

QUBO Constraints, Penalty Weights, and GPBO

20 頁精簡授課版：從 Penalty 到 Gaussian Process Bayesian Optimization

QUBO 教學投影片

2026

QUBO：數學形式與 Fred Glover

Quadratic Unconstrained Binary Optimization (QUBO)：

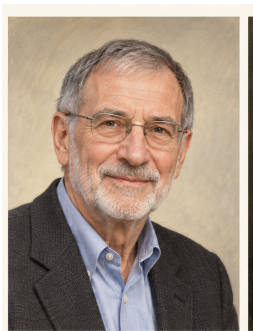
$$\min_{x \in \{0,1\}^n} E(x) = x^T \mathbf{Q} x.$$

$x = [x_1, \dots, x_n]^T$ 為 binary decision vector； $\mathbf{Q}$ 為 coefficient matrix； E 越小越好。因 $x_i^2 = x_i$ ，diagonal quadratic terms 可視為 linear terms。

Fred Glover 與 Gary Kochenberger 等人長期推廣現代 QUBO 建模與 metaheuristics；QUBO 並非由單一人物在單一年份發明，其根源包含 0-1 quadratic 與 pseudo-Boolean optimization。

中學生版直覺

x 是一排 0/1 開關， $\mathbf{Q}$ 是計分規則，QUBO 是讓許多最佳化問題共用的「翻譯語言」。



Fred Glover

Fred Glover

Constraints 如何變成 Penalty Terms ?

原問題

$$\min f(x) \quad \text{s.t.} \quad g_r(x) = 0$$

可改寫成 unconstrained objective

$$E(x) = f(x) + \sum_r \lambda_r P_r(x), \quad P_r(x) = [g_r(x)]^2.$$

要求：constraint 成立時 $P_r = 0$ ；違規時 $P_r > 0$ ； $\lambda_r > 0$ 控制處罰強度。

Exactly-one example: $x_1 + x_2 + x_3 = 1$

$$P = (x_1 + x_2 + x_3 - 1)^2 = 1 - x_1 - x_2 - x_3 + 2x_1x_2 + 2x_1x_3 + 2x_2x_3.$$

中學生版直覺

原本「不能違規」寫在規則欄；QUBO 改成「違反規則就增加成本」，所以 constraint 被搬進 objective。

Penalty Weight λ ：為什麼不能太小？

若 x_F 是最佳 feasible solution， x_I 是某個 infeasible solution，要讓 feasible 勝出需

$$f(x_F) < f(x_I) + \lambda P(x_I),$$

因此

$$\lambda > \frac{f(x_F) - f(x_I)}{P(x_I)}.$$

例：違規可讓原 objective 從 10 降到 6，而最小 violation penalty 為 1，則

$$10 < 6 + \lambda \quad \Rightarrow \quad \lambda > 4.$$

λ 太小：solver 可能接受違規來換取較低的原始成本；太大：objective 差異被 penalty 淹沒、數值尺度變差。

中學生版直覺

若違反規則可讓原始成本少 4 元，違規增加的成本就必須超過 4 元，否則「違規仍比較便宜」。

三種最簡單的 Static Penalty Weight 方法

- 1 **Fixed large weight:** 先選一個明顯較大的 λ 。簡單，但可能過大。
- 2 **Objective-range / gain bound:** 估計違規最多能改善多少 objective，令 penalty 超過此 gain，例如 $\lambda > \Delta f_{\max} / P_{\min}$ 。
- 3 **Incremental tuning:** 從小 λ 開始；若最佳解仍違規就逐步放大，例如 $\lambda \leftarrow c\lambda, c > 1$ 。

多個 constraints 時：

$$E = f + \lambda_1 P_1 + \lambda_2 P_2 + \dots$$

不同 constraints 的尺度與重要性不同，因此通常不應強迫共用一個 λ 。

中學生版直覺

不同規則的違反成本不一定相同：小違規可能只增加少量成本，重要 constraint 的違反則應增加更大的成本。

雙 Penalty Weights : Grid Search 的代價

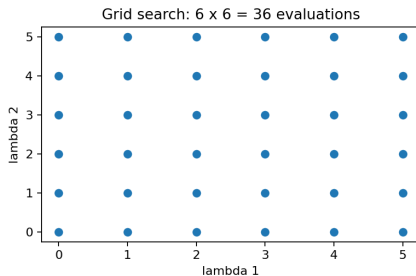
若有 $\lambda = (\lambda_1, \lambda_2)$ ，可用 grid search：

$$\lambda_1 \in \{a_1, \dots, a_m\}, \quad \lambda_2 \in \{b_1, \dots, b_n\}.$$

需做 $m \times n$ 次 expensive evaluations。若每軸 6 點就是 36 次；三個 weights 則為 $6^3 = 216$ 次。

中學生版直覺

像在整張地圖每隔固定距離挖洞找寶藏：公平、簡單，但很多洞其實沒有必要。



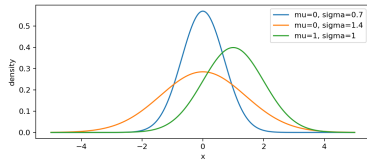
Gaussian Distribution 與 Carl Friedrich Gauss

單變量 Gaussian :

$$X \sim \mathcal{N}(\mu, \sigma^2), \quad p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}.$$

μ 是中心； σ^2 是 variance； σ 越大分布越寬。區間機率為
 $P(a \leq X \leq b) = \int_a^b p(x) dx$ 。

Carl Friedrich Gauss (1777-1855) 對數學、天文、測量與 least squares 有深遠貢獻；normal/Gaussian distribution 的名稱與其誤差研究密切相關。



Carl Friedrich Gauss
(1777-1855)

中學生版直覺

預測同學身高：不只猜「大約 165 cm」，還要說「我有多確定」。

從 Multivariate Gaussian 到 Gaussian Process

多個 random variables 組成 $\mathbf{y} = [y_1, \dots, y_m]^T$:

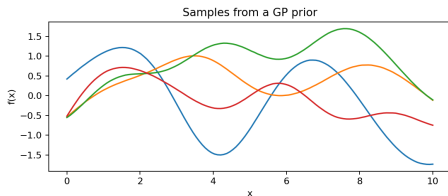
$$\mathbf{y} \sim \mathcal{N}(\mu, \Sigma),$$

其中 $\Sigma_{ij} = \text{Cov}(y_i, y_j)$ 描述不同位置之間的關聯。

Gaussian Process (GP) 是「function 上的機率分布」：

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')).$$

任取有限個輸入 $x_1, \dots, x_m$ ，其函數值向量都服從 multivariate Gaussian。



中學生版直覺

不是只猜一個點，而是想像很多條「可能的曲線」；資料越多，不合理的曲線就被淘汰。

Kernel 與 GP Posterior：資料如何改變預測？

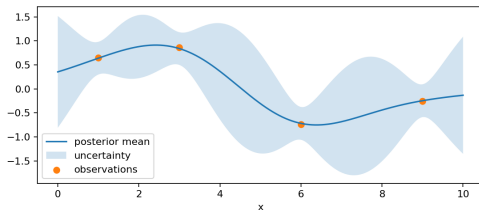
常用 RBF kernel：

$$k(x, x') = \sigma_f^2 \exp \left[-\frac{(x - x')^2}{2\ell^2} \right].$$

ℓ 控制「多遠仍算相似」。給定 observations $(X, \mathbf{y})$ 與 noise σ_n^2 ，test point x_* 的 posterior：

$$\mu_* = K_{*X}(K_{XX} + \sigma_n^2 I)^{-1} \mathbf{y},$$

$$\sigma_*^2 = K_{**} - K_{*X}(K_{XX} + \sigma_n^2 I)^{-1} K_{X*}.$$



中學生版直覺

離量過的地方近，就比較有把握；離資料很遠，就應該承認「我不確定」。

Thomas Bayes 與 Bayes' Theorem

Bayes' theorem :

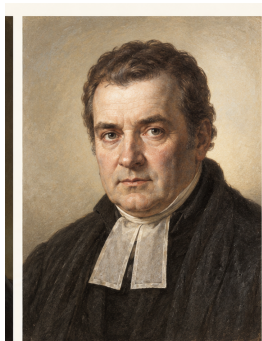
$$P(H | D) = \frac{P(D | H)P(H)}{P(D)}.$$

H : hypothesis ; D : observed data ; $P(H)$: prior ; $P(D | H)$: likelihood ; $P(H | D)$: posterior ; $P(D)$: evidence 。

Thomas Bayes (c.1701-1761) 是英國牧師與數學家；其著名機率論工作由 Richard Price 在 Bayes 過世後整理發表。常見 Bayes 肖像的歷史真實性存在爭議。

中學生版直覺

原本覺得某同學「可能很會投籃」是 prior ；看到他連進 8 球是 data ；更新後更相信他真的很會投，就是 posterior 。



Thomas Bayes

historical likeness uncertain
(c. 1701–1761)

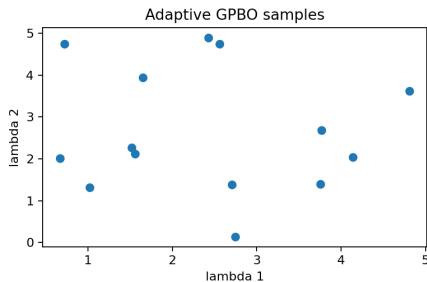
Bayesian Optimization (BO)：用少量昂貴實驗找最佳參數

BO 適合 objective $F(\lambda)$ 每次 evaluation 很貴、沒有解析 gradient、甚至有 noise 的情況。

- 1 選少量初始 λ 並真正求解 QUBO。
- 2 用資料 fit GP surrogate。
- 3 GP 給每點 $\mu(\lambda)$ 與 $\sigma(\lambda)$ 。
- 4 acquisition function 決定下一個最值得測的點。
- 5 真正 evaluate，加入資料，再更新 GP。

中學生版直覺

GP 是「地圖繪製員」，BO 是「探險隊長」：地圖同時標示預測與未知區，隊長決定下一站。



Exploration vs. Exploitation 與 Acquisition Function

Exploitation：去 GP 預測目前最好的地方；**Exploration**：去 uncertainty 大、可能藏有更好解的地方。Acquisition function $a(\lambda)$ 把「預測好不好」與「不確定性值不值得探索」合成一個便宜分數：

$$\lambda_{t+1} = \arg \max_{\lambda} a_t(\lambda)$$

(若採 minimization 形式，也可改寫成 argmin)。Acquisition 本身很便宜，所以可以密集搜尋；真正昂貴的 QUBO evaluation 只做在選中的點。

中學生版直覺

選餐廳：一直去目前最高分的店是 exploitation；偶爾試沒吃過但很有潛力的新店是 exploration。

Expected Improvement (EI)

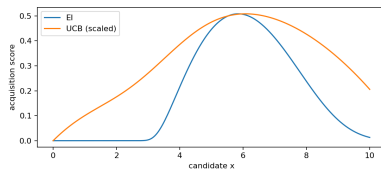
以 maximization 為例，現有最佳值 f^+ ，GP 在候選 x 預測 $Y(x) \sim \mathcal{N}(\mu(x), \sigma^2(x))$ 。Improvement：

$$I(x) = \max(0, Y(x) - f^+ - \xi).$$

Expected Improvement：

$$EI(x) = \mathbb{E}[I(x)] = (\mu - f^+ - \xi)\Phi(Z) + \sigma\phi(Z),$$
$$Z = \frac{\mu - f^+ - \xi}{\sigma}.$$

第一項偏向高 μ (exploitation)；第二項會獎勵較大 σ (exploration)。 ξ 可控制探索傾向。



中學生版直覺

跳遠選手：不只看平均能跳多遠，也看「有沒有機會破目前紀錄」。EI 就是在算預期能進步多少。

Upper Confidence Bound (UCB)

Maximization 常用

$$UCB(x) = \mu(x) + \kappa\sigma(x), \quad x_{t+1} = \arg \max_x UCB(x).$$

Minimization 可用 lower confidence bound

$$LCB(x) = \mu(x) - \kappa\sigma(x), \quad x_{t+1} = \arg \min_x LCB(x).$$

κ 小：偏 exploitation； κ 大：偏 exploration。

EI vs. UCB：EI 問「預期能比目前最佳改善多少？」；UCB/LCB 問「把 uncertainty 的潛力也算進去後，哪裡最值得試？」

中學生版直覺

球探看球員：平均表現是 μ ，尚未完全展現的潛力是 σ ； κ 越大，球探越願意押寶未知潛力。

Single- λ GPBO：先從一個 Penalty Weight 學起

外層問題：

$$\lambda^* = \arg \min_{\lambda > 0} F(\lambda),$$

其中一次 $F(\lambda)$ evaluation 代表：建立 $E(x; \lambda) = f(x) + \lambda P(x)$ 、呼叫 solver、多次 sampling、計算 feasibility 與 solution quality。

流程：初始試 $\lambda = 0.1, 1, 10 \rightarrow$ fit GP $\rightarrow$ 用 EI/LCB 選下一個 $\lambda \rightarrow$ 真正求解 $\rightarrow$ 更新 GP $\rightarrow$ 重複。

注意兩層變數： x 是內層 QUBO decision variables； λ 是外層 hyperparameter。GP 學的是 $\lambda \mapsto F(\lambda)$ ，不是直接預測最佳 x 。

中學生版直覺

像調烤箱溫度：每烤一次蛋糕都很花時間，所以不要把 100 個溫度全部試完；先試幾個，再用結果猜下一個最值得試的溫度。

從 Single- λ 延伸到雙 Penalty Weights

$$E(x; \lambda_1, \lambda_2) = f(x) + \lambda_1 P_1(x) + \lambda_2 P_2(x),$$

外層 GP 學習

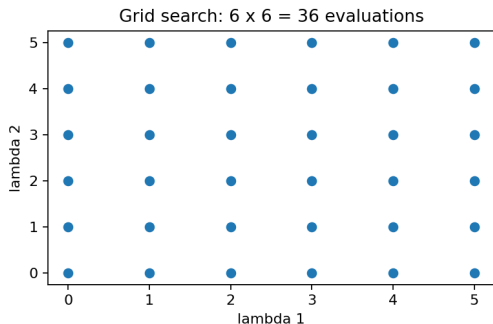
$$F(\lambda_1, \lambda_2) \sim \mathcal{GP}(m(\lambda), k(\lambda, \lambda')).$$

每次 acquisition 從二維平面選下一組 (λ_1, λ_2) 。若 weights 跨多個 orders of magnitude，通常可令 $z_i = \log_{10} \lambda_i$ 後再做 GPBO。

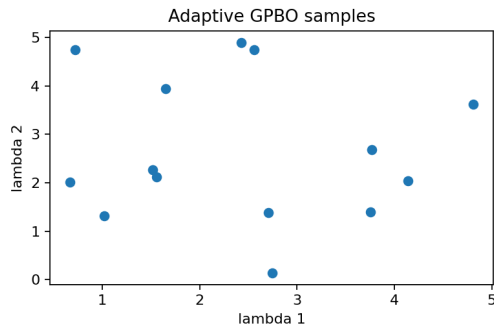
中學生版直覺

一個旋鈕變成兩個旋鈕：例如兩種不同規則各有自己的「違規增加成本」。BO 不是把所有組合試完，而是逐步鎖定有希望的區域。

Grid Search vs. GPBO：為何 GPBO 更省 Samples？



Grid：規則地測所有位置



BO：根據已知資料自適應選下一點

若每軸 6 點，grid 需 36 次。GPBO 可先用少量 initial samples，再做約 8-12 次 adaptive evaluations；例如總計約 12-16 次就可能找到優良區域。這是 **sample-efficiency** 的典型優勢，不是保證 GPBO 每次都勝過 grid search。

GPBO 的外層 Objective 如何設計？

不能只追求低 QUBO energy；更常見的是把 feasibility 與 solution quality 合在一起。例如 minimization：

$$F(\lambda) = \bar{f}_{\text{feasible}}(\lambda) + \rho [1 - r_{\text{feasible}}(\lambda)],$$

其中 $r_{\text{feasible}} \in [0, 1]$ 是多次 solver runs 中 feasible solutions 的比例， $\rho > 0$ 控制違規懲罰。也可加入 runtime / stability。

因 annealer / heuristic solver 有 stochastic noise，同一 λ 可重複多次估計 mean 與 variance；GP 的 σ_n^2 可表示 observation noise。

中學生版直覺

選球員不能只看一次最高得分；也要看「穩不穩、犯規多不多、平均表現如何」。

Practical Recipe : QUBO Penalty Weight GPBO

- 1 寫出 $f(x)$ 與每個 $P_r(x)$ ，先檢查 $P_r = 0$ 是否真的等價於 constraint satisfied。
- 2 用 objective-range bound 或 incremental tuning 找合理 weight 範圍。
- 3 對 λ_r 取 log scale (若跨 orders of magnitude)。
- 4 用少量 space-filling initial samples 啟動 GP。
- 5 Fit GP，使用 EI 或 UCB/LCB 選下一組 weights。
- 6 每組 weights 重複 solver runs，計算 feasibility、quality、noise。
- 7 到 evaluation budget、改善趨緩或達標後停止，最後以獨立 runs 驗證最佳 weights。

中學生版直覺

先知道合理價格範圍，再聰明試價；不是從 1 元到 1 萬元每一元都試一次。

Take-Home Messages

- 1 QUBO 用 penalty term 把 constraints 搬進 objective ; λ 決定「違反規則會增加多少成本」。
- 2 靜態方法簡單、便宜，適合先取得合理範圍；多 constraints 時通常需要多個 weights 。
- 3 GP 不只預測 mean，也預測 uncertainty；kernel 定義「哪些參數點應該相似」。
- 4 BO 用 acquisition function 在 exploitation 與 exploration 間取捨；EI 與 UCB/LCB 是兩個代表方法。
- 5 GPBO 特別適合「每次 QUBO/annealing evaluation 很昂貴」的 penalty tuning，可比 dense grid 更 sample-efficient 。

References: Glover, Kochenberger, Du (2019), arXiv:1811.11538; Kochenberger et al. (2014), J. Combinatorial Optimization; Rasmussen & Williams (2006), *Gaussian Processes for Machine Learning*; Jones, Schonlau, Welch (1998), J. Global Optimization; Srinivas et al. (2010), ICML.