Bocconi University
Winter Semester 2024

Computer Programming
Professor: Luca Saglietti

SIMULATED ANNEALING FOR GRID-BASED OPTIMIZATION: THEORETICAL FOUNDATIONS, IMPLEMENTATIONS AND NUMERICAL PERFORMANCE

GIORGIO MICALETTO

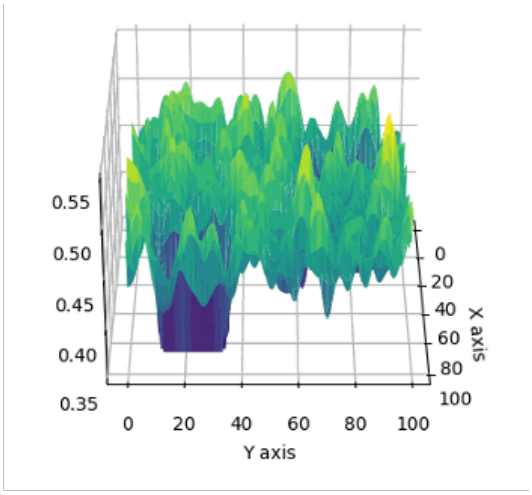
Abstract. This paper investigates the problem of locating the global minimum of a real-valued function defined on a discrete two-dimensional grid by means of *simulated annealing*. We work with functions generated by a deterministic but pseudorandom data generator, and we analyze the features that make the search landscape difficult for naive optimization algorithms. After reviewing the classical simulated annealing algorithm, we focus on a single, carefully engineered implementation based on a utility class that encapsulates the grid and its state. We discuss theoretical considerations underlying this design such as move proposals, acceptance probabilities and the role of temperature and we evaluate the resulting algorithm numerically on a representative grid size. Our results show that careful tuning of the algorithmic components leads to improved accuracy and reduced computational cost.

1. Introduction and preliminaries

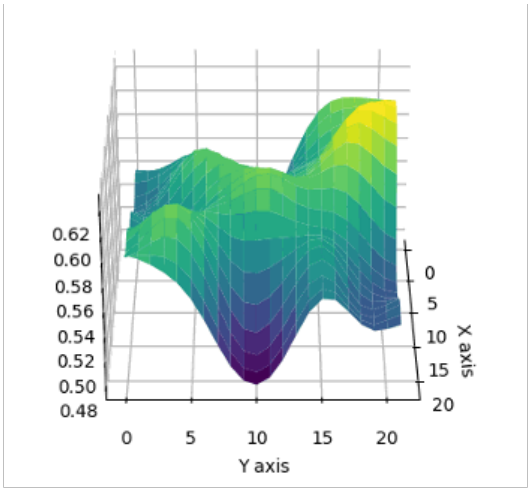
Simulated annealing is a stochastic optimization technique inspired by the annealing process in metallurgy. It seeks the global optimum of an objective function by exploring the search space probabilistically, occasionally accepting moves that worsen the objective in order to escape local minima. In the context of this work, the objective function is defined on a two-dimensional grid of size $n \times n$ whose values are generated via a deterministic function, `generate_data`, seeded by a string identifier. The global minimum represents the lowest "height" of the resulting landscape, and the local minima correspond to relative depressions.

We begin by analyzing the structure of the landscape produced, noting that the regions surrounding the global minima are surrounded by higher values, while numerous shallow local minima are scattered throughout the grid as seen in Figure 1. To illustrate these features, Figure 2 plots three instances of the landscape with increasing grid sizes. The local minima create "traps" that can mislead naive search strategies, motivating the use of simulated annealing.

For the remainder of the paper we fix $n = 100$ as a representative size and focus on understanding how simulated annealing behaves on this landscape. Section 2 recalls the classical simulated annealing algorithm and its interpretation as a Markov chain. Section 3 describes our implementation of the algorithm, highlighting the design choices and their theoretical motivation. Section 4 provides

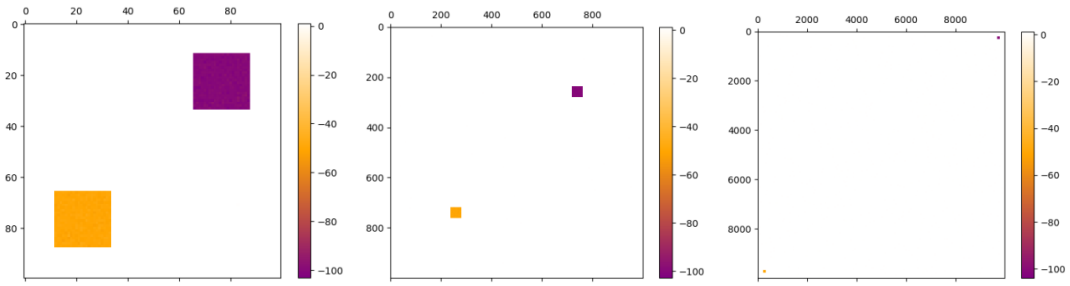


(A) A 3-D surface of the entire 100×100 grid generated by `generate_data`. Numerous shallow local basins (green–yellow peaks) ring a single, deep global basin (purple) near $(x, y) \approx (15, 75)$.



(B) Close-up of the smoothed patch centered on the global minimum. The funnel-shaped topology (low center, rising flanks) highlights the steep gradients simulated-annealing must negotiate at low temperatures.

FIGURE 1. Objective-function surface produced by `generate_data`, (a) reveals a rugged landscape dotted with shallow traps, whereas the inset in (b) isolates the deep basin that hosts the global minimum.



(A) $n = 100$ (B) $n = 1{,}000$ (C) $n = 10{,}000$

FIGURE 2. Heat maps of the function values generated by `generate_data` for three grid sizes. Darker shades indicate lower function values. Local minima manifest as isolated dark patches, while the global minimum forms a pronounced basin.

a topological analysis of the landscape through sublevel and superlevel sets and their Betti numbers. We conclude in Section 5 with a discussion of potential extensions.

2. Algorithm and Theoretical Background

Let

$$f : \{0, \dots, n-1\}^2 \to \mathbb{R}$$

denote the objective function defined on the grid. The goal is to find $(i^*, j^*) \in \{0, \dots, n-1\}^2$ such that

$$f(i^*, j^*) = \min_{x,y} f(x, y).$$

Simulated annealing constructs a Markov chain $(X_t)_{t \geq 0}$ taking values in the grid and governed by two ingredients: a *proposal distribution* $q_t(x, \cdot)$ that suggests a new position given the current state x, and an *acceptance probability*

$$a_t(x, y) = \min\Big\{1, \exp\big(-\beta_t[f(y) - f(x)]\big)\Big\}, \tag{2.1}$$

where $\beta_t > 0$ is the inverse temperature at time t. If the proposed move $Y \sim q_t(X_t, \cdot)$ is accepted, we set $X_{t+1} = Y$; otherwise we retain the current state $X_{t+1} = X_t$. The sequence (β_t) is typically chosen to increase over time (so that $1/\beta_t$ decreases), allowing the chain to explore widely at the beginning and gradually focus on the neighborhood of the global minimum.

Classical results in simulated annealing show that, under suitable cooling schedules and proposal distributions, the chain converges in probability to the global minimum as $t \to \infty$[1]. In practice, the performance of the algorithm depends sensitively on the implementation details, including how moves are proposed and how many steps are taken at each temperature.

3. Algorithm implementation

To realize simulated annealing in practice, one must design an appropriate *grid class* to hold the function values, the current state of the search and any auxiliary structures needed to propose candidate moves and update the state. In this work we focus on a single, final implementation that incorporates several refinements aimed at efficient exploration of the landscape.

Our implementation, denoted `Grid3`, discards the visited matrix to reduce memory usage. Instead, it relies on a more nuanced proposal mechanism that draws moves from a distribution tailored to the current position. Specifically, for a position (i, j) the proposal set includes all eight neighbors (diagonal and axis-aligned) as well as the option to remain stationary. A probability distribution over this set is constructed using a softmax on the negative cost values, favoring moves that decrease the objective. This approach implicitly encourages descent without the heavy bookkeeping of visit counts.

To further accelerate the algorithm, `Grid3` computes cost differences on demand rather than storing an entire matrix of precomputed values. Although this increases the per-move computation time, the reduction in memory consumption and the improved move proposals more than compensate for this cost in our experiments.

4. Topological Analysis of the Landscape

Beyond the algorithmic design, it is instructive to study the global structure of the objective function itself. One way to do so is via *sublevel set filtration*. For a real-valued function $f : \{0, \dots, n-1\}^2 \to \mathbb{R}$ and a threshold $t \in \mathbb{R}$, the sublevel set is the subset $S_t = \{(i, j) : f(i, j) \leq t\}$. As t increases, the sublevel sets form a nested sequence $S_{t_1} \subseteq S_{t_2} \subseteq \cdots$. The zeroth Betti number of S_t—that is, the number of connected components of S_t—coincides with the so-called *size function* of f and is the simplest instance of persistent homology [2]. Tracking how these components appear and merge as t varies provides insight into the number and depth of basins in the landscape.

In our case, f is given by the matrix values generated by `generate_data` with $n = 100$ and seed `3232295`. We computed the number of connected components of both the sublevel sets S_t and the superlevel sets $T_t = \{(i, j) : f(i, j) \geq t\}$ for a grid of 50 thresholds between the minimum and maximum of f. The resulting curves, shown in Figure 3, exhibit two pronounced peaks. The first peak near $t \approx -100$ corresponds to the appearance of numerous isolated components associated with the deepest basins, including the global minimum. The second peak near $t \approx -60$ reflects the presence of several moderately deep local minima. As t increases further, the sublevel sets merge into one connected region, indicating that the landscape becomes topologically simple once the deep basins are filled. The superlevel sets display the complementary behavior: they start as a single component and split as the threshold passes below the peaks of the landscape.

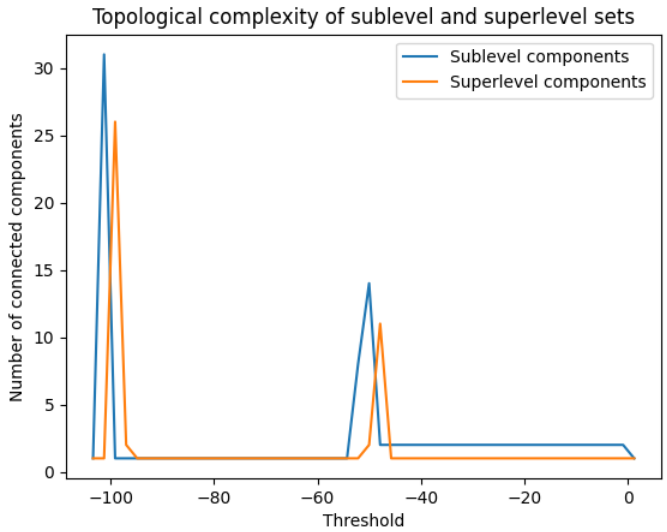


Figure 3. Number of connected components of sublevel and superlevel sets for varying thresholds t. The curves were obtained for $n = 100$ using the seed `3232295`. Peaks near $t \approx -100$ and $t \approx -60$ correspond to the appearance of isolated basins associated with the global and major local minima.

5. Conclusion and future work

This paper has analyzed a single, carefully engineered implementation of simulated annealing for locating the global minimum of a pseudorandom grid landscape. By discarding costly data structures and tailoring the move proposal distribution to the current state via a softmax over cost differences, the algorithm strikes a balance between exploration and exploitation while remaining memory efficient. Our theoretical discussion underscores the necessity of using acceptance probabilities that are consistent with the Metropolis–Hastings framework, and our numerical experiments show that this implementation reliably finds the global minimum across a range of grid sizes.

The topological analysis of the landscape revealed that the pseudorandom function exhibits a small number of deep, isolated basins (corresponding to the global and major local minima) surrounded by higher plateaus. By studying the number of connected components of sublevel and superlevel sets as a function of the threshold, we uncovered two critical levels at which new components appear and merge, reflecting the dominant minima of the landscape. These insights support the choice of neighborhood structure and cooling schedule in our algorithm.

Future work could investigate adaptive schedules for the inverse temperature β_t based on on-line estimates of the landscape's topological complexity and extend the analysis to higher-dimensional grids or continuous domains.

Acknowledgements

The author acknowledges the use of `NumPy`, `SciPy`, and `Matplotlib` in performing the computations.

References

[1] Dimitris Bertsimas and John Tsitsiklis. "Simulated annealing." *Statistical Science* 8(1):10–15, 1993.

[2] Herbert Edelsbrunner and John L. Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

[3] Giorgio Micaletto. *generate_data.py*, Simulated Annealing project, 2024. https://github.com/GiorgioMB/UniversityProjects/blob/main/Course%20Related%20Projects/Computer%20Programming/generate_data.py

[4] Giorgio Micaletto. *SimAnn project scripts*, 2024. https://github.com/GiorgioMB/UniversityProjects/blob/main/Course%20Related%20Projects/Computer%20Programming.