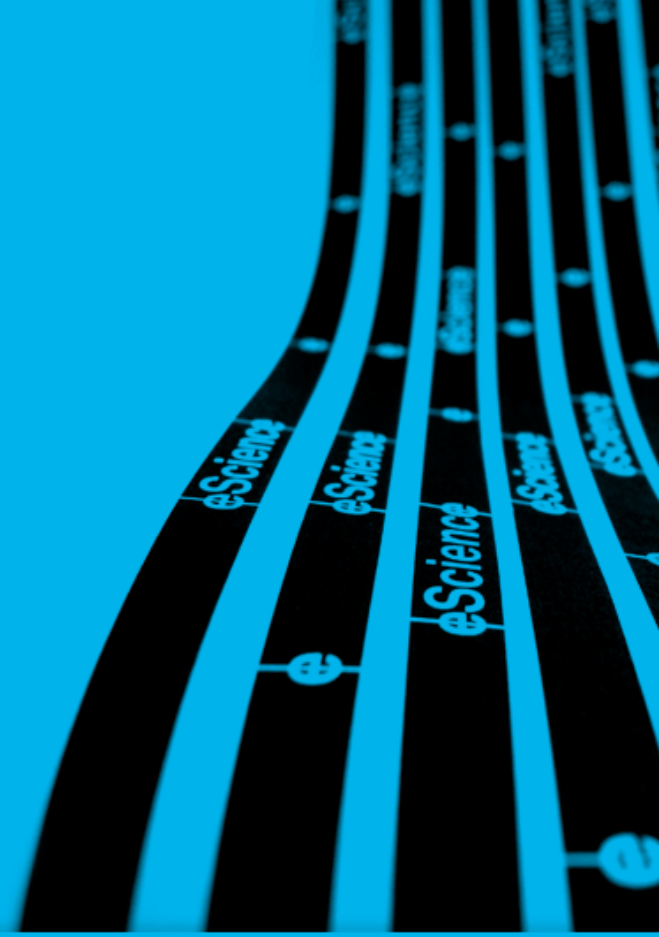
Differential Evolution

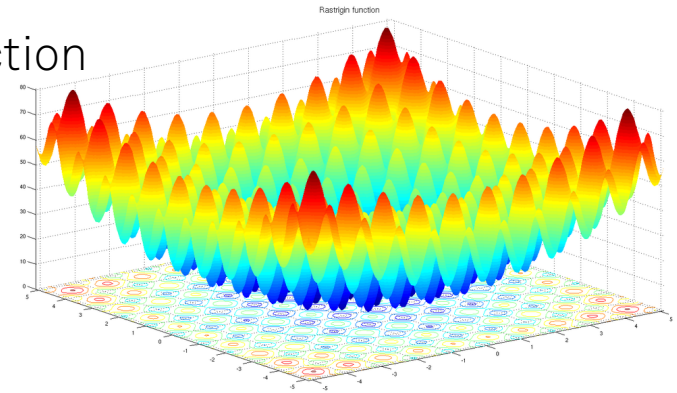


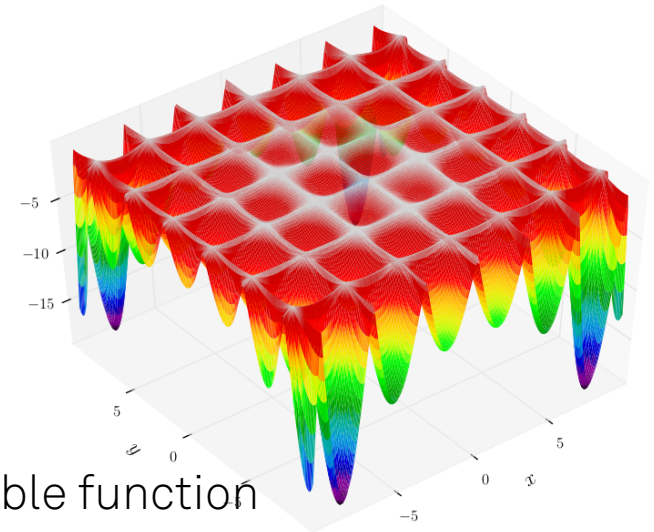
Jurriaan H. Spaaks
Netherlands eScience Center

Differential Evolution

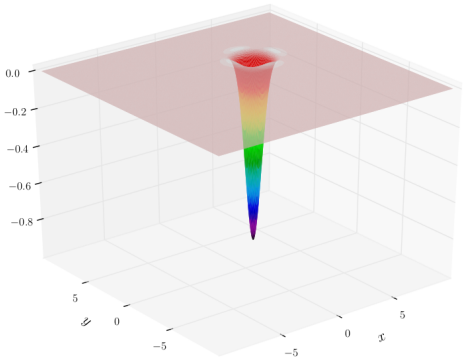
how to deal with difficult 'landscapes'

Rastrigin function
(local optima)





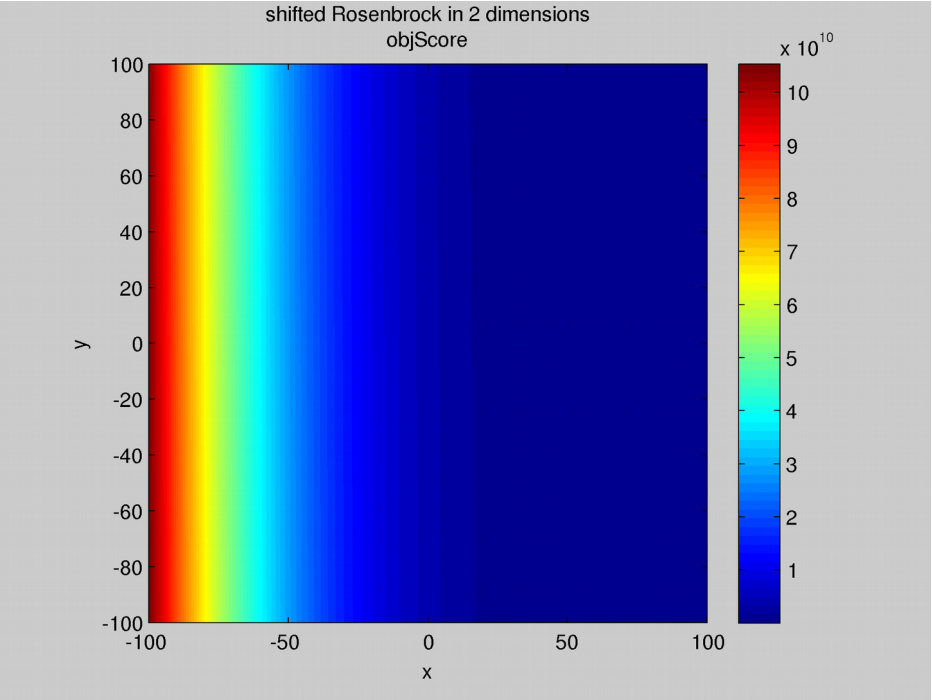
Hölder table function
(distant, multiple
global optima)



Easom function
(insensitivity)

Differential Evolution

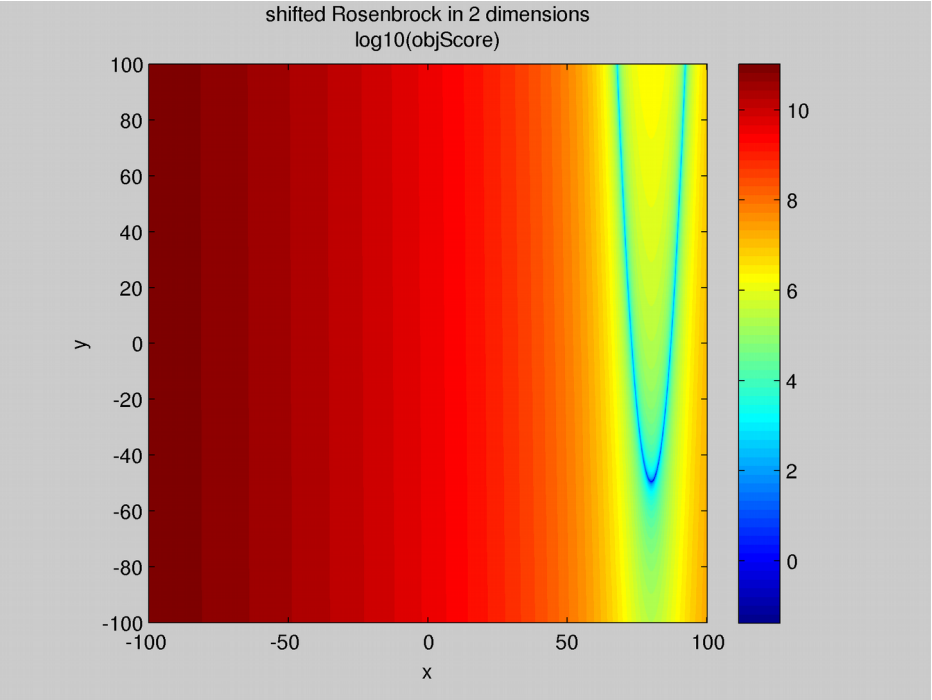
- shifted Rosenbrock function
- 2-D
- constrained parameter space: [-100, 100]
- response surface (4 million points)



can't see much detail, so let's apply `log10` transform on the objective score

Differential Evolution

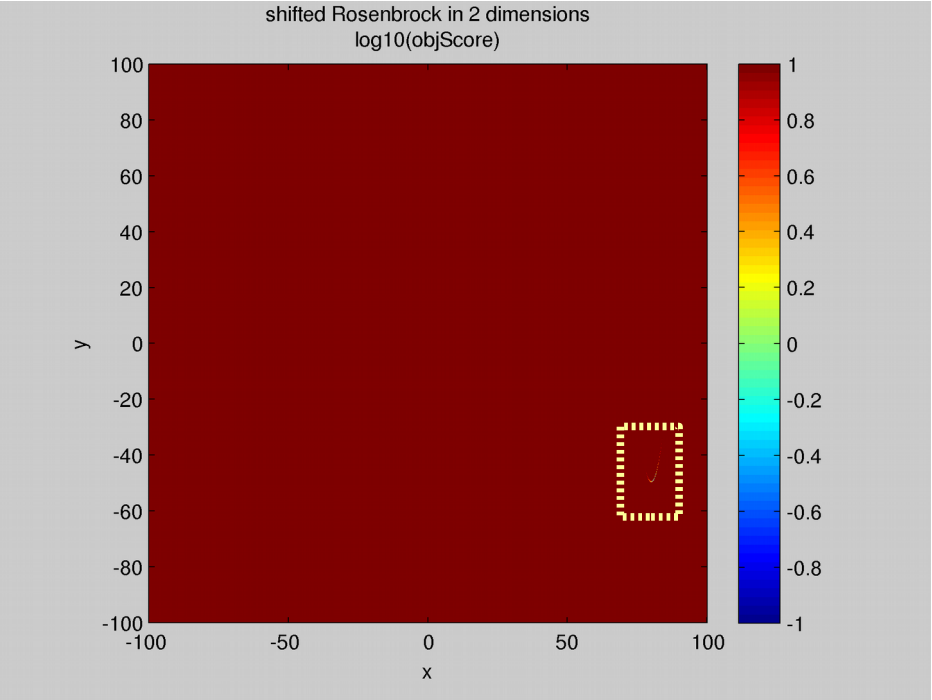
- shifted Rosenbrock
- 2-D
- constrained parameter space: [-100, 100]
- response surface (4 million points)



only interested in blue areas -- let's limit color scale to [-1,1]

Differential Evolution

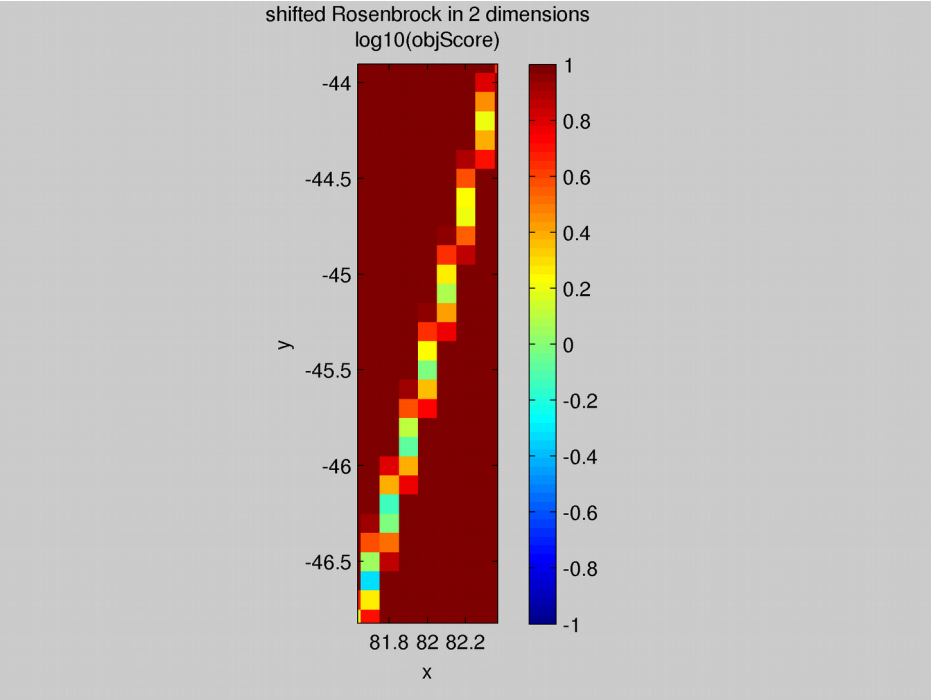
- shifted Rosenbrock
- 2-D
- constrained parameter space: [-100, 100]
- response surface (4 million points)



only interested in dashed area
let's zoom in

Differential Evolution

- shifted Rosenbrock
- 2-D
- constrained parameter space: [-100, 100]
- response surface (4 million points)



local minima!

Differential Evolution

- Storn & Price 1997
- biology inspired
  - mutation
  - crossover
  - selection
- improve population, not point
- no 1st, 2nd derivative needed

Journal of Global Optimization **11:** 341–359, 1997.

Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces

RAINER STORN
Siemens AG, ZFE T SN2, Otto-Hahn Ring 6, D-81739 Muenchen, Germany.
(e-mail:rainer.storn@mchp.siemens.de)

KENNETH PRICE
836 Owl Circle, Vacaville, CA 95687, U.S.A. (email: kprice@solano.community.net)

(Received: 20 March 1996; accepted: 19 November 1996)

Abstract. A new heuristic approach for minimizing possibly nonlinear and non-differentiable continuous space functions is presented. By means of an extensive testbed it is demonstrated that the new method converges faster and with more certainty than many other acclaimed global optimization methods. The new method requires few control variables, is robust, easy to use, and lends itself very well to parallel computation.

Key words: Stochastic optimization, nonlinear optimization, global optimization, genetic algorithm, evolution strategy.

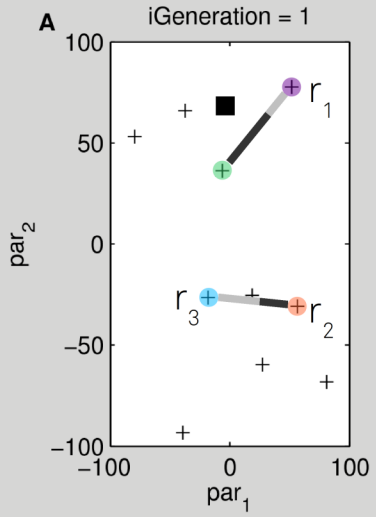
1. Introduction

Problems which involve global optimization over continuous spaces are ubiquitous throughout the scientific community. In general, the task is to optimize certain properties of a system by pertinently choosing the system parameters. For convenience, a system's parameters are usually represented as a vector. The standard approach to an optimization problem begins by designing an objective function that can model the problem's objectives while incorporating any constraints. Especially in the circuit design community, methods are in use which do not need an objective function but operate with so-called regions of acceptability: Brayton *et al.* (1981), Lueder (1990), Storn (1995). Although these methods can make formulating a problem simpler, they are usually inferior to techniques which make use of an objective function. Consequently, we will only concern ourselves with optimization methods that use an objective function. In most cases, the objective function defines the optimization problem as a minimization task. To this end, the following investigation is further restricted to minimization problems. For such problems, the objective function is more accurately called a "cost" function.

When the cost function is nonlinear and non-differentiable, direct search approaches are the methods of choice. The best known of these are the algorithms

Differential Evolution

the principles



- population of points
- (population of distances)

Legend

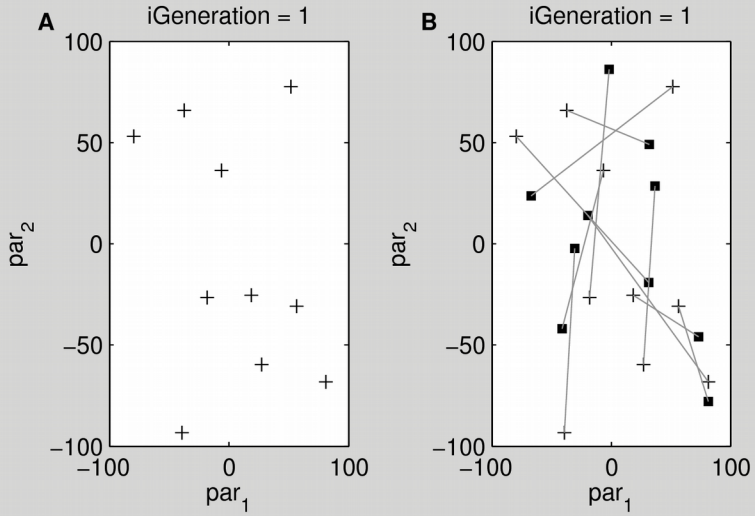
+ parents

Code Snippet 2.2: Generating proposal points given a parent population.

```
% define which columns in parents and in proposals hold the parameter vector
parCols = 1:nDims;

% preallocate the space needed by the samples and their corresponding
% objective score
proposals = repmat(NaN, [nPop, nDims + 1]);

% iterate over the members of the population
for iPop = 1:nPop

    % draw 4 integer numbers from 1 to nPop
    v = randperm(nPop, 4);

    % if iPop happens to be drawn, remove it
    v(v == iPop) = [];

    % retrieve selected rows from the parents array; use only the parameter
    % vector columns
    r1 = parents(v(1), parCols);
    r2 = parents(v(2), parCols);
    r3 = parents(v(3), parCols);

    % calculate the two distances dist1 and dist2
    dist1 = r1 - parents(iPop, parCols);
    dist2 = r3 - r2;

    % calculate the proposal point
    proposals(iPop, parCols) = parents(iPop, parCols) + F * dist1 + K * dist2;
end
```

Differential Evolution

the principles



- proposals generated
- evaluate objective function
- accept proposal if better than parent

Legend

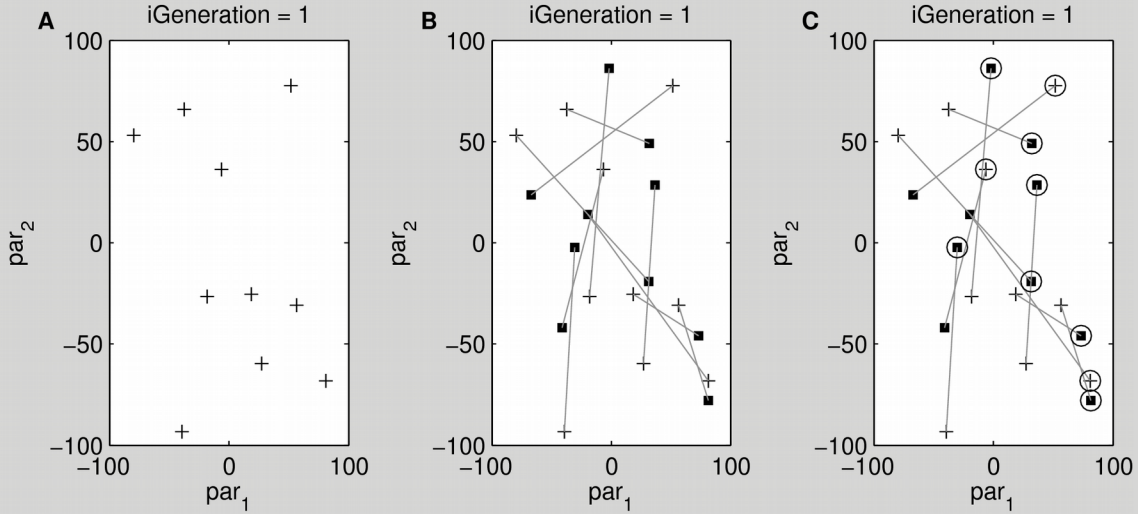
+ parents

■ proposals

parents and their associated proposals

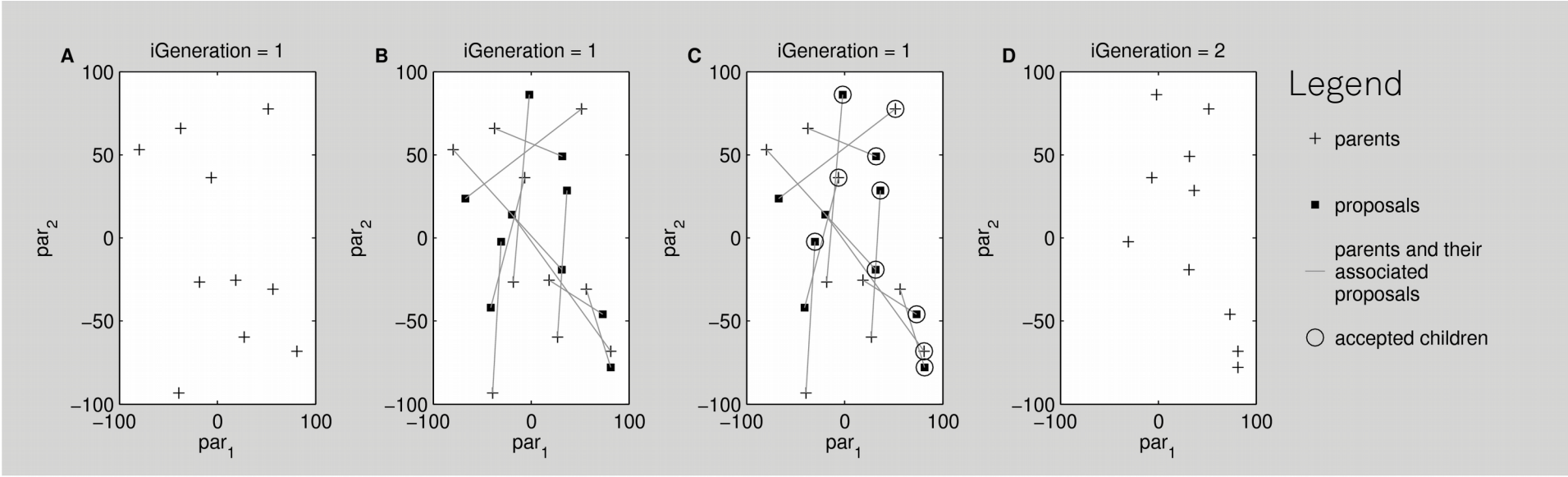
Differential Evolution

the principles



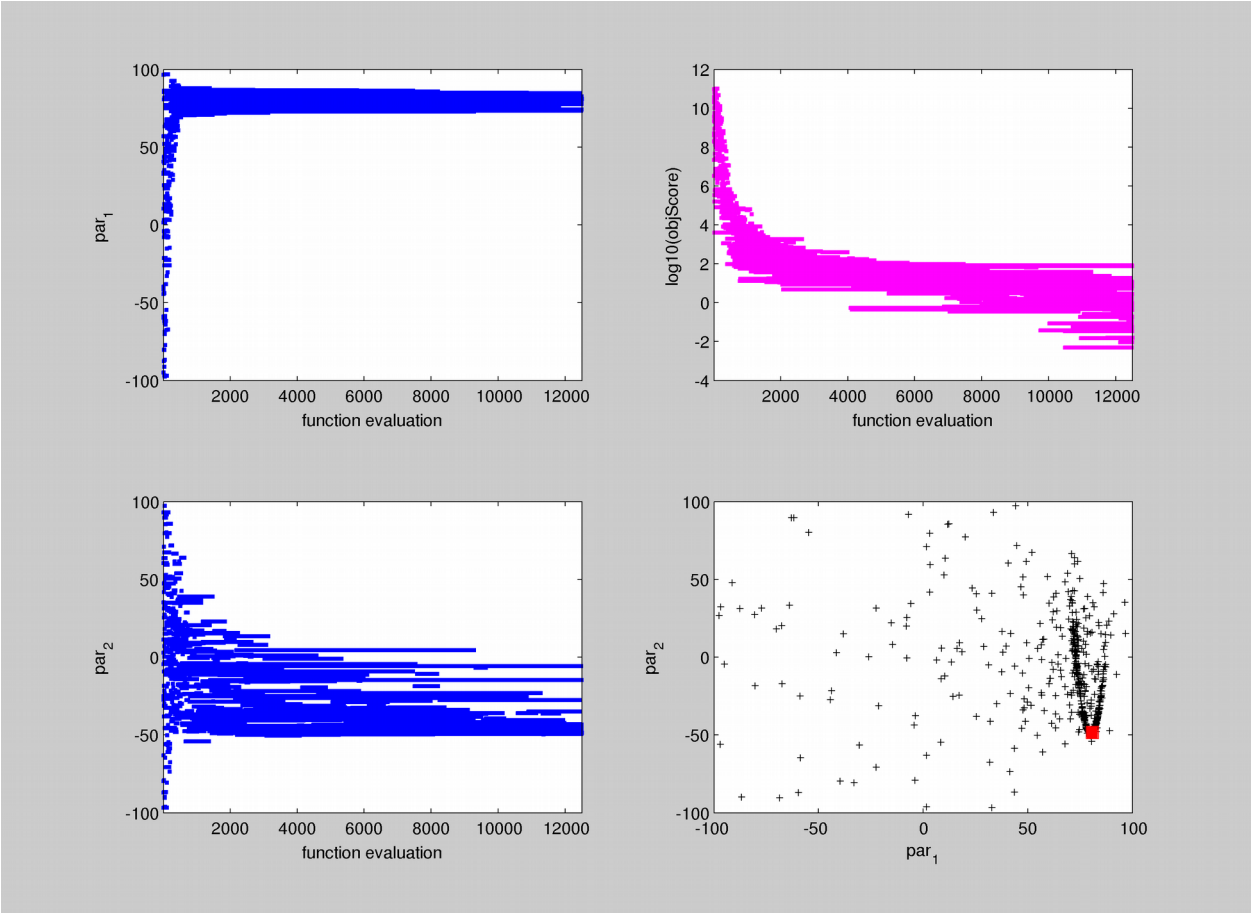
Differential Evolution

the principles



Differential Evolution

- only 12500 function evaluations (not 4 million)
- better resolution near optimum
- (much) better best score



Differential Evolution

Questions?

~~Hammer~~ demo time!