Deterministic minimisation algorithms Free and with inequality restrictions: penalty and barrier methods

Lluís Alsedà

Centre de Recerca Matemàtica

Departament de Matemàtiques
Universitat Autònoma de Barcelona

http://www.mat.uab.cat/∼alseda

November 14 & 15, 2023

BGSMath Course:

Fitting data with dynamical models:
ten lessons on mathematical field work

Outline

Introduction

Deterministic optimization methods form a crucial category of mathematical techniques employed to find the optimal solution to a given problem with well-defined parameters and constraints. In contrast to stochastic optimization methods, which incorporate randomness and uncertainty into the decision-making process, deterministic optimization focuses on solving problems where all the variables and parameters are precisely known.

The primary objective of deterministic optimization is to identify the best possible solution from a finite set of feasible alternatives, taking into account specific constraints and objectives.

Introduction (II)

These methods are widely utilized in various fields, including engineering, operations research, economics, finance, and management, where making informed and efficient decisions is paramount.

The fundamental principle underlying deterministic optimization is to mathematically model a real-world problem, incorporating relevant constraints and defining an objective function that needs to be either maximized or minimized. The challenge then lies in navigating through the solution space to pinpoint the optimal values for the decision variables.

Introduction (III)

Common deterministic optimization methods include linear programming, integer programming, nonlinear programming, and convex optimization. Linear programming, for instance, deals with linear relationships among variables and is extensively applied in resource allocation, production planning, and transportation logistics. Integer programming extends this approach to problems where decision variables are restricted to integer values, often encountered in project scheduling and network design.

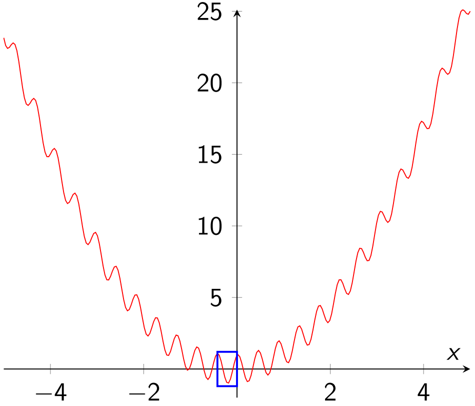
Nonlinear programming methods are employed when the relationships among variables are not strictly linear, allowing for more intricate models. Convex optimization, on the other hand, addresses problems where the objective function and constraints are convex, enabling the application of efficient algorithms to find optimal solutions.

Introduction (IV)

Deterministic optimization methods play a pivotal role in addressing complex decision-making challenges, contributing to enhanced efficiency, resource utilization, and overall system performance. As technology advances and computational capabilities grow, these methods continue to evolve, offering powerful tools for tackling a wide array of real-world problems.

Introduction — A word of caution

The graph of the "easy" function $x(x+0.2)+cos(14.5x-0.3)$



Remark

Unless we know that the minimum is contained in the "valley" $x \in [-0.41, 0]$ we will not find it with deterministic methods.

Unconstrained minimization for non-differentiable functions: A direct search method

The recommended algorithm is the Nelder–Mead method (also downhill simplex method, amoeba method, or polytope method). It is a direct search method (based on function comparison) and is often applied to nonlinear optimization problems for which derivatives may not be known.

See

https://en.wikipedia.org/wiki/Nelder-Mead_method

Descent Methods for differentiable functions

For this part we follow:

Numerical mathematics, Alfio Quarteroni, Riccardo Sacco, Fausto Saleri, Texts in Applied Mathematics **37**, Springer Science, 2006.

An informal description of descent methods

is an iterative procedure that can be formulated as follows: given an initial vector $x^{(0)} \in \mathbb{R}^n$, compute for $k \geq 0$ until convergence

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)},$$

where $d^{(k)}$ is a suitably chosen direction and α_k is a positive parameter called *stepsize* that measures the step along the direction $d^{(k)}$.

Descent Methods for differentiable functions

Descent Direction

The direction $d^{(k)}$ is a *descent direction* if

$$\begin{cases} d^{(k)\intercal} \nabla f\left(x^{(k)}\right) < 0 & \text{if } \nabla f\left(x^{(k)}\right) \neq 0, \\ d^{(k)} = 0 & \text{if } \nabla f\left(x^{(k)}\right) = 0. \end{cases}$$

Descent Method

A *descent method* is a method $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$, in which the vectors $d^{(k)}$ are descent directions.

Descent Methods for differentiable functions

From Taylor Theorem we know that

$$f\left(x^{(k)} + \alpha_k d^{(k)}\right) = f\left(x^{(k)}\right) + \alpha_k \nabla f\left(x^{(k)}\right)^{\mathsf{T}} d^{(k)} + \varepsilon,$$

where ε tends to zero as α_k tends to zero. As a consequence, if $d^{(k)}$ is a descent direction and $\alpha_k > 0$ is sufficiently small,

$$f\left(x^{(k)} + \alpha_k d^{(k)}\right) < f\left(x^{(k)}\right),$$

which justifies the name of a *descent direction*.

Notice that different choices of descent directions $d^{(k)}$ correspond to different methods.

An elementary list of descent methods

Newton's methods

$$d^{(k)} = -H^{-1}\left(x^{(k)}\right)\nabla f\left(x^{(k)}\right),$$

provided that $H(\cdot)$ is positive definite within a sufficiently large neighborhood of an optimal point.

Inexact Newton's methods

$$d^{(k)} = -B_k^{-1}\nabla f\left(x^{(k)}\right),$$

where B_k is a suitable approximation of $H\left(x^{(k)}\right)$ (much in the spirit of Broyden Method).

An elementary list of descent methods (II)

The *gradient* or *steepest descent method*

$$d^{(k)} = -\nabla f\left(x^{(k)}\right).$$

This method is thus an inexact Newton's method, in which $B_k = \mathrm{Id}$ for every k. It can also be regarded as a gradient-like method, since $d^{(k)\mathsf{T}} \nabla f\left(x^{(k)}\right) = -\left\|\nabla f\left(x^{(k)}\right)\right\|_2^2$.

An elementary list of descent methods (III)

The *conjugate gradient method*

$$d^{(k)} = -\nabla f\left(x^{(k)}\right) + \beta_k d^{(k-1)},$$

where β_k is a scalar to be suitably selected in such a way that the directions $d^{(k)}$ turn out to be mutually orthogonal with respect to a suitable scalar product.
For instance, if we consider the standard scalar product,

$$0 = d^{(k)\intercal} d^{(k-1)} = -\nabla f\left(x^{(k)}\right)^{\intercal} d^{(k-1)} + \beta_k d^{(k-1)\intercal} d^{(k-1)}.$$

Hence,

$$\beta_k = \frac{\nabla f\left(x^{(k)}\right)^{\intercal} d^{(k-1)}}{\left\|d^{(k-1)}\right\|_2^2}.$$

On the goodness of the stepsize

Selecting $d^{(k)}$ is not enough to completely identify a descent method, since it remains an open problem how to determine α_k in such a way that $f\left(x^{(k)} + \alpha_k d^{(k)}\right) < f\left(x^{(k)}\right)$ is fulfilled without resorting to excessively small stepsizes (and, thus, to methods with a slow convergence).

A method for computing α_k consists of solving the following minimization problem in one dimension:

(1) find α such that $\varphi(\alpha) = f\left(x^{(k)} + \alpha d^{(k)}\right)$ is minimized.

In such a case we have the following result:

On the goodness of the stepsize (II)

Theorem

Consider a descent method. If at the generic step k, the parameter α_k is such that $\varphi(\alpha_k)$ is minimum, then the following orthogonality property holds:

$$\nabla f\left(x^{(k+1)}\right)^{\mathsf{T}} d^{(k)} = 0.$$

Unfortunately, except for some very special but relevant cases, providing an exact solution of (1) is not feasible, since this is a nonlinear problem. One possible strategy consists of approximating f along the straight line $x^{(k)} + \alpha d^{(k)}$ through an interpolating polynomial and then minimizing this polynomial (see the quadratic interpolation and cubic interpolation methods).

Generally speaking, a process that leads to an approximate solution to (1) is said to be a *line search technique*.

Line Search Techniques

Practical experience reveals that it is not necessary to solve (1) accurately in order to devise efficient methods. Rather, it is crucial to enforce some limitation on the admissible values for α_k.

The weakest requirement for the stepsizes α_k is that the new iterates $x^{(k+1)}$ satisfy the inequality

$$f\left(x^{(k)} + \alpha_k d^{(k)}\right) = f\left(x^{(k+1)}\right) < f\left(x^{(k)}\right)$$

for fixed $x^{(k)}$ and $d^{(k)}$.

For this purpose, the procedure based on starting from a (sufficiently large) value of the step length α_k and halve this value until the above inequality is fulfilled, can yield completely wrong results because two kinds of difficulties may arise: a slow descent rate of the sequence and the use of small stepsizes.

The recommended line search techniques put special emphasis in overcoming these two difficulties/limitations.

Line Search Techniques (II)

Fighting with slow descent rates

From Taylor Theorem we know that

$$\nabla f\left(x^{(k)}\right)^{\intercal} d^{(k)} \approx \frac{1}{\alpha}\left[f\left(x^{(k)}+\alpha d^{(k)}\right)-f\left(x^{(k)}\right)\right].$$

Hence, $\nabla f\left(x^{(k)}\right)^{\intercal} d^{(k)}$ can be called the *(theoretical) average descent rate of f at the point $x^{(k)}$ along the direction $d^{(k)}$*.

To avoid slow descent rates, we require that the (real) average descent rate of f at the point $x^{(k)}$ (when moving to a close candidate $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$), must be smaller than or equals to a given fraction $\sigma \in \left(0, \frac{1}{2}\right)$, of the (theoretical) average descent rate of f at $x^{(k)}$ along the direction $d^{(k)}$:

Anti slow descent rates condition

$$\frac{1}{\alpha_k}\left[f\left(x^{(k)}+\alpha_k d^{(k)}\right)-f\left(x^{(k)}\right)\right] \leq \sigma \nabla f\left(x^{(k)}\right)^{\intercal} d^{(k)} < 0$$

Line Search Techniques (III)

Fighting with small stepsizes

To avoid small stepsizes, we require that the (theoretical) average descent rate of f at $x^{(k)} + \alpha_k d^{(k)}$ along the direction $d^{(k)}$, must be smaller than or equals to a given fraction $\beta \in (\sigma, 1)$, of the (theoretical) average descent rate of f at $x^{(k)}$ along the direction $d^{(k)}$:

Anti small stepsizes condition

$$\left|\nabla f\left(x^{(k)} + \alpha_k d^{(k)}\right)^{\mathsf{T}} d^{(k)}\right| \leq \beta \left|\nabla f\left(x^{(k)}\right)^{\mathsf{T}} d^{(k)}\right|$$

Usual computational choices of σ and β are, $\sigma \in \left[10^{-5}, 10^{-1}\right]$, and $\beta \in \left[10^{-1}, \frac{1}{2}\right]$.

Line Search Techniques (IV)

On the compatibility of the above conditions

Property

Assume that $f(x) \geq M$ for any $x \in \mathbb{R}^n$. Then, for the descent method, there exists an interval $\mathscr{K} = [a, B]$ with $0 < a < B$, such that for every $\alpha_k \in \mathscr{K}$, the *anti slow descent rates condition* and the *antismall stepsizes condition* are satisfied, for some $\sigma \in \left(0, \frac{1}{2}\right)$ and $\beta \in (\sigma, 1)$.

This property ensures that, under suitable assumptions, it is possible to find out values of α_k which satisfy both the *anti slow descent rates condition* and the *antismall stepsizes condition*.

Line Search Techniques (V)

Among the most up-to-date strategies to find appropriate stepsizes α_k, we recall here the

Backtracking techniques

Fix $\sigma \in \left(0, \frac{1}{2}\right)$ and the starting value $\alpha_k = 1$. Then keep on reducing its value by a suitable scale factor $\rho \in (0, 1)$ (*backtrack step*) until the *anti slow descent rates condition* is satisfied.

Other commonly used strategies (slightly more involved) are those developed by Armijo and Goldstein.

Next we show the

$$\textbf{procedure}\ \textsc{BackTrack}\left(\sigma, \rho, x^{(k)}, d^{(k)}, \boldsymbol{f}, \boldsymbol{\nabla f}\right)$$

pseudocode, which requires σ (usually of the order of 10^{-4}), the scale factor ρ and the vectors $x^{(k)}$ and $d^{(k)}$ as input parameters, and references to the function $\boldsymbol{f}$ and to the procedure $\boldsymbol{\nabla f}$ which computes the gradient of f. In output, the code returns a suitable value of α_k.

Line Search Techniques (V)

Algorithm Bactracking for line search

```
procedure $\textsc{BackTrack}(\sigma, \rho, x^{(k)}, d^{(k)}, \boldsymbol{f}, \boldsymbol{\nabla f})$
    $\mathrm{fx}_k \leftarrow \boldsymbol{f}(x^{(k)})$                                  ▷ Initialization
    $\mathrm{AvDRx}_k \leftarrow \sigma\, \boldsymbol{\nabla f}(x^{(k)})^{\mathsf{T}} d^{(k)}$
    $\alpha \leftarrow 1$
    $\mathrm{fx}_{\mathrm{new}} \leftarrow \boldsymbol{f}\left(x^{(k)} + \alpha\, d^{(k)}\right)$
    while $\mathrm{fx}_{\mathrm{new}} > \mathrm{fx}_k + \alpha \cdot \mathrm{AvDRxk}$ do        ▷ Main $\alpha_k$ reducing loop
        $\alpha \leftarrow \rho \cdot \alpha$
        $\mathrm{fx}_{\mathrm{new}} \leftarrow \boldsymbol{f}\left(x^{(k)} + \alpha\, d^{(k)}\right)$
    end while
    return $\alpha$
end procedure
```

Descent Methods for Quadratic and Quadratic-like Functions

The problem of minimizing a

Quadratic Function

$$f(x) = \frac{1}{2}x^{\intercal}Ax - b^{\intercal}x,$$

where A is an $n \times n$ real symmetric positive definite matrix and $b \in \mathbb{R}^n$ is of remarkable interest because the parameter α_k can be exactly computed.

Analogously, Quadratic-like Functions, have a unique global minimum and the parameter α_k can be well approximated by using quadratic estimates.

Descent Methods for Quadratic Functions

It can be checked that f is a quadratic function,

$$\nabla f(x) = Ax - b \quad \text{and} \quad H(x) = A.$$

Thus, a necessary condition for $x^\star$ to be a minimizer for f is that $x^\star$ is the solution of the linear system

$$A x^\star = \nabla f\left(x^\star\right) + b = b.$$

Given a fixed a descent direction $d^{(k)}$, we can determine the optimal value of the acceleration parameter α_k by solving (1). That is, by finding the point where the function f, restricted to the direction $d^{(k)}$ at the point $x^{(k)}$ is minimized.

Descent Methods for Quadratic Functions

Determination of α_k

This can be computed simply by setting to zero the derivative with respect to α_k :

$$0 = \frac{\mathrm{d}}{\mathrm{d}\alpha_k} f\left(x^{(k)} + \alpha_k d^{(k)}\right) = d^{(k)\mathsf{T}} \nabla f\left(x^{(k)} + \alpha_k d^{(k)}\right) =$$

$$d^{(k)\mathsf{T}}\left(A x^{(k)} + \alpha_k A d^{(k)} - b\right) =$$

$$d^{(k)\mathsf{T}}\left(A x^{(k)} - b\right) + \alpha_k d^{(k)\mathsf{T}} A d^{(k)}.$$

Equivalently,

$$\alpha_k = \frac{d^{(k)\mathsf{T}}\left(b - A x^{(k)}\right)}{d^{(k)\mathsf{T}} A d^{(k)}}.$$

Descent Methods for Quadratic Functions

On the explicit error formulae of the iterative method

$$\left\|x^{(k+1)} - x^\star\right\|_A^2 := \left(x^{(k+1)} - x^\star\right)^{\mathsf{T}} A \left(x^{(k+1)} - x^\star\right) =$$

$$\left\|x^{(k)} - x^\star\right\|_A^2 + 2\alpha_k d^{(k)\mathsf{T}} A \left(x^{(k)} - x^\star\right) + \alpha_k^2 d^{(k)\mathsf{T}} A\, d^{(k)} =$$

$$\left(1 - \sigma_k\right) \left\|x^{(k)} - x^\star\right\|_A^2,$$

with

$$\sigma_k = \frac{\left(d^{(k)\mathsf{T}} r^{(k)}\right)^2}{d^{(k)\mathsf{T}} A\, d^{(k)} r^{(k)\mathsf{T}} A^{-1} r^{(k)}},$$

and $r^{(k)} := b - A\,x^{(k)} = -\nabla f\left(x^{(k)}\right)$.

Since A is symmetric and positive definite, σ_k is always positive. Moreover, it can be directly checked that $1 - \sigma_k$ is strictly less than 1, except when $d^{(k)}$ is orthogonal to $r^{(k)}$ (in which case $\sigma_k = 0$).

The choice $d^{(k)} = r^{(k)}$, which leads to the steepest descent method, prevents this last circumstance from arising.

Descent Methods for Quadratic Functions

On the explicit error formulae of the iterative method

Kantorovich Theorem

Let A be an $n \times n$ real symmetric positive definite matrix whose eigenvalues with largest and smallest module are given by $\lambda_{\max}$ and $\lambda_{\min}$, respectively. Then, for every $y \in \mathbb{R}^n$, $y \neq 0$,

$$\frac{\left(y^{\intercal} y\right)^2}{\left(y^{\intercal} A\, y\right)\left(y^{\intercal} A^{-1} y\right)} \geq \frac{4\lambda_{\max}\lambda_{\min}}{\left(\lambda_{\max}+\lambda_{\min}\right)^2}$$

From this theorem we deduce the following error formula for the steepest descent method for quadratic functions (that corresponds to the choice $d^{(k)} = -\nabla f\left(x^{(k)}\right) = b - A\,x^{(k)} = r^{(k)}$):

$$\left\|x^{(k+1)} - x^{\star}\right\|_A^2 \leq \frac{\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}} \left\|x^{(k)} - x^{\star}\right\|_A^2.$$

Then, if A is ill-conditioned ($\lambda_{\min} \approx 0$), the error reducing factor for the steepest descent method is close to 1, yielding a slow convergence to the minimizer $x^{\star}$. This drawback can be overcome by introducing directions $d^{(k)}$ that are *mutually A-conjugate*.

Descent Methods for Quadratic Functions

Mutually A-conjugate directions for ill conditioned matrices

Mutually A-conjugate directions

The directions $\left\{d^{(k)}\right\}_k$ are said to be *mutually A-conjugate* if and only if

$$d^{(k)\mathsf{T}} A\, d^{(m)} = 0 \quad \text{if } k \neq m.$$

On the mutually A-conjugate directions properties. Why to use them . . .

A method for computing the minimizer $x^\star$ of a quadratic function which employs mutually A-conjugate directions terminates after at most n steps if the acceleration parameter α_k is selected as in Slide 23. Moreover, for any k, $x^{(k+1)}$ is the minimizer of f over the subspace generated by the vectors $x^{(0)}, d^{(0)}, d^{(1)}, \dots, d^{(k)}$, and

$$r^{(k+1)\mathsf{T}} d^{(m)} = 0 \quad \text{for every} \quad m \leq k.$$

Descent Methods for Quadratic Functions

On the computation of mutually A-conjugate directions for ill conditioned matrices

The A-conjugate directions can be determined by following the same strategy as in the *conjugate gradient method*:

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}, \text{ and}$$
$$d^{(k+1)} = r^{(k)} + \beta_k d^{(k)}, \text{ with}$$
$$\beta_k = -\frac{r^{(k+1)\mathsf{T}} A\, d^{(k)}}{d^{(k)\mathsf{T}} A\, d^{(k)}} = \frac{r^{(k+1)\mathsf{T}} r^{(k+1)}}{r^{(k)\mathsf{T}} r^{(k)}}$$

where, at initialization, we set $d^{(0)} = r^{(0)}$.

Descent Methods for Quadratic Functions

On the error of iterative methods employing mutually A-conjugate directions

An iterative method for quadratic functions which employs mutually A-conjugate directions, satisfies the following error estimate:

$$\left\|x^{(k)} - x^{\star}\right\|_A \leq 2 \left(\frac{\sqrt{K_2(A)} - 1}{\sqrt{K_2(A)} + 1} \right)^k \left\|x^{(0)} - x^{\star}\right\|_A,$$

where[1]

$$K_2(A) := \|A\|_2 \, \|A^{-1}\|_2 = \frac{\lambda_{\max}}{\lambda_{\min}} = \rho(A)\rho\left(A^{-1}\right)$$

denotes the *condition number of A* in the 2-norm, and $\rho(\cdot)$ denotes the *spectral radius*.

[1]The last two identities are valid only for symmetric positive definite matrices.

Penalty and Barrier methods

The idea is to approximate a constrained optimization problem with a sequence of unconstrained ones whose solutions approximate the solution of the constrained problem.

In the case of penalty methods the approximation is obtained by adding a term to the objective function that highly penalizes the violation of constraints.

Barrier methods search inside the feasible region instead of near the boundary.

To be more specific, consider the problem

$$\begin{aligned} &\min \mathrm{F}(\vec{\mathbf{x}}) \\ \text{subject to } &g_i(\vec{\mathbf{x}}) \leq 0 \quad \text{for } i = 1, 2, \dots, \ell. \end{aligned}$$

where F and the g_i's are continuous functions from $\mathbb{R}^n$ to $\mathbb{R}$.

Penalty methods

The idea of a penalty function method is to replace the above problem by an unconstrained problem of the form

$$\text{Minimize } \mathscr{F}_\beta(\vec{\mathbf{x}}) := \mathrm{F}(\vec{\mathbf{x}}) + \beta\,\mathscr{P}(\vec{\mathbf{x}})$$

where $\beta > 0$ is the *penalty parameter*, and $\mathscr{P}$ is a continuous non-negative function from $\mathbb{R}^n$ to $\mathbb{R}$ such that $\mathscr{P}(\vec{\mathbf{x}}) = 0$ if and only if $g_i(\vec{\mathbf{x}}) \leq 0$ for every $i = 1, 2, \ldots, \ell$.

Example

$$\mathscr{P}(\vec{\mathbf{x}}) = \frac{1}{2}\sum_{i=1}^{\ell}\left(\max\left\{0, g_i(\vec{\mathbf{x}})\right\}\right)^2$$

Observe that each unsatisfied constraint influences $\mathscr{P}$ by adding a penalty equal to the square of the violation.

In what follows we denote the minimum of $\mathscr{F}_\beta$ by $\vec{\mathbf{x}}^\beta$, if it exists.

Lemma 1

Let $\widetilde{\beta} > \beta$. Then the following statements hold:

1. $\mathscr{F}_\beta(\vec{\mathbf{x}}^\beta) \leq \mathscr{F}_{\widetilde{\beta}}(\vec{\mathbf{x}}^{\widetilde{\beta}})$.
2. $\mathscr{P}(\vec{\mathbf{x}}^\beta) \geq \mathscr{P}(\vec{\mathbf{x}}^{\widetilde{\beta}})$.
3. $\mathsf{F}(\vec{\mathbf{x}}^\beta) \leq \mathsf{F}(\vec{\mathbf{x}}^{\widetilde{\beta}})$.

Lemma 2

Let $\vec{\mathbf{x}}^\star$ be a solution of

$$\begin{aligned} &\min \mathsf{F}(\vec{\mathbf{x}}) \\ &\text{subject to } g_i(\vec{\mathbf{x}}) \leq 0 \quad \text{for } i = 1, 2, \ldots, \ell. \end{aligned}$$

Then, for every penalty parameter β,

$$\mathsf{F}(\vec{\mathbf{x}}^\star) \geq \mathscr{F}_\beta(\vec{\mathbf{x}}^\beta) \geq \mathsf{F}(\vec{\mathbf{x}}^\beta).$$

The Penalty Theorem

Let $\left\{\beta_k\right\}_{k\in\mathbb{N}}$ be a sequence tending to infinity such that $\beta_{k+1} > \beta_k > 0$ for every $k \in \mathbb{N}$. Assume that the minimum $\vec{\mathbf{x}}^{\beta_k}$ of $\mathscr{F}_{\beta_k}$ exits and it has been computed for every $k \in \mathbb{N}$. Then, any limit point of the sequence $\left\{\vec{\mathbf{x}}^{\beta_k}\right\}_{k\in\mathbb{N}}$ is a solution of the problem

$$\begin{gathered}\min \mathrm{F}\bigl(\vec{\mathbf{x}}\bigr)\\ \text{subject to } g_i\bigl(\vec{\mathbf{x}}\bigr) \le 0 \quad \text{for } i = 1, 2, \ldots, \ell.\end{gathered}$$

The algorithm: Initialization Step

Select and fix:

- a maximum number of iterates M,
- a growth parameter $\kappa > 1$,
- a small stopping tolerance $\varepsilon_1 > 0$,
- a second small stopping tolerance $\varepsilon_2 > 0$,
- an initial value of the penalty parameter $\beta_0 > 1$, and
- choose an initial seed $\vec{\mathbf{x}}^{\beta_0}$ that violates at least one constraint.

The algorithm: Iterative Step

For every $k = 1, 2, \ldots, M$ **do**

- compute $\beta_k = \kappa\, \beta_{k-1}$,
- formulate the augmented objective function $\mathscr{F}_{\beta_k}$,
- starting from $\vec{\mathbf{x}}^{\beta_{k-1}}$, use an unconstrained search technique to find the point $\vec{\mathbf{x}}^{\beta_k}$ that minimizes $\mathscr{F}_{\beta_k}$, if possible;
- determine which constraints (if any) are violated at $\vec{\mathbf{x}}^{\beta_k}$.

The algorithm: Stopping Rule

- If a point $\vec{\mathbf{x}}^{\beta_k}$ does not violate any constraint, then it is a solution of the problem.
- If $\left\|\vec{\mathbf{x}}^{\beta_k} - \vec{\mathbf{x}}^{\beta_{k-1}}\right\| < \varepsilon_1$ then stop because of convergence according to criterium one.
- If $\left|\mathrm{F}\left(\vec{\mathbf{x}}^{\beta_k}\right) - \mathrm{F}\left(\vec{\mathbf{x}}^{\beta_{k-1}}\right)\right| < \varepsilon_2$ then stop because of convergence according to criterium two.
- Finally, if $k > M$ and the algorithm did not stop before, halt the computation with fail (without finding any good approximation to a solution).

Barrier function

The idea do a search by interior points of the feasible region. This is achieved by replacing the initial problem by

$$\text{Minimize } \mathsf{F}(\vec{\mathbf{x}}) + \rho\,\mathscr{B}(\vec{\mathbf{x}})$$

where $\rho > 0$ is the *barrier parameter*, and $\mathscr{B}$ is a continuous non-negative function on the interior of the feasible region. This function can be defined for instrance by

$$\mathscr{B}(\vec{\mathbf{x}}) = \sum_{k=1}^{\ell} \frac{-1}{g_k(\vec{\mathbf{x}})},$$

and is valid only for interior points (such that all constraints are strictly satisfied for all i: $g_i(\vec{\mathbf{x}}) < 0$). Observe that the closer a point $\vec{\mathbf{x}}$ gets to a constraint boundary, then the larger $\mathscr{B}$ becomes. Hence $\mathscr{B}$ in a sense is opposite to the exterior penalty functions.

Why barriers

The interior point methods start with a feasible point and a relatively large value of ρ which prevents the point from approaching the boundary of the feasible region.

At each iteration, the value of ρ is monotonically decreased in such a way that the resultant problem is relatively easy to solve if the optimal solution of its immediate predecessor is used as the starting point.

The Barrier Theorem

The following result is the analogue of the Penalty Theorem, and gives an alike algorithm to solve the minimization problem with restrictions.

The Barrier Theorem

Let $\{\rho_k\}_{k\in\mathbb{N}}$ be a strictly monotonically decreasing sequence converging to zero. Assume that, for every $k \in \mathbb{N}$, the minimum $\vec{\mathbf{x}}^{\rho_k}$ of $\mathrm{F} + \rho\mathscr{B}$ exists and it has been computed, and verifies $g_i(\vec{\mathbf{x}}^{\rho_k}) < 0$ for every $i = 1, 2, \ldots, \ell$. Then, any limit point of the sequence $\left\{\vec{\mathbf{x}}^{\rho_k}\right\}_{k\in\mathbb{N}}$ is a solution of the problem

$$\begin{gathered}\min \mathrm{F}(\vec{\mathbf{x}})\\ \text{subject to } g_i(\vec{\mathbf{x}}) \le 0 \quad \text{for } i = 1, 2, \ldots, \ell.\end{gathered}$$

Levenberg–Marquardt algorithm. Introduction

The Levenberg-Marquardt algorithm is commonly used for solving nonlinear least squares problems. It has several key advantages and characteristics:

Efficiency in Convergence: The Levenberg-Marquardt algorithm often converges faster than other optimization methods, especially in cases where the initial guess is far from the optimal solution. This makes it particularly effective for nonlinear optimization problems with complex and non-convex objective functions.

Adaptability: The algorithm dynamically adjusts its step size during optimization, combining features of both the steepest descent method and the Gauss-Newton method. This adaptability allows it to perform well across a variety of optimization landscapes, making it a versatile choice for a broad range of applications.

Levenberg–Marquardt algorithm. Introduction (II)

Robustness: The Levenberg-Marquardt algorithm is known for its robustness in handling noisy or ill-conditioned data. It includes a damping parameter that helps control the step size during optimization, preventing large steps that could lead to divergence.

Global and Local Optimization: While it is primarily designed for local optimization, the Levenberg-Marquardt algorithm can also provide reasonable results in global optimization scenarios. This makes it useful when a good initial guess is available but may not be the global minimum.

Widely Applicable: to a wide range of problems, including parameter estimation, curve fitting, and optimization in various scientific and engineering domains. It has found applications in fields such as computer vision, machine learning, physics, and finance.

Levenberg–Marquardt algorithm. Introduction (III)

Inverse Problems: The Levenberg-Marquardt algorithm is commonly employed in solving inverse problems, where the goal is to determine the parameters of a mathematical model that best fit observed data. This is particularly valuable in fields like image reconstruction, medical imaging, and signal processing.

Implementation Simplicity: Implementing the Levenberg-Marquardt algorithm is relatively straightforward, making it accessible to researchers and practitioners. Many numerical libraries and software packages provide pre-built implementations of the algorithm, further facilitating its use.

The Levenberg–Marquardt algorithm aka damped least–squares

For this part we follow "our way"

📕 *Nonlinear Least-Squares Fitting*, GSL Documentation.

📕 Wikipedia

As already said, Levenberg–Marquardt algorithm (LMA) interpolates between the Gauss–Newton algorithm (GNA) and the method of gradient descent. The LMA is more robust than the GNA, which means that in many cases it finds a solution even if it starts very far off the final minimum. For well-behaved functions and reasonable starting parameters, the LMA tends to be slower than the GNA. LMA can also be viewed as Gauss–Newton using a *trust region* approach.

LMA — How it works?

The problem consists in,

- given a set of m empirical pairs $\big(a_i, b_i\big) \in \mathbb{R}^2$,
- a parameters' vector $x \in \mathbb{R}^p$, and
- a parameterized model function $f : \mathbb{R} \times \mathbb{R}^p \longrightarrow \mathbb{R}$,

find the parameters x of the model curve so that the sum of the squares of the deviations

$$\Phi(x) := \frac{1}{2} \sum_{i=1}^{m} \Big(b_i - f\big(a_i, x\big)\Big)^2$$

is minimized.

LMA — How it works?

The function Φ can be re-written in vector notation as

$$\Phi(x) := \frac{1}{2}\sum_{i=1}^{m} \varphi_i(x)^2$$

by introducing the function $\varphi : \mathbb{R}^p \longrightarrow \mathbb{R}^m$ defined by

$$\varphi(x) := \Big(b_1 - f\,(a_1, x), b_2 - f\,(a_2, x), \dots, b_m - f\,(a_m, x)\Big).$$

LMA — How it works?

In trust region methods, the objective (or cost) function Φ is approximated by a model function $\Phi_k(\delta)$ in the vicinity of some point x_k.

The model function is often quadratic; being simply a second order Taylor series expansion around the point x_k, i.e.:

$$\Phi\big(x_k+\delta\big) \approx \Phi_k(\delta) = \Phi\big(x_k\big) + \nabla\Phi\big(x_k\big)^{\intercal}\delta + \frac{1}{2}\delta^{\intercal} H\Phi\big(x_k\big)\delta$$

where $\nabla\Phi\big(x_k\big) = J\big(\varphi(x_k)\big)^{\intercal}\varphi(x_k)$, $J\big(\varphi(x_k)\big)$ is the *Jacobian* of φ at the point x_k, and $H\Phi\big(x_k\big) = J\big(\varphi(x_k)\big)^{\intercal} J\big(\varphi(x_k)\big)$ is the *Hessian matrix* of Φ at the point x_k.

Observe that $J\big(\varphi(x_k)\big)$ is a $m \times p$ matrix and $H\Phi\big(x_k\big)$ is a $p \times p$ matrix.

LMA — How it works?

In order to find the next step δ, the algorithm minimizes the model function $\Phi_k(\delta)$, but search for solutions only within a region where we trust that $\Phi_k(\delta)$ is a good (quadratic) approximation to the objective function $\Phi(x_k + \delta)$.

In other words, we seek a solution of the *trust region subproblem*:

$$\min_{\delta \in R^p} \Phi_k(\delta) = \Phi\big(x_k\big) + \Big(J\big(\varphi(x_k)\big)^{\intercal}\varphi(x_k)\Big)\delta + \frac{1}{2}\delta^{\intercal} H\Phi\big(x_k\big)\delta$$

such that $\|D_k\delta\| \le \Delta_k$, where $\Delta_k > 0$ is the *trust region radius*, and D_k is a *scaling matrix*.

If D_k is the identity matrix, then the trust region is a ball of radius Δ_k centered at x_k.

LMA — How it works?

This strategy divides every iteration of the algorithm into two problems:

- A minimize the function $\Phi_k(\delta)$ within the trust region, and
- B find a *good* trust region. That is, a region where the solution δ_k of (A) verifies $\Phi\big(x_k\big) > \Phi\big(x_k + \delta_k\big)$. Fortunately, in the spirit of backtracking algorithm in dimension one, if the trust region is too big and the objective function does not decrease, the computations can be re-done with a smaller trust region until the goal $\Phi\big(x_k\big) > \Phi\big(x_k + \delta_k\big)$ is achieved.

NOTE: In some sense the trust region algorithm is a multi-dimensional backtracking procedure.

On the trust region subproblem

Once a δ_k is computed (by minimizing Φ_k in the trust region), it is checked whether or not $\Phi\big(x_k + \delta_k\big)$ reduces the value of objective function $\Phi\big(x_k\big)$. A useful statistic for this is to look at the ratio

$$\rho_k := \frac{\Phi\big(x_k\big) - \Phi\big(x_k + \delta_k\big)}{\Phi_k(0) - \Phi_k\big(\delta_k\big)}$$

where the numerator is the actual reduction of the objective function due to the step δ_k, and the denominator is the predicted reduction due to the model Φ_k.

- If ρ_k is negative, the step δ_k increases the objective function and so it is rejected. When a step is rejected, the trust region is made smaller and the TRS is solved again.
- If ρ_k is positive, then we have found a step which reduces the objective function and it is accepted.
- If ρ_k is close to 1, then this indicates that the model function is a good approximation to the objective function in the trust region, and so on the next iteration the trust region is enlarged in order to take more ambitious steps.

On the solution of $\min_{\delta \in R^p} \Phi_k(\delta)$

We use the Gauss-Newton Method to find zeros of the gradient of the model function:

$$
\begin{aligned}
0 = \nabla\Phi_k(\delta) = \\
&\nabla\Big(\Phi\big(x_k\big) + \Big(J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k)\Big)\delta + \frac{1}{2}\delta^{\mathsf{T}} H\Phi\big(x_k\big)\delta\Big) = \\
&\qquad H\Phi\big(x_k\big)\delta + J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k).
\end{aligned}
$$

Hence,

$$\delta_k = -H\Phi\big(x_k\big)^{-1} J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k),$$

which is the descent direction of the Newton Method (see [Quarteroni et al]).

Since we never, never, never (and in case of doubt never) have to invert a matrix, the direction δ_k is better computed by solving the *normal equations system*:

$$J\big(\varphi(x_k)\big)^{\mathsf{T}} J\big(\varphi(x_k)\big)\delta_k = H\Phi\big(x_k\big)\delta_k = -J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k)$$

On the solution of $\min_{\delta \in R^p} \Phi_k(\delta)$ (II)

Levenberg's contribution is to replace the above system by a "damped" version of it:

$$\Big(J\big(\varphi(x_k)\big)^{\mathsf{T}} J\big(\varphi(x_k)\big) + \mu_k D_k^{\mathsf{T}} D_k\Big)\delta_k = -J\big(\varphi(x_k)\big)^{\mathsf{T}} \varphi(x_k)$$

where μ_k is the non-negative *damping factor*, which is adjusted at each iteration.

Observe that:

- if reduction of Φ_k is rapid, a smaller value of μ_k can be used, bringing the algorithm closer to the Gauss–Newton algorithm (when D_k is the identity matrix).
- If an iteration gives insufficient reduction in the residual, μ_k can be increased, giving a step closer to the gradient-descent direction (again when D_k is the identity matrix).

On the solution of $\min_{\delta \in R^p} \Phi_k(\delta)$ (III)

Explanation: Recall that the gradient of $\Phi\big(x_k + \delta\big)$ with respect to δ is $J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k)$. Therefore, when $D_k = \mathrm{Id}$ and μ_k is large, the step δ_k (i.e. the solution of the above system) is well approximated by $-\frac{1}{\mu_k}J\big(\varphi(x_k)\big)^{\mathsf{T}}\varphi(x_k)$ which goes approximately in the direction opposite to the gradient.

The damping parameter is usually updated by multiplying or dividing it by a factor $\nu > 1$.

Note

Fletcher proposed to replace the matrix $D_k^{\mathsf{T}} D_k$ with the diagonal matrix consisting of the diagonal elements of $J\big(\varphi(x_k)\big)^{\mathsf{T}} J\big(\varphi(x_k)\big)$. That is with $\operatorname{diag}\Big(J\big(\varphi(x_k)\big)^{\mathsf{T}} J\big(\varphi(x_k)\big)\Big)$.

Fitting of the first epoch parameters' for the Audouin's gulls: the Model

$$\frac{\mathrm{d}}{\mathrm{d}t} x(t) = \alpha x(t) - \beta x(t)^2$$

with parameters

Parameter	Units	Range or value	Ecological meaning or description
ϑ	year^{-1}	$[0, +\infty)$	Intrinsic reproduction rate
ω	year^{-1}	$[0, +\infty)$	Rate of entry of individuals from other patches
$K = \frac{\alpha+\varepsilon}{\beta}$	birds	$[1, +\infty)$	Carrying capacity
ε	year^{-1}	0.11	Death rate estimated from field data
$\alpha = \vartheta + \omega - \varepsilon$	year^{-1}	$(-\infty, \vartheta + \omega]$	Population growth rate including death of individuals (without linear dispersal)
$x(0)$	birds	$[0, K]$	Initial condition
β	(birds $\times$ year)$^{-1}$	$[0, +\infty)$	Intrinsic growth rate over the carrying capacity

Model fitting and parameters estimation

Initial phase 1981–1997

The model in the initial phase is a particular case of a Ricatti Equation with constant coefficients. Its closed analytical solution is

Lemma (A Ricatti Equation with constant coefficients)

Since β must be non-negative, the solution $x(t)$ of the above model is:

- If $\alpha \neq 0$,

$$x(t) = \frac{\alpha x(0) \exp(\alpha t)}{\alpha + \beta x(0)(\exp(\alpha t) - 1)} = \frac{\alpha x(0)}{\alpha \exp(-\alpha t) + \beta x(0)(1 - \exp(-\alpha t))}.$$

- If $\alpha = 0$,

$$x(t) = \frac{x(0)}{x(0)\beta t + 1}.$$

Model fitting and parameters estimation

Initial phase 1981–1997

The observed population of Audouin's gulls at the years 1981 to 1997 is:

$$\eta(\ell, \ell = 0:16) = \text{Audouin's\_Gulls\_Observed\_Population\_at\_year}(1981+\ell, \ell = 0:16) =$$
$$\Big[\overset{[0]}{36}, \overset{[1]}{200}, \overset{[2]}{546}, \overset{[3]}{1200}, \overset{[4]}{1200}, \overset{[5]}{2200}, \overset{[6]}{1850}, \overset{[7]}{2861}, \overset{[8]}{4266}, \overset{[9]}{4300}, \overset{[10]}{3950},$$
$$\overset{[11]}{6714}, \overset{[12]}{9373}, \overset{[13]}{10143}, \overset{[14]}{10327}, \overset{[15]}{11328}, \overset{[16]}{11725}\Big].$$

The solution of the above model with $\beta \geq 0$ and initial condition $\kappa \in \mathbb{R}^+$ will be denoted by $x(t) = x_{\kappa,\alpha,\beta}(t)$. Observe that $\kappa = x_{\kappa,\alpha,\beta}(0)$ must be considered a free parameter as well.

Model fitting and parameters estimation

Initial phase 1981–1997

We introduce a map L to measure the agreement between the solution of the model with initial condition κ and parameters α and β, and the observed data $\eta(\ell, \ell = 0 : 16)$, through the Euclidean norm:

$$\begin{array}{llcl} \mathsf{L}: & \mathscr{F} & \longrightarrow & \mathbb{R}^+ \\ & \big(\kappa, (\alpha, \beta)\big) & \longmapsto & \sqrt{\sum_{\ell=0}^{16} \big(x_{\kappa,\alpha,\beta}(\ell) - \eta(\ell)\big)^2}, \end{array}$$

where

$$\mathscr{F} := \mathbb{R}^+ \times \big\{(\beta K - \varepsilon, \beta) : \beta \in \mathbb{R}^+\big\}$$

is the parameter space.

Recall that $\alpha = \beta K - \varepsilon \in (-\varepsilon, \infty)$.

Model fitting and parameters estimation

Initial phase 1981–1997

Of course, if the dynamics of the Audouin's gulls population size during the years 1981 to 1997 is governed by some instance of the model $\alpha x(t) - \beta x(t)^2$ with parameters $x(0) = \kappa^*$, $\alpha = \alpha^*$ and $\beta = \beta^*$, then the value of $\mathsf{L}\big(\kappa^*, (\alpha^*, \beta^*)\big)$ must be small and likely it must correspond to

$$\begin{aligned} \min\ & \mathsf{L}\big(\kappa, (\alpha, \beta)\big) \\ \text{subject to } & \big(\kappa, (\alpha, \beta)\big) \in \mathscr{F}, \\ & x(0) = \kappa \geq 0, \\ \text{and } & x(t) \geq 0 \text{ for } t \in [0, 16]. \end{aligned}$$

The solution of this problem is called the *fitting of the model* and identifies a valid analytical model for the dynamics of the Audouin's gulls at the years 1981 to 1997 (of course provided that the value $\min \mathsf{L}\big(\kappa, (\alpha, \beta)\big)$ is small).

Model fitting and parameters estimation

Initial phase 1981–1997

To solve the above problem we have used the algorithm in Slides 33–35 with the following objective function:

$$\mathsf{L}\big(\kappa,(\alpha,\beta)\big)+c\left(\sum_{\ell=0}^{16}\Big(\max\Big\{0,-x_{\kappa,\alpha,\beta}(\ell)\Big\}\Big)^2+\max\Big\{0,10^{-16}-\beta\Big\}^2\right)=$$

$$\sqrt{\sum_{\ell=0}^{16}\big(x_{\kappa,\alpha,\beta}(\ell)-\eta(\ell)\big)^2}+c\left(\sum_{\ell=0}^{16}\Big(\max\Big\{0,-x_{\kappa,\alpha,\beta}(\ell)\Big\}\Big)^2+\max\Big\{0,10^{-16}-\beta\Big\}^2\right),$$

where c is the penalty parameter.

Model fitting and parameters estimation

Initial phase 1981–1997

To solve the free minimization problems we have used Levenberg–Marquardt algorithm with numerical computation of derivatives (laziness) with the following initialization:

- the maximum number of iterates M is set to 1000,
- the growth parameter $\widetilde{c}$ is set to 2,
- The two tolerances $\varepsilon_1 = \varepsilon_2$ are set to $1.0e - 16$
- The initial value of the penalty parameter c_0 is set to 2.

The seed has been set to have $\beta = 0$, and $\kappa = x(0)$ and α equals to the result of a logarithmic regression of the observed data with respect to years (that is, $\big(\ell, \log \eta(\ell)\big)$), since we know that the solution of the EDO must be of exponential type.

Model fitting and parameters estimation: The result

Initial phase 1981–1997

Parameters' values at the optimum

Parameter	**Value**
$\kappa = x_{\kappa,\alpha,\beta}(0)$	288.04096 ± 117.9663
α	$0.348949410467223 7 \pm 0.04958259$
β	$0.0000243826356697 \pm 0.00000598145$
ε	0.11 (estimated from data)
$\gamma = \alpha + \varepsilon$	$0.45894940\cdots$
$K = \frac{\gamma}{\beta}$	$18822.79734\cdots$

Quadratic Error $= \mathrm{L}\big(\kappa, (\alpha, \beta)\big) = 2593.053\cdots$

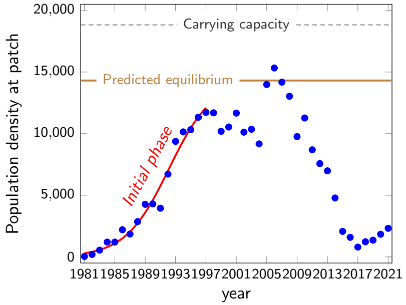
Model fitting and parameters estimation: The result

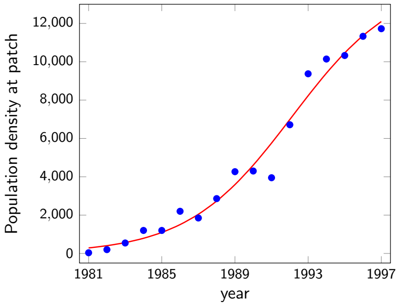
Initial phase 1981–1997

Year	Population Data	
	Observed	Predicted
1981	36	288.040962
1982	200	404.917270
1983	546	567.299150
1984	1200	791.095769
1985	1200	1096.137854
1986	2200	1505.703287
1987	1850	2044.623822
1988	2861	2735.234088
1989	4266	3590.827296
1990	4300	4607.530655
1991	3950	5757.502051
1992	6714	6987.807453
1993	9373	8228.125064
1994	10143	9405.848001
1995	10327	10462.226529
1996	11328	11362.442028
1997	11725	12096.688846

Model fitting and parameters estimation: The result

Initial phase 1981–1997





Left plot: Dynamics of the local population for the estimated parameter values shown above (red line). The predicted equilibrium, computed from $x_1^\star = \frac{\alpha}{\beta} = 14311.390089$ is shown with a horizontal brown line, while the carrying capacity is shown with a horizontal dashed line. The model predictions suggest that the population have not reached the steady state on the onset of the perturbation, and that the large population increase suffered in 2005–2006 did not surpass the carrying capacity.

Right plot: Zoom in the period 1981–1997.