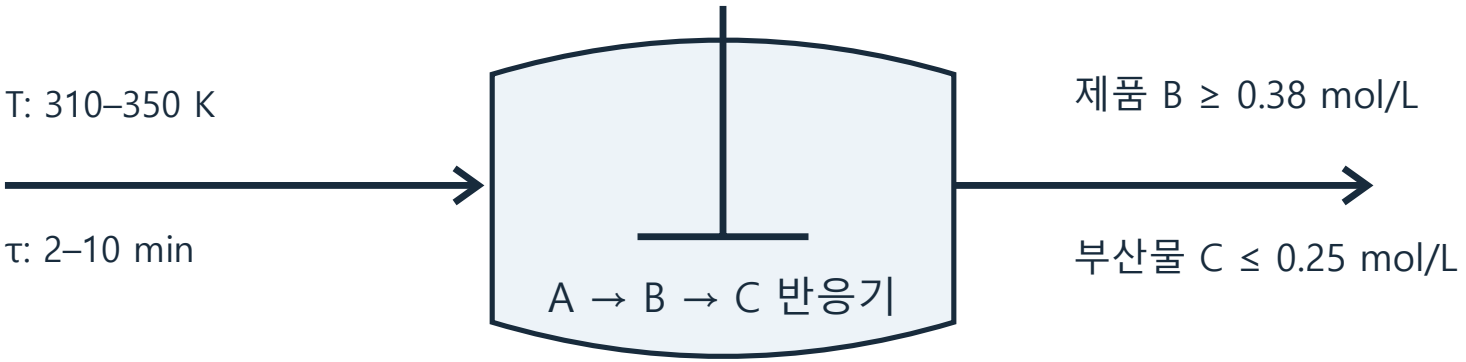
6주차: 신경망 기반 최적화의 풀이 속도와 결과 해석

제품 규격을 만족하는 반응기 조건

반응기 설계 검토에서 온도와 체류시간 후보를 반복해서 계산합니다. 제품 규격을 만족하는 조건을 찾았어도 더 나은 조건이 남아 있는지 판단해야 합니다.



계산 시간 안에 찾은 해를 어떻게 판단할까?

이번 주에는 계산을 마치기 전에도 해의 품질을 평가하고 bounds와 정식화를 조정하는 방법을 다뤄봅니다.

비교할 문제와 계산 조건

5주차 반응기 모델에 온도를 입력으로 추가합니다. 유량을 1 L/min로 고정하므로 체류시간은 반응기 부피도 결정하는 설계변수입니다.

학습을 마친 ReLU 모델, 입력 영역, 목적함수와 제품 규격을 고정하고 정식화만 비교합니다. 온도 범위를 임의로 줄이면 원래 설계 문제와 달라집니다.

모든 LP와 MILP는 오픈소스 HiGHS로 풉니다.

정식화와 solver 결과의 학습 목표

1. 가능해의 목적값, 최적값에 대한 bound와 gap
2. 구간 전파와 LP 기반 bound tightening 비교
3. Pyomo·OMLT의 고정 신경망과 scaling 연결
4. 종료 상태, 계산 시간, forward 일치와 제약 잔차

5주차의 Big-M 동치성과 입력 bounds 계산을 복습한 뒤 시작합니다.

고정 신경망과 MILP 표현

각 ReLU의 입력 a에 유효한 $L \le a \le U$가 있고 $L < 0 < U$이면 다음 식에 이진변수 z를 사용합니다.

$$h \ge 0, h \ge a$$
$$h \le Uz, h \le a - L(1 - z)$$

가중치·bias·scaling은 고정합니다. 다른 제약과 목적함수까지 선형이면 전체 문제가 MILP입니다. 정확한 신경망 표현은 공정 예측 오차와 별개입니다.

가능해, bound와 gap

최소화에서 incumbent는 지금까지 찾은 가장 좋은 정수 가능해입니다. 이 가능해의 목적값을 상한으로 사용하면 유효한 하한과 함께 최적값의 범위를 정할 수 있습니다.

$$J_{\text{bound}} \le J_{\text{opt}} \le J_{\text{inc}}$$

$$g_{\text{abs}} = J_{\text{inc}} - J_{\text{bound}}, g_{\text{rel}} = \frac{J_{\text{inc}} - J_{\text{bound}}}{\left|J_{\text{inc}}\right|}$$

상한 100, 하한 80이면 절대 gap은 20이고 이 정의의 상대 gap은 20%입니다. Incumbent가 없으면 이 gap을 계산할 수 없으며, 목적값이 0에 가까우면 절대 gap을 함께 보고합니다.

가능해 하나를 찾으면 그 비용보다 나쁜 설계를 선택할 필요가 없으므로 상한을 얻습니다. 반대로 완화 문제는 원래보다 유리한 선택까지 허용하므로 하한을 줍니다. 두 값 사이가 아직 개선될 수 있는 범위입니다.

Branch-and-bound의 진행

이진변수를 연속 구간으로 완화한 LP를 풀면 최소화 문제의 하한을 얻습니다. LP 해에서 분수값인 이진변수 하나를 선택해 $z = 0$과 $z = 1$인 하위 문제로 나눕니다.

정수 가능해를 찾으면 incumbent를 갱신합니다. 불가능한 노드나 하한이 incumbent보다 나쁘거나 같은 노드는 더 탐색할 필요가 없습니다.

열린 노드들의 유효한 하한을 모아 전체 하한을 계산합니다. 실제 HiGHS는 presolve, cutting plane과 heuristic도 사용하므로 원래 LP 완화값과 solver의 최종 bound는 다를 수 있습니다.

현재 가능해 비용이 100인데 어떤 노드의 하한이 105이면 그 노드의 모든 정수 해는 105 이상이므로 탐색을 멈춥니다. 하한 80인 노드는 100보다 좋은 해가 남아 있을 수 있어 더 조사합니다.

ReLU의 연속 완화

z를 [0, 1]로 완화하고 제거하면 L < 0 < U에서 다음 삼각형을 얻습니다.

$$h \geq 0, h \geq a, h \leq \frac{U}{U-L}\left(a - L\right)$$

a = 0에서 허용하는 최대 h는 −UL/(U − L)입니다. L = −1, U = 1이면 0.5지만 실제 ReLU 출력은 0입니다.

이 삼각형은 스칼라 (a, h) 그래프의 convex hull입니다. 여러 뉴런과 입력 간 상관관계를 함께 표현하는 데에는 추가 제약이 필요할 수 있습니다.

L=−1,U=1에서 a=0을 넣으면 상한은 (0+1)/2=0.5입니다. 실제 h=0보다 높게 허용하는 이 여유가 완화 오차입니다. a의 유효 범위를 좁히면 이 여유도 줄어들 수 있습니다.

등식 제약이 있는 두 뉴런 예제

다음 무차원 예제로 bounds가 놓치는 관계를 직접 계산

$$0 \le x_1, x_2 \le 1, x_1 + x_2 = 1$$
$$h_1 = \max\left(0, x_1 + x_2 - 1\right),$$
$$h_2 = \max\left(0, x_1 - x_2\right)$$
$$\min_{x_1, x_2} x_1 \text{subject to} h_1 + h_2 \ge .4$$

입력 구간만 사용하면 두 pre-activation의 bounds는 모두 [−1, 1]입니다. 하지만 첫 번째 pre-activation은 입력 등식 때문에 항상 0입니다.

상자 범위만 보면 x_1=x_2=0 또는 1을 조합할 수 있습니다. 등식 x_1+x_2=1은 두 조합을 금지하고 첫 pre-activation을 항상 0으로 만듭니다. 각 변수 범위와 변수 간 관계가 주는 정보가 다른 이유입니다.

두 뉴런 예제의 정확한 해

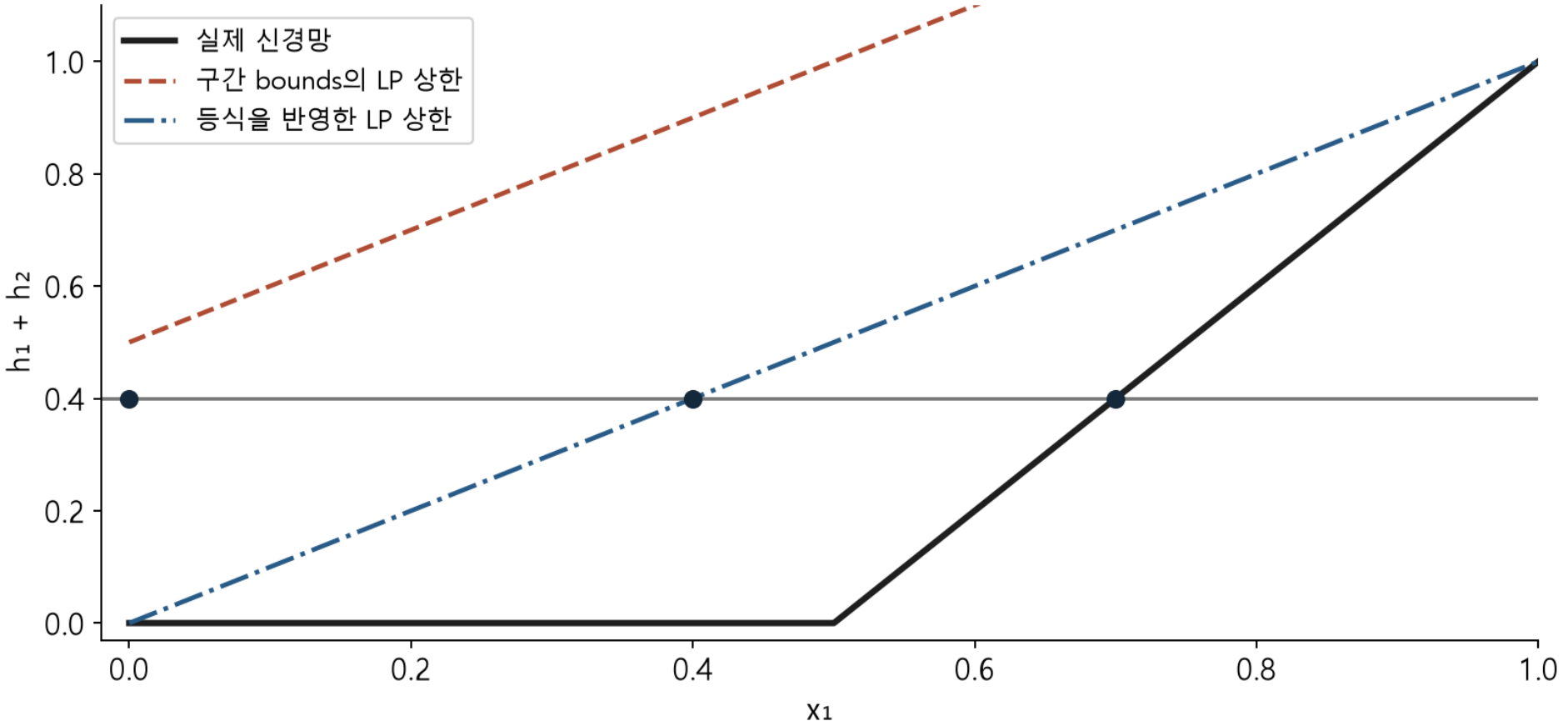
$$h_1 = 0, h_2 = \max\left(0, 2x_1 - 1\right)$$
$$h_2 \geq .4 \Rightarrow x_1 \geq .7$$

따라서 최소 x_1은 0.7입니다. 구간 bounds를 사용하는 MILP와 등식을 반영한 MILP 모두 같은 최적값을 냅니다.

LP 완화에서는 h가 실제 신경망 출력보다 커질 수 있으므로 같은 결론이 바로 성립하지 않습니다.

$x_1 \leq 0.5$이면 두 번째 ReLU도 0이어서 출력 하한 0.4를 채울 수 없습니다. 활성 구간에서는 $2x_1 - 1 \geq 0.4$를 풀어 $x_1 \geq 0.7$을 얻습니다. 0.7을 직접 대입해 가능함을 보이면 하한과 달성 가능한 값이 일치합니다.

Bounds 강화 전후의 LP 완화



원래 LP는 $x_1 = 0$에서도 첫 뉴런의 $h_1 = 0.4$를 허용합니다. $h_1 = 0$을 반영하면 LP 최적값은 0.4로 올라가지만 MILP 최적값 0.7에는 아직 못 미칩니다.

완화 해와 실제 신경망의 대조

각 방식에서 목적값 x_1을 최소화합니다. $x_2 = 1 - x_1$이고 필요한 출력 합은 0.4입니다.

방식	x_1, x_2	허용된 h_1, h_2	실제 ReLU 출력	해석
상자 bounds LP	0,1	0.4,0	0,0	가짜 h_1로 규격 충족
등식 반영 LP	0.4,0.6	0,0.4	0,0	가짜 h_2로 규격 충족
MILP	0.7,0.3	0,0.4	0,0.4	원래 출력과 일치

등식 반영 뒤 두 번째 삼각형의 상한은 $h_2 \le x_1$입니다. 따라서 $h_2 \ge 0.4$를 만족하려면 $x_1 \ge 0.4$입니다. 그러나 $x_1 = 0.4$에서 실제 $a_2 = -0.2$이므로 ReLU 출력은 0입니다. 하한이 좋아져도 완화가 원래 그래프와 같아진 것은 아닙니다.

두 뉴런 예제의 분기와 탐색 종료

$h_1=0$을 반영한 문제에서 h_2의 이진변수 z_2만 남깁니다. $L=-1, U=1$이고 $a_2=2x_1-1$입니다. 루트 LP의 $h_2\le z_2$, $h_2\le 2x_1-z_2$를 더하면 $h_2\le x_1$입니다. $h_2\ge 0.4$이므로 $x_1\ge 0.4$가 하한입니다.

문제	계산	결론
루트 LP, $0\le z_2\le 1$	$x_1=0.4, h_2=0.4, z_2=0.4$ 가능	하한 0.4, 정수해 아님
$z_2=0$인 자식	$h_2=0$과 $h_2\ge 0.4$ 충돌	불가능, 종료
$z_2=1$인 자식	$h_2=2x_1-1\ge 0.4$	$x_1=0.7$에서 최소 달성

두 자식은 가능한 이진값을 모두 덮습니다. 한쪽은 불가능하고 다른 쪽은 최소 0.7이므로 전역 하한도 0.7입니다. 가능해 비용 0.7과 같아져 gap=0으로 종료합니다.

여러 열린 노드가 남으면 그중 가장 작은 하한이 아직 더 좋은 해를 허용하는 가지를 나타냅니다. 하한이 incumbent 이상인 가지는 개선할 수 없어 제외합니다. 실제 solver의 분기 순서와 cut은 다를 수 있습니다.

Bounds의 유효한 범위

Bounds는 허용한 모든 입력에서 성립해야 합니다. 학습 데이터의 최솟값·최댓값만으로는 입력 영역 전체를 덮는다고 보장할 수 없습니다.

너무 좁은 bounds는 유효한 신경망 점을 잘라내고, 넓은 bounds는 정수해의 그래프를 유지하면서 완화를 약하게 만들 수 있습니다.

제품 규격까지 사용해 계산한 bounds는 해당 규격의 가능집합에만 유효합니다. 규격이나 입력 영역을 바꿀 때는 bounds를 다시 확인합니다.

가중치 부호를 반영한 구간 전파

이전 층의 각 출력이 $\ell \le h \le u$에 있고 $w^+ = \max(w, 0)$, $w^- = \min(w, 0)$라고 놓습니다.

$$L_j = b_j + \sum_i \left(w_{ij}^{+} \ell_i + w_{ij}^{-} u_i \right)$$

$$U_j = b_j + \sum_i \left(w_{ij}^{+} u_i + w_{ij}^{-} \ell_i \right)$$

ReLU 뒤에는 [max(0, L), max(0, U)]를 다음 층에 전달합니다. 계산 비용은 작지만 각 변수의 극값을 동시에 달성할 수 있는지는 반영하지 않습니다.

예를 들어 $h_1 = h_2$인 관계를 버리고 두 값의 범위만 [0,1]로 두면 $h_1 - h_2$ 범위는 [−1,1]입니다. 실제 차이는 0입니다. 구간 전파는 빠르게 안전한 외곽 범위를 만들지만 이런 상관관계는 사용하지 못합니다.

LP 기반 bound tightening

P를 신경망 MILP 가능집합을 포함하는 LP 완화라고 놓습니다. 각 pre-activation을 목적함수로 바꾸어 최소화·최대화합니다.

$$L_j = \min_{v \in P} a_j\left(v\right)$$
$$U_j = \max_{v \in P} a_j\left(v\right)$$

이 방식은 optimization-based bound tightening(OBBT). 실습은 제품 규격을 제외한 입력 영역 전체에서 LP 54개를 풀고, bounds를 적용한 MILP를 새로 만듭니다.

하한 계산은 “가능한 모든 입력에서 이 뉴런이 얼마나 작아질 수 있는가”라는 별도 최적화입니다. 상한은 같은 식을 최대화합니다. 원래 신경망의 모든 점을 포함한 완화 집합에서 풀어야 원래 점을 자르지 않는 bounds가 됩니다.

예를 들어 a의 최소값을 구하다가 가능점 a=−0.2와 유효 하한 −0.6에서 멈췄다면 실제 최소는 [−0.6,−0.2]에 있습니다. 뉴런의 하한에는 −0.6을 사용합니다. −0.2를 사용하면 실제 a=−0.4인 점을 잘라낼 수 있습니다. 최대값 계산에서는 반대로 유효 상한을 사용합니다.

활성이 고정된 뉴런의 처리

$$U \leq 0 \Rightarrow h = 0$$
$$L \geq 0 \Rightarrow h = a$$

이 두 경우에는 이진변수가 필요하지 않습니다. 0을 가로지르는 뉴런만 분기 대상으로 남깁니다.

직접 작성한 MILP의 이진변수 수: 넓은 bounds 24개, 구간 전파 22개, LP-OBBT 19개. 집계 시점: solver presolve 적용 전

입력 scaling과 물리 단위

$$x_1 = \frac{T - 310}{40}, x_2 = \frac{\tau - 2}{8}$$
$$T = 310 + 40x_1, \tau = 2 + 8x_2$$

학습 때 사용한 offset과 scale을 그대로 가져옵니다. 물리 입력의 bounds를 scaled 입력에 잘못 전달하면 신경망을 평가하는 영역이 바뀝니다.

실습 출력 단위: mol/L. 표준화된 출력을 사용한다면 제품 규격에 연결하기 전에 고정된 역변환도 넣어야 합니다.

330 K, 6 min은 두 축에서 모두 중앙이므로 scaled input [0.5,0.5]입니다. 입력 순서를 바꾸면 온도 가중치가 체류시간을 읽게 됩니다. 동일한 physical input에 대한 forward 비교는 이 순서 오류도 찾습니다.

목적함수의 상수와 상대 gap

$$J = \tau + \alpha\left(T - 310\right), \alpha = .02\text{min/K}$$
$$J = 2 + 8x_2 + .8x_1$$

교육용 목적함수: 체류시간 + α로 가중한 온도 증가. 단위: min 등가

상수 2를 빼도 최적 입력은 같지만 상대 gap의 분모는 바뀝니다. 실습은 상수를 포함한 목적값과 bound를 그대로 보고

목적함수 상수와 gap의 계산

동일한 가능해와 하한에서 목적함수의 상수 2만 제거한다고 둡니다.

계산	상수 포함 J	상수 제거 J−2
가능해 목적값	3.2	1.2
유효 하한	2.8	0.8
절대 gap	3.2−2.8=0.4	1.2−0.8=0.4
상대 gap	0.4/3.2=12.5%	0.4/1.2≈33.3%

모든 후보의 값에서 같은 수를 빼므로 후보 순위와 최적 입력은 유지됩니다. 하지만 상대 gap의 분모는 달라집니다. 두 보고서를 비교할 때 목적함수 정의가 다르면 %만으로 풀이 품질을 비교할 수 없습니다.

가능해가 없으면 분모에 사용할 값도 없습니다. 그때는 유효 하한과 가능해 부재를 보고하고, 변수의 초기값을 해로 해석하지 않습니다.

오픈소스 계산 도구의 역할

도구	역할
NumPy	신경망 forward 계산과 선택 재학습
Pyomo	변수·목적함수·공정 제약 작성
OMLT	고정 신경망을 Pyomo 제약으로 변환
HiGHS / highspy	LP와 MILP 풀이

모든 LP와 MILP는 MIT 라이선스의 오픈소스 HiGHS로 풉니다.

OMLT가 만든 제약도 입력 순서, scaling, bounds와 함께 확인. HiGHS 공식 안내

OMLT와 공정 제약 연결

가중치와 scaling을 담은 `definition`을 만든 뒤 다음과 같이 연결. `nn.inputs`는 물리 단위이고 `scaled_inputs`는 변환된 입력

```
model.nn = OmltBlock()
model.nn.build_formulation(ReluBigMFormulation(definition))
model.product = pyo.Constraint(expr=model.nn.outputs[1] >= .38)
model.byproduct = pyo.Constraint(expr=model.nn.outputs[2] <= .25)
model.obj = pyo.Objective(
    expr=model.nn.inputs[1] + .02*(model.nn.inputs[0]-310))
```

outputs[1]은 B, outputs[2]는 C입니다. inputs[0]은 K 단위 온도이고 inputs[1]은 min 단위 체류시간입니다. OMLT의 scaling 등식이 physical input을 신경망 좌표로 연결하므로 목적식에 scaled input을 잘못 사용하지 않습니다.

5주차 예제의 OMLT 재현

온도 330 K, 유입 A 농도 1.2 mol/L에서 체류시간만 선택하는 기존 보간 모델을 먼저 확인. $B \geq 0.40$, $C \leq 0.25$ mol/L를 적용

정식화	최소 체류시간 [min]	생성된 이진변수
직접 작성	5.353846	3
OMLT	5.353846	4

OMLT는 이 버전에서 ReLU마다 이진변수를 생성합니다. 직접 작성한 모델은 항상 활성인 뉴런 하나를 선형으로 표현하므로 이진변수가 하나 적고 출력은 같습니다.

온도와 체류시간을 선택하는 문제

$$\min_{T,\tau} J = \tau + .02\left(T - 310\right)$$
$$310 \le T \le 350, 2 \le \tau \le 10$$
$$y = f_\theta\left(T, \tau\right), y_B \ge .38, y_C \le .25$$

농도는 mol/L, 온도는 K, 체류시간은 min. 각 예측 농도에는 비음수 제약도 적용

신경망 구조: 입력 2개, 뉴런 12개씩의 ReLU 은닉층 두 개, 선형 출력 3개. 비교 기준: 동일한 고정 계수

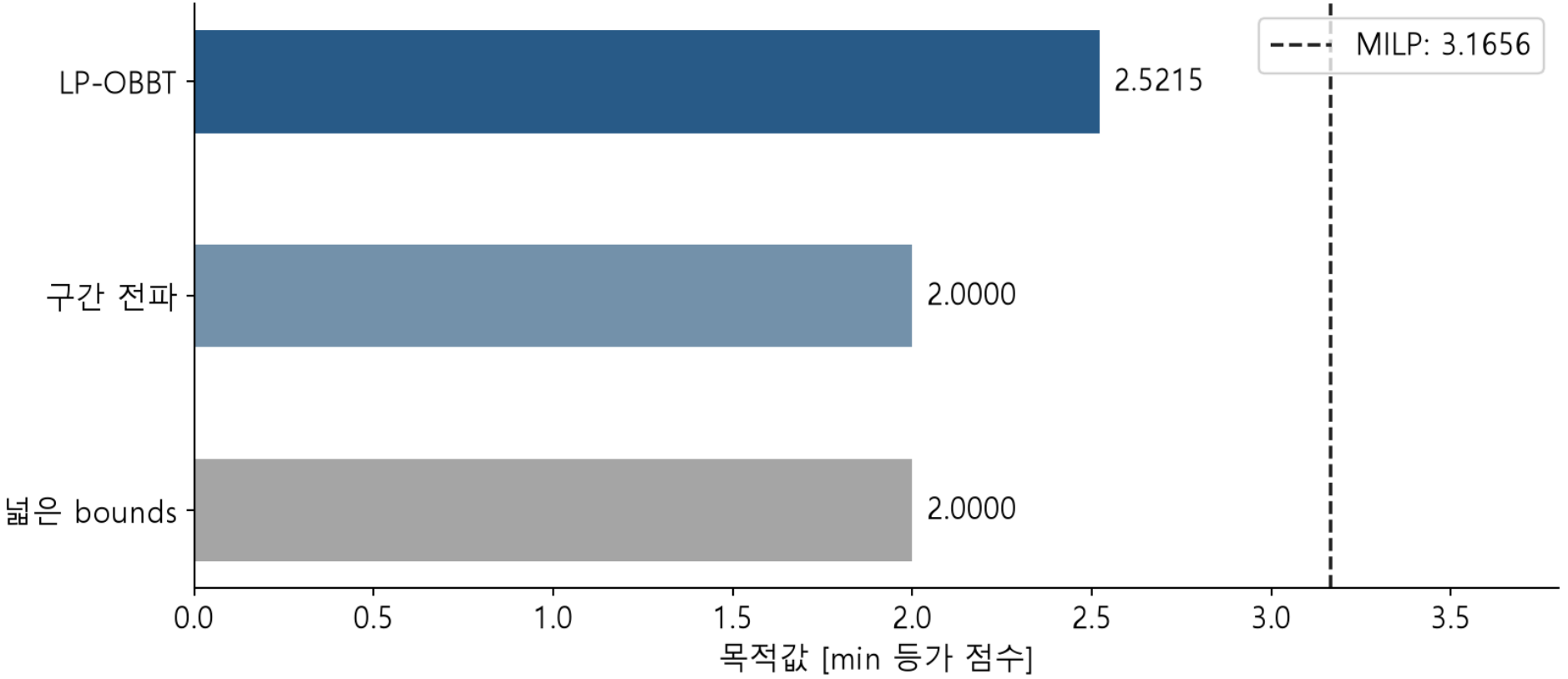
고정 모델의 학습과 독립 예측 평가

이상적 등온 A → B → C CSTR 수지에서 학습 입력 1,200개의 농도를 생성하고 NumPy Adam으로 6,000회 갱신했습니다. 독립 난수 입력 500개에서 물리 단위 오차를 평가

성분	Test RMSE [mol/L]
A	0.00546
B	0.00440
C	0.00564

기본 실습은 제공된 계수를 읽습니다. `--refit`은 선택 재학습이며 최적화 도중 가중치를 바꾸지 않습니다.

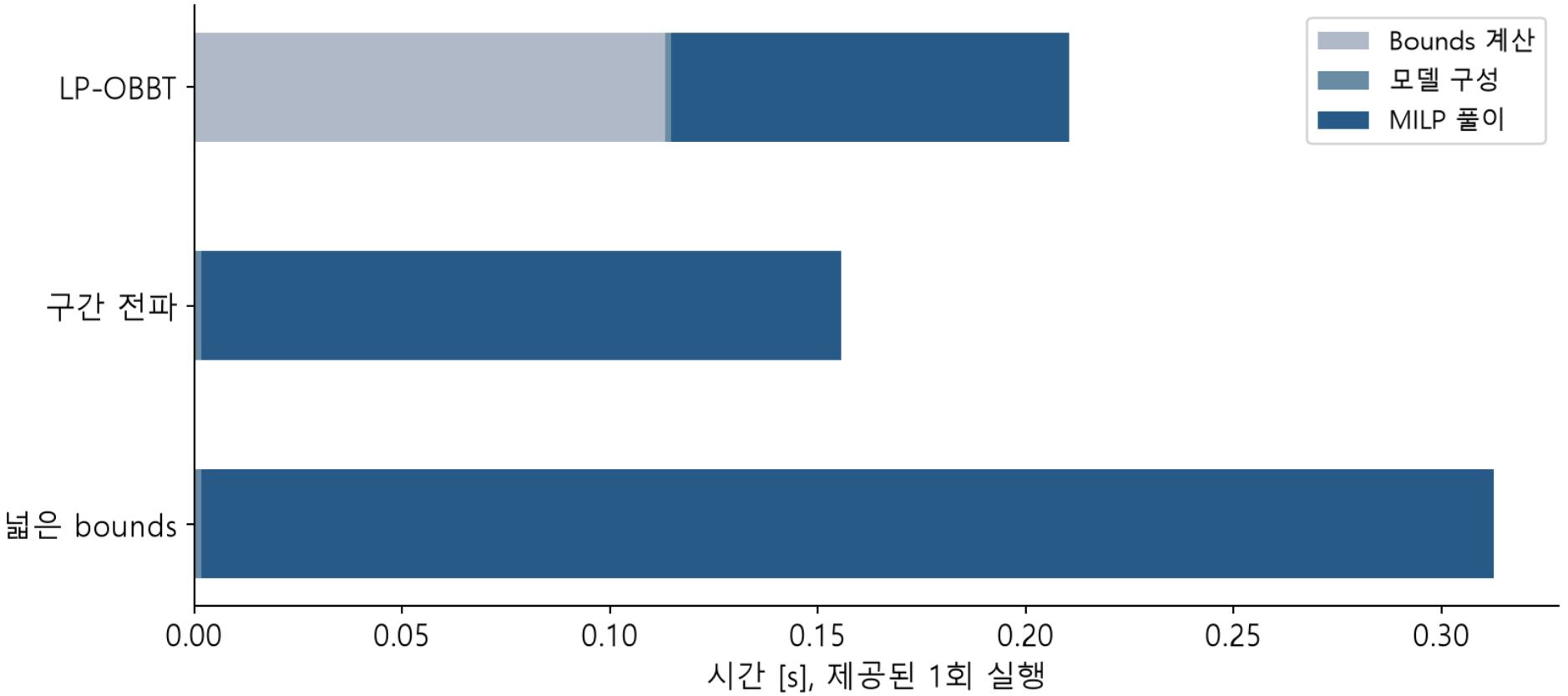
Bounds에 따른 완화값 비교



LP-OBBT는 원래 LP 하한을 2.0000에서 2.5215로 올렸습니다. 세 MILP의 최적값은 모두 3.1656으로 일치합니다. 세 비교에는 같은 예측 모델을 사용했습니다.

하한 2.5215와 가능해 3.1656 사이의 차이는 약 0.6441입니다. 느슨한 하한 2.0000일 때의 1.1656보다 좁습니다. 모델 예측은 같아도 탐색 시작 시 남는 최적성 불확실성이 줄어드는 것입니다.

Bounds 계산 비용을 포함한 시간



제공 실행에서 LP-OBBT의 MILP 풀이 시간은 줄었지만 bounds 계산을 더한 총시간은 구간 전파보다 길었습니다. 인터페이스 비용을 포함해 한 번 측정한 결과이므로 이 속도 순위가 일반적으로 성립하는지는 알 수 없습니다.

bounds를 재사용할 때의 총시간

기록된 한 번의 실행에서 구간 전파의 MILP 시간은 0.1539 s, LP-OBBT는 0.0955 s입니다. OBBT 전처리는 0.1132 s가 필요했습니다.

풀이 수 m	구간 전파: 0.0001+m×0.1555	OBBT: 0.1132+m×0.0971
1	0.1556 s	0.2103 s
2	0.3111 s	0.3074 s
5	0.7776 s	0.5987 s

열별 1회 비용에는 build 0.0016 s를 포함합니다. 표는 반올림된 기록 시간을 사용하고 각 반복의 풀이 시간이 같다고 가정한 산술 계산입니다. 실제 반복 실험 결과는 아닙니다.

고정된 가중치와 입력 영역에서 bounds를 재사용할 수 있을 때 전처리 비용을 나누어 부담합니다. 새 모델이나 더 넓은 영역이면 bounds 유효성을 다시 확인해야 하며 이 계산을 그대로 적용할 수 없습니다.

선택된 조건과 solver의 결론

항목	제공 실행
온도 [K]	342.059300
체류시간 [min]	2.524440
목적값 [min 등가]	3.165626
예측 B [mol/L]	0.380000
종료 상태 / 상대 gap	optimal / 0

HiGHS는 설정한 수치 허용오차 안에서 이 surrogate MILP의 최적성을 확인했습니다. B 하한은 활성이고 C 상한에는 여유가 있습니다.

목적값은 2.524440 + 0.02 × (342.059300−310) = 3.165626입니다. 온도 상승 비용 0.641186이 체류시간 비용에 더해집니다. 입력과 목적값을 다시 계산하면 solver 보고 값의 단위·offset 처리도 확인할 수 있습니다.

Forward 일치와 제약 잔차 검사

제품 규격을 뺀 상태에서 입력을 고정하고 신경망 표현만 풉니다. 입력 영역의 네 모서리와 독립 내부점 12개를 직접 작성한 모델과 OMLT에 각각 넣습니다.

32회 검사에서 출력의 최대 절대 차이는 7.77e-16 mol/L였습니다. 선택된 최적 조건에서도 forward 출력을 다시 계산합니다.

제약 잔차와 이진변수의 정수성 잔차도 확인합니다. 표본 검사는 구현 오류를 찾는 데 사용합니다. 영역 전체의 수학적 동치는 5주차의 정식화 조건이 성립해야 보장됩니다.

부등식 잔차는 만족할 때 0이 되도록 계산합니다. 예를 들어 $B \geq 0.38$의 위반량은 $\max(0, 0.38 - B)$, $C \leq 0.25$는 $\max(0, C - 0.25)$입니다. 이진값 $z = 0.999999$의 정수성 잔차는 가까운 정수 1과의 거리 0.000001이며, solver 허용오차와 비교합니다.

종료 상태와 가능해 존재 여부

종료 상태	해석
optimal	설정한 gap·수치 허용오차 안에서 완료
시간 제한, incumbent 있음	가능해와 남은 gap을 보고
시간 제한, incumbent 없음	사용할 해를 아직 확보하지 못함
infeasible	작성한 문제의 가능집합이 비어 있음
unbounded / error	Bounds·목적식·로그와 모델 확인

시간 제한만으로 infeasible을 결론 내리지 않습니다. 실습의 0초 제한은 `maxTimeLimit`과 incumbent 없음, $B \geq 2$인 경우는 `infeasible`을 반환했습니다.

가능해 확인과 결과 불러오기

```
opt = Highs()
opt.config.load_solution = False
opt.config.time_limit = 30.0
result = opt.solve(model)
inc = result.best_feasible_objective
print(result.termination_condition)
if inc is not None and np.isfinite(inc):
    result.solution_loader.load_vars()
    print(inc, result.best_objective_bound)
```

가능해가 없는 종료에서 변수 값을 읽으면 초기값이나 이전 계산값을 결과로 오해할 수 있습니다. 실습 함수는 gap과 제약 잔차도 반환합니다.

load_solution=False는 풀이 직후 변수에 값을 넣지 않도록 합니다. 유한한 가능해 목적값이 확인된 경우에만 load_vars를 호출합니다. 그 다음 읽은 값에 대해 규격 잔차와 정수 제약 위반을 계산합니다.

Solver gap과 공정 예측 오차

선택한 같은 입력	B 농도 [mol/L]
신경망 및 MILP	0.380000
기준 CSTR	0.371157
요구 하한	0.380000

기준 CSTR의 B 농도는 규격보다 낮습니다. 정식화가 forward와 일치하고 solver gap이 0이어도 공정 모델의 제약 만족은 따로 확인해야 합니다.

8주차에서는 이런 선택 지점의 오차를 바탕으로 조건을 재검토하고 추가 데이터를 정합니다.

실습 실행과 비교 보고서

실습 묶음을 풀고 같은 폴더에서 실행합니다.

```
python -m pip install -r requirements.txt
python week06_lab.py --output-dir my-results
```

1. 두 뉴런 예제의 LP 값 0, 0.4와 MILP 값 0.7을 손으로 확인
2. 동일한 고정 모델에서 bounds·이진변수 수·LP 하한·총시간을 비교
3. 종료 상태·gap·forward 차이와 기준 CSTR의 B 규격 위반을 보고

읽기자료 · 실습 ZIP

출처와 지정 읽기

OMLT 논문: Ceccon 등, JMLR 23(349), 2022 · 공식 사용 예제

Anderson 등, Strong mixed-integer programming formulations for trained neural networks, Mathematical Programming, 2020

HiGHS 공식 사이트와 MIT 라이선스 안내 · Pyomo의 HiGHS 인터페이스