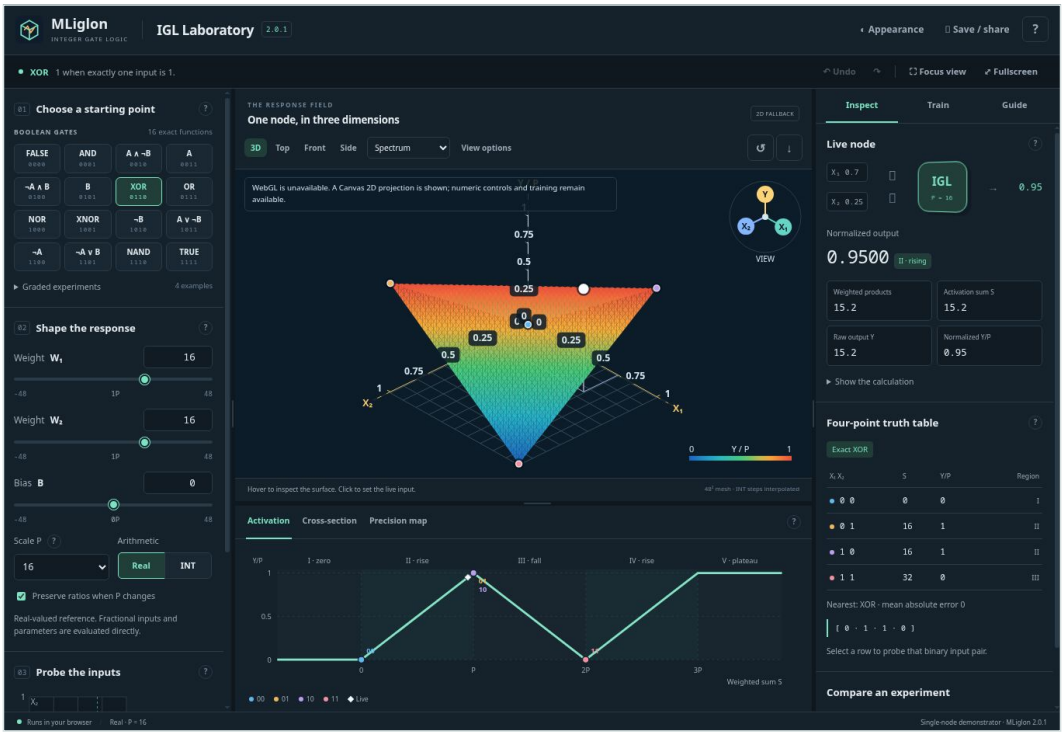
IGL Laboratory

Technical User Guide

Version 2.0.1 | 11 November 2026



A complete illustrated reference to the one-node laboratory: controls, equations, Boolean presets, 3D interpretation, precision, demonstration training, reproducible exports and terminology.

[Open the IGL Laboratory](https://mliglon.com/igl-lab/)

https://mliglon.com/igl-lab/

Contents

Select a chapter or section to jump to it. Page numbers include the cover and contents pages.

1. Start here . . . 4

1.1 Purpose, scope and reading paths . . . 4

1.2 Workspace map . . . 5

1.3 A first session in five minutes . . . 6

1.4 Starting state and application controls . . . 6

1.5 Starting points, parameters and live inputs . . . 7

1.6 Using help while you work . . . 8

2. Node mathematics and worked outputs . . . 10

2.1 The node and its units . . . 10

2.2 The five activation regions and their exact boundaries . . . 10

2.3 Real arithmetic and Fixed INT arithmetic . . . 11

2.4 P, quantized inputs and what the application stores . . . 12

2.5 Worked example 1 - exact XOR corners, graded interior . . . 13

2.6 Worked example 2 - Asymmetric at P = 4 and 1024 . . . 14

2.7 Worked example 3 - a negative half-way sum . . . 16

3. Presets and Boolean interpretation . . . 17

3.1 Boolean truth vectors, nearest gates and loss measures . . . 17

3.2 All 16 Boolean presets and four graded experiments . . . 17

4. Surfaces, probes and diagnostics . . . 19

4.1 What the surface and reference mesh represent . . . 19

4.2 Scene controls and display options . . . 20

4.3 Mouse, touch and keyboard reference . . . 22

4.4 Analysis, inspection, probes and comparisons . . . 23

4.5 Reading the precision map without sampling traps . . . 24

5. Single-node training . . . 27

5.1 Single-node training: what is optimized . . . 27

5.2 Limits and controls . . . 28

5.3 Local search, Step and ×10 . . . 29

5.4 Multiscale search and deterministic restarts . . . 29
5.5 Execution, cancellation and progress . . . 29
5.6 Training controls and built-in walkthroughs . . . 32

6. Appearance, layout and history . . . 33

6.1 Appearance: all themes, fonts and sizes . . . 33
6.2 Workspace layouts and responsive dividers . . . 35
6.3 Exactly what Undo and Redo restore . . . 36

7. Saving, sharing and reproducibility . . . 37

7.1 Save, share and export reference . . . 37
7.2 Experiment JSON fields and compatibility . . . 39
7.3 A practical reproducibility record . . . 40

8. Three reproducible walkthroughs . . . 41

8.1 Walkthrough 1 - explain XOR with one node . . . 41
8.2 Walkthrough 2 - compare P4 and P1024 . . . 41
8.3 Walkthrough 3 - local plateau versus multiscale search . . . 42

9. Troubleshooting and operational checks . . . 43

9.1 Practical troubleshooting . . . 43
9.2 Quick numerical and interface checks . . . 44
9.3 Launching and linking a hosted installation . . . 44

10. Glossary and reference . . . 45

10.1 Technical glossary . . . 45
10.2 Acronym and notation legend . . . 48
10.3 Implementation and browser references . . . 49

CHAPTER 01

Start here

1.1 Purpose, scope and reading paths

IGL Laboratory 2.0.1 is an interactive technical demonstrator of one two-input Integer Gate Logic node. It connects adjustable parameters to an arithmetic trace, a five-region activation, a four-row truth table, a response surface and a bounded candidate search. This manual is intended for computer scientists, engineers, instructors and other readers comfortable with algebra and Boolean logic.

The program runs in its own browser tab and uses the browser viewport directly. A WordPress page can link to it, but no WordPress plugin is required. Open https://mliglon.com/igl-lab/ to use the deployed application. No build tool, Python environment, account or browser extension is needed to operate the installed laboratory.

> **One node, four training examples**
> The training demonstration adjusts W1, W2 and B at fixed P and arithmetic mode. Its data are the four Boolean input pairs. It does not train a multilayer network, import a dataset, demonstrate a CNN, measure device-level SIMD speed, or establish test-set generalization. A future small-network trainer would be a separate application.

Reading goal	Recommended route
Start a demonstration	Read this chapter, then follow Walkthrough 1 in Chapter 8.
Reproduce a number	Read Chapter 2, especially Real versus INT and the worked examples.
Interpret a surface or precision map	Read Chapter 4. Distinguish exact evaluation from finite visual sampling.
Understand the optimizer	Read Chapter 5 and Walkthrough 3.
Save a reproducible experiment	Read Chapter 7, including what each export leaves out.
Look up a term or control	Use the clickable contents, PDF bookmarks and the glossary in Chapter 10.

This manual describes release 2.0.1. Program names and numerical behavior were checked against the released HTML, JavaScript and stylesheet, and against the live application. The source index in Chapter 10 identifies the implementation files and browser references used.

1.2 Workspace map

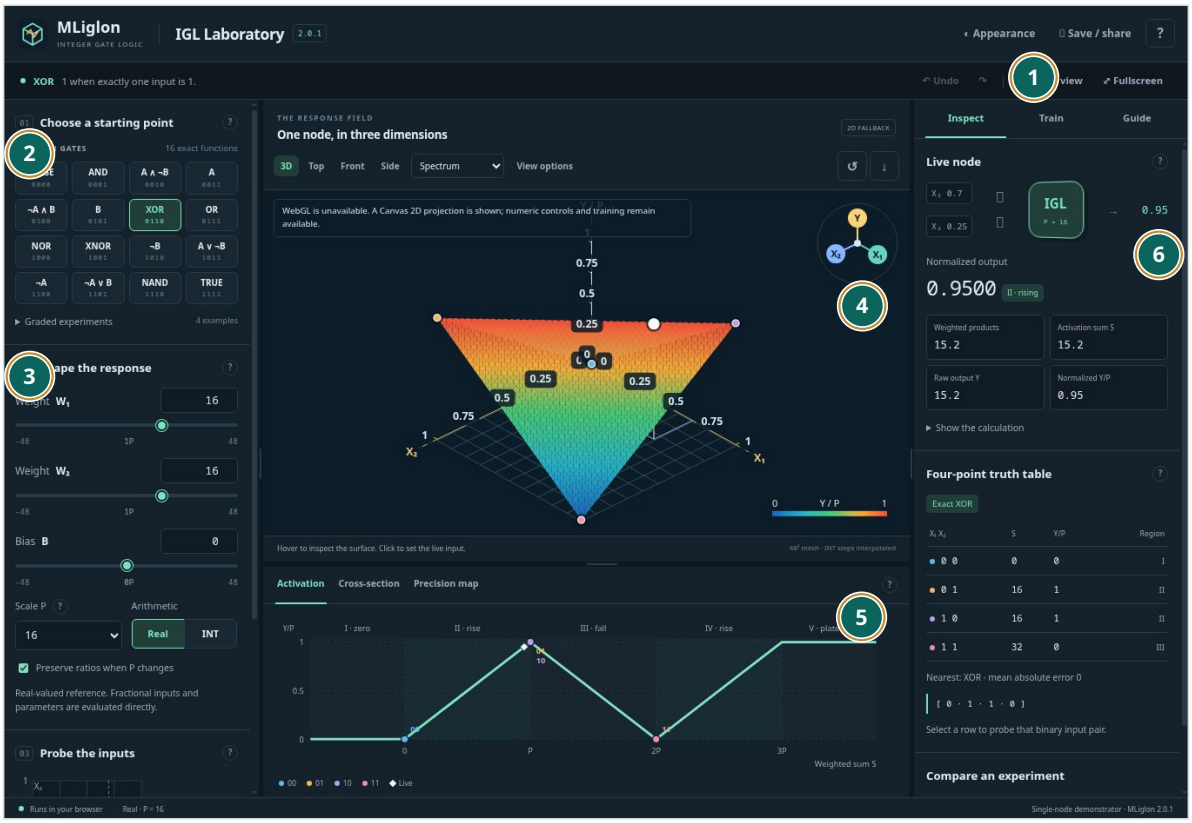


Figure 1. Default workspace: XOR, P = 16, Real, inputs 0.7 and 0.25. The numbered callouts identify the regions described below.

Callout	Region	What to watch
1	Application toolbar	Appearance, Save / share, guide, Undo / Redo, Focus view and Fullscreen.
2	Starting points	Sixteen Boolean presets and four graded experiments. A preset changes the model, not the training target.
3	Node and input controls	Weights, bias, P and arithmetic; scroll this panel to reach the live input pad and exact input fields.
4	Response field	Height and color encode normalized output across the input square. Camera controls change the view, not the calculation.
5	Analysis tabs	Activation, Cross-section and Precision map give different explanations of the same current node.
6	Inspect / Train / Guide	Read numerical results, optimize the node, or open built-in walkthroughs. Panels scroll independently on a desktop.

Screenshot convention
Screenshots in this guide were captured from the live 2.0.1 interface. Graphite is used for the overview; Technical Paper is used for readable control close-ups. The capture browser uses Canvas 2D fallback, as its badge states. Native GPU WebGL and PNG downloading were separately confirmed working by the user. Computed mathematical illustrations are labeled and are distinct from screenshots. Individual close-ups may show different documented experiment settings.

1.3 A first session in five minutes

1. **Open a clean starting point**
   Select Real, choose P = 16, and then choose XOR. Loading the preset last ensures W1 = W2 = 16 and B = 0 regardless of an earlier Preserve ratios setting. Enable Preserve ratios when P changes for the remaining steps. If you entered from an existing experiment, select the XOR training target separately when you want its MSE.

2. **Check the four corners**
   Inspect should report Exact XOR and the vector [0, 1, 1, 0] in 00, 01, 10, 11 order. Click a truth-table row to move the live input to that corner.

3. **Move into the interior**
   Enter x1 = 0.7 and x2 = 0.25 in Real mode. The weighted sum and raw output are 15.2; the normalized output is 0.95. Expand Show the calculation to see every step.

4. **Explore one display at a time**
   Try Activation, then Cross-section. Drag the scene to orbit; use the camera presets for standard views. Camera motion does not change the truth table.

5. **Introduce integer rounding**
   Choose INT and P = 4, then inspect the effective input coordinates. Read the precision-map values, not just its colors. Chapter 2 explains why changing P can preserve coefficients while changing quantization.

6. **Keep a portable record**
   Use Save / share and download Experiment JSON. A copied experiment link is useful for the numerical model; the JSON carries additional view, appearance and search settings.

Inputs and selectors take effect immediately. A numeric field commits when you finish the edit and leave it or otherwise trigger its change event. During asynchronous training, a focused numeric field is protected from replacement while you type; in 2.0.1 it refreshes to the current model after you leave it.

1.4 Starting state and application controls

A fresh page without an experiment fragment starts at XOR: $W_1 = 16$, $W_2 = 16$, B = 0, P = 16, Real arithmetic, $X_1 = 0.7$ and $X_2 = 0.25$. The training target is XOR; Preserve ratios when P changes is on. Inspect and Activation are selected. Saved appearance preferences are applied automatically; a saved model is restored only when requested.

Control	Action and availability
Appearance	Opens Appearance & layout; changes apply immediately.
Save / share	Opens Save, share & export.
? / Open laboratory guide	Selects Guide; on widths ≤1100 px, scrolls that panel into view.
Undo / ↷ Redo	Restore model snapshots; disabled when their history is empty. See the history scope below.
Focus view / Show panels	Hides the sidebars and analysis panel to enlarge the scene; toggles back without resetting the model.
Fullscreen / Exit fullscreen	Requests or exits browser fullscreen. Permission and browser support determine availability.

Control	Action and availability
Skip to laboratory	Keyboard-visible link moves to the workspace.
MLiglon brand; About IGL; Visit MLiglon	Open the MLiglon website in another browser context. Detailed user guide opens the packaged guide.
Contextual ? buttons	Explain presets, parameters, scale, inputs, trace, truth table, training and analysis. Hover, focus or click to show; Escape, scrolling or clicking elsewhere dismisses.
Dialog × buttons	Close Appearance, View options, Save / share or Copy manually. Native Escape and clicking outside the dialog also close it.

Implementation reference: HTML and controller [R2, R5].

1.5 Starting points, parameters and live inputs

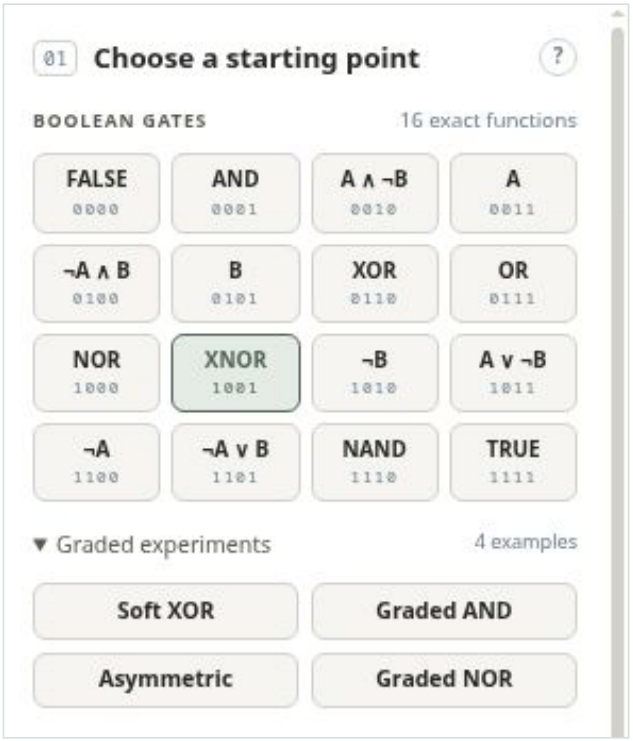


Figure 2. Starting points: all 16 Boolean functions and the four expanded graded presets.

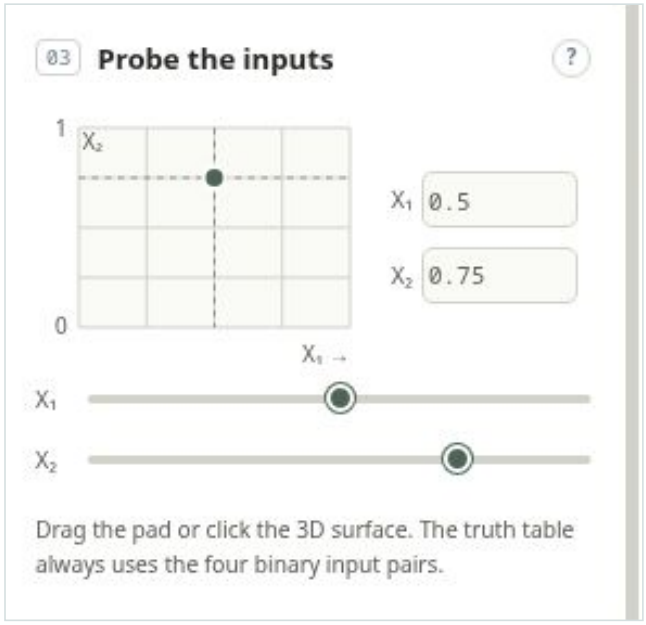


Figure 3. Live input pad and exact fields in the P4 integer example. Requested values have snapped to 0.5 and 0.75.

The Boolean buttons are FALSE, AND, A ∧ ¬B, A, ¬A ∧ B, B, XOR, OR, NOR, XNOR, ¬B, A ∨ ¬B, ¬A, ¬A ∨ B, NAND and TRUE. Their four-bit labels use input order 00, 01, 10, 11. Graded experiments expands Soft XOR, Graded AND, Asymmetric and Graded NOR. A preset changes the parameters, not the selected training target. In INT mode, a fractional graded preset can promote P to 16 to represent its parameters exactly.

Control	Default / range	Effect
Weight W_1; Weight W_2; Bias B	16; 16; 0. Each -3P…+3P.	Slider step: P/64 in Real, 1 in INT. Number fields accept finer Real values; INT rounds. Parameter edits stop training and clear its displayed history.
Scale P	16; 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024	Changes activation and integer scales; recomputes limits and input quantization.

Control	Default / range	Effect
Real / INT	Real	INT rounds parameters and snaps inputs. Returning to Real does not recover discarded fractions.
Preserve ratios when P changes	On	Multiplies parameters by new P / old P. When off, values stay fixed unless clamped to the new limits.
X_1; X_2 sliders and number fields	0.7; 0.25. Both 0…1.	Real sliders step by 0.001; INT sliders and effective inputs step by 1/P. Number entry accepts a finite value in range.
Input pad	Live input point	Click or drag: X_1 increases rightward, X_2 upward. Changes the probe, not the four training examples.
Reset to XOR	Always available	Loads XOR at the current P and mode. It is not a full reset: target, appearance and live inputs remain, subject to quantization.

Number fields normally display up to eight decimal places; exported Real values retain their stored precision. Invalid or out-of-range entries restore the current valid model value. In 2.0.1, a number field kept focused during training refreshes after focus leaves, while an actively typed edit remains undisturbed.

Implementation reference: Preset and evaluation core [R1]; event handlers [R2].

1.6 Using help while you work

The question-mark buttons explain their neighboring controls without leaving the experiment. The main help button opens the Guide tab. Its lesson buttons apply useful example settings; treat them as actions that can change the current demonstration, not merely headings.

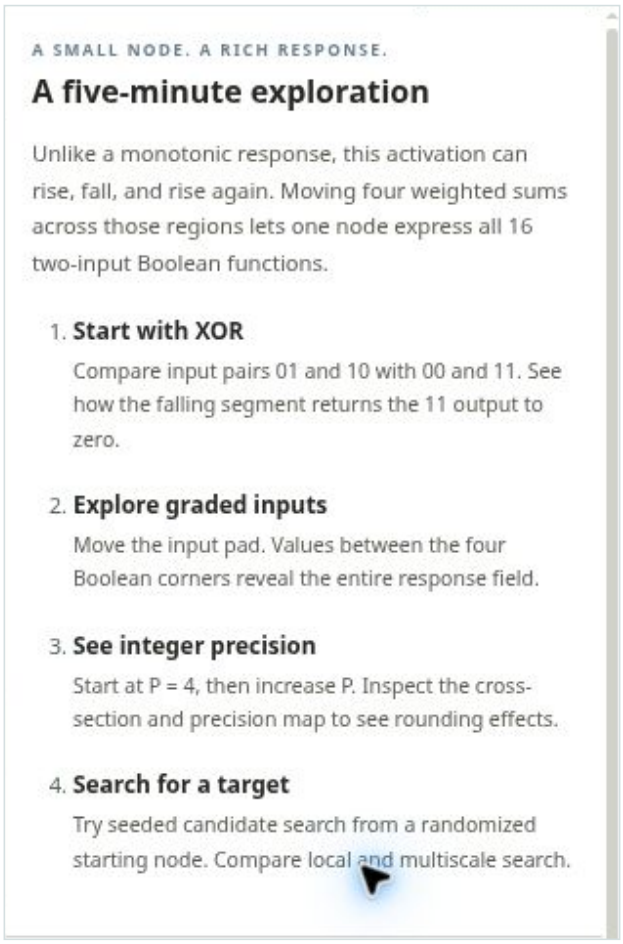


Figure 4. Four executable lesson starting points in the Guide tab. Its activation reference is available farther down the panel.

Use the mouse wheel over the panel you want to scroll. If a control appears missing, first check that panel's scroll position. Focus view hides the side panels deliberately; use Show panels to return. On narrow windows the workspace stacks and the document scrolls.

For a reproducible example, follow the explicit settings in Chapter 8. The guide lessons are convenient starting points, while the worked examples explain the exact numbers.

CHAPTER 02

Node mathematics and worked outputs

2.1 The node and its units

The laboratory evaluates one node with two inputs, two weights and one bias. Inputs x_1 and x_2 lie in the unit square. The node first forms an activation sum S, then applies the five-region function f_p. Keep the raw output Y separate from the normalized output q = Y/P: charts, Boolean comparisons and training losses use the normalized value. P sets the output scale and, in Fixed INT mode, the quantization grid.

Symbol	Meaning	Allowed or resulting range
x_1, x_2	Normalized inputs	0 through 1
W_1, W_2, B	Stored weights and bias	-3P through +3P each
P	Positive scale	One of 11 supported powers of two
S	Activation sum before f_p	May be negative or exceed 3P
Y	Raw activation output	0 through P
q = Y/P	Normalized output	0 through 1

The parameter bounds constrain each weight and the bias, not S itself. With valid inputs, S can reach -9P or +9P. The activation safely maps that wider interval back into 0…P. A larger S does not always mean a larger output: the function rises, falls and rises again. That nonmonotonic shape is what lets one such node reproduce XOR at the four binary input corners.

$$q = \frac{Y}{P} = \frac{f_P(S)}{P}$$

The displayed height and normalized numerical result are q. Raw Y changes units when P changes.

2.2 The five activation regions and their exact boundaries

The function is continuous at 0, P, 2P and 3P, although its slope changes there. Region labels use the exact inequalities below. At a boundary, adjacent formulas give the same value; the assigned label still matters when checking the trace or exported CSV.

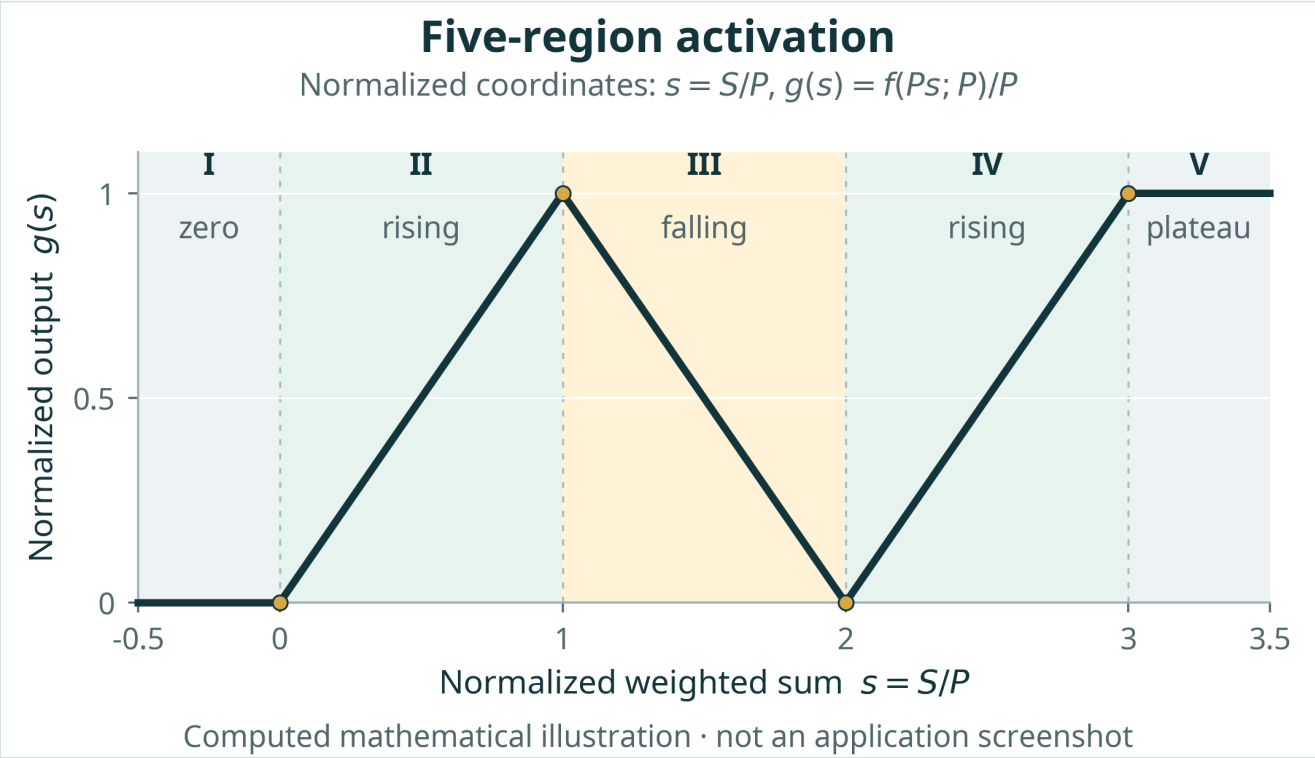


Figure 5. Computed activation diagram. With s = S/P, the normalized response is g(s) = fP(S)/P. Boundary values are continuous even though region labels and slopes change.

Region	Exact interval	Raw output Y	Slope with respect to S
I	S ≤ 0	0	0
II	0 < S ≤ P	S	+1
III	P < S ≤ 2P	2P - S	-1
IV	2P < S ≤ 3P	S - 2P	+1
V	S > 3P	P	0

Consequently S = 0 belongs to I and yields 0; S = P belongs to II and yields P; S = 2P belongs to III and yields 0; S = 3P belongs to IV and yields P. The plateau label V begins strictly above 3P. The activation chart plots this same rule, with the four binary sums and current live sum marked on the curve.

$$f_P(Pt) = P\,f_1(t)$$

The normalized activation shape is independent of P when expressed against t = S/P.

2.3 Real arithmetic and Fixed INT arithmetic

Real mode evaluates fractional inputs and parameters directly. “Real” is a numerical reference implemented with JavaScript Number values, so ordinary binary floating-point rounding still applies. It is not symbolic or arbitrary-precision arithmetic. The display may round a value to fewer decimal places than the underlying stored number.

$$D_{\text{Real}} = W_1x_1 + W_2x_2, \qquad S_{\text{Real}} = D_{\text{Real}} + B$$

In Real mode the weighted-products trace already has the same scale as S.

Fixed INT performs several separate operations. It rounds the parameters to integers, quantizes each normalized input to an integer count between 0 and P, forms the integer weighted-products sum D, divides D by P, rounds that scaled sum, and only then adds the integer bias. It does not round the two products separately or divide the bias by P.

$$\widehat{W}_i = \mathrm{round}(W_i), \quad \widehat{B} = \mathrm{round}(B), \quad X_{iP} = \mathrm{round}(P\,\mathrm{clip}(x_i, 0, 1))$$

round means JavaScript Math.round. The evaluated normalized input is X_{ip}/P.

$$D_{\mathrm{INT}} = \widehat{W}_1 X_{1P} + \widehat{W}_2 X_{2P}, \qquad S_{\mathrm{INT}} = \mathrm{round}\left(\frac{D_{\mathrm{INT}}}{P}\right) + \widehat{B}$$

Unlike Real D, INT D contains an additional input-scale factor P. Compare S and q across modes, not the unscaled D values.

For half-way cases, Math.round chooses the integer toward positive infinity. This is neither ties-to-even rounding nor the common ties-away-from-zero convention. Negative ties are especially important when reproducing the node in another language.

Input to Math.round	Result
1.5	2
0.5	1
-0.5	-0, displayed as 0
-1.5	-1
-2.5	-2

JavaScript can retain a signed negative zero internally. In this application, zero comparisons and activation outputs behave as zero; normal text display and JSON serialization show 0. The sign does not create an additional output level. Use Math.round semantics when porting the calculation rather than assuming another language's default round function is equivalent.

At $P \le 1024$, the bounded INT path stays comfortably within JavaScript's exact integer range: $|D| \le 6P^2$, at most 6,291,456. Division by a power-of-two P is exactly representable here before rounding. Fixed INT is therefore a faithful integer-valued simulation for these bounds, while still using Number storage rather than a native integer or hardware-performance benchmark.

2.4 P, quantized inputs and what the application stores

The available P values are 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024. INT has P + 1 distinct input levels: 0, 1/P, 2/P, ..., 1. Raw INT outputs are integers from 0 to P, so normalized output levels also lie on that grid, although an individual node may use only some of them.

P	INT input levels	Input increment	Parameter bounds
1	2	1	±3
4	5	1/4	±12
16	17	1/16	±48
64	65	1/64	±192

P	INT input levels	Input increment	Parameter bounds
1024	1025	1/1024 = 0.0009765625	±3072

Switching the application to INT changes stored weights and bias to rounded integers and replaces stored inputs with their snapped normalized values. Switching back to Real uses those current values; it does not recover the earlier fractions. Undo, a pinned baseline or a saved experiment can recover an earlier state. Likewise, increasing P does not reconstruct an input originally entered before a coarser quantization; re-enter the requested input if you want a fresh comparison at the finer grid.

With Preserve ratios when P changes enabled, changing from P to P′ multiplies each parameter by r = P′/P. In Real mode, this preserves normalized output for the same inputs, subject to ordinary floating-point limitations. It scales S and Y together. The weights remain within the proportional ±3P′ limits when starting from a valid bounded model.

$$P \longrightarrow rP, \quad (W_1, W_2, B) \longrightarrow (rW_1, rW_2, rB) \quad \Rightarrow \quad q_{\text{Real}} \text{ unchanged}$$

This invariance concerns normalized output, not the raw numerical magnitude of Y.

INT additionally rounds scaled parameters and re-quantizes the currently stored inputs. The rounding of D/P can also change the normalized result even when ratios are preserved exactly. With ratio preservation off, the raw parameters stay numerically the same except for INT rounding or clamping to the new bounds; their ratios to P change, so the surface generally changes.

Real parameter sliders move in steps of P/64; their number fields accept finer fractional values. INT parameter steps are 1. Number fields normally display at most eight decimal places, while JSON and numerical exports retain the stored Number value. A rounded field or chart label should not be treated as evidence that the underlying value is exactly that short decimal.

2.5 Worked example 1 - exact XOR corners, graded interior

Select XOR in Real mode at P = 16 and use $x_1 = 0.70$, $x_2 = 0.25$. The preset supplies $W_1 = W_2 = 16$ and B = 0.

1. **Form the weighted sum**
   D = 16 × 0.70 + 16 × 0.25 = 11.2 + 4 = 15.2; S = D + 0 = 15.2.
2. **Apply the region rule**
   Because $0 < 15.2 \leq 16$, the point lies in region II. Y = S = 15.2.
3. **Normalize and compare**
   q = 15.2/16 = 0.95. Meanwhile the binary sums are 0, 16, 16, 32, producing [0, 1, 1, 0]. Exact XOR and the interior value 0.95 are fully consistent.

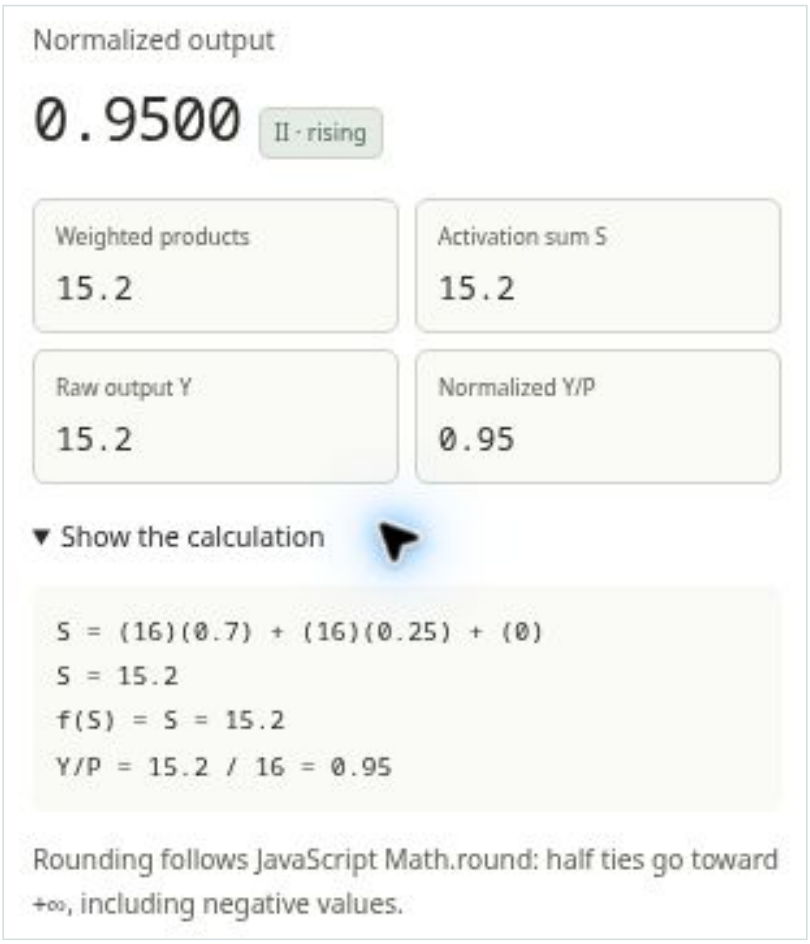


Figure 6. Live Real-mode trace for XOR at P16 and inputs 0.7, 0.25: S = 15.2, Y = 15.2 and Y/P = 0.95.

2.6 Worked example 2 - Asymmetric at P = 4 and 1024

Use the Asymmetric preset with requested inputs $x_1 = 0.38$ and $x_2 = 0.66$. At P = 4 the parameters are $W_1 = 4$, $W_2 = 2$, B = 0. First record the Real result, then select INT to observe both input snapping and scaled-sum rounding.

1. **Real at P = 4**
   S = 4 × 0.38 + 2 × 0.66 = 2.84. This is region II, so Y = 2.84 and q = 0.71.

2. **Quantize the INT inputs**
   X_{1p} = round(1.52) = 2 and X_{2p} = round(2.64) = 3. The effective inputs become 2/4 = 0.5 and 3/4 = 0.75.

3. **Complete the INT calculation**
   D = 4 × 2 + 2 × 3 = 14. Then S = round(14/4) + 0 = round(3.5) = 4. This endpoint belongs to region II; Y = 4 and q = 1. The change from the original Real result is 0.29.

Now increase P to 1024 with parameter ratios preserved. W_1 becomes 1024 and W_2 becomes 512. If you simply keep the already snapped inputs, they remain 0.5 and 0.75; S becomes 896 and q becomes 0.875. Re-enter 0.38 and 0.66 to compare the original requested point at the finer grid.

$$X_{1P} = 389, \quad X_{2P} = 676, \quad D = 1024(389) + 512(676) = 744448$$

These counts follow round(1024 × 0.38) and round(1024 × 0.66).

$$S = \text{round}(744448/1024) = 727, \qquad q = 727/1024 = 0.7099609375$$

The re-entered P1024 result differs from the original Real value 0.71 by only 0.0000390625.

State	Effective x_1	Effective x_2	Normalized output
Real, original request	0.38	0.66	0.71
INT P4	0.5	0.75	1
INT P1024, old snapped inputs	0.5	0.75	0.875
INT P1024, request re-entered	389/1024	676/1024	0.7099609375

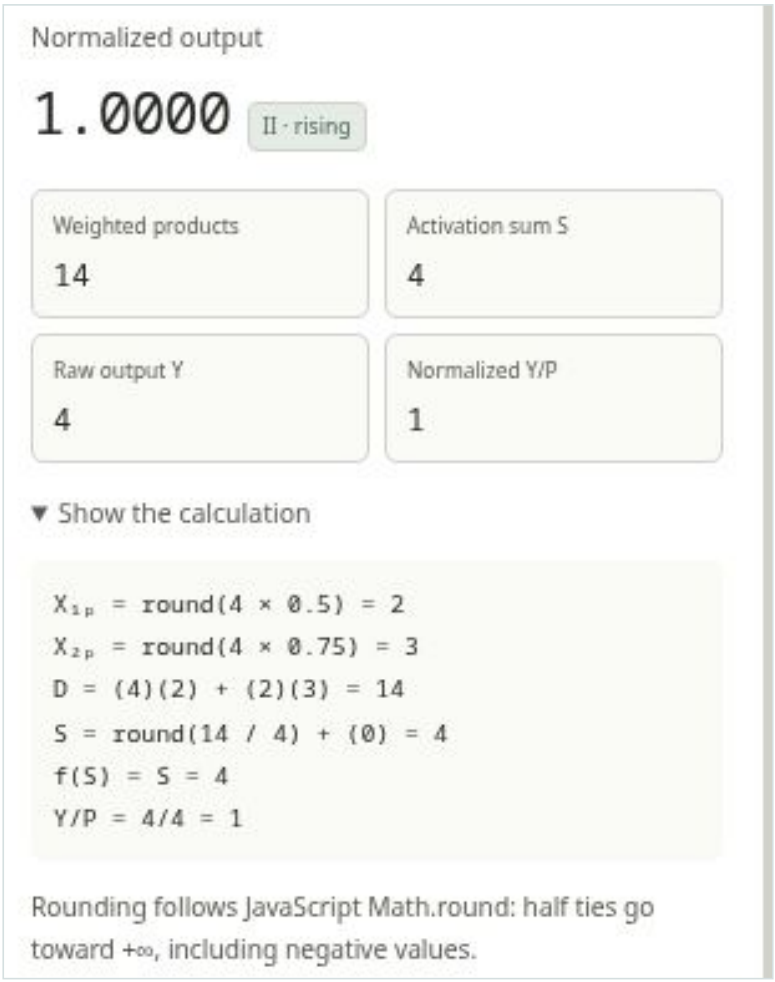


Figure 7. The live P4 INT trace after entering requested inputs 0.38, 0.66 in the Asymmetric preset. The stored effective inputs are 0.5, 0.75; D = 14 and round(14/4) = 4.

Re-enter requested coordinates when comparing P
INT mode stores snapped normalized inputs. Increasing P later preserves those already-snapped coordinates; it cannot reconstruct the earlier requested values. Re-enter 0.38 and 0.66 after choosing P1024 to reproduce the finer-precision example.

2.7 Worked example 3 - a negative half-way sum

Set P = 4, W_1 = -3, W_2 = 0, B = 3, and inputs x_1 = 0.5, x_2 = 0. Both inputs already lie on the P4 grid. This example isolates scaled-sum rounding from input snapping.

1. **Real reference**
   D = -3 × 0.5 = -1.5. Adding B gives S = 1.5, in region II; Y = 1.5 and q = 0.375.

2. **INT weighted products**
   X_{1p} = 2, X_{2p} = 0 and D = (-3) × 2 + 0 × 0 = -6. Dividing by P gives -1.5.

3. **Apply the actual tie rule**
   Math.round(-1.5) = -1, so S = -1 + 3 = 2. Region II gives Y = 2 and q = 0.5. A ties-away-from-zero routine would instead produce S = 1 and q = 0.25, which would not reproduce this application.

Implementation reference: Numerical core and state handling [R1, R2]; JavaScript rounding [R8].

CHAPTER 03

Presets and Boolean interpretation

3.1 Boolean truth vectors, nearest gates and loss measures

Every truth vector in this guide uses the order 00, 01, 10, 11, with A = x_1 and B = x_2. The logical input B in a preset name is distinct from the bias parameter B. The application evaluates exactly those four binary corners; the live input can be somewhere else. A Boolean preset name describes corner behavior, not a claim that all interior inputs produce only 0 or 1. The training target is an independent selection: loading a preset does not automatically change it.

$$\mathrm{MAE}(g) = \frac{1}{4}\sum_{j=0}^{3} |q_j - t_{g,j}|$$

Nearest compares the node against all 16 Boolean target vectors using mean absolute error.

An exactly equal nearest-score tie uses the declared gate order in the following table; there is no tolerance that groups merely similar scores. For example, [0.5, 0.5, 0.5, 0.5] is equally distant from every gate, and FALSE is reported first. Asymmetric [0, 0.5, 1, 0.5] reports A AND NOT B, tied with other equally near alternatives at MAE 0.25.

$$\text{Exact gate } g \iff |q_j - t_{g,j}| < 10^{-8} \text{ for every } j$$

Exact classification is a strict per-row comparison; equality with the tolerance does not pass.

$$\mathrm{MSE}(g) = \frac{1}{4}\sum_{j=0}^{3} (q_j - t_{g,j})^2$$

Training uses normalized mean squared error against the currently selected target. Target matched uses MSE ≤ 10^{-12}.

These are different tests. For Real P = 16, $W_1 = W_2 = 0$ and B = 15.9999984, every corner is 0.9999999. Its TRUE-target MSE is approximately 10^{-14}, so search can already report Target matched; the per-row error is about 10^{-7}, so Exact TRUE is absent. Displayed rounding can make both outputs and loss look exact even when the stricter classification is not satisfied.

For supported INT scales, a nonzero corner error is at least 1/P, making nonzero four-corner MSE at least $1/(4P^2)$. Even at P1024 this lower bound is about 2.384×10^{-7}, well above 10^{-12}. Thus a valid INT node cannot pass the training tolerance with a nonzero corner error; the tolerance distinction above concerns Real mode.

3.2 All 16 Boolean presets and four graded experiments

The tables below provide the exact preset definitions, expressed as coefficients relative to P. Multiply each coefficient by P to obtain the stored parameter before any INT rounding. A preset is a model configuration, while Target function is a separate training setting.

Coefficients below are normalized ratios. Multiply each by the chosen P to obtain the actual W_1, W_2 and B. Every Boolean coefficient is integral, so these presets reproduce the listed binary vector in both modes at every supported P. This table also gives the declared order used for nearest-score ties.

Preset	W_1/P	W_2/P	B/P	q at 00, 01, 10, 11
FALSE	0	0	0	[0, 0, 0, 0]
AND	1	1	-1	[0, 0, 0, 1]
A AND NOT B	1	-1	0	[0, 0, 1, 0]
A (X_1)	1	0	0	[0, 0, 1, 1]
NOT A AND B	-1	1	0	[0, 1, 0, 0]
B (X_2)	0	1	0	[0, 1, 0, 1]
XOR	1	1	0	[0, 1, 1, 0]
OR	2	2	-1	[0, 1, 1, 1]
NOR	-1	-1	1	[1, 0, 0, 0]
XNOR	1	1	1	[1, 0, 0, 1]
NOT B	0	-1	1	[1, 0, 1, 0]
A OR NOT B	2	1	1	[1, 0, 1, 1]
NOT A	-1	0	1	[1, 1, 0, 0]
NOT A OR B	1	2	1	[1, 1, 0, 1]
NAND	-2	-2	3	[1, 1, 1, 0]
TRUE	0	0	1	[1, 1, 1, 1]

The graded experiments intentionally have fractional corner outputs. Their names suggest a shape rather than an exact Boolean truth table. The vectors below apply in Real mode at every P and in INT when the preset parameters are exactly representable.

Graded preset	(W_1/P, W_2/P, B/P)	Corner vector
Soft XOR	(0.75, 0.75, 0)	[0, 0.75, 0.75, 0.5]
Graded AND	(0.5, 0.5, 0)	[0, 0.5, 0.5, 1]
Asymmetric	(1, 0.5, 0)	[0, 0.5, 1, 0.5]
Graded NOR	(-0.5, -0.5, 1)	[1, 0.5, 0.5, 0]

When a graded preset is selected in INT and its fractional ratios cannot be represented at the current P, the interface raises P to 16 and explains the change. For example, Soft XOR needs quarter-scale coefficients, so P = 1 or 2 is insufficient. At $P \geq 4$ all four graded presets have integral parameters. This promotion avoids silently changing the intended graded coefficients.

Implementation reference: GATES, EXPERIMENTS, exactGate, nearestGate and mse [R1].

CHAPTER 04

Surfaces, probes and diagnostics

4.1 What the surface and reference mesh represent

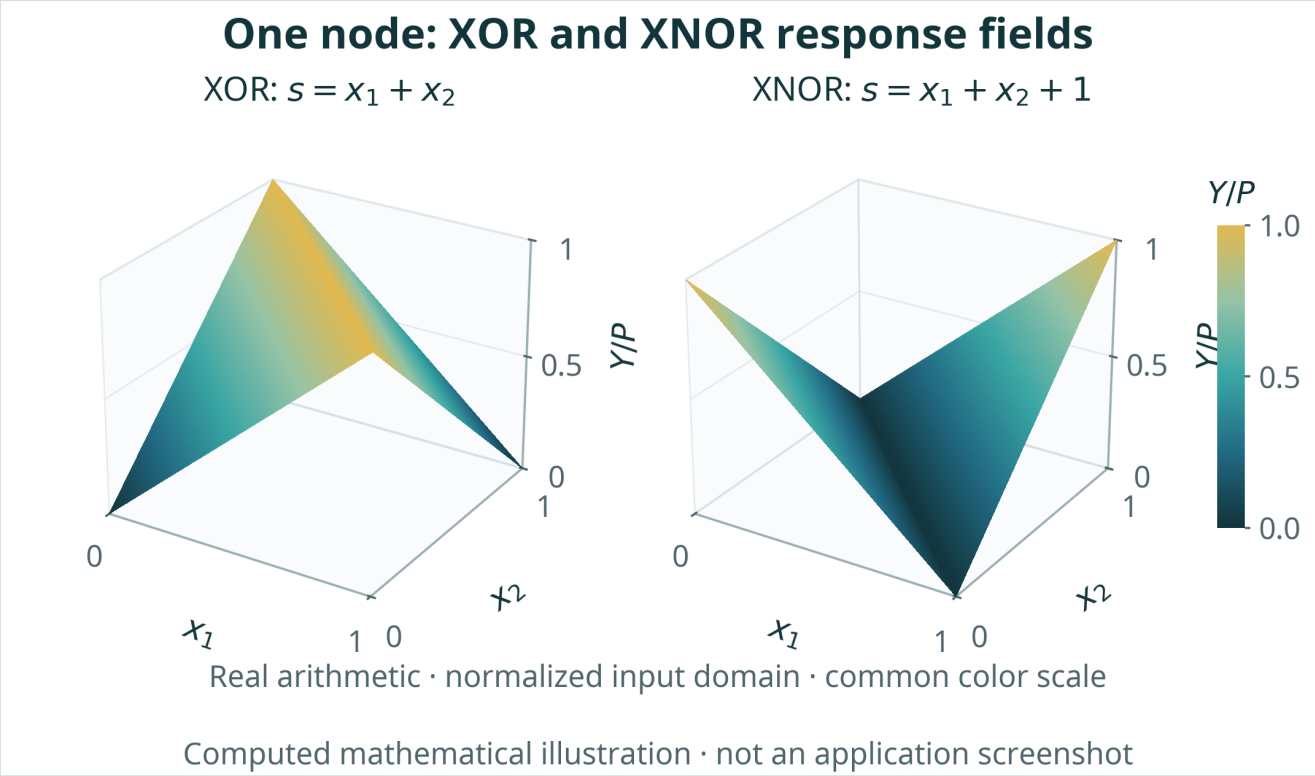


Figure 8. Computed Real-mode surfaces for the canonical XOR and XNOR presets. The four Boolean corners can be exact even when most interior points have graded outputs.

In Real mode, S is an affine plane over the input square. Within each activation region, normalized output is another plane or a constant plateau. The borders between those pieces are straight lines clipped to the input square:

$$W_1 x_1 + W_2 x_2 + B = kP, \qquad k \in \{0, 1, 2, 3\}$$

A border outside the input square is not visible. If both weights vanish, the output is constant and the line description degenerates.

A displayed n × n mesh samples $(n + 1)^2$ vertices and joins them into $2n^2$ triangles. Thus 64 × 64 means 4,225 vertices and 8,192 triangles, not 64^2 independent exact faces of the underlying function. A triangle crossing an activation border can interpolate across a kink. In INT, the true evaluated function has quantization steps; the drawn triangles interpolate between samples and can look smooth. P = 1024 does not request a 1024 × 1024 mesh.

WebGL uses the selected resolution. The Canvas2D fallback caps the effective resolution at 48 for responsiveness. Mesh resolution changes appearance and rendering cost, not the numerical evaluator, truth table or training objective. Surface picking finds a location on the sampled mesh and then evaluates the numerical core at the corresponding input coordinates.

Real reference mesh draws a sparse gold grid only in INT mode. It evaluates the current stored parameters in Real mode; it cannot restore pre-INT fractional parameters. For visibility, that grid is lifted by 0.004 in

normalized output before projection. The small visual separation is intentional and should not be read as measured numerical error.

The cross-section chart also compares a dashed Real reference with the active mode at the same fixed live-input coordinate. In Real mode the curves coincide. It samples between 256 and 4,096 intervals, according to P, and joins samples with line segments; the numerical trace remains the authoritative value at a selected point.

4.2 Scene controls and display options

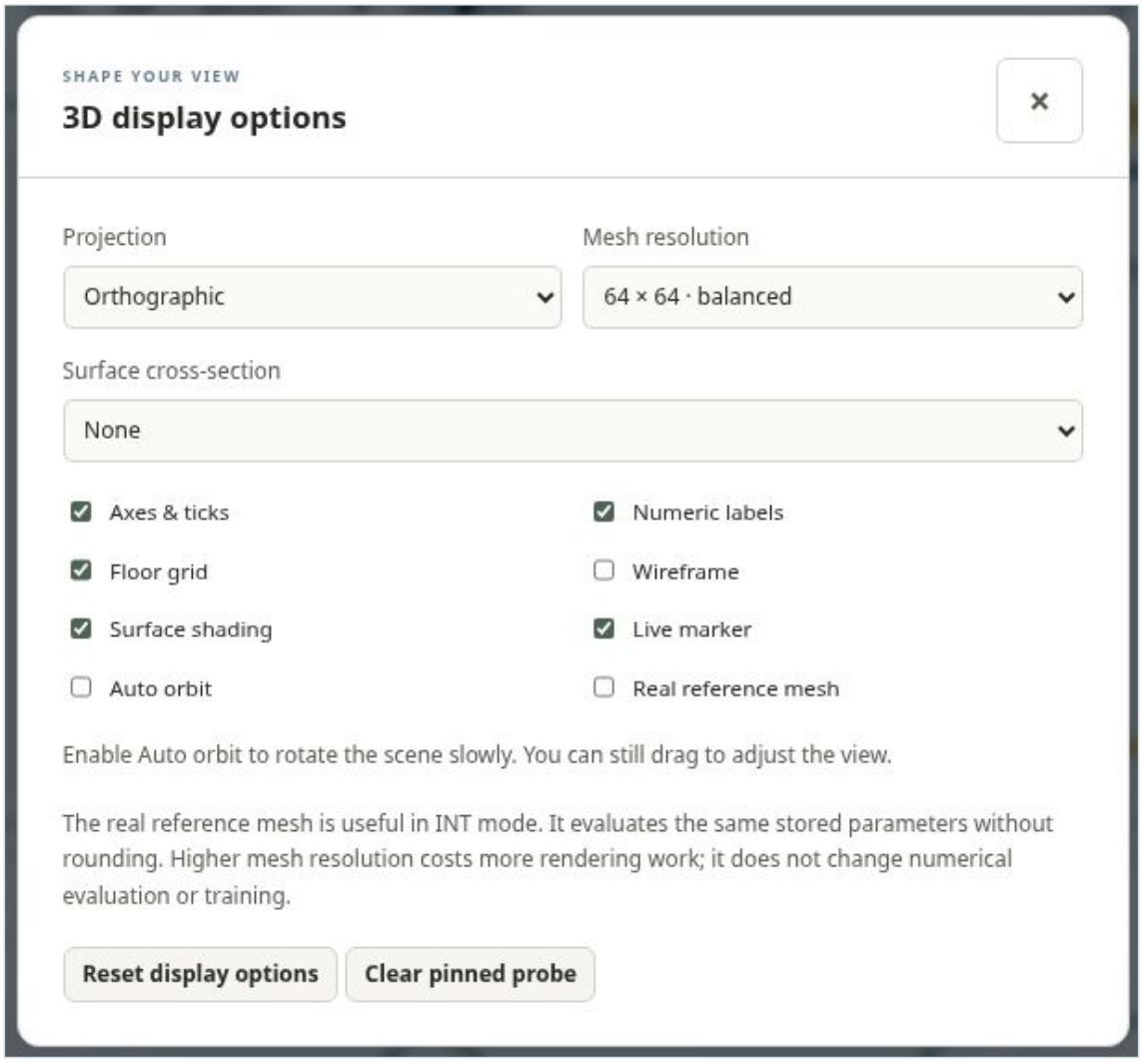


Figure 9. View options exposes projection, sampling resolution, cross-sections and rendering toggles. These settings do not alter the node parameters.

Control	Choices / default	Use
3D; Top; Front; Side	3D initially	Standard camera orientations. Front looks along X_2; Side along X_1.
Surface colors	Spectrum; Ocean / Cyan; Thermal; Terrain; Graphite; Cobalt / Gold	Spectrum is default. Color represents normalized output from 0 to 1; this palette is separate from the interface theme.
View options	3D display options dialog	Opens the controls listed below.
Reset camera	Initial 3D orientation	Resets orbit, pan and zoom without changing node parameters or the pinned baseline.

Control	Choices / default	Use
Save surface PNG	Toolbar download arrow	Same export as Surface PNG in Save / share.
Projection	Orthographic (default); Perspective	Changes camera projection only.
Mesh resolution	32 × 32 · lighter; 64 × 64 · balanced (default); 96 × 96; 128 × 128; 192 × 192 · finer	Controls sampled surface density, independently of P and training.
Surface cross-section	None (default); X_1 fixed at live input; X_2 fixed at live input; Both live-input cuts	Adds cuts through the current probe coordinates.
Axes & ticks; Numeric labels; Floor grid	All on	Show coordinates and orientation aids. Numeric labels need Axes & ticks enabled.
Wireframe	Off	WebGL shows a thinned rectangular sample grid, without triangle diagonals. The Canvas2D fallback outlines individual triangles.
Surface shading	On	Enables WebGL lighting; has no lighting effect in the Canvas2D fallback.
Live marker	On	Shows the current probe and its guides. Corner, hover and pinned-probe markers are separate.
Auto orbit	Off	Rotates slowly; pauses during pointer interaction, when the tab is hidden, or when the scene is offscreen.
Real reference mesh	Off	In INT mode, overlays the same stored parameters evaluated without integer rounding. No additional mesh appears in Real mode.
Open Appearance	Shown when Reduce motion blocks orbit	Closes View options, opens Appearance, and focuses Reduce motion without changing the preference.
Reset display options	Restores all defaults above	Also resets the camera and clears the pinned surface probe; keeps the node, appearance and baseline.
Clear pinned probe	Always available	Removes the selected surface highlight and hover readout; keeps the current live inputs.

In 2.0.1, Reduce motion disables and unchecks Auto orbit, with a persistent explanation inside View options. Turn Reduce motion off explicitly if rotation is wanted, then enable Auto orbit. Merely opening Appearance or disabling Reduce motion does not start rotation.

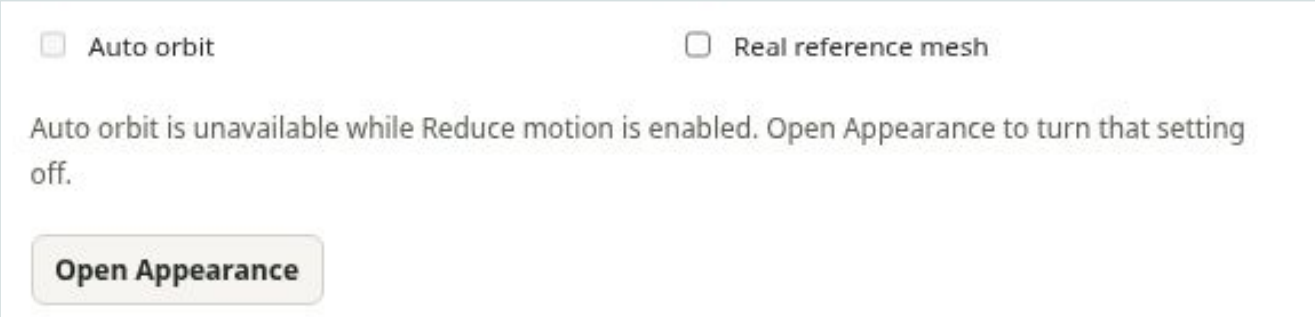


Figure 10. Version 2.0.1 explains why Auto orbit is unavailable when Reduce motion is enabled. Open Appearance takes you directly to that preference.

4.3 Mouse, touch and keyboard reference

Focus / location	Gesture or key	Action
Surface	Left drag or one-finger drag	Orbit; a click/tap without a drag selects a numerical probe.
Surface	Shift + drag; middle/right drag	Pan. The surface suppresses its context menu.
Surface	Wheel; two-finger pinch	Zoom, bounded to 0.4...4 times the renderer's reference scale.
Surface	Hover	Inspect the picked input and exact core-computed output without changing live inputs.
Surface focused	Arrow keys; + or =; -	Orbit; zoom in; zoom out.
Surface	Double-click; Home or 0 when focused	Reset camera.
VIEW orientation widget	Click X_1 / Y / X_2 / center	Side / Top / Front / 3D camera respectively.
Input pad or Precision map	Click/drag; focused arrows	Set inputs. Arrow step is 0.01 in Real or 1/P in INT; Shift multiplies it by ten.
Truth-table row	Click; focused Enter or Space	Set the live inputs to that row's Boolean corner.
Analysis or inspector tabs	Left/Right; Home/End	Select adjacent, first or last tab and move focus.
Dividers	Drag; focused arrows	Resize in 10 px keyboard increments. Up increases and Down decreases the panel size. Right increases the left/analysis size but decreases the right sidebar; Left reverses this.
Ordinary controls	Tab / Shift+Tab; native activation keys	Move focus and operate buttons, selectors, checkboxes and expandable sections. Undo/Redo use their toolbar buttons; no global application Ctrl+Z binding is defined.

4.4 Analysis, inspection, probes and comparisons

Control / display	Purpose
Activation	Default analysis tab: activation curve, four Boolean markers and the live probe.
Cross-section → Fix	X_2 · vary X_1 is default; X_1 · vary X_2 is the alternative. Uses the live coordinate for the fixed input. Solid current-mode and dashed Real curves are compared.
Precision map	Shows sampled absolute INT-versus-Real differences at 65 × 65 midpoints. Mean and maximum are sampled statistics. It is also an input pad.
Inspect → Show the calculation	Expands the arithmetic trace. Four-point truth table rows remain the Boolean corners, regardless of the live probe.
Pin current as baseline	Copies the complete current node snapshot for comparison. Pinning again replaces it.
Restore baseline / Clear	Shown with a pinned baseline. Restore reapplies its node, inputs, target and rescale preference; Clear removes only the baseline.
Exact / Nearest classification	Describes the four Boolean outputs. Baseline comparison uses the currently selected target for both MSE values.

A pinned surface probe is a coordinate highlight, not a saved model. It follows the response at those coordinates when the model changes; moving inputs elsewhere can leave the pin and live marker at different positions. A baseline instead preserves an earlier numerical node for four-output and error comparison. Neither survives reload or appears in experiment JSON.



Figure 11. Cross-section through the Asymmetric P4 INT example at fixed x2 = 0.75. The solid sampled integer cut and dashed Real reference use the same stored parameters.

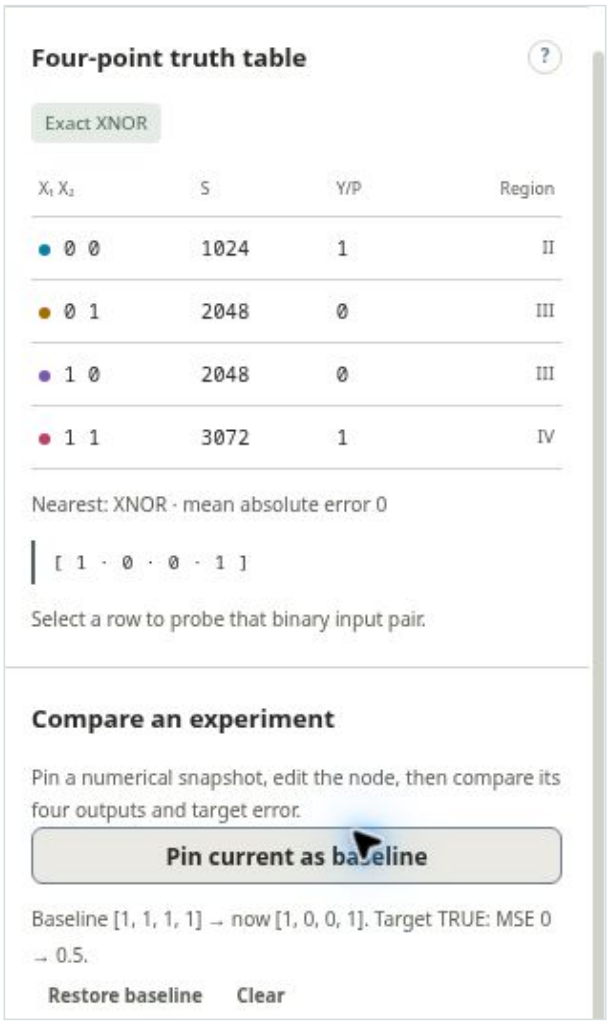


Figure 12. The four-point table and baseline summary. Here a TRUE baseline is compared with XNOR using the still-selected TRUE training target: MSE changes from 0 to 0.5.

4.5 Reading the precision map without sampling traps

Precision compares INT and Real at the same requested sample coordinates using the same current stored parameter values. In Real mode this comparison includes potential parameter rounding. After switching to INT, the parameters may already have been rounded, so the comparison is against that updated model. The map is not a remembered comparison with an earlier experiment.

$$u_i = \frac{i + 1/2}{65}, \quad v_j = \frac{j + 1/2}{65}, \quad i, j = 0, \ldots, 64$$

The diagnostic uses 4,225 cell-center samples, not the input-square endpoints.

$$e_{ij} = |q_{\mathrm{INT}}(u_i, v_j) - q_{\mathrm{Real}}(u_i, v_j)|, \quad \mathrm{MAE} = \frac{1}{4225} \sum_{i,j} e_{ij}$$

The other reported metric is the maximum of these sampled absolute errors. Neither metric proves a global error bound.

The Surface CSV uses a different 65 × 65 grid: i/64 and j/64, including both endpoints, in the active arithmetic mode. These coordinates align with INT input grids for P ≥ 64. For the XOR preset, every such endpoint sample agrees with Real at P64 and P1024, even though errors remain between them. The midpoint precision map exposes those errors. A zero difference on the CSV endpoint grid is therefore not proof of equivalence over the whole input square.

Asymmetric preset	Sample mean error	Sample maximum error
P4	0.0969686	0.3076923
P16	0.0242986	0.0769231
P64	0.0060765	0.0192308
P1024	0.0003798	0.0012019

The map rescales its colors to the current sampled maximum: dark blue is zero and gold is that maximum. A gold area at P1024 can represent much less error than a gold area at P4; compare the numbers, not color alone. Moving the live input changes the marker but not the error field, because the field depends on the parameters and P. Changing the training target alone also does not change it.

The preceding numbers are reference results for the Asymmetric coefficients with ratios preserved. They do not assert that every arbitrary parameter change or increase of P must reduce every individual error. Parameter rounding, retained snapped inputs and activation-region crossings all affect a particular comparison.

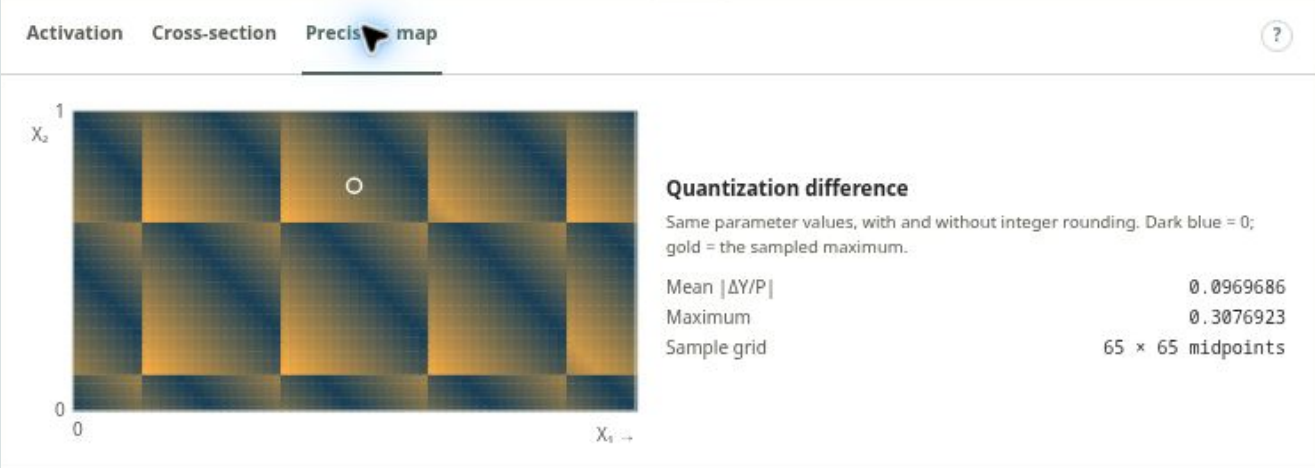


Figure 13. Asymmetric P4 precision map: sampled mean 0.0969686 and maximum 0.3076923. Dark blue is zero difference; gold is this map's own sampled maximum.

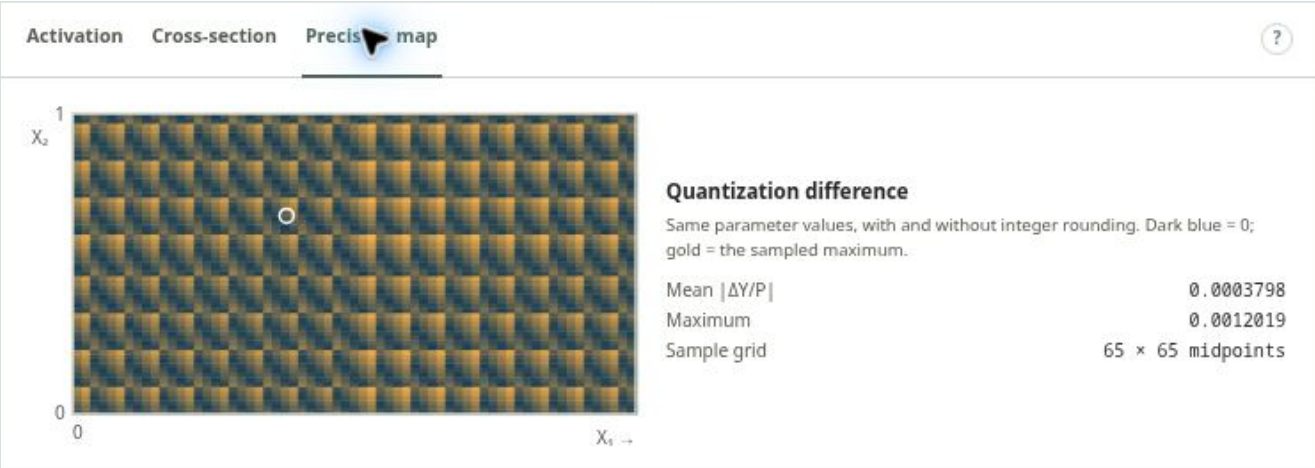


Figure 14. Asymmetric P1024 precision map: sampled mean 0.0003798 and maximum 0.0012019. The palette is rescaled, so a gold pixel here represents a much smaller error than gold in the P4 map.

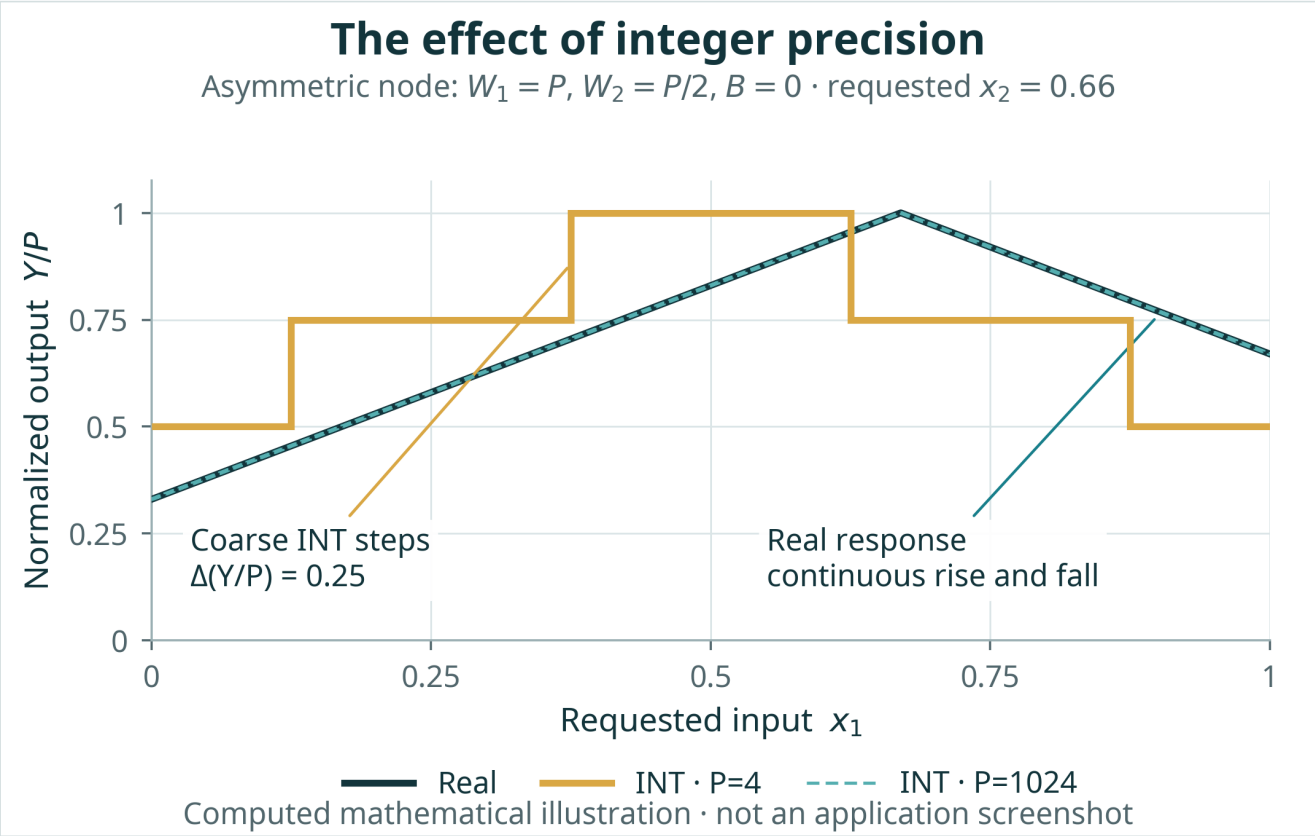


Figure 15. Computed comparison at fixed requested x2 = 0.66. The P4 staircase is exact for the stated rounding rule; the P1024 response follows the Real curve much more closely.

Implementation reference: Renderer [R3] and diagnostic/controller functions [R2].

CHAPTER 05

Single-node training

5.1 Single-node training: what is optimized

Train adjusts only W_1, W_2 and B. P and arithmetic mode remain fixed throughout a run. Its examples are always the four Boolean input pairs, in 00, 01, 10, 11 order. The selected target supplies four binary desired outputs. Moving live inputs probes the model without changing these examples.

$$E(\theta) = \frac{1}{4}\sum_{i=0}^{3}(\hat{y}_i(\theta) - t_i)^2, \qquad \hat{y}_i = \frac{f(S_i; P)}{P}$$

Here $\theta=(W_1,W_2,B)$; t_i is the selected Boolean target; $\hat{y}_i$ is the normalized prediction. Every candidate is scored using all four rows.

For predictions [0,0,0,0.5] against TRUE=[1,1,1,1], MSE=(1+1+1+0.25)/4=0.8125. This objective fits four examples; it is not a test-set accuracy, confidence, or generalization estimate. The application trains no CNN or network and does not reproduce complete IGL training. WebGL renders the scene; JavaScript reference arithmetic scores candidates. No training-speed or hardware-efficiency benchmark is implied.

5.2 Limits and controls

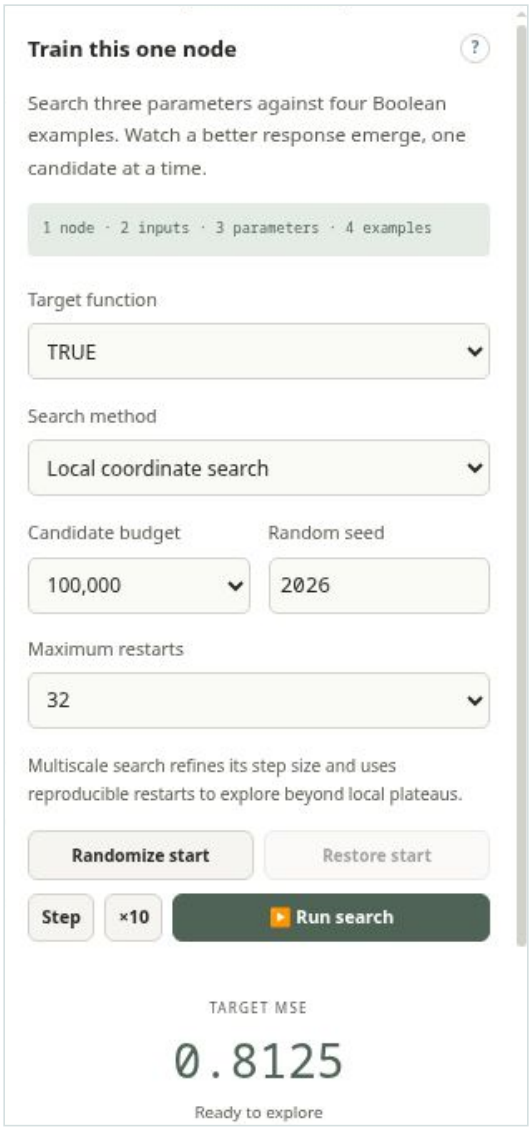


Figure 16. Training controls for Walkthrough 3. P1024 INT, target TRUE, Local coordinate search, candidate budget 100,000 and seed 2026. The starting model has MSE 0.8125.

Setting	Default and accepted values	Effect
Search method	Multiscale + seeded restarts; or Local coordinate search	Controls Run search only.
Candidate budget	Default 12,000; integer 1-100,000	Selector: 1,000 / 5,000 / 12,000 / 50,000 / 100,000. Valid custom imports are retained.
Maximum restarts	Default 32; integer 0-128	Selector: 0 / 4 / 8 / 16 / 32 / 64 / 128. Applies only to multiscale; valid custom imports remain selectable.
Random seed	Default 2026; integer 0-4,294,967,295	Reproducible random starts and restart ordering.
Parameter bounds	$-3P \le W_1, W_2, B \le 3P$	Bounds constrain every candidate; P is a supported power of two through 1024.

Restore start reapplies the complete node snapshot saved before the latest Run search or Step/×10 batch: parameters, P, arithmetic mode, live inputs, target, preset and Preserve ratios flag. It stops any active run

but does not resume one or restore the search controls. Ordinary model edits and imports do not replace that stored start; a new run/batch overwrites it, while Randomize start clears it. Model edits clear the displayed search result, so export history first if you need its record.

5.3 Local search, Step and ×10

A local poll varies one parameter at a time: W_1-δ, W_1+δ, W_2-δ, W_2+δ, B-δ, B+δ. Out-of-range candidates are skipped; every parameter must remain within ±3P. The default δ is 1 in INT and P/4 in Real. The best strict improvement is accepted; equal scores do not replace an earlier winner.

Step runs one local iteration. ×10 runs up to ten and stops early when no improvement is found or the target is solved. Each click creates a new displayed batch and resets its counters/history. These buttons always use local search, independently of the Search method selected for Run search; their one/ten-iteration limits are independent of the Run search candidate budget.

Run search with Local selected repeats coordinate polls until solved, stalled or budget-limited. It never restarts. Step/×10 require improvement greater than 10^{-10}; Run search uses 10^{-14}. Ordinary examples agree, but extremely small Real-mode improvements can be treated differently. A local plateau means only that the tested neighborhood did not improve the objective.

5.4 Multiscale search and deterministic restarts

$$\theta' = \theta + \delta d, \qquad d \in \{-1, 0, 1\}^3 \setminus \{(0, 0, 0)\}, \qquad 3^3 - 1 = 26$$

Each poll considers up to 26 joint parameter directions. Bounds can remove candidates; a solved result or exhausted budget can end a poll early.

The first step is δ=P. A successful poll moves the exploratory current point to its best improving neighbor and retains that step. An unsuccessful poll halves the step. Refinement continues to δ=1 in INT or P/256 in Real. Success at a smaller step does not reset it to P; only a restart does.

Candidate directions use a fixed nested order over -1,0,+1 for W_1, then W_2, then B; B varies fastest. Improvement must exceed 10^{-14}. Equal-scoring later candidates do not replace the earlier winner. Best-so-far parameters update as candidates are evaluated, even before the poll finishes. A solved candidate stops the run immediately.

After stalling at the smallest step, a permitted restart takes the next point from a seeded shuffle of all 343 triples (aP,bP,cP), with a,b,c∈{-3,…,3}. The lattice is independent of the target; the code does not load a target preset. Each restart is scored once, consumes one candidate evaluation, and resets δ to P. A worse exploratory restart cannot overwrite the best-so-far displayed model.

The same version, starting parameters, target, P, mode, algorithm, budget, restart limit and seed reproduce the sequence. Worker timing and batch size do not determine candidate order. Randomize start separately recreates three parameters from the seed: integers in INT or a P/16 grid in Real. Changing the seed changes that start and the restart shuffle, not the four training examples.

5.5 Execution, cancellation and progress

A Web Worker evaluates at most 256 candidates per batch, then requests an 8 ms yield. If the worker is unavailable, the main-thread fallback evaluates at most 128 candidates and requests a 16 ms yield. These are scheduling requests, not guaranteed frame rates or timing measurements. Graphics and live-input inspection remain separate from search scoring.

Stop retains the best result already received by the interface. The controller terminates the worker, clears fallback timers and ignores messages belonging to an obsolete run. Hiding the page stops training; returning does not resume it. Camera movement or changing the live input does not cancel the search.

Changing a model parameter, P, mode, target or preset cancels the active run. A committed valid numeric edit is protected from later progress. A focused number field preserves its current text while the underlying model changes; version 2.0.1 refreshes it when focus leaves. Typing and committing are distinct: do not assume the first keystroke has already stopped the worker.

Readout or status	Exact meaning
Target MSE	Error of the currently displayed node against the selected target.
Candidates	New parameter candidates scored; includes restart points, excludes the initial baseline score. Each candidate evaluates four examples.
Iterations	Run search counts polls finalized before termination. A solving candidate stops the final poll before its counter increment, even when it was the last neighbor. Stop or budget exhaustion can also interrupt a poll. Step/×10 count completed local iterations; a restart point is not an iteration.
Restarts	New coarse-lattice starts actually used, excluding the original start.
History	Run search retains the initial MSE and best improvements. Step/×10 retain the initial MSE and each iteration outcome, which can repeat the loss. Rejected candidates are not logged; long Run histories are thinned as described below.
Ready / Starting / Searching	No active run yet / run initializing / bounded search active; best-so-far node displayed.
Target matched	MSE $\leq 10^{-12}$. Exact gate classification separately requires each corner error $<10^{-8}$; tiny Real residuals can therefore have different labels.
Plateau reached / Local plateau	No improving tested neighborhood remained under the applicable method and restart limit; not proof of impossibility.
Budget reached	Candidate limit reached, possibly mid-poll; best result retained.
Stopped	A run was cancelled, for example by Stop or page hiding. A cancelling model edit then clears the result/history and normally resets the display to Ready to explore.
Local improvement	Step/×10 accepted progress without establishing a solved target.
Error feedback	Invalid options prevent a run; execution errors stop it. Read the message instead of interpreting an error as convergence.

Run search thins its retained history whenever an appended improvement brings the count above 512: 513 records become 257 by keeping the first, last and even-indexed records. Repeated thinning preserves the initial score and latest recorded improvement, not every earlier improvement. The chart extends the last retained loss horizontally to the displayed candidate count (with a minimum horizontal plotting range of one). That display-only endpoint is not added to Search history CSV. Consequently the CSV's last candidate count can precede the final Candidates readout when no improvement occurred near termination.

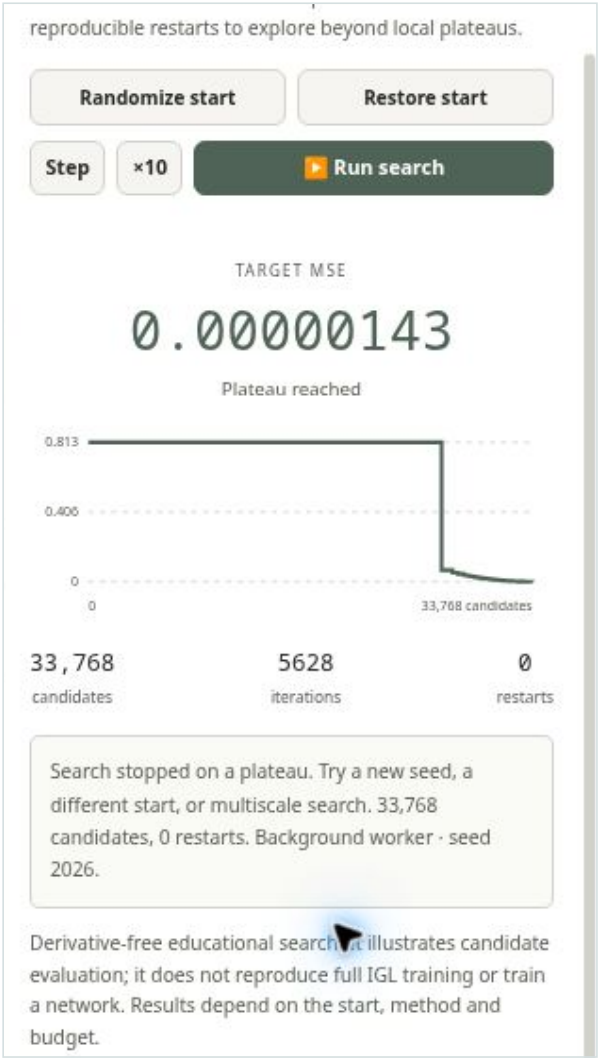


Figure 17. Local search: plateau after 33,768 candidates and 5,628 iterations; best MSE is approximately 0.00000143.



Figure 18. After Restore start, multiscale search solves the same target after 816 candidates, 36 iterations and 3 restarts.

5.6 Training controls and built-in walkthroughs

Control	Default / choices	Behavior
Target function	XOR; all 16 named Boolean functions	Changes the objective, not the starting weights; stops an active run.
Search method	Multiscale + seeded restarts (default); Local coordinate search	Applies to Run search; Step and ×10 always use local coordinate steps.
Candidate budget	12,000; menu 1,000 / 5,000 / 12,000 / 50,000 / 100,000	Imported integer budgets 1…100,000 are retained with an imported label.
Random seed	2026; integer 0…4,294,967,295	Controls reproducible starts and restart ordering.
Maximum restarts	32; menu 0 · no restarts / 4 / 8 / 16 / 32 / 64 / 128	Multiscale only. Imported integer limits 0…128 are retained.
Randomize start	Uses current seed	Changes three bounded parameters, keeps target/P/mode, and clears the previous Restore start snapshot.
Restore start	Enabled after a run or local-step batch	Restores the pre-batch node snapshot, including P, mode, inputs, target and Preserve ratios; stops an active run. Search method, seed, budget and restart settings remain unchanged.
Step / ×10	One / up to ten local iterations	Replace displayed batch history; ×10 stops early on a solution or no improvement.
Run search / Stop	Run replaced by Stop while active	Run shows best-so-far parameters. Stop retains the best result received.
Guide → Start with XOR	Real, XOR, live inputs 1 and 1	Selects Activation at the current P; target remains unchanged.
Explore graded inputs	Real, Soft XOR, inputs 0.7 and 0.35	Selects Cross-section and both surface cuts.
See integer precision	Asymmetric; P = 4; INT	Requests inputs 0.38/0.66, which snap to 0.5/0.75. Selects Precision map and Real reference mesh.
Search for a target	Target XOR; current seed	Selects Train and randomizes; the user starts Run search.
Guide expandable sections	What “integer” means here; Reading the surface correctly; Controls and accessibility; Privacy and scope	Open explanatory text without changing the model.

During a run, search method, budget, seed, restart limit, Randomize start, Step and ×10 are disabled. Camera, appearance and live-input probing remain usable. Changing model parameters, P, mode, preset or target stops the run and clears its displayed history. A Web Worker normally performs bounded batches; if unavailable, the same search uses short CPU batches. Hiding or leaving the page stops training; it does not resume automatically when you return. Offscreen scene rendering may pause without stopping a still-visible page's search.

Implementation reference: createSearch [R1], controller training events [R2] and worker [R4].

CHAPTER 06

Appearance, layout and history

6.1 Appearance: all themes, fonts and sizes

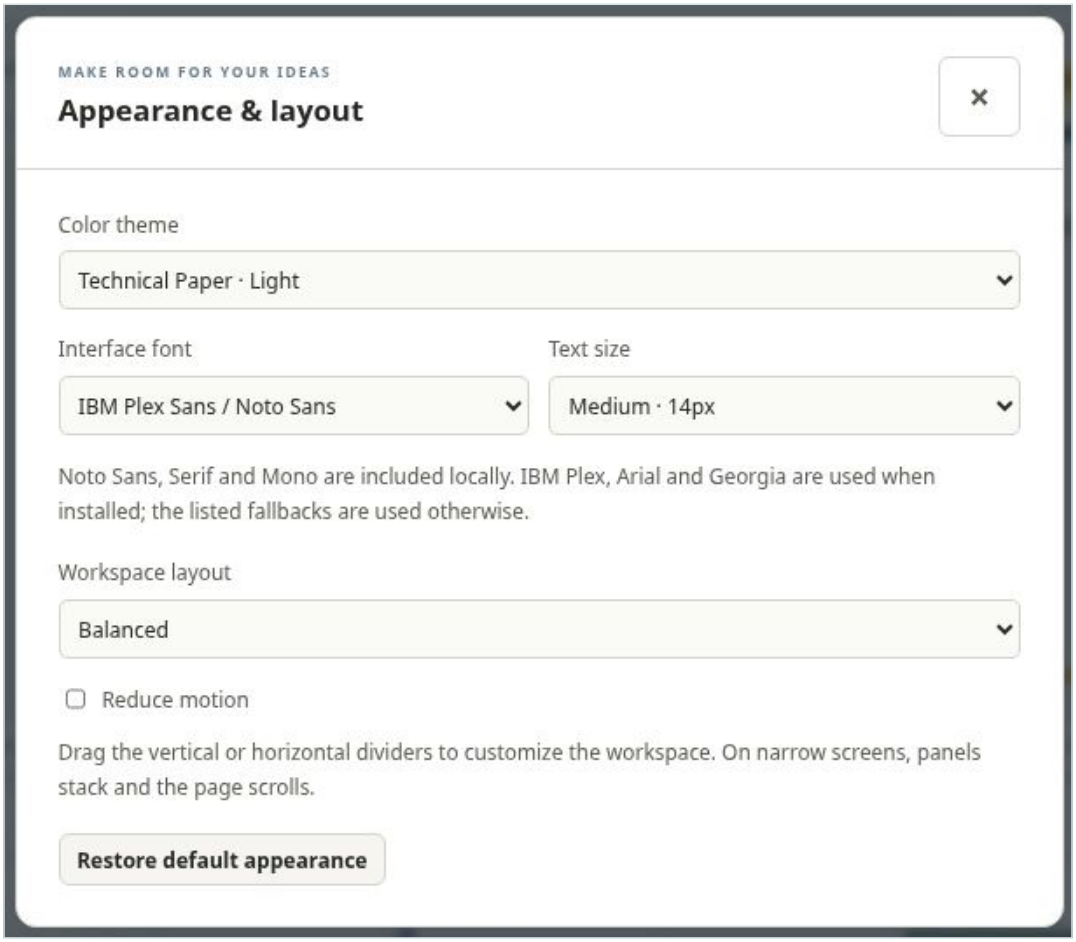


Figure 19. Appearance and layout. The shown Technical Paper theme is used for most close-ups in this manual.

Defaults are Graphite · Dark, IBM Plex Sans / Noto Sans, Medium · 14px and Balanced. Reduce motion initially follows the operating-system preference detected at page load, unless a saved preference overrides it. Appearance choices apply and save immediately. Restore default appearance restores these defaults, including that initial motion preference.

Color theme - exact menu label	JSON value
Graphite · Dark	graphite
Midnight · Dark	midnight
Electric Cobalt · Dark	cobalt
Amethyst · Dark	plum
High Contrast · Dark	contrast
Phosphor · Dark	terminal

Color theme - exact menu label	JSON value
Ocean Blue · Light	ocean
Evergreen · Light	forest
Warm Sand · Light	sand
Glacier Steel · Light	steel
Ivory & Gold · Light	ivory
Rose Quartz · Light	rose
Technical Paper · Light	paper

Interface font - exact label	JSON value	Actual fallback order
IBM Plex Sans / Noto Sans	plex	Installed IBM Plex Sans → bundled Noto Sans → sans-serif
IBM Plex Sans Condensed / Sans	condensed	Installed IBM Plex Sans Condensed → Arial Narrow → bundled Noto Sans → sans-serif
IBM Plex Serif / Noto Serif	serif	Installed IBM Plex Serif → bundled Noto Serif → serif
IBM Plex Mono / Noto Mono	mono	Installed IBM Plex Mono → bundled Noto Mono → monospace
System Sans	system	system-ui → -apple-system → Segoe UI → sans-serif
Arial / Helvetica	arial	Installed Arial → Helvetica → bundled Noto Sans → sans-serif
Noto Sans · bundled	noto	Bundled Noto Sans → sans-serif
Noto Serif · bundled	noto-serif	Bundled Noto Serif → serif
Noto Mono · bundled	noto-mono	Bundled Noto Mono → monospace
Georgia / Serif	georgia	Installed Georgia → bundled Noto Serif → serif

Regular and bold Noto files are included locally; IBM Plex and other named system families are not downloaded automatically. A missing installed family therefore falls back. Many numerical readouts deliberately keep a monospace font independently of the interface selection.

Text size - exact label	JSON value
Extra small · 12px	xsmall
Small · 13px	small
Medium · 14px	medium
Large · 16px	large

Text size - exact label	JSON value
Extra large · 18px	xlarge
Maximum · 20px	xxlarge

6.2 Workspace layouts and responsive dividers

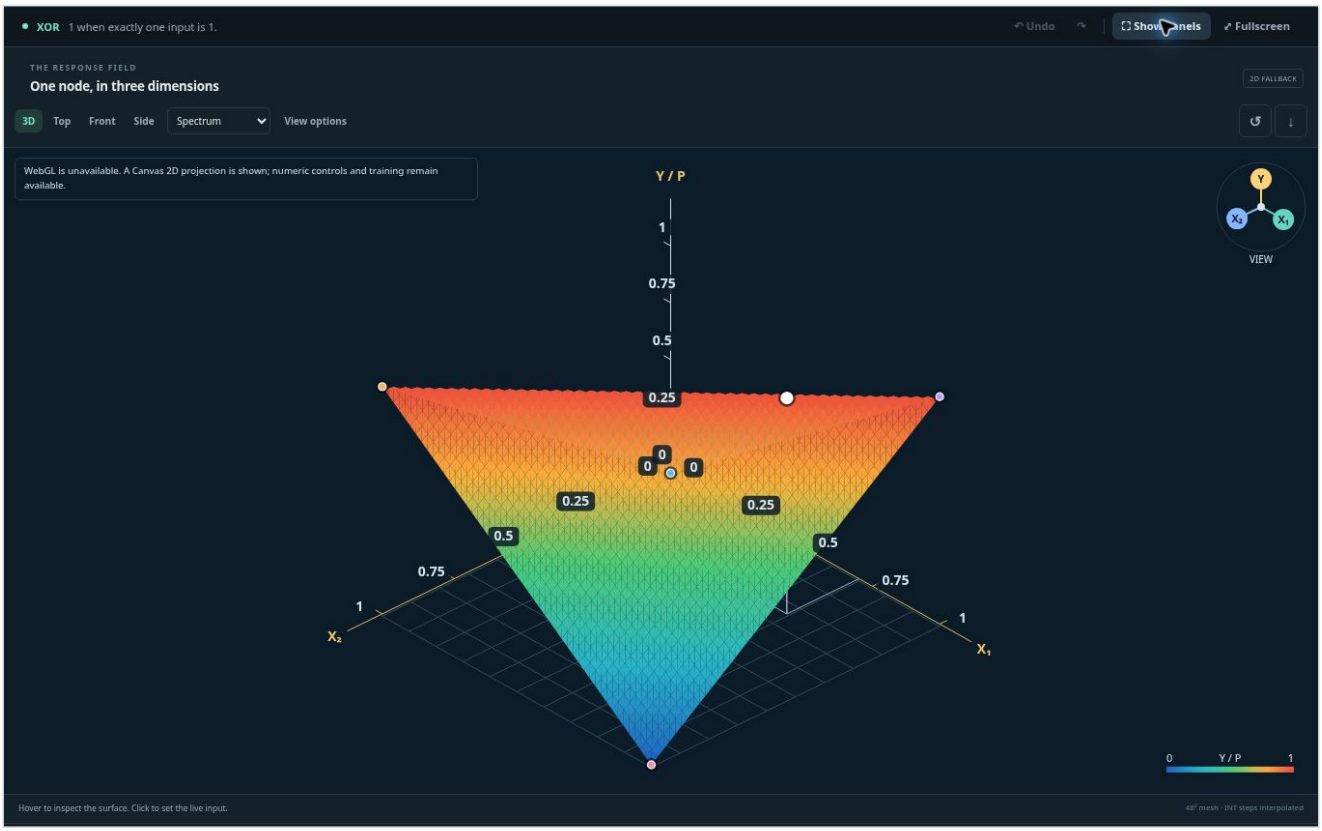


Figure 20. Focus view uses the display area for the scene. Show panels restores the ordinary workspace; Fullscreen is a separate browser request.

Workspace layout	JSON value	Nominal left / right / analysis size, px
Balanced	balanced	294 / 316 / 265
Spacious controls	spacious	340 / 360 / 280
Larger 3D scene	scene	264 / 282 / 220

Selecting a layout clears custom divider sizes. Drag vertical dividers to resize the node/inspection panels; drag the horizontal divider upward to enlarge analysis. Nominal bounds are left 260-460 px, right 280-480 px and analysis 180-520 px. Actual maxima also reserve space for the scene and adapt to window dimensions. Saved desired sidebar widths return when a wider window permits them.

Viewport width	Arrangement
>1380 px	Three columns; center minimum 360 px. At ≥1900 px, panel padding and some scene headings become roomier.
1101-1380 px	Three columns; center minimum 300 px. Version 2.0.1 honors layout/divider widths within the available space.

Viewport width	Arrangement
≤1100 px	Controls occupy a fixed left column; scene and inspector stack on the right. Vertical dividers are hidden.
≤760 px	Scene, controls and inspector stack vertically; the page scrolls. Some toolbar labels collapse to icons; Redo is hidden.
≤480 px	Controls and inspection sections stack further; header Help is hidden, but the Guide tab remains.

Analysis height is additionally limited by the window height; very short windows may require scrolling. Focus view enlarges the scene without altering the saved layout. Bigger text trades visible content density for readability.

6.3 Exactly what Undo and Redo restore

History retains up to 80 model snapshots. Parameter edits, presets, scale/mode changes, target changes, randomization, node restores/imports and training results are covered; a parameter-slider drag is grouped into one edit. A new recorded edit clears Redo. Undo or Redo stops active training and clears the displayed search history.

Camera, palette, view options, appearance, panel sizes, tabs and search-setting changes are not undoable model edits. Moving live inputs and toggling Preserve ratios do not create their own entries. However, snapshots contain inputs and the rescale flag, so undoing a different model edit can restore their earlier values. Baseline/probe pinning and clearing are separate from model history. History is temporary and is not exported or saved to browser.

Implementation reference: Appearance/layout and history handlers [R2], responsive stylesheet and labels [R5].

CHAPTER 07

Saving, sharing and reproducibility

7.1 Save, share and export reference

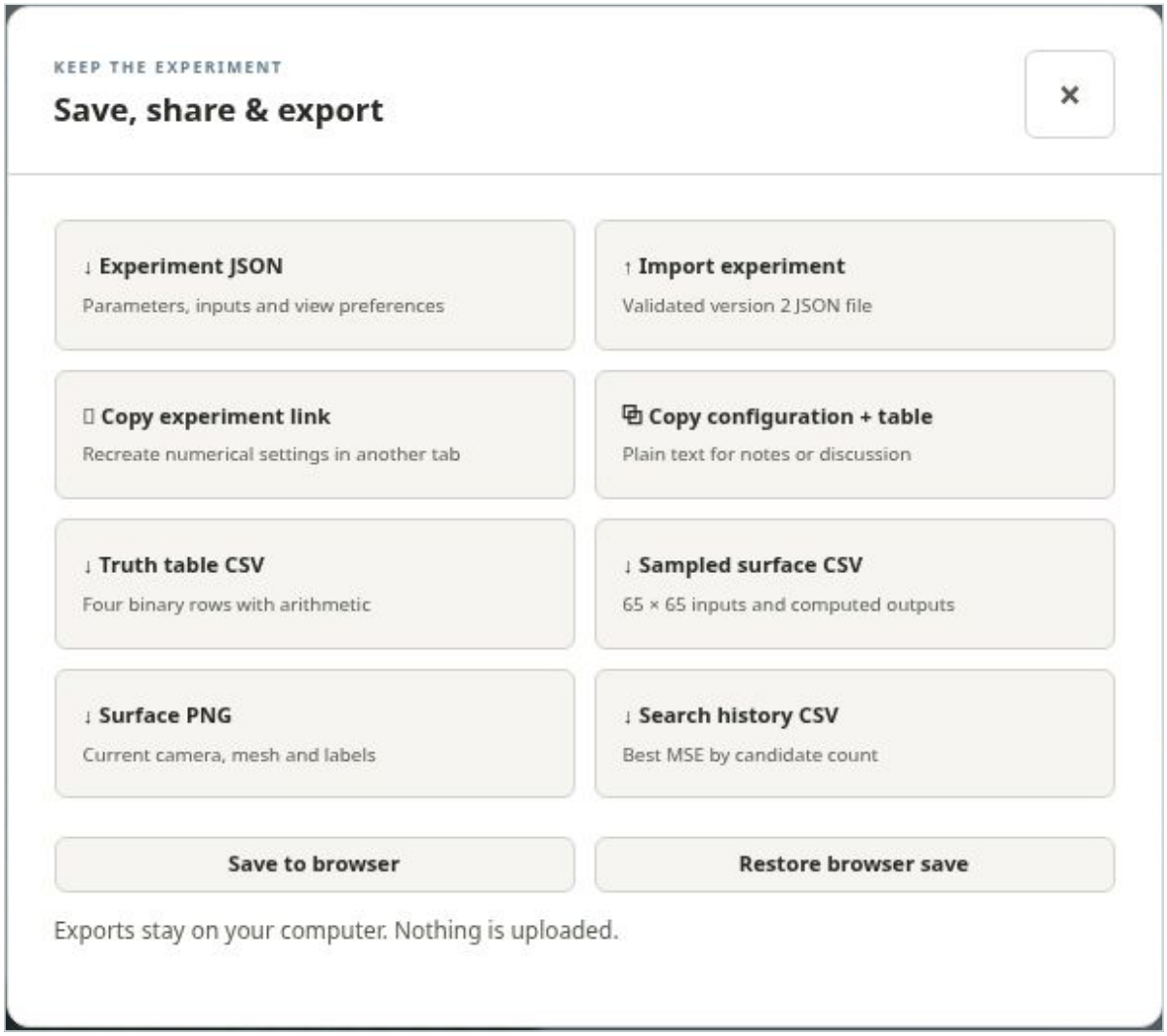


Figure 21. Save, share and export choices. File downloads, clipboard copies and browser-local storage are different ways to preserve an experiment.

Control / output	Contents and limits
Experiment JSON → igl-experiment-2.0.1.json	Node and inputs, target, rescale flag, view options, appearance/layout, current search settings, format/version and creation timestamp.
Import experiment	Reads a local version-2 JSON file, up to 128 KiB (131,072 bytes). Validates before applying; rejected files preserve the experiment. Nothing is uploaded.
Copy experiment link	Copies a #exp= URL fragment containing version and node fields only. Maximum accepted fragment length: 8,192 characters. Does not include view, appearance, camera or search controls.
Copy configuration + table	Plain text: version, target, parameters, inputs, arithmetic/activation formulas, target MSE and four truth rows. Does not include appearance or camera.

Control / output	Contents and limits
Truth table CSV → igl-truth-table.csv	Four rows. Columns: X1, X2, P, mode, W1, W2, B, dot, S, Y, normalized, region, target. Target is the expected Boolean value for that row.
Sampled surface CSV → igl-surface-65x65.csv	4,225 endpoint-grid rows. Columns: requested_X1, requested_X2, evaluated_X1, evaluated_X2, P, mode, W1, W2, B, S, Y, normalized, region. INT can map multiple requested inputs to the same effective coordinates.
Surface PNG → igl-surface-2.0.1.png	Current scene/camera and labels, plus version, W_1/W_2/B/P, arithmetic mode, effective live inputs, Y/P, mesh dimensions and an INT interpolation note. Not a full-page screenshot or editable experiment.
Search history CSV → igl-search-history.csv	candidate_evaluations,best_mse. Run search exports its initial score and retained best improvements; Step/×10 export their initial score and each iteration outcome, including repeated losses. Long Run histories are thinned. Requires an existing run/batch history.
Save to browser	Replaces this browser/origin's saved experiment with the same information as JSON.
Restore browser save	Validates and restores that saved experiment; does not resume a search.
Copy manually	Appears if clipboard access fails. The read-only Text to copy field is selected for manual copying; close with ×.

JSON and browser saves omit camera orientation/pan/zoom, surface probe pin, baseline, Undo/Redo, active tabs, Focus view/fullscreen, running-worker state, Restore start and all search history. Save history CSV separately before changing the model, because model edits clear it. JSON imported during training stops the run. Saving during training records the currently received node/settings, not a resumable checkpoint.

Appearance is automatically remembered, but ordinary model edits are not autosaved. Reload starts the default model unless an experiment fragment is present or you choose Restore browser save. Browser-profile, site-data or storage changes can remove saves; JSON is the portable copy. Importing appearance settings also saves those preferences locally. Download requested means the browser was asked to save; destination and completion remain browser-controlled.

Before Experiment JSON or Save to browser, an invalid or blank Random seed is reset to 2026 with a notice. This keeps the saved search settings usable.

Run history is limited to at most 512 retained records by repeated thinning: when a new improvement creates 513 records, the first, last and even-indexed records are kept, leaving 257. The graph can extend the last loss to the final candidate count without adding a CSV row. Thus a plateau, budget limit or Stop can leave the CSV's last recorded improvement earlier than the final Candidates readout; Step/×10 may instead include an unchanged final iteration result.

7.2 Experiment JSON fields and compatibility

Field or group	Exported form / accepted values
format; version; createdAt	format = mliglon-igl-lab; version = 2.0.1; createdAt is an ISO timestamp. Import requires this format and a version string with major 2; the timestamp is informational.
node	Required object. p is a listed power of two; mode is real or int; w1/w2/b are finite numbers within ±3p; x1/x2 are finite 0…1; target is a full Boolean gate name; preset is a recognized preset or empty; rescale is exported as a boolean; a missing or nonboolean imported rescale defaults to true.
Missing node fields	Defaults: p 16, mode real, w1=p, w2=p, b=0, x1=x2=0.5, target XOR, preset empty, rescale true. INT rounds parameters and snaps effective inputs.
view	projection: orthographic/perspective; palette: spectrum/ocean/thermal/terrain/graphite/gold; resolution: numeric 32/64/96/128/192; slice: none/x1/x2/both; boolean axes, labels, grid, wireframe, lighting, marker, autorotate, compare.
appearance	theme/font/size/layout use the values in the appearance tables; reducedMotion is boolean. Optional finite leftWidth/rightWidth/analysisHeight are clamped to 260…460 / 280…480 / 180…520 before viewport adaptation.
search	algorithm: local or multiscale; integer seed 0…4,294,967,295; integer budget 1…100,000; integer maxRestarts 0…128. Defaults within a supplied partial group: multiscale, 2026, 12000, 32.

Full target names, also used by the Target function selector, are FALSE, AND, A AND NOT B, A (X_1), NOT A AND B, B (X_2), XOR, OR, NOR, XNOR, NOT B, A OR NOT B, NOT A, NOT A OR B, NAND and TRUE. The preset grid uses compact symbolic labels for several of these.

Version-2.0.0 exports are compatible with 2.0.1; the old WordPress plugin archive is not an experiment file. Numeric strings are not substitutes for numbers. Unknown fields are ignored, and unknown preset labels are discarded. An omitted optional view, appearance or search group leaves that group unchanged; a supplied partial view changes its recognized fields, while a supplied appearance group fills missing or unrecognized choices from defaults and clamps layout sizes. Malformed JSON, invalid model/search/view values and oversized files reject the import without partially applying it.

7.3 A practical reproducibility record

For a technical note or demonstration handout, keep the Experiment JSON together with the truth-table CSV. Add a PNG to preserve the particular view, and export Search history CSV when the optimization path matters. These outputs complement one another: none is a complete recording of an interactive session.

Record	Why retain it
Release and browser	Record 2.0.1, the browser, and whether the display reports WebGL or fallback. Rendering appearance can differ while numerical evaluation agrees.
Numerical configuration	Preserve P, mode, W1, W2, B, live inputs and the selected training target. State whether inputs were requested coordinates or already-snapped values.
Search configuration	Preserve the starting model, algorithm, seed, candidate budget and restart limit. The final model alone does not identify the start.
Search outcome	Record status, candidates, iterations, restarts and normalized MSE. Compare methods using the same start and stopping limits.
View configuration	JSON retains the named display options; it does not retain camera rotation, pan or zoom. PNG documents the visual viewpoint.
Interpretation	Say whether a claim concerns four exact Boolean corners, a sampled interior diagnostic, or a rendered mesh. These are different tests.

A copied experiment link places numerical settings in the URL fragment. Share the copied link deliberately if you want another person to open it; the Copy command does not send a message. Browser saves belong to the current browser and origin and are not a substitute for a downloaded portable record.

References: released controller export/import handlers [R2], the bundled user guide [R6], and browser-local storage semantics [R9].

CHAPTER 08

Three reproducible walkthroughs

8.1 Walkthrough 1 - explain XOR with one node

1 **Set the experiment**
Select Real, P=16, then XOR. Confirm W_1=16, W_2=16, B=0. Selecting a preset does not automatically change the training target; set target XOR if you want its displayed MSE to be zero.

2 **Inspect the corners**
Select each truth-table row. S is [0,16,16,32]; raw Y is [0,16,16,0]; normalized output is [0,1,1,0]. The 11 sum reaches the descending segment's zero endpoint.

3 **Probe interior inputs**
Enter X_1=0.25, X_2=0.5: S=12 and Y/P=0.75. Then use 0.75,0.75: S=24 and Y/P=0.5. Follow the live marker on Activation and a cross-section.

4 **Interpret**
An exact Boolean corner table can coexist with graded interior responses. The falling activation region explains this behavior; camera or palette changes cannot change the arithmetic.

8.2 Walkthrough 2 - compare P4 and P1024

1 **Establish the Real reference**
Choose Real, P=4 and the Asymmetric graded preset. W_1=4, W_2=2, B=0. Enter requested inputs 0.38 and 0.66. S=2.84 and Y/P=0.71.

2 **Use INT at P4**
Switch to INT. Inputs snap to 0.5 and 0.75: scaled integers 2 and 3. The dot product is 14; round(14/4)=4, so Y/P=1. The difference from the original requested-point reference is 0.29.

3 **Increase precision correctly**
Enable Preserve ratios, choose P=1024, and re-enter 0.38 and 0.66. Re-entry matters: changing P alone preserves previously snapped live inputs. Parameters become 1024,512,0; scaled inputs are 389,676; dot=744448; S=727; Y/P=0.7099609375.

4 **Read the comparison**
The requested-point difference is now 0.0000390625. Effective inputs are 0.3798828125 and 0.66015625. Precision map separately compares both arithmetic modes over 65×65 midpoints. Its color scale is relative to its sampled maximum; inspect the numerical statistics when comparing scales.

This example demonstrates quantization of one response. It does not establish classification accuracy or guarantee that every model/input improves monotonically with P. The Real reference for the same already-snapped live input is a different comparison from the original requested input.

8.3 Walkthrough 3 - local plateau versus multiscale search

1. **Create an identical start**
   Set P=1024, INT, W_1=1536, W_2=1536, B=-1536; target TRUE; seed 2026; candidate budget 100,000. Initial outputs are [0,0,0,0.5], so MSE=0.8125. Keep the tab visible.

2. **Run local search**
   Choose Local coordinate search and Maximum restarts=0. Run search completes 33,768 candidates and 5,628 iterations, reaching a plateau at W_1=1, W_2=2, B=1022. MSE=0.000001430511474609375. This plateau is expected, not a software failure.

3. **Restore and change the method**
   Click Restore start; verify 1536,1536,-1536. Choose Multiscale + seeded restarts, Maximum restarts=32, retaining seed 2026 and budget 100,000. Run again.

4. **Check the reproducible result**
   Expect 816 candidates, 36 completed polls and 3 restarts; W_1=1024, W_2=0, B=3072; MSE=0. All four outputs are 1. This solution differs from the TRUE preset, demonstrating that search did not simply load that preset.

5. **Export the evidence**
   Search history has five (candidate,MSE) points: (0,0.8125), (6,0.5625), (14,0.3125), (25,0.25), (816,0). Export JSON, truth CSV and history before editing. These results were independently replayed and match the live 2.0.1 retest.

The comparison is specific to this start, target and version. Candidate counts describe search work; elapsed time depends on the browser and machine and is not an IGL-versus-backpropagation benchmark.

Implementation reference: Released numerical/search core [R1] and live 2.0.1 control execution; all stated numerical outcomes were reproduced.

CHAPTER 09

Troubleshooting and operational checks

9.1 Practical troubleshooting

Symptom	What to do
Fullscreen is unavailable or blocked	Use Focus view, maximize the window, or use the browser's own fullscreen command. Fullscreen is optional.
2D FALLBACK appears	Numerical controls and training remain usable. The projection uses Canvas2D, with effective mesh capped at 48; Surface shading has no effect. Reload after addressing browser/WebGL availability if 3D acceleration is required.
WebGL context lost	The application temporarily switches to Canvas2D and attempts restoration. Save JSON if you need a portable copy before reloading.
Auto orbit cannot be selected	Read the explanation in View options. Open Appearance and explicitly turn off Reduce motion, then enable Auto orbit. Hidden/offscreen scenes and active dragging pause rotation.
INT surface has sloping or coarse faces	Triangles connect discrete samples. Raise Mesh resolution for a finer picture; use numerical probes and Precision map for exact values. Increasing P alone does not increase mesh density.
Preset name and target differ	This is intentional: a preset chooses starting parameters; Target function chooses what training should fit.
Number field seems stale after training	Finish or cancel the typed edit, then move focus away; 2.0.1 refreshes the current value on blur.
Saved model did not return on reload	Choose Restore browser save. Only appearance loads automatically. Use JSON if storage is unavailable, cleared, private or on a different browser/device.
Clipboard is blocked	Use the Copy manually dialog and the browser/operating-system copy command.
A download is not visible	Check browser Downloads, download permissions and the chosen destination. A request toast is not proof of completion. PNG captures the scene only; use JSON to preserve the experiment independently.
Import is rejected	Use an actual version-2 experiment JSON, ≤128 KiB, with numeric values inside supported bounds. Correct the reported field; the current experiment is retained.
Search stopped after changing tabs	Page hiding intentionally stops training. Return and choose Run search for a new run; Restore start recovers the previous start when available.
A chosen font looks unchanged	An unavailable installed font falls back. Choose a bundled Noto option for a predictable local family.
Controls are cramped	Use a different layout/text size, resize dividers, enlarge the window, or scroll the stacked narrow-screen layout. Focus view temporarily gives the scene more room.

9.2 Quick numerical and interface checks

Check	Expected result
XOR at any supported P	Canonical preset corner outputs [0, 1, 1, 0], ordered 00, 01, 10, 11.
Default Real interior probe	P16, XOR, inputs 0.7 and 0.25: S = Y = 15.2 and normalized output 0.95.
Asymmetric P4 INT example	Enter requested 0.38 and 0.66: stored 0.5 and 0.75; D = 14, S = 4, Y/P = 1.
Asymmetric P1024 INT example	Re-enter the same requests: scaled inputs 389 and 676; D = 744448, S = 727, Y/P = 0.7099609375.
Reduce motion	Auto orbit is disabled with a visible explanation and an Open Appearance shortcut.
Focused field after training	Leaving a numeric field refreshes it to the current model after any asynchronous change.
Downloaded truth table	One header plus four binary-example data rows.
Downloaded sampled surface	One header plus 4,225 endpoint-grid data rows, independently of the scene's mesh resolution.

A plateau in local search is an expected possible outcome. A sampled precision maximum is not a proof of the global worst case. A successful demonstration is also not a hardware benchmark or a claim of learned generalization beyond the four training examples.

9.3 Launching and linking a hosted installation

Open the installed HTTPS URL directly or launch it from a WordPress link that opens a new tab. The laboratory is a static HTML/CSS/JavaScript application with bundled fonts. Its Web Worker and assets are loaded from the same installation; no WordPress plugin participates in the node calculation.

For a WordPress Custom HTML block, the essential link is: Open IGL Laboratory. The application also runs when that URL is entered directly in the address bar.

For a separate server installation, preserve the directory structure supplied with 2.0.1, including index.html and assets/. Serve it through a web server and verify the browser receives the versioned JavaScript, stylesheet and font files. Opening index.html directly from a local file URL is not the documented hosting path; worker and storage behavior can differ under local-file origins.

Use the deployment package's INSTALL.md for server setup, WordPress linking and file-transfer instructions. This manual describes program operation; web-server ownership, file permissions, TLS and WordPress administration remain host responsibilities.

References: installation instructions [R7] and runtime links in index.html [R5].

CHAPTER 10

Glossary and reference

10.1 Technical glossary

Term	Definition	Significance in this laboratory
IGL	Integer Gate Logic.	This laboratory illustrates one node, not the complete training system.
Node	One parameterized two-input response function.	It is the only trainable unit here.
Parameter	A value defining model behavior.	Search changes W_1, W_2 and B only.
Weight	An input's multiplicative coefficient.	W_1 and W_2 control how each input moves S.
Bias	An additive offset B.	Shifts sums between activation regions without moving inputs.
Scale P	Positive activation and integer scaling factor.	Controls breakpoints and INT input/output levels, not mesh density.
Normalized input	An input coordinate between zero and one.	Live controls and plots use this domain.
Scaled input	Integer round(Px) in INT mode.	The trace reveals the effective quantized input.
Dot product D	Sum of the two weighted input products.	INT divides D by P before adding bias.
Activation sum S	Value supplied to the activation function.	Its interval selects the applicable region.
Activation	Piecewise map from S to raw Y.	The five-region response can rise, fall and rise again.
Activation region	One of five intervals with a defined formula.	Region labels connect trace, chart and exported rows.
Normalized output	Raw Y divided by P.	Makes outputs and errors comparable across scales.
Quantization	Mapping continuous values onto discrete levels.	INT rounds inputs, parameters and the scaled sum.
Rounding convention	JavaScript nearest-integer rounding; half ties toward positive infinity.	Negative ties must match when reproducing the arithmetic.
Saturation	Constant output beyond an activation boundary.	The zero and upper plateau can produce flat neighborhoods.
Boolean function	Mapping binary inputs to binary output.	Sixteen distinct two-input targets are available.
Truth table	Outputs at 00,01,10,11.	Supplies all four training examples and corner diagnostics.
Graded response	An intermediate normalized output.	Interior responses need not be Boolean.

Term	Definition	Significance in this laboratory
Exact gate	All four corners match a Boolean table within tolerance.	A classification of corners, not every interior input.
Nearest gate	Boolean table with least mean absolute corner error.	Describes similarity, not probability or confidence.
MAE	Mean absolute error.	Used for nearest-gate comparison; differs from training MSE.
MSE	Mean squared error over four normalized predictions.	The search objective penalizes larger deviations more strongly.
Target	Desired four-bit Boolean output vector.	Independent of the currently loaded preset.
Candidate	A proposed parameter triple.	One evaluation scores all four examples.
Coordinate search	Moving one parameter at a time.	Local search has up to six neighbors per poll.
Poll	Evaluation of a neighborhood around the exploratory point.	Completed polls increment the iteration counter.
Multiscale search	Neighbor search with successively smaller steps.	Combines joint directions with seeded restarts.
Step δ	Magnitude of a proposed parameter change.	Controls the neighborhood's scale; distinct from the Step button.
Objective plateau	No accepted improvement in the tested neighborhood.	Does not rule out a better distant solution.
Restart	Beginning exploration from another lattice point.	Consumes budget while retaining the global best result.
Coarse lattice	343 parameter triples with coefficients -3 through 3 times P.	Provides target-independent restart locations.
Seed	Integer initializing the reproducible pseudorandom sequence.	Controls random starts and restart ordering.
Greedy acceptance	Accepting only a sufficiently lower objective.	Equal scores retain the earlier candidate.
Best-so-far	Lowest-error model encountered in the run.	Displayed even when exploration visits worse restart points.
Candidate budget	Maximum number of scored proposals.	Bounds run work, independently of rendering resolution.
Iteration	A completed neighborhood poll.	Its count is not the number of candidates.
Solved tolerance	MSE threshold for Target matched.	Separate from the per-corner Exact gate tolerance.
Web Worker	Background JavaScript execution context.	Search can run without monopolizing the interface thread.
Time-sliced fallback	Short main-thread batches separated by scheduled yields.	Provides bounded search when workers cannot run.

Term	Definition	Significance in this laboratory
WebGL	Browser graphics API using a rendering context.	Draws the 3D view; does not score training candidates.
GPU	Graphics processing unit.	May execute WebGL rendering; performance is not measured here.
Mesh resolution	Number of subdivisions along each plotted input axis.	Changes visualization detail and rendering cost, not P.
Rasterization	Converting projected geometry into pixels.	Rendered appearance remains separate from exact core evaluation.
Wireframe	Visible edges of the sampled surface.	Helps inspect mesh geometry and sampling density.
Orthographic projection	Projection without perspective size reduction.	Supports direct visual comparison of spatial dimensions.
Perspective projection	Projection making distant geometry appear smaller.	Changes spatial perception without changing the model.
Orbit	Camera rotation around the scene.	Changes viewpoint; Auto orbit makes it continuous.
Cross-section	One-dimensional cut through the response field.	Fixes one live input while the other varies.
Precision map	Sampled absolute INT-versus-Real response difference.	Reports sampled error; its maximum is not a global bound.
Baseline	Pinned numerical model snapshot.	Supports comparison and restoration without training a second node.
Undo/redo	Navigation through model-edit history.	Distinct from restoring the initial state of a search.
JSON	Structured text representation of an experiment.	Saves validated model, view, appearance and search settings.
CSV	Delimited table of numerical results.	Exports truth rows, sampled surfaces or search history.
PNG	Raster image format.	Exports the current rendered view and its metadata.
URL fragment	State encoded after the # in an experiment link.	Recreates numerical settings without storing a server-side experiment.
Local storage	Browser storage associated with the site origin.	Holds preferences and explicitly saved experiments when permitted.
Metadata	Descriptive information accompanying data or an image.	Identifies version, parameters and export context.
ARIA	Accessibility semantics attached to controls.	Labels, states and divider values assist nonvisual interaction.
Focus view	Expanded scene within the existing browser viewport.	Remains available when native fullscreen is refused.

Term	Definition	Significance in this laboratory
Fullscreen	Browser-controlled display mode beyond ordinary page layout.	Requires browser acceptance and differs from Focus view.
Reduced motion	Preference limiting optional animation.	Disables Auto orbit until explicitly changed.

10.2 Acronym and notation legend

Abbreviation	Expansion
IGL	Integer Gate Logic
INT	Integer arithmetic mode
MAE	Mean absolute error
MSE	Mean squared error
CPU	Central processing unit
GPU	Graphics processing unit
UI	User interface
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
DOM	Document Object Model
WebGL	Web Graphics Library
JSON	JavaScript Object Notation
CSV	Comma-separated values
PNG	Portable Network Graphics
ARIA	Accessible Rich Internet Applications
DPR	Device pixel ratio
W_1, W_2; B; P	Weights; bias; activation/integer scale
D; S; Y; Y/P	Weighted products; activation input; raw and normalized output
δ; θ; t_i; $\hat{y}_i$	Search step; parameter triple; target and predicted row output

10.3 Implementation and browser references

Program-specific statements in this manual describe the released 2.0.1 implementation. The listed program files were matched to the live installation in the accompanying release retest. Browser documentation supplies the platform semantics, rather than the application's model-specific rules. Accessed 11 November 2026.

Reference	Primary source and use
[R1] Numerical core	assets/core.js?v=2.0.1 - activation, region, evaluate, preset definitions, exact/nearest gates, MSE and createSearch. Open release source
[R2] Application controller	assets/app.js?v=2.0.1 - model normalization, quantizeInputs, changeScale, view/appearance state, diagnostics, training coordination and exports. Open release source
[R3] Renderer	assets/renderer.js?v=2.0.1 - cameras, surface mesh, picking, reference overlay, fallback and PNG snapshot. Open release source
[R4] Search worker	assets/training-worker.js?v=2.0.1 - bounded worker chunks, progress messages and cancellation. Open release source
[R5] Interface and stylesheet	index.html and assets/styles.css?v=2.0.1 - exact labels, selectors, fonts, themes, dialogs and responsive layout. Open release source
[R6] Bundled release guide	docs/USER-GUIDE.md - release documentation. Open release source
[R7] Package administration documents	INSTALL.md and docs/MAINTENANCE-2.0.1.md in the 2.0.1 distribution - installation and the maintenance changes.
[R8] JavaScript rounding	MDN Math.round(): ties toward positive infinity and the negative-zero detail. Open MDN documentation
[R9] Browser persistence	MDN Window.localStorage: origin-specific persistence and browser policy behavior. Open MDN documentation
[R10] Worker execution	MDN Using Web Workers: background JavaScript execution and message communication. Open MDN documentation

Illustrations and screenshots: genuine interface captures from release 2.0.1, plus separately labeled computed activation, quantization and surface diagrams. The mathematical illustrations were compared with the released numerical core. Version-specific implementation hashes and the detailed test record are contained in the separate Installation and Security Retest report.

Edition: IGL Laboratory 2.0.1 Technical User Guide, 11 November 2026. The manual documents the released browser demonstrator and its three-parameter, four-example training scope.