当智能进入现实
从养老照护检验人机安全合作

王建明（Jianming Wang）

2026 年 11 月 20 日 · v2.2

智能时代文明转型三部曲 · 安全篇

摘要

判断建议出错、系统故障或退出 AI 渠道后，养老机构仍须完成已经登记的必要照护。本文以原期限内实际交付为主要结果，提出独立复核和备用接手的比较方案。对固定权重的缺失结局，推导同时满足分层和总体宽度上限的最少核验量，并扩展到每条核验费用不同的情形。等成本和异成本算法分别在 157,464 与 12,000 组合成配置下核对。对美国 14,474 家养老机构的单季公开工时按机构配对复算，周末零工时的差异取决于选取直接护理 RN 或全体 RN 岗位。公开工时用于说明核查口径，照护交付和接手效果需要现场验证。

关键词　人工智能安全；养老照护；有限授权；独立复核；服务接续；人机合作

Abstract

We study on-time delivery of registered elder-care obligations after incorrect AI advice, system failure, or AI opt-out. A proposed cluster trial compares independent review and staffed backup. For fixed-weight missing-outcome bounds, we derive the minimum number of successful equal-cost audits and the minimum cost of successful audits with unequal costs under both stratum-level and overall width caps. We check the algorithms against enumeration on 157,464 and 12,000 synthetic configurations, respectively. Paired facility analysis of public staffing records from 14,474 US nursing homes shows that weekend differences in reported zero hours depend on whether direct-care or all registered nurse roles are counted. Staffing records inform measurement choices. Care delivery and the effects of backup remain to be measured in the field.

一、当帮助离开屏幕

一个独居老人可以借助设备提醒日常事项，也可以用它联系护理人员。设备发现异常后，能把信息发给值班人员，使有限的照护人力更快到达需要帮助的人。第一天演示顺利，几个月后，机构可能因此调整排班，把原先的电话回访交给设备。如果夜间信号没有送达，又或者接到信号的人无权安排上门，日志上仍可能留下「已发送」，家里的老人却一直没有等到帮助。这是一个分析情境。它问的是部署之前就能查清的事：哪项服务依赖了这个系统，故障之后谁继续提供它？

工业和信息化部与民政部于 2025 年启动智能养老服务机器人结对攻关与场景应用试点，覆盖居家、社区和机构养老，随后公布 32 个入围项目。通知把应用验证和评价体系列为试点任务。项目入围说明已经具备开展验证的组织条件，长期照护结果仍需项目数据来回答。[1–2]

经济篇讨论生产和就业变化后普通人怎样取得生活资料，认知篇讨论能力怎样转为理解、参与和判断。[3–4] 安全篇以照护作为观察场景：技术带来的节省能否转化为可持续的支持，接受支持的人在看不懂系统判断、拒绝额外采集或设备停用后，能否继续获得必要服务？

二、怎样知道保护到达了人

检验保护是否到达时，逐一记录三个环节。**告警**使问题被发现。**接手**要求有责任的人收到信息、判断并具备行动条件。**恢复**要回到人的处境核实，例如确认当次必要巡视已完成、遗漏的服务已补上、老人知道下次怎样求助。前一步的成功只能作为后一步的线索。每一步都需要事件时间、责任人以及可复核的完成状态。

研究可按一段明确的照护窗口记录四类结果。第一，必要服务是否在预先确定的时间内实际到达。第二，个人或受托复核者能否查到与决定相关的依据，并在期限内得到答复。第三，出错后从发现到恢复用了多久，是否有无法挽回的损失。第四，拒绝附加采集、关闭设备或更换机构后，同一项基本服务是否仍可通过人来取得。这四项分别报告。一个高告警准确率不能抵消无法退出，低平均成本也不能抵消夜间服务连续中断。

可核查性和可恢复性是不同的。一次提醒误发也许容易更正，记录却未必足以重建它为何发出。一次排班指令可以完整留下，也可能因此错过关键的照护时间。对前者，应改进复核所需的记录。对后者，更早的人工介入或保留原有服务才有价值。

国务院办公厅关于解决老年人运用智能技术困难的实施方案，在高频服务中要求保留传统服务渠道。评估照护设备可以采用同一思路，把电话或人工渠道的实际可达性纳入观察。[5] 澳大利亚 Robodebt 调查涉及自动化行政决定，并提出清楚的复核途径及独立审查等建议。与照护的可比之处是：正式的系统结果可能改变人的生活，复核是否可达需要单独检验。[6]

谁判断照护必要及服务恢复

试点以老人和照护团队共同确认的服务计划为准，在事件前登记必要项目、最迟完成时点、允许的人工替代方式和可核实的完成标准。症状、偏好或情形变化由有职责的人判断，另存新版本并记理由和时点。研究保留事件开始时的版本，事后改期不能覆盖原期限。本人停止整个服务项目和仅退出 AI 渠道分别记录。后者仍希望取得服务时，保留同一义务和原期限。跨场地先按居家、社区和机构分层核计。

必要照护与退出后的观测规则

对每项义务只判一次主要结果，即原期限内完成、未完成或未知。晚于期限的补救、不可恢复的损失与完成证据另记。退出 AI 渠道时，同时记录退出时间、首次接通人工和同项义务的完成。已核实的完成率分子以独立照护记录、本人回访或相应可追溯证据确认。无法确认结局时保留义务在册。

设某组在固定观察窗有 N 个登记义务，T 个确认按时，U 个未知。未知中可有 0 至 U 个按时，故该组按时率在 $[T/N, (T+U)/N]$。对两组分别记界限 $[L_1, V_1]$、$[L_0, V_0]$，组 1 的完成率至少 L_1，组 0 至多 V_0，因此差至少 $L_1 - V_0$。交换各组端点得上界 $V_1 - L_0$。当只知道上述计数时，把组 1 的未知全记失败、组 0 全记通过达到下界，反向达到上界。因此无需假定两组缺失机制独立、相同或随机。若有跨组共享的真实约束，按该约束可进一步收紧。漏登记会改变分母，需要另从计划簿核查覆盖率。这是识别区间，与随机分配的抽样误差分别报告。

同一照护义务怎样连接各项量

操作量	对应框架的测量域	本文的操作定义
照护义务、原期限与按时完成	WHO 的及时性及管理连续性。AHRQ 的需求与目标评估、预先照护计划。	核定计划版本、原期限、允许替代与完成证据。按是否真实到达计入主要结果。
复核可达与证据取得	AHRQ 的信息沟通及责任确认。WHO 的信息连续性。	记录复核者是否能取得原计划、明确回应和签核。原系统失效时检验独立路径。
人工接手及恢复时间	AHRQ 的促进交接、跟踪变化。SEIPS 的人员、任务、工具和组织。	记录安全撤销、值班确认及实际执行的时差。分记备援共享账号或排班的失效。
退出后的同项服务	WHO 的连续性和以人为本。AHRQ 的支持自我管理、社区资源联结。	退出数字途径后按原期限测量人工可达、完成以及结局未知。
告警和干预	AI Control 的监控、干预与审计。Async Control 区分审计延迟。	记录告警、审计和停用时刻。独立复核检验证据和决定，备援班组检验替代服务交付。

表中连续性与工作系统对应关系分别来自 WHO、AHRQ 及 SEIPS。[9–10,16–18] AI 控制研究中的监控、干预和审计主要回答一项行为是否受控。[7–8] 照护中还存在独立的到期义务。及时制止一项错误自动指令后，仍须检验人手和渠道能否把原任务送达。因此用同一个义务编号连接两类到达与完成记录，分别辨认已发现但未送达、已送达但逾期的事件。

模型仍在线时怎样失效

掉线只覆盖一种路径。对于模型提供判断和建议的系统，还要在仿真演练或历史回放中观察三种情况。第一，输入已过期或新版本的告警规则改变，界面仍显示正常。按版本、时间与日型保存规则、输入是否齐全及独立复核的需要事件，比较相同证据条件下的告警漏检。第二，系统给出语气确定的判断，但按独立来源查不到依据。分别登记原决定、展示的说明、人员作出的核对或暂缓及是否有权恢复。第三，照护人员原先采取合适措施，看到错误建议后反而取消，或没有按时接手。仅在可撤销的演练场景记录建议展示前后的判断，并由另一人核对完成证据。临床支援的情境实验曾报告从正确判断转为错误的情形，其风险大小仍须在所用场景测量。[34] 这些标记分别连接告警、接手与服务到达，避免把系统在线或收到回复当作照护已经恢复。

怎样判定一次复核有效

将所有提出复核的适用事件纳入分母，记录提出时间、首次反馈时间、说明送达时间、复核决定与补救完成时间。一次有效复核同时满足三项条件：在预先公布的期限内给出反馈，解释原决定与所用材料并允许补充信息，有权改正的人作出可追踪的决定并告知下一步异议路径。补救完成另作结果，不能从已经答复推定。期限按原服务的保护窗口与影响分类登记，例如会影响下一轮夜间巡视的事项应在原计划巡视时间前决定，普通安排争议以试点事先公布的工作日窗口评价。所有未送达或资金、人手不足的复核也纳入报告。

与安全工程及人机协作研究的关系

STAMP 把安全看成系统在变化环境中维持约束的能力，并用控制和反馈检查责任如何落空。[24] 本文把「原期限内完成」写成一项可量度的照护约束。复核人取得原计划，是责任判断所需的反馈。备用人手能够接单、执行并留下完成证据，是另一条控制路径。两个路径共用同一账号、排班或通讯线时，需在演练中记录共同失效。DECIDE-AI 强调早期临床评价应同时记

录人员与系统如何互动以及表现受何种条件影响。[25] 这里沿用对流程和人员作用的记录要求，测量的是照护接续，而非诊断或治疗质量。上述框架提供设计参照，实际作用仍要按接续结果检验。

交接、漏做照护与电子告警的实证参照

这三步是从照护义务到期时刻往前追踪故障的记录方式。既有研究分别观测了其中的环节。

Starmer 等在九家医院实施 I-PASS 医师交接方案，报告实施前后医疗错误率下降 23%。该研究为交接流程提供实证参照，前后比较的结果来自多项联合改动。[27] Ball 等在英国医院的横断面问卷中发现，自报未完成照护与工作负荷相关，为单独测量交接之后的实际执行提供依据。[28] Wong 等在住院场景外部验证一项商用脓毒症预测模型。该模型在研究使用的一个阈值下漏掉 67% 的脓毒症患者，同时对全部住院患者的 18% 产生告警。这项研究测量告警检出和负担，临床处置与养老照护效果属于另外的结果。[29] 本研究在同一预登记义务上连接时间戳、负责接手的人员和按原期限实际交付的证据，并记录人工替代与退出渠道。

工作流是否按规则运行与照护是否按时完成

Medda 与 Gong 的 GCAC 预印本提出受治理的痴呆居家照护代理。在观测、记忆、计划和执行之间加入确定性权限检验与结果记录，并以 18 条从既有证据提炼的场景轨迹验证系统契约。场景包括陈旧信息、服务失败和同意变更。研究报告 18 条均通过预设检查，同时明确把健康、服务工作量和成本的效果留给前瞻评估。[35] 这些轨迹可用于部署前检验，例如核查撤回同意后系统有无继续执行权限。

本研究向前瞻评估增加两个观察条件。在实验分配前登记照护义务及最迟完成时点，跟踪同一义务在告警、人工接手和实际完成三个环节的证据；若完成状态未知，则报告对接手效果可识别到何种程度，按班次核算还需要多少可核验记录。工程系统中的「已创建交接」和「已关闭工作流」应与值班者实际收到、以及原期限内独立证实照护已完成分别保留。英国 ATTILA 试验比较全套辅助技术及远程照护与基础支持，较完整的配置没有显示延长独立居家时间。[36] 它的参与者和结果不同，但为把流程通过与人获得的结果分别实测提供了经验参照。

公开排班数据复算

已有 PBJ 研究在较早季度记录过美国养老机构日际人力波动及周末人手较少的问题。[30] 这里使用美国联邦医疗保险和医疗补助服务中心（CMS）2026 年第二季度的 Payroll-Based Journal（PBJ）公开文件，在同一机构内比较周一至周五和周六、周日的申报零工时比例。[31–32] 每行是一个机构工作日。原表有 14,474 家机构、1,317,134 行，按机构编号和日期无重复。去除居民数为零的 97 行后，14,474 家机构均保留两类日期，对应 940,741 个周中机构日及 376,296 个周末机构日。先分别计算每个机构在两类日期中的零工时比例，再对机构等权取平均。周中按日历划分，包含落在周一至周五的公众假日。

全部 RN 岗位的口径使用直接护理、行政 RN 和护理主管三列的合计工时，直接护理 RN 的口径仅使用直接护理一列。两种定义都包括 PBJ 同类岗位汇总中的雇员和合同人员。

机构范围	机构数	直接护理 RN 为零	全 RN 岗位为零	仅行政岗位有 RN 工时
全部	14,474	-2.15 [-2.38, -1.91]	+0.68 [+0.60, +0.76]	-2.83 [-3.06, -2.60]
规模不超过 40	1,965	-5.71 [-6.57, -4.88]	+1.26 [+0.94, +1.58]	-6.97 [-7.83, -6.15]
规模超过 40	12,509	-1.59 [-1.81, -1.36]	+0.59 [+0.52, +0.67]	-2.18 [-2.41, -1.95]
规模不超过 60	4,829	-4.58 [-5.09, -4.09]	+1.15 [+0.97, +1.34]	-5.73 [-6.25, -5.24]
规模超过 60	9,645	-0.93 [-1.17, -0.69]	+0.45 [+0.38, +0.52]	-1.38 [-1.62, -1.14]
规模不超过 80	7,463	-3.77 [-4.15, -3.40]	+1.03 [+0.89, +1.18]	-4.81 [-5.19, -4.43]
规模超过 80	7,011	-0.42 [-0.66, -0.17]	+0.31 [+0.24, +0.38]	-0.72 [-0.97, -0.47]
规模不超过 100	10,167	-3.02 [-3.34, -2.71]	+0.88 [+0.77, +0.99]	-3.90 [-4.21, -3.58]
规模超过 100	4,307	-0.09 [-0.35, +0.17]	+0.21 [+0.15, +0.30]	-0.31 [-0.58, -0.04]

机构为重采样单位。同一次抽样保留同一机构的两类日期和三个工时定义，按所列规模分层，独立有放回抽 6,000 次，取 2.5% 和 97.5% 分位数。单位均为百分点，区间反映这一季度相似机构组合下的结果波动。它们为逐项描述区间，不作为多组同时检验。分层界值 40、60、80、100 均以季度非零日居民数中位数计算。

全样本直接护理 RN 的零工时均值从周中 5.84% 降至周末 3.69%。合计各 RN 岗位后，均值从 0.26% 上升至 0.94%。两种口径的差异可以用第三类机构日解释，即直接护理 RN 为零但行政 RN 或护理主管工时为正的日期。该类占比到周末减少 2.83 个百分点。以 100 人分层后，较大机构的直接护理差异区间包含零，所观察的全岗位差仍为正。这只说明按日汇总中的岗位组成和周中周末分布不同。行政岗位正工时不能判断床旁是否有人可用，同一季度的数据也无法给出在地夜班或人工接手的效果。

试点在随机前用两周人员和义务记录，按原期限小时与本地日历固定四层（周中白班、周中夜班、周末或节假日白班、周末或节假日夜班）。每层同时核对直接照护人员、能真正调度的备用人员和独立证据来源。各层分别报告登记数、按时数和未知数。机构配组时优先考虑有无夜班备用、基线完成率和规模。若预试记录显示周末缺失集中，则另加该层核验名额，第 4 节给出预算算法。我们从公开数据借用的是必须分开观察日型及管理和直接执行岗位的理由。夜班缺失率、义务数量和簇内相关以本地预试为准，零申报工时比例不用于代填这些量。

三、比较设计及估计对象

为什么分别分配复核和接手

从必要义务被影响到服务完成，至少经过报警或主动发现、核实依据、获得有权决定的人、能够执行的班组和交付。独立复核首先改变核验依据和决定渠道。备援班组首先改变能否执行和执行所需时间。初次侦测可由同一套人工和技术配置产生，保留漏报事件，并将侦测、签核和调度分别记录。两项干预可以影响其他环节，所以主要比较估计的是被分配到的整套安排。流程记录用于定位停在哪一步，不把事后观察到的中间状态当作随机干预。

若把两项改进一起部署，只能比较联用结果，不能判断哪个环节值得加人。若把报警、决定权、独立证据和班组分别作为四个因素，则要取得更多可独立分配的簇。在本比较中先固定基本侦测流程和既有人工权限，由两因素独立分配识别最早可以实施的两项改进。一个预先可检验的情景是，独立复核增加证据送达而缺少人接手时完成率几乎不变。另一个情景是，新增备援可以直接恢复人工服务，使联用结果接近只增加备援。这两种情景的组间比较不同。试验仍须记录未完成是发生在识别、决定、调度还是交付。

本文提出一项可预先登记的比较研究。先从现有人工照护流程取得基线，再在模拟演练中比较四种安排。进入现场的各组继续享有原有照护、紧急电话和专业接手条件。方案中的「增加」均指在共同底线之上的变化。研究可以从一次设备离线、一次重复或误发提醒、一次当事人撤回附加授权开始，不需要在真实紧急事件中诱发故障。

表 2　相同基本照护条件下的四种试验安排。独立复核按人员、证据与备用渠道逐项验收。

安排	独立复核渠道	已排班且演练过的备用接手
A	通常流程	通常流程
B	增加	通常流程
C	通常流程	增加
D	增加	增加

以预定窗口内完成必要照护的比例为主结果，分母纳入所有原应提供必要照护的事件，包括没有告警的漏报。以独立照护记录、本人回访和复核记录交叉确认，在研究注册中明确缺失记录的处理方式。可另报告适用事件中接手完成时间的中位数及尾部、复核送达率、拒绝额外采集后的服务可达率，以及每百次服务所用的人员分钟数。完成照护的判断应由不直接负责部署的人员按同一标准作出。

以纯演算为例，若四组各有 100 次必要照护，相应接续次数为 80、82、90、94，增加备用接手在常规复核条件下对应 10 个百分点，在增加独立复核时对应 12 个百分点，两者的差为 2 个百分点。这四个数仅说明怎样分别计算单项改进与交互，不能用来预判现实数据。

对本设计有三项直接检验。若独立复核提高了告警和复核率而未减少中断，其作用应限于信息质量。若备用接手只在额外配置大量人手时起效，读者应能从人力成本和排班分布辨认出来。若联用提高总体完成率，却使夜间、独居或数字使用困难者更难得到支持，不能仅凭总体平均值扩大部署。比较时宜按机构或班次处理同组相关性，报告样本量、区间以及人力变动。严重事件罕见，早期阶段可先比较演练中的流程失效，并将现实安全效果留待有统计能力的研究。

复核方案的估计边界

用 R 表示增设独立证据、复核人和渠道，用 S 表示能否按预定授权暂缓当次自动变更，用 B 表示是否部署备用人手。S 表示获配权限，实际有无触发另外记录。现行四格只安排 $(R,S)=(0,0)$ 或 $(1,1)$，每种各与 $B=0,1$ 相交。通常照护、人工决定权与应急入口属于共同条件。对 $r,s,b\in\{0,1\}$，记 μ_{rsb} 为被分配到 $(R,S,B)=(r,s,b)$ 时，目标总体的簇等权平均按时率。固定 $B=b$ 的复核对比是 $\mu_{11b}-\mu_{00b}$。暂缓在有独立复核时的增量是 $\mu_{11b}-\mu_{10b}$，其中 μ_{10b} 没有被分配。即使全部结果观测完整，只知道 $0\le\mu_{10b}\le1$ 时，该增量也只能界为 $[\mu_{11b}-1,\mu_{11b}]$。在实际触发者身上作比较会引入分配后选择。若后续确需估计这一增量，可在已通过安全审查并保留同一人工路径的复核配置内，另行设计权限子试验。

如果现场能安全实施权限子试验，可对每个 $B=b$ 增加 $(R,S)=(1,0)$，与既有 $(1,1)$ 比较授权暂缓的按分配效果，在两个备用条件下共六格。两组均保留通常人工决定权、应急入口和相同复核证据。研究只在两种权限安排均可接受的预先界定情景内分配。六格等分总簇数 K，在各格簇率方差均为 σ_c^2 且互相独立的规划近似下，固定 b 的两格差方差为 $12\sigma_c^2/K$。只关心一个 b 时，可以另做在同一备援条件下的两臂权限试验。未经预先分配的触发者比较没有这项识别保证。

检测、介入、审计和服务送达使用同一义务号。除按时照护率外，下表各项均是过程量，用于解释等待或失败发生在哪一步。这里的结局是照护接续，AI 控制研究用于设计核查过程。[7–8] 具体执行规则和演练判据见附录 A。

环节	分母或起点	测量
检测	事前定义的适用义务	有告警或人工发现的比例、发现时钟；漏报保留
介入	符合事前规则的自动变更	警示到人工响应或授权暂缓的时长，另记无权和误阻断
审计	应留证的事件和操作	独立渠道取证率、理由和签核可核验比例、缺失数
照护送达	全部登记的主分母	按原期限完成比例；故障或退出后的同一标准

两臂、四格和阶梯式上线

先取总共有 K 个独立分配簇，且各臂均衡、簇率方差为 σ_c^2。若只决定在既有复核条件下增加备用人手，两臂同期试验估计 C 减 A；两组各有 $K/2$ 簇，差的方差为 $4\sigma_c^2/K$。四格设计的备用主对照是 C、D 平均减 A、B 平均，各格权重绝对值为二分之一，每格簇数为 $K/4$，方差也为 $4\sigma_c^2/K$，但估计的是跨两个复核条件平均的效果。若两复核条件下的效应不同，两个目标自然不同。在机构已经承诺维持固定复核流程时，两臂直接回答是否应加聘备用人员；若复核和备用均可能变更，四格可同时观察二者各自的效果和联用差。暂缓权限只有在具备独立复核时才有定义，若要估计该任务应采用前述六格并重新计算规模；直接做八格会包含不具备独立复核却由复核者暂缓的无定义格。四格交互的权重绝对值均为 1，方差为 $16\sigma_c^2/K$；在同方差、同目标差异的比较中需约四倍簇数。按基线簇率作事前分层可进一步提高精度，收益须由预试校准。

阶梯式设计随机分配上线次序，可在逐步部署时比较同一时期的预定状态。其簇数规划还依赖时期数、各期义务数、簇内跨期相关和是否有延迟效应。[21–22] 若容许每一种上线序列在各时期有任意不同的未经干预趋势，平均上线指标是序列及时期的确定函数；给该指标的系数加一个常数、再从相应序列时期的趋势减去同一项，拟合均值不变。故无法从这种无限制的模型中识别上线效果。共享班组逐步覆盖其他簇，也会改变分期所估计的安排。部署能够同期分配时，保留四格主设计；若只够回答一个已固定复核条件下的备用问题，则用两臂并明示目标。只有分期不可避免且能核对同期对照、时间变化和跨期延迟时，才另订阶梯式方案。

四、缺失结局的识别分析

符号	含义
g, j	分配格和格内分配簇
N_{gj}, T_{gj}, U_{gj}	登记、确认按时、未知的照护义务条数
P_{gj}, L_{gj}, V_{gj}	完整按时率及其可达下、上端点
a_g, w_{gj}, b_{gj}	对比系数、预先固定的簇权重、二者之积
θ_L, θ_U	给定观察记录时可达的对比下、上端点

固定权重的锐界沿用有界结局的最坏情形原理。[33] 随机抽查使用无放回抽样的超几何分布，反演本身是标准统计方法。单个对比的端点是把 Manski 式最坏赋值应用到预先固定的簇权重。Imbens–Manski 讨论的是怎样让置信区间覆盖部分识别的参数，本研究另将已登记记录产生的确定性宽度和分配误差分开。本文新增的设计量是：给定总体和各班次允许的确定性宽度，至少需要核实多少条可达记录，以及核验预算不足时哪一层仍不能分辨。有限总体的抽查误差和跨簇随机分配的抽样误差需分别计入。[19,23]

从一条未知记录开始

假设四格各登记两条义务。A、B、C 各确定一条按时，另一条不按时。D 有一条按时，一条未知。若 D 的未知记录最终不按时，复核主效应、备用主效应和交互均为零。若它按时，两项主效应各为 25 个百分点，交互为 50 个百分点。因此两项主效应各自的端点是 0 和 25，但不能用同一个真实记录得到「复核为 0、备用为 25」。这里的锐界指在给定记录下确能达到的最小值和最大值。联合区域保留同一缺失记录同时决定多项比较的关系。

在随机前固定事件册、核查窗口和主要权重。主要人群为参与分配且有预定照护义务的机构或团队，以簇等权，目标是随机分配总体中典型簇的平均按时完成比例差。若把簇换成义务等权，另报明确的义务总体及其分母。大机构可在义务等权结果中贡献更多。起初有义务的簇若全无可核实结果，整个簇按 $[0,1]$ 计入。新增或终止义务按同一规则核准并另报。处理后决定簇是否纳入会改变估计对象。

若每支团队获得同样一条备用排班，且决策把团队按时率每提高一个百分点赋予相同价值，

则净改进按团队平均，与簇等权主对照一致。若排班成本在团队间明显不同，另将固定成本与完成收益逐团队登记。若拨款按全覆盖地区能够多完成多少项义务分配，主要量应改为义务等权。义务等权回答从已登记的义务中抽到一条时，两种安排的接续差异，用 $w_{gj}=N_{gj}/\sum_{j}N_{gj}$。当簇规模与结果相关时，这两个量会有不同数值甚至不同符号。[26] 例如两臂各有一小簇（5 条）和一大簇（35 条），对照的按时数是 4、28，干预是 5、25。簇等权差约为正 5.7 个百分点，义务等权差为负 5 个百分点。二者使用同一组完整记录。如果团队等权改善而义务等权方向相反，须核对大团队的负担、班次和完成率，然后再讨论是否按同一模式扩展。按估计方差倒数作权重会偏重结果精度较高的簇；若较大的机构精度更高，该权重会系统改变问题所指向的人群。如果考虑这类权重，只把事前由外部资料固定的方案作为次要结果，并公布实际权重与对象。用本试验的完成或缺失结果事后给主对照改权，会改变权重及其推断。

记 g 为试验格，j 为该格内的簇，N_{gj}、T_{gj}、U_{gj} 分别为预登记义务数、确认按时数和未知数。P_{gj} 是全部登记义务的真实按时率，$L_{gj}=T_{gj}/N_{gj}$ 和 $V_{gj}=(T_{gj}+U_{gj})/N_{gj}$。每条义务可再用格内簇 (g,j) 下的局部索引 r 表示，$1\le r\le N_{gj}$。格内固定权重 $w_{gj}\ge 0$ 且 $\sum_{j}w_{gj}=1$，主要分析用 $w_{gj}=1/J_{g}$。a_{g} 是对比的有符号系数。备用接手主效应的系数为 $(-1,-1,1,1)/2$，联用交互为 $(1,-1,-1,1)$。

命题 1（固定权重的锐界）。对任意实系数 a_{g}，定义 $\theta=\sum_{g,j}a_{g}w_{gj}P_{gj}$，并令 $b_{gj}=a_{g}w_{gj}$，$b_{+}=\max(b,0)$，$b_{-}=\min(b,0)$。仅约束已观察二值结果，且未对未知结局施加联合限制时，可达的最小值和最大值为

$$\theta_{L}=\sum_{g,j}(b_{gj,+}L_{gj}+b_{gj,-}V_{gj}),\qquad \theta_{U}=\sum_{g,j}(b_{gj,+}V_{gj}+b_{gj,-}L_{gj}).$$

区间宽度为 $\sum_{g,j}|b_{gj}|U_{gj}/N_{gj}$。这些是锐端点；有限个二值未知量的实际可达集合可以有离散空隙。

证明。将每个未知结局写为独立坐标 $y_{i}\in\{0,1\}$（这里的独立是可分别赋值，非概率假设）。展开 $\theta=\sum b_{gj}L_{gj}+\sum_{i}(b_{g(i)j(i)}/N_{g(i)j(i)})y_{i}$。求最小值时，取正系数的未知为 0、负系数的为 1。求最大值时反过来。两种赋值均满足全部已知记录，故端点可达。由两个端点率的定义，$V_{gj}-L_{gj}=((T_{gj}+U_{gj})-T_{gj})/N_{gj}=U_{gj}/N_{gj}$。将最大值与最小值按簇相减，第 (g,j) 簇贡献 $(b_{gj,+}-b_{gj,-})(V_{gj}-L_{gj})=|b_{gj}|U_{gj}/N_{gj}$。按所有簇求和，即得宽度。证毕。

备用接手主效应的宽度化为 $(u_{A}+u_{B}+u_{C}+u_{D})/2$，其中 $u_{g}=\sum_{j}w_{gj}U_{gj}/N_{gj}$。两个边际组的未知率若都是 5%，识别宽度已达 10 个百分点。若需要 5 个百分点的宽度预算，两个边际组各约 2.5% 是对称计划。一条新近查实的记录会将给定对比的宽度缩小 $|b_{gj}|/N_{gj}$，幅度与查得按时或未按时无关。用它可估算追加核查的最大收益。实际核查仍须保留全簇覆盖，并单独检查夜间、独居或人工渠道的记录缺失。

推论（多个固定对比的可达投影）。令 $\mathcal{U}=\{(g,j,r):1\le r\le U_{gj}\}$ 为全部未知记录的索引集，$y_{gjr}=1$ 表示该条按时。对复核、备用与交互三个对比，令 $\boldsymbol{a}_{g}$ 为格 g 的三个系数组成的列向量，则该未知记录的系数向量为 $\boldsymbol{v}_{gjr}=\boldsymbol{a}_{g}w_{gj}/N_{gj}$。把未知均记为未按时所得对比向量记为 $\boldsymbol{\theta}_{0}$。对任意固定的方向 $\boldsymbol{t}\in\mathbb{R}^{3}$，

$$\max_{y\in\{0,1\}^{\mathcal{U}}}\boldsymbol{t}^{\top}\left(\boldsymbol{\theta}_{0}+\sum_{(g,j,r)\in\mathcal{U}}y_{gjr}\boldsymbol{v}_{gjr}\right)=\boldsymbol{t}^{\top}\boldsymbol{\theta}_{0}+\sum_{(g,j,r)\in\mathcal{U}}\max\{0,\boldsymbol{t}^{\top}\boldsymbol{v}_{gjr}\}.$$

证明。展开内积，常数之外每项只依赖一个二值坐标。该项系数为正时取 1，为负时取 0，为零时两取值均可。可以同时采用这些选择，因为尚无约束把未知记录绑在一起。取 $-\boldsymbol{t}$ 可得下投影。单项端点可以由互斥的赋值达到，联合比较时使用所关心的方向。证毕。

用附加资料收紧未知数

如果各个未知记录仍是自由的二值，则命题 1 已给出锐界。紧界要依赖新资料或明示的限制。举例来说，在事前同一类义务内，已观测按时率为 p_{obs}，可把未知记录的实际按时比例限制在

$[p_{\mathrm{obs}}-\delta, p_{\mathrm{obs}}+\delta]\cap[0,1]$，将上下乘以未知数，再分别向上和向下取整。若有 20 条未知、观测率 0.75，则 $\delta=0, 0.10, 0.20$ 分别限制未知成功数为 15、13 至 17、11 至 19。这个假定限制的是该批实际条数。即使缺失在概率模型下随机出现，有限批次的观察率和未知率仍可不同。小批次的整数取整还可能使假设无可行赋值，遇此应报告矛盾。整簇失访时没有簇内参考率。若借助同类幸存簇，应按预设场景另报其可交换性和差异容许量。主分析仍在缺乏约束的情况下保留全幅。

若独立名册、抽查或敏感性假设将某一事前分层的未知按时总数限制在整数 $[l,h]$，可精确得到加权端点。设该层每条未知记录的对比系数为 $c_i=b_{gj}/N_{gj}$，固定共有 j 条按时时，取 j 个最小系数之和得到下端，j 个最大系数之和得到上端。允许 j 在 $[l,h]$ 中变化，下端取 j 为把负系数个数截到此区间，上端把正系数个数截到此区间。证明是加入更早排序的记录总能替换更差的记录，而最小值随加入负系数而下降，加到正系数才上升。上端同理。不相交分层的端点相加，重叠限制须共同求解。附录 D 穷举短记录和正、负、零系数校验了这一算法。

随机核查未知记录

从一份封存的未知名册 U 条中，不放回等概率抽 k 条，由独立资料核定原期限内是否完成。设名册内实际有 S 条按时。理想的核定成功数 X 服从超几何分布 $\mathrm{Hypergeom}(U,S,k)$，这是由选中名单的随机性产生的。枚举 $S=0,\dots,U$，保留同时满足 $\Pr_S(X\ge x)\ge\alpha/2$ 和 $\Pr_S(X\le x)\ge\alpha/2$ 的值。每个尾部误拒概率最多 $\alpha/2$，故此区间在固定名册及其真实结果下覆盖至少 $1-\alpha$。如果抽中后仍有 u 条无法核实，把理想统计量放宽为 x 至 $x+u$，再取这些可行统计量对应集合的并。它包含完整核查本可给出的集合。抽查的置信界与命题 1 的确定性端点分别报告。

例如某层有 100 条登记义务，原来确认 50 条按时、10 条未按时，40 条未知。随机抽核 20 条确认 14 条按时、6 条未按时。只合并这些核定结果，实际按时率在 64% 至 84%。反演超几何分布得原 40 条未知中的按时数在 21 至 33，对应 95% 抽查置信区间为 71% 至 83%。若抽中后还有两条无法核实，其余确认 14 条按时，抽查区间扩为 71% 至 86%。程序逐一遍历 40 条未知的可能成功总数核对覆盖。多层抽查时，给各层分配的误差水平须合计在预先给定的抽查误差预算内。若还估计分配总体中的完整结果均值，可将两类误差预算相加。例如独立区组的完整对比都在 $[-1,1]$ 时，先由抽查得到样本对比的外区间，再按下一节的有界均值半径向两边扩展。由并集界，总错误率至多为抽查误差与扩展所用的总体抽样误差之和。

若目标是缩小一个既定对比的确定性区间，每条成功核验的收益是 $|c_i|$。成本相同、独立核验且逐条可确定时，按收益从大到小选可使同样条数的宽度减少最多。成本不等时，可在工时预算和重点人群最低覆盖量下最大化所选收益之和。真正的记录消失还取决于证据可达性，故须记录实际完成核验的条数和成本。定向核验只更新该条结果。要把抽查所见推广到尚未核验的记录，应使用已知抽样设计。

先保证班次可分辨，再分配核验名额

按原期限所在的日型与班次预登记分层 d。在每个可报告的层中，令 N_{gjd} 为格 g、簇 j 在该层的登记数，w_{gjd} 为该层在格内预定的簇权重，格内权重和为 1。只有事先确定在该层有登记义务的簇进入该层分母。未知记录 r 的加权宽度贡献为 $x_r=|a_g|w_{gjd}/N_{gjd}$。记该层确定性界限宽度为 W_d。如果要报告一个固定班次构成下的总体对比，设 $\pi_d\ge0$、$\sum_d\pi_d=1$，令 $\theta^{\pi}=\sum_d\pi_d\theta_d$。班次不重叠且各未知记录可独立赋值，所以其锐界宽度恰为 $\sum_d\pi_dW_d$。π_d 由随机前的登记构成或事先明示的政策目标给出。实际交付后的班次构成另报。

命题 2（分层与总体精度约束下的最少核验）。每条核验成本相同，并能独立确定目标记录的结局。已知不可核验记录的权重也计入 W_d。若每层宽度须不超过 L_d、总体宽度不超过 L，可先在各层按 x_r 从大到小取最短前缀使 $W_d\le L_d$。再跨层按 π_dx_r 从大到小补足总体约束。此方案达到两类目标所需的核验条数最少。若某层取尽仍不达标，或取尽所有记录仍达不到总体要求，即使使用全部可核验记录也无可行方案。

证明。给定在某一层核验的条数，其宽度减量不超过该层同条数的最大权重之和。用这些较大的权重替换原选记录，层宽度和总体宽度只会减少。因此任一最少条数的可行方案可取为各层前缀，并必含各层为 L_d 所需的最短前缀。在这些前缀固定后，同样数目的追加核验对总体宽度的减量之和至多等于剩余记录中最大的 $\pi_d x_r$ 之和。在同一层中 π_d 固定，排序时可将靠前者置于相等权重记录之前，所以跨层补足规则仍可保持各层前缀。由此最早达标的追加条数也是下界。证毕。

算例中各层依次是周中白班、周中夜班、周末白班、周末夜班，总体权重为 45%、20%、25%、10%。未知记录数分别为 8、5、4、3，每条带来的层宽度减量分别是 0.3、0.6、0.5、1.2 个百分点。只要求总体宽度不超过 1.9 个百分点时，核验五条周中白班记录即足够，周末夜班的宽度仍为 3.6。若另要求周中夜班和周末夜班宽度分别不超过 1.8、1.2 个百分点，最少核验六条，按层为 2、2、0、2 条。合成穷举对照 157,464 种目标与记录配置，包含 52,488 种不可行情形，附录 F 提供程序。若核验结果仍未知，将其保留在宽度里，并用剩余名册重算。此规则只决定定向核验的最少条数，不把定向查到的比例推广给其他记录。要推断未核验结果的比例，使用前述独立的事前随机抽查方案。

核验费用因记录而异

等成本情形下，缩小宽度最多的记录往往值得先核查。核验的可靠性和成本是不同条件。这里仅考虑在选入后有独立证据且可确定结局的记录，永久无法证实的部分计入残留。若一项核验只能在某个概率下成功，它不能在此判据中提前算作已缩小的宽度。

命题 3（费用不同的最小核验成本）。令 B_d 为第 d 层初始宽度，U_d 为所有可核验记录均核定后仍不可消除的宽度。可核验的第 i 项缩减该层宽度 $v_{di} \ge 0$，费用 $c_{di} \ge 0$，其中 $B_d = U_d + \sum_i v_{di}$。总体宽度使用非负权重 w_d。令分层宽度上限为 b_d，总体上限为 b。若某个 $U_d > b_d$ 或 $\sum_d w_d U_d > b$，无可行方案。否则，记

$$t_d = (B_d - b_d)_+, \qquad t = \Big(\sum_d w_d B_d - b\Big)_+ .$$

目标是以最低总费用获得每层至少 t_d 的缩减和总体至少 t 的加权缩减。

先在每一层用 0–1 背包递推，保留每个可达缩减 x 对应的最小费用 $F_d(x)$。初始只有 $F_d(0) = 0$。处理每条记录时，从上一步的每个状态复制「不核验」状态，也可转移到 $x + v_{di}$ 并增加 c_{di}。相同 x 只留最低费用，完成后丢弃 $x < t_d$ 的状态。再按层合并，以 $G_d(z)$ 表示已通过前 d 层各自宽度要求、累计获得截顶总体缩减 $z \in [0, t]$ 的最低费用。从 $G_0(0) = 0$ 开始，对每个上一层状态 z 和可行本层状态 x，用 $G_{d-1}(z) + F_d(x)$ 更新状态 $\min\{t, z + w_d x\}$。最后一个层数记为 D，所求为 $G_D(t)$，并可按保留的前驱恢复应核验的记录。

证明。任一核验集合唯一分解为各层子集。层内递推每次只添加一项并从上一步更新，因此遍历该层可行子集，保留同一缩减的最小费用不影响后续。各层阈值已由 $x \ge t_d$ 检验，跨层递推只需累加每层对总体的贡献，截顶部分不会成为后续约束。较大缩减且不更贵的状态可以丢弃其所支配的状态。按两次递推归纳得到最小费用。若固定残留不超任何上限，选择全部可核验项即为一组可行解。

例如一层需要缩小 10 个记账单位，有三项可各缩小 7、5、5，费用为 6、3、3。按缩减幅度从大到小取会花费 9。最优是后两项，费用为 6。宽度用精确有理数或统一整数刻度计算。状态数可能随记录数增长，实际大规模数据可用整数规划。附录穷举对照 12,000 组小型配置，其中 6,690 组无可行解。扩展的目标是在一个预先固定的单项总体宽度与分层宽度下计最小成本；多种总体对比各自设阈值时需要多维状态。

从缺失界限到抽样区间

将四格组成区组，每个完整区组各有 A、B、C、D 一簇，分别计算区组的 θ_L 和 θ_U。以区组为单位对两个端点作单侧区间，各用 2.5% 的尾部水平。若各单侧区间对其总体端点的覆盖至

少 97.5%，由并集界可知外侧区间覆盖整个总体识别区间至少 95%。组间独立、数量充分且无单一组支配时可用区组均值和 t 近似。少组与整簇失访时的实际覆盖须另核查。该构造与部分识别研究对识别区间和抽样区间的区分一致。[19] 只有外侧下限高于零时，才报告在最不利未知分配下仍有改善。对整个区间的覆盖和只覆盖某一个真实差值分别计算。

若完整区组彼此独立，备用主对照的每个区组端点都属于 $[-1,1]$。令 J 为区组数、α_* 为单侧误差预算。Hoeffding 不等式给出同时覆盖两个均值端点的保守外区间，只需从样本下端均值减去 $\sqrt{2\log(1/\alpha_*)/J}$，给样本上端均值加上同一项，将总误差控制在 $2\alpha_*$。例如 $J=20$、$\alpha_*=0.025$ 时，每侧约 0.61，说明仅靠有界性，即使取得理论覆盖也可能过宽。模拟中使用的 t 区间需要对小区组数和失访下的覆盖进行校准。

用 v1.6 的八簇合成记录，枚举 11 个未知事件的 2,048 种赋值，得到备用接手主效应的区间约 2.07 至 18.65 个百分点。这里用于核对锐界和代码的算术结果，簇数和比例均属合成。

五、样本量与合成模拟

样本量怎样估算

备用接手的边际主效应作为唯一主要检验。设两边际组的按时率为 $p_0=0.80$ 和 $p_1=0.90$，$\bar p=(p_0+p_1)/2$。假定不同个体独立、各边际组为二项比例。零假设用合并方差，备择用两个组别方差，未对连续性作校正。等组、双侧 5%、把握度 80% 的两比例正态近似，每个边际组所需独立事件数为

$$n_0=\frac{\left[1.96\sqrt{2\bar p(1-\bar p)}+0.84\sqrt{p_0(1-p_0)+p_1(1-p_1)}\right]^2}{(p_1-p_0)^2}\simeq 199.$$

以机构或互不串组的团队为分配簇。若各簇有 m 个合格事件、同簇二项结果的相关系数为 ρ，簇总和的方差含 m 个方差项与 $m(m-1)$ 个协方差项。因此每簇的方差膨胀因子为 $D=1+(m-1)\rho$。每个边际组包括两个设计格，每格簇数向上取整为 $\lceil n_0 D/(2m)\rceil$。差异用按原分配估计的照护完成风险差，以预登记方法处理簇相关，报告区间与缺失界限。簇大小不等时，需用预试的簇大小分布重新模拟。

严重事件很少时。 上述比例正态近似用于常见完成结局。低频严重事件按分配簇另报有无、次数和暴露窗口。举例，若 K 个相互独立的簇在相同观察窗内各有一次「至少发生一项严重事件」的机会，发生概率为 p_j，完全没有此类记录的概率是 $\prod_j(1-p_j)$。由算术与几何均值不等式，它至多为 $(1-\bar p)^K$，$\bar p=K^{-1}\sum_j p_j$。因此当全部核实且零个簇报告严重事件时，对平均簇风险的单侧 95% 上界可取 $1-0.05^{1/K}$。即使有 20 个互相独立的簇，上界仍约 13.9%。有失访或共同故障路径时须重新处理独立性和未知结局。这个量衡量每簇观察窗内至少一次事件的风险，不能换算为单条义务的风险。

样本量敏感性与试验边界

联用对照为 $I=p_D-p_B-p_C+p_A$。若每格有 n 个有效独立事件，四格近似独立，则 $\operatorname{Var}(\hat I)\simeq\sum_g p_g(1-p_g)/n$。使用上文合成的四个概率，方差分子为 0.454。用同样的双侧检验粗算，交互差为 10 个百分点时每格需要 356 个独立事件，为 5 个百分点时每格需要 1,424 个。此处固定合成概率提供方差，只用于设计敏感性，真实基线须重算。

每格需将上述独立事件需求乘以 D 并除以 m，再向上取整。以 10 个百分点交互为例，$m=20$ 时，$\rho=0.01,0.05,0.10$ 对应每格 22、35、52 个簇。边际主效应在相同三种情景下分别为 6、10、15 个簇。若 $m=10$、$\rho=0.05$，边际主效应需要每格 15 个簇，交互需要 52 个簇。若独居者占事件四分之一且各簇均有约 5 个该亚组事件，在相同方差和 $\rho=0.05$ 的情景中，只估计亚组自身的 10 个百分点边际效应也需每格约 24 个簇，亚组交互需约 86 个簇。比较两亚组效

应的差还需独立的把握度计算。亚组估计和交互应报告区间，只有按预先确定的效应量达到所需样本后才能作为确认性结论。

这些簇数还须根据可获得的独立簇、簇大小离散程度和分配限制重算，少簇情形考虑有限样本校正。若无法取得足够独立的机构或团队，先做可行性研究。若排班需要分期上线，可评估随机阶梯式次序，但必须另估时间趋势和学习对后期班组的影响。对低频严重事件实行独立监测。模拟流程通过或观察期间零严重事件均不足以估计长期伤害率。

从报名机构数到可随机的簇数

招募前根据实际资源画无向图。点是候选机构或团队，若两个点共用能改变对照组照护的复核者、待命人员、调度台或记录权限，则连边。要让每条共享关系留在同一分配臂，相连两点须采用同一分配，沿任一路径传递即得整个连通分量同臂。把每个连通分量内的候选点合并，才得到这套共享规则下可独立分配的最细单位。分量间无相应外溢仍是设计假设，需演练核查。例如 20 个报名团队若各五个通过共享排班连成四组，按该图只有四个分配单位。即使登记义务很多，也只能支持流程试点。若合并改变每簇体量或覆盖人群，应提前重新指定目标人群、对比权重与试验容量。

沿用上文的合成基线，以每簇 20 项、$\rho = 0.05$、效果差异目标 10 个百分点的等容量规划例为例，双侧 5% 和 80% 把握度下可按以下顺序决定。若主要决策是在现有复核安排下增加人手，只随机备用接手，两臂各约需 20 个独立簇，合计 40。若要估计跨复核条件的边际效应，四格各约需 10 簇，合计 40，需确保每格安排能落实。若交互本身决定部署，四格各约需 35 簇，合计 140；$\rho = 0.10$ 时升至每格 52 簇、合计 208。这些是平衡分配与完整结局的粗算。独立簇低于重算后的要求时，采用匹配主要问题的简约设计，或先做流程与测量预试。对于计划规模，应按合并后的簇规模、失访及预定随机规则模拟覆盖。小样本随机化检验可检验明确的锐零假设，覆盖整个缺失结局识别区间需另行校准。

簇大小、整簇失访和外溢的压力测试

采用两个预先定下的比较值：相互隔离且在同一人群构成下的 10 个百分点备用接手效应，以及每一情景下实际分配环境对应的均值差。先取 A、B、C、D 的按时概率为 0.78、0.82、0.88、0.92。每簇用一个两级构成变量使其名义概率上下浮动 0.06，再按 beta 二项模型生成事件。条件簇内相关为 0.05。均匀小组大小取 20，大小不等时独立等概率取 5 或 35，平均仍为 20。构成浮动 0.06 和额外 3% 的整簇失访分别测试病例组合波动和整套记录随场地离开的情形。它们是压力测试设定。在地预试将按班次直接估计两类参数，再决定使用哪些模拟情景。

合成情景按下面矩阵给出结果丢失概率；行次序与 A、B、C、D 一致。条件为该事件的真实按时状态，模拟时先生成状态，再遮蔽记录。第一列是 $m_1 = \Pr(\text{未知} \mid \text{真实按时})$，第二列是 $m_0 = \Pr(\text{未知} \mid \text{真实未按时})$。这是每一格分别使用的概率，未对两个格之间作加总：

	m_1	m_0
A	.020	.050
B	.020	.050
C	.025	.100
D	.025	.100

第 3、4、7 行使用这组事件遮蔽率。第 4 行另使 C 和 D 各有 0.03 的概率丢失整簇；A 和 B 的整簇丢失概率为 0。其余各行的事件、整簇概率均为 0。给定真实按时率 p 和整簇丢失概率 ω，格内期望下端为 $(1-\omega)p(1-m_1)$，上端为 $\omega + (1-\omega)[p + (1-p)m_0]$。表格直接由附录运行输出生成，沿用 v1.7 的参数与随机种子。

外溢情景将 A、B 的按时概率加 0.04，C、D 减 0.02，代表共享增益和拥挤。构成失衡情景中，高基线率簇在 A、B 的概率为 0.75，在 C、D 为 0.25，模拟分配后筛选所造成的失衡。两种情景的实际对比均为四个百分点。真零情景让 C、D 的基础概率分别降低 0.10，使备用主效应为零。其余输入与前述相同。

每格簇数分别取 10、20、40、80，每个情景和规模独立模拟 4,000 次。基于区组均值和 t 近似计算外侧区间。表中左列识别端点由生成概率直接求得；「下限大于零」是模拟频率。在接近 50% 时，该频率的 Monte Carlo 标准误约 0.8 个百分点。

表 3　每格指定簇数时，外侧下限高于零的模拟频率（%）。

合成情景	总体缺失界限（百分点）	20 簇	40 簇	80 簇
1 等大簇，记录完整	10.0 至 10.0	92.7	99.9	100.0
2 不等簇，记录完整	10.0 至 10.0	79.2	98.8	100.0
3 不等簇，事件差异缺失	6.8 至 12.6	48.6	78.6	97.7
4 上行再加备用组整簇失访	4.1 至 12.9	18.6	31.2	51.9
5 不等簇，共享备用与拥挤	4.0 至 4.0	20.9	38.1	67.0
6 不等簇，分配后构成失衡	4.0 至 4.0	20.2	38.2	65.8
7 真零效应，事件差异缺失	-3.0 至 3.6	0.4	0.1	0.0

簇大小不等时，20 簇每格检出正下限的频率从 92.7% 降到 79.2%。加入事件差异缺失后，40 簇也仅为 78.6%。再加入整簇失访，80 簇时约为 51.9%。80 簇在共享备用和分配后失衡情景仍可经常得出正的实现对比，但外侧区间覆盖隔离条件目标 10 个百分点仅为 5.1% 和 5.8%。这种目标偏离不能靠增加簇数弥补。取 20 簇时，各情景的外侧区间对其自身识别区间的覆盖频率介于 93.6% 和 95.7%。这包括整簇未知下低于名义 95% 的情形，提示正式研究应在筹备阶段校准区组数及推断方法。这些数值仅用于比较设计决策。模拟的绝对检出率取决于名义概率、构成分布和实际缺失模式。

改变簇内相关和规模分布

在表 3 的差异缺失情景上，另行改变条件簇内相关 $\rho = 0.01, 0.05, 0.15$，对应 beta 形状参数之和约 99、19、5.67。在给定簇构成和簇大小 N 时，若均值为 p，完整簇率方差为 $p(1-p)[\rho + (1-\rho)/N]$。簇等权时，与规模无关的 p 下应平均 $1/N$，而不能只把平均规模代入。两点分布取 5 或 35 条，等概率。长尾分布取 2、8、80 条，概率分别为 0.4、0.4、0.2。平均规模都为 20，平均倒数约为 0.114 和 0.253。合成数值不取自养老现场。两分布各模拟有无备用格 3% 整簇失访，共 12 个情景、四种规模，每情景和规模 4,000 次，共 192,000 次。

表 4　不同条件相关及簇规模时，外侧下限高于零的频率（%）。

条件相关	规模分布	差异缺失 20 簇	差异缺失 80 簇	整簇失访 20 簇	整簇失访 80 簇
0.01	等概率 5、35	54.7	99.2	22.9	55.4
0.01	长尾 2、8、80	32.7	88.5	14.8	39.1
0.05	等概率 5、35	47.8	97.7	20.0	51.4
0.05	长尾 2、8、80	30.0	85.1	12.9	37.9
0.15	等概率 5、35	37.0	90.4	14.6	42.7
0.15	长尾 2、8、80	26.2	77.2	12.9	33.4

表 4 中两种规模分布在所有相关设定下的总体识别宽度相同。差异缺失为 5.85 个百分点，加整簇失访后为 8.75 个百分点。主要变化发生在抽样精度。以条件相关 0.05 为例，每格 80 簇且无整簇失访时，正下限频率由 97.7% 降至 85.1%。加整簇失访后，两分布分别为 51.4% 和 37.9%。每格 10 簇时，有整簇失访情景的外区间覆盖介于 92.8% 和 94.5%。这些结果来自特定生成过程。正式计算前，从各场景的独立登记表记录每簇义务数、同班与跨班完成的相关，以及缺失是否连带消失。按整体簇重采样其规模分布，在合理的相关和缺失网格上复算，重点看能否达到预定识别宽度及区间覆盖。

分配前把共享夜班、复核人员、排班和应急联络的机构合并成分配单位，预登记照护义务及其严重程度，再按基线完成率、规模配区组。对偏离安排的情况，保留原分组和全部登记义务，公开独立簇数、共享路径、跨组接手次数和各组退出。若实地仍必须共享备用团队，需按照覆

盖环境注明研究所估计的安排；隔离条件下的效果应另行设计识别。样本量和事件分母按簇随机试验的报告建议预先列出。[20]

六、在依赖形成之前保留回路

可以先让 AI 支持记录和提示，在独立记录中确认它能及时送达，再试行有限授权。每次扩大范围，都应核对哪些人工技能、岗位和检修能力仍在。一次试验顺利并不能说明失效时有人可以接手，维护人员离岗、服务商更换、夜间欠员或家属无法接电话，都会改变已经测得的保护能力。暂停或缩小部署后，应检查老人是否收到解释，是否知道谁负责下一步，以及缺失的照护何时补上。

更强的智能可能给保护者带来新的考验。本文只从已有服务中给出一个可检验的起点：保护如果依赖单一系统的同意，或者改变决定之后没有人能提供替代服务，就难以独立纠正。先把记录、申诉和接手建在现有组织里，可以使人们在新能力进入生活之前具有实际选择。人工智能是否真正带来安全，最后要看人有没有得到依然可以拒绝、核查和继续获得的帮助。

同意、数据和实际可及性

现场研究应按服务与研究用途分别取得适当授权，说明哪些数据将被看见、由谁复核、何时删除，以及拒绝研究后仍可获得哪些服务。对于依赖数字设备有困难的人，提供现场、电话或由其选定的协助者转达的渠道。试验记录按数字使用能力、居住情况和是否拒绝附加采集分组，对服务到达、等待和申诉分别报告。在研究伦理审查中，应为掉线、无法联络和不可接受的服务延误预先设停试条件。涉及医疗健康等敏感个人信息时，中国《个人信息保护法》规定了目的、必要性及严格保护要求，具体处理还需核对适用的处理依据和同意规则。[11] 若产品承担医疗器械功能，须按适用法域的预定用途和分类判断。欧盟《人工智能法》第 6 条及附件一提供了结合产品类型与合格评定要求分类的例子。本文给出的照护结果可补充评估，法定符合性仍由相应程序决定。[12]

在解释风险后，本人仍可选择不使用设备、调整可自由安排的活动或请求人工服务。澳大利亚养老质量监管机构把获得信息及选择所愿承担的合理风险纳入这一义务。[15] 本研究以此作为伦理参照。在场人员用本人能理解的方式说明辅助方案、已知风险、替代路径和撤回方式。个人作出选择时可以获得支持。存在争议时应登记本人意愿和负责复核者，允许本人或其受托人从可用渠道提出更正，并在复核期间保留基本照护。各地具体同意、代理决定与救济程序须以当地制度为准。

致谢与 AI 参与说明

王建明提出三部曲的核心问题与价值取向。感谢 Kimi 与千问参与讨论和早期审阅，ChatGPT 协助初稿组织。Codex 推导并穷举复核分层核验方法及异成本扩展，完成公开工时的机构配对复算，并协助收束养老照护主线、核对公开来源、设计接续评估与对照研究，并推导缺失锐界与随机核查区间，复算压力测试和演练判据。文中演算为说明研究方法所设，未使用个人照护记录。

参考文献

[1] 工业和信息化部办公厅、民政部办公厅（2025）。《关于开展智能养老服务机器人结对攻关与场景应用试点工作的通知》。官方文本

[2] 工业和信息化部办公厅、民政部办公厅（2025）。《关于公布智能养老服务机器人结对攻关与场景应用试点项目名单的通知》。官方文本

[3] 王建明（通讯人，2026）。《当 AI 与具身机器人承担更多生产，人靠什么生活？智能时代的经济转型与分配模式设想》。aiXiv.261027.000001

[4] 王建明（通讯人，2026）。《当智能成为共同能力：人类与 AI 怎样走向新的文明可能?》aiXiv.261103.000002

[5] 国务院办公厅（2020）。《关于切实解决老年人运用智能技术困难的实施方案》，国办发〔2020〕45 号。文件全文

[6] Royal Commission into the Robodebt Scheme (2023). *Report*, recommendations 17.1–17.2. Official report

[7] Greenblatt, R., Shlegeris, B., Sachan, K., & Roger, F. (2024). *AI Control: Improving Safety Despite Intentional Subversion*. ICML 2024

[8] Stickland, A. C., et al. (2025). *Async Control: Stress-testing Asynchronous Control Measures for LLM Agents*. UK AI Security Institute

[9] WHO, OECD, & World Bank (2018). *Delivering quality health services: a global imperative for universal health coverage*. WHO

[10] World Health Organization (2018). *Continuity and coordination of care*. WHO

[11] 全国人大常委会（2021）。《中华人民共和国个人信息保护法》，第二十八至三十条等。官方文本

[12] European Union (2024, as amended). *Regulation (EU) 2024/1689*, Article 6 and Annex I. EUR-Lex

[13] Bainbridge, L. (1983). *Ironies of automation*. Automatica, 19(6), 775–779. 期刊原文

[14] Mosier, K. L., Skitka, L. J., Heers, S., & Burdick, M. (1998). *Automation Bias: Decision Making and Performance in High-Tech Cockpits*. DOI

[15] Aged Care Quality and Safety Commission. *Choice, independence and quality of life*. Official guidance

[16] Haggerty, J. L., et al. (2003). Continuity of care: a multidisciplinary review. BMJ, 327, 1219–1221. 原文

[17] Agency for Healthcare Research and Quality (2014). Care Coordination Measures Atlas, Chapter 3. 原文

[18] Holden, R. J., et al. (2013). SEIPS 2.0: A human factors framework for studying and improving the work of healthcare professionals and patients. Ergonomics, 56(11), 1669–1686. 原文

[19] Imbens, G. W., & Manski, C. F. (2004). Confidence intervals for partially identified parameters. Econometrica, 72(6), 1845–1857. 原文

[20] Campbell, M. K., et al. (2012). CONSORT 2010 statement: extension to cluster randomised trials. BMJ, 345, e5661. 原文

[21] Hussey, M. A., & Hughes, J. P. (2007). Design and analysis of stepped wedge cluster randomized trials. Contemporary Clinical Trials, 28, 182–191. 原文

[22] Hemming, K., et al. (2018). Reporting of stepped wedge cluster randomised trials: CONSORT extension. BMJ, 363, k1614. 原文

[23] Clopper, C. J., & Pearson, E. S. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika, 26(4), 404–413. 原文

[24] Leveson, N. (2004). A new accident model for engineering safer systems. Safety Science, 42(4), 237–270. 原文

[25] Vasey, B., et al. (2022). Reporting guideline for the early-stage clinical evaluation of decision support systems driven by artificial intelligence: DECIDE-AI. Nature Medicine, 28,

924–933. 原文

[26] Kahan, B. C., et al. (2023). Estimands in cluster-randomized trials: choosing analyses that answer the right question. International Journal of Epidemiology, 52(1), 107–118. 原文

[27] Starmer, A. J., et al. (2014). Changes in medical errors after implementation of a handoff program. New England Journal of Medicine, 371, 1803–1812. 原文

[28] Ball, J. E., Murrells, T., Rafferty, A. M., Morrow, E., & Griffiths, P. (2014). Care left undone during nursing shifts. BMJ Quality & Safety, 23, 116–125. 原文

[29] Wong, A., et al. (2021). External validation of a widely implemented proprietary sepsis prediction model in hospitalized patients. JAMA Internal Medicine, 181, 1065–1070. 原文

[30] Geng, F., Stevenson, D. G., & Grabowski, D. C. (2019). Daily nursing home staffing levels highly variable, often below CMS expectations. Health Affairs, 38, 1095–1100. 原文

[31] Centers for Medicare & Medicaid Services (2026). Payroll Based Journal Daily Nurse Staffing, 2026 Q2. 数据版本及编号。

[32] Centers for Medicare & Medicaid Services. Payroll Based Journal Methodology. 公开文件口径。

[33] Manski, C. F. (1990). Nonparametric bounds on treatment effects. American Economic Review, 80(2), 319–323. 原文

[34] Goddard, K., Roudsari, A., & Wyatt, J. C. (2014). Automation bias: empirical results assessing influencing factors. International Journal of Medical Informatics, 83(5), 368–375. 原文

[35] Medda, F., & Gong, H. (2026). Governed AI-Agent Coordination for Dementia Care: Architecture, Safety Contracts, and Evidence-Derived Workflow Verification.

arXiv:2609.25956v2. 原文

[36] Gathercole, R., et al. (2021). Assistive technology and telecare to maintain independent living at home for people with dementia: the ATTILA RCT. Health Technology Assessment, 25(19), 1–156. 原文

附录 A　执行规范与演练验收

执行系统与检查系统可以承担不同任务。前者整理信息、联系值班人员，后者发现关键记录缺失，提示复核，或在预先授权的范围内暂缓一项高风险的自动指令。受控的 AI Control 研究比较了不同监控和干预安排，后续关于异步审计的研究进一步检查了审计延迟。这些研究能说明为什么要分别记录发现、干预和后果发生的时刻，养老场景的时间窗口仍须从服务过程取得。[7–8]

例如，系统即将取消一次必要的夜间巡视。若核查来得及，应先由值班人员判断。若时间不足，检查系统可以按事先批准的规则暂缓取消，巡视沿原计划继续，同时把原始决定、暂停原因和负责人交给值班团队。暂停到期后如何处理、谁能撤销、怎样通知当事人，应在部署前确定。临时介入的范围仅覆盖当次有争议的操作。

所谓独立检查，需要落实到值班安排和证据来源。即使两套模型由不同公司提供，共用一个失效的账户、通信渠道或日志，告警仍可能同时消失。试验应记录这些共因故障，并核验人工团队能否从备用渠道取得完成照护所需的最少信息。检查系统误报所造成的负担也要计入，包括占用值班时间、延误其他求助，以及对拒绝采集者形成额外限制。

独立性怎样验收

人员、证据和渠道分别验收。人员方面，复核者能够在执行团队没有同意的情况下看到材料、提出复核，涉及自身先前决定时有替补。证据方面，暂停原系统读取或修改权限后，复核者仍能在合法权限内取得核查必需的服务计划、本人陈述或独立照护记录。渠道方面，切换到备用联络方式后，值班人员仍能收到并确认办理。用脱敏演练逐项记录通过与失败，登记共用依赖。三项均通过的配置才计入方案 B、D 的「独立复核」。模型数量和厂商数量作为配置说明。

用脱敏演练模拟执行方不可用、原日志读不到、主联络渠道不可用和三者的共因故障。安排配对评分员按义务编号、演练开始时点和事前确定的场景窗口记录。人员检查通过要求执行方不参与时，另一名有权复核且无责任冲突的人员确认接手并留下签核。证据检查要求隔离原系统后，复核者能从独立授权来源取得该条义务的版本、期限和最少完成证据。渠道检查要求主通道隔离时，备援端确认收到同一条义务及其期限。超时或无法核对均记为未通过，无法获取观测记录标作未知。三项都按窗口完成才记单次演练通过。

部署前演练的停止规则与判断精度

初步安排四类故障各做两次脱敏演练，用来查找可重复的流程缺陷。出现违反必要照护底线、越权或没有可用人工接手时，停止相应演练、整改并核对受影响环节。这八次活动是流程检查，不设「全过即通过可靠性门槛」的结论。即使独立且同风险，八次零失败给出的单侧 95% 精确上界也约为 31.2%；若每类只看两次，该类上界约为 77.6%。[23]

若后续需要用演练衡量某一可恢复流程的失败率，必须先规定四类情景的抽样范围、失败定义及独立分配单位。同一班次的重复测试可能共用人员、证据或渠道，不能直接当成独立复测。在独立同风险的设计例中，假定项目方希望在任何一类失败率达到 10% 时，把全类误通过控制在 5% 以内；若四类均只有 2% 失败，则希望至少 80% 概率全类通过。这些量只用于展示取舍，以后须结合实际误失成本另定。

每类最多测 n 次，累计失败达到 $c+1$ 时提前停止该类，完成且失败不超过 c 则该类达标。设 $F_{n,p}(c)=\Pr\{\mathrm{Bin}(n,p)\le c\}$。如果至少一类失败率不低于 0.10，全类通过的概率至多为 $F_{n,.10}(c)$；因为全类通过必包含该类通过。如果四类失败率均不高于 0.02，由并集界，全类通过概率至少为 $1-4[1-F_{n,.02}(c)]$。这两个总体界均无需假定不同情景之间独立，每类内部仍使用二项抽样假设。以 $n=89, c=4$，上界约为 4.97%，下界约为 86.64%。每类实际执行次数的期望是

$$\mathbb{E}_p T=\sum_{j=0}^{n-1}\Pr\{\mathrm{Bin}(j,p)\le c\}.$$

当 $p=0.02,0.10,0.20$ 时，该例每类平均约需 88、49、25 次，均有至多 89 次的开销。若场地无法提供足够独立单元或失败定义不能稳定复核，保留演练观察和缺陷清单，可靠性仅作为待测量。通过任何演练条件均不能代替现场主结果。

怎样限定第二套系统的介入

评价前由专业负责人和照护对象代表按原照护制度登记必要义务。比如一次计划巡视、约定的人身协助或紧急求助接通都可有各自的完成标准。普通日程提醒和自选资讯提示归入可选项目。必要性来自已确认的计划及专业人员的新决定，第二套系统只读取这份登记和变更记录。涉及用药、急诊或现场安全判断时，交由有职责的专业人员及既有应急流程处置，系统仅能提示和联络。

规则顺序如下。拟变更若未触及登记的必要义务，照常走原有人工与技术流程。若拟取消、推迟到原期限以后或切断必要项目的唯一可用渠道，检查方发出有时间戳的警示，并通知值班人。能够在窗口内人工确认时，由人核定。来不及人工确认时，只有在提前授权、可安全维持原服务且备用渠道已验收的范围内，检查方才可暂缓这一次自动变更。例如拟取消原有人工巡

视，短暂保留原排班。存在现场信息不全、继续原安排也可能有害或没有既定备援时，立即升级给值班专业人员按已有流程处理。

在四格试验中，独立复核组 B、D 可按同一预设协议使用有限暂缓；A、C 沿用原有的人工安全与应急权限。有限暂缓是独立复核方案的组成，备用人手仍单独分配。记录每次能否触发和触发后的服务情况，当前设计只估计复核整套方案的效果。

暂缓记录需有义务编号、依据、触发时刻、最晚截止和接手人。计时以该服务原窗口和许可暂缓时长中较早者为界。值班人与系统判断不同时，由有责任的人员签署决定、告知当事人并留痕。若两名工作人员之间仍有争议，交预先登记的当班主管或独立复核员，原有应急途径始终可用。超时须进入已授权的人工备援流程。延误、误阻断及其对其他服务的影响由独立记录者复核，机构依据既有职责处理。

人工签核和备用服务优先在哪里

Bainbridge 指出，自动化留下的人工工作往往发生在罕见、困难且练习不足的时刻。航空实验也观察到，对自动化建议的依赖会影响独立核查和错误判断。这些材料没有估计养老场景的风险，却支持把实际接手能力和过度信任纳入试验。值班演练记录每小时告警数、复核所用时间、独立判断与默认采纳比例，同时记录通常流程中相同指标。人有权限但因无法取得证据、同时处理多件告警而来不及行动，应算接手过程失败。[13–14]

若人工复核能在原服务窗口内完成，优先采用人工签核并记录其误判、等待和人力成本。若保持原排班或备用班组即可不断供，优先比较这一服务冗余方案。第二套 AI 只有在人工判断来不及且能够实施有限、可撤销的暂缓时，才值得在演练中比较。它可能改善通知速度，也可能增加误报与工作量。应先以同样的人工接手条件，比较「人工直接签核」「常规服务冗余」「增加 AI 提示与允许的有限暂缓」，分别记录发现、介入和照护恢复。只有在真实实施条件下净收益得到证据支持，才有理由扩大这种授权。

预试门槛对结论有什么影响

四周试两处场地，每处先抽连续记录 30 项，独立双评 60 项。分别报告核验覆盖、必要性与结果的原始一致率及 Cohen κ，在小类很少时并报分歧格，期限另报误差范围。在 90%、95%、98% 三档记录核验门槛下，60 项分别需至少 54、57、59 项可核验。一致率门槛 85%、90%、95% 分别需至少 51、54、57 项一致。若有 58 项可核验、54 项一致，基准的 95% 与 90% 门槛通过，较严门槛下两项都不通过。这里的数量是拟议验收灵敏度，无现场观察值。基准门槛用于发现记录问题，主研究另按上式和现场预计未知率拟定识别宽度预算及簇数。

附录 B　合成模拟复算（Python 3、NumPy 和 SciPy）

```
"""Synthetic clustered design stress test. Probabilities are design inputs."""
import json

import numpy as np
from scipy.stats import t

BASE = np.array([.78, .82, .88, .92])
CONTRAST = np.array([-.5, -.5, .5, .5])
SHIFT = .06
RHO = .05
REPEATS = 4000
SIZES = (10, 20, 40, 80)


def simulate(rng, blocks, *, unequal=False, masking=False, lost=False,
             biased=False, spill=False, null=False, rho=RHO, size_profile=None):
```

```
    # A block holds one cluster per cell. Cells have exchangeable case mix
    # except in the intentionally invalid post-allocation selection scenario.
    mix = np.array([.75, .75, .25, .25]) if biased else np.repeat(.5, 4)
    effects = np.array([.04, .04, -.02, -.02]) if spill else np.zeros(4)
    base = BASE.copy()
    if null:
        base[2:] -= .10
    shape = (REPEATS, blocks, 4)
    higher = rng.random(shape) < mix
    nominal = base + effects + SHIFT * (2 * higher.astype(int) - 1)
    mean = base + effects + SHIFT * (2 * mix - 1)
    mass = 1 / rho - 1
    probabilities = rng.beta(nominal * mass, (1 - nominal) * mass)
    if size_profile is not None:
        values, proportions = size_profile
        counts = rng.choice(values, size=shape, p=proportions)
    else:
        counts = rng.choice([5, 35], size=shape) if unequal else np.full(shape, 20)
    successes = rng.binomial(counts, probabilities)
    miss_success = np.array([.02, .02, .025, .025]) if masking else np.zeros(4)
    miss_failure = np.array([.05, .05, .10, .10]) if masking else np.zeros(4)
    lost_probability = np.array([0, 0, .03, .03]) if lost else np.zeros(4)
    masked_success = rng.binomial(successes, miss_success)
    masked_failure = rng.binomial(counts - successes, miss_failure)
    all_lost = rng.random(shape) < lost_probability
    lower = np.where(all_lost, 0, (successes - masked_success) / counts)
    upper = np.where(all_lost, 1, (successes + masked_failure) / counts)
    block_lower = np.where(CONTRAST > 0, lower, upper) @ CONTRAST
    block_upper = np.where(CONTRAST > 0, upper, lower) @ CONTRAST

    critical = t.ppf(.975, blocks - 1)
    radius = critical / np.sqrt(blocks)
    ci_lower = block_lower.mean(axis=1) - radius * block_lower.std(axis=1, ddof=1)
    ci_upper = block_upper.mean(axis=1) + radius * block_upper.std(axis=1, ddof=1)
    lower_expected = (1 - lost_probability) * mean * (1 - miss_success)
    upper_expected = lost_probability + (1 - lost_probability) * (
        mean + (1 - mean) * miss_failure)
    pop_lower = np.where(CONTRAST > 0, lower_expected, upper_expected) @ CONTRAST
    pop_upper = np.where(CONTRAST > 0, upper_expected, lower_expected) @ CONTRAST
    contrast = mean @ CONTRAST
    return dict(blocks=blocks, contrast=float(contrast),
                population_bounds=[float(pop_lower), float(pop_upper)],
                covers_identified_set=float(np.mean((ci_lower <= pop_lower + 1e-12)
                                                    & (ci_upper >= pop_upper - 1e-12))),
                covers_isolated_target=float(np.mean((ci_lower <= (.00 if null else .10))
                                                     & (ci_upper >= (.00 if null else .10)))),
                positive_lower=float(np.mean(ci_lower > 0)))


def main():
    scenarios = {
        'equal, complete': {},
        'unequal, complete': {'unequal': True},
        'unequal, outcome-dependent missingness': {'unequal': True, 'masking': True},
        'plus 3% full-cluster missingness in backup arms': {
            'unequal': True, 'masking': True, 'lost': True},
        'unequal, shared backup and congestion': {'unequal': True, 'spill': True},
        'unequal, post-allocation imbalance': {'unequal': True, 'biased': True},
        'true null, unequal with missingness': {'unequal': True, 'masking': True, 'null': True},
    }
    for index, (name, kwargs) in enumerate(scenarios.items()):
        rng = np.random.default_rng(261124 + index)
        levels = [simulate(rng, blocks, **kwargs) for blocks in SIZES]
```

```
        print(json.dumps({'scenario': name, 'results': levels}))


if __name__ == '__main__':
    main()
```

附录 C　重建表 3 与演练判据

将前一程序存为 `cluster_sensitivity.py`，与本程序同目录。带参数 `--simulation-table` 运行可重建表 3；无参数运行可复算演练判据。

```
"""Regenerate the simulation table and calculate example drill operating rates."""
import argparse
import io
import json
from contextlib import redirect_stdout

from scipy.stats import binom
from cluster_sensitivity import main as run_cases


LABELS = (
    ' 等大簇，记录完整',
    ' 不等簇，记录完整',
    ' 不等簇，事件差异缺失',
    ' 上行再加备用组整簇失访',
    ' 不等簇，共享备用与拥挤',
    ' 不等簇，分配后构成失衡',
    ' 真零效应，事件差异缺失',
)


def table_rows():
    output = io.StringIO()
    with redirect_stdout(output):
        run_cases()
    rows = [json.loads(line) for line in output.getvalue().splitlines()]
    assert len(rows) == len(LABELS)
    for number, (label, row) in enumerate(zip(LABELS, rows), 1):
        sizes = {v['blocks']: v for v in row['results']}
        lower, upper = sizes[20]['population_bounds']
        yield (f'{number} {label}', f'{100*lower:.1f} 至 {100*upper:.1f}',
               *(f"{100*sizes[k]['positive_lower']:.1f}" for k in (20, 40, 80)))


def drill_plan():
    bad, good, alpha = .10, .02, .05
    for trials in range(1, 90):
        for allowed in range(trials + 1):
            false_pass = binom.cdf(allowed, trials, bad)
            pass_when_good = binom.cdf(allowed, trials, good)
            if false_pass <= alpha and 4 * (1 - pass_when_good) <= .20:
                return trials, allowed, false_pass, pass_when_good
    raise RuntimeError('No design within the stated budget')


def expected_runs(trials, allowed, failure_rate):
    # Stop on failure number allowed+1, otherwise run to the fixed end.
    return sum(binom.cdf(allowed, so_far, failure_rate) for so_far in range(trials))
```

```
if __name__ == '__main__':
    parser = argparse.ArgumentParser(description=__doc__)
    parser.add_argument('--simulation-table', action='store_true')
    if parser.parse_args().simulation_table:
        for row in table_rows():
            print(' | '.join(row))
    else:
        n, c, false_pass, good_pass = drill_plan()
        assert (n, c) == (89, 4)
        print('n, c, worst-case false pass, all-good lower bound:',
              n, c, float(false_pass), float(1 - 4 * (1 - good_pass)))
        for probability in (.02, .10, .20):
            print(probability, expected_runs(n, c, probability))
```

附录 D　精确抽查、加权端点与额外模拟

将本程序放在 `cluster_sensitivity.py` 的同一目录。使用 Python 3、NumPy 和 SciPy 运行。附录 B 保留前表所用的默认参数。

```
"""Reproduce cluster sensitivity and exact missing-record audit examples."""
from fractions import Fraction
from itertools import product
from math import ceil, floor

import numpy as np
from scipy.stats import hypergeom

from cluster_sensitivity import simulate


def bounded_sum(coefficients, minimum, maximum):
    """Exact weighted endpoints when the unknown ones have a bounded count."""
    ordered = sorted(coefficients)
    if not 0 <= minimum <= maximum <= len(ordered):
        raise ValueError("Infeasible number of successes")
    negatives = sum(value < 0 for value in ordered)
    positives = sum(value > 0 for value in ordered)
    lower_count = min(max(negatives, minimum), maximum)
    upper_count = min(max(positives, minimum), maximum)
    return sum(ordered[:lower_count]), sum(ordered[len(ordered)-upper_count:])


def support_count_interval(unknown, reference, delta):
    """Treat reference ± delta as a sensitivity assumption on realized rates."""
    lower = max(0, reference - delta)
    upper = min(1, reference + delta)
    return ceil(unknown * lower), floor(unknown * upper)


def audit_interval(unknown, selected, confirmed, unresolved=0, alpha=.05):
    """Invert two hypergeometric tails, allowing unresolved sampled records."""
    sample_low = confirmed
    sample_high = confirmed + unresolved
    if not 0 <= sample_low <= sample_high <= selected <= unknown:
        raise ValueError("Inconsistent audit counts")
    possible = []
    for total in range(sample_low, unknown - selected + sample_high + 1):
        model = hypergeom(unknown, total, selected)
        if model.sf(sample_low - 1) >= alpha/2 and model.cdf(sample_high) >= alpha/2:
            possible.append(total)
    return min(possible), max(possible)
```

```
def audit_checks():
    for length in range(6):
        for vector in product((-2, 0, 1, 3), repeat=length):
            by_count = {count: [] for count in range(length + 1)}
            for outcomes in product((0, 1), repeat=length):
                by_count[sum(outcomes)].append(sum(a*b for a, b in zip(vector, outcomes)))
            for low in range(length + 1):
                for high in range(low, length + 1):
                    values = [value for k in range(low, high + 1) for value in by_count[k]]
                    assert bounded_sum(vector, low, high) == (min(values), max(values))

    # Exact sampling coverage, checked over all states of this illustrative audit.
    unknown, selected = 40, 20
    intervals = [audit_interval(unknown, selected, found)
                 for found in range(selected + 1)]
    full_coverage = []
    for actual in range(unknown + 1):
        probability = hypergeom(unknown, actual, selected).pmf(range(selected + 1))
        full_coverage.append(sum(p for p, (low, high) in zip(probability, intervals)
                                 if low <= actual <= high))
    assert min(full_coverage) >= .95
    lower, upper = audit_interval(unknown, selected, 14)
    with_unknown = audit_interval(unknown, selected, 14, unresolved=2)
    print('exact audit with 14 confirmations:', lower, upper,
          'two unresolved:', with_unknown, 'minimum coverage:', min(full_coverage))
    print('additive sensitivity, 20 missing, observed 3/4:',
          [(str(d), support_count_interval(20, Fraction(3, 4), d))
           for d in (Fraction(0), Fraction(1, 10), Fraction(1, 5))])


def simulation_rows():
    for index, (rho, profile, lost) in enumerate(product(
        (.01, .05, .15),
        ('two-point', 'long-tail'),
        (False, True),
    )):
        sizes = ((2, 8, 80), (.4, .4, .2)) if profile == 'long-tail' else None
        rng = np.random.default_rng(261190 + index)
        rows = [simulate(rng, blocks, unequal=True, masking=True, rho=rho,
                         size_profile=sizes, lost=lost) for blocks in (10, 20, 40, 80)]
        yield rho, profile, lost, rows


if __name__ == '__main__':
    audit_checks()
    for rho, profile, lost, rows in simulation_rows():
        print(rho, profile, lost,
              'width', round(100*(rows[0]['population_bounds'][1] -
                                  rows[0]['population_bounds'][0]), 2),
              'coverage', [round(100*row['covers_identified_set'], 1) for row in rows],
              'positive', [round(100*row['positive_lower'], 1) for row in rows])
```

附录 E　公开排班数据复算

复算使用 CMS 数据版本 `e94f581f-9328-48e4-8ebc-52e3748d2a1c`。原始 CSV 的 SHA-256 为

`c4c232a0d71670072551e8775a82a21ef095fb31e36668d80f2b355d7fb8b1b8`

将该文件路径作为以下 Python 3 程序的唯一命令行参数。

```
"""Descriptive weekday/weekend comparison using the CMS 2026 Q2 PBJ CSV."""
import csv
from collections import defaultdict
from datetime import date, timedelta
from fractions import Fraction
from pathlib import Path
from statistics import median
from sys import argv


def facility_days(path):
    start = date(2026, 4, 1)
    calendar = [start + timedelta(days=i) for i in range(91)]
    weekends = {day.strftime("%Y%m%d"): day.weekday() >= 5 for day in calendar}
    facilities = defaultdict(dict)
    excluded = 0
    with Path(path).open(encoding="latin-1", newline="") as file:
        for row in csv.DictReader(file):
            key, when = row["PROVNUM"], row["WorkDate"]
            days = facilities[key]
            assert row["CY_Qtr"] == "2026Q2" and when in weekends and when not in days
            census = int(row["MDScensus"])
            rn, admin, don = (float(row[field]) for field in
                              ("Hrs_RN", "Hrs_RNadmin", "Hrs_RNDON"))
            assert min(census, rn, admin, don) >= 0
            excluded += census == 0
            days[when] = (census, rn == 0, rn == admin == don == 0)

    return facilities, weekends, excluded


def analyze(path):
    facilities, weekends, excluded = facility_days(path)
    raw_days = sum(map(len, facilities.values()))
    print(f"raw facilities={len(facilities)} days={raw_days} zero census={excluded}")
    comparisons = []
    pooled = [[0, 0, 0], [0, 0, 0]]
    for days in facilities.values():
        counts = [[0, 0, 0], [0, 0, 0]]
        censuses = []
        for when, (census, direct_zero, all_zero) in days.items():
            if census == 0:
                continue
            censuses.append(census)
            group = counts[weekends[when]]
            group[0] += 1
            group[1] += direct_zero
            group[2] += all_zero
        if any(group[0] == 0 for group in counts):
            continue
        for aggregate, group in zip(pooled, counts):
            for i, value in enumerate(group):
                aggregate[i] += value
        rates = [[Fraction(value, group[0]) for value in group[1:]]
                 for group in counts]
        comparisons.append((median(censuses), rates))

    print(f"paired facilities={len(comparisons)} pooled counts={pooled}")
    for label, subset in (
        ("all", comparisons),
        ("median census <= 60", [item for item in comparisons if item[0] <= 60]),
```

```
        ("median census > 60", [item for item in comparisons if item[0] > 60]),
    ):
        for i, name in enumerate(("direct RN", "all RN roles")):
            pairs = [(rates[0][i], rates[1][i]) for _, rates in subset]
            means = [sum(values) * 100 / len(pairs) for values in zip(*pairs)]
            changes = [weekend - weekday for weekday, weekend in pairs]
            print(f"{label}: n={len(pairs)} {name}; weekday={float(means[0]):.4f}% "
                  f"weekend={float(means[1]):.4f}% difference={float(means[1]-means[0]):.4f}pp "
                  f"positive/zero/negative="
                  f"{sum(x>0 for x in changes)}/{sum(x==0 for x in changes)}/"
                  f"{sum(x<0 for x in changes)}")


if __name__ == "__main__":
    analyze(argv[1])
```

配对重采样程序与上一个程序放在同一目录。安装 NumPy，运行以下程序并传入同一 CSV。

```
"""Paired facility bootstrap for descriptive staffing contrasts, with size cuts."""

from statistics import median
from sys import argv

import numpy as np

from analyze_pbj_2026q2 import facility_days


def facility_pairs(path):
    facilities, weekends, _ = facility_days(path)
    for days in facilities.values():
        census = []
        counts = np.zeros((2, 3), dtype=np.int32)
        for when, (residents, direct_zero, all_zero) in days.items():
            if residents == 0:
                continue
            census.append(residents)
            counts[int(weekends[when])] += (1, direct_zero, all_zero)
        if (counts[:, 0] == 0).any():
            continue
        weekday, weekend = counts[:, 1:] / counts[:, :1]
        direct_difference, all_difference = weekend - weekday
        yield median(census), direct_difference, all_difference, direct_difference - all_difference


def paired_bootstrap(contrasts, rng, replicates=6000):
    # Each draw keeps both days-of-week rates and both RN measures from a facility together.
    replicates_mean = []
    n = len(contrasts)
    for start in range(0, replicates, 64):
        indices = rng.integers(0, n, size=(min(64, replicates - start), n))
        replicates_mean.append(contrasts[indices].mean(axis=1))
    means = np.concatenate(replicates_mean)
    limits = np.quantile(means, (0.025, 0.975), axis=0)
    return zip(100 * contrasts.mean(axis=0), 100 * limits[0], 100 * limits[1])


def analyze(path):
    data = np.array(list(facility_pairs(path)))
    rng = np.random.default_rng(26022026)
    strata = [("all", data)]
    for maximum in (40, 60, 80, 100):
        strata.append((f"<= {maximum}", data[data[:, 0] <= maximum]))
        strata.append((f"> {maximum}", data[data[:, 0] > maximum]))
    print("facility stratum; n; direct RN, all RN, administrative-only")
    for label, stratum in strata:
        assert len(stratum) > 0
        estimates = paired_bootstrap(stratum[:, 1:], rng)
        cells = [f"{mean:+.2f} [{lo:+.2f}, {hi:+.2f}]" for mean, lo, hi in estimates]
        print(label, len(stratum), *cells, sep="; ")
```

```
if __name__ == "__main__":
    analyze(argv[1])
```

附录 F　分层核验名额的独立复算

使用 Python 3 标准库，检验已知可核验记录及不可核验残留两种口径下的可行性和最少条数。取整后的整数刻度用于避免舍入误差。

```
"""Verify the minimum successful audit count under local and overall width caps."""

from itertools import combinations, product


def choose_audits(available, residual, weights, stratum_caps, total_cap):
    """Widths and scale weights are exact nonnegative integers."""
    ranks = [sorted(group, reverse=True) for group in available]
    widths = [fixed + sum(group) for fixed, group in zip(residual, ranks)]
    counts = [0] * len(ranks)
    for d, cap in enumerate(stratum_caps):
        if residual[d] > cap:
            return None
        while widths[d] > cap:
            widths[d] -= ranks[d][counts[d]]
            counts[d] += 1

    overall = sum(w * width for w, width in zip(weights, widths))
    remaining = sorted(
        [(weights[d] * value, d)
         for d, group in enumerate(ranks) for value in group[counts[d]:]],
        reverse=True,
    )
    for gain, d in remaining:
        if overall <= total_cap:
            break
        overall -= gain
        counts[d] += 1
    if overall > total_cap:
        return None
    return tuple(counts)


def verify():
    weights = (2, 3, 1)
    checked, impossible = 0, 0
    for values in product(range(1, 4), repeat=6):
        groups = (values[:2], values[2:4], values[4:])
        for residual in ((0, 0, 0), (1, 0, 2)):
            entries = tuple((d, value) for d, group in enumerate(groups)
                            for value in group)
            initial = tuple(fixed + sum(group) for fixed, group in zip(residual, groups))
            enumeration = []
            for count in range(7):
                for selected in combinations(entries, count):
                    widths = list(initial)
                    for d, value in selected:
                        widths[d] -= value
                    total_width = sum(w * x for w, x in zip(weights, widths))
                    enumeration.append((count, widths, total_width))
```

```
            for caps in product(*(sorted({0, x // 2, x}) for x in initial)):
                feasible = [(n, total) for n, widths, total in enumeration
                            if all(width <= cap for width, cap in zip(widths, caps))]
                for total_cap in (0, 4, 10, 30):
                    expected = min((n for n, width in feasible if width <= total_cap),
                                   default=None)
                    result = choose_audits(groups, residual, weights, caps, total_cap)
                    assert (None if result is None else sum(result)) == expected
                    checked += 1
                    impossible += result is None

    print(f"{checked:,} policies checked; {impossible:,} infeasible")
    groups = ((3,) * 8, (6,) * 5, (5,) * 4, (12,) * 3)
    weights, residual = (45, 20, 25, 10), (0, 0, 0, 0)
    assert choose_audits(groups, residual, weights, (24, 30, 20, 36), 1900) == (5, 0, 0, 0)
    assert choose_audits(groups, residual, weights, (24, 18, 20, 12), 1900) == (2, 2, 0, 2)


if __name__ == "__main__":
    verify()
```

异成本方法的独立复算使用 Python 3 标准库。

```
"""Minimum-cost audits with local and overall caps on remaining width."""

from itertools import product
from random import Random


def frontier(costs):
    """Keep gain-cost choices that cannot be improved on both coordinates."""
    remaining = {}
    best = float("inf")
    for gain, cost in sorted(costs.items(), reverse=True):
        if cost < best:
            remaining[gain] = cost
            best = cost
    return remaining


def minimum_cost(groups, residual, weights, local_caps, overall_cap):
    """Use integer gains, weights, widths, and nonnegative costs.

    An item (gain, cost) is one resolution with a known, additive reduction.
    Residual widths belong to records with no available resolution option.
    """
    initial = [fixed + sum(gain for gain, _ in group)
               for fixed, group in zip(residual, groups)]
    if (any(fixed > cap for fixed, cap in zip(residual, local_caps))
            or sum(w * width for w, width in zip(weights, residual)) > overall_cap):
        return None
    target = max(0, sum(w * width for w, width in zip(weights, initial)) - overall_cap)
    combined = {0: 0}

    for group, width, weight, cap in zip(groups, initial, weights, local_caps):
        local_target = max(0, width - cap)
        gain_limit = max(local_target, (target + weight - 1) // weight) if weight else local_target
        options = {0: 0}
        for gain, cost in group:
            for old_gain, old_cost in tuple(options.items()):
                new_gain = min(gain_limit, old_gain + gain)
                new_cost = old_cost + cost
                options[new_gain] = min(options.get(new_gain, new_cost), new_cost)
            options = frontier(options)

        next_combined = {}
        for gained, cost in combined.items():
            for local_gain, local_cost in options.items():
                if local_gain < local_target:
```

```
                    continue
                gain = min(target, gained + weight * local_gain)
                total_cost = cost + local_cost
                next_combined[gain] = min(next_combined.get(gain, total_cost), total_cost)
        combined = frontier(next_combined)
    return combined.get(target)


def exhaustive(groups, residual, weights, local_caps, overall_cap):
    items = [(d, gain, cost) for d, group in enumerate(groups) for gain, cost in group]
    initial = [fixed + sum(gain for gain, _ in group)
               for fixed, group in zip(residual, groups)]
    best = None
    for mask in product((False, True), repeat=len(items)):
        widths = initial.copy()
        total_cost = 0
        for included, (d, gain, cost) in zip(mask, items):
            if included:
                widths[d] -= gain
                total_cost += cost
        if (all(width <= cap for width, cap in zip(widths, local_caps))
                and sum(w * width for w, width in zip(weights, widths)) <= overall_cap):
            best = total_cost if best is None else min(best, total_cost)
    return best


def verify():
    rng = Random(260221)
    infeasible = 0
    for _ in range(12_000):
        groups = tuple(tuple((rng.randrange(5), rng.randrange(7))
                             for _ in range(rng.randrange(4))) for _ in range(3))
        residual = tuple(rng.randrange(4) for _ in groups)
        weights = tuple(rng.randrange(5) for _ in groups)
        local_caps = tuple(rng.randrange(12) for _ in groups)
        overall_cap = rng.randrange(24)
        args = groups, residual, weights, local_caps, overall_cap
        result = minimum_cost(*args)
        assert result == exhaustive(*args), args
        infeasible += result is None
    assert minimum_cost((((7, 6), (5, 3), (5, 3)),), (0,), (1,), (7,), 7) == 6
    print(f"12,000 exact checks; {infeasible:,} infeasible")


if __name__ == "__main__":
    verify()
```