum x^*, then Q is to close to f; and if f is positive definite at x^*, then the stationary point x^* of Q will be close to true optimum x^* of f. In fact, convergence is expected to be quadratic as in Newton's root-finding method, for the same reasons.

This method generally fails if the quadratic interpolant Q is not a good match to the goal function; either because function is "noisy" or because the initial guess is not close enough to the minimum, or because the initial simplex radius $\theta\delta$ is too large.

1.4 Examples

To ilustrate the method we test it with various functions and various starting guesses. For each test we ran the algorithm for several starting guesses located on a regular array around the true minimum. We show the results as an "arrow plot" (such as figure 3) where each arrow, representing one run of algorithm, connects the starting guess to the final result of that run. We also show a trace of a single run of the algorithm (such as figure 4) where each arrow connects one intermediate guess x to the next guess x^*, and each circle has center at the current guess x and radius equal to the uncertainty δ. In all tests, we set $\alpha = 0.9$, $\beta = 0.9$, $\theta = 0.5$, $\epsilon = 0.001$ and $t = 10$.

1.4.1 Quadratic function

For this test, the goal function is a quadratic polynomial in two variables [3].

$$f_1(x_0, x_1) = (x_0 + 2x_1 - 7)^2 + (2x_0 + x_1 - 5)^2 \tag{12}$$

The minimum x^* of f_1 is the point $(1, 3)$ where $f1$ has the value 0. As expected for a quadratic function the algorithm finds the minimum in a single iteration.

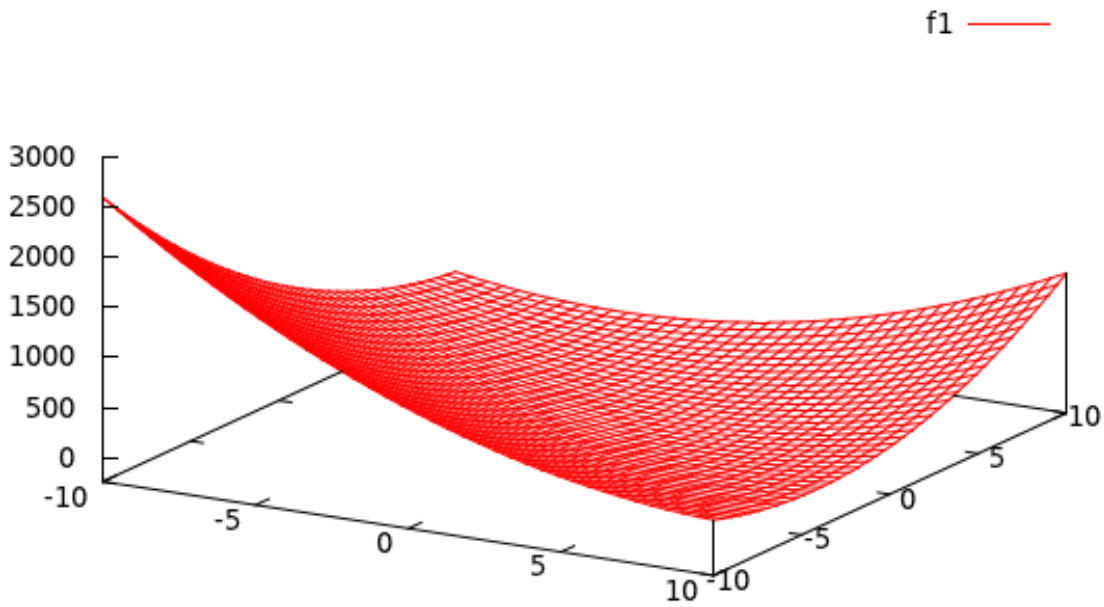

Figure 2: Graph of the function f_1. The plotted region is $[-11, 11] \times [-11, 11]$.

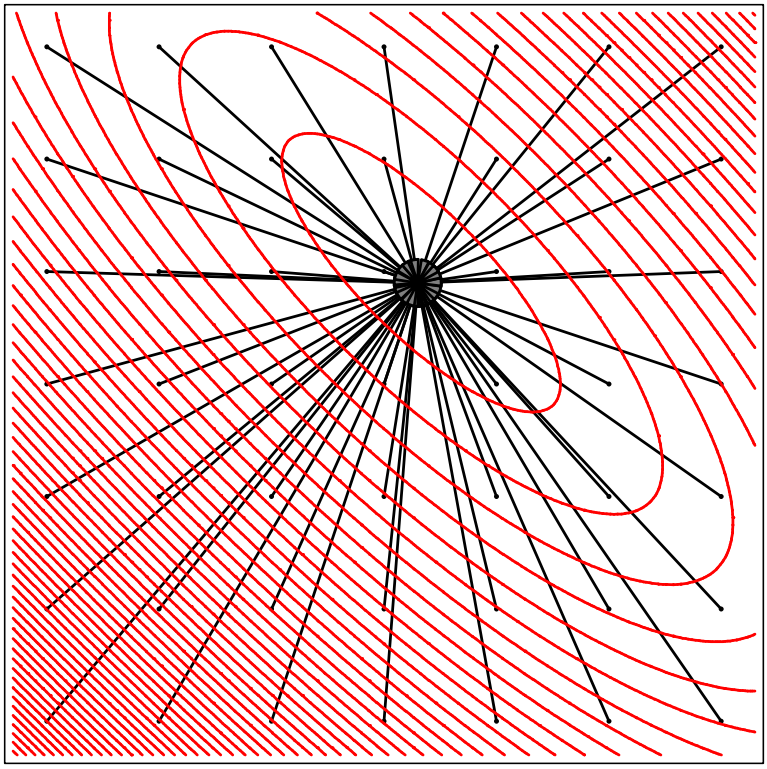

Figure 3: The initial guesses and final results of our algorithm applied to f_1 at several points in the square $[-10, 10] \times [-10, 10]$. The plotted region is $[-11, 11] \times [-11, 11]$. The uncertainty initial δ was set to 15.

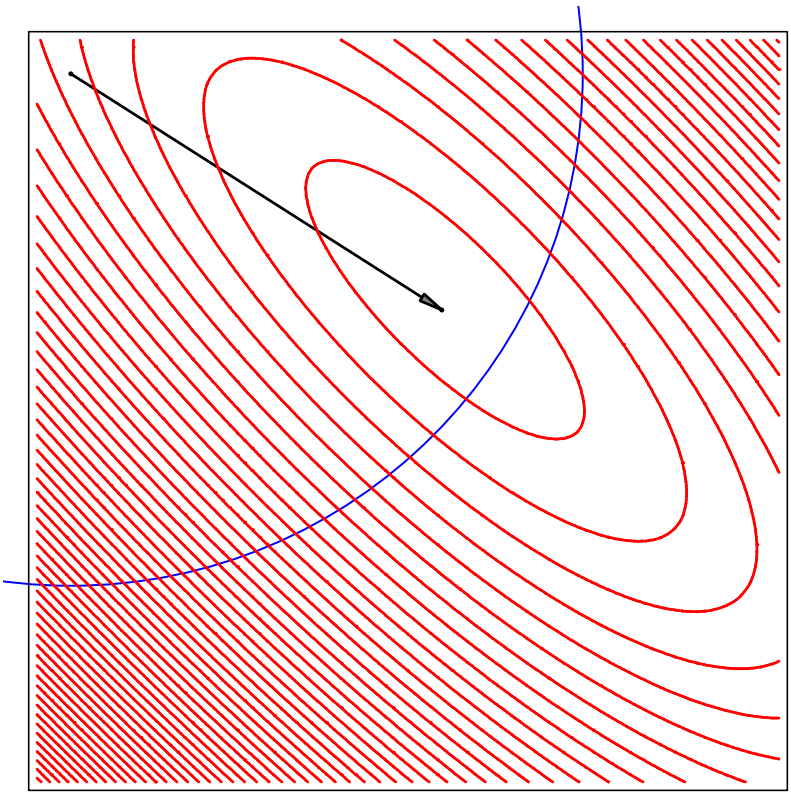

Figure 4: Behavior of our algorithm applied to f_1 with initial guess $x = (-10, 10)$ and uncertainty $\delta = 15$. The plotted region is $[-11, 11] \times [-11, 11]$.

1.4.2 Trigonometric polynomial

For this test, the goal function f_2 [3] is a sum of sinusoidal waves along x_0 and x_1, with various frequencies and phases

$$\begin{array}{lllll} f_2(x_0, x_1) & = & & & \\ & & [\cos(2x_1+1) & + & 2\cos(3x_1+2)+ \\ & & 3\cos(4x_1+3) & + & 4\cos(5x_1+4) \quad + \quad 5\cos(6x_1+5)] \\ & & [\cos(1) & + & 2\cos(x_0+2)+ \\ & & 3\cos(2x_0+3) & + & 4\cos(3x_0+4) \quad + \quad 5\cos(4x_0+5)] \end{array} \tag{13}$$

The minimum point x^* of f_2 is approximately $(4.97648, 4.85806)$, where has value ≈ -176.542.

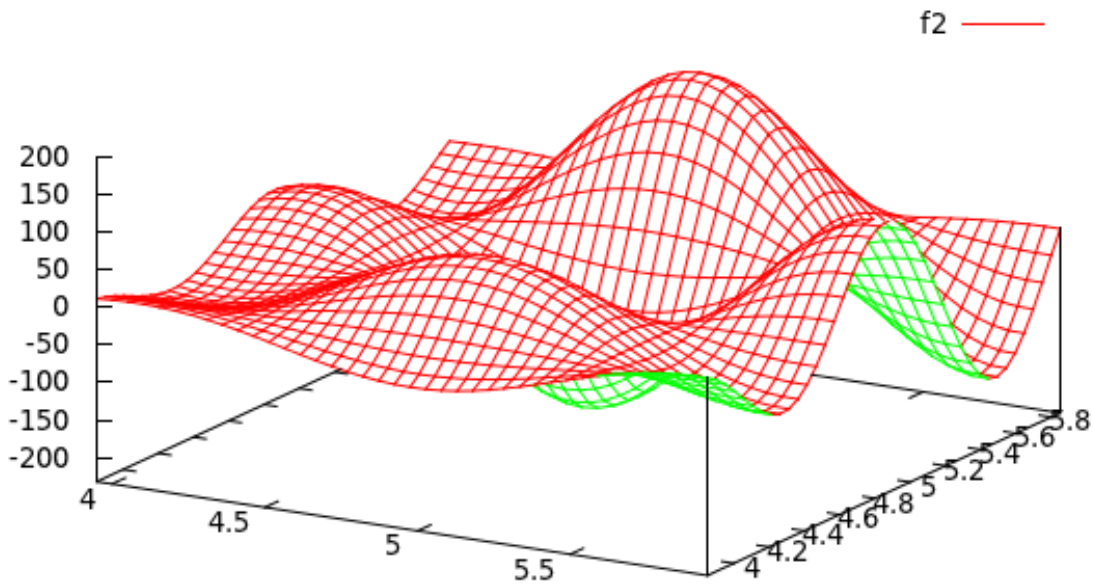

Figure 5: Graph of the function f_2. The plotted regions is $[3.95, 5.95] \times [3.85, 5.85]$.

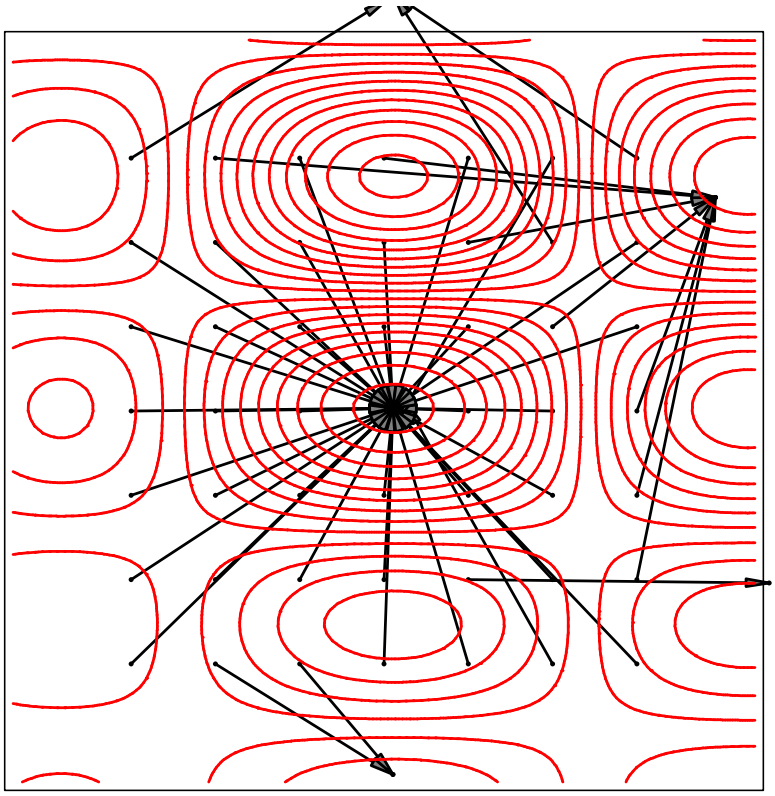

Figure 6: The initial guesses and final results of our algorithm applied to f_2 at several points in the square $[4.2, 5.7] \times [4.1, 5.6]$. The initial uncertainty δ was set to 0.85. The plotted region is $[3.95, 5.95] \times [3.85, 5.85]$.

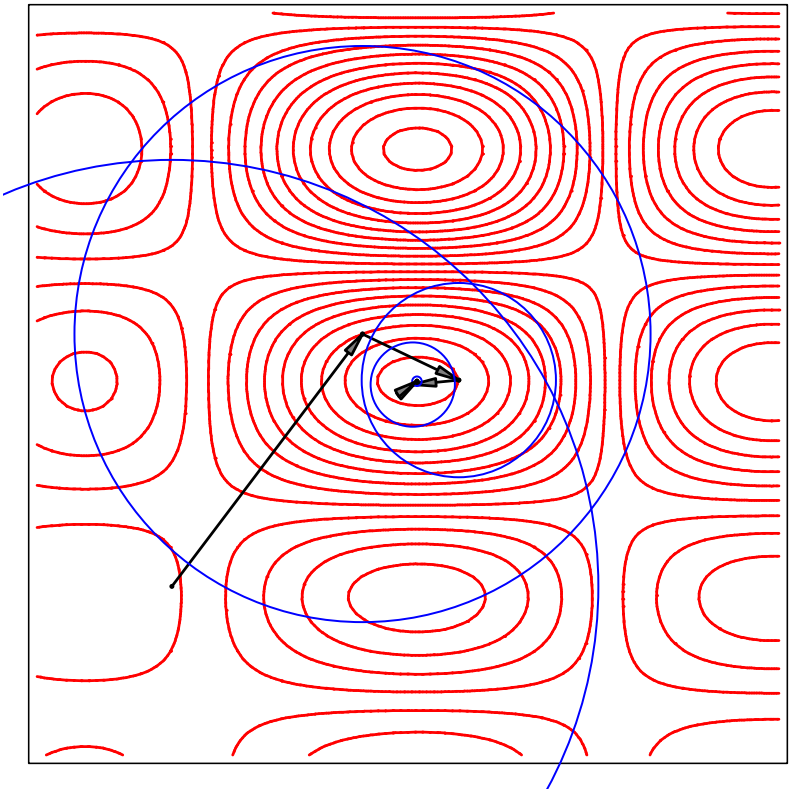

Figure 7: Behavior of our algorithm applied f_2 at initial guess $x = (4.25, 4.25)$ and uncertainty $\delta = 1.25$. The plotted region is $[3.95, 5.95] \times [3.85, 5.85]$.

1.4.3 Sextic polynomial

For this test, the goal function f_3 is the "six-hump camel back function," [2] a polynomial of the sixth degree

$$f_3(x_0, x_1) = (4 - \frac{21}{10}\, x_0^2 + \frac{x_0^4}{3})\, x_0^2 + x_0\, x_1 + (-4 + 4x_1^2)x_1^2 \tag{14}$$

This function has two local minima, approximately at $(-0.0898, 0.7126)$ and $(0.0898, -0.7126)$, both with value ≈ -1.0316.

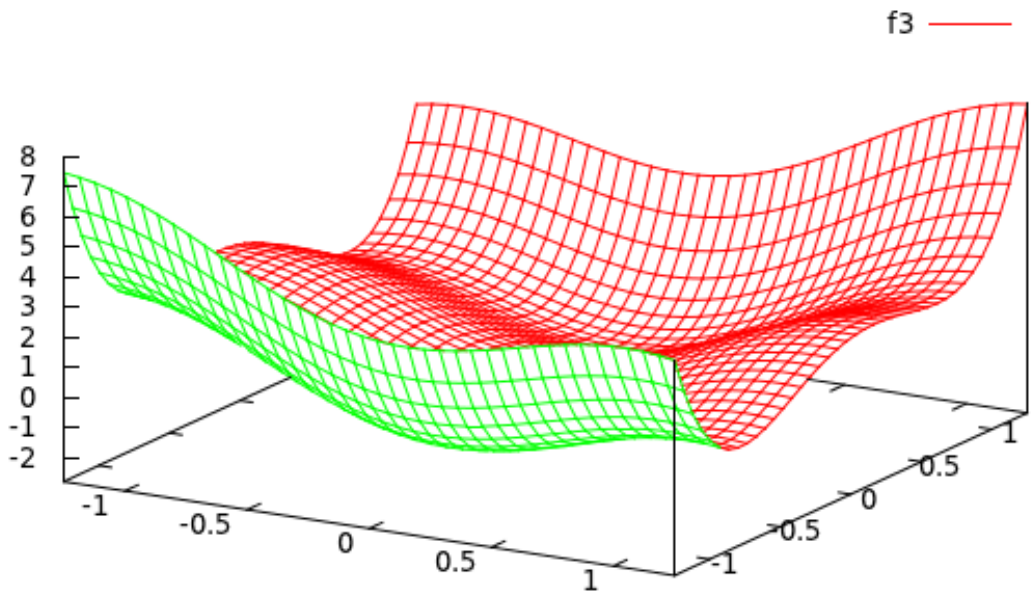

Figure 8: Graph of the function f_3. The plotted region is $[-1.25, 1.25] \times [-1.25, 1.25]$.

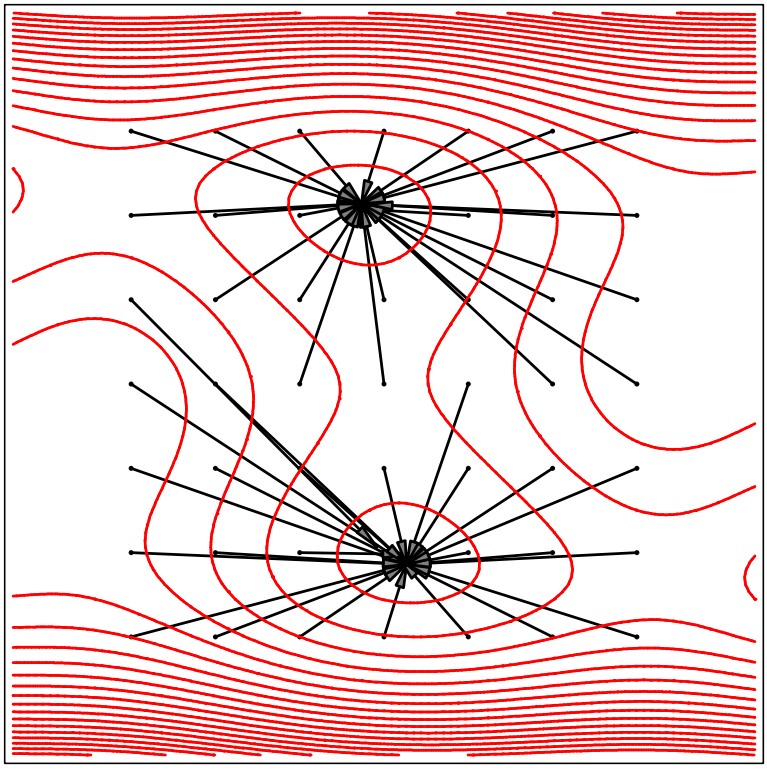

Figure 9: The initial guesses and final results of our algorithm applied to f_3 at several points in the square $[-1, 1] \times [-1, 1]$, with initial uncertainty δ was set to 0.85. The plotted region is $[-1.25, 1.25] \times [-1.25, 1.25]$.

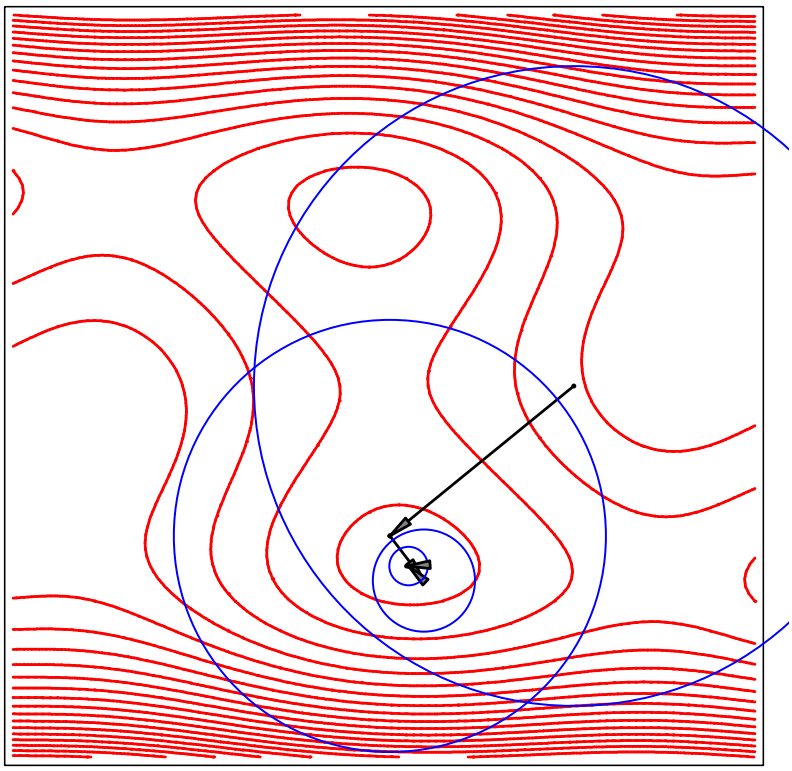

Figure 10: Behavior of our algorithm applied to f_3 with initial guess $x = (0.75, 0)$, uncertainty $\delta = 1.1$. The plotted region is $[-1.25, 1.25] \times [-1.25, 1.25]$.

1.4.4 Trigonometric noise function

For this test, the goal function f_4 is a sum of four sinusoidal waves of various directions and phases, and widely different frequencies

$$\begin{array}{rcllcl} f_4(x_0, x_1) & = & 2\cos(3x_0 + 4x_1) & + & \cos(5x_0 - 2x_1) \\ & + & \frac{1}{12}\cos(30x_0 + 12x_1) & + & \frac{1}{15}\cos(13x_0 - 27x_1) \end{array} \tag{15}$$

This function has several local minima; some of them are approximately located at $(0.6829, 0.22)$, $(-0.6829, -0.22)$, $(1.218, 1.418)$ and $(-1.218, -1.418)$, with slightly different function values close to -3.041.

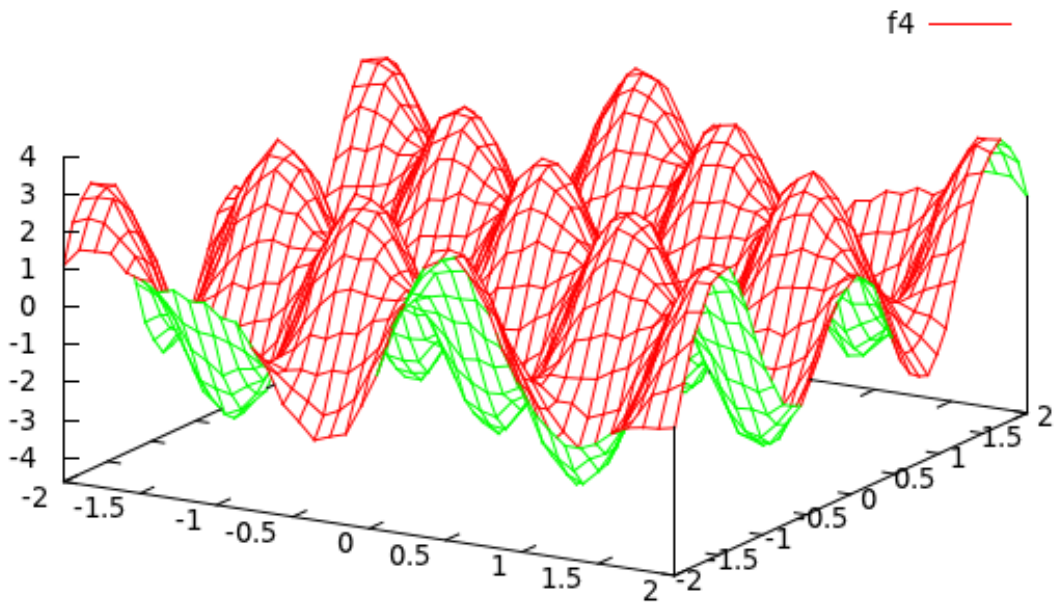

Figure 11: Graph of the function f_4. The plotted region is $[-2, 2] \times [-2, 2]$.

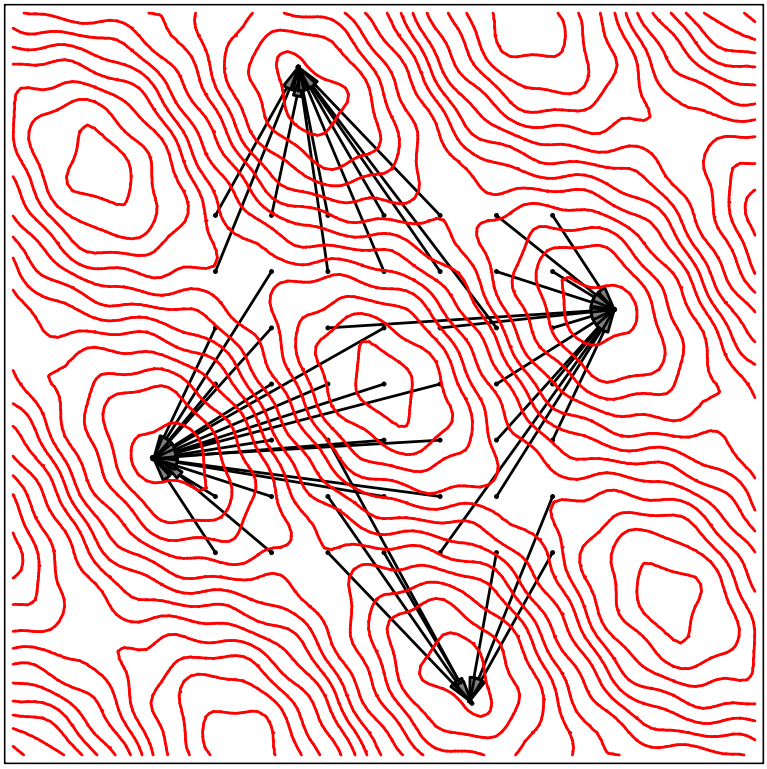

Figure 12: The initial guesses and final results of our algorithm applied to f_4 at several points in the square $[-1, 1] \times [-1, 1]$. The initial uncertainty δ was set to 0.95. The plotted region is $[-2, 2] \times [-2, 2]$.

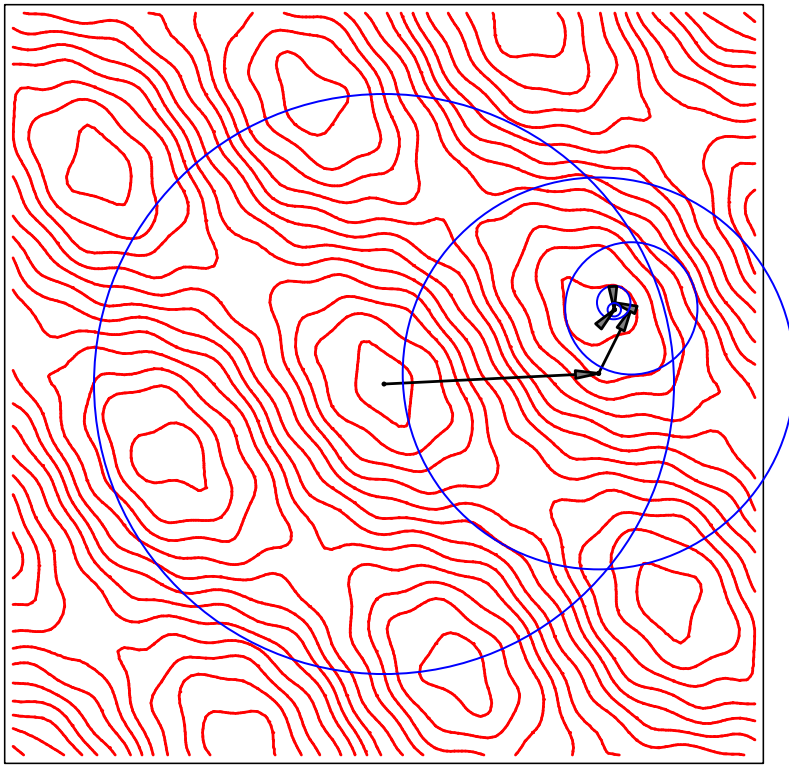

Figure 13: Behavior of our algorithm applied to f_4 with initial guess $x = (0, 0)$, uncertainty $\delta = 0.85$. The plotted region is $[-2, 2] \times [-2, 2]$.

References

[1] The source code available at `http://www.liv.ic.unicamp.br/~danillorp/Onl/`.

[2] Molga, Marcin and Smutnicki, Czesaw. Test functions for optimization needs (2005). `http://www.bioinformaticslaboratory.nl/twikidata/pub/Education/NBICResearchSchool/Optimization/VanKampen/BackgroundInformation/TestFunctions-Optimization.pdf`.

[3] L. H. de Figueiredo, R. Van Iwaarden, and J. Stolfi. Fast interval branch-and-bound methods for unconstrained global optimization with affine arithmetic. Technical Report IC-97-08, Institute of Computing, Univ. of Campinas, June 1997.