arXiv:2603.29986v1 [q-bio.QM] 31 Mar 2026

ParetoEnsembles.jl：使用帕累托最优集成技术进行多目标参数估计的 Julia 软件包

Jeffrey D. Varner[1]

[1]罗伯特·弗雷德里克·史密斯化学与生物分子工程学院，康奈尔大学，伊萨卡，纽约州 14853，美国

摘要

自然系统和人造系统的数学模型通常包含许多可调参数，必须依据多个可能相互冲突的数据集进行估计。与其报告单个最佳拟合参数向量，生成一组参数集构成的集成通常更有助益，因为它们能够共同描绘竞争目标之间的权衡。本文介绍 ParetoEnsembles.jl，这是一款开源 Julia 软件包，采用帕累托最优集成技术（POETs）生成此类集成。该技术是一种基于模拟退火的算法，无需梯度信息。此实现将原始支配关系从弱帕累托支配修正为严格帕累托支配，通过增量更新方案将每次迭代的排序成本从 $O(n^2m)$降至$O(nm)$，并加入多链并行执行以提高前沿覆盖率。我们将该软件包应用于一个根据实验数据拟合的无细胞基因表达模型，以及一个包含十个估计速率常数和三个目标函数的血液凝固级联模型。受控合成数据研究揭示了参数可辨识性结构：各个速率常数的估计值可相差数倍，但模型预测仍准确至 6–7%。五次重复的覆盖分析证实，时间特征能够得到可靠覆盖，而峰值振幅则存在系统性过度置信。通过已发表的实验性凝血酶生成数据进行验证表明，尽管存在固有的模型近似误差，该集成对留出条件的预测误差仍在 1–10% 以内。ParetoEnsembles.jl 旨在让集成生成更加轻量且易于使用，从而降低在机制建模中常规进行不确定性表征的门槛。

关键词：多目标优化，帕累托前沿，模拟退火，集成方法，参数估计，不确定性表征

1 引言

生物系统和工程系统的数学模型通常包含许多可调参数，必须根据实验数据进行估计。当多个数据集测量同一系统的不同方面（例如，基因表达电路中的信使RNA和蛋白质浓度，或反应器中的产率和选择性）时，将参数拟合到一个数据集往往会降低对另一个数据集的拟合效果。这便引出了一个多目标优化（MOO）问题 [1–3]。更棘手的是，许多模型表现出“松散”的参数敏感性：目标空间中存在宽阔平坦的谷地，在这些区域里，大范围的参数空间都能产生几乎等效的拟合结果 [4]。在这种情况下，单个最佳拟合参数向量无法表明数据实际上对参数的约束有多严格；相反，我们需要一组参数集构成的集成，共同描绘竞争目标之间的权衡关系。这一集成在目标空间中的边界就是帕累托前沿，即无法在不使至少一个其他目标变差的情况下改善任何目标的解集；而围绕该前沿的近似最优解整体分布，则刻画了与现有数据相符的模型行为范围。

目前已有多种算法家族被开发出来，用于近似帕累托前沿。基于种群的进化方法会维护一代候选解，并通过选择、重组和变异使其逐渐逼近前沿。典型例子包括非支配排序遗传算法 II（NSGA-II） [5], 、其面向多目标的后继算法 NSGA-III [6], ，以及基于分解的多目标进化算法（MOEA/D）；后者将多目标优化问题分解为一组由协同方式求解的标量子问题 [7]。另一种基于轨迹的策略是多目标模拟退火（SA），它通过利用帕累托支配来指导接受决策，将经典 SA 框架 [8] 扩展到多个目标；早期方法包括 Czy˙zak 和 Jaszkiewicz [9] 提出的帕累托模拟退火方法，以及 Bandyopadhyay 等人提出的存档式多目标模拟退火（AMOSA）算法 [10]。Song 等人 [11] 提出、后来由 Bassen 及其同事 [12], 使用 Julia编程语言实现的帕累托最优集成技术（POETs），也属于这一基于模拟退火的算法家族。POETs算法通过统计其他解支配每个候选解的次数，对当前档案集进行排序，然后将秩用作 Metropolis 型接受准则中的能量。多种语言中都有成熟的软件实现可供使用，包括 Python 中的 pymoo [13]和 Julia 中的 Metaheuristics.jl [14], ，这两者都主要侧重于基于种群的进化策略。刻画参数不确定性的另一种方法是贝叶斯推断：给定数据后，使用马尔可夫链蒙特卡罗（MCMC）或序贯蒙特卡罗（SMC）方法从参数的后验分布中采样。常用的实现包括 Stan [15], 、Turing.jl [16], 和集成 MCMC 采样器 emcee [17]。贝叶斯方法虽然能为参数不确定性提供正式的概率解释，但需要指定先验分布和标量似然，难以扩展到多个目标，而且在刚性常微分方程模型上计算开销可能很大。另一方面，POETs 等多目标集成方法提供了一种轻量级替代方案，能够自然地处理多个竞争目标，并在无需概率建模假设的情况下返回一组近似最优解。

在本研究中，我们介绍 ParetoEnsembles.jl，这是对 POETs 算法 [11, 12] 的 Julia 实现进行大幅重写后的版本。该软件包接受用户定义的模型和一组

目标函数，每个目标函数都用于衡量候选参数向量对特定数据集的复现效果；随后返回一组覆盖权衡曲面的参数集，以及它们的目标值和支配等级。相较于原始实现，我们对算法进行了若干修正和改进：将支配性检查从弱帕累托支配修正为严格帕累托支配；通过增量更新方案，将每次迭代的排序成本从 $O(n^2m)$ 降至 $O(nm)$（其中 n 是档案集大小， m 是目标函数的数量）；并加入多链并行执行，以提高前沿覆盖率。我们在两个复杂度递增的生物系统上展示了该软件包（第3节），并通过标准基准测试和 NSGA-II 对其进行验证，同时评估超参数敏感性（S4至S6节）。通过让集成生成变得轻量且易于使用，该软件包降低了在机制建模中常规开展不确定性表征的门槛。

2 方法

2.1 帕累托支配与排序

考虑一个档案集，其中包含 n 个解，每个解在 m个目标函数上进行评估，且数值越低越好。当且仅当对于所有 $k \in \{1, \ldots, m\}$， $f_k(\mathbf{x}_j) \leq f_k(\mathbf{x}_i)$，并且至少存在一个目标函数 k，使得 $f_k(\mathbf{x}_j) < f_k(\mathbf{x}_i)$（公式 (1)）时，解$\mathbf{x}_j$ 严格支配解$\mathbf{x}_i$，记作$\mathbf{x}_j \prec \mathbf{x}_i$。解$\mathbf{x}_i$ 的帕累托等级是严格支配它的档案成员数量（公式 (2)），因此等级为零的解位于帕累托前沿；而相同的解在严格支配下不会相互支配，因此具有相同的等级。原始 POETs 实现 [11, 12] 采用了弱支配，只检查对于所有 k， $f_k(\mathbf{x}_j) \leq f_k(\mathbf{x}_i)$，而不要求至少一个目标函数上的不等式为严格不等式；在弱支配下，相同的解会被视为彼此支配，从而抬高其等级，并导致档案集中过度剪除优质解。当前实现采用严格帕累托支配。成对检验会遍历各个目标函数，并设置提前退出条件：一旦发现 j 在任何目标函数上的表现劣于 i，就立即终止检验，从而避免不必要的比较（伪代码见算法 S2）。

$$f_k(\mathbf{x}_j) \leq f_k(\mathbf{x}_i) \quad \forall\, k \in \{1, \ldots, m\} \qquad \text{and} \qquad \exists\, k : f_k(\mathbf{x}_j) < f_k(\mathbf{x}_i). \tag{1}$$

$$R(\mathbf{x}_i) = \left| \{\, j \neq i \mid \mathbf{x}_j \prec \mathbf{x}_i \,\} \right|. \tag{2}$$

2.2 增量排序

一项核心计算成本在于排序。当将单个候选解$\mathbf{x}_{\text{new}}$ 追加到包含 n 个秩已知解$\mathbf{R}$的档案集中时，naïve 方法会以 $O(n^2m)$ 的成本重新执行所有成对支配检查，其中 m 是目标函数的数量。然而，由于只有一个解发生变化，我们可以改为对现有档案集执行一次 $O(nm)$ 遍历：对于每个现有解$\mathbf{x}_i$，我们双向检查支配关系；如果新解支配$\mathbf{x}_i$ ，则增加其 R_i ，如果$\mathbf{x}_i$ 支配新解，则增加新解的秩。这一增量过程将每次迭代的成本从 $O(n^2m)$ 降至 $O(nm)$，只有在从档案集中移除

高秩解后，才会完整地重新进行一次 () 排序，其中 n' 是缩减后的档案集大小（伪代码见算法 S3 ）。

2.3 帕累托模拟退火

完整的帕累托模拟退火流程将支配性检查和增量排序整合到一个嵌套循环中，该循环遍历各温度层级并评估候选解。算法以初始参数向量 $\mathbf{x}_0$, 、初始温度 $T = 1$以及一个档案集作为起点，该档案集仅包含一个已评估的解（$\mathbf{x}_0, \mathbf{f}(\mathbf{x}_0)$）。在每个温度层级， N_{iter} 通过用户提供的邻域函数扰动当前最优参数向量，生成候选解；秩数组则以每个候选解$O(nm)$的代价进行增量更新。根据秩数组和当前温度计算出的接受概率决定是否保留候选解；一旦接受，就会移除秩达到或超过截断值 R_{cutoff} 的解，并对剩余集合重新进行完整排序。如果档案集仍超过最大容量 $n_{\max}$，则仅保留秩最低的 $n_{\max}$ 个解。若拒绝该候选解，则立即将其移除，并从已保存的快照恢复秩数组；我们将这种策略称为拒绝时弹出策略。在原始实现中，被拒绝的候选解仍留在档案集中，导致 n 在已接受的移动之间无限增长，排序操作的成本也随之不断增加；拒绝时弹出策略解决了这一问题。为提高前沿覆盖率，可以从不同起始点运行多条相互独立的模拟退火链，并合并其档案集，最后对合并后的集合重新排序。这种多链策略可实现近线性的并行加速比，并确保对权衡曲面进行多样化探索（主循环和多链封装的伪代码见算法 S1 和 S4；软件设计详情及公共 API 见 S2 节）。

3 结果

我们首先在两个标准多目标基准测试上验证了 ParetoEnsembles.jl：有约束的 Binh–Korn 问题 [18] 和无约束的 Fonseca–Fleming 问题 [19], ；两者均重现了已知的帕累托前沿（S4 节）。在评估预算相同（$\sim$110，000）的条件下，与 NSGA-II [5] 进行正面对比发现，ParetoEnsembles.jl 达到了相近的超体积（即帕累托前沿覆盖的目标空间体积，数值越大越好），反世代距离降低了 5–7$\times$ （即真实前沿到计算所得前沿的平均距离，数值越低表示近似越准确），且得到的帕累托前沿密集度提高了 10–30$\times$ ，保留的解有数千个，而 NSGA-II $\sim$200 则不然（S5 节）。超参数敏感性分析证实，该方法对秩截断值、冷却速率和每个温度的迭代次数均具有稳健性（S6 节）。因此，ParetoEnsembles.jl 对权衡曲面给出了更密集、更完整的近似，但由于模拟退火链本质上是顺序运行的，实际运行时间更长（根据问题规模，比 NSGA-II 慢 6–150$\times$）。接下来，我们聚焦于两个生物学应用，以说明基于集成的参数估计的实际价值。

3.1 无细胞基因表达

作为第一个生物学应用，我们考虑了对无细胞基因表达电路动力学参数的估计。该电路描述了 deGFP 的生成过程。deGFP 是 Shin 和 Noireaux [20],开发的一种增强型绿色荧光蛋白高翻译效率变体，由内源性 σ 因子 σ_{70} 通过 P70启动子驱动。我们采用了 Adhikari 等人提出的有效生物物理学框架 [21] （图 1）。系统由两个常微分方程描述，分别对应信使RNA（m）和蛋白质（p）；其中，$\dot{m} = \alpha\, u\,(\sigma_{70}) - \delta_m\, m$和 $\dot{p} = \kappa\, m\, w\,(t) - \delta_p\, p$，而 $u\,(\sigma_{70}) = \sigma_{70}^n/\,(K^n + \sigma_{70}^n)$表示希尔型启动子活性函数，$w\,(t) = \exp\,(-\ln 2 \cdot t/\tau_{1/2})$则刻画了无细胞反应过程中实验观察到的翻译能力衰减。五个参数被视为未知参数（最大转录速率 α、有效翻译速率常数 κ、信使RNA降解速率 δ_m、启动子解离常数 K和翻译能力半衰期 $\tau_{1/2}$），其余参数（$\sigma_{70} = 35$ nM、$n = 1.5$、$\delta_p = 0.005$ h^{-1}）则固定为文献中的取值。

我们将模型拟合到 Adhikari 等人 [21], 的实验数据，该数据包括在五个时间点（0、2、4、8 和 16 小时）以三重复测得的信使RNA和 deGFP 蛋白质浓度，并报告了均值和标准差。我们定义了两个目标函数，即以标准差加权的平方误差和：$\varepsilon_{\mathrm{mRNA}} = \sum_i[(m_i^{\mathrm{sim}} - m_i^{\mathrm{obs}})/\sigma_i^m]^2$ 和$\varepsilon_{\mathrm{protein}} = \sum_i[(p_i^{\mathrm{sim}} - p_i^{\mathrm{obs}})/\sigma_i^p]^2$；随后运行了十条并行链，秩截断值为 $R_{\mathrm{cutoff}} = 8$，每个温度下有$N_{\mathrm{iter}} = 50$ 个候选解，冷却速率为 $\alpha = 0.90$。初始参数向量从原始研究 [21]指定的参数边界内均匀随机抽取。所得集成捕捉了信使RNA和蛋白质表达的动态变化（图 1a、b）：集合模拟的 95%置信区间涵盖了所有时间点的实验数据，且集成均值与观测到的轨迹吻合。帕累托前沿显示，拟合信使RNA和蛋白质时间过程之间存在明显的权衡（图 1c）：使 $\varepsilon_{\mathrm{mRNA}}$ 最小化的参数集往往具有较大的 $\varepsilon_{\mathrm{protein}}$，反之亦然。这种权衡自然源于信使RNA降解速率和翻译能力半衰期对两种物种的影响方向相反。该集成提供了一系列在这些竞争目标之间进行权衡的合理参数集，而不是一个偏向其中某个目标的点估计。即使对于这个小型系统，集成也揭示了估计参数之间的权衡结构；接下来，我们将探讨该方法能否扩展到规模大得多、且与临床更为相关的模型。

3.2 凝血级联

为了展示该软件包在更大规模系统中的应用，我们研究了组织因子启动的凝血 Hockin–Mann模型 [22] （图 2）。该机理常微分方程模型描述了 34 种化学物种通过 27 个反应相互作用，这些反应由 42 个速率常数控制，涵盖凝血酶生成的起始、放大和抑制阶段。所有速率常数均见于文献，并用作真实值。我们选取了十个关键的催化和抑制速率常数进行估计，涵盖外源性因子X酶（k_{cat}）、内源性因子Xase（k_{cat}）、凝血酶原酶（k_{cat}）以及抗凝血酶III抑制途径。由于这些速率常数跨越六个数量级（从 ~ 1 到 $\sim 10^7$），

我们在对数空间中进行处理，并将边界设为文献值上下 ± 1.5个数量级。我们通过在三种组织因子浓度（5、15 和 25 皮摩尔）下使用真实速率常数模拟该模型，并向每种条件下的凝血酶总轨迹添加乘性高斯噪声（$x' = x \cdot (1 + \epsilon)$、 $\epsilon \sim \mathcal{N}(0,\ 0。15_2)$），生成带噪声的合成凝血酶生成试验（TGA）数据。我们定义了三个目标函数，即三种组织因子浓度下凝血酶轨迹的归一化平方误差和，不含正则化项；参数恢复仅反映训练数据能够约束的内容，而不借助将估计值拉向已知真实值的惩罚项。

我们运行了八条并行链，每个温度下的秩截断值设为 $R_{\text{cutoff}} = 10$, $N_{\text{iter}} = 40$ 个候选解，冷却速率为 $\alpha = 0.92$。所得的集成包含 222 个参数集（秩 ≤ 1），生成的凝血酶轨迹与三种组织因子浓度下的含噪训练数据都高度吻合（图 2a），且整个时间过程中，95%置信区间均涵盖了数据。然而，参数恢复的效果参差不齐，揭示了该系统在实际中的可辨识性结构（图 2b）。有些速率常数得到了很好的恢复（IIa因子+抗凝血酶III抑制速率的误差在 2% 以内，凝血因子Va激活速率的误差在 5% 以内），而仅凭凝血酶数据无法充分约束另一些参数：内源性因子 Xase k_{cat} 的估计值偏差达五倍，直接由 Xa因子介导的凝血酶生成速率（$k_{\text{Xa}\to\text{IIa}}$）则被高估了两倍以上。三维帕累托前沿（图 2c）展现了拟合三种组织因子条件时预期的权衡，表明 ParetoEnsembles.jl 能够处理具有中等复杂度的系统，这类系统包含数十种物种、十个或更多估计参数以及三个竞争目标。尽管若干参数的恢复效果不佳，模型仍然很好地拟合了训练数据：这与这样一种情况相符，即速率常数之间的补偿性调整，使得差异极大的参数向量也能生成几乎相同的凝血酶轨迹。这促使我们开展下文所述的基于集成的下游研究。

3.3 基于集成的预测和可辨识性分析

与单个点估计相比，生成参数集成的一项关键优势是能够将不确定性向前传播至模型中，从而获得预测区间；其宽度反映了训练数据能够和无法约束的内容（图3）。为检验这一点，我们使用仅基于 5、15 和 25 pM TF 条件下的 TGA 曲线训练的凝血集成，预测三个留出的组织因子浓度（10、20 和 30 pM）下完整的凝血酶时间过程；这些浓度在优化过程中从未使用过。在这三种条件下，集成均值都能追踪真实轨迹，预测的凝血酶峰值与真实值相差不超过 6–7%，且 95%预测区间涵盖了凝血酶生成曲线的整体形状（图3a）。然而，在峰值处，预测带未能完全覆盖真实轨迹，揭示了峰值振幅存在系统性正偏差，而集成的不确定性未能将其纳入考量。集成通过提供宽度随输出和条件变化的预测区间，使这一系统性偏差显现出来。为进一步量化这一点，我们从每个留出条件下的集成预测和真实轨迹中提取了三个临床相关的凝血酶生成试验特征（滞后时间、凝血酶峰值浓度和内源性凝血酶潜能（ETP））（图3d）。滞后时间的预测效果良好，95%区间宽度适当，且在所有三种条件下均涵盖真实值

种浓度下，峰值凝血酶和内源性凝血酶潜能（ETP）均表现出一致的 4–7% 高估，且区间过窄，无法达到覆盖率；这表明，尽管集成能够正确捕捉凝血反应的时间特征，但对凝血酶振幅的估计过于自信。

集成中的两两相关性揭示了模型的可辨识性结构（图3b）。最强的负相关出现在外源性因子X酶 k_{cat}与凝血酶原酶 k_{cat} ($r = -0.81$)之间，这表明存在补偿行为：当起始阶段的酶活性更快时，放大阶段的酶必须更慢，才能产生相同的凝血酶轨迹。Xa直接介导的凝血酶生成速率与凝血因子Va激活 ($r = -0.93$)之间也出现了类似的补偿耦合，反映了直接通路与凝血酶原酶介导通路之间的竞争；TF=VIIa→IX催化速率与mIIa→IIa转化速率 ($r = -0.73$)之间也存在类似关系，连接了起始与传播。这些关系解释了为何个别参数即使相差数倍，模型预测仍能保持在几个百分点以内的准确度：一个速率常数的相关偏移会被其他速率常数的补偿性变化抵消，因此，尽管单个参数差异很大，集成仍覆盖了参数空间中的一个区域，在该区域内可产生相似的轨迹。

最后，我们使用集成模拟了一个具有临床相关性的扰动：不同严重程度的凝血因子VIII缺乏（血友病A）。在将初始凝血因子VIII浓度降至名义血浆水平的30%（轻度血友病）和5%（重度血友病），同时保持所有估计速率常数为其集成值的情况下，我们生成了虚拟患者的凝血酶生成预测，结果捕捉到了预期的、具有剂量依赖性的凝血酶峰值下降及其相关不确定性（图3c）。凝血酶峰值从正常血浆中的 684 ± 22 nM降至轻度血友病的 563 ± 22 nM和重度血友病的 401 ± 21 nM，且在这三种条件下，真实轨迹均落在集成预测带内。随着凝血因子VIII水平降低，预测区间逐渐变宽，这与以下预期一致：当系统进一步偏离用于训练的条件时，模型受到的约束会减弱。尽管若干单独速率常数的恢复效果较差，集成仍正确包络了这些分布外扰动下的真实轨迹，这表明，在模型预测用于支持临床决策的转化研究场景中，基于集成的不确定性表征可能具有实用价值。综上，这些分析表明，集成揭示了可辨识性结构，将不确定性传播至留出条件，并支持具有临床相关性的扰动研究；但所有分析均依赖于合成数据，在这种数据中，模型按构造即为正确模型。为检验模型误差下的稳健性，我们将六个固定速率常数扰动了 $\pm 30\%$，并重复了凝血研究；尽管轨迹级覆盖率降至31–48%，集成对留出数据中的峰值凝血酶的预测仍处于真实值的4–6%以内（S7节）。接下来，我们将检验当模型仅为现实的近似时，这些发现是否仍然成立。

3.4 与实验数据进行验证

上述凝血分析使用了真实值已知的合成数据，从而能够系统地评估参数恢复和覆盖率。然而，这些测试并未回答一个问题：当模型只是对现实的近似时，集成方法是否仍然有效。为此，我们将Hockin–Mann模型拟合到Butenas等人发表的实验数据，他们测量了重构合成血浆系统中的凝血酶生成 [23], ，该系统的凝血酶原浓度为：

50%至150%的平均血浆值，共设五种浓度，其他所有凝血因子均处于标称浓度；时间序列数据取自图3 [23], ，并依据表2校准峰值。该系统是合适的测试案例，因为其初始条件经过了充分表征（表1 [23]），不存在蛋白C通路（与Hockin–Mann模型相符），且凝血酶原浓度范围涵盖了临床相关的个体间变异来源（图4）。

我们将三个训练目标定义为模拟与实验凝血酶时间曲线之间的归一化平方误差和（SSE），分别对应正常凝血酶原水平的50%、100%和150%。我们将75%和125%条件留作验证。使用与合成研究相同的超参数运行了八条并行链，得到包含390个参数集的集成（秩 ≤ 1）。该集成捕捉到了训练条件下的实验凝血酶曲线（图4a），在三个凝血酶原水平下，对凝血酶峰值的预测误差均为0.1–10%；残余差异反映了模型与数据之间真实存在的张力：Hockin–Mann模型对凝血酶原浓度的敏感性与实验系统不符，这种不匹配在帕累托前沿的权衡结构中清晰可见（图4c）。在留出的验证条件下，集成对75%和125%凝血酶原水平下凝血酶峰值的预测与实验值相差不超过1–10%（图4b）；集成均值紧密追踪数据，且95%预测区间涵盖了这两个条件下的实验时间点。估计的速率常数偏离了文献标称值（图4d），这符合拟合真实数据时的预期，因为模型结构只是对底层生物化学过程的近似；这些偏差反映了对估计参数的补偿性调整，以弥补模型与实验系统之间的差异。集成中各参数的离散程度表明数据对相应速率常数的约束有多严格。这些结果证实，集成方法并不局限于合成环境：即使模型无法完美表征底层系统，该方法仍能给出准确的留出预测，并提供有参考价值的不确定性带。

4 讨论

我们介绍了 ParetoEnsembles.jl，这是一个轻量级的 Julia 实现，用于多目标优化的帕累托最优集成技术（POETs），解决了原始实现中的若干局限 [11, 12]。当前版本将支配关系从弱支配修正为严格支配，通过增量更新将每次迭代的排序成本从 $O(n^2m)$ 降至 $O(nm)$，采用拒绝时弹出策略和硬性大小上限来限制档案集增长，并加入了多线程排序、多链并行执行、内置超体积计算和收敛诊断。基于回调的设计将优化算法与问题定义解耦，使用户能够将 ParetoEnsembles.jl 应用于任何可逐点评估目标函数的领域。在标准基准问题上，ParetoEnsembles.jl 在评估预算相同的情况下，生成的帕累托前沿在超体积和反世代距离方面可与 NSGA-II 相媲美，同时保留的集成密度是 NSGA-II 所返回结果的 10–$30\times$ 倍（S5 节）。这种密度在实践中十分重要：要从集成中有效计算预测

区间、参数相关性和覆盖率统计，需要数百至数千个成员填满参数空间中的近似最优区域，而不仅是基于种群的方法通常得到的 100–200 个非支配解。虽然 NSGA-II 的运行时间明显更短，这反映了 SA 链本质上按顺序执行，而进化方法采用种群并行评估，但 ParetoEnsembles.jl 面向的是那些目标在于刻画近似最优参数集的云团 而不仅仅是帕累托前沿本身的问题。此外，简洁的接口降低了在建模工作流中生成参数集成的门槛。

此处展示的生物学应用凸显了基于集成的参数估计为何不同于传统的单点优化。无细胞基因表达示例表明，即便是只有五个参数和两个目标函数的小型模型，在拟合真实实验数据时也能揭示有意义的权衡；凝血级联示例则说明，该方法可扩展到更大型的系统，其中包含十个参数，跨越六个数量级，并涉及三个相互竞争的目标函数。凝血研究的一项核心发现是，准确的模型预测并不需要准确的参数恢复。尽管各个速率常数的偏差最高可达五倍，该集成仍能将留出数据中不同组织因子浓度下的凝血酶生成预测到真实峰值的 6–7% 以内，这是因为两两参数相关性揭示了广泛存在的补偿结构。两两参数相关性表明，许多不同的参数组合都能产生几乎相同的凝血酶轨迹。留出验证还揭示了一项局限：虽然集成正确预测了凝血酶生成的时间（所有条件下的滞后时间均被覆盖），但峰值振幅和内源性凝血酶潜能均表现出系统性正偏差，且置信区间过窄，无法达到名义覆盖率。这表明，要充分约束凝血酶爆发的振幅，还需要额外的数据类型。这项受控实验验证了该算法探索权衡曲面的能力，并为评估覆盖率和可辨识性提供了真实值基准，但并未检验其对模型设定错误的稳健性。为此，我们将六个固定速率常数扰动 $\pm 30\%$，使拟合模型出现设定错误，并重复了该研究（S7节）；集成对留出数据中的凝血酶峰值的预测仍在 4–6% 以内，但轨迹级覆盖率降至 31–48%，这证实该方法能够稳健应对中等程度的结构误差，同时也揭示了仅靠参数不确定性无法解释模型不足。血友病A模拟表明，集成能够将参数不确定性传递到临床相关扰动中；随着系统逐渐偏离训练条件，预测区间会随之变宽，即使底层参数单独来看难以准确辨识，也能正确涵盖真实轨迹。这些结果为 ParetoEnsembles.jl 的实际应用提供了范例。超参数敏感性分析（S6 节）进一步表明，该算法对秩截断值、冷却速率和每个温度的迭代次数选择均具有稳健性，因此用户无需进行大量调参即可采用默认设置；此外，收敛轨迹功能还提供了内置诊断工具，用于监测退火进度。

ParetoEnsembles.jl 生成的集成并非贝叶斯后验分布。该集成由相对于多个目标函数近似最优的参数向量组成，这些参数向量通过带有基于秩的接受准则的模拟退火选出；它并不对应于在给定数据条件下从某个概率分布中抽取的样本。本文通篇报告的 95%区间是集成预测的经验分位数，

而非贝叶斯可信区间。该集成并不声称不同参数向量具有不同的相对概率，只表明它们都能产生质量相当的拟合；其覆盖特性取决于模拟退火链对近最优区域的探索是否充分，而非是否收敛至平稳分布。MCMC [15, 16] 或集成马尔可夫链蒙特卡罗 [17] 等贝叶斯方法为参数不确定性提供了严谨的概率框架，但这类方法需要指定先验分布和标量似然函数；当必须同时平衡多个不可通约的目标时，标量似然函数并不容易定义。此外，这些方法还面临多峰后验分布和刚性常微分方程似然等众所周知的难题，可能导致收敛缓慢或不可靠。帕累托集成方法规避了这些困难，但代价是失去概率解释性；需要经过校准的后验可信区间的用户应考虑将贝叶斯方法作为补充分析。一个有望衔接这两种框架的方向是吉布斯后验：在这种方法中，本文使用的基于损失的目标函数取代对数似然，进行广义贝叶斯更新；对集成成员相对于此类后验进行重要性重加权，或可在重复利用优化过程中已经完成的高成本模型评估的同时，得到经过校准的可信区间。为验证覆盖模式具有系统性，而非单次噪声实现造成的假象，我们使用相互独立的噪声种子，将合成凝血研究重复了五次，并计算了各次重复实验的特征级覆盖率。结果证实了一个清晰且可重复的模式：滞后时间和达峰时间的覆盖率均为 100%（5/5 次重复实验，所有留出条件下均如此）；而凝血酶峰值仅在 40% 的重复实验中得到覆盖，ETP 则为 60%，各条件下的轨迹级覆盖率平均为 87–90%。留出峰值误差的均值为 7.1 ± 5.0%（范围为 0.9–17.9%），证实了上文报告的单次重复实验结果具有代表性。这一系统性模式表现为时间指标校准良好，而振幅指标过度自信；这可能反映了集成沿相关性分析所识别的补偿结构集中分布：模拟退火链能高效探索参数空间中目标值会发生变化的方向（即沿帕累托前沿的方向），但可能未能充分探索与前沿正交的方向；在这些方向上，目标值几乎不变，但模型预测却有所不同，从而导致某些输出的预测区间校准良好，而另一些输出的预测区间则过度自信。我们还验证了下游预测对纳入集成的档案成员选择并不敏感：将秩截断值从 0（仅帕累托前沿）逐步提高至 rank≤ 5，留出峰值误差的均值变化不到 0.2 个百分点。这证实了前沿周围的近似最优解与前沿本身携带的预测信息基本相同。

当前范围有意省略了若干功能。该软件包未实现基于梯度的精化、代理建模或自适应冷却计划，也不提供内置可视化功能或基准问题库。未来的发展方向包括：与 Julia 的自动微分生态系统（例如 ForwardDiff.jl 或 Enzyme.jl）集成，以便针对部分目标函数有梯度、但并非所有目标函数都有梯度的问题，采用无梯度／基于梯度的混合策略；使用超体积指标 [24] 替代简单的基于排名的剪枝，以自适应地管理档案集，更合理地控制保留解的分布；以及通过前文讨论的吉布斯后验框架，与贝叶斯推断建立正式联系。尽管存在这些局限，当前版本仍提供了一种简洁易用的工具，可用于生成帕累托最优参数集成

；我们预计，这一工具将有助于系统生物学及其他领域的研究人员刻画机理模型中的不确定性。

5 结论

本研究介绍了 ParetoEnsembles.jl，这是一个 Julia 软件包，利用模拟退火和增量支配排序生成帕累托最优参数集成。该软件包修正了原始 POETs 实现中长期存在的弱支配问题，将每次迭代的排序成本从随档案集大小呈二次增长降至线性增长，并新增多链并行执行，从而通过多样化的起始点改善前沿覆盖。我们将该软件包应用于复杂度逐步提高的两个生物系统：一个根据实验信使RNA和蛋白质时间序列数据拟合的无细胞基因表达回路，以及包含34种物种、估计十个跨越六个数量级的速率常数的 Hockin–Mann 血液凝固级联模型。一项受控合成数据研究表明，准确的模型预测并不要求准确恢复参数：尽管个别参数相差数倍，集成对留出凝血酶生成的预测仍在6–7%以内。可辨识性分析、血友病A患者特异性模拟以及模型设定错误研究（S7节）共同表明，基于集成的不确定性表征能够反映数据可以和不可以约束的内容。我们通过 Butenas等人发表的实验性凝血酶生成数据 [23]进行验证，确认了该方法可推广至合成环境之外：基于三个凝血酶原水平训练的集成对留出条件的预测误差在1–10%以内，尽管该模型仅仅是对底层实验系统的近似。ParetoEnsembles.jl 是开源软件，已在 Julia General 注册表中注册，且只需一条命令即可安装；我们预计，其精简的接口和零依赖设计将使其成为系统生物学及其他领域研究人员生成机理模型参数集成的实用工具。

致谢

J.D.V. 感谢美国国立卫生研究院（NIH）下属美国国家心肺血液研究所（NHLBI）通过资助项目 R33 HL141787 和 R01 HL71944 提供支持。

作者贡献

J.D.V. 构思了这项研究，开发了软件，完成了所有分析，并撰写了稿件。

利益冲突

作者声明不存在任何利益冲突。

数据和代码可用性

ParetoEnsembles.jl 采用 MIT 许可证开源。源代码可在 `https://github.com/varnerlab/ParetoEnsembles.jl`获取，文档托管于 `https://varnerlab.github.io/ParetoEnsembles.jl/dev/`。该软件包已在 Julia General 注册表中注册，可通过 `Pkg.add("ParetoEnsembles")`安装。复现本文所有结果的示例代码位于源代码仓库的 `paper/code` 目录中。

参考文献

[1] Kaisa Miettinen。《非线性多目标优化》。Springer，1999年。[2] Kalyanmoy Deb。《使用进化算法的多目标优化》。John Wiley & Sons，2001年。[3] Carlos A. Coello Coello、Gary B. Lamont 和 David A. Van Veldhuizen。《求解多目标问题的进化算法》。Springer，第二版，2007年。[4] Ryan N. Gutenkunst、Joshua J. Waterfall、Fergal P. Casey、Kevin S. Brown、Christo- pher R. Myers 和 James P. Sethna。系统生物学模型中的普遍松散参数敏感性。《PLoS Computational Biology》，3(10):e189，2007年。[5] Kalyanmoy Deb、Amrit Pratap、Sameer Agarwal 和 T. Meyarivan。一种快速的精英多目标遗传算法：NSGA-II。《IEEE进化计算汇刊》，6(2):182–197，2002年。[6] Kalyanmoy Deb 和 Himanshu Jain。一种采用基于参考点的非支配排序方法的进化多目标优化算法，第一部分：求解具有盒约束的问题。《IEEE进化计算汇刊》，18(4):577–601，2014年。[7] Qingfu Zhang 和 Hui Li。MOEA/D：一种基于分解的多目标进化算法。《IEEE进化计算汇刊》，11(6):712–731，2007年。[8] Scott Kirkpatrick、C. Daniel Gelatt 和 Mario P. Vecchi。利用模拟退火进行优化。《Science》，220(4598):671–680，1983年。[9] Piotr Czy˙zak 和 Andrzej Jaszkiewicz。帕累托模拟退火——一种用于多目标组合优化的元启发式技术。《Journal of Multi-Criteria Decision Analysis》，7(1):34–47，1998年。[10] Sanghamitra Bandyopadhyay、Sriparna Saha、Ujjwal Maulik 和 Kalyanmoy Deb。一种基于模拟退火的多目标优化算法：AMOSA。《IEEE进化计算汇刊》，12(3):269–283，2008年。

[11] Satyaprakash Song、Anirikh Chakrabarti、Jay Bhatt、Devendranath Bhatt 和 Jef- frey D. Varner。利用帕累托最优集成技术（ POETs ）识别信号转导模型集成。《生物技术杂志》，5(7):768–780，2010年。[12] David Bassen、Michael Vilkhovoy、Mason Minot、Jonathan Butcher 和 Jeffrey D. Varner。JuPOETs：一种在 Julia编程语言中估计生化模型集成的约束多目标优化方法。《BMC Systems Biology》，11(1):10，2017年。[13] Julian Blank 和 Kalyanmoy Deb。pymoo：Python中的多目标优化。《IEEE Access》，8:89497–89509，2020年。[14] Jes´us-Adolfo Mej´ıa-de Dios、Efr´en Mezura-Montes 和 Carlos A. Coello Coello。Meta-heuristics.jl：迈向可复现优化研究。《Journal of Open Source Software》，7(78):4723，2022年。[15] Bob Carpenter、Andrew Gelman、Matthew D. Hoffman、Daniel Lee、Ben Goodrich、Michael Betancourt、Marcus Brubaker、Jiqiang Guo、Peter Li 和 Allen Riddell。Stan：一种概率编程语言。《统计软件杂志》，76(1):1–32，2017年。[16] Hong Ge、Kai Xu 和 Zoubin Ghahramani。Turing：一种用于灵活概率推断的语言。载于第*21*届人工智能与统计国际会议论文集，第1682–1690页，2018年。[17] Daniel Foreman-Mackey、David W. Hogg、Dustin Lang 和 Jonathan Goodman。em- cee：MCMC锤子。《太平洋天文学会刊》，125(925):306–312，2013年。[18] To Thanh Binh 和 Ulrich Korn。MOBES：一种用于约束优化问题的多目标进化策略。载于第三届遗传算法国际会议论文集（*Mendel 97*），第176–182页，1997年。[19] Carlos M. Fonseca 和 Peter J. Fleming。多目标优化中进化算法概述。《进化计算》，3(1):1–16，1995年。[20] Jongmin Shin 和 Vincent Noireaux。大肠杆菌无细胞表达系统中的信使RNA失活与蛋白质降解研究。《生物工程杂志》，4:9，2010年。[21] Abhinav Adhikari、Michael Vilkhovoy、Sandra Vadhin、Ha Eun Lim 和 Jeffrey D. Varner。对无细胞转录和翻译过程进行有效的生物物理建模。《生物工程与生物技术前沿》，8:539081，2020年。[22] Matthew F. Hockin、Kenneth C. Jones、Scott J. Everse 和 Kenneth G. Mann。血液凝固化学计量调控模型。《生物化学杂志》，277(21):18322–18333，2002年。[23] Saulius Butenas、Cornelis van't Veer 和 Kenneth G. Mann。“正常”的凝血酶生成。《Blood》，94(7):2169–2178，1999年。

[24]Eckart Zitzler、Lothar Thiele、Marco Laumanns、Carlos M. Fonseca 和 Viviane Grunert da Fonseca。多目标优化器的性能评估：分析与综述。*IEEE*进化计算汇刊，7(2)：117–132，2003。

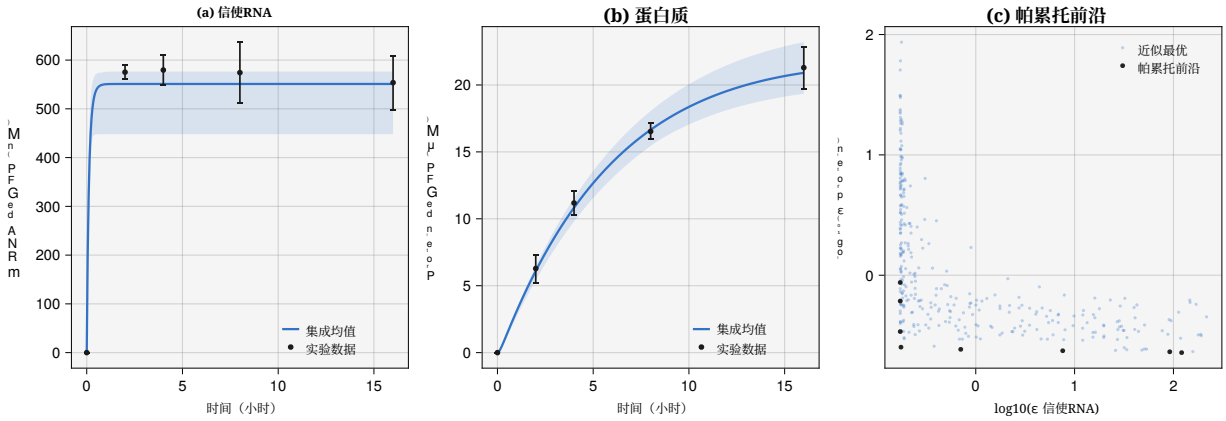

图 1：针对 Adhikari 等人 [21] 的实验数据拟合的无细胞基因表达回路（$\sigma_{70} \rightarrow$ P70 $\rightarrow$ deGFP）的集成估计。(a) 信使RNA和 (b) 蛋白质浓度随时间的变化；蓝色虚线表示集成均值，阴影区域表示 95%置信区间，黑点表示带有误差棒的实验数据。(c) 帕累托前沿，展示了信使RNA和蛋白质拟合误差之间的权衡。

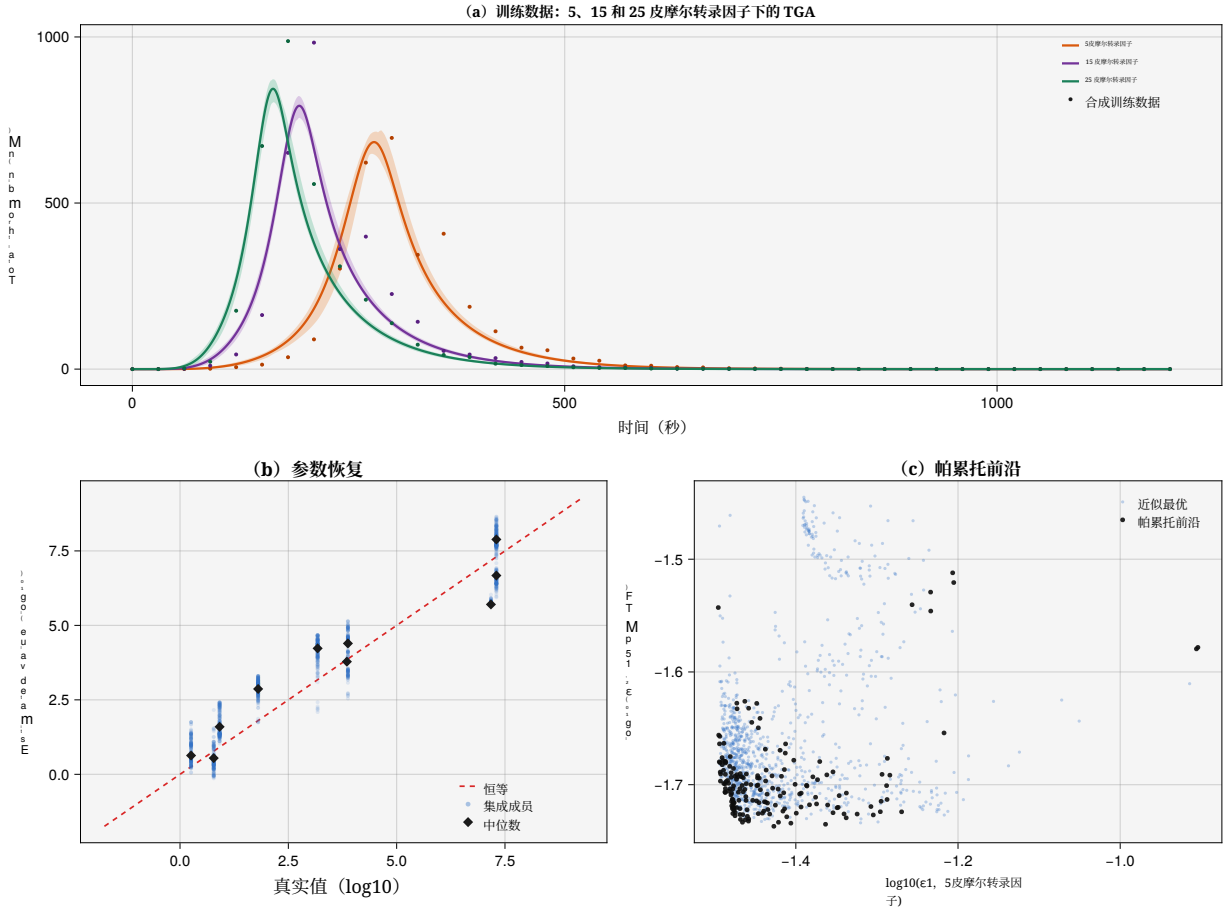

图 2：Hockin–Mann 血液凝固模型的训练结果（34 种物种、估计 10 个速率常数、3 个数据驱动目标、无正则化）。（a）在 5 皮摩尔（琥珀色）、15 皮摩尔（紫色）和 25 皮摩尔（青绿色）组织因子下，总凝血酶浓度随时间的变化；阴影区域表示集成 95% 置信区间，虚线表示集成均值，数据点表示含噪声的合成数据（15% 变异系数）。（b）对数空间中的参数恢复；红色虚线表示恒等线，菱形表示集成估计中位数；各参数周围的散点展示了仅凭凝血酶数据确定其可辨识程度。（c）帕累托前沿投影（ε_1 与 ε_2，按 ε_3着色），零阶解以黑色表示。

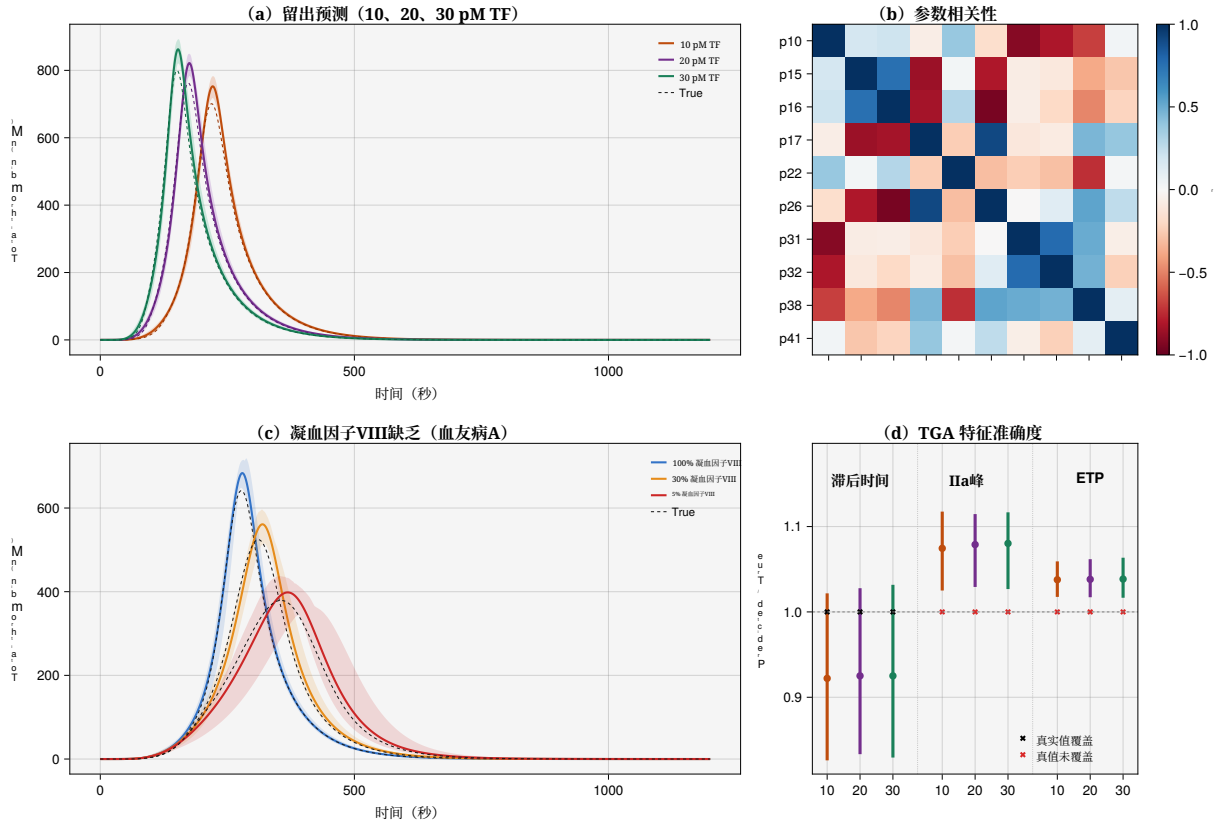

图3：凝血模型的基于集成的置信度表征。(a) 在 10、20 和 30 pM TF 条件下的留出凝血酶预测（训练期间未使用）；实线表示真实轨迹，虚线表示集成均值，阴影区域表示 95 % 置信区间。集成捕捉到了整体形状，但在凝血酶峰值处显示出系统性正偏差。(b) 成对参数相关性热图揭示了广泛的补偿结构；强负相关（例如，外源性凝血酶原激活酶 k_{cat} 与凝血酶原酶 k_{cat}，$r = -0.81$；Xa→IIa 因子与 IIa因子→Va因子，$r = -0.93$）解释了为何单个参数可能难以准确恢复，而模型预测仍然准确。(c) 凝血因子VIII缺乏（血友病A）患者在名义凝血因子VIII水平的 100%、30% 和 5% 下的患者特异性预测；集成预测带捕捉到凝血酶峰值随剂量依赖性降低，真实轨迹（黑色虚线）均落在三个预测带内。(d) 留出条件下的 TGA 特征准确度：集成预测的滞后时间、凝血酶峰值和 ETP 相对于真实值的归一化结果及其 95% 置信区间；黑色叉号表示真实值落在区间内，红色叉号表示未落在区间内。

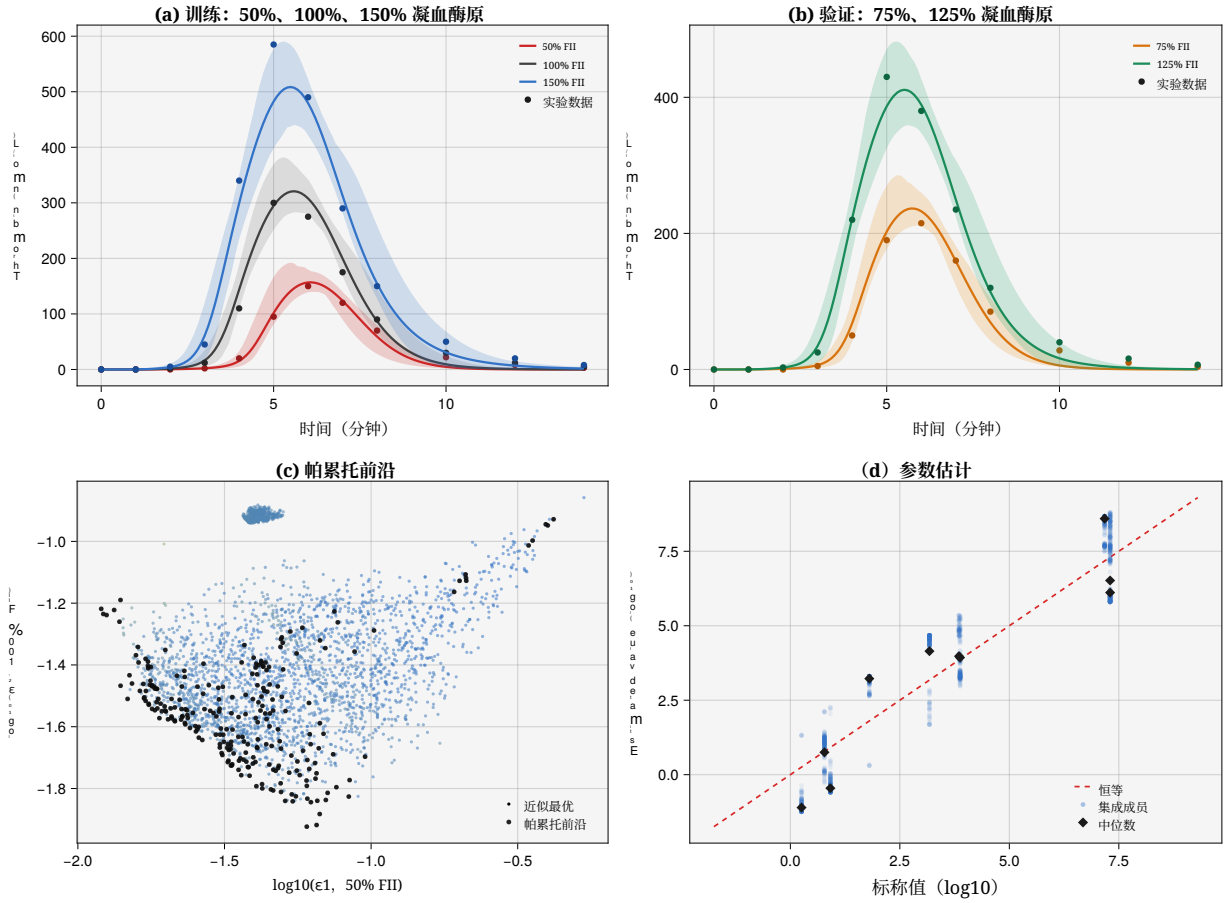

图4：将集成估计拟合至 Butenas等人 [23]的实验性凝血酶生成数据。在由 5 pmol/L 组织因子–活化因子VII 启动的重构合成血浆系统中，凝血酶原浓度在其平均血浆浓度的 50 % 至 150% 之间变化。(a) 凝血酶原浓度为 50%（红色）、100%（灰色）和 150%（蓝色）时的训练拟合；集成在低水平和正常水平的凝血酶原条件下捕捉到了实验数据，但低估了 150% 条件下的峰值，揭示了模型–数据张力。(b) 凝血酶原浓度为 75%（琥珀色）和 125%（青绿色）时的留出验证；对于训练期间从未使用过的条件，集成对凝血酶峰值的预测误差在 1–10% 以内。(c) 帕累托前沿投影，显示三个训练目标之间的权衡。(d) 参数估计与文献标称值的对比；偏差反映了使用近似模型拟合实验数据时所作的补偿性调整。

补充材料

S1 算法伪代码

本节提供 Pare- toEnsembles.jl 四个算法组件的完整伪代码。我们首先介绍主模拟退火循环，它负责协调优化过程；随后介绍该循环依赖的子程序：作为所有排序决策基础的严格支配检验、使每次迭代的开销保持线性的增量秩更新，以及用于并行化搜索的多链封装。

主算法。 ParetoEnsembles.jl 的核心是一个帕累托模拟退火循环（算法 S1），它会维护一个档案集，记录已评估的解及其帕累托等级。在每个温度层级，通过扰动当前最优参数向量来生成候选解，并通过 RankInsert增量更新档案集。接受的候选解会触发剪枝步骤，移除受到高度支配的解（秩 $\geq R_{\text{cutoff}}$）并强制执行档案集大小上限，随后对剩余的较小集合重新进行完整排序。拒绝的候选解会立即移除，秩数组则从快照中恢复；这就是正文所述的拒绝时弹出策略，可避免档案集在接受移动之间无限增长。

算法 S1 ParetoEnsembles：采用增量排序的帕累托模拟退火**要求：**用户回调函数：Objective, Neighbor, Accept, Cool**要求：**初始参数向量$\mathbf{x}_0$；超参数 N_{iter}, R_{cutoff}, $T_{\min}$, $n_{\max}$**确保：**包含目标值、参数和帕累托等级的解档案集

$\mathbf{x}_{\text{best}} \leftarrow \mathbf{x}_0$
$\mathcal{A} \leftarrow \{(\mathbf{x}_0, \text{Objective}(\mathbf{x}_0))\}$; $\mathbf{R} \leftarrow [0]$; $T \leftarrow 1$
当 $T > T_{\min}$ **时**
  对于 $t = 1, \ldots, N_{\text{iter}}$ **do**
    $\mathbf{x}' \leftarrow \text{Neighbor}(\mathbf{x}_{\text{best}})$; $\mathbf{f}' \leftarrow \text{Objective}(\mathbf{x}')$
    $\mathbf{R}_{\text{saved}} \leftarrow$复制$(\mathbf{R})$
    追加 $(\mathbf{x}', \mathbf{f}')$ to $\mathcal{A}$; RankInsert$(\mathcal{A}, \mathbf{R})$
    如果 Accept$(\mathbf{R}\ T) > U(0,1)$ **则**
      剪枝：$\mathcal{A} \leftarrow \{(\mathbf{x}_i, \mathbf{f}_i) \in \mathcal{A} \mid R_i < R_{\text{cutoff}}\}$
      $\mathbf{R} \leftarrow \text{FullRank}(\mathcal{A})$
      如果 $|\mathcal{A}| > n_{\max}$ **则**
        仅保留 $n_{\max}$ 个最低 R_i的解
        $\mathbf{R} \leftarrow \text{FullRank}(\mathcal{A})$
      结束 if
      $\mathbf{x}_{\text{best}} \leftarrow \mathbf{x}'$
    else
      从 $\mathcal{A}$ 中移除$(\mathbf{x}', \mathbf{f}')$； $\mathbf{R} \leftarrow \mathbf{R}_{\text{saved}}$
    结束 if
  结束 for
  $T \leftarrow \text{Cool}(T)$
结束循环
$\mathbf{R} \leftarrow \text{FullRank}(\mathcal{A})$
返回 $\mathcal{A}$, $\mathbf{R}$

严格支配检验。成对支配检查（算法 S2）通过遍历所有 m 目标，判断一个解是否严格支配另一个解。仅当解 j 在每个目标上都不劣于 i，且至少在一个目标上严格更优时，检验才返回TRUE；这就是严格帕累托支配关系，它修正了原始 POETs 实现中使用的弱支配。提前退出条件会在发现j 在任何一个目标上劣于 i时立即终止循环，从而避免不必要的比较。

算法 S2 StrictlyDominates($\mathbf{f}_j$,$\mathbf{f}_i$, m): 解 j 是否严格支配解 i?

要求： 目标向量 $\mathbf{f}_j$，$\mathbf{f}_i \in \mathbb{R}^m$ （越低越好）

确保： TRUE 如果 $\mathbf{f}_j \prec \mathbf{f}_i$, FALSE 否则

严格更优 $\leftarrow$ FALSE
对于 $k = 1, \ldots, m$ **do**
  如果 $f_{j,k} > f_{i,k}$ **那么**
    返回 FALSE ▷ j 在目标函数 k上更差
  否则如果 $f_{j,k} < f_{i,k}$ **那么**
    严格更优 $\leftarrow$ TRUE ▷ j 在 ≥ 1 目标函数上更优
  结束 if
结束 for
返回严格更优 _

增量秩更新。 将单个候选解追加到档案集时，重新计算所有两两支配关系的成本为 $O(n^2m)$。 RankInsert 过程（算法 S3）利用了只有一个解发生变化这一事实：它对现有档案集进行一次遍历，测试新解与每个现有解之间两个方向的支配关系。新解支配现有解时，现有解的秩会递增；新解的秩则根据支配它的现有解数量累加得出。这将每次迭代的成本降至 $O(nm)$，并将完整的 $O(n'^2m)$ 重新排序推迟到剪枝步骤，此时档案集规模更小。

算法 S3 RankInsert($\mathcal{A}$,$\mathbf{R}$): 追加一个解后的增量秩更新

要求： 档案集 $\mathcal{A} = \{(\mathbf{x}_1, \mathbf{f}_1) \ldots, (\mathbf{x}_n, \mathbf{f}_n)\}$ 其中 $(\mathbf{x}_n, \mathbf{f}_n)$ 是新解

要求： 对解 1、.....、 $n-1$有效的秩数组 $\mathbf{R} = [R_1, \ldots, R_{n-1}]$

确保： 更新后的、对完整档案集有效的 $\mathbf{R} = [R_1, \ldots, R_n]$

$R_{\text{new}} \leftarrow 0$
对于 $i = 1, \cdots, n-1$ **执行**
  如果 StrictlyDominates($\mathbf{f}_n$, $\mathbf{f}_i$, m) **则**
    $R_i \leftarrow R_i + 1$ ▷ 新解支配 i
  结束条件判断
  如果 StrictlyDominates($\mathbf{f}_i$, $\mathbf{f}_n$, m) **则**
    $R_{\text{new}} \leftarrow R_{\text{new}} + 1$ ▷ i支配新解
  结束条件判断
结束循环
追加 R_{new} 至$\mathbf{R}$

多链并行执行。 由于模拟退火在单链内本质上是顺序执行的，因此最有效的并行化策略是从不同起始点运行 C 独立链，并合并结果（算法 S4）。每条链都会获得一个独立的确定性随机数生成器，该生成器由基础种子和链索引设定种子，从而确保完全可复现。所有链运行完成后，将它们的档案集连接起来

并在合并集合上进行最终的完整排序，生成一个组合集成，其对权衡曲面的覆盖范围比任何单条链都更广。

算法 S4 ParetoEnsemblesParallel：多链并行执行

要求： 初始状态 $\{\mathbf{x}_0^{(1)}, \ldots\ldots, \mathbf{x}_0^{(C)}\}$；基础种子 s

确保： 带有帕累托等级的合并档案

并行处理 $c = 1, \ldots\ldots, C$

  $\text{rng}_c \leftarrow \textsc{Seed}(s, c)$

  $(\mathcal{A}_c, \mathbf{R}_c) \leftarrow \textsc{ParetoEnsembles}(\mathbf{x}_0^{(c)}, \text{rng}_c)$

4: </style}结束循环

$\mathcal{A} \leftarrow \mathcal{A}_1 \cup \cdots \cup \mathcal{A}_C$

$\mathbf{R} \leftarrow \textsc{FullRank}(\mathcal{A})$

返回 $\mathcal{A}$, $\mathbf{R}$

S2 软件设计

ParetoEnsembles.jl 是一个纯 Julia 软件包，除标准库模块 `Random`外没有其他依赖，其公共 API 由五个函数组成（表 S1）。主要入口 `estimate_ensemble`运行单条帕累托模拟退火链，并返回保留的目标值、参数向量和帕累托等级；其并行对应函数 `estimate_ensemble_parallel`则并发运行 C 条独立链，合并其档案集，并对合并后的集合重新排序。独立函数 `rank_function`可为任意 $m \times n$ 目标矩阵计算帕累托等级，也可独立于优化循环使用； `hypervolume` 使用扫描线算法计算二维前沿的超体积指标，时间复杂度为 $O(n \log\ n)$；而 `pareto_front` 则从档案集中提取零级子集，以供下游分析。

该算法由四个用户提供的回调函数定义（表 S2）：目标函数将参数向量 $\mathbf{x} \in \mathbb{R}^d$ 映射为一个目标值数组；邻域函数扰动给定的参数向量以生成候选解；接受概率函数根据秩数组和温度返回一个标量概率；冷却函数则将当前温度映射为较低的值。除这四个回调函数外， `estimate_ensemble` 的行为还由关键字参数控制，包括剪枝阈值 `rank_cutoff`（默认值为 5）、每个温度级别的候选解数 `maximum_number_of_iterations`（默认值为 20）、停止温度 `temperature_min` （默认值为 10^{-4}）以及档案集大小硬上限 `maximum_archive_size`（默认值为 1000）。当启用 `trace=true`时，算法会在每个冷却步骤记录收敛诊断信息（温度、档案集大小和超体积），并在输出元组中返回第四个元素，供用户监测收敛情况并评估退火计划是否充分。

表 S1：ParetoEnsembles.jl 的公共 API。

函数	**描述**
`estimate_ensemble`	运行单条帕累托模拟退火链。返回目标值、参数向量和保留档案集的帕累托等级。
`estimate_ensemble_parallel`	并行运行 C 条独立链（每个起始点运行一条），合并档案集并重新排序。
`rank_function`	计算 $m \times n$ 目目标矩阵。
`hypervolume`	计算以下对象的超体积指标：给定参考点的二维帕累托前沿。
`pareto_front`	从档案集中提取帕累托最优（秩 $= 0$）解从档案集中选取

表 S2：用户提供的回调函数。

回调函数	签名	用途
目标函数 _	（x） $\to \mathbb{R}^{m\times 1}$	计算以下位置的 m目标参数向量 **x**
邻域函数 _	**(x)** $\to \mathbb{R}^{d}$	通过以下方式生成候选解扰动 **x**
接受概率函数 _	**(R** T**)** $\to [0,1]$	接受 概率根据秩和温度 ture
冷却函数	(T) $\to T'$	退火计划

S3 并行执行

该算法的结构带来了两种并行化机会。首先，完整排序和增量式 RankInsert 过程的外层循环都会独立遍历档案解，因此设置 `parallel evaluation=true`后，会调度线程化变体，利用 Julia 的`Threads.@threads`结构将这些独立迭代分配到可用线程上；在增量式情形下，由于多个线程都会对 R_{new} 作出贡献，因此会使用原子累加器。其次，也是更重要的一点，由于模拟退火在单链内部本质上是顺序执行的，最有效的并行化策略是从不同起始点运行 C 条独立链，并合并生成的档案（算法 S4）；每条链都会使用由基础种子和链索引生成种子的专属确定性随机数生成器，以确保可复现性。所有链运行完毕后，将拼接各档案，并对合并集合执行最终的完整排序。

为量化多链并行的优势，我们测量了无细胞集成估计（十条链， $N_{\text{iter}} = 50$）在不同 Julia 线程数下的墙钟时间（表 S3）。每种配置运行五次，并报告中位数时间；加速比在最多四个线程时近似线性增长（$3.1\times$），在八个线程时达到 $4.1\times$。由于十条链分配到八个线程时存在负载不均衡，以及最终档案合并和重新排序步骤的串行开销，继续增加线程数的收益逐渐减少。

表 S3：无细胞基因表达模型上的多链并行扩展性（十条链，$N_{\text{iter}} = 50$，五次运行的中位数）。

线程数	挂钟时间（秒）	加速比
1	18.52	$1.0\times$
2	9.38	$2.0\times$
4	5.94	$3.1\times$
8	4.51	$4.1\times$

S4 标准基准测试

我们在两个标准多目标基准测试上验证了 ParetoEnsembles.jl：有约束的 Binh–Korn 问题 [18]（方程 (3)）和无约束的 Fonseca–Fleming 问题 [19]，其决策变量数为 $d = 3$（方程 (4) 和图 S1）。Binh–Korn 问题中的约束通过添加到目标值中的二次惩罚项来施加，边界约束则由邻域函数中的截断处理。我们使用十条并行链运行这两个基准测试，每个温度下有 $N_{\mathrm{iter}} = 60$ 个候选解，秩截断值为 $R_{\mathrm{cutoff}} = 12$，冷却速率为 $\alpha = 0.95$；所得目标空间帕累托前沿再现了两个问题各自典型的权衡曲线，且非支配前沿周围可见一簇近似最优解（图 S1a、b）。Fonseca–Fleming 问题计算得到的前沿恰好落在理论曲线上，证实该算法收敛到了正确的解集；在参数空间中，Binh–Korn 的帕累托最优决策向量沿曲面流形从原点延伸至 (5,3)，而 Fonseca–Fleming 的解则如预期那样沿对角线聚集，因为其目标具有对称结构（图 S1c、d）。在 Binh–Korn 问题上比较单链与十条并行链，展示了多链执行的优势（图 S2）：单链能够找回权衡曲线，但由于该链档案中每个保留的解都是非支配解，秩数组只包含秩零条目；而合并的多链运行得到的前沿更密集，对可行域的覆盖也更广。

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & f_1 = 4x_1^2 + 4x_2^2, \qquad f_2 = (x_1 - 5)^2 + (x_2 - 5)^2 \\
\text{s.t.} \quad & (x_1 - 5)^2 + x_2^2 \le 25, \qquad (x_1 - 8)^2 + (x_2 - 3)^2 \ge 7.7 \\
& 0 \le x_1 \le 5, \quad 0 \le x_2 \le 3.
\end{aligned}
\tag{3}
$$

$$
f_1(\mathbf{x}) = 1 - \exp\!\left(- \sum_{i=1}^{d} \left(x_i - \tfrac{1}{\sqrt{d}} \right)^2 \right), \qquad f_2(\mathbf{x}) = 1 - \exp\!\left(- \sum_{i=1}^{d} \left(x_i + \tfrac{1}{\sqrt{d}} \right)^2 \right). \tag{4}
$$

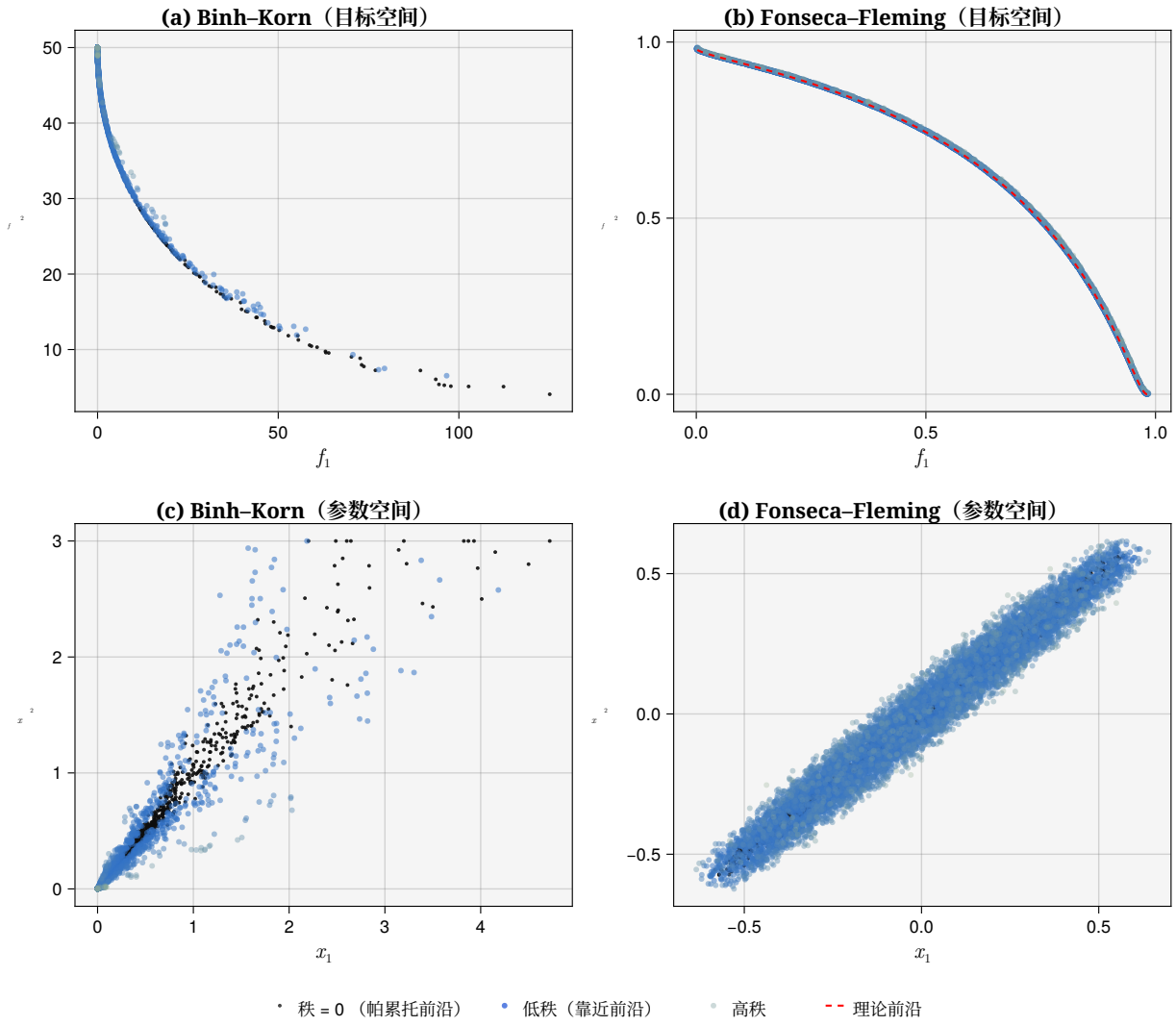

图 S1： (a,c) Binh–Korn 和 (b,d) Fonseca–Fleming 的基准测试结果（$d = 3$）。上排：目标空间；下排：参数空间投影（x_1 与 x_2）。非支配解（秩 $= 0$，深色）定义了帕累托前沿，而近似最优解（蓝色，按秩着色）则在其周围形成一个点云，代表保留集成。(b) 中的红色虚线表示理论帕累托前沿。使用了十条并行链，参数为 $R_{\text{cutoff}} = 12$、$N_{\text{iter}} = 60$ 和 $\alpha = 0.95$。

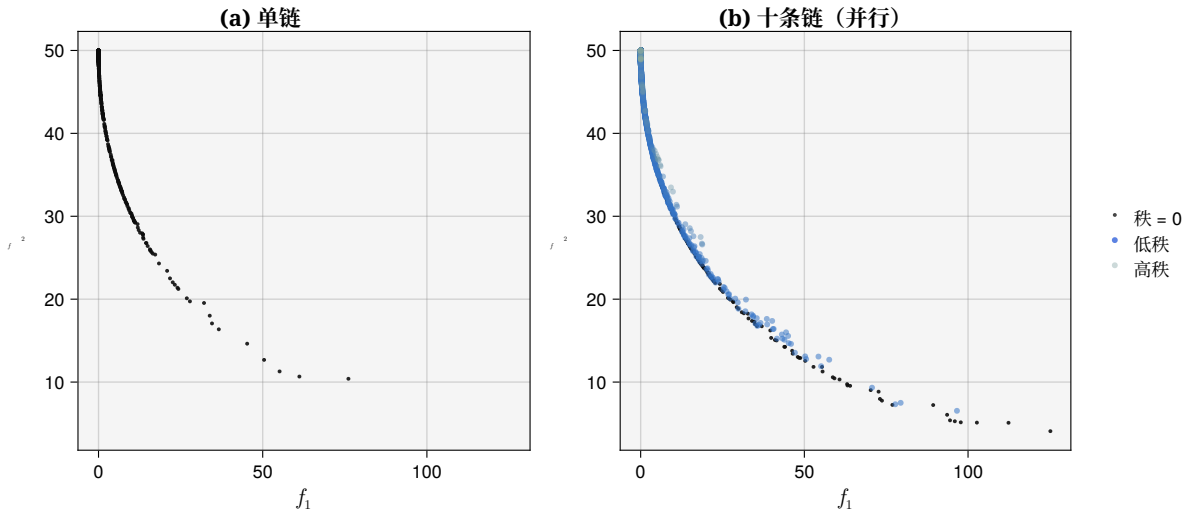

图 S2：Binh–Korn 问题中 (a) 单链与 (b) 十条链并行执行的比较。两个面板使用相同的坐标轴范围。单链产生的非支配解档案集较为稀疏，而合并后的多链档案具有更密集的前沿覆盖，并且可以看到前沿周围分布着近似最优解（蓝色）。

S5 与 NSGA-II 的比较

为明确 ParetoEnsembles.jl 相对于基于种群的进化方法的定位，我们将其与 Metaheuristics.jl 中实现的 NSGA-II [5] 进行了比较，涵盖两个基准问题，并使每个求解器的函数评估预算大致匹配为 110,000 次评估（ParetoEnsembles：10 条链、每个温度对应 $N_{\text{iter}} = 60$ 个候选解、冷却速率$\alpha = 0.95$；NSGA-II：种群规模 200、550 代）。每个求解器均进行了五次独立重复实验，并在表 S4 中报告了超体积指标（HV）和反世代距离（IGD）的中位数；其中，Fonseca–Fleming 问题的 IGD 相对于解析帕累托前沿计算，而 Binh–Korn 的 IGD 则相对于一次高预算运行中密集采样的参考前沿计算。ParetoEnsembles.jl 在两个问题上的超体积均达到与 NSGA-II 相当或更高的水平，且 IGD 显著更低（在 Binh–Korn 上低 $5\times$，在 Fonseca–Fleming 上低 $7\times$），表明其对真实帕累托前沿的近似更为接近（图 S3）。同时，它生成的帕累托前沿密度也高得多，包含 2,000–6,000 个非支配解，而 NSGA-II 仅有 200 个；这体现了其作为集成生成器的设计目标，即刻画完整的近似最优解云，而不仅仅是近似前沿边界。根据问题规模不同，NSGA-II 的运行时间快 6–$150\times$ 倍，这与其种群并行评估策略一致；但这种速度优势是以权衡曲面表示更为稀疏为代价的，可能不足以支持后续的不确定性分析。

表 S4：ParetoEnsembles.jl 与 NSGA-II 在基准问题上的比较（5 次重复实验的中位数）。HV 越高、IGD 越低越好。两个求解器均进行了 $\sim$110,000 次函数评估。

问题	求解器	HV	IGD	前沿大小	时间（秒）
Binh–Korn	ParetoEnsembles	**8035**	**0.014**	6172	77
	NSGA-II	8027	0.071	200	0.5
Fonseca–Fleming	ParetoEnsembles	**0.44**	**0.0004**	2051	3.1
	NSGA-II	0.44	0.003	200	0.5

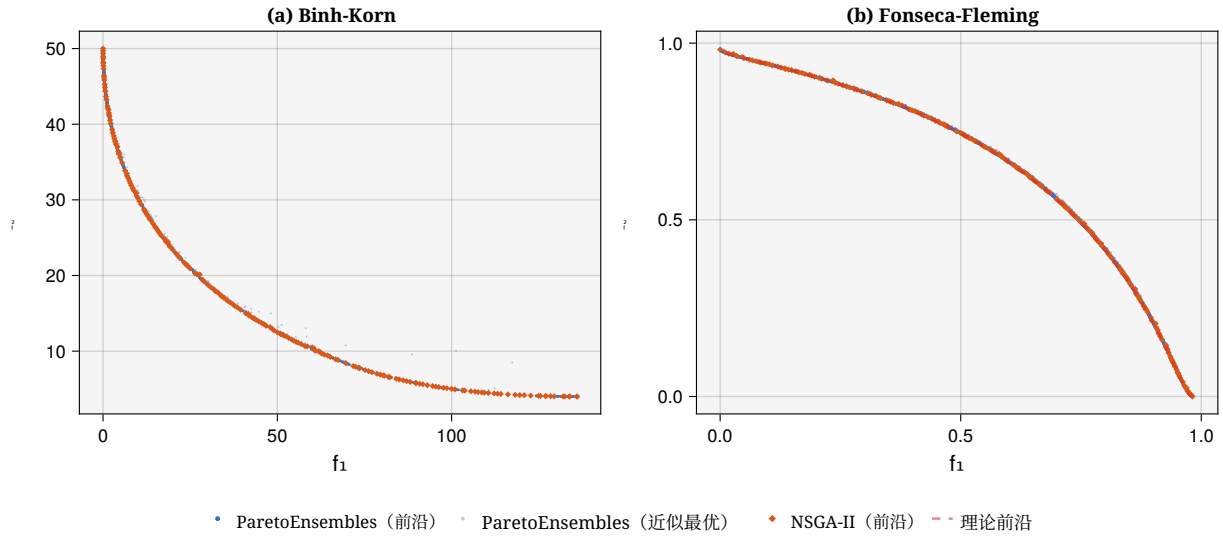

图 S3：在匹配的评估预算（∼110,000）下，ParetoEnsembles.jl（左）与 NSGA-II（右）在（a、b）Binh–Korn 和（c、d）Fonseca–Fleming 问题上的比较。除帕累托前沿（黑色）外，ParetoEnsembles.jl 还保留了近似最优解云（蓝色），而 NSGA-II 仅返回非支配前沿。（c、d）中的红色虚线曲线为理论 Fonseca–Fleming 前沿。

S6 超参数敏感性

我们通过逐一调整三个关键超参数（秩截断值R_{cutoff}、冷却速率 α 和每个温度的迭代次数 N_{iter}），同时将其他参数保持为默认值（$R_{\text{cutoff}} = 8$, $\alpha = 0.90$, $N_{\text{iter}} = 40$），在 Binh–Korn 基准上评估了 ParetoEnsembles.jl 的敏感性；每种配置均使用五条链和五次重复实验（图 S4）。在所有测试范围内，超体积指标都表现稳健：秩截断值从 2 到 12 时，结果几乎相同；当取值为 $R_{\text{cutoff}} = 20$ 时，结果有所下降，因为此时档案集保留了许多质量较低的解；较快的冷却（$\alpha = 0.80$）与较慢的冷却（$\alpha = 0.95$）表现相当，后者的变异性更大；即使每个温度层级仅迭代 10 次，也能得到具有竞争力的帕累托前沿，不过迭代 20 次或更多可降低重复实验间方差。这些结果表明，默认超参数可为大多数问题提供合理的起点，用户无需投入大量精力进行调优；对于 ParetoEnsembles.jl 主要面向的系统生物学应用而言，这是一个实际优势。

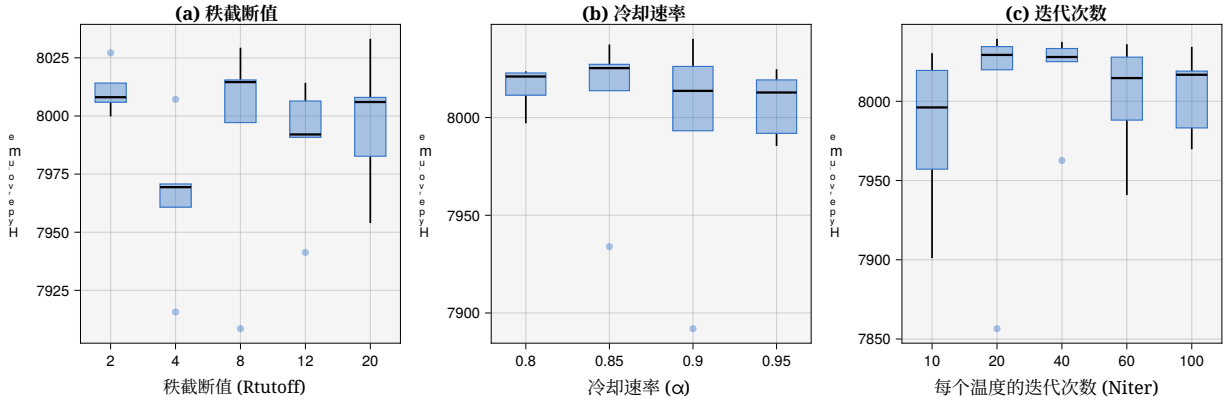

图 S4：Binh–Korn 基准上的超参数敏感性。箱线图展示了 5 次重复实验的超体积分布（中位数、四分位距和须线）。(a) 秩截断值 R_{cutoff}，(b) 冷却速率 α，(c) 每个温度的迭代次数 N_{iter}。每次扫描时，另外两个参数均保持默认值不变。

S7 对模型设定错误的稳健性

正文中的凝血结果使用同一模型生成并拟合数据，因此拟合模型按构造而言完全正确。在反问题文献中，这种情形有时称为“逆犯罪”（图 S5）。为评估当这一假设不成立时集成方法的表现，我们使用一个有意设定错误的模型重复了凝血研究：生成合成训练数据之前，将十个估计参数之外的六个速率常数扰动了 $\pm 30\%$（偏离其标称值），因此拟合模型的搜索空间中不存在能够完美再现数据的参数向量。受扰动的参数涉及起始（TF=VIIa+VII→VIIa）、结合（外源性凝血酶原激活酶和 IX 激活 K_m）、组装（凝血酶原酶缔合速率）及抑制（Xa+TFPI 和 mIIa+ATIII）通路，从而确保模型设定错误会影响凝血级联的多个阶段。

尽管存在模型误差，集成在全部三种 TF 浓度下仍生成了与训练数据高度吻合的凝血酶轨迹（图 S5a），表明十个估计参数通过补偿性调整吸收了大部分设定错误。在留出验证条件（10、20 和 30 皮摩尔 TF）下，集成对凝血酶峰值的预测与真实值相差仅 4–6%（图 S5b），与正文中使用正确模型时观察到的 6–7% 误差相当。然而，轨迹级覆盖率显著下降：95%预测区间仅在 31–48% 的时间点覆盖真实轨迹，而正确模型下的覆盖率更高，这反映出集成仅凭参数不确定性无法解释结构性模型误差。

在设定错误的情况下，TGA 特征准确度的表现模式也发生了变化（图 S5c）。在正确模型研究中覆盖良好的滞后时间出现了 5–9% 的系统性高估，在某些条件下真实值落在 95% 区间之外；而在正确模型下存在偏差的内源性凝血酶潜能（ ETP ），此时预测值与真实值相差仅 0.3–2.5%，且覆盖率良好。凝血酶峰值仍存在 4–6% 的偏差，与正确模型情形相近。这些变化反映出，估计参数会根据所评估的凝血酶曲线特征，以不同方式补偿设定错误的固定参数。它们还凸显了参数不确定性表征的一个根本局限：集成只能在给定模型结构内探索参数向量；当该结构仅为近似时，即使点预测仍然准确，预测区间也可能校准不当。参数恢复（图 S5d）显示出比正确模型情形更大的离散程度和偏差，这符合预期，因为在设定错误的模型下，“真实”参数值不再是最优值。

这些结果证实，集成方法对中度模型错误设定具有稳健性，预测性能会逐渐下降；同时也揭示了参数不确定性与结构不确定性之间的界限：集成能够捕捉参数不确定性，而结构不确定性则需要通过模型扩展或显式的模型误差项来处理。

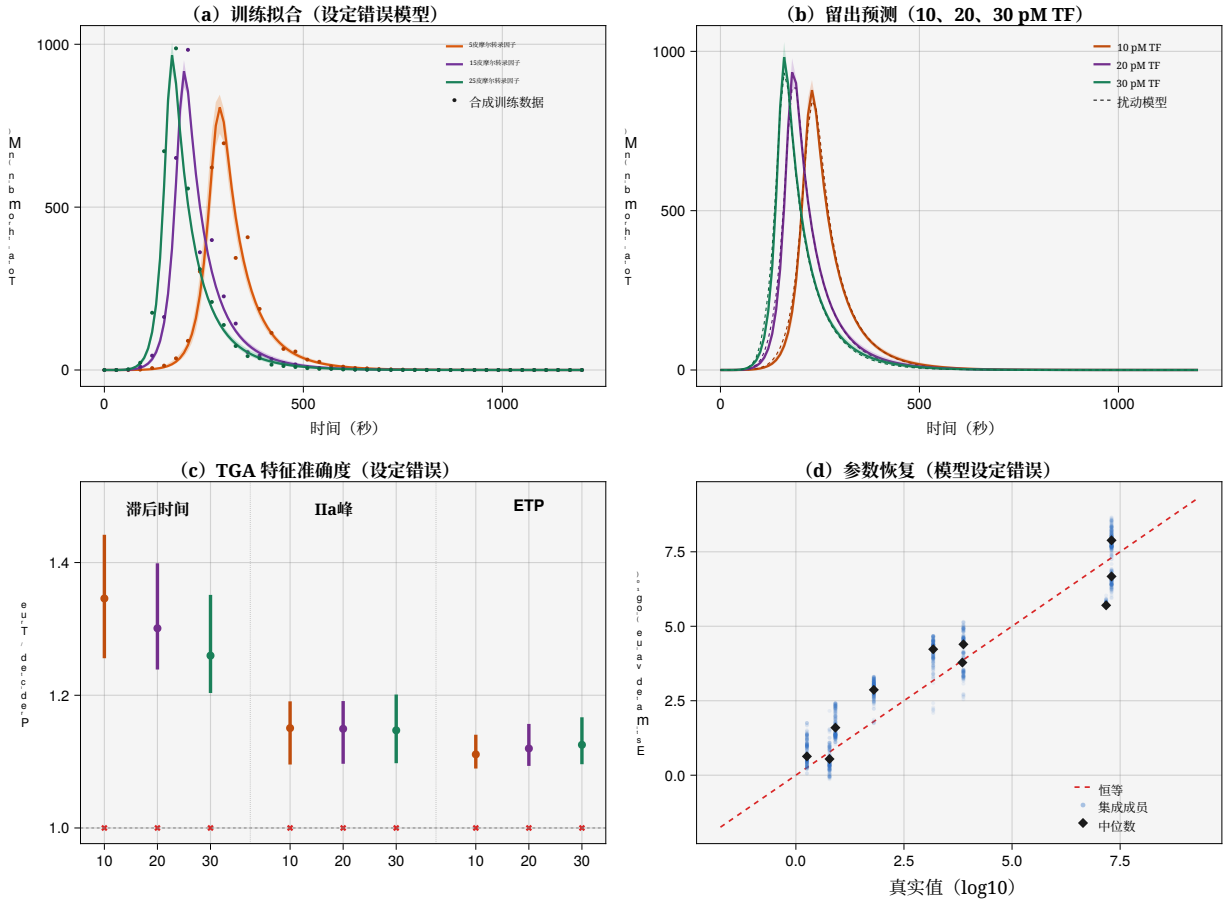

图 S5：模型设定错误研究：生成训练数据前，将六个固定速率常数扰动 ±30%，随后使用名义模型进行拟合。(a) 在 5、15、25 皮摩尔转录因子条件下的训练拟合；尽管模型设定错误，模型仍能追踪数据。(b) 在 10、20、30 皮摩尔转录因子条件下的留出预测；集成均值（虚线）与真实轨迹（实线）的偏差在 4–6% 以内，但 95% 置信区间（阴影部分）未能完全覆盖真实值。(c) 留出条件下的 TGA 特征准确度，以真实值归一化；相较于正确模型情形，偏差模式有所变化。(d) 参数恢复显示，其离散程度更大、偏差也高于正确模型；当拟合模型在结构上只是近似模型时，这是预期的结果。

S8 代码示例

代码清单 1：使用 ParetoEnsembles.jl 的 Binh–Korn 基准。

```
使用ParetoEnsembles

函数 目标函数_function(x)
    f = zeros(2, 1)
    f[1] = 4.0 * x[1]^2 + 4.0 * x[2]^2
    f[2] = (x[1] - 5)^2 + (x[2] - 5)^2
    # penalty for constraint violations
    λ = 100.0
    v1 = 25 - (x[1] - 5)^2 - x[2]^2
    v2 = (x[1] - 8)^2 + (x[2] - 3)^2 - 7.7
    f[1] += λ * (min(0, v1))^2
    f[2] += λ * (最小值(0, v2))^2
    返回 f
end

邻居_function(x) = clamp.(
    x .* (1 .+ 0.05 * randn(length(x))),
    [0.0, 0.0], [5.0, 3.0])

接受概率函数(R, T)__ =
    exp(-R[end] / T)

冷却函数(T)_ = 0.9 * T

(EC, PC, RA) = 估计集成(_
    目标函数_, 邻居函数_,
    接受率_概率_函数, 冷却函数_函数,
    [2.5, 1.5]; 秩_截断=4.0,
    最大值 -数量 -of-迭代次数=40 , 显示_跟踪=否)

帕累托_索引 =查找全部(RA .== 0)
```

代码清单 2：多目标在 Binh–Korn 问题上并行执行链（需要 `julia -t4`）。

```
使用ParetoEnsembles

# ... (same callbacks as Listing 1) ...

初始_状态 = [
    [2.5, 1.5], [0.5, 2.5],
    [4.0, 0.5], [1.0, 1.0]
]

(EC, PC, RA) = 估计_集成_并行(
```

```
目标函数_函数，邻居_函数，12 接受_概率_函数，冷却函数_函
数，13 初始_状态；14 秩_截断=4.0，最大_数量_的_迭代次数=40,
显示_轨迹=false，随机数生成器_种子=42) 16 17 println("解：
␣", size(EC, 2)) 18 println("帕累托最优：␣", count(RA .== 0))
```