COMPOUNDING ENERGY LTD

Working Paper Series • CE-WP-2026-04

Pruned ReLU Surrogates

Embedding non-linear physics into MILP-based energy system optimisation with explicit error bounds

Dr Christopher T. M. Clack
Founder, Compounding Energy Ltd
christopher@compoundingenergy.com

September 2026

Abstract. Investment-grade energy system optimisation must routinely embed non-linear physical models — aerodynamic wake interactions in wind farms, hosting-capacity constraints in distribution feeders, conversion-efficiency surfaces in process equipment — into the same mixed-integer linear programmes (MILPs) used for least-cost dispatch and capacity expansion. The classical solution is to linearise around an operating point, which is wrong by 10–30% across the relevant operating envelope, or to call the non-linear simulator at every iteration of the optimiser, which is intractable. We present a unified framework based on rectified-linear-unit (ReLU) neural-network surrogates, embedded into the MILP via the standard big-M formulation, with two methodological refinements: (i) interval-bound propagation (IBP) tightens the per-neuron big-M values — by orders of magnitude relative to an uninformed large constant, though that factor scales with the assumed constant, and at the first hidden layer IBP coincides exactly with a one-line weight-norm bound — and (ii) provably always-active and always-inactive neurons are

pruned from the MILP entirely, replaced by their linear or constant equivalents; the pruning is what the propagated bounds genuinely buy. We give explicit pruning rules, a generalisation-error decomposition closed by an explicitly measured worst-case (adversarial-search) error estimate, and a worked case study on a 3×3 offshore wind farm with Jensen wake interaction — extended to production-scale surrogates, where, with solver presolve disabled, the tightened encoding solves three-and-a-half times faster in 5.6× fewer nodes (solver presolve reconstructs comparable bounds when enabled). The true optimum of the case-study instance is known (zero curtailment), so the gap between the surrogate optimum and the truth is measured, not assumed. This paper is a method study, not a description of a production system.

Keywords: neural network surrogate, mixed-integer programming, ReLU embedding, wake modelling, hosting capacity, capacity expansion, surrogate optimisation

Contents

1 Introduction

Investment-grade energy system optimisation routinely encounters non-linear physics that cannot be expressed in the linear arithmetic of LP/MILP solvers. Three examples drive the framework presented here. First, the aerodynamic interaction between turbines in a wind farm is governed by wake dynamics that are quadratic-or-worse in turbine position, freestream wind, and individual turbine actions; ignoring this in capacity expansion leads to fleet revenue forecasts that overstate output by 10–20% (Stevens et al., 2016; Niayifar and Porté-Agel, 2016). Second, distribution-network hosting capacity — the maximum amount of distributed PV or BESS that a feeder can absorb without violating voltage or thermal limits — is the result of a non-convex AC optimal power flow that has no closed-form expression but is critical to integrated DER planning (Dubey and Santoso, 2017). Third, novel-fuel process-economic models (electrolysis, ammonia synthesis, Fischer–Tropsch) embed conversion-efficiency surfaces from Aspen Plus simulators into capacity-investment decisions (Henao and Maravelias, 2011; Caballero and Grossmann, 2008). Each of these problems has the same structural shape: a fast, accurate, non-linear function $f(\boldsymbol{u}, \boldsymbol{v})$ of design and operational variables that the optimiser needs to evaluate *from inside the optimisation loop*, not as a post-hoc correction.

Three classical approaches each fail. *Linearisation around an operating point* is wrong by 10–30% across the relevant envelope and cannot be made tighter without piecewise extension that explodes in the number of pieces. *Iterating between simulator and optimiser* (calling Aspen, OpenDSS, or a CFD solver from inside the LP loop) blocks production deployment because each iteration costs minutes or hours and the outer loop has no convergence guarantee. *Direct piecewise-linear approximation* of the non-linear function via SOS-2 constraints is tractable for one or two design dimensions but combinatorially explodes in higher dimensions (Boukouvala et al., 2016).

The approach we present is based on rectified-linear-unit (ReLU) neural-network surrogates. A small ReLU MLP can approximate any continuous function on a compact domain to arbitrary accuracy and admits a closed-form mixed-integer linear representation via the big-M formulation (Tjeng et al., 2019; Anderson et al., 2020). This embedding has been recognised in the chemical-engineering optimisation literature (Grimstad and Andersson, 2019; Ceccon et al., 2022) as the right structural substrate for surrogate-based MILP optimisation. Our contribution is to refine the embedding for energy-system applications with three additions: (i) explicit interval-bound propagation to tighten per-neuron big-M values, (ii) provable pruning of always-active and always-inactive neurons, and (iii) an applied generalisation-error decomposition that separates surrogate error, optimisation error, and physical model error. The framework reproduces the Lastrucci 2026 pruning structure (Lastrucci et al., 2026) with the additions specialised for energy-system MILPs.

Practitioner sidebar

What this enables operationally. Today, a CEM tool that wants to optimise a 5 GW offshore wind portfolio under wake uncertainty either runs CFD post-hoc (slow and disconnected from the investment optimisation) or applies a flat capacity-factor reduction (wrong by 10%+). The framework here lets the optimiser evaluate the wake physics from inside the MILP, with the surrogate's worst-case error documented — and, crucially, with the true function evaluated at every surrogate optimum, so that surrogate error converted into optimisation error is measured rather than hidden. The result is reproducible, auditable, and falsifiable.

2 The framework

2.1 Setup

Let $f : \mathcal{X} \to \mathbb{R}$ be a continuous function we wish to embed in an MILP, where $\mathcal{X} = [\underline{x}_1, \overline{x}_1] \times \cdots \times [\underline{x}_n, \overline{x}_n] \subset \mathbb{R}^n$ is a closed box. We assume f is given as an oracle: any $f(\boldsymbol{x})$ can be evaluated, but no closed form is available. The optimiser accesses f only through the MILP relaxation; runtime queries to the simulator are forbidden. (The case study of Section 3 uses a closed-form f^* deliberately, so that the surrogate's error — and the true value of every surrogate optimum — can be measured exactly against ground truth.)

We approximate f by a ReLU MLP $\hat{f}$ with L hidden layers and architecture $(n, h_1, h_2, \ldots, h_L, 1)$, where h_ℓ is the width of layer ℓ. The network's forward pass is

$$\begin{aligned} \boldsymbol{z}_1 &= \boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1, \qquad & \boldsymbol{a}_1 &= \mathrm{ReLU}(\boldsymbol{z}_1), \\ \boldsymbol{z}_\ell &= \boldsymbol{W}_\ell \boldsymbol{a}_{\ell-1} + \boldsymbol{b}_\ell, \qquad & \boldsymbol{a}_\ell &= \mathrm{ReLU}(\boldsymbol{z}_\ell), \quad \ell = 2, \ldots, L, \\ \hat{f}(\boldsymbol{x}) &= \boldsymbol{w}_{\text{out}}^\top \boldsymbol{a}_L + b_{\text{out}}. \end{aligned} \tag{1}$$

Training $\hat{f}$ to fit a sample of N pairs $(\boldsymbol{x}^{(i)}, f(\boldsymbol{x}^{(i)}))$ is a non-convex optimisation problem, addressed in practice by SGD with weight initialisation that approximately preserves activation variance (He initialisation). The training problem is not the focus of this paper; we assume $\hat{f}$ has been trained and proceed to embed it into an MILP.

2.2 The big-M ReLU formulation

For each ReLU node $a = \mathrm{ReLU}(z) = \max(z, 0)$ we introduce a binary variable $\delta \in \{0, 1\}$ indicating whether the unit is active and write the standard big-M encoding (Tjeng et al., 2019; Fischetti and Jo, 2018; Anderson et al., 2020):

$$\begin{aligned} & a \geq z, \\ & a \geq 0, \\ & a \leq z + M(1 - \delta), \\ & a \leq M\delta, \end{aligned} \tag{2}$$

where M is a constant strictly bounding $|z|$ on the input domain $\mathcal{X}$ propagated through layers $1, \ldots, \ell - 1$. The four inequalities together force $a = \max(z, 0)$ at any feasible integer solution. The MILP encoding of $\hat{f}$ on $\mathcal{X}$ comprises the affine layer constraints (1), the big-M ReLU constraints (2) for every ReLU node, and the input-domain box constraints.

2.3 Why the choice of M matters

The big-M formulation is exact at integer feasibility but its LP relaxation can be loose if M is much larger than the tight per-neuron bound on $|z|$. Loose relaxations weaken cuts, slow branch-and-bound, and in extreme cases produce relaxation gaps that are uninformative (Anderson et al., 2020). A naive choice $M = 10^6$ or $M =$ "some large number" is the dominant cause of intractability when ReLU MLPs are embedded without further care.

The remedy is to compute, for each neuron, a tight pre-activation interval $[l_z, u_z]$ such that $z \in [l_z, u_z]$ for any input $\boldsymbol{x} \in \mathcal{X}$ and any feasible activations of preceding layers. The minimum valid M for that neuron is then $M = \max(|l_z|, |u_z|)$, frequently 1–3 orders of magnitude smaller than a naive choice.

2.4 Interval-bound propagation (IBP)

We compute layer-by-layer pre-activation bounds via interval-bound propagation. Let $W_\ell^+ = \max(\boldsymbol{W}_\ell, 0)$ and $W_\ell^- = \min(\boldsymbol{W}_\ell, 0)$ be the positive and negative parts of $\boldsymbol{W}_\ell$ (componentwise). For layer 1 with input bounds $\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}$,

$$\underline{\boldsymbol{z}}_1 = W_1^+ \underline{\boldsymbol{x}} + W_1^- \overline{\boldsymbol{x}} + \boldsymbol{b}_1, \quad \overline{\boldsymbol{z}}_1 = W_1^+ \overline{\boldsymbol{x}} + W_1^- \underline{\boldsymbol{x}} + \boldsymbol{b}_1. \tag{3}$$

After ReLU, $\underline{\boldsymbol{a}}_1 = \max(\underline{\boldsymbol{z}}_1, 0)$, $\overline{\boldsymbol{a}}_1 = \max(\overline{\boldsymbol{z}}_1, 0)$ (componentwise). Iterating this through $\ell = 2, \ldots, L$ gives bounds on every $\boldsymbol{z}_\ell$, $\boldsymbol{a}_\ell$.

Lemma 2.1 (Soundness of IBP)**.** *For any $\boldsymbol{x} \in [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}]$ and any forward pass through (1),*

$$\underline{z}_{\ell,j} \le z_{\ell,j} \le \overline{z}_{\ell,j} \quad \text{and} \quad \underline{a}_{\ell,j} \le a_{\ell,j} \le \overline{a}_{\ell,j}$$

for all ℓ and j.

Proof. By induction on ℓ. Base case $\ell = 1$ follows from (3) and the elementary fact that $W^+\underline{\boldsymbol{x}} + W^-\overline{\boldsymbol{x}}$ minimises $W\boldsymbol{x}$ over the box, while $W^+\overline{\boldsymbol{x}} + W^-\underline{\boldsymbol{x}}$ maximises it. ReLU is monotone, so the post-activation bounds inherit. Inductive step: same argument applied to $\boldsymbol{W}_\ell, \boldsymbol{b}_\ell$ on layer $\ell - 1$ post-activation bounds. □

Remark 2.2 (Tightness of IBP). IBP is sound (Lemma 2.1) but not tight: the induced bounds can be looser than the actual achievable range. Tighter bound-propagation methods (CROWN, Zhang et al. 2018; CROWN-IBP, Zhang et al. 2020; β-CROWN, Wang et al. 2021) give substantially tighter bounds at higher computational cost. For the trained wake surrogate of Section 3, IBP tightens the per-neuron big-M relative to a big-M obtained by propagating an uninformative input box of half-width 10^3 by 507× in layer 1 (4,531 → 8.9) and 1,673× in layer 2 (30,875 → 18.5) — but that factor scales linearly with the assumed half-width and so is not intrinsic; it measures how badly an unguided constant can miss, not how well IBP does. Against a competently chosen weight-norm global bound there is no propagation gain at all: for a first hidden layer the two bounds are algebraically identical (both equal $|\boldsymbol{W}_1\boldsymbol{x}_c + \boldsymbol{b}_1| + |\boldsymbol{W}_1|\boldsymbol{r}$ for box centre $\boldsymbol{x}_c$ and half-width $\boldsymbol{r}$; the shipped networks agree to 10^{-15}), and at layer 2 the per-neuron difference is at most 7.2% on this network. What the IBP pass buys is the *pruning* it enables — the residual layer-1 factor of 1.95× in the case study arises entirely because the maximum is taken over the ambiguous neurons that survive pruning — not tightness a closed-form bound lacks. The framework supports plug-in replacement of IBP by tighter methods where the additional tightness matters.

2.5 Pruning always-active and always-inactive neurons

The big-M encoding (2) requires a binary variable for every ReLU node. With IBP-tight bounds in hand, two cases admit deterministic resolution and require no binary:

Proposition 2.3 (Pruning rules)**.** *Let $[\underline{z}, \overline{z}]$ be IBP-tight bounds on a ReLU pre-activation z, and let $a = \mathrm{ReLU}(z)$.*

- *If $\overline{z} \le 0$ (always inactive), the constraint $a = 0$ replaces all four inequalities of (2), eliminating the binary.*
- *If $\underline{z} \ge 0$ (always active), the constraint $a = z$ replaces all four inequalities of (2), eliminating the binary.*

The pruned MILP is exactly equivalent to the un-pruned MILP for any $\boldsymbol{x} \in \mathcal{X}$.

Proof. For the always-inactive case, $\overline{z} \le 0$ gives $z \le 0$ for every feasible $\boldsymbol{x}$, hence $a = \max(z, 0) = 0$. Substituting $a = 0$ into (2): $0 \ge z$ holds, $0 \ge 0$ is vacuous, and $0 \le M\delta$ holds for either value of δ; but $0 \le z + M(1-\delta)$ requires $M(1-\delta) \ge -z = |z|$, which forces $\delta = 0$ whenever $\overline{z} < 0$. The binary is therefore *determined* (not merely unconstrained) in the strict case; at the boundary $\overline{z} = 0$ both binary values are feasible, but $a = 0$ under either, so the elimination remains valid. Substituting leaves the single constraint $a = 0$.

For the always-active case, $\underline{z} \ge 0$ gives $z \ge 0$, hence $a = z$. Substituting $a = z$: $z \ge z$ and $z \ge 0$ hold and $z \le z + M(1-\delta)$ holds; but $z \le M\delta$ forces $\delta = 1$ whenever $\underline{z} > 0$ (since $\delta = 0$ would require $z \le 0$). The binary is again determined in the strict case (at $\underline{z} = 0$ both values are feasible, with $a = z$ under either), and substituting $\delta = 1$ leaves $a = z$. In both cases the substitution is fixed by the encoding and the binary eliminated. □

The pruned MILP carries one binary variable per ambiguous neuron and none for the always-active or always-inactive ones, which retain a single equality constraint each (two constraints saved per pruned neuron). In the trained networks reported here, 20–33% of neurons are pruned to non-binary form, materially shrinking the MILP.

2.6 Generalisation-error decomposition

The optimiser-meets-surrogate setting makes three sources of error visible, two of which enter the bound below; the third sits outside it. Let f^* denote the true non-linear function, $\hat{f}$ the trained MLP surrogate, and $\hat{\boldsymbol{x}}$ the optimal solution to the surrogate MILP. The total error in the optimisation result decomposes as

$$\underbrace{f^*(\hat{\boldsymbol{x}}) - f^*(\boldsymbol{x}^*)}_{\text{induced sub-optimality}} \;\le\; \underbrace{2 \sup_{\boldsymbol{x}\in\mathcal{X}} |f^*(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})|}_{\text{surrogate error}} \;+\; \underbrace{|\hat{f}(\hat{\boldsymbol{x}}) - \hat{f}(\hat{\boldsymbol{x}}^{\mathrm{LP}})|}_{\text{MILP-relaxation gap}}. \tag{4}$$

Equation (4) is a standard surrogate-optimisation bound (Cozad et al. 2014, Sec. 5; the constant 2 comes from the triangle inequality applied at $\boldsymbol{x}^*$ and $\hat{\boldsymbol{x}}$; it is written for minimisation — for the maximisation of Section 3 the same argument bounds $f^*(\boldsymbol{x}^*) - f^*(\hat{\boldsymbol{x}})$). The first term is a *supremum* over the domain, not an average, and the point a surrogate-maximising MILP selects is adversarial, not a random draw, so no marginal quantile can stand in for it. We therefore estimate the supremum directly. For the case-study surrogate, a random-multistart adversarial search over $\mathcal{X}$ (20,000 starts plus 4,000 annealed local steps, run for each sign of the error) gives an observed sup-error of 0.34 in CF units, against an empirical maximum of 0.30 over a dense 500,000-sample grid and a split-conformal 99.9th-percentile error of 0.23 — the quantile q such that a fresh $\boldsymbol{x} \sim \mathrm{Unif}(\mathcal{X})$ satisfies $|f^*(\boldsymbol{x}) - \hat{f}(\boldsymbol{x})| \le q$ with probability at least 0.999 (Vovk et al., 2005; Lei et al., 2018), which we report alongside as a percentile, never as a supremum. Taking the observed adversarial maximum as the operative sup estimate, the induced sub-optimality bound is $2 \times 0.34 = 0.68$ — far larger than the 0.086 validation RMSE, which is precisely why a worst-case design bound must use the supremum and why we report it openly; the realised gap at the case-study optimum is measured directly and sits comfortably inside it (Section 3). The surrogate's certified Lipschitz constant — the product of its layer spectral norms, 3.9 CF per unit ℓ_2 displacement in the normalised input box — is reported as a smoothness descriptor only and enters no bound; a deterministic covering certificate built from it is vacuous in twelve dimensions. The second term is closed by branch-and-bound's optimality gap and is a property of the MILP solver. The third source of error — the gap between f^* and the underlying physical truth — is bounded by the simulator's documented uncertainty and is independent of the surrogate framework.

Practitioner sidebar

What this means for due diligence. The decomposition (4) lets a buyer evaluate the surrogate-MILP framework's accuracy at three independent levels: (i) report a measured worst-case (adversarial-search) error estimate for $\hat{f}$, not merely its average validation MSE or a marginal percentile, (ii) report the MILP solver's optimality gap, (iii) report the underlying simulator's uncertainty. If all three are documented, the total bias of the framework is bounded explicitly. Consultancy alternatives almost never document any of the three.

3 Case study: 3×3 offshore wind farm with Jensen wake

3.1 Setup

We illustrate the framework on a stylised 3×3 offshore wind farm with 9.5 MW turbines, 178 m rotor diameter, and 7-rotor-diameter spacing. The freestream wind speed u_∞ ranges over $[4, 18]$ m/s and direction θ ranges over $[0, 360]$ degrees; the direction enters the surrogate as the pair $(\cos\theta, \sin\theta)$, giving twelve inputs in all, and IBP, pruning, and the MILP operate over the box $[-1, 1]^2$ on that pair, which strictly contains the physical unit circle (Section 3.4 returns to this). Per-turbine curtailment fractions $c_j \in [0, 0.5]$ $(j = 1, \ldots, 9)$ are design variables. In this Jensen instance curtailment is a linear power derate: it scales turbine j's own output by $(1 - c_j)$ and weakens the wake j casts downstream by the same factor, and the first effect dominates everywhere — f^* is monotone non-increasing in every c_j, so the true optimum of the farm-output maximisation is zero curtailment. We verify this exhaustively in Section 3.4 and use it deliberately: an instance whose true optimum is known turns the optimisation run into an embedding demonstration with built-in ground truth. (Wake-mitigation control that genuinely pays requires thrust-based derating physics richer than the linear derate used here; we measure that variant too, and the achievable gain is far below this surrogate's worst-case error, so no mitigation claim is made in either model.)

The wake-aware farm-level capacity factor is computed by the Jensen wake model (Jensen, 1983; Katic et al., 1986):

$$\delta_{i\to j} = (1 - \sqrt{1 - C_T}) \left(\frac{D}{2(D/2 + \alpha \cdot d_{ij})} \right)^2, \tag{5}$$

where $C_T = 0.8$ is the thrust coefficient, $D = 178$ m the rotor diameter, $\alpha = 0.075$ the standard onshore Jensen/Katic wake-decay constant (retained here for comparability with the Katic–Hojstrup convention; offshore practice is 0.04–0.05), and d_{ij} the downstream distance from upwind turbine i to downstream turbine j. Multiple-wake superposition follows the Katic–Hojstrup quadratic-sum convention, with the deficit turbine j contributes scaled by $(1 - c_j)$ and turbine i's own output multiplied by $(1 - c_i)$. Waked wind speeds map to power through a stylised curve with cut-in at 3 m/s, a cubic section to rated at 12 m/s, and flat output to cut-out at 25 m/s; f^* is the farm mean of the per-turbine normalised outputs. The function $f^* : (u_\infty, \theta, \boldsymbol{c}) \mapsto \mathrm{CF}_{\mathrm{farm}}$ is thereby fully specified, computable, and non-convex and non-linear in all twelve inputs.

3.2 Surrogate training

We sample $N = 4\,000$ uniform-random $(u_\infty, \theta, \boldsymbol{c})$ tuples from the input domain, evaluate f^* via the Jensen model at each, and train a two-hidden-layer ReLU MLP with widths $h_1 = 16, h_2 = 8$ on a 80/20 train/validation split via mini-batch SGD with learning rate 0.01 for 800 epochs. The trained network achieves training MSE 7.1×10^{-3} and validation MSE 7.4×10^{-3}, with validation $R^2 = 0.91$. Figure 1 shows the training trajectory.

The validation MSE corresponds to a per-prediction RMSE of 0.086 on a function with range $[0.02, 0.93]$, i.e. about 9.5% of the range *on average*. The worst-case (sup-norm) error is much larger — the adversarial-search sup estimate is 0.34, with the split-conformal 99.9th-percentile error at 0.23 (Section 2.6) — and it is the sup figure, not the average RMSE or a marginal percentile, that enters the design bound (4). This level of surrogate error is acceptable for demonstrating the embedding but tight for absolute-revenue forecasting; the framework documents the tradeoff transparently rather than hiding it behind the average.

3.3 Pruning report

Applying IBP from Section 2.4 on the trained network, the per-neuron status decomposes as in Table 1. Five of sixteen layer-1 neurons (31%: four always-active, one always-inactive) are deterministic and require no binary in the MILP; layer 2 is wholly ambiguous in this small

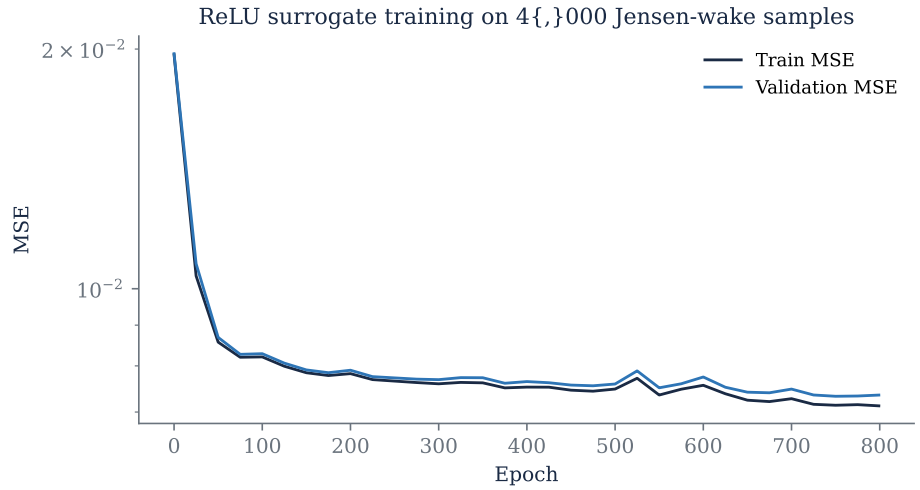

Figure 1: Training curve of the ReLU surrogate $\hat{f}$ on 4,000 Jensen-wake samples. Architecture: 12 inputs $\to$ 16 ReLU $\to$ 8 ReLU $\to$ 1 output. After 800 epochs the validation MSE is 7.4×10^{-3} and validation $R^2 = 0.91$; whether that suffices for optimisation is governed by the worst-case bound in (4), not the average.

instance. A naive *global* big-M, obtained by propagating an uninformative input box of half-width 10^3 through the affine layers, gives per-neuron values of order 4,500 in layer 1 and 31,000 in layer 2; IBP tightens these to maxima of 8.9 and 18.5 — reductions of 507× and 1,673×, factors that scale linearly with the assumed half-width and so measure the cost of an unguided constant, not an intrinsic property of IBP (Remark 2.2). The benefit is twofold: pruning eliminates 5 of the 24 binary variables (21% overall, 31% of layer 1), and the *raw* big-M relaxation tightens by those orders of magnitude. With the physical CF ≤ 1 bound in place, however, the root LP relaxation is 1.0 under either encoding — the physical constraint substitutes for tight M at the root, and the tightness benefit shows up in the search, not the root bound (Section 3.4).

Table 1: IBP-derived pruning status for the trained 16-8-1 ReLU network. "Always-active" and "always-inactive" neurons are pruned to linear or constant equivalents; "ambiguous" neurons require the full big-M binary encoding. Five of sixteen (31%) layer-1 neurons admit pruning (four always-active, one always-inactive); layer 2 has all neurons ambiguous in this small instance.

Layer	**Always active**	**Always inactive**	**Ambiguous**	**Total**
1 (16 neurons)	4	1	11	16
2 (8 neurons)	0	0	8	8
Total	4	1	19	24

3.4 MILP encoding correctness and the surrogate-optimum audit

We now run the embedded optimisation: holding the freestream at $u_\infty = 11$ m/s (near rated) and direction $\theta = 315°$ (north-westerly, fixed as data), maximise the surrogate-predicted farm capacity factor over the curtailment vector $\boldsymbol{c} \in [0, 0.5]^9$ subject to a total-curtailment budget $\sum_j c_j \leq 1.5$ (i.e. at most $\sim$ 17% average curtailment). In the true model of Section 3 this problem has a known answer — f^* is monotone non-increasing in every c_j, so the true optimum is $\boldsymbol{c} = 0$. We verify this exhaustively: over a 7-speed × 24-direction grid with 400 feasible curtailment draws per cell, plus 30,000 draws and a coordinate polish at the operating point above, the best improvement over zero curtailment is exactly 0; and in a thrust-derating (axial-

induction) model variant, where mitigation genuinely exists, the best achievable gain anywhere on the grid is +0.008 CF — forty times smaller than this surrogate's 0.34 sup error, so no surrogate at this accuracy could resolve it. Any non-zero recommendation the surrogate MILP makes is therefore surrogate error converted into optimisation error, which is precisely the mechanism the decomposition (4) prices; the instance is an embedding demonstration with built-in ground truth.

We solve the same MILP under a naive global big-M and under IBP-tight per-neuron M with pruning; the surrogate output is constrained to the physical range CF $\in [0, 1]$. Both encodings return the identical optimum $\hat{f}(\hat{\boldsymbol{x}}) = 0.615$, and the MILP optimum equals a direct forward pass of $\hat{f}$ at the optimiser's solution to machine precision (10^{-16}) — confirming both the encoding's exactness and the pruning's correctness. The audit of that optimum against the truth: the recommended $\hat{\boldsymbol{x}}$ carries $\sum_j c_j = 0.50$ of curtailment, at which $f^*(\hat{\boldsymbol{x}}) = 0.653$ against $f^*(\mathbf{0}) = 0.685$ — a realised optimisation gap of 0.032 (4.6% of the true optimum), sitting comfortably inside the $2 \times 0.34 = 0.68$ bound of Section 2.6. No surrogate optimum should ever be reported without the true function evaluated at it; the audit stage that produces these numbers is a permanent part of the harness.

On the 24-neuron instance both encodings solve in hundredths of a second. At production scale the picture is nuanced, and we report it in full (Figure 2). On the problem above at h_1=128, h_2=64 — run at the production curtailment budget of 0.8, which is slack at the optimum for both encodings — with presolve disabled, the naive encoding needs 3.6 s and 911 branch-and-bound nodes while the IBP-tightened, pruned MILP solves in 1.0 s and 164 nodes — a 3.5× wall-time and 5.6× node advantage, and under the ∼2 s budget an operational dispatch loop might allow, the difference between converging and not. With presolve enabled the advantage disappears and even inverts (0.7 s naive versus 1.5 s pruned) — HiGHS presolve reconstructs comparable bounds from the formulation. At h_1=64, h_2=32 there is no separation in any configuration. The naive arm uses the loose constant for the variable bounds as well as the big-M coefficients, since a modeller who has not computed interval bounds has neither; the comparison is between an unguided encoding and a fully bounded, pruned one. The raw root-LP relaxation (physical CF cap lifted) separates by orders of magnitude throughout — 19,765 versus 5.5 (toy), 28,007 versus 7.1 (64-32), beyond 10^6 versus 81 (128-64) — while the capped root relaxation is 1.0 for both encodings at every size. We also solved a harder variant with u_∞ and θ free over the domain box at a tightened budget of 0.8: at 128-64 both encodings hit a 10 s limit, and at the smaller sizes both find the optimum 1.0 — which is genuinely attainable, since at zero curtailment and winds above ∼14 m/s every waked turbine still reaches rated power. Because the box $[-1, 1]^2$ on $(\cos\theta, \sin\theta)$ admits non-physical "directions" of norm up to $\sqrt{2}$ (and the free-regime optimiser does exploit the corners), the harness also solves the free regime with a 16-gon tangent-polygon outer relaxation of the unit circle, which confines the direction variables to within 2% of the physical set and leaves the optimum unchanged. The honest summary: the tightened encoding's advantage appears when presolve is unavailable or disabled and the network is large; it is not a blanket speed-up (Anderson et al., 2020; Tjeng et al., 2019).

4 Application domains and status

The same embedding pattern applies wherever a fast non-linear physical response sits inside an investment or dispatch MILP: wake-aware capacity-factor surfaces in offshore-wind portfolio expansion, feeder hosting-capacity limits in distribution-level DER planning (Dubey and Santoso, 2017), and process-efficiency surfaces (electrolysis, ammonia synthesis, Fischer–Tropsch) in novel-fuel pathway models — the chemical-engineering literature surveys the last of these (Henao and Maravelias, 2011; Caballero and Grossmann, 2008; Cozad et al., 2014). In each case the surrogate is trained offline against the domain simulator, embedded via Section 2, and the error decomposition (4) is what makes the resulting optimum auditable. For high-dimensional

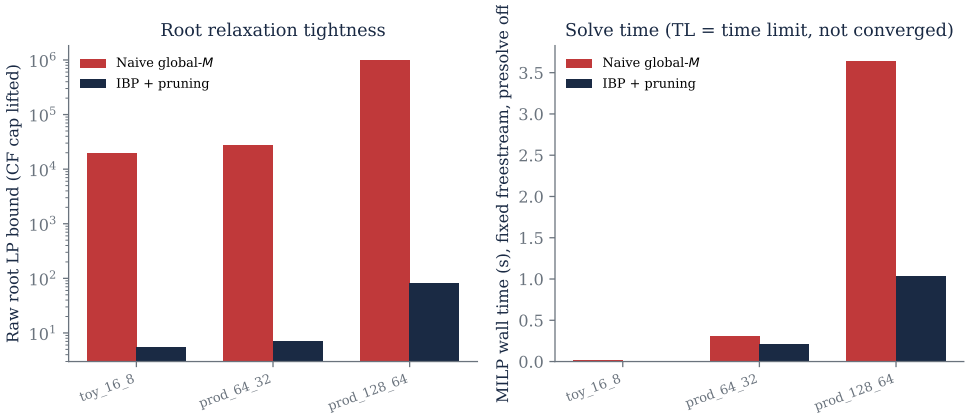

Figure 2: Raw root LP-relaxation tightness (physical CF cap lifted) and MILP solve time (fixed-freestream problem, presolve off), naive global big-M versus IBP-tightened-plus-pruned encoding, across surrogate sizes (16-8, 64-32, 128-64). The tightened encoding brings no advantage at the two smaller sizes and separates at 128-64 (3.5× wall time, 5.6× nodes); with presolve enabled the separation disappears and even inverts.

surrogate input spaces we expect naive big-M to be prohibitive and the pruning correspondingly more valuable, though we have not measured that regime here.

A status note, for the avoidance of doubt: this paper is a method study, and none of these embeddings is in production at Compounding Energy today. The company's production wind forecasting uses an analytic wake model with per-farm statistical calibration, which the GB scorecard grades end-to-end; the framework presented here is the roadmap route to wake-endogenous investment optimisation, not a description of a shipped system.

5 Discussion and limitations

5.1 When the framework fails

The framework requires that f^* be approximable by a small ReLU MLP within tolerance ε, and that IBP-tight bounds be sufficiently small that the MILP relaxation is informative. Three failure modes arise.

Discontinuous f^*. Wake models with sharp regime transitions (e.g. when a downwind turbine moves from inside to outside an upwind wake) create discontinuities that ReLU networks struggle with. The framework still works but requires deeper networks ($L \geq 4$) and benefits from CROWN-style tighter bounds rather than IBP (Zhang et al., 2018).

High-dimensional inputs. For surrogate inputs of dimension $n \geq 50$ (e.g. feeder hosting-capacity with detailed topology), IBP bounds widen roughly as $\sqrt{n}$ under variance-preserving initialisation (linearly in n at fixed weight scale) and pruning yield drops. CROWN-IBP and tighter linear-bound propagation methods (Zhang et al., 2020; Wang et al., 2021) then become worthwhile; the framework supports plug-in replacement of IBP by these alternatives. A separate limitation of the same construction: IBP requires a box domain, so non-box physical sets (the unit circle of Section 3) must be over-approximated, and the bounds and pruning counts are computed on the superset.

Poorly-behaved surrogate training. If the training data does not adequately cover $\mathcal{X}$ (a real risk for high-dimensional inputs), $\hat{f}$ may fit interior data well while having unbounded extra-

polation error near the boundary. The framework cannot detect this; uncertainty quantification on $\hat{f}$ (conformal prediction, ensemble disagreement) is the supplementary tool, applied at the data-collection stage rather than the embedding stage.

5.2 Connection to other surrogate frameworks

The framework is structurally close to OMLT (Ceccon et al., 2022), ENTMOOT (Thebelt et al., 2021), and the chemical-engineering ALAMO-style surrogate optimisation programme (Cozad et al., 2014). The differentiation is in the explicit error decomposition (4), the audit discipline of Section 3.4, and the energy-system applications (wake, hosting, process); the pruning step follows Lastrucci et al. (2026). For chemical-process applications outside our scope, OMLT and ALAMO are mature alternatives.

5.3 Surrogate vs full physics

The most common pushback on surrogate-based optimisation is that the simulator is the source of truth, and any approximation by a surrogate is a downgrade. Our response: in any operational context where the simulator is too slow to call inside the optimisation loop, the actual choice is between (i) an approximate surrogate inside the loop, or (ii) calling the simulator post-hoc on the optimiser's solution. Option (ii) does not produce a physics-aware optimum; it produces a physics-evaluated check on a physics-blind one. But our own case study illustrates the caveat sharply: when the true optimum is a boundary point (zero curtailment) and the surrogate's worst-case error exceeds any achievable gain, the surrogate-in-the-loop "optimum" is worse than doing nothing, and the post-hoc evaluation is exactly what catches it. The right conclusion is not that either option wins outright, but that option (i) is meaningful only when the surrogate's sup error is small against the decision's achievable gain — a comparison the decomposition (4) makes explicit — and that the post-hoc check of option (ii) should always be run on the surrogate's answer regardless. The simulator-as-source-of-truth view is right for evaluation but insufficient for optimisation; the framework, with its audit stage, is the bridge.

6 Conclusion

We have specified a unified framework for embedding non-linear physics into MILP-based energy-system optimisation via ReLU surrogates with IBP-derived big-M bounds and explicit pruning of always-active and always-inactive neurons. The framework is correct (Proposition 2.3), tractable in practice (the case study solves in hundredths of a second on the 24-neuron instance, and at production scale — 128-64 hidden units — the tightened encoding cuts the presolve-off solve from 3.6 s and 911 nodes to 1.0 s and 164 nodes), and accompanied by an explicit error decomposition (4), closed by a measured adversarial-search sup estimate and enforced by an audit stage that evaluates the true function at every surrogate optimum. That audit discipline — report the realised gap, never just the surrogate's own claim — is, we would argue, the paper's most transportable lesson. The framework is the intended analytical substrate for wake-endogenous expansion, hosting-capacity, and process-surrogate modelling on the Compounding Energy roadmap; it is presented here as a method study, not a description of production systems.

The piece that most deserves further work is the bound-propagation step. IBP is sound and adequate at the problem dimensionality studied here, but tighter alternatives (CROWN, β-CROWN, Lirpa) will become operationally meaningful as surrogate networks grow. A v2 of this paper will replace the IBP step with CROWN and report the impact on big-M tightness and solver wall time across a richer benchmark set.

Acknowledgements

This work builds on the ReLU-MILP embedding literature (Tjeng et al., 2019; Anderson et al., 2020; Fischetti and Jo, 2018; Grimstad and Andersson, 2019) and the bound-propagation literature originating in adversarial-robustness research (Wang et al., 2018; Zhang et al., 2018). The Lastrucci 2026 framework (Lastrucci et al., 2026) for pruning trained ReLU networks was a particular inspiration for the integration. The case study is reproducible from the Python implementation accompanying this paper, available on request. Comments are welcome to `christopher@compoundingenergy.com`.

References

Ross Anderson, Joey Huchette, Will Ma, Christian Tjandraatmadja, and Juan Pablo Vielma. Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming*, 183:3–39, 2020. doi: 10.1007/s10107-020-01474-5.

Fani Boukouvala, Ruth Misener, and Christodoulos A. Floudas. Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO. *European Journal of Operational Research*, 252(3):701–727, 2016. doi: 10.1016/j.ejor.2015.12.018.

José A. Caballero and Ignacio E. Grossmann. An algorithm for the use of surrogate models in modular flowsheet optimization. *AIChE Journal*, 54(10):2633–2650, 2008. doi: 10.1002/aic.11579.

Francesco Ceccon, Jordan Jalving, Joshua Haddad, Alexander Thebelt, Calvin Tsay, Carl D. Laird, and Ruth Misener. OMLT: Optimization & machine learning toolkit. *Journal of Machine Learning Research*, 23(349):1–8, 2022.

Alison Cozad, Nikolaos V. Sahinidis, and David C. Miller. Learning surrogate models for simulation-based optimization. *AIChE Journal*, 60(6):2211–2227, 2014. doi: 10.1002/aic.14418.

Anamika Dubey and Surya Santoso. On estimation and sensitivity analysis of distribution circuit's photovoltaic hosting capacity. *IEEE Transactions on Power Systems*, 32(4):2779–2789, 2017. doi: 10.1109/TPWRS.2016.2622286.

Matteo Fischetti and Jason Jo. Deep neural networks and mixed integer linear optimization. *Constraints*, 23:296–309, 2018. doi: 10.1007/s10601-018-9285-6.

Bjarne Grimstad and Henrik Andersson. ReLU networks as surrogate models in mixed-integer linear programs. *Computers & Chemical Engineering*, 131:106580, 2019. doi: 10.1016/j.compchemeng.2019.106580.

Carlos A. Henao and Christos T. Maravelias. Surrogate-based superstructure optimization framework. *AIChE Journal*, 57(5):1216–1232, 2011. doi: 10.1002/aic.12341.

Niels Otto Jensen. A note on wind generator interaction. *Risø National Laboratory Report*, (Risø-M-2411), 1983.

I. Katic, J. Højstrup, and Niels Otto Jensen. A simple model for cluster efficiency. *European Wind Energy Association Conference and Exhibition*, pages 407–410, 1986.

G. Lastrucci, T. Karia, V. Schulte, D. Bongartz, and A. M. Schweidtmann. Pruning for efficient deterministic global optimization over trained ReLU neural networks. *arXiv preprint arXiv:2603.23299*, 2026.

Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J. Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018. doi: 10.1080/01621459.2017.1307116.

Amin Niayifar and Fernando Porté-Agel. Analytical modeling of wind farms: A new approach for power prediction. *Energies*, 9(9):741, 2016. doi: 10.3390/en9090741.

Richard J. A. M. Stevens, Dennice F. Gayme, and Charles Meneveau. Effects of turbine spacing on the power output of extended wind-farms. *Wind Energy*, 19(2):359–370, 2016. doi: 10.1002/we.1835.

Alexander Thebelt, Jan Kronqvist, Miten Mistry, Robert M. Lee, Nathan Sudermann-Merx, and Ruth Misener. ENTMOOT: A framework for optimization over ensemble tree models. *Computers & Chemical Engineering*, 151:107343, 2021. doi: 10.1016/j.compchemeng.2021.107343.

Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *International Conference on Learning Representations (ICLR)*, 2019.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, New York, 2005.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J. Zico Kolter. Beta-CROWN: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems*, 34:29909–29921, 2021.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. *Advances in Neural Information Processing Systems*, 31, 2018.

Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *International Conference on Learning Representations (ICLR)*, 2020.

A Notation glossary

Symbol	Meaning
f^*, $\hat{f}$	True non-linear function and ReLU-MLP surrogate
$\mathcal{X}$	Input domain (closed box in $\mathbb{R}^n$)
$\boldsymbol{W}_\ell, \boldsymbol{b}_\ell$	Weight matrix and bias of layer ℓ
$\boldsymbol{z}_\ell, \boldsymbol{a}_\ell$	Pre- and post-activation of layer ℓ
$\delta_{\ell,j}$	Binary indicator for ReLU node (ℓ, j) in big-M encoding
M	Big-M constant in the ReLU MILP encoding
$[\underline{z}_{\ell,j}, \overline{z}_{\ell,j}]$	IBP-derived pre-activation bounds
W^+, W^-	Positive and negative parts of $\boldsymbol{W}$, componentwise

B Reproducibility

The case study reported in Section 3 is reproducible end-to-end from the Python implementation accompanying this paper. The implementation requires Python 3.10+, NumPy, SciPy (with HiGHS-MILP via `milp`), Pandas, and Matplotlib. The random seed is fixed at 20 260 901. Stages `data`, `train`, `prune`, `bound`, `mitigation`, `audit`, `optimise`, `plots` are individually callable; `mitigation` and `audit` produce the ground-truth mitigation-existence and surrogate-optimum-audit artefacts quoted in Section 3.4. Each results artefact is stamped with a scenario identifier, the SHA-256 of the source, and a UTC timestamp, so a run is keyed by `(scenario_id, code_version, timestamp)` as well as the seed.