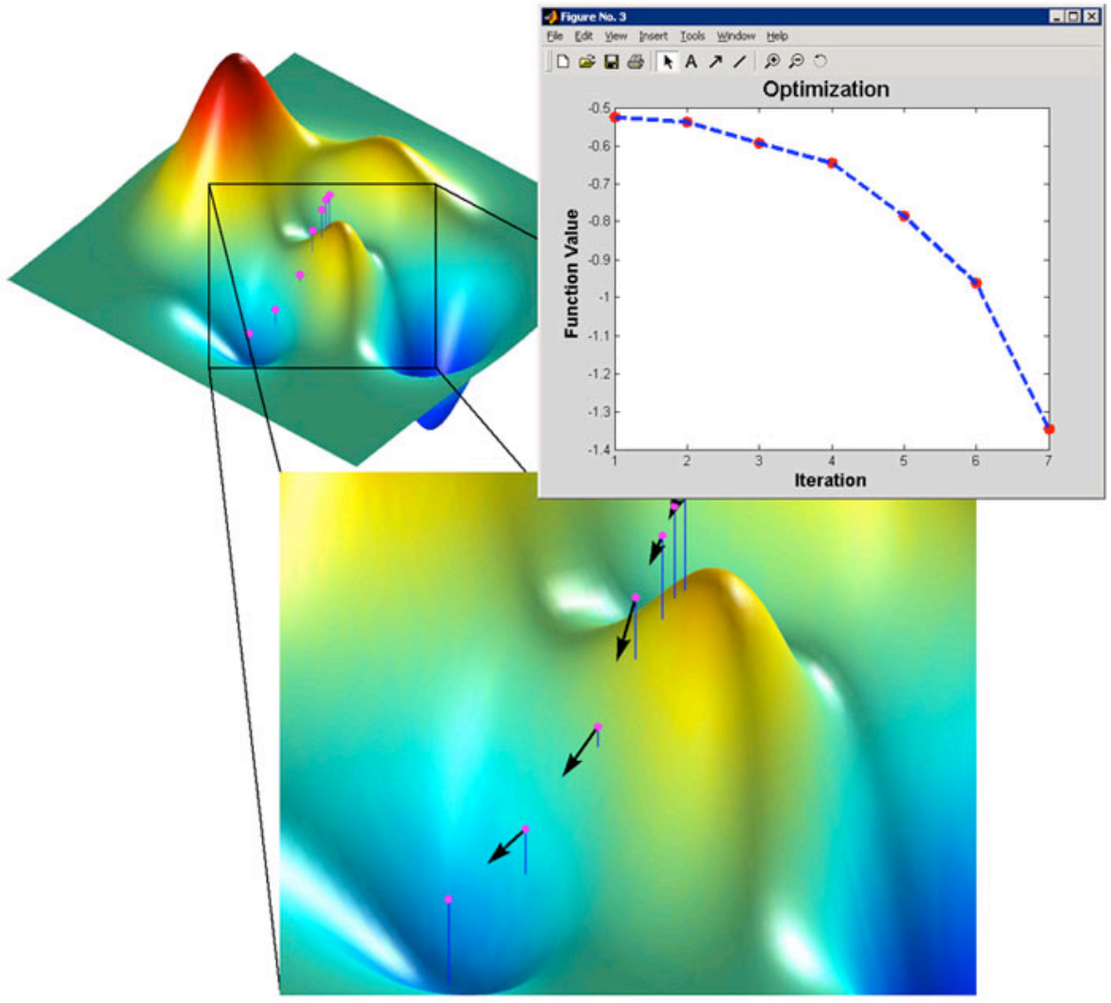
Optimization

(Minimization or Maximization)



Optimization

- The simplest case of optimization refers to the study of problems in which one seeks to minimize or maximize a real function by systematically choosing the values of real or integer variables from within an allowed set.
- This is actually a small subset of this area of applied mathematics and generalizes to study means of obtaining "best available" values of some objective function.

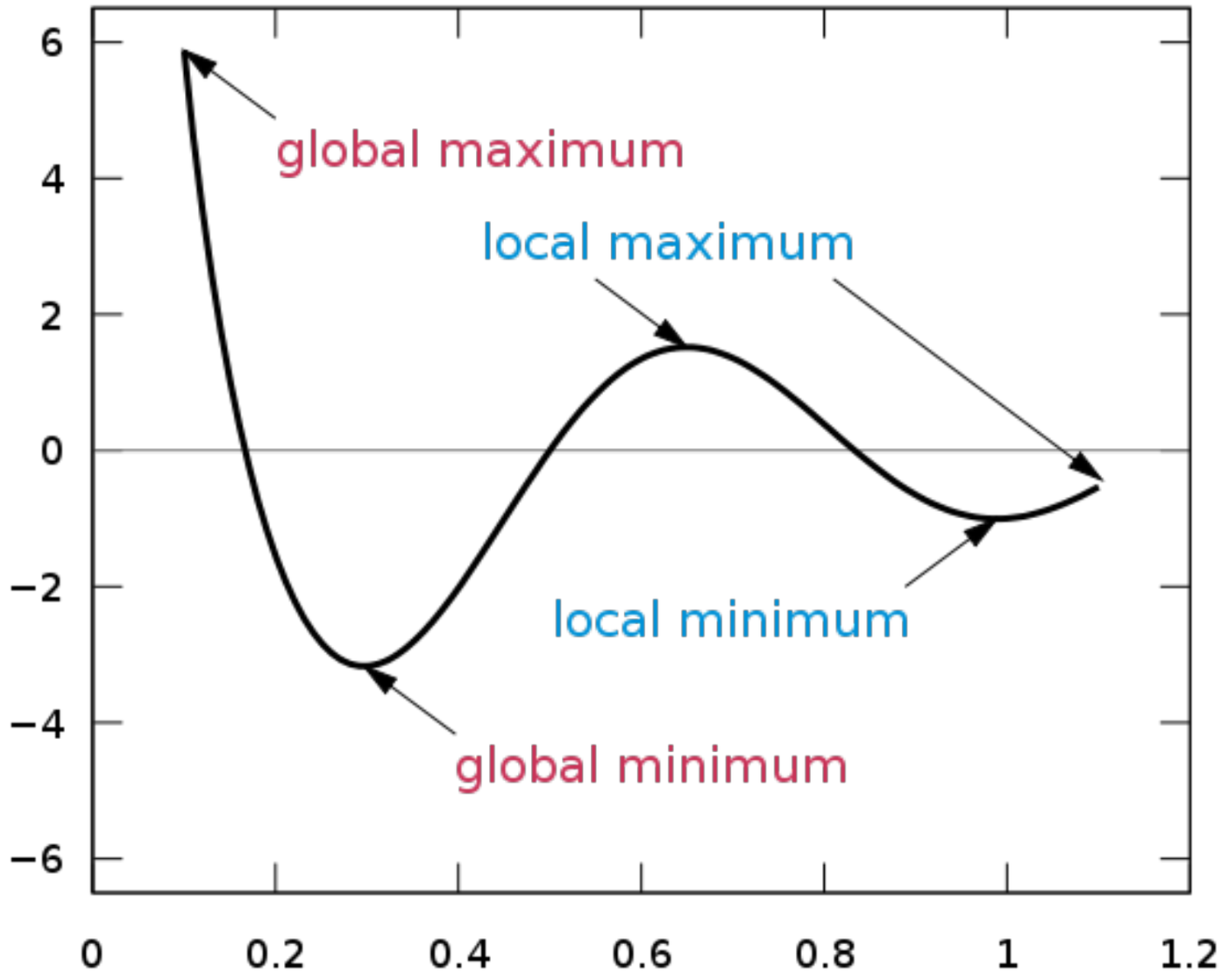
Minimization & Maximization

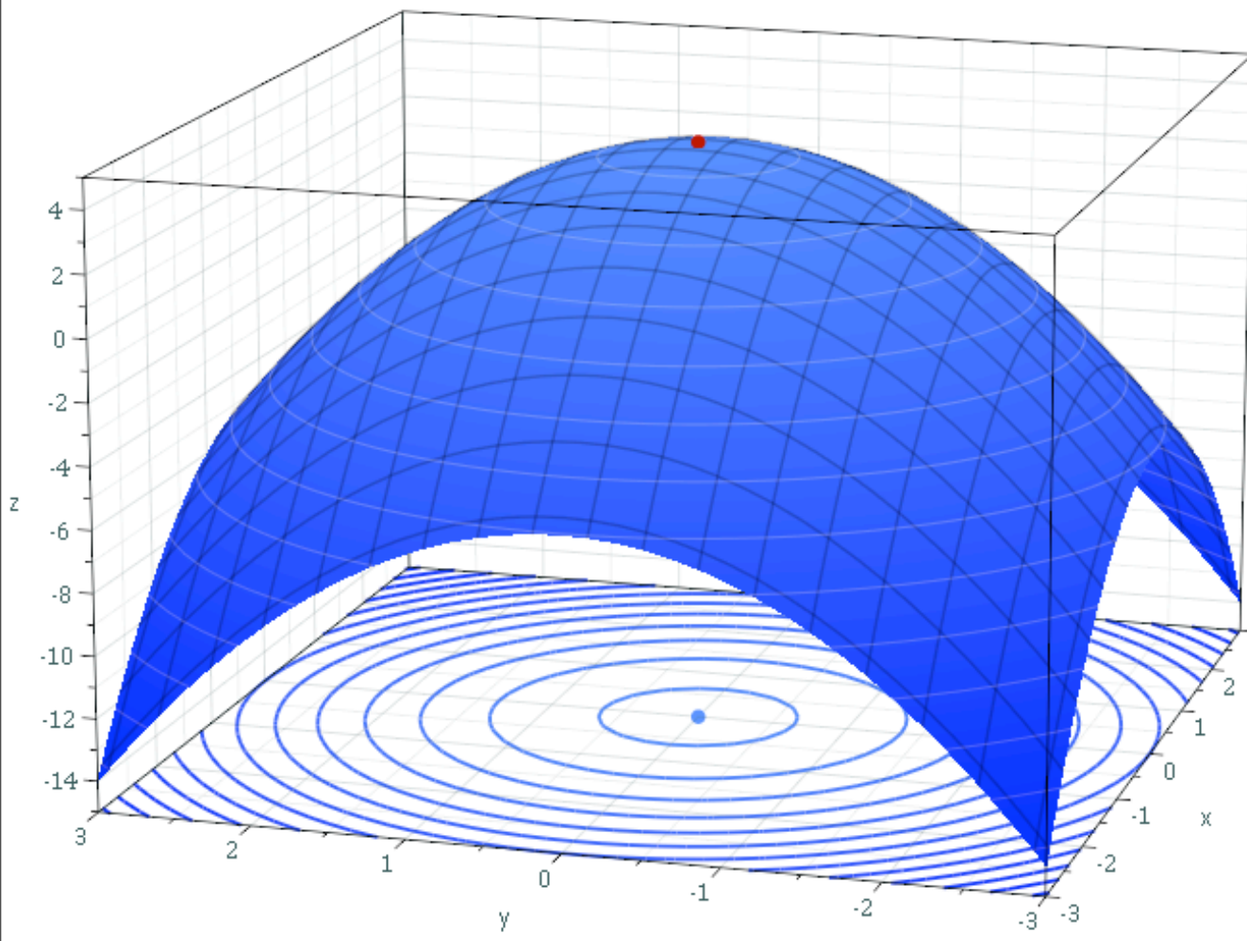
The tasks of minimization and maximization are trivially related to one another.

Maximizing a function *f* is like minimizing *-f*.

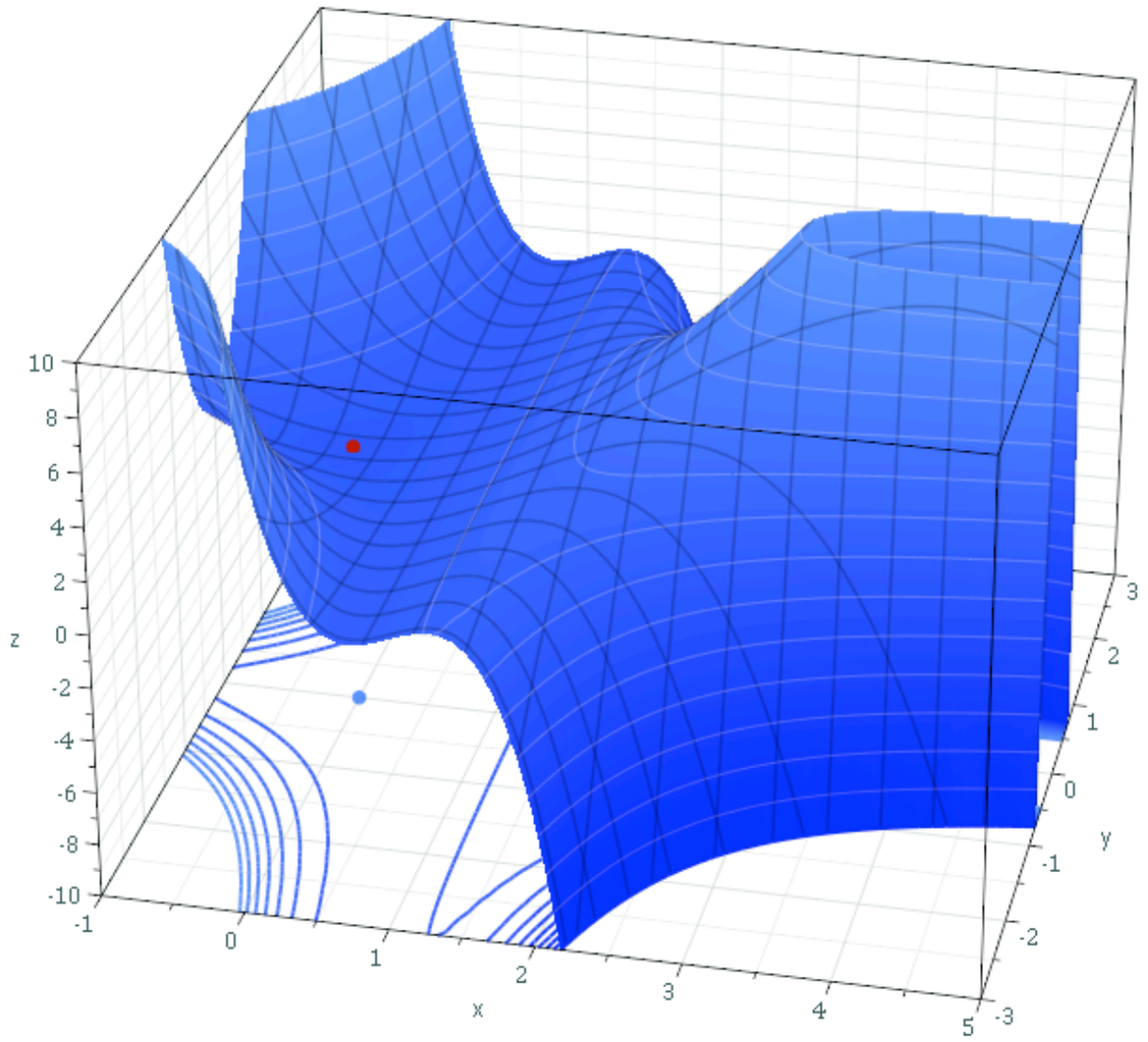
Extrema

- Maxima and minima, known collectively as extrema, are the largest value (maximum) or smallest value (minimum), that a function takes in a point either within a given neighbourhood (local extremum) or on the function domain in its entirety (global extremum).





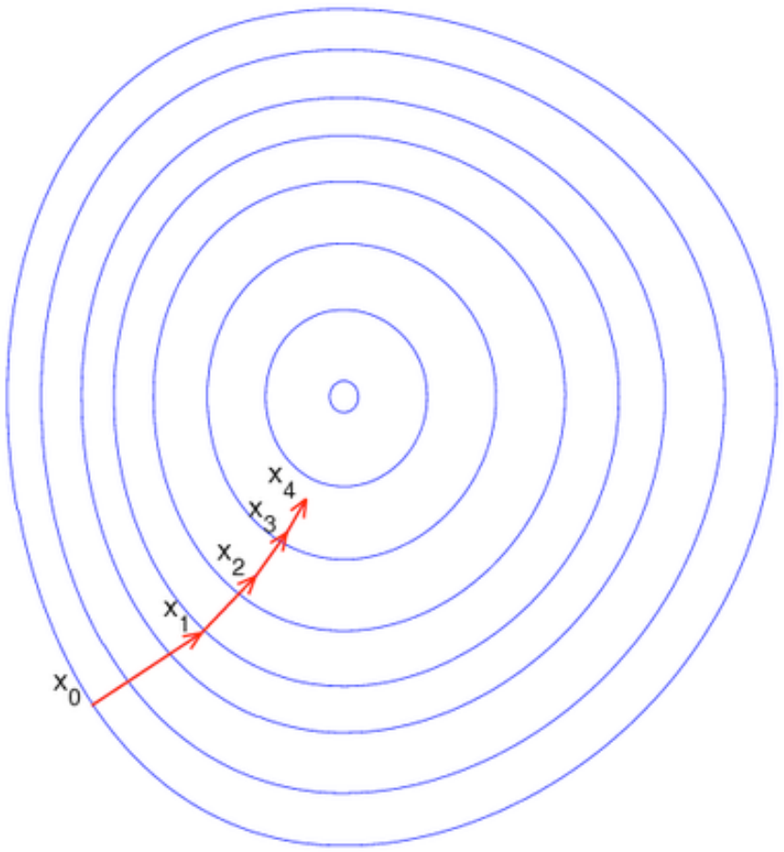
Global Maxima

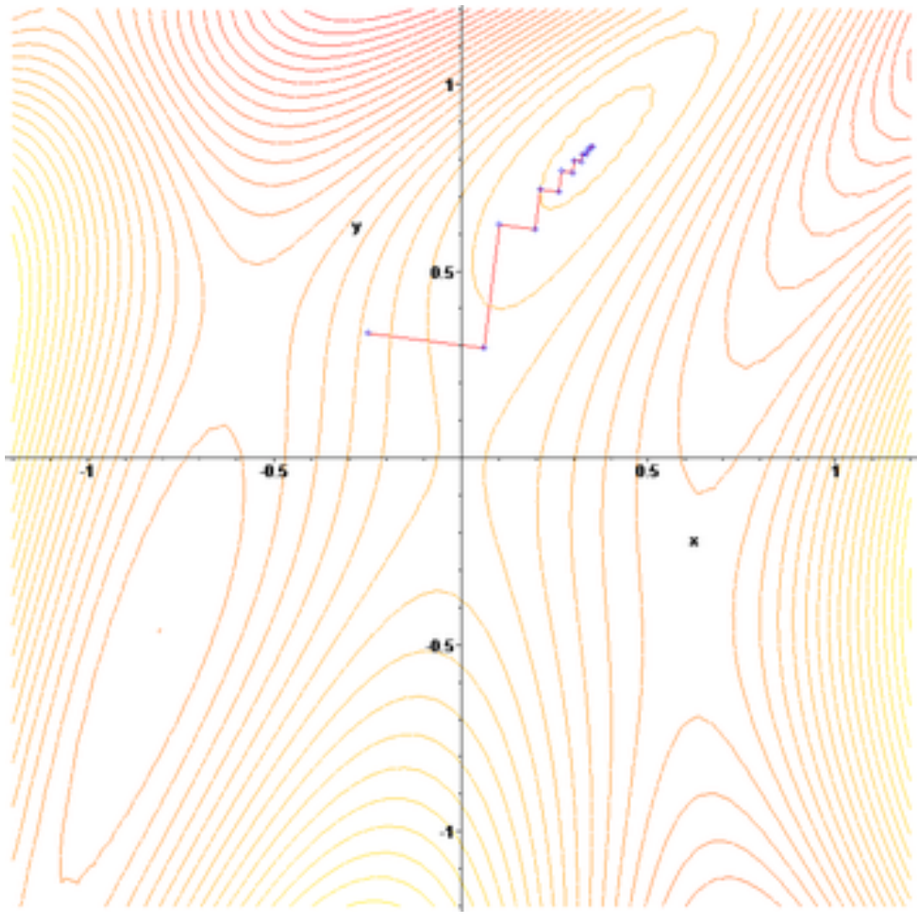


Local Minima

Steepest descent

The first optimization technique, which is known as steepest descent, goes back to Gauss.





Steepest descent

Gradient descent works in spaces of any number of dimensions, even in infinite-dimensional ones. In the latter case the search space is typically a function space, and one calculates the derivative of the functional to be minimized to determine the descent direction.

Two weaknesses of gradient descent are:

- The algorithm can take many iterations to converge towards a local minimum, if the curvature in different directions is very different.
- Finding the optimal step size can be time-consuming. Conversely, using a fixed step can yield poor results. Methods based on Newton's method and inversion of the Hessian using conjugate gradient techniques are often a better alternative.

Bracketing a Minima

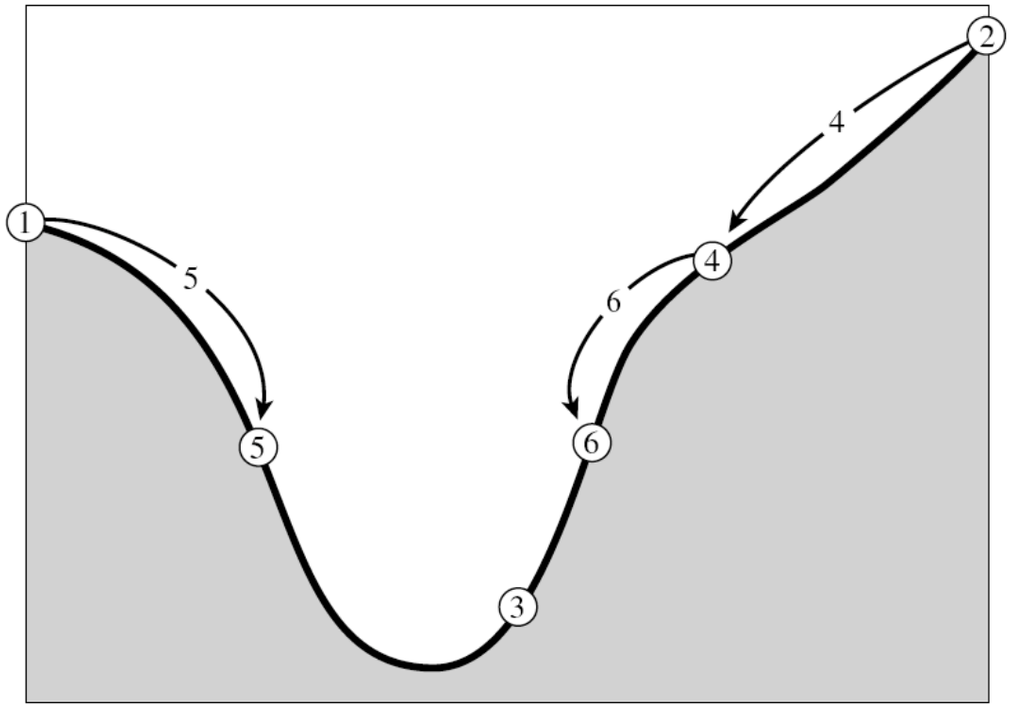
What does it mean to *bracket* a minimum? A root of a function is known to be bracketed by a pair of points, a and b, when the function has opposite sign at those two points. A minimum, by contrast, is known to be bracketed only when there is a *triplet* of points, $a < b < c$ (or $c < b < a$), such that $f(b)$ is less than both $f(a)$ and $f(c)$. In this case we know that the function (if it is smooth) has a minimum in the interval (a, c).

There is not much theory as to how to do this bracketing. Obviously you want to step downhill. But how far? We like to take larger and larger steps, starting with some (wild?) initial guess and then increasing the stepsize at each step either by a constant factor, or else by the result of a parabolic extrapolation of the preceding points that is designed to take us to the extrapolated turning point. It doesn't much matter if the steps get big. After all, we are stepping downhill, so we already have the left and middle points of the bracketing triplet. We just need to take a big enough step to stop the downhill trend and get a high third point.

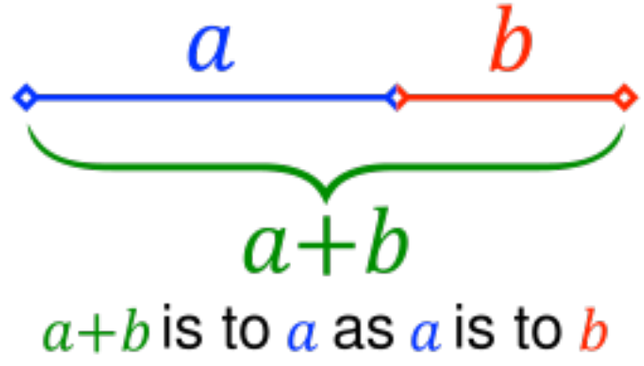
Golden section search

The golden section search is a technique for finding the extremum (minimum or maximum) of a unimodal function by successively narrowing the range of values inside which the extremum is known to exist.

The technique derives its name from the fact that the algorithm maintains the function values for triples of points whose distances form a golden ratio.



$$\frac{a+b}{a} = \frac{a}{b} = \varphi = \frac{1+\sqrt{5}}{2} \approx 1.6180339887\ldots$$

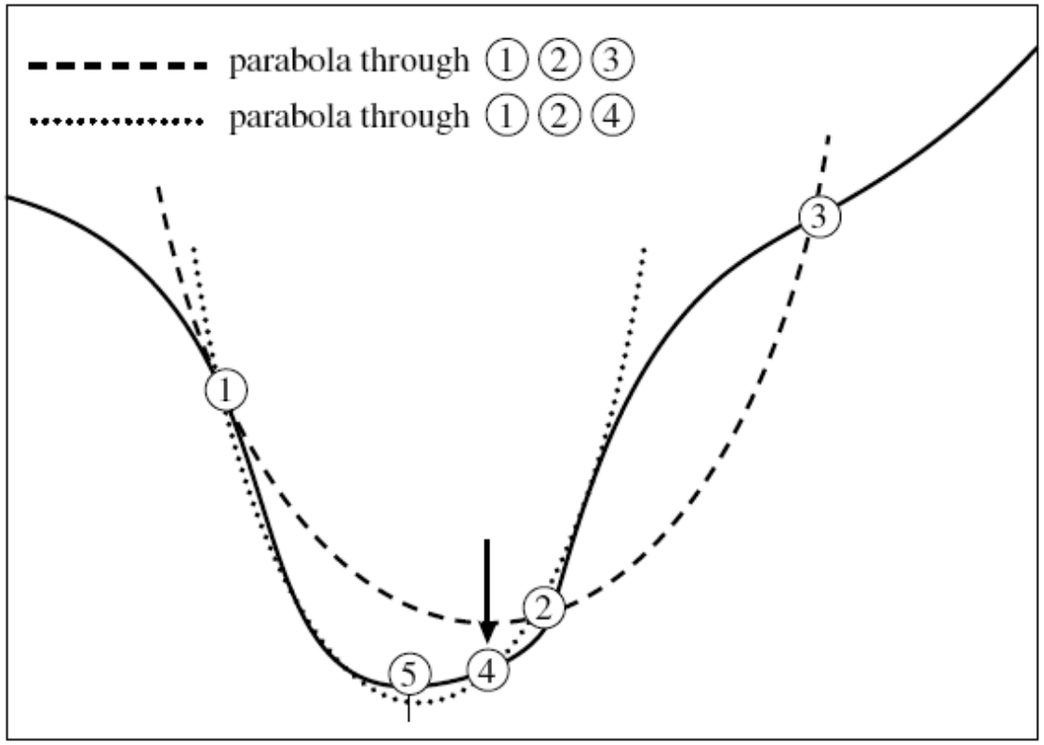


§10.2 in Numerical Recipes

Successive parabolic interpolation

The golden section search is designed to hunt down the worst possible cases. But if a function is well behaved then it is probably almost parabolic near the minima.

Successive parabolic interpolation is a technique for finding the extremum (minimum or maximum) of a continuous unimodal function by successively fitting parabolas (polynomials of degree two) to the function at three unique points, and at each iteration replacing the "oldest" point with the extremum of the fitted parabola.



In Brent's method the idea is to use the secant method or inverse quadratic interpolation if possible, because they converge faster, but to fall back to the more robust bisection method if necessary.

§10.3 in Numerical Recipes

Out of Class Reading Assignment

Numerical Recipes Chapter 10

There are four general categories of solvers:

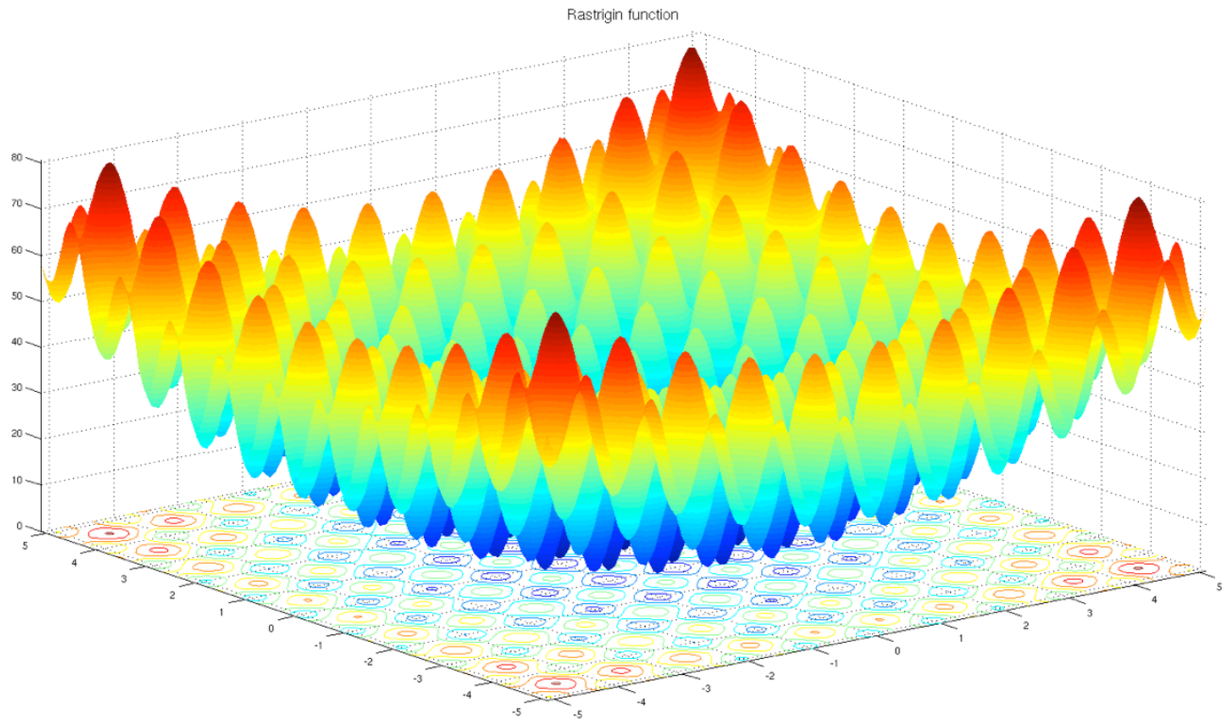
- **Minimizers.** This group of solvers attempts to find a local minimum of the objective function near a starting point x0. They address problems of unconstrained optimization, linear programming, quadratic programming, and general nonlinear programming.
- **Multiobjective minimizers.** This group of solvers attempts to either minimize the maximum value of a set of functions (fminimax), or to find a location where a collection of functions is below some prespecified values (fgoalattain).
- **Equation solvers.** This group of solvers attempts to find a solution to a scalar- or vector-valued nonlinear equation f(x) = 0 near a starting point x0. Equation-solving can be considered a form of optimization because it is equivalent to finding the minimum norm of f(x) near x0.
- **Least-Squares (curve-fitting) solvers**. This group of solvers attempts to minimize a sum of squares. This type of problem frequently arises in fitting a model to data. The solvers address problems of finding nonnegative solutions, bounded or linearly constrained solutions, and fitting parameterized nonlinear models to data.

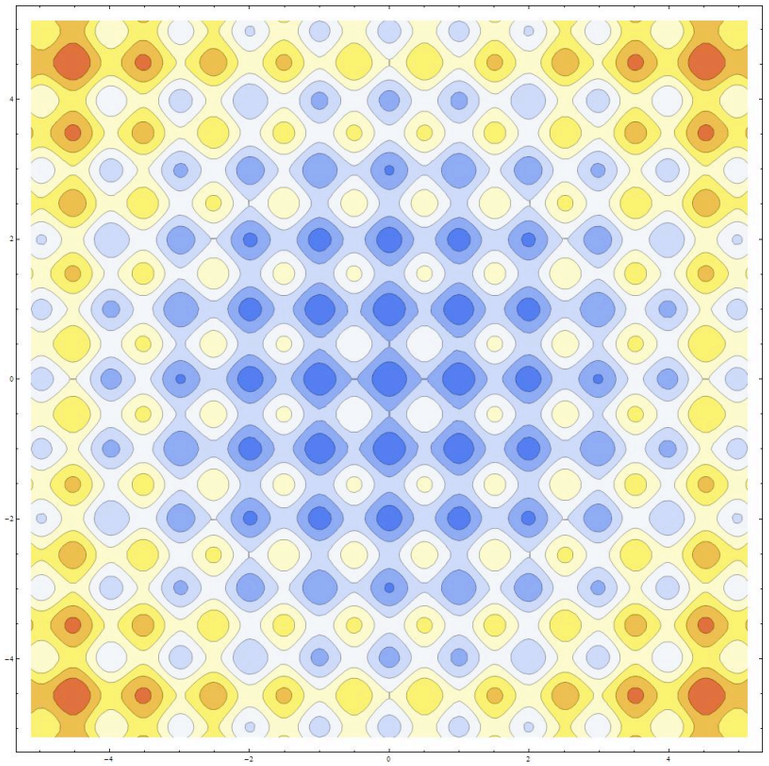
Example: The Rastrigin function

In mathematical optimization, the Rastrigin function is a non-convex function used as a performance test problem for optimization algorithms. It is a typical example of non-linear multimodal function. It was first proposed by Rastrigin. This function is a fairly difficult problem due to its large search space and its large number of local minima.

It is defined by:

$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i) \right]$$

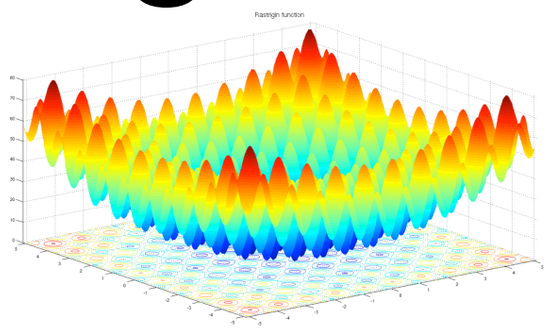


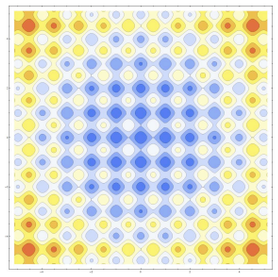


go through example `optimizationexample.m` which compares the solution of four algorithms to the optimization of the Rastrigin function. As you go through look up the help on each function

Example: The Rastrigin function

$$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i) \right]$$





Results	fminunc	patternsearch	ga	GlobalSearch
solution	[19.9 29.9]	[19.9 -9.9]	[9.9 0]	**[0 0]**
objective	12.9	5	1	**0**
# Fevals	15	174	1040	**2241**

One solution is better than another if its objective function value is smaller than the other.
The following table summarizes the results, accurate to one decimal.

Example: The Rastrigin function

- **fminunc** quickly reaches the local solution within its starting basin, but does not explore outside this basin at all. fminunc has simple calling syntax.
- **patternsearch** takes more function evaluations than fminunc, and searches through several basins, arriving at a better solution than fminunc. patternsearch calling syntax is the same as that of fminunc.
- **ga** (genetic algorithm) takes many more function evaluations than patternsearch. By chance it arrived at a better solution. ga is stochastic, so its results change with every run. ga has simple calling syntax, but there are extra steps to have an initial population near [20,30].
- **GlobalSearch** run takes more function evaluations than ga, searches many basins, and arrives at an even better solution. In this case, GlobalSearch found the global optimum. Setting up GlobalSearch is more involved than setting up the other solvers. As the example shows, before calling GlobalSearch, you must create both a GlobalSearch object (gs in the example), and a problem structure (problem). Then, call the run method with gs and problem.

Class Assignment

Add to the example `optimizationexample.m` the location of the solutions from the four solvers

Then go through the help and do the examples for the following four solvers:

- fminunc
- patternsearch
- ga (genetic algorithm)
- GlobalSearch

Examples

Use this list to find examples in the documentation.

Constrained Nonlinear Examples

Nonlinear Constrained Minimization
An Example Using All Types of Constraints
Optimization Tool with the fmincon Solver
Nonlinear Inequality Constraints
Bound Constraints
Constraints With Gradients
Constrained Minimization Using fmincon's Interior-Point Algorithm With Analytic Hessian
Equality and Inequality Constraints
Nonlinear Minimization with Bound Constraints and Banded Preconditioner
Nonlinear Minimization with Equality Constraints
Nonlinear Minimization with a Dense but Structured Hessian and Equality Constraints
Using Symbolic Math Toolbox Functions to Calculate Gradients and Hessians
One-Dimensional Semi-Infinite Constraints
Two-Dimensional Semi-Infinite Constraint
Example Using ktrlink

Least Squares Examples

Optimization Tool with the lsqlin Solver
Using lsqnonlin With a Simulink Model
Nonlinear Least-Squares with Full Jacobian Sparsity Pattern
Linear Least-Squares with Bound Constraints
Jacobian Multiply Function with Linear Least Squares
Nonlinear Curve Fitting with lsqcurvefit

Unconstrained Nonlinear Examples

fminunc Unconstrained Minimization
Nonlinear Minimization with Gradient and Hessian
Nonlinear Minimization with Gradient and Hessian Sparsity Pattern

Linear Programming Examples

Linear Programming with Equalities and Inequalities
Linear Programming with Dense Columns in the Equalities