

An Exact Branch-and-Cut Method for the Asymmetric Orienteering Problem with Terrain-Driven Fatigue

Pablo Borrego Ramos*

September 18, 2026

Abstract

We present an end-to-end computational framework for provably optimal route planning in orienteering races on real terrain. The framework consists of a GIS pipeline that combines International Orienteering Federation (IOF) map symbols, a digital elevation model (DEM), and the Minetti metabolic model into a fully asymmetric cost matrix, where directional slope costs and cumulative fatigue make travel cost depend on both direction and elapsed effort. The terrain realities captured by this pipeline lead to a new problem variant of the Orienteering Problem (OP), the Asymmetric Orienteering Problem with Fatigue (AOPF), for which we develop tailored exact solution methods. We propose two LP formulations: an MTZ-based model with McCormick linearisation and a single-commodity flow formulation. We strengthen the Branch-and-Cut solver with routing infeasibility cuts, alongside tightened flow-arc coupling bounds, fatigue-aware lifted covers, and directed adaptations of cycle cover and path inequalities. We validate the framework on two Spanish terrains across seven control-point distributions each, and release a parameterised benchmark suite of 66 instances for the asymmetric OP with fatigue. The flow-based B&C certifies global optimality on 11 of 14 real-terrain instances (all ≤ 40 control points) within a 15-minute time limit, with tighter LP gaps than the MTZ formulation (mean gap 11.4% vs. 20.0% on benchmark instances).

Keywords: orienteering problem, asymmetric routing, cumulative fatigue, branch-and-cut, valid inequalities, flow formulation, terrain cost surface, Minetti metabolic model

1 Introduction

The Orienteering Problem (OP) asks a competitor to select a subset of control points and determine a route starting and ending at a depot, maximising the total score collected while respecting a travel budget. Formally introduced by Tsiligrirides [1984], the OP is NP-hard and has been studied extensively in operations research. However, most formulations assume symmetric, Euclidean distances, which poorly approximate real-world orienteering where terrain, vegetation, slope direction, and fatigue create highly asymmetric travel costs.

This paper bridges two research streams: (i) GIS-based cost surface modelling for orienteering maps, following Albert and Sárközy [2021], and (ii) exact optimisation of the OP via integer programming. The terrain realities captured by the cost pipeline produce a problem structure not addressed by existing OP formulations, motivating both a new problem variant and tailored solution methods. Our contributions are:

1. A terrain cost model combining IOF map symbols (rasterised into an Areal Cost Raster, ACR), DEM-derived morphometry (the Hypsometry Cost Raster, HCR), directional metabolic cost (Minetti), and cumulative fatigue into a single asymmetric cost matrix.
2. An Interquartile Range (IQR)-based scaling method for merging ACR and HCR that adapts to terrain characteristics without manual calibration.

*Independent Researcher.

3. The Asymmetric Orienteering Problem with Fatigue (AOPF), a problem variant arising naturally from the terrain pipeline, formalising the OP with directed travel costs and a cumulative fatigue multiplier. To the best of our knowledge, this is the first exact formulation for the OP with position-dependent fatigue costs.
4. Two LP formulations for the AOPF: (a) an MTZ-based model with McCormick linearisation of the bilinear fatigue term, and (b) a single-commodity flow formulation that ties the fatigue state to a conservation constraint instead of a big- M relaxation, tightening the LP relaxation. The MTZ formulation serves as a natural baseline with fewer variables ($O(n)$ time variables vs $O(n^2)$ flow variables).
5. Valid inequalities tailored to the AOPF: (a) tightened flow-arc coupling bounds exploiting minimum arrival times, (b) fatigue-aware lifted cover inequalities with position-dependent cost inflation, and (c) routing infeasibility cuts that identify subsets of nodes that cannot all be visited within the fatigue-adjusted budget. We also adapt the cycle cover and path inequalities of Fischetti et al. [1998] to the directed asymmetric setting with fatigue.
6. A parameterised benchmark suite of 66 synthetic instances for the asymmetric OP with fatigue, varying node count, asymmetry, fatigue rate, and budget tightness with three random seeds per configuration, plus boundary-condition extreme cases.
7. Computational experiments on two real terrains comparing the MTZ and flow formulations, an ablation study on the benchmark suite quantifying the marginal contribution of each cut family, and a comparison against a classical (undirected, symmetric, fatigue-free) OP baseline following Fischetti et al. [1998] to directly measure what asymmetry and fatigue cost in solve difficulty relative to the special case.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 describes the cost surface model. Section 4 presents the mathematical formulation. Section 5 details the solution methods. Section 6 outlines the computational pipeline. Section 7 describes the experimental setup, and section 8 presents the results, including a synthetic benchmark suite for the asymmetric OP with fatigue. Section 9 discusses findings and limitations, and section 10 concludes.

2 Related Work

The Orienteering Problem. The OP was introduced by Tsiligrirides [1984] and has since generated a substantial body of literature. Vansteenwegen et al. [2011] provide a comprehensive survey covering exact methods, heuristics, and variants including the Team OP, OP with Time Windows, and stochastic OP. Exact approaches typically employ branch-and-bound or branch-and-cut on integer programming formulations [Fischetti et al., 1998]. Most formulations assume symmetric distances derived from Euclidean or road-network metrics. Gunawan et al. [2016] extend the survey to recent variants and applications, noting that terrain-aware formulations remain scarce. Most recently, Kobeaga et al. [2021] proposed a revisited B&C algorithm for large-scale symmetric OP instances, incorporating shrinking techniques for subtour elimination, blossom separation heuristics for comb inequalities, efficient variable pricing, and a structured separation loop. Their component evaluation methodology informs the ablation study in the present work. Our formulation differs fundamentally in addressing asymmetric costs and cumulative fatigue, which preclude the use of undirected comb inequalities and require the flow-based linearisation developed here.

Metaheuristics for the OP. Given the NP-hardness of the OP, metaheuristics are widely used for practical instances. Simulated annealing [Kirkpatrick et al., 1983], genetic algorithms, and ant colony optimisation have all been applied. Vansteenwegen et al. [2009] propose an iterated local search that is competitive with exact methods on benchmark instances. Our SA implementation incorporates Or-opt moves and stagnation detection, with parameters tuned via

grid search over 1,000 random instances.

Terrain cost modelling. Least-cost path (LCP) analysis on terrain has a long history in GIS. Tobler [1993] proposed the hiking function relating slope to walking speed, which remains widely used despite known limitations on steep descents. Minetti et al. [2002] measured the metabolic cost of locomotion across a wide range of slopes, producing a fifth-degree polynomial that better captures the energetics of both ascent and descent.

Orienteering map cost surfaces. Albert and Sárközy [2021] introduced a GIS-based approach to route planning on orienteering maps, assigning travel speeds to IOF symbol categories and computing least-cost paths. They proposed the Hypsometry Cost Raster (HCR) combining terrain ruggedness, plan curvature, and slope magnitude to capture morphometric difficulty not represented by map symbols. Our work extends their approach by: (i) replacing Tobler’s hiking function with the Minetti metabolic model for directional slope costs, (ii) proposing IQR-based scaling instead of histogram mode matching for ACR-HCR fusion, and (iii) coupling the cost surface with an exact OP solver rather than point-to-point LCP analysis.

Asymmetric and state-dependent routing. The Asymmetric Travelling Salesman Problem (ATSP) has a rich polyhedral theory, including directed subtour elimination constraints (SECs) and facet-defining inequalities such as the D_k^+ and odd-CAT families [Fischetti, 1991, Fischetti and Toth, 1997]. Our directed SECs and connectivity cuts follow the standard ATSP formulation; the budget-based cuts (B1–B4) are specific to the selective (orienteering) setting where node visits are optional. The fatigue model in the AOPF makes arc costs state-dependent: the effective cost of traversing an arc depends on the cumulative cost elapsed along the route. State-dependent travel costs arise in several routing contexts, including load-dependent vehicle routing [Bektaş and Laporte, 2011] and time-dependent VRP [Malandraki and Daskin, 1992]. In these problems, the bilinear interaction between a state variable and arc selection creates linearisation challenges analogous to our fatigue term; our flow-based formulation instead ties the state variable to a conservation constraint, avoiding the big- M relaxation that a McCormick-based linkage would otherwise require. The OP is a member of the broader family of TSP with Profits [Feillet et al., 2005], which includes the prize-collecting TSP and the profitable tour problem. To the best of our knowledge, no prior work in this family addresses position-dependent cumulative fatigue costs with exact methods.

3 Cost Surface Model

Let the terrain be discretised into a regular grid of cells with resolution δ (metres per cell), where each cell is indexed by its row r and column c . Each cell (r, c) is assigned a base traversal cost that integrates map-derived and DEM-derived information.

3.1 Areal Cost Raster (ACR)

The ACR assigns a dimensionless weight $w_{rc} \in (0, 10]$ to each cell based on the IOF map symbol classification. Following Albert and Sárközy [2021], each symbol category maps to a running speed, and the weight is inversely proportional:

$$w_{rc} = \frac{\delta}{v_{\text{base}} \cdot s_{\text{cat}}(r, c)} \quad (1)$$

where $v_{\text{base}} = 2.5 \text{ m/s}$ is the reference flat-terrain speed, s_{cat} is the speed factor for the symbol category (e.g., $s = 1.0$ for paved road, $s = 0.133$ for fight vegetation), and δ is the cell size. Impassable features (water, cliffs, buildings) receive $w = 10$.

The ACR is rasterised from the orienteering map file by parsing symbol objects and rendering them onto the Universal Transverse Mercator (UTM) grid with appropriate buffer widths. Bézier curves are tessellated to preserve cartographic accuracy.

3.2 Hypsometry Cost Raster (HCR)

The HCR captures terrain morphometry not represented by map symbols, derived from a Digital Terrain Model (DTM). Following Albert and Sárközy [2021], three morphometric variables are computed and normalised to $[\ell, u] = [0.1, 1.0]$:

Terrain Ruggedness Index (TRI). The standard deviation of elevation in a 3×3 neighbourhood [Riley et al., 1999]:

$$\text{TRI}_{rc} = \sqrt{\frac{1}{9} \sum_{(i,j) \in \mathcal{N}(r,c)} (z_{ij} - \bar{z}_{rc})^2} \quad (2)$$

where z_{ij} is the elevation at cell (i, j) and $\bar{z}_{rc}$ is the local mean.

Plan Curvature (PC). The absolute value of plan curvature detects ridges, spurs, and side valleys [Zevenbergen and Thorne, 1987]:

$$\text{PC}_{rc} = \left| \frac{-(p^2 r - 2pq s + q^2 t)}{(p^2 + q^2)^{3/2}} \right| \quad (3)$$

where $p = \partial z / \partial x$, $q = \partial z / \partial y$, $r = \partial^2 z / \partial x^2$, $s = \partial^2 z / \partial x \partial y$, $t = \partial^2 z / \partial y^2$, computed via central finite differences at resolution δ .

Slope Magnitude (SLO). The slope angle in degrees:

$$\text{SLO}_{rc} = \arctan \sqrt{\left(\frac{\partial z}{\partial x}\right)^2 + \left(\frac{\partial z}{\partial y}\right)^2} \quad (4)$$

Each variable is normalised via robust percentile clipping:

$$\hat{V}_{rc} = \ell + (u - \ell) \cdot \frac{\text{clip}(V_{rc}, V^{(1\%)}, V^{(99\%)}) - V^{(1\%)}}{V^{(99\%)} - V^{(1\%)}} \quad (5)$$

The HCR is then:

$$\text{HCR}_{rc} = \widehat{\text{TRI}}_{rc}^\alpha \cdot |\widehat{\text{PC}}_{rc}|^\beta \cdot \widehat{\text{SLO}}_{rc}^\gamma \quad (6)$$

with $\alpha = 1$, $\beta = 0.5$, $\gamma = 1$ as specified by Albert and Sárközy [2021].

3.3 IQR-Based Raster Fusion

To combine ACR and HCR, the HCR must be scaled into ACR’s numerical space. Albert and Sárközy [2021] proposed histogram mode matching: $c = f_{\text{ACR}}(\mu) / f_{\text{HCR}}(\mu)$, where $f(\mu)$ denotes the density at the mode. This is sensitive to distribution shape and can produce unstable scaling constants. For example, when the ACR distribution is multimodal (due to discrete symbol categories) and the HCR distribution is unimodal and right-skewed, the mode densities $f(\mu)$ can differ by an order of magnitude, yielding scaling constants that vary erratically with small changes in bin width.

We evaluated six scaling methods on both study areas: median ratio, mean ratio, IQR ratio, standard deviation ratio, percentile range matching, and full quantile mapping. The median and

mean ratios produced excessively large constants ($c > 7$) due to the different distribution shapes of ACR (discrete symbol weights) and HCR (continuous, right-skewed). Percentile and quantile mapping over-corrected by forcing HCR to mimic ACR's distribution. The IQR ratio was selected for its robustness to outliers and its focus on matching variability rather than central tendency, adapting automatically to terrain characteristics. A straightforward alternative, normalising both rasters to $[0, 1]$, was rejected because it would discard the ACR's physically meaningful scale, where cell weights directly encode running speed ratios derived from IOF symbol specifications.

We propose an Interquartile Range (IQR) ratio:

$$c = \frac{\text{IQR}(\text{ACR})}{\text{IQR}(\text{HCR})} = \frac{Q_{75}^{\text{ACR}} - Q_{25}^{\text{ACR}}}{Q_{75}^{\text{HCR}} - Q_{25}^{\text{HCR}}} \quad (7)$$

where Q_p denotes the p -th percentile over valid cells. This matches the *variability* of HCR to ACR rather than aligning central tendencies.

The scaled HCR is added to ACR with path protection:

$$\text{Cost}_{rc} = w_{rc} + c \cdot \text{HCR}_{rc} \cdot \pi_{rc} \quad (8)$$

where the path protection factor π_{rc} prevents HCR from distorting well-mapped features:

$$\pi_{rc} = \begin{cases} 0.1 & \text{if } w_{rc} < 0.15 \text{ (paths/roads)} \\ 0.3 & \text{if } 0.15 \leq w_{rc} < 0.25 \text{ (good tracks)} \\ 0.0 & \text{if } w_{rc} \geq 5.0 \text{ (impassable)} \\ 1.0 & \text{otherwise} \end{cases} \quad (9)$$

The path protection factor attenuates HCR on cells where the cartographer has already provided precise cost information. Paths and roads are mapped with high positional accuracy in orienteering maps; DEM-derived roughness at these locations reflects the surrounding terrain rather than the path surface. The attenuation values (0.1 for paths, 0.3 for tracks) were calibrated to allow minimal HCR influence on linear features while preserving full contribution on open terrain, where map symbols provide only coarse vegetation categories.

3.4 Directional Metabolic Cost (Minetti Model)

The metabolic cost of locomotion on a slope gradient i is modelled by the fifth-degree polynomial of Minetti et al. [2002]:

$$C(i) = 280.5 i^5 - 58.7 i^4 - 76.8 i^3 + 51.9 i^2 + 19.6 i + 2.5 \quad [\text{J/kg/m}] \quad (10)$$

valid for $i \in [-0.45, +0.45]$. The cost multiplier relative to flat terrain is:

$$\phi(i) = \frac{\max(C(i), 0.5)}{C(0)} \quad (11)$$

where the floor at 0.5 prevents unrealistic negative costs on steep descents.

The directional slope between adjacent cells $(r_1, c_1) \rightarrow (r_2, c_2)$ is computed from the DEM-derived slope gradients. At each grid cell, the partial derivatives $\partial z / \partial x$ and $\partial z / \partial y$ are computed via central finite differences on the DEM. For a step between two adjacent cells, these derivatives are averaged to obtain a representative gradient at the midpoint:

$$\bar{g}_x = \frac{1}{2} \left(\left. \frac{\partial z}{\partial x} \right|_{(r_1, c_1)} + \left. \frac{\partial z}{\partial x} \right|_{(r_2, c_2)} \right), \quad \bar{g}_y = \frac{1}{2} \left(\left. \frac{\partial z}{\partial y} \right|_{(r_1, c_1)} + \left. \frac{\partial z}{\partial y} \right|_{(r_2, c_2)} \right) \quad (12)$$

Let $\Delta c = c_2 - c_1$ and $\Delta r = r_2 - r_1$ be the step direction. The signed slope gradient along the direction of travel is:

$$i_{12} = \frac{\bar{g}_x \cdot \Delta c + \bar{g}_y \cdot \Delta r}{\sqrt{\Delta c^2 + \Delta r^2}} \quad (13)$$

This projects the terrain gradient onto the movement vector, yielding a positive value for uphill travel and negative for downhill. Least-cost paths between control points are computed by running Dijkstra's algorithm over the grid of raster cells, where each cell is a graph node and each move to a neighbouring cell is a graph edge; the traversal cost of one such edge (a single "Dijkstra step"), from cell (r_1, c_1) to an adjacent cell (r_2, c_2) , is:

$$d_{12} = \|\Delta\| \cdot \frac{\text{Cost}_{r_1 c_1} + \text{Cost}_{r_2 c_2}}{2} \cdot \phi(i_{12}) \quad (14)$$

where Cost_{rc} is the combined base cost at cell (r, c) from eq. (8), $\phi(i_{12})$ is the Minetti slope factor from eq. (11), and $\|\Delta\| = \sqrt{\Delta c^2 + \Delta r^2}$ is the Euclidean step length between adjacent cells ($\|\Delta\| = 1$ for cardinal moves, $\|\Delta\| = \sqrt{2}$ for diagonal moves in the 8-connected grid). The base cost is averaged between the source and destination cells to smooth transitions across terrain boundaries.

This produces an asymmetric traversal cost: the step cost d_{12} from cell 1 to cell 2 differs from d_{21} in the reverse direction, because the Minetti factor satisfies $\phi(i) \neq \phi(-i)$ for any non-zero slope.

3.5 Asymmetric Cost Matrix Construction

Given n control points plus the depot (node 0), the pairwise cost matrix $\mathbf{C} \in \mathbb{R}^{(n+1) \times (n+1)}$ is computed by running anisotropic Dijkstra from each node on the combined cost grid (eq. (8)) with Minetti slope factors (eq. (11)). The per-step traversal cost used as the Dijkstra edge weight is given by eq. (14), which integrates the base cost raster, the directional Minetti multiplier, and the step geometry into a single directional cost per grid move.

For each node $i \in V$, a single-source anisotropic Dijkstra is run on the (optionally downsampled) grid, starting from the grid cell corresponding to node i . The algorithm explores all 8-connected neighbours using a priority queue, computing the traversal cost to each adjacent cell via eq. (14). Because each step cost is directional, the shortest-path cost from node i to node j generally differs from the cost from j to i , yielding an asymmetric cost matrix.

The cost matrix entry C_{ij} is then the shortest-path distance from node i 's grid cell to node j 's grid cell in this directed cost field. Running $n + 1$ independent Dijkstra computations (one per node) yields the full matrix.

The grid is optionally downsampled by factor s , averaging the base cost in $s \times s$ blocks and rescaling node coordinates accordingly. This reduces computation time from $O(nHW \log(HW))$ to $O(n \frac{HW}{s^2} \log \frac{HW}{s^2})$ at the cost of path resolution.

The resulting matrix satisfies $C_{ij} \neq C_{ji}$ in general, with asymmetry measured as:

$$\mathcal{A} = \frac{\text{mean}_{i \neq j} |C_{ij} - C_{ji}|}{\text{mean}_{i \neq j} C_{ij}} \times 100\% \quad (15)$$

3.6 Cumulative Fatigue Model

Fatigue state. We track a per-node fatigue state, accumulated along the path actually taken to reach i . Each arc (i, j) contributes a signed increment:

$$\psi_{ij} = \varphi_{ij}^+ - \rho \varphi_{ij}^- + \mu \delta_{ij} \quad (16)$$

where φ_{ij}^+ and φ_{ij}^- are the cumulative uphill and downhill elevation change (metres) along the cost-optimal path for arc (i, j) (the same path underlying C_{ij} , section 3.5), and δ_{ij} is the path

length (metres) along that same path. $\rho \in [0, 1]$ is the fraction of downhill elevation that offsets accumulated fatigue. $\mu \geq 0$ converts horizontal distance into vertical-metre-equivalent fatigue units; unlike ρ it is not a free parameter but is derived from the Minetti curve already underlying C_{ij} (eq. (11)):

$$\mu = \frac{c(0)}{\min_{i>0} c(i)/i} \quad (17)$$

i.e. the metabolic cost of one flat metre divided by the metabolic cost of the cheapest-to-reach vertical metre (the gradient i^* that minimises cost per unit of *vertical* rise). Both $\varphi_{ij}^+/\varphi_{ij}^-$ and δ_{ij} are accumulated as byproducts of the same anisotropic Dijkstra used to build C_{ij} , at negligible extra cost.

Fatigue cannot recover below zero. Writing $F_0 = 0$ at the depot,

$$F_j = \max(0, F_i + \psi_{ij}) \quad (18)$$

for the arc (i, j) actually taken to reach j .

The effective (fatigue-adjusted) cost of arc (i, j) departing node i at state F_i is:

$$C_{ij}^f(F_i) = C_{ij} \cdot (1 + \lambda F_i) \quad (19)$$

and the total fatigue-adjusted cost of a route $\sigma = (0, v_1, \dots, v_k, 0)$ is $D^f(\sigma) = \sum_{\ell} C_{v_{\ell}, v_{\ell+1}}^f(F_{v_{\ell}})$, with F propagated leg-by-leg via eq. (18).

Order-dependence. $D^f(\sigma)$ is not invariant to arc order. Expanding it (ignoring the clip in eq. (18)) splits it into $\sum_{\ell} C_{\ell}$, the sum of base costs, which is indifferent to order, plus a cross term $\lambda \sum_{m < \ell} C_{\ell} \psi_m$: for every pair of legs with m before ℓ , it multiplies the later leg’s cost C_{ℓ} by the fatigue ψ_m left behind by the earlier one. Swapping the order of two legs swaps which one is “earlier” in that product, turning $C_{\ell} \psi_m$ into $C_m \psi_{\ell}$; these coincide only if $\psi_{ij} \propto C_{ij}$ on every arc, since then both reduce to the same constant times $C_{\ell} C_m$. Here ψ_{ij} (elevation- and distance-driven, eq. (16)) is not proportional to C_{ij} (which also depends on the areal terrain weight, eq. (1)), so reordering the same arcs changes the total. The $\max(0, \cdot)$ clip is a second, independent source of order-dependence (a downhill-heavy prefix can floor the state in a way a later downhill leg cannot undo retroactively).

4 Mathematical Formulation

We formulate the Asymmetric Orienteering Problem with Fatigue (AOPF) as a mixed-integer linear program. However, the fatigue-adjusted cost $C_{ij}(1 + \lambda G_i)$ has the bilinear product $G_i \cdot x_{ij}$, which cannot be written directly as a linear constraint. We present two formulations that resolve this differently: an MTZ-based formulation (section 4.5) that linearises the product via McCormick envelopes, and a flow-based formulation (section 4.6) that avoids the product altogether by tracking fatigue with a dedicated flow variable per arc.

4.1 Sets and Parameters

Let $V = \{0, 1, \dots, n\}$ be the set of nodes, where node 0 is the depot. The parameters C_{ij} , ψ_{ij} , B , λ , ρ , and μ are as defined in sections 3 and 3.6. Each node i has a score $p_i \geq 0$ (with $p_0 = 0$).

4.2 Decision Variables

The two formulations developed below share arc variables $x_{ij} \in \{0, 1\}$ and node-visit variables $y_i \in \{0, 1\}$. They differ in how the fatigue state and budget are modelled.

4.3 Objective

Maximise total score:

$$\max \sum_{i \in V} p_i y_i \quad (20)$$

4.4 Shared Constraints

The constraints below are common to both the MTZ (section 4.5) and flow (section 4.6) formulations; each is then extended with its own fatigue-state and budget constraints.

Flow conservation. Each visited node has exactly one incoming and one outgoing arc:

$$\sum_{j \in V \setminus \{i\}} x_{ji} = y_i \quad \forall i \in V \quad (21)$$

$$\sum_{j \in V \setminus \{i\}} x_{ij} = y_i \quad \forall i \in V \quad (22)$$

Depot. The depot is always visited: $y_0 = 1$.

Arc elimination. Arcs that are structurally infeasible are fixed to zero a priori. An arc (i, j) is eliminated if the base cost alone exceeds the budget:

$$x_{ij} = 0 \quad \text{if } C_{ij} + C_{j0} > B \quad \text{or} \quad C_{0i} + C_{ij} + C_{j0} > B \quad (23)$$

It is not possible to add the fatigue into this equation as it requires an upper bound $\hat{G}_i$ on the fatigue state reachable at i , but $\hat{G}_i$ is only well-defined once the surviving-arc graph is fixed (eq. (24) below). We therefore retain only the base-cost filter (23); it is a conservative superset of the true survivors.

Fatigue-state bounds $\hat{G}_i$ and $\check{G}_i$. Both formulations below need a valid upper bound $\hat{G}_i$ on the (unclipped) fatigue state reachable at node i . Section 4.5 uses it to size the big- M constants in the state-propagation constraint; section 4.6 uses it to size the flow-coupling bounds.

The natural definition of $\hat{G}_i$ is the longest ψ -weighted walk from the depot to i . But unlike elapsed cost, ψ_{ij} can be negative, so the ψ -weighted surviving-arc graph (eq. (23)) can contain positive-weight cycles. To avoid solving it exactly we compute $\hat{G}_i$ with a Bellman–Ford relaxation in the $(\max, +)$ semiring, capped at n rounds so it can never exploit a cycle:

$$\hat{G}_i^{(0)} = \begin{cases} 0 & i = 0 \\ -\infty & \text{otherwise} \end{cases}, \quad \hat{G}_i^{(k)} = \max\left(\hat{G}_i^{(k-1)}, \max_{(u,i) \in A} \max(0, \hat{G}_u^{(k-1)} + \psi_{ui})\right), \quad \hat{G}_i := \hat{G}_i^{(n)} \quad (24)$$

Note that the floor $\max(0, \cdot)$ is applied after every round, not just once at the end, this mimics how true fatigue state behaves, whenever a leg would push it below zero, eq. (18) floors it back to zero, before the next leg is added. Flooring only once, at the end, would miss resets that happen partway through the walk, and could then report a bound below the state actually reached.

$\check{G}_i$, a valid *lower* bound needed only by Formulation B (below), is computed the same way but with a $(\min, +)$ relaxation, but this time constant clipping is not needed. Both bounds run in $O(n^3)$ ($O(n^2)$ arcs times $O(n)$ rounds), trivial for the instance sizes here.

Proposition 1 (Validity of the fatigue-state bounds). *For a walk $w = (v_0=0, v_1, \dots, v_k)$ in the surviving-arc graph, let $S_k(w) = \sum_{h=1}^k \psi_{v_{h-1}v_h}$ be its raw (unclipped) sum, and let $F_k(w)$ be the true clipped state it reaches under eq. (18) ($F_0(w) = 0$, $F_l(w) = \max(0, F_{l-1}(w) + \psi_{v_{l-1}v_l})$ for*

$l = 1, \dots, k$). Then, for every such walk of length $k \leq n$ from 0 to i : (a) $F_k(w) \leq \hat{G}_i^{(k)} \leq \hat{G}_i$, and (b) $\check{G}_i \leq S_k(w) \leq F_k(w)$. Consequently, for any simple route reaching i , its true clipped fatigue state lies in $[\check{G}_i, \hat{G}_i]$.

Proof. We start proving that $F_k(w) = \max_{0 \leq m \leq k} (S_k(w) - S_m(w))$. By induction on k . The base case $k = 0$ is immediate, $F_0 = 0 = S_0 - S_0$; for the step, $F_k(w) = \max(0, F_{k-1}(w) + \psi_{v_{k-1}v_k}) = \max(0, \max_{0 \leq m \leq k-1} (S_{k-1}(w) - S_m(w)) + \psi_{v_{k-1}v_k}) = \max_{0 \leq m \leq k} (S_k(w) - S_m(w))$, using $S_{k-1}(w) + \psi_{v_{k-1}v_k} = S_k(w)$ and folding the outer $\max(0, \cdot)$ in as the $m = k$ (empty-suffix) term.

Second, the map $x \mapsto \max(0, x + \psi)$ is non-decreasing in x for any fixed ψ , immediately from $x_1 \leq x_2 \Rightarrow x_1 + \psi \leq x_2 + \psi$.

With these in two properties, (a) $(F_k(w) \leq \hat{G}_i^{(k)} \leq \hat{G}_i)$ follows by induction on k , simultaneously over all nodes. The base case $k = 0$ is the trivial walk at 0, where $F_0 = 0 = \hat{G}_0^{(0)}$. For the step, let w end at $i = v_k$ via arc (u, i) , with $u = v_{k-1}$, and let w' be its length- $(k-1)$ prefix. The induction hypothesis gives $F_{k-1}(w') \leq \hat{G}_u^{(k-1)}$, so monotonicity of the clip gives

$$F_k(w) = \max(0, F_{k-1}(w') + \psi_{ui}) \leq \max(0, \hat{G}_u^{(k-1)} + \psi_{ui}) \leq \hat{G}_i^{(k)},$$

where the last step holds because $\hat{G}_i^{(k)}$'s defining maximum already includes this exact term (arc (u, i) is present since w uses it). Since $\hat{G}_i^{(k)}$ is non-decreasing in k by construction (each round takes a max with the previous value), $\hat{G}_i^{(k)} \leq \hat{G}_i^{(n)} = \hat{G}_i$, giving (a).

For (b) $(\check{G}_i \leq S_k(w) \leq F_k(w))$, applying the suffix-max identity with $m = 0$ we get that $F_k(w) \geq S_k(w) - S_0(w) = S_k(w)$. Since $\check{G}_i$ is a minimum over all walks $0 \rightarrow i$ of length $\leq n$ we also have $\check{G}_i \leq S_k(w)$. Chaining the two gives $\check{G}_i \leq S_k(w) \leq F_k(w)$. $\square$

4.5 Formulation A: MTZ with McCormick Linearisation

This formulation introduces fatigue-state variables $G_i \in [0, \hat{G}_i]$ at each node and auxiliary variables $u_{ij} \geq 0$ to linearise the bilinear fatigue term $G_i \cdot x_{ij}$.

State propagation (MTZ-type). For each arc (i, j) with $j \neq 0$:

$$G_j \geq G_i + \psi_{ij} - M_{ij}(1 - x_{ij}) \quad (25)$$

where the big- M constant is:

$$M_{ij} = \hat{G}_i + \max(\psi_{ij}, 0) \quad (26)$$

so that the constraint is vacuous when $x_{ij} = 0$ (the largest G_i can be is $\hat{G}_i$, and the largest ψ_{ij} can contribute is $\max(\psi_{ij}, 0)$). Because G_i additionally has a hard column floor of 0 and (25) is a one-sided inequality (not an equality), the LP is free to set G_j to any value satisfying every incoming constraint. The objective rewards visited nodes, not small G , so nothing here pushes G toward the minimal value the true clipped state would take. If this is not addressed the LP relaxation at a partially-fixed branch-and-cut node would settle on an inflated G_i that makes the budget constraint appear violated for an extension that the true (minimal) state would in fact permit, pruning the true optimum away before it is ever explored. We close this gap with a lexicographic regularisation term in the objective, $\max \sum_i p_i y_i - \varepsilon \sum_i G_i$ with ε small enough to never outweigh a single point of objective value.

McCormick linearisation. The fatigue budget involves the bilinear term $G_i \cdot x_{ij}$. We introduce $u_{ij} \approx G_i \cdot x_{ij}$ with the standard McCormick envelope, using $\hat{G}_i$ as the bound:

$$u_{ij} \geq G_i - \hat{G}_i(1 - x_{ij}) \quad (27)$$

$$u_{ij} \leq \hat{G}_i x_{ij} \quad (28)$$

$$u_{ij} \leq G_i \quad (29)$$

Constraint (27) forces $u_{ij} = G_i$ when $x_{ij} = 1$; constraint (28) forces $u_{ij} = 0$ when $x_{ij} = 0$; and constraint (29) ensures $u_{ij} \leq G_i$.

Fatigue budget (MTZ).

$$\sum_{(i,j) \in A} C_{ij} x_{ij} + \lambda \sum_{(i,j) \in A} C_{ij} u_{ij} \leq B \quad (30)$$

The McCormick constraints (27)–(29) are themselves the tightest possible convex relaxation of $u_{ij} = G_i \cdot x_{ij}$ for a single arc in isolation; the looseness instead enters upstream, through the state-propagation inequality (25). Because that constraint only lower-bounds G_j via a one-sided big- M relaxation, the LP is free to inflate G_i above the value any real path could produce, and that inflation then propagates through the (otherwise tight) envelope into u_{ij} and the budget constraint. This is the same mechanism by which the *unlifted*, big- M form of MTZ-style sequencing constraints is dominated by flow-conservation formulations for subtour elimination in the ATSP [Öncan et al., 2009]—the comparison does not extend to strengthened (lifted) MTZ reformulations, which Öncan et al. [2009] themselves show can match or exceed flow’s relaxation strength—here applied to a state variable rather than a sequencing one.

4.6 Formulation B: Single-Commodity Flow

This formulation replaces the MTZ state variables and McCormick auxiliary variables with flow variables g_{ij} . We define g_{ij} so that, by construction of the constraints below, $g_{ij} = G_i \cdot x_{ij}$ at any integer-feasible solution.

Flow-arc coupling. g_{ij} needs *both* an upper and a lower coupling bound:

$$\check{G}_i \cdot x_{ij} \leq g_{ij} \leq \hat{G}_i \cdot x_{ij} \quad \forall (i, j) \in A \quad (31)$$

Fatigue flow conservation. At each non-depot node j :

$$\sum_{i \in V} (g_{ij} + \psi_{ij} x_{ij}) = \sum_{k \in V} g_{jk} \quad \forall j \in V \setminus \{0\} \quad (32)$$

Depot outgoing flow is fixed to zero, $g_{0j} = 0$ for all j .

g_{ij} is defined per arc therefore nothing forces the state value on the arc going into j to match the state value on the arc going out, so without this restriction the solver could pick disconnected values on either side of j instead of one fatigue state that actually accumulates along the route.

Fatigue budget (flow).

$$\sum_{(i,j) \in A} C_{ij} x_{ij} + \lambda \sum_{(i,j) \in A} C_{ij} g_{ij} \leq B \quad (33)$$

The flow formulation has $O(n^2)$ additional continuous variables (one g_{ij} per arc) compared to $O(n)$ state variables in the MTZ formulation. Its flow-conservation constraint (32) ties g_{ij} to the state actually accumulated along the path used, rather than to a big- M -relaxed upper bound—this, not a difference in how the bilinear product itself is linearised, is what produces a tighter LP relaxation than the MTZ formulation (section 4.5).

4.7 Subtour Elimination and Connectivity Cuts

Both formulations use the same family of cutting planes, added dynamically during the B&C search. Because the cost matrix is asymmetric ($C_{ij} \neq C_{ji}$), all cuts are formulated on the directed arc variables x_{ij} . An inequality is valid for the AOPF if it is satisfied by every integer-feasible solution; adding valid inequalities to the LP relaxation tightens the upper bound without excluding any feasible solution.

Subtour elimination constraints (SECs). For a subset $S \subseteq V \setminus \{0\}$ (identified by the separation routines in section 4.9), the directed SEC prevents cycles disconnected from the depot:

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \quad (34)$$

Connectivity cuts. These ensure that every visited node can reach and be reached from the depot:

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq y_k \quad \forall k \in S \quad (35)$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq y_k \quad \forall k \in S \quad (36)$$

A combined two-way cut further tightens the relaxation:

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} + \sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 2y_k \quad \text{for some } k \in S \quad (37)$$

4.8 Budget-Based Cutting Planes

The cuts in this subsection exploit the budget constraint to eliminate LP solutions that are structurally infeasible due to excessive routing cost. They complement the subtour and connectivity cuts above, which enforce tour structure but do not account for cost. We present four families of budget-based cuts, numbered (B1)–(B4). The separation routines used to identify violated instances of each cut are described in section 4.9.

(B1) Lifted cover cuts. The fatigue budget constraint (eqs. (30) and (33)) limits the total cost of arcs used in the solution. Consider a subset of arcs $\mathcal{C} \subseteq A$ whose combined base costs exceed the budget: $\sum_{(i,j) \in \mathcal{C}} C_{ij} > B$. A feasible solution may use some arcs from $\mathcal{C}$, but it cannot use all of them simultaneously, as doing so would exceed the budget even before accounting for fatigue. Such a set $\mathcal{C}$ is called a *cover*, and the corresponding *cover inequality* states:

$$\sum_{(i,j) \in \mathcal{C}} x_{ij} \leq |\mathcal{C}| - 1 \quad (38)$$

That is, at least one arc in the cover must be excluded from any feasible solution. Each cover can be *lifted*: for arcs $e \notin \mathcal{C}$, a coefficient $\alpha_e \geq 0$ is computed such that adding $\alpha_e x_e$ to the left-hand side preserves validity, strengthening the inequality.

(B2) Routing infeasibility cuts. The cover cuts (B1) operate on arc variables x_{ij} , which the LP can soften through fractional values. We introduce a complementary family of inequalities defined purely on the node-visit variables y_i , exploiting the fact that certain subsets of nodes cannot all be visited within the budget regardless of the route chosen.

For a subset $Q \subseteq V \setminus \{0\}$, let $C_{\text{assign}}(Q)$ denote the optimal value of the assignment relaxation on the subgraph induced by $Q \cup \{0\}$: the minimum total base cost of a set of arcs where every node has exactly one incoming and one outgoing arc, solvable in $O(|Q|^3)$ by the Hungarian algorithm. Since fatigue only increases travel cost, $C_{\text{assign}}(Q)$ is a lower bound on the fatigue-adjusted cost of any cycle through $Q \cup \{0\}$.

Proposition 2 (Routing infeasibility). *For any $Q \subseteq V \setminus \{0\}$ with $C_{\text{assign}}(Q) > B$, the inequality*

$$\sum_{i \in Q} y_i \leq |Q| - 1 \quad (39)$$

is satisfied by every integer-feasible solution of the AOPF.

Proof. Suppose an integer-feasible solution has $y_i = 1$ for all $i \in Q$. The solution defines a cycle σ through the depot visiting a set W with $Q \subseteq W$. By definition, $D^f(\sigma) \geq C_{\text{assign}}(Q)$ since σ visits all nodes in $Q \cup \{0\}$ and $C_{\text{assign}}(Q)$ is a lower bound on the base cost of any such cycle, and fatigue only increases cost. But $D^f(\sigma) \leq B$ (budget feasibility), contradicting $C_{\text{assign}}(Q) > B$. $\square$

Unlike cover cuts (B1), which the LP can satisfy by reducing fractional arc values, routing infeasibility cuts constrain the y_i variables directly. The LP cannot circumvent them by redistributing arc flow, since the inequality involves no arc variables.

Remark. Propositions 2 and 3 (below) only need from $D^f(\sigma)$ the fact that fatigue never reduces cost, i.e. $D^f(\sigma) \geq \sum_{(i,j) \in \sigma} C_{ij}$. This holds here because $F_i \geq 0$ always, by construction (section 3.6), so $C_{ij}^f(F_i) = C_{ij}(1 + \lambda F_i) \geq C_{ij}$. Their proofs go through unchanged.

(B3) Directed cycle cover cuts. We adapt the cycle cover inequalities of Fischetti et al. [1998] from the undirected OP to our directed formulation. Let $F \subset A$ be a set of directed arcs forming a directed cycle through nodes $V(F)$ whose base cost $\sum_{(i,j) \in F} C_{ij}$ exceeds B .

Proposition 3 (Directed cycle cover). *For any directed cycle $F \subset A$ with $\sum_{(i,j) \in F} C_{ij} > B$, the inequality*

$$\sum_{(i,j) \in F} x_{ij} \leq \sum_{v \in V(F)} y_v - 1 \quad (40)$$

is satisfied by every integer-feasible solution of the AOPF.

Proof. Suppose for contradiction that an integer-feasible solution has $\sum_{(i,j) \in F} x_{ij} = \sum_{v \in V(F)} y_v$, i.e., every arc in F is used and every node in $V(F)$ is visited. Then the route contains all arcs of F , whose base cost alone exceeds B . Since arc costs are non-negative, $D^f(\sigma) \geq \sum_{(i,j) \in F} C_{ij} > B$, contradicting budget feasibility. Therefore $\sum_{(i,j) \in F} x_{ij} \leq \sum_{v \in V(F)} y_v - 1$ in every integer-feasible solution. $\square$

Inequality (40) is strictly stronger than the cover (B1) because the right-hand side involves the y_v variables: when the LP sets $y_v < 1$ for some $v \in V(F)$, the right-hand side decreases, forcing even fewer arcs to be used. Cover cuts (B1) remain useful because their separation is computationally cheaper.

(B4) Directed path inequality cuts. We adapt the path inequalities of Fischetti et al. [1998] to the directed asymmetric OP with fatigue. Let $P = (v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k)$ be a directed path through k non-depot nodes, where each $v_\ell \in V \setminus \{0\}$ is a node and the arcs $(v_\ell, v_{\ell+1})$ for $\ell = 1, \dots, k-1$ are traversed in sequence. Let $V(P) = \{v_1, \dots, v_k\}$ denote the node set of P . Propagate the exact (clipped) fatigue state deterministically along P using the base matrices: $\phi_1 = \max(0, \psi_{0,v_1})$ and $\phi_\ell = \max(0, \phi_{\ell-1} + \psi_{v_{\ell-1}, v_\ell})$ for $\ell = 2, \dots, k$, matching (18) exactly since P is a single fixed sequence with no branching. Let $\Phi(P) = C_{0,v_1} \cdot (1 + \lambda \cdot 0) + \sum_{\ell=1}^{k-1} C_{v_\ell, v_{\ell+1}}^f(\phi_\ell)$ be the fatigue-adjusted cost of traversing P from the depot (the state is 0 on departure, so the first leg is unpenalised). Define the *feasible closing set*:

$$W(P) = \left\{ v \in V \setminus V(P) : \Phi(P) + C_{v_k, v}^f(\phi_k) + C_{v, 0}^f(\max(0, \phi_k + \psi_{v_k, v})) \leq B \right\} \quad (41)$$

where $C_{i,j}^f(\cdot)$ is the fatigue-adjusted cost of arc (i, j) as defined in (19), evaluated at the deterministic state reached at each point along P then closing through v . The set $W(P)$ contains

exactly those nodes through which the path can be completed into a budget-feasible cycle. The path inequality states:

$$\sum_{\ell=1}^{k-1} x_{v_\ell, v_{\ell+1}} - \sum_{v \in V(P)} y_v + y_{v_1} + y_{v_k} - \sum_{v \in W(P)} x_{v_k, v} \leq 0 \quad (42)$$

Intuitively, if all path arcs are used (first sum is large), then either some interior path nodes are unvisited (second and third terms compensate), or the endpoint v_k connects to a node in $W(P)$ (last sum compensates). If neither holds, the path cannot be completed within budget. As in the original model, the fatigue-inflated return cost from v_k restricts $W(P)$ relative to the fatigue-free case, producing tighter cuts on high-fatigue instances; the effect is now asymmetric in ρ as well as in λ , since a downhill-heavy prefix leaves more headroom in ϕ_k than an uphill-heavy one of the same base cost.

4.9 Separation Routines

The valid inequalities defined in the preceding subsections are added dynamically during the B&C search. At each node of the search tree, the solver iterates through a cut loop that attempts to find violated inequalities in the current LP solution (y^*, x^*, f^*) . The loop terminates when no violated cuts are found or a maximum number of iterations is reached. Below we describe how violated instances of each cut family are identified.

Subtour and connectivity cuts. A directed support graph $H^* = (V^*, A^*)$ is constructed from arcs with $x_{ij}^* > 0.5$. Two complementary procedures are applied:

1. *Kosaraju strongly connected component (SCC) detection.* Kosaraju’s algorithm identifies all strongly connected components of H^* in $O(|V^*| + |A^*|)$ time. Any SCC not containing the depot yields a violated subset S , and all four cut types (eqs. (34) to (37)) are added for S .
2. *Reverse breadth-first search (BFS).* A reverse BFS from the depot identifies nodes that can reach the depot in H^* . Any visited node ($y_i^* > 0.5$) not in this reachable set requires an outgoing connectivity cut.

This two-phase separation is essential for the asymmetric OP: Kosaraju detects closed subtours, while the reverse BFS catches paths that leave the depot but cannot return.

Max-flow connectivity cuts. The rounding-based separation above (using threshold $x_{ij}^* > 0.5$) may miss violated connectivity cuts when the fractional arc values are spread thinly. To strengthen the separation, for each visited node k with $y_k^* > 0.1$, a max-flow computation (Dinic’s algorithm) is performed from the depot to k on the fractional support graph, using x_{ij}^* as arc capacities. If the max-flow value is less than y_k^* , the min-cut identifies a subset S with $k \in S$ and $0 \notin S$ for which the connectivity cut (35) is violated. This exact separation complements the heuristic rounding-based approach and is particularly effective on asymmetric instances where fractional flow is distributed across many arcs.

(B1) Cover cuts. Arcs with $x_{ij}^* > 0$ are sorted by decreasing x_{ij}^* and accumulated greedily until their total base cost exceeds B . The resulting set $\mathcal{C}$ defines a violated cover if $\sum_{(i,j) \in \mathcal{C}} x_{ij}^* > |\mathcal{C}| - 1$, i.e., the LP uses more of these arcs than any feasible solution allows. Each violated cover is optionally strengthened via sequential lifting. Up to 3 cover cuts are added per cut-loop iteration; adding more would slow the LP solve without proportional improvement in the bound. To account for fatigue, the arc weights used in the greedy accumulation are inflated from the base cost C_{ij} to $C_{ij}(1 + \lambda \max(\check{G}_i, 0))$: the true clipped state satisfies $F_i \geq \check{G}_i$ always (as $\check{G}_i$ is a valid lower bound on the unclipped state, eq. (24)), and $F_i \geq 0$ trivially, so $\max(\check{G}_i, 0)$ is

the tightest simple lower bound on the minimum possible fatigue state at i available without solving for the specific route. This produces smaller covers yielding tighter inequalities, and is only useful (multiplier > 1) on arcs where $\tilde{G}_i$ is positive, i.e. where every budget-admissible route reaching i is net uphill.

(B2) Routing infeasibility cuts. Nodes with $y_i^* > 0.5$ are sorted by decreasing y_i^* and added one by one to a candidate set Q . After each addition, the assignment bound $C_{\text{assign}}(Q)$ is computed via the Hungarian algorithm ($O(|Q|^3)$). When $C_{\text{assign}}(Q) > B$ and $\sum_{i \in Q} y_i^* > |Q| - 1$, a violated routing infeasibility cut has been found. The set Q is then minimised: nodes are removed in order of increasing y_i^* while $C_{\text{assign}}(Q)$ remains above B , yielding a minimal infeasible set and hence the tightest cut.

(B3) Cycle cover cuts. Short directed cycles (length 3–4) are enumerated in the support graph H^* . For each cycle F , the base cost $\sum_{(i,j) \in F} C_{ij}$ is computed. If $\sum_{(i,j) \in F} C_{ij} > B$ and the inequality (40) is violated by (x^*, y^*) , the cut is added.

(B4) Path inequality cuts. Directed paths of length 2–3 are enumerated in H^* . For each path $P = (v_1 \rightarrow \dots \rightarrow v_k)$, the feasible closing set $W(P)$ is computed using the base matrices C_{ij}, ψ_{ij} (not the LP solution values): the fatigue state at each path node is accumulated deterministically via the clip recursion $\phi_\ell = \max(0, \phi_{\ell-1} + \psi_{v_{\ell-1}, v_\ell})$, and the fatigue-adjusted cost of each subsequent arc is evaluated at this state, as in (41). If $\Phi(P)$ (the path’s own fatigue-adjusted cost) already exceeds B , $W(P)$ is empty, or a direct return $v_k \rightarrow 0$ is itself budget-feasible (in which case the path inequality would add nothing, since closing directly is always available), the path is skipped. Otherwise the cut is added if inequality (42) is violated by the current LP solution.

5 Solution Methods

5.1 Simulated Annealing (SA)

Method selection. The SA metaheuristic was selected after a comparative study on the full 66-instance benchmark suite (section 8.3) across varying sizes ($n = 20$ – 200), asymmetry levels, and budget tightness, at the fixed calibration $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$ (section 7). Four methods were evaluated with equal computational budget (2 seconds per method per instance, 10 independent seeds for stochastic methods): Greedy Insertion (deterministic), Genetic Algorithm (GA, population 50), Ant Colony Optimisation (ACO, 30 ants), and Simulated Annealing (SA).

Table 1: Heuristic comparison: mean gap below SA (%) across 66 benchmark instances. Significance is an exact Wilcoxon signed-rank test (`scipy.stats.wilcoxon`) on the 66 paired instance means, Holm-Bonferroni corrected across the 3 comparisons against SA; effect size is the matched-pairs rank-biserial correlation r .

Method	Mean gap	Std	Min gap	Max gap	$p_{\text{Holm}} (r)$
Greedy	19.0%	8.8%	3.4%	40.0%	4.9×10^{-12} (1.00)
GA	54.1%	14.8%	15.7%	81.1%	4.9×10^{-12} (1.00)
ACO	8.6%	5.1%	0.0%	22.9%	4.9×10^{-12} (1.00)
SA	—	—	—	—	—

SA significantly outperforms all three alternatives under equal time budgets: an exact Wilcoxon signed-rank test on the 66 paired instance means, Holm-Bonferroni corrected across the 3 com-

parisons, gives $p_{\text{Holm}} \approx 4.9 \times 10^{-12}$ for each, with a rank-biserial effect size of $r = 1.00$ in every case.

SA’s mean score exceeds every competitor’s on all 66 instances with no ties. The GA performs worst (54.1% below SA on average), as crossover on directed routes frequently produces infeasible offspring requiring expensive repair, and the rugged fitness landscape under asymmetric costs degrades convergence. ACO is the closest competitor (8.6% below SA) but its pheromone mechanism weakens on asymmetric instances where edge quality is context-dependent on elapsed time and travel direction. Greedy Insertion (19.0% below SA) is fast but myopic, committing to locally attractive nodes without ability to backtrack.

SA’s performs better because it evaluates each candidate move in full context without needing to decompose the solution into reusable components. The Or-opt and swap moves are particularly effective on asymmetric instances because they directly evaluate the actual cost change of rearranging the visit sequence, accounting for directionality.

Parameter tuning. The SA parameters were fine-tuned via grid search over a separate set of 1,000 synthetic instances with randomised control-point distributions the benchmark areas. The tunable parameters and their search ranges were:

- Initial temperature $T_0 \in \{50, 100, 200, 500\}$
- Final temperature $T_{\text{end}} \in \{0.01, 0.1, 1.0\}$
- Number of iterations $N_{\text{iter}} \in \{20000, 40000, 80000, 120000\}$
- Number of restarts $N_{\text{restart}} \in \{4, 8, 12, 20\}$
- Move probabilities $(p_{\text{remove}}, p_{\text{insert}}, p_{\text{or-opt}}, p_{\text{swap}})$, tested in 10 configurations
- Or-opt segment length $L_{\text{seg}} \in \{1-2, 1-3, 1-4\}$
- Stagnation limit as fraction of N_{iter} : $\{1/3, 1/4, 1/5, 1/6\}$

The best configuration, used throughout the experiments, is: $T_0 = 100$, $T_{\text{end}} = 0.1$, move probabilities $(0.30, 0.30, 0.20, 0.20)$, $L_{\text{seg}} \in \{1, 2, 3\}$, and stagnation limit $N_{\text{iter}}/5$.

The cooling schedule is geometric: $T_k = T_0 \cdot \eta^k$ with decay rate $\eta = (T_{\text{end}}/T_0)^{1/N_{\text{iter}}}$.

Neighbourhood structure. Starting from a greedy insertion solution, the SA applies four neighbourhood moves with the tuned probabilities:

1. **Remove** ($p = 0.30$): delete a random node from the route.
2. **Insert** ($p = 0.30$): add an unvisited node at its best position (minimum cost increase, computed in $O(k)$ where k is the current route length).
3. **Or-opt** ($p = 0.20$): extract a segment of 1–3 nodes and reinsert at a different position.
4. **Swap** ($p = 0.20$): exchange a visited node for an unvisited one.

Moves that violate the fatigue budget ($D^f(\sigma') > B$) are rejected. The search terminates early if no improvement is found for $N_{\text{iter}}/5$ consecutive iterations (stagnation detection).

The full procedure, including the iterated multi-restart wrapper, is given in algorithm 1.

Algorithm 1 Iterated Simulated Annealing for AOPF

Require: Cost matrix $\mathbf{C}$, scores $\mathbf{p}$, budget B , fatigue rate λ **Ensure:** Best route σ^*

```
1:  $\sigma^* \leftarrow \emptyset, z^* \leftarrow 0$ 
2: Compute  $N_{\text{iter}} \leftarrow \text{clip}(2500n, 10000, 120000)$ 
3: Compute  $N_{\text{restart}} \leftarrow \text{clip}(3000/n, 40, 200)$ 
4: for  $r = 1, \dots, N_{\text{restart}}$  do
5:    $\sigma \leftarrow \text{GREEDYINSERTION}(\mathbf{C}, \mathbf{p}, B)$ 
6:    $z \leftarrow \sum_{i \in \sigma} p_i, T \leftarrow T_0, \eta \leftarrow (T_{\text{end}}/T_0)^{1/N_{\text{iter}}}, s \leftarrow 0$ 
7:   for  $k = 1, \dots, N_{\text{iter}}$  do
8:     if  $s \geq N_{\text{iter}}/5$  then break ▷ Stagnation
9:     end if
10:     $T \leftarrow T \cdot \eta$ 
11:    Draw  $u \sim \text{Uniform}(0, 1)$ 
12:    if  $u < 0.30$  then
13:       $\sigma' \leftarrow \text{REMOVE}(\sigma)$  ▷ Delete random node
14:    else if  $u < 0.60$  then
15:       $\sigma' \leftarrow \text{INSERT}(\sigma)$  ▷ Add node at best position
16:    else if  $u < 0.80$  then
17:       $\sigma' \leftarrow \text{OROPT}(\sigma)$  ▷ Relocate segment of 1–3 nodes
18:    else
19:       $\sigma' \leftarrow \text{SWAP}(\sigma)$  ▷ Exchange visited  $\leftrightarrow$  unvisited
20:    end if
21:    if  $D^f(\sigma') > B$  then
22:       $s \leftarrow s + 1$ ; continue ▷ Infeasible move
23:    end if
24:     $z' \leftarrow \sum_{i \in \sigma'} p_i, \Delta \leftarrow z' - z$ 
25:    if  $\Delta > 0$  or  $\exp(\Delta/T) > \text{Uniform}(0, 1)$  then
26:       $\sigma \leftarrow \sigma', z \leftarrow z'$ 
27:      if  $z > z^*$  then  $z^* \leftarrow z, \sigma^* \leftarrow \sigma, s \leftarrow 0$ 
28:      else  $s \leftarrow s + 1$ 
29:      end if
30:    else  $s \leftarrow s + 1$ 
31:    end if
32:  end for
33: end for
34: return  $\sigma^*$ 
```

5.2 Branch-and-Cut (B&C)

The B&C solver uses the LP relaxation of the AOPF formulation (section 4), solved via the HiGHS dual simplex [Huangfu and Hall, 2018]. HiGHS was chosen as it is open-source, provides a C API suitable for embedding in a custom B&C loop, and its dual simplex implementation supports efficient warm-starting after cut addition. The algorithm proceeds as follows:

Algorithm 2 Branch-and-Cut for AOPF

```
1: Build root LP relaxation
2: Warm-start incumbent  $z^* \leftarrow$  SA solution value
3: Push root node onto stack
4: while stack non-empty and time  $< T_{\max}$  do
5:   Pop node, apply variable fixings
6:   for  $k = 1, \dots, K_{\max}$  do ▷ Cut loop,  $K_{\max} = 20$ 
7:     Solve LP relaxation
8:     if  $\bar{z} \leq z^* + \epsilon$  then prune and continue
9:     end if
10:    Find violated SECs via Kosaraju SCC
11:    Find depot-unreachable nodes via reverse BFS
12:    Find violated max-flow connectivity cuts
13:    Find violated budget-based cuts (B1–B4, section 4.9)
14:    if no violations found then break
15:    end if
16:    Add all violated cuts to LP
17:  end for
18:  if  $y$  and  $x$  are all integral then
19:    Extract route, validate fatigue feasibility
20:    if score  $> z^*$  then update incumbent
21:    end if
22:  else
23:    Branch on most fractional  $y_i$ ; if  $y$  is already integral, branch on most fractional  $x_{ij}$ 
    instead
24:    Push child nodes (DFS order)
25:  end if
26: end while
27: if stack empty then solution is provably optimal
```

Key implementation details:

- Branching primarily targets the node-visit variables y_i rather than arc variables x_{ij} . Setting $y_i = 0$ removes node i from all future solutions in the subtree, eliminating $O(n)$ arc variables at once; setting $y_i = 1$ forces the node to be visited. By contrast, fixing a single x_{ij} only excludes one arc, requiring up to $O(n^2)$ branching decisions to resolve all fractional arcs.
- y -integrality alone does not certify a genuine route: two fractional sub-routings can share the same integral y -pattern without yet being separated by any SEC or connectivity cut found so far, and route extraction would silently misroute in that case. A leaf is therefore only accepted once y and x are both fully integral; if y is integral but some x_{ij} is not, the solver falls back to branching on the most fractional x_{ij} , using the same fixing mechanism as for y .
- LP models are deep-copied via `Highs_passLp` for thread safety.
- Duplicate SECs are tracked via a hash set to avoid redundant cuts.
- Maximum branching depth is capped at $D_{\max} = 15$.
- Time limit $T_{\max} = 900$ s (15 minutes).

5.3 Node Selection Strategy

The B&C algorithm admits two complementary node selection strategies, and we employ each on the instance class where it is most effective. *Depth-first search* (DFS) maintains nodes on a stack and always processes the most recently pushed child; it descends quickly into promising

subtrees, finds integer-feasible improvements early, and prunes the remaining tree once a strong incumbent is established. *Best-first search* (BFS) maintains nodes on a priority queue ordered by the LP upper bound and always processes the node most likely to improve the global bound; it minimises the number of nodes examined to close the optimality gap at the cost of slower incumbent discovery. This tradeoff between depth-first and best-first node selection is a well-studied aspect of branch-and-bound design, Linderoth and Savelsbergh [1999] provide a comprehensive computational study of search strategies for mixed integer programming and document that the relative effectiveness of DFS and BFS depends on instance characteristics such as bound quality and tree size.

For the real-terrain instances ($n \leq 40$) we use DFS, and for the benchmark suite (up to $n = 100$) we use BFS. In preliminary experiments the DFS configuration proved more terrain instances optimal, while the BFS configuration yielded lower mean optimality gaps on the larger benchmark problems. Both configurations share the same formulation, cut families, and warm-start heuristic.

Both strategies use the same formulation, the same cut families, and the same warm-start heuristic; only the node selection rule and the strong-branching parameters differ. This allows a clean decomposition of the gains from formulation, cuts, and search strategy, as reported in the results below.

6 Computational Pipeline

The full pipeline consists of two stages:

Stage 1: Cost Surface Generation (Python).

1. Parse OMAP file $\rightarrow$ rasterise symbols onto UTM grid $\rightarrow$ ACR.
2. Load DTM, compute TRI, PC, SLO $\rightarrow$ normalise $\rightarrow$ HCR (eq. (6)).
3. Scale HCR via IQR ratio (eq. (7)), merge with ACR (eq. (8)).
4. Compute directional slopes from DTM.
5. Run anisotropic Dijkstra from each node with Minetti factors.
6. Export asymmetric cost matrix, scores, and budget as JSON.

Stage 2: Route Optimisation (C++).

1. Parse JSON input.
2. Run iterated SA $\rightarrow$ warm-start solution.
3. Build LP model with HiGHS, run B&C (algorithm 2).
4. Report solution, optimality status, and timing.

7 Experimental Setup

7.1 Study Areas

Table 2: Terrain characteristics of the two study areas.

	Torremocha	La Muela
Grid size	1056×1463	1376×1622
Resolution	2.0 m	2.0 m
Area	2.1×2.9 km	2.8×3.2 km
Elevation range	447–501 m	1164–1628 m
Relief	55 m	464 m
Median slope	3.2°	15.9°
Max slope	58.9°	81.2°
CRS	EPSG:25829	EPSG:25830
IQR scaling c	0.00	1.09
HCR median increase	+0.0%	+9.3%
Cost asymmetry	17.5%	36.6%

Torremocha is a gently undulating terrain in Extremadura, Spain, with moderate forest cover and extensive rock features. La Muela is a mountainous area with steep slopes, cliffs, and significant elevation variation.

7.2 Control Distributions

Seven control-point distributions are generated for each terrain to test solver robustness:

1. **Standard**: grid-zone uniform spread (40 controls).
2. **Clustered**: 5 hotspot clusters (35 controls).
3. **Ring**: concentric rings from depot (30 controls).
4. **Path-biased**: 60% near paths, 40% far (35 controls).
5. **Elevation-biased**: weighted toward high elevation (35 controls).
6. **Sparse-far**: few controls, all far from depot (25 controls).
7. **Mixed-density**: dense near depot, sparse far (40 controls).

Point values (10–50) are assigned based on terrain difficulty at each control location.

7.3 Parameters

Race duration: 1 hour. Fatigue rate: $\lambda = 4.1 \times 10^{-5}$ (derivation below). Base speed: 2.5 m/s. Dijkstra downsampling: $s = 8$. B&C time limit: 900 s. SA iterations: auto-calibrated per instance. These values reflect the typical duration of a middle-distance orienteering race.

Fatigue-rate calibration. λ multiplies the fatigue state F_i (section 3.6), which is expressed in metres of elevation and distance, so it must be small for $1 + \lambda F_i$ to stay in a physically sensible range.

No literature value shows a coefficient on the fatigue state F_i itself (eq. (16)), combining net elevation and distance via μ , not elevation alone. Most exercise-physiology and trail-running sources report only whole-race pace loss without a pinned source, or fatigue as a function of instantaneous power rather than cumulative elevation (e.g. Jaszczak and Płociniczak [2024]’s rate constant ($K = 6 \times 10^{-5} \text{s}^{-1}$)). However, Jaén-Carrillo et al. [2025], reports a mean 18.7% decline in grade-adjusted pace from race start to the final descent among 12 elite finishers of

the Trail Short event at the 2023 World Mountain and Trail Running Championships (44.6 km, $D^+ = 3,132$ m; $D^- \approx D^+$).

ρ (the downhill-recovery fraction in eq. (16)) is fixed to $\rho = 0$. The literature we found leans toward no net recovery, Vernillo et al. [2015] measured energy cost before and after a mountain ultramarathon in 14 trail runners and found downhill running cost increased by 13.1% post-race. μ (the distance-to-elevation weight, eq. (17)) is derived from the Minetti curve already used for C_{ij} ; for the terrain parameters in this study it evaluates to $\mu \approx 0.067$.

As a rough order-of-magnitude anchor we use Jaén-Carrillo et al.’s race, where both D^+ and total distance are known, and compute F via eq. (16): $F = \varphi^+ - \rho \varphi^- + \mu \delta \approx 3,132 - 0.5(3,132) + 0.067(44,600) \approx 4,568$, and $1 + \lambda F \approx 1.187$ at that F gives $\lambda \approx 4.1 \times 10^{-5}$. Given the absence of a defensible point estimate, this is the value we fix λ to for all experiments reported below.

8 Results

We evaluate the framework on the two real-terrain study areas (Torremocha, 17.5% cost asymmetry; La Muela, 36.6% cost asymmetry) and on the synthetic benchmark suite. For each instance, the SA warm-start solution seeds the B&C’s initial incumbent; the B&C then searches for a better integer-feasible solution while tightening the LP relaxation bound, within a 15-minute time limit. Both formulations (MTZ and flow) receive the same SA incumbent, but each runs its own independent search from there, so the two can (and sometimes do) end on different final point totals. Each table therefore reports two distinct quantities per formulation: the **pts** column is the B&C’s own final incumbent z^* —not necessarily equal to the SA points shown separately, since the exact search sometimes improves on the heuristic—and the *gap* column is the remaining **optimality** gap, $\text{gap} = 100 \times (\bar{z} - z^*)/z^*$, where $\bar{z}$ is the best LP relaxation bound the search reached. A gap of 0.0% with “YES”/“proven” means the B&C has certified its own final incumbent z^* —whatever value it settled on, whether or not that improved on SA—as the true global optimum; a nonzero gap means the time limit was reached with $\bar{z}$ still strictly above z^* , a valid but unproven upper bound.

8.1 Torremocha Results

Table 3: Torremocha results: SA and B&C under both formulations (IQR scaling $c = 0.00$, see section 9 on why ACR’s interquartile range is degenerate here). B&C uses depth-first search (see section 5.3); gap is the LP-relaxation gap defined above. $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$ throughout (section 7).

Distribution	ctrls	SA		MTZ		Flow	
		pts	time (s)	pts	gap	pts	gap
Standard	40	1090	1.6	1090	12.8% (no)	1090	12.7% (no)
Clustered	35	630	1.2	630	0.0% (YES)	630	0.0% (YES)
Ring	30	720	1.3	720	0.0% (YES)	720	0.0% (YES)
Path-biased	21	230	1.2	230	0.0% (YES)	230	0.0% (YES)
Elev-biased	35	670	1.5	670	0.0% (YES)	670	0.0% (YES)
Sparse-far	25	940	1.1	940	0.0% (YES)	940	0.0% (YES)
Mixed-density	40	950	1.7	980	0.0% (YES)	980	0.0% (YES)

Both formulations prove 6 of 7 instances optimal, failing only on Standard (a 41-node instance, the largest on this terrain), where both reach the 900 s time limit at a near-identical gap (12.8% MTZ vs. 12.7% flow). On Mixed-density, both formulations improve on the SA incumbent (950

→ 980) and prove the same route optimal. On every other instance, the exact solvers only certify the SA solution as optimal without improving it.

8.2 La Muela Results

Table 4: La Muela results: SA and B&C under both formulations (IQR scaling $c = 1.09$, cost asymmetry $\mathcal{A} = 36.6\%$). B&C uses depth-first search (see section 5.3); gap is the LP-relaxation gap defined above. $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$ throughout (section 7).

Distribution	ctrls	SA		MTZ		Flow	
		pts	time (s)	pts	gap	pts	gap
Standard	31	750	2.1	750	9.9% (no)	790	0.0% (YES)
Clustered	35	960	2.3	960	3.1% (no)	990	0.0% (YES)
Ring	30	690	2.0	700	2.9% (no)	710	0.0% (YES)
Path-biased	25	380	2.4	410	0.0% (YES)	410	0.0% (YES)
Elev-biased	35	730	2.5	730	9.6% (no)	730	8.8% (no)
Sparse-far	25	1020	1.7	1060	0.0% (YES)	1060	0.0% (YES)
Mixed-density	40	980	2.4	980	7.2% (no)	980	6.7% (no)

On this harder, more asymmetric terrain, flow proves more instances optimal than MTZ (5 of 7 versus 2 of 7): flow certifies Standard, Clustered, and Ring outright (790, 990, and 710 points respectively), where MTZ only reports an unproven incumbent at a small residual gap (9.9%, 3.1%, and 2.9%). Both formulations agree exactly on Path-biased and Sparse-far (both proven, 410 and 1060 points), and on the two hardest instances, Elev-biased and Mixed-density, both remain unproven for both formulations, with flow’s gap again at or below MTZ’s in each case (8.8% vs. 9.6%; 6.7% vs. 7.2%).

Across both terrains (14 instances total), flow proves 11 of 14 instances optimal versus 8 for MTZ. The mean LP-relaxation gap on the unproven instances is higher for flow (9.4% across its 3 unproven instances versus 7.6% across MTZ’s 6 unproven instances).

8.3 Synthetic Benchmark Instances

To the best of our knowledge, no standard benchmark exists for the asymmetric OP with fatigue. We generate a parameterized benchmark suite of 66 instances (3 random seeds per configuration) by varying three dimensions independently from a baseline configuration ($n = 40$, $\alpha = 0.25$, medium budget):

- Node count $n \in \{20, 30, 40, 50, 75, 100, 125, 150, 200\}$
- Asymmetry parameter $\alpha \in \{0.0, 0.1, 0.25, 0.5\}$, controlling directional cost perturbation
- Budget tightness: loose ($\sim 70\%$), medium ($\sim 50\%$), tight ($\sim 30\%$), defined as a fraction of the nearest-neighbour tour cost

Additionally, six extreme-case instances test boundary conditions (e.g., symmetric with a loose budget, highly asymmetric with a tight budget), and two size $\times$ asymmetry combinations outside the main size sweep. Nodes are placed uniformly at random within the domain of the map. Asymmetric costs are generated by perturbing Euclidean distances with a directional slope factor proportional to α , plus 10% random terrain noise. Scores are correlated with distance from the depot (10–50 points, rounded to multiples of 10). Note that the synthetic asymmetry is spatially uncorrelated, while in real terrain the slope-induced cost differences exhibit spatial autocorrelation. The real-terrain experiments on Torremocha and La Muela complement the benchmark by testing the solver on naturally structured asymmetry.

Instance file names additionally encode a nominal fatigue-rate label (f00/f10/f20/f30) from the generation script. Following the fixed-calibration policy adopted in section 7 ($\lambda = 4.1 \times 10^{-5}$

from Jaén-Carrillo et al. [2025], $\rho = 0$ from Vernillo et al. [2015], applied uniformly), every instance below is solved at the same λ and ρ regardless of this label. We therefore do not report a fatigue-rate sweep as a separate results axis: it would not reflect a real experimental contrast, since all instances share one λ .

8.4 Benchmark Results

Both formulations (MTZ and flow) are evaluated on the full benchmark suite under identical conditions (same SA warm start, $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$, 300 s per-node time limit). Gaps and optimal-count columns are averaged over the 3 seeds of each configuration; “optimal?” reports how many of the 3 seeds the B&C proved optimal.

Table 5: Benchmark results: effect of instance size, asymmetry, and budget tightness. Each block varies one dimension from the baseline configuration.

	SA pts	MTZ			Flow		
		pts	gap	optimal?	pts	gap	optimal?
<i>Node count n ($\alpha = 0.25$, medium budget)</i>							
20	337	360	0.0%	3/3	360	0.0%	3/3
30	477	530	0.0%	3/3	530	0.0%	3/3
40	650	690	9.4%	1/3	713	0.0%	3/3
50	897	900	20.9%	0/3	940	7.1%	1/3
75	1130	1130	46.0%	0/3	1130	39.4%	0/3
100	1613	1613	37.5%	0/3	1613	33.5%	0/3
125	2177	2177	28.9%	0/3	2177	26.8%	0/3
150	2537	2537	32.6%	0/3	2537	31.5%	0/3
200	2793	2793	49.6%	0/3	2793	48.6%	0/3
<i>Asymmetry α ($n = 40$, medium budget)</i>							
0.00	677	717	10.3%	1/3	713	2.7%	2/3
0.10	737	803	17.3%	1/3	830	5.0%	2/3
0.25	653	660	33.4%	0/3	747	3.7%	2/3
0.50	607	637	26.6%	0/3	693	7.5%	2/3
<i>Budget tightness ($n = 40$, $\alpha = 0.25$)</i>							
Loose	873	873	24.1%	0/3	1020	0.0%	2/3
Medium	670	717	9.0%	2/3	717	2.1%	2/3
Tight	367	410	6.9%	2/3	410	0.0%	3/3

The benchmark reveals that size drives difficulty far more than any other dimension. Both formulations prove every seed optimal up to $n = 30$, flow also closes all of $n = 40$ and one seed at $n = 50$, but nothing at $n \geq 75$ is proven within the time limit, and the mean gap does not decay monotonically with n (flow: 0.0% at $n = 40$ rising to 48.6% at $n = 200$, with a local dip to 26.8% at $n = 125$). It can also be noted that flow’s LP relaxation is tighter than MTZ’s at every asymmetry level, most sharply at $\alpha = 0.25$ (33.4% vs. 3.7%), and flow proves more seeds optimal uniformly across all four levels (2/3 in every case, versus MTZ’s 1/3 at $\alpha \in \{0, 0.1\}$ and 0/3 at $\alpha \in \{0.25, 0.5\}$). Budget tightness shows a clear monotone trend for MTZ (24.1% $\rightarrow$ 9.0% $\rightarrow$ 6.9% as the budget tightens) but a non-monotone one for flow, whose gap reaches 0.0% at both the loose and tight extremes yet rises to 2.1% at the medium setting.

Table 6: Benchmark results: extreme/boundary-case instances, $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$ throughout

Instance	SA pts	MTZ			Flow		
		pts	gap	optimal?	pts	gap	optimal?
Symmetric, loose budget ($n=20$, $\alpha=0$)	580	580	0.0%	YES	580	0.0%	YES
Asymmetric, tight budget ($n=100$, $\alpha=0.5$)	830	830	42.4%	no	830	37.4%	no
Asymmetric, medium size ($n=50$, $\alpha=0.5$)	920	920	7.0%	no	920	2.5%	no
Symmetric, medium size ($n=50$, $\alpha=0$)	880	910	33.5%	no	880	27.0%	no
Asymmetric, tight budget, large ($n=200$, $\alpha=0.5$)	1930	1930	61.5%	no	1930	59.2%	no
Symmetric, loose budget, large ($n=150$, $\alpha=0$)	3740	3740	3.1%	no	3740	2.6%	no

Among the extreme cases, only the smallest, least constrained instance (symmetric $n = 20$, loose budget) is proven optimal by both formulations; every other case remains open, with flow’s gap consistently at or below MTZ’s. This is not a weakness of the McCormick linearisation itself (section 4.5), but of the big- M state-propagation constraint feeding it: MTZ’s G_i can be inflated above what any real path produces, whereas flow’s conservation constraint ties g_{ij} to the state actually accumulated along the route (see section 9 for further details).

8.5 Ablation Study: Effect of Cut Families

To quantify the marginal contribution of each cut family introduced in section 4.9, we evaluate five cumulative configurations of the flow-based B&C solver on the full benchmark suite of 66 instances, following the component evaluation methodology of Kobeaga et al. [2021]. Each configuration adds one cut family to the previous one, using the same SA warm start, the fixed $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$ of section 7, and a 900 s time limit. Table 7 reports the mean optimality gap and number of instances proven optimal for each configuration.

Table 7: Ablation study on the benchmark suite (66 instances, 900 s time limit, $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$). Each row adds one cut family to the previous configuration. B&C uses best-first search (see section 5.3). Gap is $100 \times (\bar{z} - z^*)/z^*$.

Cuts enabled		Mean gap	Max gap	Opt.
1	Base (SECs + conn. + tight. + fat.)	12.02%	59.23%	37/66
2	+ B1 (lifted covers)	11.68%	60.05%	37/66
3	+ B2 (routing infeas.)	11.49%	60.01%	38/66
4	+ B3 (cycle covers)	11.52%	60.00%	38/66
5	+ B4 (path ineq.)	11.46%	59.58%	38/66

Across the 66-instance suite, adding cut families produces a small, monotone improvement, concentrated almost entirely in the first two cuts. The B2’s routing infeasibility cuts constrain the y_i variables directly, which the LP cannot soften through fractional relaxation, giving the largest single contribution, whereas B3 and B4 target increasingly specific substructures (short cycles, particular paths) that mostly overlap with ground B1–B2 already cover. We do not read B3–B4’s flat contribution here as evidence they are ineffective in general (section 4.9’s derivations remain valid regardless), only that on this suite their marginal value is already captured by the earlier-added families.

8.6 Warm-Start Ablation

A natural question is whether the SA warm start actually matters to the B&C, or merely supplies a first feasible solution the solver could have found on its own. To answer this we run the full-cut-family configuration (config 5 of table 7) on all 66 synthetic benchmark instances twice, once with

the SA incumbent passed to the B&C as usual and once with the B&C given no incumbent at all (starting from `best_pts = 0`), all other settings remain fixed. We call it a *zero-incumbent failure* when the B&C reaches the time limit without ever having found an integer-feasible solution.

Table 8: Warm-start ablation on the full benchmark suite (66 instances, all cut families, 900 s time limit, $\lambda = 4.1 \times 10^{-5}$, $\rho = 0$).

	Proven optimal	Zero-incumbent failures
Warm start off	37 / 66	27 / 66
Warm start on	38 / 66	0 / 66

Without a warm start, the B&C fails to find any feasible integer solution on 27 of 66 instances (41%) within the time limit. On the instances it does resolve it is competitive (37 vs. 38 proven optimal), and its own search occasionally improves on what SA alone would have found. This is consistent with the standard role of a strong incumbent in branch-and-cut for NP-hard combinatorial problems. section 9 discusses the cases where it does improve on SA directly. Instance size is the main factor in zero-incumbent failures, every instance with $n \geq 50$ has a zero-incumbent failure, while no $n = 20$ instance has, and the $n = 40$ instances split roughly evenly.

This is the expected behaviour of a B&C with no primal heuristic of its own. Locating any integer-feasible route by branching alone is itself combinatorially hard and scales poorly with n , so small instances are stumbled onto quickly while larger ones essentially never are within the time limit.

8.7 Comparison Against a Classical OP Baseline

To compare against a known baseline, we strip our synthetic benchmark back to the classical, undirected, fatigue-free OP of Fischetti et al. [1998] and apply the branch-and-cut he originally developed for it, replacing each arc’s two directional costs with their average $c_{ij} = \frac{1}{2}(C_{ij} + C_{ji})$ and dropping the fatigue term entirely. On this comparison we are analyzing two things, what asymmetry and fatigue cost in achievable score, and what a much smaller cut arsenal costs in the classical solver’s own ability to prove or even find good solutions on larger instances.

On the 21 of 66 instances where both solvers independently certify global optimality the classical and flow scores have similar results. Defining the per-instance ratio as the classical solver’s proven-optimal points divided by the flow formulation’s, this ratio averages 0.971 (median 1.000) across the 21 instances. One outlier falls well below this: `bench_038` ($n = 40$, asymmetry $\alpha = 0.5$, the highest level swept), where the classical solver’s proven-optimal score is only 55% of the flow formulation’s (ratio 0.554). This means that symmetrising cost via $\frac{1}{2}(C_{ij} + C_{ji})$ inflates the cheaper of two directional costs on any arc where asymmetry is large, and a route that specifically exploits the cheap direction can become effectively unaffordable once that arc is priced at the average instead.

Across the full 66-instance suite (not restricted to the jointly-proven subset), the picture looks different, as the classical solver proves only 28/66 instances optimal (vs. 38/66 for flow) and its mean score ratio drops to 0.733 (median 0.649). This is mainly caused because the classical solver’s much smaller cut arsenal fails to close the gap on larger instances ($n \geq 75$).

9 Discussion

SA and B&C effectiveness. On 6 of the 14 real-terrain instance the B&C the proven-optimal incumbent is strictly better than SA’s own solution. On the remaining proven instances the B&C confirms SA’s solution without improving it, in line with the literature on metaheuristics for the OP [Vansteenwegen et al., 2011]. However, section 8.6 shows why the exact solver’s own search

still cannot be dropped in favour of SA alone: without a warm start, the same B&C fails to find any feasible solution at all on 27 of the 66 synthetic benchmark instances within the time limit, despite proving 37 of them optimal on the ones it does resolve. The B&C’s role is therefore closer to the standard one in branch-and-cut for hard combinatorial problems, where it needs a strong incumbent to prune aggressively, and its own contribution is a mix of occasional improvement and, a certified gap (or a proof of optimality).

Cost asymmetry. The 36.6% asymmetry on La Muela (vs. 17.5% on Torremocha) demonstrates that symmetric OP formulations would be inadequate for mountainous terrain. The Minetti model captures this directionality naturally through the metabolic cost polynomial.

Fatigue linearisation. The comparison between the MTZ and flow formulations confirms that MTZ’s big- M state-propagation constraint (25) is the main source of LP relaxation looseness. On the benchmark suite, flow’s relaxation is tighter than MTZ’s at every asymmetry level tested, most sharply at $\alpha = 0.25$ (3.7% vs. 33.4%; table 5). This mirrors the dominance of flow-conservation formulations over the unlifted, big- M form of MTZ-style sequencing constraints for subtour elimination in the ATSP [Öncan et al., 2009]. On real-terrain instances the effect is smaller and instance-dependent: flow’s gap is at or below MTZ’s on every unproven La Muela instance (table 4), but the two formulations are close on Torremocha, where nearly every instance is proven optimal by both regardless of relaxation tightness.

Computational complexity and runtime. The overall framework has three computational bottlenecks, each with different scaling behaviour.

Cost matrix construction. For n control points on a grid of $H \times W$ cells downsampled by factor s , the anisotropic Dijkstra runs n times on a grid of $(H/s)(W/s)$ cells. Each Dijkstra has complexity $O(\frac{HW}{s^2} \log \frac{HW}{s^2})$, giving a total cost matrix construction time of $O(n \frac{HW}{s^2} \log \frac{HW}{s^2})$. In practice, with Numba JIT compilation and $s = 8$, the cost matrix for 41 nodes on a 1056×1463 grid computes in under 4 seconds.

Simulated Annealing. Each SA iteration evaluates a neighbourhood move in $O(k)$ time, where k is the current route length. With N_{iter} iterations and N_{restart} restarts, the total SA time is $O(N_{\text{restart}} \cdot N_{\text{iter}} \cdot k)$. Since $k \leq n$ and both N_{iter} and N_{restart} are bounded by constants dependent on n , the SA runs in $O(n^2)$ per restart in the worst case. Empirically, SA completes in under 1 second for all tested instances ($n \leq 41$).

Branch-and-Cut. The B&C has worst-case exponential complexity in n , as the number of branching nodes can grow as $O(2^n)$. The LP relaxation at each node has $O(n^2)$ variables and $O(n^2)$ constraints (before cuts), solvable in polynomial time by the simplex method. However, the number of subtour elimination cuts can be exponential in $|V|$. In our experiments, the B&C runtime varies dramatically with instance structure: instances with fewer nodes or simpler topology (e.g., Clustered with 24 nodes: 14.1s) are solved orders of magnitude faster than dense instances (e.g., Standard with 31 nodes: 900s, hitting the time limit). This exponential sensitivity to instance structure is characteristic of branch-and-cut methods for routing problems and motivates the SA warm start, which provides a strong incumbent that enables aggressive pruning early in the search.

Cost model robustness. The IQR-based ACR-HCR fusion (section 3) is not guaranteed to be well-behaved: on Torremocha, two IOF terrain categories jointly cover the majority of the walkable area, concentrating the ACR cost distribution enough that its interquartile range is exactly zero within that area, forcing the scaling constant $c = 0$ (eq. (7))—HCR contributes nothing to the combined cost there, despite genuine morphological variation (slope, curvature) being present in the terrain. On La Muela, whose relief is more varied, the same procedure yields $c = 1.09$, a substantial HCR contribution. This degenerate case was not anticipated in the

original design and was only found by inspecting the fused cost distribution directly. The method should be treated as adapting to terrain characteristics only when the ACR distribution is not itself near-degenerate, not unconditionally as originally claimed. A fallback (e.g. a fixed default c when ACR’s IQR falls below a threshold, or a rank-based rather than interquartile-range-based scaling statistic) is left for future work.

Limitations. The framework has not been validated against GPS tracks from real orienteers. The fatigue model’s downhill-recovery term ρ (eq. (16)) has no literature-derived point estimate; the one study we found that measures how fatigue changes downhill-specific running economy [Vernillo et al., 2015] reports increased, not decreased, downhill cost after a mountain ultramarathon, which is why we fix $\rho = 0$ rather than assume partial recovery, but this remains a single, indirect data point, not a validated parameter. The empirical evaluation covers only two real terrains, differing substantially in relief and asymmetry (17.5% vs. 36.6%); this is insufficient to establish generality across terrain types, and additional terrain diversity is needed before the framework’s performance claims can be considered broadly representative. The B&C does not scale well beyond ~ 50 nodes due to the exponential growth of the branching tree; for larger instances, the SA solution should be treated as a high-quality heuristic without an optimality guarantee. We compare against a classical (undirected, symmetric, fatigue-free) OP baseline (section 8) to isolate what asymmetry and fatigue cost in solve difficulty, but not against modern large-scale exact OP solvers such as Kobeaga et al. [2021], which solve symmetric instances an order of magnitude larger than those tested here; whether their component techniques (shrinking, blossom separation, variable pricing) transfer to the asymmetric, fatigue-state-dependent setting is an open question. The Dijkstra downsampling ($s = 8$) introduces path approximation error that has not been formally quantified, though empirical comparison with $s = 4$ suggests the effect on the cost matrix is small ($< 2\%$ median deviation); this is an approximation of the terrain itself, so the solver’s optimality guarantee applies to the downsampled cost matrix, not to the original terrain exactly.

10 Conclusion

We presented an integrated framework for optimal route planning in orienteering, combining GIS-based cost surface generation with exact combinatorial optimisation. The key methodological contributions are: (i) a GIS pipeline that transforms orienteering map data and terrain models into a fully asymmetric, fatigue-aware cost matrix, naturally giving rise to the Asymmetric Orienteering Problem with Fatigue (AOPF). (ii) IQR-based raster fusion adapting to terrain characteristics. (iii) Two Branch-and-Cut formulations for the AOPF. An MTZ-based model with McCormick linearisation and a flow-based model with a natively linear fatigue budget. (iv) Valid inequalities tailored to the AOPF: tightened flow-arc coupling, fatigue-aware lifted covers, and routing infeasibility cuts, complemented by directed adaptations of cycle cover and path inequalities. (v) S parameterised benchmark suite for the asymmetric OP with fatigue. And (vi) Computational evidence that Simulated Annealing finds optimal solutions on real-terrain instances, with an ablation study quantifying the contribution of each cut family.

Future work includes validation against real GPS data, extension to multi-day events, and integration of weather and time-of-day effects on terrain traversability.

Reproducibility. The source code, benchmark instances (14 JSON files with asymmetric cost matrices and proven optimal solutions), and supplementary data are publicly available at <https://github.com/PabloAlmendralej/Asymmetric-Orienteering-Problem/tree/main>. To the best of our knowledge, no benchmark instances exist for the asymmetric OP with terrain-based costs and fatigue; we release ours to facilitate future comparison.

References

- T. Tsiligirides. Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, 35(9):797–809, 1984.
- A. E. Minetti, C. Moia, G. S. Roi, D. Susta, and G. Ferretti. Energy cost of walking and running at extreme uphill and downhill slopes. *Journal of Applied Physiology*, 93(3):1039–1046, 2002.
- G. Albert and Z. Sárközy. Route planning on orienteering maps with least-cost path analysis. *Proceedings of the ICA*, 4:4, 2021. doi:10.5194/ica-proc-4-4-2021.
- S. J. Riley, S. D. DeGloria, and R. Elliot. A terrain ruggedness index that quantifies topographic heterogeneity. *Intermountain Journal of Sciences*, 5(1–4):23–27, 1999.
- L. W. Zevenbergen and C. R. Thorne. Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms*, 12(1):47–56, 1987.
- Q. Huangfu and J. A. J. Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018.
- P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011.
- M. Fischetti, J. J. Salazar González, and P. Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998.
- A. Gunawan, H. C. Lau, and P. Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.
- S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
- P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research*, 36(12):3281–3290, 2009.
- W. Tobler. Three presentations on geographical analysis and modeling. Technical Report 93-1, National Center for Geographic Information and Analysis, 1993.
- B. Jaszczak and Ł. Płociniczak. Optimal strategy for trail running with nutrition and fatigue factors. *arXiv preprint arXiv:2401.02919*, 2024.
- D. Jaén-Carrillo, A. Margarit-Boscà, F. García-Pinillos, and M. Holler. Pacing strategy and resulting performance of elite trail runners: Insights from the 2023 World Mountain and Trail Running Championships. *International Journal of Sports Physiology and Performance*, 20(3):449, 2025.
- G. Vernillo, A. Savoldelli, A. Zignoli, S. Skafidas, A. Fornasiero, A. La Torre, L. Bortolan, B. Pellegrini, and F. Schena. Energy cost and kinematics of level, uphill and downhill running: fatigue-induced changes after a mountain ultramarathon. *Journal of Sports Sciences*, 33(19):1998–2005, 2015.
- M. Fischetti. Facets of the asymmetric traveling salesman polytope. *Mathematics of Operations Research*, 16(1):42–56, 1991.
- M. Fischetti and P. Toth. A polyhedral approach to the asymmetric traveling salesman problem. *Management Science*, 43(11):1520–1536, 1997.

- T. Bektaş and G. Laporte. The pollution-routing problem. *Transportation Research Part B: Methodological*, 45(8):1232–1250, 2011.
- C. Malandraki and M. S. Daskin. Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms. *Transportation Science*, 26(3):185–200, 1992.
- D. Feillet, P. Dejax, and M. Gendreau. Traveling salesman problems with profits. *Transportation Science*, 39(2):188–205, 2005.
- G. Kobeaga, M. Merino, and J. A. Lozano. A revisited branch-and-cut algorithm for large-scale orienteering problems. *European Journal of Operational Research*, 290(3):865–880, 2021.
- J. T. Linderoth and M. W. P. Savelsbergh. A computational study of search strategies for mixed integer programming. *INFORMS Journal on Computing*, 11(2):173–187, 1999.
- T. Öncan, İ. K. Altinel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36(3):637–654, 2009.