Balancing Earthwork: The GoSurvey Grading Optimizer and Why a Single Offset Solves It

Chet Jones
GoSurvey Project

November 2026

REQ-523 | REQ-524 | GitHub Issue #813 | Pull Request #1189

Abstract

A designer often has a finished grading group, a surface that describes proposed ground, and needs its elevation raised or lowered so the earthwork balances (zero net cut and fill) or hits a stated net volume. This note explains how the GoSurvey Grading Optimizer does that without moving any plan line. It proves that net volume is an exactly linear function of one shared vertical offset, with slope equal to the common plan area. That proof explains why a single offset is enough, why the solution is unique, and why the bracketing and bisection search converges. It works three examples by hand, checks the linear law with an independent numerical experiment, and lists the automated tests and the known limits.

1 What problem this solves

Cut and fill are the two quantities a grading plan is judged on. Cut is material that must be removed to reach the design surface; fill is material that must be brought in. A plan with more fill than cut needs imported material, and a plan with more cut than fill leaves spoil. Designers often want the *net* to be zero, or to equal a specified amount, without redrawing the footprint of the pad, road or berm.

Changing elevation is the one lever that keeps the footprint fixed. Raising every point of a design surface by the same amount moves the design up or down together, and the net moves with it. The question is whether one shared offset is enough to reach any target, and whether the search for that offset can be trusted. Those are the questions this note answers.

2 The model

Let $G(x, y)$ be the ground surface (the base) and $Z(x, y)$ be the design surface (the comparison), both triangulated surfaces. Their common plan area is A: the part of the plane where both are defined. Over A define the signed difference

$$d(x, y) \;=\; Z(x, y) - G(x, y). \tag{1}$$

Following GoSurvey's REQ-073 convention, which uses the cut and fill terms of Civil 3D (Autodesk, n.d.), the volumes are

$$\text{fill} \;=\; \int_A \max(d, 0)\, dA, \qquad \text{cut} \;=\; \int_A \max(-d, 0)\, dA, \qquad N \;=\; \text{fill} - \text{cut}, \tag{2}$$

where a positive N means net fill. The optimizer does not change G. It replaces Z with $Z + \Delta$, a uniform vertical shift, and looks for the value of Δ that makes N equal the requested target T.

3 The governing law: net volume is affine in the offset

Theorem 1 (Affine net-volume law)**.** *For fixed surfaces with common area $A > 0$, the net volume after shifting the design by Δ is*

$$N(\Delta) \;=\; N(0) + A\,\Delta. \tag{3}$$

Proof. For every real number u, $\max(u, 0) - \max(-u, 0) = u$. Apply this with $u = d + \Delta$ at each point of A:

$$\begin{aligned} N(\Delta) &= \int_A \max(d + \Delta, 0)\, dA - \int_A \max(-(d + \Delta), 0)\, dA \\ &= \int_A \big(\max(d + \Delta, 0) - \max(-(d + \Delta), 0)\big)\, dA \\ &= \int_A (d + \Delta)\, dA \\ &= \int_A d\, dA + \Delta\, A \;=\; N(0) + A\,\Delta. \qquad \blacksquare \end{aligned}$$

□

The proof uses nothing about the shape of the surfaces. Terrain that is steep or flat, convex or concave, gives the same law. The only property that matters is that a uniform shift adds the same amount Δ to every point of the common area, so its total is ΔA.

Consequences.

(i) **A unique offset exists.** For a target T the equation $N(\Delta) = T$ has exactly one solution,

$$\Delta^* \;=\; \frac{T - N(0)}{A}, \tag{4}$$

provided $A > 0$. This is the closed-form answer.

(ii) **The solution is monotone.** Because $A > 0$, N strictly increases with Δ. Raising the design never reduces fill, and lowering it never reduces cut.

(iii) **One variable is enough.** The target is a single number, and the affine law shows a single offset reaches any value of N. Other variables (moving individual members, changing slopes) are not needed for the net objective and would make the answer non-unique.

4 How GoSurvey computes it

Sampling the volumes

Earthwork volumes are routinely estimated by sampling a footprint on a grid or by cross-sections (Schofield, 2001), and a triangulated surface is treated as linear over each triangle (de Berg et al., 2008). GoSurvey does not integrate d exactly. It samples the common footprint on a regular grid with a fixed budget of about 250,000 cells (the constant `kTargetVolumeSamples` in `src/util/surfacevolume.cpp`). The cell size is therefore $\sqrt{A/250000}$, independent of how many triangles either surface has, which keeps a recompute's cost bounded. Within each cell the two triangulated surfaces are compared at the cell corners. Where the difference keeps one sign over a cell, its contribution is the mean difference times the cell area. Where it changes sign, the cell is split along the lines where $d = 0$ and each piece is treated the same way. The per-cell areas add up to the common area A_s of the sampled model, and the net is the sum of those contributions.

Each piece has a single sign, so its contribution is its area times the mean of d over the piece. A shift changes every corner's d by Δ, so each piece's mean changes by Δ and its contribution by Δ times its area. Summing over pieces gives

$$N_s(\Delta) \;=\; N_s(0) + A_s\,\Delta, \tag{5}$$

which is Equation 3 for the sampled model, with A_s the sampled common area. The grid is laid over the bounding-box intersection, so its layout does not depend on Δ and every candidate uses the same cells.

Solving for the target

The solver, `OptimizeGradingElevationOffset`, works in four steps:

1. Evaluate $N(0)$. If it already equals T to within 0.27 ft^3 (0.01 yd^3), return $\Delta = 0$.
2. Bracket the answer. Start at ± 1 ft and double the step until the sign of $N(\Delta) - T$ changes. Stop with a message if no finite bracket is found within 64 doublings.
3. Bisect the bracket. Each step halves the interval and keeps the half that still brackets T. Stop when $|N(\Delta) - T| \le 0.27$ ft^3 or after 64 steps.
4. Report the offset, the cut, the fill, the net, the error and the step count. Nothing is written to the drawing until the user presses Apply.

Bisection is the standard method for a continuous function with a known sign change (Burden and Faires, 2015). Its error after n halvings is at most $(b-a)/2^n$. The bracket from Step 2 supplies (a, b) with a sign change, and monotonicity (consequence (ii)) guarantees that the bisected half always keeps the sign change, so Step 3 converges to the unique solution.

Why bisection, when a closed form exists. Equation 5 gives the answer in one step, $\Delta^* = (T - N_s(0))/A_s$, and the solver can compute A_s from its own sampling. The search is kept because it does not depend on the affine identity holding exactly at the sampled level. It uses only evaluations of the same function that Apply will commit, and the reported result is always an evaluated point. The cost is more evaluations than the closed form needs (Section 6 shows 20 for a typical case). A secant step built from the two evaluations already made would reach the same answer in one further evaluation. That is a possible speed-up, not a change in the result, and it is not implemented.

5 Worked examples by hand

All three use A and $N(0)$ computed by hand, then apply Equation 4. Each is also a test case in the suite.

Case	Setup	$N(0)$ (ft^3)	Δ^* (ft)
A. Balance a flat design 2 ft low	ground 100 ft, design 98 ft, $A = 100$ ft^2	-200	$+2.00$
B. Target $+200$ ft^3 fill	same surfaces, $T = +200$	-200	$+4.00$
C. Sloped ground, balance	ground $100 + 0.1x$, design 98, square $A = 100$ ft^2 over $0 \le x, y \le 10$	-250	$+2.50$

Case A. The design sits 2 ft below the ground over 100 ft^2, so $N(0) = \int (98-100)\, dA = -2 \times 100 = -200$ ft^3, a cut of 200 ft^3. Equation 4 with $T = 0$ gives $\Delta^* = 200/100 = +2$ ft. Raising the design to 100 ft makes $d = 0$ everywhere, so $N = 0$.

Case B. The same surfaces with a target of $+200$ ft^3 net fill give $\Delta^* = (200-(-200))/100 = +4$ ft. The design rises to 102 ft, so $d = +2$ ft over 100 ft^2, which is 200 ft^3 of fill.

Case C. Now the ground slopes. Over the square $0 \le x, y \le 10$ ft, $\int_A x\, dA = 10 \cdot \frac{10^2}{2} = 500$ ft^3/ft, so $N(0) = \int (98-(100+0.1x))\, dA = -200-0.1 \cdot 500 = -250$ ft^3. The offset is $\Delta^* = 250/100 = +2.5$ ft. The slope of the ground changes $N(0)$ but not the form of the answer: the offset adds the same Δ everywhere, so only the starting net enters Equation 4.

6 Numerical check of the affine law

The algebra above was checked with an independent numerical experiment, not by reading the C++ alone. A short script (`verify_linearity.py`, kept with the build artifacts) reproduces

the cell-centre form of the rule on two synthetic surfaces with a common footprint of 809,600 ft^2, sampled on 400×400 cells of 5.06 ft^2 each. It is a re-implementation of the sampling idea, not the production code, so it tests the mathematics rather than the C++ implementation. The C++ itself is covered by the test suite in Section 9.

Quantity	Result
Cells × area per cell	$160{,}000 \times 5.06$ ft^2
Measured slope $\partial N/\partial\Delta$	809,600.000 ft^2
Common area A	809,600.000 ft^2
Largest error of $N(\Delta)$ against $N(0) + A\Delta$ over 6 offsets	3.0×10^{-8} ft^3
Closed-form offset for $T = 3500$ ft^3	0.972117653 ft
Bisection offset (tolerance 0.27 ft^3), 20 steps	0.972117424 ft
Difference between the two offsets	2.3×10^{-7} ft

The slope equals the area to every digit printed, and the residual is at the level of floating-point round-off. Bisection agrees with the closed form to 2×10^{-7} ft, far inside its own tolerance on N (0.27 ft^3 on a volume of about 3500 ft^3). Twenty steps is the cost of the search for this case.

7 What changes for the user

The designer chooses Balance cut and fill or a target net in yd^3 (1 yd^3 = 27 ft^3), sees the offset and the resulting cut, fill, net and iteration count, and watches each candidate appear in the viewport. The footprint, member relationships and source gradings do not change. Apply commits the exact previewed offset to the group's baked surface and its member strips as one undoable step, and it refuses if the drawing changed after the preview. The optimized group remains an ordinary surface, so it still participates in volumes, labels and profiles.

Two consequences follow from the design. Because the offset is uniform, every member of a group moves together, so relative elevations between members are preserved. Because the common area sets the slope, a group whose footprint does not overlap the ground has $A = 0$ and no solution; the solver refuses that case with a message rather than dividing by zero.

8 Limits

- **The answer is for the sampled model.** The affine law is exact for the sampled volumes. The sampled volume differs from the true integral of a triangulated surface by the sampling error of the grid, which shrinks as the cell count grows. The fixed budget keeps that cost bounded.
- **One shared offset only.** Independent member elevations, slopes, locked points and footprint changes are out of scope (REQ-523, item 8). A different objective would need a second rule to choose among the many solutions, and none has been specified.

- **A different objective from Civil 3D.** Civil 3D's Grading Optimization weighs several objectives, and its documentation describes a *Balance cut and fill* objective without giving a formula (Autodesk, n.d.). Users have reported overshoot against that balance (Autodesk Community, n.d.). GoSurvey optimizes one net target with one variable. Within the sampled model it reaches the target to the stated tolerance, and it does not weigh any other objective.
- **Common footprint only.** Volumes count only the area where both surfaces are defined. Area outside the overlap is ignored, as REQ-073 states.
- **Timing is not claimed.** Each candidate re-samples the common footprint through a spatial index, so its cost is set by the fixed sample budget and the index, not directly by the triangle count. Timing on a very large surface depends on the machine and has not been measured for this note.

9 Automated test summary

File	Test case	What it proves
`SurfaceVolumeTests.cpp`	Grading optimization finds a common elevation offset for both net-volume targets	Equation 4 for zero balance and for a non-zero target
`SurfaceVolumeTests.cpp`	Grading optimization emits every valid evaluation in solver order	the search order is deterministic and complete
`SurfaceVolumeTests.cpp`	An already balanced grading emits its zero-offset evaluation	no offset is applied when $N(0)$ is already on target
`SurfaceVolumeTests.cpp`	Grading optimization rejects no-overlap, malformed, and non-finite targets	$A = 0$ and bad inputs are refused with no change
`SurfaceVolumeTests.cpp`	Grading optimization reports finite unbracketable targets and cancellation	the bracketing cap and the cancel path both terminate
`SurfaceVolumeTests.cpp`	Grading optimization brackets negative targets and above-ground designs	both sides of the sign change are reached
`DrapeCommandTests.cpp`	Volume reports persist per drawing but are excluded from templates	the reported net survives save and reopen
`QtEditorTests.cpp`	gradingOptimizationRibbonPreviewsApplies UI path commits	the UI path commits ElevationOffset exactly one undoable step
`QtEditorTests.cpp`	gradingOptimizationLeavesGroundStaleWhenHUsesSurfaces	the UI test surfaces are not marked current incorrectly

The Release build runs all 194 CTest cases, including the six Qt editor shards, and all pass. The source is PR #1189, branch `feat/issue-813-1`, against `beta`.

References

Autodesk (n.d.). *Civil 3D Grading Optimization Tutorials: Setting Objectives and Running Optimization*. Autodesk, Inc. Accessed 10 November 2026. Describes the weighted *Bal-*

ance cut and fill objective; gives no formula. https://help.autodesk.com/cloudhelp/ENU/C3DGO-Tutorials/files/tutorials/set_obj_and_run_opt.html

Autodesk Community (n.d.). *Grading Optimizer overshoots cut and fill balance*. Civil 3D forum. https://forums.autodesk.com/t5/civil-3d-forum/grading-optimizer-overshoots-cut-and-fill-balance/td-p/10258202

Burden, R. L., and Faires, J. D. (2015). *Numerical Analysis*, 10th ed. Boston: Brooks/Cole, Cengage Learning. Section 2.1, the bisection method.

de Berg, M., Cheong, O., van Kreveld, M., and Overmars, M. (2008). *Computational Geometry: Algorithms and Applications*, 3rd ed. Berlin: Springer. Chapter 9, Delaunay triangulations and height interpolation.

Schofield, W. (2001). *Engineering Surveying*, 5th ed. Oxford: Butterworth-Heinemann. Chapter 9, Earthworks (areas, cross-sections, volumes).

GoSurvey Project (2026). *GitHub Issue #813: Grading Optimization: iterative cut/fill balancing solver*. https://github.com/chetjones003/GoSurvey/issues/813