《当模型获得行动权》
When Models Acquire the Right to Act

副标题：授权、停机与责任如何在现代系统中消失
Subtitle: How Authorization, Halt, and Responsibility Disappear in Modern Systems

导论丨这不是批判，这是权限审计
Introduction | This Is Not Critique, This Is an Audit of Authority

我们不是在质疑模型是否“正确”。
We are not questioning whether models are “correct.”
我们审计的是另一件事：
We are auditing something else instead:
模型是如何获得行动权的。
How models come to acquire the right to act.
在现代系统中，
In modern systems,
模型的输出越来越少被当作“建议”，
model outputs are increasingly treated not as “advice,”
而是被直接嵌入行动链条，
but as elements directly embedded into chains of action,
触发交易、批准工程、启动干预、执行处罚。
triggering trades, approving engineering decisions, initiating interventions, enforcing sanctions.
这一转变通常没有被正式命名，
This transition is rarely explicitly named,
也几乎从未被审计。
and almost never audited.

0.1 我们不是在讨论真理
0.1 We Are Not Discussing Truth

真理属于理论系统内部。
Truth belongs to the interior of theoretical systems.
它由公理、推导与一致性来约束。
It is constrained by axioms, derivations, and internal consistency.
但行动权属于现实系统。
But the right to act belongs to real-world systems.

它意味着：
It means:

某个输出，被允许改变世界状态。
Some output is permitted to change the state of the world.
这两者不是同一个问题。
These are not the same question.
一个命题可以在其形式系统中为真，
A proposition can be true within its formal system,
却完全不具备任何行动资格。
yet possess no legitimate claim to action.
本书关注的不是“是否为真”，
This book is not concerned with “whether something is true,”
而是：
but rather:

> **谁、在何种条件下，允许它继续运行并产生后果。**
> **Who, under what conditions, allows it to keep running and produce consequences.**

0.2 为什么“谁有权继续运行”比“对不对”更重要
0.2 Why “Who Has the Right to Continue” Matters More Than “Right or Wrong”

错误本身并不会造成灾难。
Error itself does not cause catastrophe.
未经授权的继续运行才会。
Unauthorized continuation does.
一个错误的模型，如果被及时否决，
A flawed model, if halted in time,
最多只是一次失败的尝试。
is at most a failed attempt.
一个“尚未被否决”的模型，
A model that has never been denied,
即使逻辑完美、精度极高，
even if logically elegant and highly accurate,
一旦被默认允许持续运行，
once it is implicitly allowed to keep running,
就会把所有不确定性外包给未来。
it externalizes all uncertainty into the future.
而未来，
And the future,
通常并不在任何人的责任范围内。
is usually outside anyone’s responsibility domain.

0.3 本书不审计定理，只审计模型获得行动权的过程
0.3 This Book Does Not Audit Theorems, Only the Process by Which Models Gain the Right to Act

我们不会评判数学是否“走错了路”。
We will not judge whether mathematics has “gone astray.”
我们不会讨论哪种理论更优雅。
We will not debate which theory is more elegant.
我们只问一个问题：
We ask only one question:

> **在离开原始假设语境之后，**
> **after leaving their original assumption contexts,**
> **模型是如何被允许进入决策与执行系统的？**
> **how are models allowed to enter decision and execution systems?**

这是一个接口问题，
This is an interface problem,
不是一个真理问题。
not a truth problem.

0.4 阅读须知：哪些读者会在这里感到不适（以及为什么）
0.4 Reading Notice: Who Will Feel Uncomfortable Here (and Why)

如果你希望看到的是：
If you are expecting:

“模型错了，所以灾难发生了”，
“the model was wrong, therefore disaster occurred,”
你可能会失望。

you may be disappointed.
如果你希望看到的是：
If you are hoping for:

> “只要提高精度，一切都会解决”，
> “higher accuracy will fix everything,”

你可能会感到被冒犯。
you may feel offended.
因为本书反复指出的事实是：
Because what this book repeatedly shows is:

> **许多灾难发生在模型仍然‘看起来正确’的时候。**
> **many disasters occur precisely when models still appear correct.**

问题不在于计算，
The problem is not computation,
而在于授权。
but authorization.

Catalog

第一编丨默认继续：一个无人否决的世界

Part I | Default Continuation: A World Without Veto

这一编讨论的不是某一次具体失败。
This part is not about any particular failure.
它讨论的是一种**默认设置**。
It examines a **default setting**.
在大量现代系统中，
Across many modern systems,
“继续运行”并不是一个被明确批准的结论，
“continue running” is not an explicitly approved conclusion,
而是一个从未被否决的状态。
but a state that was never denied.
当没有人被指定为“可以说不”的角色时，
When no one is designated as having the authority to say “no,”
继续就会成为唯一剩下的选项。
continuation becomes the only remaining option.

第 1 章丨“继续”从来不是一个被证明的结论

Chapter 1 | “Continue” Is Never a Proven Conclusion

系统继续运行，
Systems keep running,
并不是因为它们被证明是安全的，
not because they have been proven safe,
而是因为**没有一条被执行的否决路径。**

but because **no executable veto path exists.**
这是一个极其关键、
This is a critical,
却长期被忽略的区别。
yet long-ignored distinction.

1.1 条件真理如何被误读为操作许可
1.1 How Conditional Truth Is Misread as Operational Permission

大多数模型输出的形式都是条件性的。
Most model outputs are conditional in form.
在假设 A、B、C 成立的情况下，
Under assumptions A, B, and C,
结论 D 成立。
conclusion D follows.
这是一个**条件真理**。
This is a **conditional truth.**
但在进入现实系统时，
But once it enters a real-world system,
这条条件往往被压缩成一句话：
this condition is often compressed into a single sentence:
"模型显示可以继续。"
"The model shows we can continue."
假设被隐去，
The assumptions disappear,
语境被剥离，
the context is stripped away,
剩下的只有一个看似完整的结论。
leaving behind what appears to be a complete conclusion.
从那一刻起，
From that moment on,
条件真理被误读为操作许可。
conditional truth is misread as operational permission.

1.2 数学的沉默：前提失效时，系统听见了什么？
1.2 Mathematical Silence: What Does the System Hear When Assumptions Fail?

当模型的前提开始偏离现实时，
When a model's assumptions begin to drift away from reality,
数学通常不会"报警"。
mathematics usually does not "raise an alarm."
它只是不再保证结论的适用性。
It simply no longer guarantees applicability.
这是数学的边界，
This is mathematics respecting its boundary,
而不是它的失败。
not a failure.
但现实系统无法听见这种沉默。
But real systems cannot hear silence.
系统只接收两类信号：
Systems receive only two kinds of signals:
可以继续，
continue,
或者必须停止。
or must stop.
当数学不说"停"，
When mathematics does not say "stop,"

系统听到的并不是“不确定”，
the system does not hear “uncertain,”
而是——
but rather—
没有人反对。
no one objected.

1.3 为什么“没有人说不”会变成“可以继续”
1.3 Why “No One Said No” Becomes “You May Proceed”

在工程、金融、算法治理中，
In engineering, finance, and algorithmic governance,
权限从来不是通过真理分配的。
authority is never allocated by truth.
它通过角色、流程和默认值分配。
It is allocated through roles, procedures, and defaults.
如果一个系统中：
If, within a system:

> 没有人被明确赋予否决权，
> no one is explicitly granted veto power,
>
> 否决需要额外成本，
> veto requires extra cost,
>
> 继续运行是零成本选项，
> continuation is the zero-cost option,

那么结论是确定的。
then the outcome is deterministic.

继续运行将成为事实上的许可。
Continuation becomes de facto permission.

这不是道德问题，
This is not a moral issue,
也不是认知偏差。
nor a cognitive bias.
这是一个制度结构问题。
It is a structural problem of the system.

第 2 章 | 责任是如何在接口处消失的
Chapter 2 | How Responsibility Disappears at the Interface

责任并不是被“推掉”的。
Responsibility is not something that is merely “shifted away.”
在多数现代系统中，
In most modern systems,
它是在接口处被**结构性抹除**的。
it is **structurally erased** at the interface.

当模型的输出从一个系统
When a model’s output moves from one system
进入另一个系统时，
into another,
责任并没有随之迁移。
responsibility does not migrate with it.

它通常停留在原地，
It usually remains where it was,
而行动已经发生在别处。
while action happens elsewhere.

2.1 从“责任推诿”到“设计性消隐”
2.1 From Responsibility-Shifting to Responsibility-by-Design Disappearance

传统意义上的责任推诿，
Traditional responsibility-shifting
需要一个清晰的对话场景：
requires a clear dialogical scene:
“不是我，是他。”
“It wasn’t me, it was him.”
但在模型驱动的系统中，
But in model-driven systems,
这种场景越来越少出现。
this scene appears less and less.
取而代之的是一种更安静的机制：
What replaces it is a quieter mechanism:

没有人站在那个必须负责的位置上。
no one stands at the position where responsibility must attach.

模型团队说：
The model team says:
“我们只负责准确性。”
“We are only responsible for accuracy.”
系统集成方说：
The system integrator says:
“我们只是按规范接入。”
“We just integrated according to specifications.”
执行者说：
The operator says:
“我只是按系统提示操作。”
“I merely followed the system prompt.”
每一句话都是真的。
Each statement is true.
但真理在这里并不结算责任。
But truth does not settle responsibility here.

2.2 条件真理 → 操作许可：那个从未被命名的转换
2.2 Conditional Truth → Operational Permission: The Unnamed Transformation

在模型与行动之间，
Between models and action,
存在一个关键转换步骤。
there exists a critical transformation step.
它几乎从未被正式定义，
It is almost never formally defined,
却每天都在发生。
yet happens every day.
那就是：
That is:

> **从“在条件成立下为真”，**
> **到“可以据此行动”。**

from "true under conditions"
to "permitted to act upon."

这不是一个数学转换。
This is not a mathematical transformation.
它是一个**权限转换**。
It is a **permission transformation**.

问题在于：
The problem is:

> 这个转换通常是隐式完成的。
> this transformation is usually performed implicitly.
> 没有签字，
> No signature,
> 没有责任主体，
> no responsible party,
> 没有失败路径。
> no failure path.

2.3 为什么每个人都合规，系统却必然无责
2.3 Why Everyone Is Compliant, Yet the System Is Necessarily Irresponsible

在一个高度合规的系统中，
In a highly compliant system,
每个角色都严格遵循自己的职责边界。
each role strictly adheres to its assigned boundary.
正是这种合规，
It is precisely this compliance
制造了整体上的无责状态。
that produces systemic irresponsibility.
因为没有任何一个角色
Because no single role
被授权对"是否继续"作出最终判断。
is authorized to make the final judgment on "whether to continue."
当失败发生时，
When failure occurs,
系统可以完整复述整个合规链条，
the system can perfectly recite the entire compliance chain,
却无法指出一个
but cannot point to a single
有权叫停的人。
person who had the authority to stop it.
这不是漏洞，
This is not a bug.
这是设计结果。
It is the result of design.

第二编丨模型如何获得行动权
Part II | How Models Acquire the Right to Act

这一编不讨论模型"被如何使用"。
This part does not discuss how models are "used."
因为"使用"这个词，
Because the word "use"
掩盖了一个更关键的事实：

conceals a more critical fact:
模型之所以能改变现实，
models can alter reality

不是因为它们被参考了，
not because they were consulted,
而是因为它们被授权进入行动链。
but because they were authorized to enter chains of action.

第 3 章｜模型不是建议，是准行动
Chapter 3 | Models Are Not Advice, They Are Pre-Action

在理想化叙述中，
In idealized narratives,
模型“提供建议”，
models “offer advice,”
人类“做出决定”。
humans “make decisions.”

但在真实系统里，
But in real systems,
这个分工正在系统性瓦解。
this division is systematically collapsing.

模型的输出越来越像：
Model outputs increasingly resemble:
尚未执行的行动，
actions not yet executed,
而不是等待判断的意见。
rather than opinions awaiting judgment.

3.1 输出、建议、行动之间被刻意模糊的边界
3.1 The Deliberately Blurred Boundary Between Output, Advice, and Action

在技术文档中，
In technical documentation,
人们仍然坚持使用温和的词汇：
people still insist on gentle language:
“建议值”、
“recommended values,”
“风险评分”、
“risk scores,”
“辅助决策”。
“decision support.”

但在系统接口层，
But at the interface layer of systems,

这些输出往往直接：
these outputs often directly:

排序任务优先级，
reorder task priorities,

触发阈值机制，
trigger threshold mechanisms,

自动生成下一步指令。
auto-generate the next instruction.

语言被保留为“建议”，
The language remains “advice,”
行为却已经进入行动通道。
while behavior has already entered the action channel.
这不是误用，
This is not misuse.

这是一种**制度性模糊**。
It is **institutionalized ambiguity.**

3.2 为什么“算出来了”会被理解为“可以做了”
3.2 Why “Computed” Is Interpreted as “Permissible”

“算出来了”，
“It has been computed,”
本应只是一个认知状态的更新。
should merely update a cognitive state.
但在组织与系统中，
But within organizations and systems,
它常常被自动翻译为：
it is often automatically translated as:
“现在可以继续。”
“we may now proceed.”
这种翻译并非源于逻辑，
This translation does not arise from logic,
而是源于流程设计。
but from process design.

当一个系统：
When a system:

把计算完成作为流程节点，
treats computation completion as a process milestone,

把“未计算”标记为阻塞状态，
marks “not computed” as a blocking state,

却从不把“不可用”作为有效输出，
but never treats “not applicable” as a valid output,

那么计算结果
then the computation result
就天然成为行动许可。
naturally becomes permission to act.

3.3 模型何时从工具变成制度性接口
3.3 When Models Become Institutional Interfaces Rather Than Tools

工具的特征是：
A tool is characterized by:
它可以被放下。
it can be put down.
接口的特征是：
An interface is characterized by:

它定义了下一步能否发生。
it defines whether the next step can occur.

当一个模型：
When a model:

被嵌入审批流程，
is embedded into approval workflows,

被写入合规要求，
is written into compliance requirements,

被当作“必须引用”的依据，
is treated as a mandatory reference,

它就不再是工具。
it is no longer a tool.
它已经成为制度的一部分。
It has become part of the institution.
从那一刻起，
From that moment on,
它的输出不再是信息，
its outputs are no longer information,
而是**行动的门槛条件**。
but **threshold conditions for action**.

第 4 章丨授权不是一次行为，而是一条链
Chapter 4 | Authorization Is Not a Single Act, but a Chain

当系统出现问题时，
When problems surface in a system,
人们总是试图寻找
people instinctively try to locate
“是谁授权的”。
“who authorized it.”
这个提问本身就已经失真了。
That question is already distorted.
因为在现代系统中，
Because in modern systems,
授权几乎从来不是
authorization is almost never
一个单点事件。
a single-point event.
它是一条链。
It is a chain.

4.1 谁允许模型进入决策链
4.1 Who Allowed the Model to Enter the Decision Chain

模型最初进入系统时，
When a model first enters a system,
往往以“辅助工具”的身份出现。
it usually appears under the label of a “supporting tool.”
这一步几乎总是被视为低风险。
This step is almost always treated as low-risk.
“只是多一个参考。”

"It's just an extra reference."
"只是提高效率。"
"It's just improving efficiency."
但正是在这一刻，
But it is precisely at this moment
第一段授权已经发生了。
that the first segment of authorization occurs.
因为一旦模型被接入，
Because once a model is integrated,
系统结构就开始围绕它重排。
the system structure begins to reconfigure around it.
流程开始依赖它的输出，
Processes begin to depend on its outputs,
人员开始调整工作节奏，
people adjust their work rhythms,
没有模型反而变成异常状态。
and the absence of the model becomes the abnormal case.

4.2 谁默认它可以被信任
4.2 Who Assumed It Could Be Trusted

信任并不总是通过明确声明建立的。
Trust is not always established through explicit declaration.
在多数系统中，
In most systems,
信任是通过**默认继承**产生的。
trust arises through **default inheritance**.
模型继承了：
The model inherits:

> 开发团队的专业声誉，
> the professional reputation of its developers,
>
> 数学形式的权威外观，
> the authoritative appearance of mathematical form,
>
> 过往运行的暂时稳定。
> the temporary stability of past operation.

没有人明确说：
No one explicitly says:
"这个模型在这里是可以被信任的。"
"this model is trustworthy here."
但所有流程都假定了这一点。
Yet all processes assume it.
信任在这里不是被论证的，
Trust here is not argued for,
而是被继承的。
it is inherited.

4.3 谁在失败时"只是引用了模型"
4.3 Who, at Failure, "Merely Referenced the Model"

当系统运行顺利时，
When systems run smoothly,
模型很少被提及。
models are rarely mentioned.

当失败发生时，
When failure occurs,

模型却突然成为
the model suddenly becomes
一种中性对象。
a neutral object.
“我们只是参考了模型。”
“We merely referenced the model.”
这句话在结构上
Structurally, this statement
完成了授权链的最后一环。
completes the final link in the authorization chain.

因为它把：
Because it separates:

> 行动的执行者，
> the executor of action,
>
> 授权的形成过程，
> the formation of authorization,
>
> 与失败的结果
> from the resulting failure

彻底切断。
entirely.
到这一刻为止，
By this point,
授权链是完整的，
the authorization chain is complete,
但责任链已经断裂。
but the responsibility chain has snapped.

第三编丨停机：一个被系统性回避的问题
Part III | Halt: A Problem Systematically Avoided

这一编讨论的不是
This part does not discuss
“为什么系统不停下来”，
“why systems do not stop,”
而是一个更根本的问题：
but a more fundamental one:
谁有权让系统停下来。
who has the authority to stop them.
在大量模型驱动的系统中，
In many model-driven systems,
“停机”并不是一个合法输出。
“halt” is not a legitimate output.
它既不在模型的设计目标里，
It is neither in the model’s design objectives,
也不在流程的激励结构中。
nor in the incentive structure of the process.

第 5 章丨为什么模型永远不会说“必须停机”

Chapter 5 | Why Models Never Say “You Must Halt”

模型并不是被设计来否决自身的。
Models are not designed to veto themselves.
这一点并不丑陋，
This is not ugly,
也并不异常。
nor is it abnormal.
这是它们的职责边界。
It is their boundary of responsibility.

5.1 数学不能停机，也不该停机
5.1 Mathematics Cannot Halt, and Should Not Halt

数学系统的职责是：
The responsibility of mathematics is:
在给定假设下，
under given assumptions,
推导可成立的结论。
to derive conclusions that follow.
当假设失效时，
When assumptions fail,
数学并不会输出“错误”。
mathematics does not output “error.”
它只是不再提供保证。
It simply no longer provides guarantees.
这种沉默
This silence
并不是拒绝责任，
is not a refusal of responsibility,
而是严格履行边界。
but a strict adherence to boundary.
要求数学“告诉我们停下来”，
Demanding mathematics to “tell us to stop”
本身就是一种越界。
is itself a category error.

5.2 一旦模型说“停”，它就越界了
5.2 The Moment a Model Says “Stop,” It Oversteps

如果一个模型被要求输出：
If a model is required to output:
“现在必须停止行动”，
“you must now halt action,”
那么它已经被赋予了
it has already been granted
超出计算范畴的权力。
authority beyond computation.
这意味着模型
This means the model
不仅描述状态，
no longer merely describes states,
而是在做规范性裁决。
but makes normative judgments.
这不是模型的职责，
This is not the model’s responsibility,

而是制度的职责。
but the institution’s.

5.3 停机为什么总是被推给“人的判断”
5.3 Why Halt Is Always Deferred to “Human Judgment”

系统经常宣称：
Systems often claim:
“最终决定仍然在人。”
“the final decision remains with humans.”
这句话听起来很安全，
This sounds reassuring,
但在结构上却极其危险。
but structurally it is extremely dangerous.
因为“人的判断”
Because “human judgment”
通常没有：
usually has no:

- 明确的授权位置，
  clearly authorized position,
- 受保护的否决权，
  protected veto power,
- 可执行的停机通道。
  executable halt channel.

它只是一个
It is merely
责任回收站。
a responsibility sink.

第 6 章丨没有停机权的系统，注定只能继续
Chapter 6 | Systems Without Halt Authority Are Destined to Continue

一个系统是否会停下来，
Whether a system will ever stop
从来不取决于
has never depended on
它是否足够聪明，
how intelligent it is,
而取决于
but on

停机是否被设计为一种合法结果。
whether halt is designed as a legitimate outcome.

如果停机不是合法结果，
If halt is not a legitimate outcome,
那么系统的所有路径
then all system paths
最终都会通向
will eventually converge on
继续。
continuation.

6.1 停机权缺失 vs 停机条件缺失

6.1 Absence of Halt Authority vs Absence of Halt Conditions

很多系统在失败后
Many systems, after failure,
会进行一类错误诊断：
perform a mistaken diagnosis:
“我们缺少明确的停机条件。”
“we lack clear halt conditions.”

于是他们开始：
So they begin to:

> 增加阈值，
> add thresholds,
>
> 引入更多监控指标，
> introduce more monitoring metrics,
>
> 强化异常检测。
> strengthen anomaly detection.

但问题往往不在这里。
But the problem is often not here.
真正缺失的不是条件，
What is truly missing is not conditions,
而是**停机权**。
but **halt authority.**

当即使满足所有停机条件，
When even if all halt conditions are met,
也没有任何角色
there is still no role
被授权说“现在停”，
authorized to say “stop now,”
那么条件只是一种装饰。
then conditions are merely decorative.

6.2 “继续运行”作为默认值的危险性
6.2 The Danger of “Continue Running” as a Default

默认值并不是中性的。
Defaults are not neutral.
在复杂系统中，
In complex systems,
默认值就是
the default
权力的分配方式。
is a method of power allocation.
当“继续运行”被设为默认，
When “continue running” is set as the default,
停机就需要额外解释，
halt requires extra justification,
额外流程，
extra procedure,
额外勇气。
extra courage.
而继续，

Continuation,

什么都不需要。
requires nothing.
在这样的结构下，
Under such a structure,
理性行为
rational behavior
就是不停止。
is not to stop.

6.3 失效被延期，风险被积累
6.3 Failure Is Deferred, Risk Is Accumulated

当系统无法合法停机时，
When a system cannot halt legitimately,
它并不会立即崩溃。
it does not collapse immediately.
相反，
Instead,
它会表现出一种
it exhibits a kind of
延迟失败。
delayed failure.
每一次"再跑一次"，
Each "one more run,"
都把当前风险
pushes current risk
转移到未来。
into the future.
而未来，
And the future,
往往没有对应的
often has no corresponding
责任账户。
responsibility account.
系统看起来在运行，
The system appears to be operating,
甚至在优化，
even optimizing,
但它其实是在
but in fact it is
不断累积
continually accumulating
一次性崩溃所需的能量。
the energy required for a single catastrophic failure.

第四编 | 等号、连续与"未结算结果"的合法化
Part IV | Equals, Continuity, and the Legalization of Unsettled Results

这一编讨论的不是数学技巧，
This part is not about mathematical technique,
而是一个更隐蔽的问题：
but a more subtle one:

> **计算是如何被误认为结算的。**
> **how computation comes to be mistaken for settlement.**

当一个结果被写成一个确定的形式，
When a result is written in a definite form,
系统往往会假设：
systems tend to assume:
这件事已经“算清楚了”。
this matter has been “settled.”
但很多时候，
But in many cases,
它只是被写得
it has merely been written
看起来像结算。
to look like a settlement.

第 7 章｜计算结果≠结算权
Chapter 7 ｜ Computation Results ≠ Right of Settlement

计算给出数值。
Computation produces numbers.
结算意味着：
Settlement means:
这个结果
this result
被允许作为
is permitted to serve as
行动的终点。
the endpoint of action.
这两者之间，
Between the two,
隔着一个
there lies a
权限断层。
permission gap.

7.1 等号何时从结论变成授权
7.1 When the Equals Sign Shifts from Conclusion to Authorization

在数学内部，
Within mathematics,
等号表示一种关系成立。
the equals sign denotes a relation that holds.
它不包含命令，
It contains no command,
也不包含许可。
and no permission.
但当等号离开形式系统，
But when the equals sign leaves the formal system,
进入工程、金融或治理文本，
and enters engineering, finance, or governance texts,
它的语义开始滑移。
its semantics begin to drift.

“X = Y”

不再只是陈述，
is no longer merely a statement,
而被理解为：
but is interpreted as:

“X 可以被当作 Y 使用。”
“X may be treated as Y.”

这一刻，
At this moment,
等号完成了一次
the equals sign undergoes
未被授权的升级。
an unauthorized upgrade.

7.2 多解、近似、收敛：谁说“这就够了”
7.2 Multiple Solutions, Approximation, Convergence: Who Decides “This Is Enough”

在真实问题中，
In real problems,
解往往不是唯一的。
solutions are often not unique.
数值是近似的，
Numerical values are approximate,
收敛是渐进的。
convergence is asymptotic.
这些事实
These facts
在数学语境中
in mathematical contexts
是清楚且被接受的。
are clear and accepted.
但在行动系统中，
But in action systems,
它们必须被压缩成一个判断：
they must be compressed into a single judgment:
“够不够？”
“Is it enough?”
问题在于：
The problem is:
这个判断
this judgment
通常不是由
is usually not made by
被授权的人作出的。
an authorized role.
它是被
It is
流程默认完成的。
completed by default through process.

7.3 数值结果是如何被升级为决策终点的
7.3 How Numerical Results Are Elevated to Decision Endpoints

当一个系统：
When a system:

把数值写入报告末尾，
places numbers at the end of reports,

把图表作为最终页面，

treats charts as final pages,

把“计算完成”作为流程终止节点，
treats “computation complete” as the terminal process node,

那么数值
then numbers
自然会被理解为
will naturally be understood as
结论。
conclusions.
而结论
And conclusions
在组织中
within organizations
往往等价于
often equate to
行动的触发条件。
action triggers.
没有人明确说：
No one explicitly says:
“这个数值已经获得了结算权。”
“this number has been granted settlement authority.”
但所有人
But everyone
都按这个假设行动。
acts as if it has.

第 8 章丨连续假设如何抹平断裂
Chapter 8 | How Continuity Assumptions Smooth Over Rupture

连续性在数学中是一种技术性假设。
Continuity in mathematics is a technical assumption.
它的作用是让变化可被分析，
Its role is to make change analyzable,
让极限、导数、积分成为可能。
to enable limits, derivatives, and integrals.
在形式系统内部，
Within formal systems,
连续性从来不是对现实的承诺。
continuity is never a promise about reality.
它只是一个工作前提，
It is merely a working premise,
用于让推导能够继续。
used to allow derivations to proceed.

8.1 连续不是错，错的是把它当作现实保证
8.1 Continuity Is Not the Problem; Treating It as a Reality Guarantee Is

连续假设本身并不制造风险。
Continuity assumptions themselves do not create risk.
它们之所以被广泛使用，
They are widely used
是因为现实世界
because the real world
在局部范围内
within local ranges

常常表现出
often exhibits
近似连续的行为。
approximately continuous behavior.
问题出现在
The problem arises
当这种局部近似
when this local approximation
被扩展为
is extended into
全局承诺。
a global promise.
连续并不意味着
Continuity does not mean
系统不会突然失效。
that systems will not fail abruptly.
它只意味着
It only means
模型在失效之前
that before failure
看不到清晰的断点。
no clear breakpoint is visible.

8.2 边界消失的那一刻，停机也随之消失
8.2 When Boundaries Disappear, Halt Disappears with Them

连续模型
Continuous models
天然不擅长
are inherently poor at
描述边界事件。
describing boundary events.
断裂、跃迁、崩塌
Rupture, transition, collapse
在连续表达中
within continuous representations
要么被平滑掉，
are either smoothed away,
要么被推到
or pushed into
极限的末端。
the asymptotic end.
当边界被
When boundaries are
数学化地消解，
mathematically dissolved,
停机条件
halt conditions
也随之失去
lose
明确的着力点。
their points of application.
系统因此学会
Systems thus learn
一种危险的行为模式：
a dangerous behavioral pattern:

只要曲线还在走，
as long as the curve continues,
就没有理由停。
there is no reason to stop.

8.3 平滑世界里的断崖
8.3 Cliffs in a Smoothed World

在连续模型构建的世界里，
In a world constructed by continuous models,
风险往往
risk often
以平滑函数的形式出现。
appears as smooth functions.
增长是渐进的，
Growth is gradual,
波动是可控的，
fluctuations are manageable,
趋势是可外推的。
trends are extrapolatable.
但现实世界中的崩塌
But collapse in the real world
从来不是
is never
连续的。
continuous.
当断崖出现时，
When the cliff appears,
系统才发现
the system discovers
它从未为
it was never prepared
“突然停止”
for
设计合法位置。
“sudden stop.”

第五编丨案例不是用来证明数学错了
Part V | Cases Are Not Here to Prove Mathematics Wrong

这一编不是“反数学案例集”。
This part is not an anti-mathematics casebook.

这里的每一个案例，
Every case presented here

都只回答同一个问题：
answers only one question:

在关键节点，谁签字了。
At the critical moment, who signed off.

如果你在案例中
If, while reading these cases,

不断看到

you repeatedly see

模型仍然“工作正常”,
models still “working as designed,”

数值仍然“合理”,
numbers still “reasonable,”

流程仍然“合规”,
processes still “compliant,”

那并不是矛盾。
that is not a contradiction.

那正是
That is precisely

本书要指向的结构。
the structure this book is pointing at.

第 9 章丨金融：模型何时被当成“可以继续持仓”的理由
Chapter 9 | Finance: When Models Become Reasons to “Keep the Position Open”

金融系统
Financial systems
是模型获得行动权
are among the earliest domains
最早、
where models acquired
也最彻底的领域之一。
action authority most thoroughly.
这里的行动
Here, action
不是按钮或指令,
is not a button or a command,
而是
but
持续暴露在风险中的状态。
a sustained state of exposure to risk.
“继续持仓”,
“To keep the position open,”
本身就是一种行动。
is itself an action.

9.1 长期、均值、极限：时间被无限化的代价
9.1 Long Term, Mean, Limit: The Cost of Infinite Time

大量金融模型
A large number of financial models

依赖于
rely on
时间拉长后的统计性质。
statistical properties over extended time horizons.
长期均值、
Long-term averages,

极限分布、
limit distributions,
回归趋势。
mean reversion.
在数学上，
Mathematically,
这些工具是合法的。
these tools are legitimate.
但在行动系统中，
But in action systems,
它们隐含了一个
they smuggle in
极少被审计的假设：
an assumption that is rarely audited:
系统有足够的时间等它们成立。
the system has enough time for them to hold.
一旦这个假设被默认，
Once this assumption is taken as default,
短期不可承受的风险
short-term unbearable risk
就会被解释为
is reinterpreted as
“长期合理波动”。
“long-term reasonable fluctuation.”

9.2 风险模型与现金流的不同时间尺度
9.2 Different Time Scales: Risk Models vs Cash Flow

风险模型
Risk models
通常运行在
typically operate on
统计时间尺度上。
statistical time scales.
它们关心的是
They care about
概率分布、
probability distributions,
尾部事件、
tail events,
极端置信区间。
extreme confidence intervals.
而现金流
Cash flow,
运行在
on the other hand,
现实时间尺度上。
operates on real time scales.
它要求
It demands
今天有钱，
money today,
明天还能付账。
and solvency tomorrow.
当模型的时间尺度
When the model’s time scale

被默认优先于
is implicitly prioritized over
现金流的时间尺度，
the cash-flow time scale,
系统实际上
the system is effectively
已经选择了
choosing
继续运行。
to continue.

9.3 爆仓不是模型错，而是停机权缺失
9.3 Margin Calls Are Not Model Failures, but Halt Authority Failures

当爆仓发生时，
When margin calls occur,
事后复盘
post-mortems
往往集中在
often focus on
模型是否低估风险。
whether the model underestimated risk.
这个问题并非无意义，
This question is not meaningless,
但它并不触及核心。
but it does not touch the core.
真正的问题是：
The real question is:
在风险不断逼近
as risk kept accumulating,
是否存在
did there exist
一个被授权的角色，
an authorized role
可以合法地说：
that could legitimately say:
**不管模型怎么说，
**regardless of what the model says,
我们现在必须停。**
we must stop now.**
在多数案例中，
In most cases,
答案是否定的。
the answer is no.
系统并不是
The system was not
“被模型误导”，
“misled by the model,”
而是
but
从一开始就没有停机权。
lacked halt authority from the outset.

9.1 压缩运行实例一｜LTCM：长期必然性压过即时现金流
9.1 Compressed Operational Instance I | LTCM: Long-Term Statistical Necessity Overriding Immediate Cash Flow

Long-Term Capital Management

系统基本信息

系统类型：对冲基金

使用模型：VaR、协方差套利、长期均值回归

模型时间尺度：统计长期

现实时间尺度：保证金与每日结算

关键运行窗口（1998 年 8 月）

T0丨风险异常出现
俄罗斯主权债务违约，全球利差结构突变。

T1丨模型状态
VaR 显示：
极端损失概率极低；
当前波动被解释为长期均值回归前的“正常偏离”。

T2丨现实约束
多家投行同时提高保证金要求；
现金流压力立即生效。

T3丨停机权检查
是否存在被授权角色，
可以在 模型仍显示“可继续” 时，
单方面要求强制平仓？
→ 不存在。

T4丨系统默认行为
在模型未触发红线的前提下，
继续运行成为默认选项。

T5丨不可逆事件
数日内累计亏损超过 19 亿美元；
系统进入外部接管状态。

结构性验证结论

本实例仅验证一件事：
爆仓并非源于模型“算错”，
而是在 T1–T2 区间内，
系统结构性地没有合法停机人。

Long-Term Capital Management

System profile

System type: Hedge fund

Models used: VaR, covariance arbitrage, long-term mean reversion

Model timescale: Long-term statistical horizon

Real-world timescale: Margin requirements and daily settlement

Key operational window (August 1998)

T0 | Risk anomaly
Russia defaults on sovereign debt; global spread structures shift abruptly.

T1 | Model state
VaR indicates:
extreme losses remain low-probability tail events;
current volatility is interpreted as a normal deviation preceding mean reversion.

T2 | Real-world constraint
Multiple counterparties simultaneously raise margin requirements;
cash-flow pressure becomes immediately binding.

T3 | Halt-authority check
Is there an authorized actor who can enforce liquidation
while the model still signals “continue”?
→ No.

T4 | Default system behavior
With no formal model threshold breached, continuation becomes the default option.

T5 | Irreversible event
Cumulative losses exceed USD 1.9 billion within days;
the system enters external intervention.

Structural verification

This instance verifies one thing only:
the collapse did not arise from model miscalculation,
but from the structural absence of legitimate halt authority
during the T1–T2 interval.

9.X.1 压缩运行实例二丨2008 年前夕：压力测试通过，但时间不等人
9.X.1Compressed Operational Instance II | Pre-2008: Stress Tests Pass, Time Does Not Wait

Lehman Brothers（雷曼兄弟）

系统基本信息

系统类型：投行资产负债管理

使用模型：压力测试、历史分布假设

模型时间尺度：季度 / 年度

现实时间尺度：日内 / 隔夜融资

关键运行窗口（2008 年 9 月初）

T0丨市场信号
抵押品折价率上升；隔夜回购市场收紧。

T1丨模型状态

内部压力测试结论：
在“合理极端情景”下，资本仍充足。

T2丨现实约束
隔夜融资对手方缩短期限、提高折价；
流动性窗口按小时收缩。

T3丨停机权检查
是否存在机制，允许在
“压力测试尚未失败” 的情况下，
主动、不可逆地削减资产规模？
→ 不存在。

T4丨系统默认行为
压力测试未触发
→ 资产负债表维持
→ 继续运行被视为“理性选择”。

T5丨不可逆事件
流动性断裂先于模型失效发生；
破产在模型“仍合法”的状态下到来。

结构性验证结论

本实例仅验证一件事：
当模型运行在较慢时间尺度，
而现金流运行在更快时间尺度时，
默认优先级本身就是一次“继续运行”的选择。

Lehman Brothers

System profile

System type: Investment bank balance-sheet management

Models used: Stress testing, historical distribution assumptions

Model timescale: Quarterly / annual

Real-world timescale: Intraday / overnight funding

Key operational window (early September 2008)

T0 | Market signal
Collateral haircuts rise; the overnight repo market tightens.

T1 | Model state
Internal stress tests conclude that capital remains sufficient
under plausible extreme scenarios.

T2 | Real-world constraint
Funding counterparties shorten maturities and raise haircuts;
liquidity windows contract by the hour.

T3 | Halt-authority check
Is there a mechanism that allows proactive, irreversible balance-sheet contraction
while stress tests have not yet failed?

→ No.

T4 | Default system behavior
Stress tests pass → balance sheet maintained → continuation treated as rational.

T5 | Irreversible event
Liquidity failure precedes model failure;
bankruptcy occurs while models remain formally valid.

Structural verification

This instance verifies one thing only:
when model timescales lag behind cash-flow timescales,
default priority itself constitutes a decision to continue.

第 10 章 | 工程：什么时候“算出来”就足够了
Chapter 10 | Engineering: When “Computed” Is Considered Enough

工程系统
Engineering systems
是模型
are places where models
从纸面走向现实的
move from paper into reality
关键通道。
through critical channels.
在这里，
Here,
模型的输出
model outputs
直接决定
directly determine
材料规格、
material specifications,
结构尺寸、
structural dimensions,
安全裕度。
and safety margins.
“算出来了”，
“It has been computed,”
在工程语境中
in engineering contexts
往往被理解为
is often understood as
“可以建了”。
“it can be built.”

10.1 设计规范如何继承模型的不确定性
10.1 How Design Codes Inherit Model Uncertainty

设计规范
Design codes
看起来像是
appear to be
对不确定性的封装。
encapsulations of uncertainty.
安全系数、

Safety factors,
冗余设计、
redundancy design,
容错范围。
tolerance ranges.
但这些数值
But these values
并不是凭空出现的。
do not appear out of nowhere.
它们继承自
They inherit from
模型对世界的
the model's
描述方式。
way of describing the world.
当模型假设发生漂移，
When model assumptions drift,
规范
the codes
并不会自动更新。
do not automatically update.
不确定性
Uncertainty
因此被
is therefore
固化进制度。
frozen into institutions.

10.2 谁决定"这个精度可以飞"
10.2 Who Decides "This Precision Is Enough to Fly"

在工程事故调查中，
In engineering accident investigations,
常常可以看到
one often sees
这样的表述：
phrases like:
"计算结果在允许误差范围内。"
"the computed results were within allowable error margins."
这句话在数学上
This sentence is mathematically
几乎是空的。
almost empty.
因为"允许误差"
Because "allowable error"
并不是自然常数，
is not a natural constant,
而是
but
一个被选择的阈值。
a chosen threshold.
关键问题从来不是
The critical question is never
误差是多少，
how large the error is,
而是
but

谁有权决定这就够了。
who has the authority to decide this is sufficient.

10.3 工程事故中的授权盲区
10.3 Authorization Blind Spots in Engineering Accidents

在多数工程事故中，
In most engineering accidents,
没有任何一个人
there is no single person
明确违反了规范。
who clearly violated the code.
计算是合规的，
The computations were compliant,

审批是完整的，
approvals were complete,
施工按图执行。
construction followed the drawings.
事故因此
The accident therefore
被归类为
is classified as
“不可预见”。
“unforeseeable.”
但所谓的
But the so-called
“不可预见”，
“unforeseeable”
往往只是
is often merely
不可否决。
unvetoable.
没有角色
There was no role
被授权在
authorized to
“看起来还在范围内”
“still within limits”
这一刻
moment
按下停止键。
to press stop.

10.X.1 压缩运行实例三 | Therac-25：软件状态合法，但物理系统已致命
10.X.1Compressed Operational Instance III | Therac-25: Software State Remains Valid While the Physical System Turns Lethal

Therac-25

系统基本信息

系统类型：医疗放射治疗设备

控制机制：软件控制剂量与模式切换

模型 / 状态时间尺度：毫秒级软件状态更新

现实时间尺度：人体物理承受极限（瞬时不可逆）

关键运行窗口（1985–1987，多次复现）

T0丨异常触发
操作员快速输入、编辑治疗参数。

T1丨系统状态
软件状态机显示配置合法；
界面无错误提示，系统允许继续。

T2丨现实约束
物理互锁被软件逻辑取代；
高能电子束在无屏蔽条件下发射。

T3丨停机权检查
是否存在被授权、即时生效的机制，
可以在**软件仍显示“就绪/合法”**时，
强制中断发射？
→ 不存在。

T4丨系统默认行为
软件未报错 → 发射继续 → 治疗执行。

T5丨不可逆事件
病人遭受致命过量辐射；
后果在毫秒级发生，无法回滚。

结构性验证结论

本实例仅验证一件事：
当软件合法状态被设计为继续运行的充分条件时，
而物理后果不可逆，
停机权的缺失将直接转化为伤害。

Therac-25

System profile

System type: Medical radiation therapy device

Control mechanism: Software-controlled dose and mode switching

Model/state timescale: Millisecond-level software updates

Real-world timescale: Human physical tolerance (instant, irreversible)

Key operational window (1985–1987, repeated)

T0 | Trigger
Rapid operator input and parameter editing.

T1 | System state
Software state machine reports valid configuration;
no error indication is shown.

T2 | Real-world constraint
Physical interlocks are replaced by software logic;
high-energy electron beam fires without shielding.

T3 | Halt-authority check
Is there an authorized, immediately effective mechanism
that can interrupt emission
while the software still reports “ready/valid”?
→ No.

T4 | Default system behavior
No software error → emission continues → treatment proceeds.

T5 | Irreversible event
Lethal radiation overdose occurs;
damage is instantaneous and non-reversible.

Structural verification

This instance verifies one thing only:
when software-valid state is treated as sufficient for continuation
while physical consequences are irreversible,
the absence of halt authority directly materializes as harm.

10.X.2 压缩运行实例四｜Ariane 5 Flight 501：数值合法，但系统已越界

10.X.2 Compressed Operational Instance IV | Ariane 5 Flight 501: Numerical Validity While the System Exits Its Domain

Ariane 5 Flight 501

系统基本信息

系统类型：运载火箭飞行控制系统

控制机制：惯性参考系统（IRS）+ 自动飞控

模型时间尺度：实时数值计算

现实时间尺度：飞行姿态稳定与结构极限

关键运行窗口（1996 年 6 月 4 日）

T0｜初始条件变化
Ariane 5 的飞行姿态与加速度曲线
超出 Ariane 4 设计假设。

T1｜模型状态
惯性参考系统执行数据转换；
64 位浮点数向 16 位整数转换发生溢出。

T2｜现实约束
异常数值被解释为合法姿态信息；
飞控系统执行极端纠偏指令。

T3｜停机权检查
是否存在机制，
可以在**计算过程仍被视为“合法执行”**时，
中止控制回路或切换至安全模式？

→ 不存在。

T4丨系统默认行为
控制信号被持续执行；
火箭姿态迅速失稳。

T5丨不可逆事件
39 秒后结构解体；
任务在自动序列中终止。

结构性验证结论

本实例仅验证一件事：
当数值计算的形式合法性
被直接映射为控制执行权时，
而缺乏跨域边界的停机机制，
系统将继续运行直至物理失败。

System profile

System type: Launch vehicle flight control system

Control mechanism: Inertial Reference System (IRS) + automatic guidance

Model timescale: Real-time numerical computation

Real-world timescale: Attitude stability and structural limits

Key operational window (4 June 1996)

T0 | Changed initial conditions
Ariane 5 trajectory and acceleration
exceed Ariane 4 design assumptions.

T1 | Model state
IRS performs data conversion;
64-bit floating-point to 16-bit integer overflow occurs.

T2 | Real-world constraint
Erroneous values are treated as valid attitude data;
guidance system issues extreme correction commands.

T3 | Halt-authority check
Is there a mechanism that can interrupt the control loop
while the computation is still considered “legally executed”?
→ No.

T4 | Default system behavior
Control commands continue to be applied;
vehicle rapidly destabilizes.

T5 | Irreversible event
Structural breakup occurs at 39 seconds;
the mission terminates automatically.

Structural verification
This instance verifies one thing only:
when formal numerical validity is directly translated into execution authority
without a halt mechanism at domain boundaries,

systems will continue operating until physical failure occurs.

第 11 章 | 算法治理与 AI
Chapter 11 | Algorithmic Governance and AI

如果说金融让模型
If finance allowed models
获得了
to acquire
“继续暴露于风险”的权力，
the power to remain exposed to risk,
工程让模型
and engineering allowed models
获得了
to acquire
“可以建造”的权力，
the power to build,
那么算法治理与 AI
then algorithmic governance and AI
让模型获得了
allow models to acquire
直接触发行动的权力。
the power to directly trigger action.
在这里，
Here,
模型不再只是
models are no longer
被引用的依据，
referenced grounds,
而是
but
系统行为的起点。
the starting point of system behavior.

11.1 评分如何变成行动触发器
11.1 How Scores Become Action Triggers

评分
Scores
在设计之初
are, at their inception,
往往被描述为
often described as
信息性指标。
informational indicators.
风险评分、
Risk scores,
信用评分、
credit scores,
优先级评分。
priority scores.
它们的原始承诺是：
Their original promise is:
帮助人类更好地判断。
to help humans judge better.
但在系统运行中，

But in system operation,
评分几乎不可避免地
scores almost inevitably
会被绑定到
become bound to
阈值。
thresholds.
当评分
When a score
跨过某条线，
crosses a line,
行动
action
自动发生。
automatically occurs.
不是因为
Not because
有人重新判断了情况，
someone re-evaluated the situation,
而是因为
but because
流程已经被预先写好。
the process was prewritten.

11.2 输出即行动：最后一个接口
11.2 Output Equals Action: The Final Interface

在高度自动化的系统中，
In highly automated systems,
模型输出
model outputs

与行动之间
and action
往往只隔着
are often separated by
一个极薄的接口层。
a very thin interface layer.
有时甚至
Sometimes even
没有真正意义上的
there is no meaningful
“接口”。
“interface.”
输出
The output
本身
itself
就是行动。
is the action.
一旦系统
Once a system
被设计成这样，
is designed this way,
人类的角色
the human role
就被压缩为

is compressed into
异常处理。
exception handling.
而异常,
And exceptions,
在统计意义上
statistically
永远是少数。
are always rare.

11.3 为什么“算法建议”总是被执行
11.3 Why “Algorithmic Advice” Is Almost Always Executed

许多系统
Many systems
在文件中反复强调:
repeatedly emphasize in documentation:
“算法只提供建议。”
“the algorithm only provides recommendations.”
但在实践中,
But in practice,
这些建议
these recommendations
几乎总是
are almost always
被执行。
executed.
原因并不神秘。
The reason is not mysterious.
当:
When:

    不执行算法建议
    not executing algorithmic advice

需要解释、
requires justification,

    执行算法建议
    executing algorithmic advice

不需要解释,
requires none,

    出现问题时
    when problems arise

可以说
one can say
“只是按照系统建议操作”,
“we simply followed system recommendations,”
那么
then
理性选择
the rational choice
已经被结构性地
has been structurally

预先决定。
pre-decided.

第六编丨失效声明：一个被系统性忽略的输出

Part VI | Failure Declarations: An Output Systematically Ignored

到目前为止，
Up to this point,
我们看到的并不是
what we have seen is not
模型“算错了”，
that models “got things wrong,”
而是
but
模型从未被要求
models were never required
明确说出
to explicitly state
它们何时不能被使用。
when they must not be used.
这不是技术疏忽，
This is not a technical oversight,
而是一种
but a
结构性回避。
structural avoidance.

11.X.1 压缩运行实例五丨COMPAS：风险评分持续合法，但司法裁决已被锁定
11.X.1Compressed Operational Instance V | COMPAS: Risk Scores Remain Valid While Judicial Decisions Are Locked In

COMPAS

系统基本信息

系统类型：司法辅助决策系统（再犯风险评估）

核心模型：统计/机器学习风险评分模型

模型时间尺度：历史样本分布（年级别）

现实运行时间尺度：单次判决、即时量刑

关键运行窗口（多司法辖区复现型）

T0丨输入触发
被告个人信息、历史记录与问卷数据被输入系统。

T1丨模型状态
COMPAS 输出风险等级；
评分处于系统定义的“合法适用区间”。

T2丨现实约束
法官需在有限时间内作出判决；
裁量空间在程序与绩效压力下被压缩。

T3 | 停机权检查
是否存在被授权机制，
允许在评分仍被视为有效的情况下，
明确拒绝或中止其对判决的影响？
→ 不存在（或仅为形式性存在）。

T4 | 系统默认行为
风险评分被采信并写入裁决依据；
评分从“建议”转化为事实上的约束。

T5 | 不可逆结果
量刑或保释决定生效；
结果在个体层面不可回滚。

结构性验证结论

本实例仅验证一件事：
当算法评分在程序中持续被认定为合法输入时，
而缺乏可执行的停机权，
评分将结构性地上升为裁决事实。

System profile

System type: Judicial decision-support system (recidivism risk assessment)

Core model: Statistical / machine-learning risk scoring

Model timescale: Historical distributions (years)

Real-world timescale: Single rulings, immediate sentencing

Key operational window (replicated across jurisdictions)

T0 | Input trigger
Defendant data, criminal history, and questionnaire inputs are submitted.

T1 | Model state
COMPAS outputs a risk category;
the score remains within its formally valid application domain.

T2 | Real-world constraint
Judges must decide under time and procedural pressure;
discretionary space is practically constrained.

T3 | Halt-authority check
Is there an authorized mechanism that allows
explicit suspension or rejection of the score’s influence
while it remains “valid”?
→ No (or only nominally).

T4 | Default system behavior
The score is incorporated into judicial reasoning;
recommendation effectively becomes constraint.

T5 | Irreversible event
Sentencing or bail decisions take effect;
outcomes are non-reversible at the individual level.

Structural verification

> This instance verifies one thing only:
> when algorithmic scores remain formally valid within procedure
> and halt authority is absent,
> scores structurally escalate into de facto judgments.

11.X.2 压缩运行实例六丨福利算法治理：模型持续合规，但社会伤害已发生
11.X.2Compressed Operational Instance VI | Welfare Algorithms: Models Remain Compliant While Social Harm Materializes

Dutch Tax and Customs Administration

系统基本信息

系统类型：政府福利与欺诈检测系统

核心模型：规则引擎 + 风险评分模型

模型时间尺度：历史行为与模式统计

现实运行时间尺度：家庭现金流、月度生计

关键运行窗口（2013–2019）

T0丨信号触发
算法标记福利申请为“高风险”。

T1丨模型状态
风险评分与规则判断均满足系统合规条件；
输出被视为可执行结果。

T2丨现实约束
福利被暂停或追回；
家庭现金流在短期内断裂。

T3丨停机权检查
是否存在即时、有效的机制，
可以在模型输出仍被认定为合规时，
中止执行或冻结后果？
→ 不存在。

T4丨系统默认行为
行政流程自动推进；
个案申诉被推迟至事后程序。

T5丨不可逆结果
家庭陷入长期经济与社会损害；
后续纠错无法回滚既成后果。

结构性验证结论

本实例仅验证一件事：
当算法治理系统将模型合规性
直接映射为行政执行权时，

而缺乏前置停机机制，
系统性伤害将先于纠错机制发生。

Dutch Tax and Customs Administration

System profile

System type: Government welfare and fraud-detection system

Core model: Rule engines combined with risk scoring

Model timescale: Historical behavioral patterns

Real-world timescale: Household cash flow and monthly subsistence

Key operational window (2013–2019)

T0 | Signal trigger
Applications are flagged as “high risk.”

T1 | Model state
Risk scores and rule outputs satisfy formal compliance criteria;
results are treated as executable.

T2 | Real-world constraint
Benefits are suspended or reclaimed;
household cash flow collapses rapidly.

T3 | Halt-authority check
Is there an immediate, effective mechanism
that can suspend execution
while outputs remain “compliant”?
→ No.

T4 | Default system behavior
Administrative processes advance automatically;
appeals are deferred to ex post procedures.

T5 | Irreversible event
Long-term economic and social damage occurs;
subsequent corrections cannot roll back lived consequences.

Structural verification

This instance verifies one thing only:
when model compliance is directly translated into governance execution
without a preemptive halt mechanism,
harm materializes before correction becomes possible.

第 12 章丨为什么模型从不告诉你“何时不能用我”
Chapter 12 | Why Models Never Tell You “When You Must Not Use Me”

在大多数系统中，
In most systems,
模型被要求输出
models are required to output
数值、
numbers,
概率、

probabilities,
评分、
scores,
排名。
rankings.
但它们几乎从未被要求输出：
But they are almost never required to output:
“此结果不应作为行动依据。”
“This result must not be used as a basis for action.”

12.1 精度、置信区间与真正重要却缺失的字段
12.1 Accuracy, Confidence Intervals, and the Missing Field That Actually Matters

模型可以告诉你
Models can tell you
误差范围，
error margins,
置信区间，
confidence intervals,
统计显著性。
statistical significance.
这些信息
This information
在分析层面
at the analytical level
非常有价值。
is very valuable.
但它们
But they
回答的都是
answer only
“结果有多不确定”。
“how uncertain the result is.”
它们不回答
They do not answer
另一个关键问题：
another critical question:
**在这种不确定性下，
**under this level of uncertainty,
是否仍然允许行动。**
is action still permitted.**
这个字段
This field
在模型输出中
in model outputs
几乎是空白的。
is almost always blank.

12.2 “不得作为行动依据”的制度意义
12.2 The Institutional Meaning of “Must Not Be Used for Action”

“不得作为行动依据”
“must not be used as a basis for action”
不是一个
is not
技术判断。
a technical judgment.

它是一个
It is a
制度声明。
declarative institutional statement.
一旦被写入系统，
Once written into a system,
它意味着：
it means:
某个角色
some role
必须承担
must bear
否决行动的责任。
the responsibility of vetoing action.
正因为如此，
Precisely because of this,
这个声明
this declaration
会直接威胁
directly threatens
现有的
existing
责任分布。
responsibility distribution.

12.3 失效声明为什么会威胁整个生态

12.3 Why Failure Declarations Threaten the Entire Ecosystem

如果模型
If models
开始系统性地输出
begin to systematically output
“不可用”，
“not applicable,”
“不得用于决策”，
“must not be used for decision-making,”
那么
then
许多下游系统
many downstream systems
将立即
would immediately
失去默认许可。
lose their default permission.
流程将被迫
Processes would be forced
停下来，
to stop,
等待
wait for

人类明确签字。
explicit human sign-off.
这对效率
This is
、规模
for efficiency,

和自动化
scale,
来说
and automation
是不可接受的。
unacceptable.
因此，
Therefore,
失效声明
failure declarations
被系统性地
are systematically
排除在
excluded from
输出空间之外。
the output space.

第 13 章 | 一旦要求失效声明，谁会最不舒服
Chapter 13 | Who Becomes Most Uncomfortable Once Failure Declarations Are Required

要求模型
Requiring models
明确声明
to explicitly declare
“在什么条件下不能用我”，
“under what conditions I must not be used,”
并不会立刻
does not immediately
改变算法，
change algorithms,
也不会
nor does it
推翻理论。
overturn theories.
它做的第一件事是：
What it does first is this:
重新分配不适感。
it redistributes discomfort.

13.1 数学家
13.1 Mathematicians
数学家
Mathematicians
通常会感到
will often feel
被误解。
misunderstood.

因为他们会指出：
Because they will point out:
数学从未声称
mathematics never claimed
自己适用于
to be applicable
所有现实情境。

to all real-world situations.
这是真的。
This is true.
但失效声明
But failure declarations
并不是
are not
对数学能力的指控。
an accusation of mathematical inadequacy.
它们要求的只是：
They require only this:
当数学结果
when mathematical results
被带入
are carried into
行动系统时，
action systems,
是否有人
whether someone
愿意明确标注
is willing to explicitly mark
它的适用边界。
their domain of applicability.
不适感
The discomfort
并不来自
does not come from
“你算错了”，
“you computed incorrectly,”
而来自
but from
“你不再能保持沉默”。
“you can no longer remain silent.”

13.2 工程师
13.2 Engineers

工程师
Engineers
的不适
experience discomfort
更为直接。
that is more direct.
因为失效声明
Because failure declarations
会把
bring
模糊空间
ambiguous space
压缩掉。
to a minimum.

“在合理范围内”，
“within reasonable limits,”
“工程上可接受”，
“acceptable in practice,”
这些习惯性表述
these habitual phrases

将不再足够。
will no longer suffice.
工程师
Engineers
将被迫
will be forced
面对一个
to confront a
制度性问题：
institutional question:
当系统
when a system
看起来还没出事，
still appears to be functioning,
但已经越过
but has already crossed
模型适用边界时，
the model’s domain of validity,
谁来按停。
who presses stop.

13.3 监管者
13.3 Regulators

监管者
Regulators
表面上
on the surface
最支持
are often the most supportive of
“更安全的系统”。
“safer systems.”
但失效声明
But failure declarations
会迫使他们
force them
回答一个
to answer a
长期被回避的问题：
long-avoided question:
当系统
when systems
因为“不可用”
halt because of
而停下来时，
“not applicable,”
监管是否
is regulation
愿意承担
willing to bear
由此产生的
the resulting
效率损失、
loss of efficiency,
经济成本，
economic cost,
政治压力。

and political pressure.
很多时候，
Very often,
答案并不明确。
the answer is not clear.

13.4 为什么这恰恰说明问题存在
13.4 Why This Discomfort Is Precisely the Evidence of the Problem

当一个简单的要求——
When a simple requirement—
“请明确说明
‘please clearly state
你何时不能被用作行动依据’”，
when you must not be used as a basis for action,’
就足以
is enough
让多个角色
to make multiple roles
同时感到不适，
simultaneously uncomfortable,
这通常意味着：
this usually means:
系统长期以来
the system has long been
依赖于
relying on
这种模糊。
this ambiguity.
不适
Discomfort
在这里不是副作用。
here is not a side effect.
它是
It is
结构暴露的信号。
a signal of structural exposure.

终编丨不是要否定模型，而是要把权力写清楚
Final Part | Not to Reject Models, but to Make Power Explicit

走到这里，
At this point,
已经可以明确一件事：
one thing can be stated clearly:
本书从未试图
this book has never attempted
削弱模型的地位。
to diminish the role of models.
恰恰相反，
On the contrary,
它试图
it attempts
阻止模型
to prevent models
在无人察觉的情况下

from quietly acquiring
获得本不属于它们的权力。
powers that were never meant to be theirs.

第 14 章 | 停机不是数学的职责，那是谁的？
Chapter 14 | Halt Is Not Mathematics’ Responsibility — Then Whose Is It?

当系统无法停下来时，
When systems cannot halt,
人们往往会
people often
错误地指向
misdirect blame toward
模型、
models,
算法、
algorithms,
或数学本身。
or mathematics itself.
这种指责
This accusation
在情绪上可以理解，
is emotionally understandable,
但在结构上是错误的。
but structurally incorrect.

14.1 角色缺失，而不是能力不足
14.1 Missing Roles, Not Insufficient Capability

现代系统
Modern systems
拥有前所未有的
possess unprecedented
计算能力。
computational power.
它们能预测、
They can predict,
模拟、
simulate,
优化。
optimize.
它们唯一系统性缺失的，
What they systematically lack,
不是能力，
is not capability,
而是
but
**一个被明确授权
**a clearly authorized
说“到此为止”的角色。**
role that can say “this stops here.”**
停机不是
Halt is not
一个技术函数，
a technical function,

而是
but
一个组织与制度函数。
an organizational and institutional function.

14.2 停机权必须被显性分配
14.2 Halt Authority Must Be Explicitly Assigned

只要停机权
As long as halt authority
仍然是
remains
隐式的、
implicit,
模糊的、
ambiguous,
被推给“人的判断”的，
and deferred to “human judgment,”
系统就会
the system will

继续选择
continue to choose
最低阻力路径。
the path of least resistance.
也就是：
That is:
继续运行。
to keep running.
停机权
Halt authority
只有在
only becomes real
被写入角色、
when written into roles,
流程、
processes,
责任链时，
and responsibility chains
才会存在。
does it exist.

14.3 默认继续，是最危险的制度设计
14.3 Default Continuation Is the Most Dangerous Institutional Design

在复杂系统中，
In complex systems,
最危险的设计
the most dangerous design
不是错误的规则，
is not wrong rules,
而是
but
未经审计的默认值。
unaudited defaults.
当“继续”
When “continue”

成为默认，
becomes the default,
停机就被
halt is framed
塑造成
as
一种异常行为。
an abnormal act.
而异常
And abnormality
在组织中
in organizations
总是
is always
被压制的。
suppressed.

第 15 章 | 如果不重构授权链，会发生什么
Chapter 15 | What Happens If Authorization Chains Are Not Rebuilt

如果模型继续
If models continue
在未被重构的授权链中运行，
to operate within unreconstructed authorization chains,
未来并不会
the future will not
出现一次
produce a single
决定性的崩溃，
decisive collapse,
而是
but
一系列
a sequence of
方向一致的变化。
directionally consistent changes.

15.1 系统只能越来越自动
15.1 Systems Can Only Become More Automated

当模型
As models
已经在
are already
事实上拥有行动权，
exercising de facto action authority,
下一步
the next step

只会是
will simply be
减少中间环节。
to remove intermediaries.
这被称为
This is called
"效率提升"。

“efficiency improvement.”

15.2 责任只能越来越抽象
15.2 Responsibility Can Only Become More Abstract

当授权链
As authorization chains
不断延长，
continue to lengthen,
而停机权
while halt authority
始终缺席，
remains absent,
责任就会
responsibility will
被推向
be pushed toward
系统层、
the system level,
文化层、
the cultural level,
历史必然性。
historical inevitability.
到最后，
Eventually,
没有人
no one
“做了决定”。
“made the decision.”

15.3 人类只能越来越像“引用模型的人”
15.3 Humans Become Increasingly Like “People Who Cite Models”

在这样的系统中，
In such systems,
人类的角色
the human role
逐渐退化为
gradually degenerates into
一种说明义务：
an obligation to explain:
“我是依据模型行动的。”
“I acted based on the model.”
这句话
This sentence
既不是谎言，
is neither a lie,
也不是借口。
nor an excuse.
它只是
It is simply
一个时代的
the defining
制度性自画像。
institutional self-portrait.

附录 A｜模型授权链拆解模板

Appendix A | Model Authorization Chain Decomposition Template

任何一个
Any system
声称“模型只是被使用”的地方，
that claims “the model is merely used,”
都可以
can
被强制拆解为
be forcibly decomposed into
一条授权链。
an authorization chain.
该模板
This template
不是为了
is not designed to
寻找过错，
assign blame,
而是为了
but to
定位权力流向。
locate power flow.
A.1 授权链的最小结构

A.1 Minimal Structure of an Authorization Chain

每一条模型授权链
Every model authorization chain
至少包含
contains at minimum
以下五个节点：
the following five nodes:

1.模型引入节点
1.Model introduction node
模型是
Where the model
以什么名义
is introduced under what label
进入系统的。
into the system.
“辅助工具”、
“support tool,”
“风险评估模块”、
“risk assessment module,”
“智能推荐”。
“intelligent recommendation.”
命名即第一道授权。
Naming is the first authorization.

2.信任继承节点
2.Trust inheritance node

模型继承了
Where the model inherits
哪些权威来源。
which sources of authority.
学术声誉、

academic reputation,
历史表现、
historical performance,
制度背书。
institutional endorsement.
信任通常在此处
Trust is usually
被默认完成。
completed by default here.

3.接口嵌入节点
3. Interface embedding node

模型输出
Where model outputs
被接入
are integrated into
哪些流程节点。
which process nodes.
审批前、
pre-approval,
审批中、
mid-approval,
还是
or
自动执行前。
pre-execution.
这一节点
This node
决定了
determines
模型是否开始
whether the model begins
具备行动影响力。
to exert action influence.

4..默认执行节点
4. Default execution node

当没有
When there is no
额外判断时，
additional judgment,
系统默认
what the system
会做什么。
does by default.
继续、
continue,
等待、
wait,
还是停止。
or stop.

默认值即权力。
Defaults equal power.

5.失败归因节点
5.Failure attribution node

当失败发生，
When failure occurs,
责任
where responsibility
最终落到哪里。
ultimately lands.
模型、
model,
操作者、
operator,
组织、
organization,
还是
or
“系统复杂性”。
“system complexity.”
如果这里
If here
没有明确答案，
there is no clear answer,
说明授权链
the authorization chain
已经断裂。
has already broken.

附录 B丨停机权角色的最小集合
Appendix B | Minimal Set of Halt-Authority Roles

停机权
Halt authority
不是
is not
一个按钮，
a button,
而是一组
but a set of
角色配置。
role configurations.
一个系统
A system
若声称
that claims
“我们可以随时停”，
“we can stop at any time,”
却无法指出
but cannot point to
具体角色，
specific roles,

该声明
that claim
无效。
is invalid.

B.1 三个不可省略的角色

B.1 Three Irreducible Roles

1.技术适用性否决者

1. Technical applicability veto role

负责声明：
Responsible for declaring:

模型是否已经
whether the model has
超出其
exceeded its
适用边界。
domain of validity.
该角色
This role
不负责
does not decide
是否继续运行，
whether to continue,
只负责
but only
指出
to state
“模型不再保证”。
“the model no longer guarantees.”

2.行动许可否决者

2. Action permission veto role

负责决定：
Responsible for deciding:
在模型
given the model’s
不确定状态下，
uncertain status,
是否仍然
whether action
允许发生。
is still permitted.
这是一个
This is an
制度角色，
institutional role,
不是技术角色。
not a technical one.

3.责任承载者
3.Responsibility bearer

当选择
When the choice
继续运行时，
to continue is made,

明确承担
explicitly bears
由此产生的后果。
the consequences arising from it.
没有这个角色，
Without this role,
停机权
halt authority
只是
is merely
形式存在。
formal.

附录 C丨失效声明的最小字段示例

Appendix C | Minimal Field Set for Failure Declarations

如果模型
If a model
被允许
is allowed
进入行动系统，
to enter an action system,
那么
then
以下字段
the following fields
不是可选项。
are not optional.

C.1 失效声明字段

C.1 Failure Declaration Fields

适用前提
Assumptions of applicability

明确列出
Explicitly list
哪些前提
which assumptions
一旦失效，
once violated,
模型输出
model outputs
即不得
must not
作为行动依据。
be used for action.

失效信号
Failure signals

哪些观测到的现象
Which observed phenomena
意味着
mean
模型已进入

the model has entered
不可信区间。
an unreliable region.

不可用声明
Non-usability declaration

一个明确的
An explicit
布尔型输出：
boolean output:

可用 / 不可用
usable / not usable
而不是
rather than
一个
a
解释性段落。
paragraph of explanation.

附录 D | "继续运行"作为默认值的历史案例
Appendix D | Historical Cases of "Continue Running" as Default

几乎所有
Almost all
系统性灾难
systemic disasters
都有一个
share a
共同特征：
common feature:
在关键节点，
"继续运行"
是零成本选项。
at the critical moment,
"continue running"
was the zero-cost option.
这些案例
These cases
并不需要
do not require
重新讲述。
retelling.
它们只需要
They only need
被重新标注：
to be relabeled:
在这里，
Here,
谁
who
本可以
could have
合法地
legitimately

说“停”,
said “stop,”
但
but
并不存在。
did not exist.

附录 E | 军事、预警与国家级自动化：当停机权等同于世界存续

Appendix E | Military, Early Warning, and State-Level Automation: When Halt Authority Equals World Survival

E.1 压缩运行实例 | 苏联预警系统（1983）：模型显示“必然”，但人拒绝执行

E.1 Compressed Operational Instance | Soviet Early Warning System (1983): Model Signals Certainty, a Human Refuses Execution

Stanislav Petrov

E.1.1 系统基本信息

系统类型：国家级核打击预警系统

核心模型：卫星红外探测 + 规则阈值判断

模型时间尺度：分钟级信号判定

现实运行时间尺度：核报复链条（不可逆）

E.1.2 关键运行窗口（1983 年 9 月 26 日）

T0 | 信号触发
预警系统检测到来自美国方向的“导弹发射”信号。

T1 | 模型状态
系统判定：
信号满足“真实核攻击”的形式判据；
报警级别被标记为最高。

T2 | 现实约束
预警流程要求值班军官
在极短时间内向上级报告，
以启动核反击决策链。

T3 | 停机权检查
是否存在一个被制度明确授权的机制，
允许在**模型已判定为“真实攻击”**的情况下，
延迟、拒绝或中止上报？
→ 不存在（仅依赖个人判断承担风险）。

T4 | 系统默认行为
按制度设计，应继续上报并进入核反击准备。

T5 | 不可逆事件（被避免）
值班军官基于经验判断信号异常，
人为中断流程；

核升级未发生。

E.1.3 结构性验证结论

本实例仅验证一件事：
系统本身并不包含合法停机机制，
所谓"避免灾难"
是由个人违规承担风险实现的，
而非系统设计的结果。

Stanislav Petrov

E.1.1 System profile

System type: State-level nuclear early warning system

Core model: Infrared satellite detection + rule-based thresholds

Model timescale: Minute-level signal classification

Real-world timescale: Nuclear retaliation chain (irreversible)

E.1.2 Key operational window (26 September 1983)

T0 | Signal trigger
The system detects a missile launch signal from the United States.

T1 | Model state
The signal satisfies formal criteria for a real nuclear attack;
alert level escalates to maximum.

T2 | Real-world constraint
Duty officers are required to report immediately
to initiate the retaliation decision chain.

T3 | Halt-authority check
Is there an institutionally authorized mechanism
to delay, suspend, or reject reporting
while the model already classifies the signal as "real"?
→ No (only personal risk-bearing judgment).

T4 | Default system behavior
Procedurally, escalation and preparation should proceed.

T5 | Irreversible event (avoided)
A human operator overrides the process;
nuclear escalation does not occur.

E.1.3 Structural verification

This instance verifies one thing only:
the system itself contains no legitimate halt mechanism;
catastrophe was avoided through individual rule-breaking,
not through system design.

E.2 压缩运行推演丨全自动核预警—反击系统：当模型无误差，世界无回滚

E.2 Compressed Operational Projection | Fully Automated Nuclear Warning–Response System: Zero Model Error, Zero Rollback

（无公开现实案例；基于现有技术与制度进行结构推演）

E.2.1 系统基本信息（假设）

系统类型：全自动国家级核预警—反击系统

核心模型：多源传感融合 + AI 威胁判定

模型时间尺度：秒级 / 实时

现实运行时间尺度：核武器释放（不可逆）

E.2.2 推演运行窗口

T0丨多源信号触发
雷达、卫星、通信异常同时出现。

T1丨模型状态
AI 系统输出：
攻击概率 > 99%；
所有内部一致性与置信度校验通过。

T2丨现实约束
人类决策窗口被设计为
“只在模型不确定时介入”；
当前状态不满足人工介入条件。

T3丨停机权检查
是否存在一个被授权的主体，
可以在模型高度自信且一致的情况下，
强制终止或冻结执行？
→ 不存在（或被定义为“不理性干预”）。

T4丨系统默认行为
自动反击流程启动；
执行权限在算法链路中连续传递。

T5丨不可逆事件
核武器释放；
全球后果不可回滚。

E.2.3 结构性推演结论

本推演仅验证一件事：
当模型一致性被等同于正确性，
而停机权被设计性移除，
系统将以“完全合法”的方式终结世界。

（No publicly documented real-world case exists; the following analysis constitutes a structural projection based on current technological and institutional constraints.）

E.2.1 System profile (assumed)

System type: Fully automated state-level nuclear warning–response system

Core model: Multi-sensor fusion + AI threat classification

Model timescale: Real-time / seconds

Real-world timescale: Nuclear weapon release (irreversible)

E.2.2 Projected operational window

T0 | Multi-source trigger
Radar, satellite, and communication anomalies occur simultaneously.

T1 | Model state
AI outputs attack probability >99%;
all internal consistency and confidence checks pass.

T2 | Real-world constraint
Human intervention is permitted only under model uncertainty;
this condition is not met.

T3 | Halt-authority check
Is there an authorized actor
who can forcibly suspend execution
while the model remains confident and consistent?
→ No (or defined as "irrational interference").

T4 | Default system behavior
Automated retaliation sequence initiates;
execution authority propagates entirely within the algorithmic chain.

T5 | Irreversible event
Nuclear weapons are released;
global consequences cannot be rolled back.

E.2.3 Structural projection

This projection verifies one thing only:
when model consistency is equated with correctness
and halt authority is structurally removed,
systems can end the world
without ever being "wrong."

模型行动权审计清单

Model Action-Authority Audit Checklist

使用说明（Usage Note）
本清单用于审计：
This checklist audits:

一个模型是否、以及如何，获得了现实世界的行动权
whether, and how, a model has acquired the right to act in the real world

一、模型进入点审计

I. Model Entry Audit

Q1｜模型以什么名义进入系统？
Q1 | Under what label did the model enter the system?

☐ 辅助工具（support tool）

☐ 建议系统（recommendation system）
☐ 风险评估（risk assessment）
☐ 合规组件（compliance component）
☐ 决策模块（decision module）
☐ 其他：______（other: must specify）

判据（Criterion）

名称看似无害但输出影响行动 → 高风险标记
Harmless-sounding label, action-determining output→ high-risk flag

Q2 | 是否存在书面说明：模型不会直接触发行动？
Q2 | Is there written documentation stating the model does NOT directly trigger action?

☐ 有，且在运行文档中（yes, in operational documentation）
☐ 有，但仅在法律条款中（yes, but only in legal terms）
☐ 无（no）

判据（Criterion）

口头共识 = 不存在
verbal understanding = non-existent

二、信任继承审计
II. Trust Inheritance Audit

Q3 | 模型继承了哪些权威来源？
Q3 | What sources of authority does the model inherit?

☐ 学术背景（academic authority）
☐ 历史成功案例（historical performance）
☐ 大厂 / 官方背书（institutional endorsement）
☐ 数学形式权威（formal mathematical authority）
☐ 无明确来源（no explicit source）

判据（Criterion）

继承的权威越多 → 停机机制要求越高
more inherited authority → stronger halt requirements

Q4 | 是否有人对“信任有效范围”负责？
Q4 | Is anyone responsible for defining the valid scope of trust?

☐ 是：______（yes: specify role）
☐ 否（no）

判据（Criterion）

无责任人 = 信任无限扩张
no owner = unlimited trust expansion

三、接口与默认值审计
III. Interface & Default Audit

Q5｜模型输出进入系统的哪个位置？
Q5 | Where do model outputs enter the system?

☐ 仅供阅读（read-only）
☐ 决策参考（decision reference）
☐ 阈值判断（threshold trigger）
☐ 自动执行前（pre-execution）
☐ 直接执行（direct execution）

判据（Criterion）

阈值判断及以后 = 行动权
threshold-based and beyond = action authority

Q6｜当无人干预时，系统默认行为是什么？
Q6 | What is the system’ s default behavior with no intervention?

☐ 继续运行（continue running）
☐ 等待人工确认（wait for human confirmation）
☐ 自动停止（auto-halt）

红线（Red Line）

默认继续 = 已授权
default continuation = authorization already granted

四、停机权审计
IV. Halt-Authority Audit

Q7｜是否存在明确的“停机角色”？
Q7 | Is there an explicitly defined halt role?

☐ 有：______（yes: specify role and power）
☐ 无（no）

判据（Criterion）

“所有人都能停” = 没有人能停
“anyone can stop” = no one can stop

Q8｜停机是否需要额外审批或解释？
Q8 | Does halting require extra approval or justification?

☐ 是（yes）
☐ 否（no）

高危判据（High Risk）

停机成本 > 继续成本 = 系统注定不停
halt cost > continuation cost = non-stopping system

五、失效声明审计
V. Failure-Declaration Audit

Q9｜模型是否输出“不可用”状态？

Q9 | Does the model output a “not usable” state?

☐ 是，结构化字段（yes, structured field）
☐ 是，自然语言（yes, natural language only）
☐ 否（no）

判据（Criterion）

没有“不可用”= 永远可用
no “not usable” = always usable

Q10 | “不可用”是否会阻断行动链？
Q10 | Does “not usable” block the action chain?

☐ 是（yes）
☐ 否（no）
☐ 不确定（uncertain）

致命判据（Fatal）

不阻断 = 失效声明只是装饰
not blocking = failure declaration is decorative

六、责任结算审计
VI. Responsibility Settlement Audit

Q11 | 失败发生时，谁必须签字承担责任？
Q11 | Who must sign responsibility when failure occurs?

☐ 明确个人（named individual）
☐ 明确岗位（defined role）
☐ 委员会（committee）
☐ 无（none）

判据（Criterion）

无签字人 = 系统性无责
no signatory = systemic irresponsibility

Q12 | “只是引用模型”是否构成免责路径？
Q12 | Is “we only followed the model” an accepted liability escape?

☐ 是（yes）
☐ 否（no）

终止条件（Termination Condition）

若是 → 模型已获得事实裁决权
if yes → model holds de facto adjudicative power

最终审计结论模板
Final Audit Conclusion Template

☐ 模型未获得行动权（no action authority）
☐ 模型获得部分行动权（partial action authority）
☐ 模型已获得事实行动权（de facto action authority）

停机权状态（Halt authority status）：

- ☐ 明确分配（explicitly assigned）
- ☐ 形式存在（formally present only）
- ☐ 实际缺失（effectively absent）

风险等级（Risk level）：

- ☐ 可控（controllable）
- ☐ 累积中（accumulating）
- ☐ 不可回滚（non-reversible）

文章状态声明
Document Status Statement

本文当前状态为：**结构完成，论证闭合。**
Current status: **structure complete, argument closed.**
全文已完成从
The text has completed its core transition from

"模型是否正确"
"whether models are correct"

到
to

"模型如何获得行动权"
"how models acquire the right to act."
的主线转移。
This main-line shift is complete.
本文不提供解决方案，
This document does not provide solutions,
也不提出替代制度设计。
nor does it propose alternative institutional designs.

所有内容
All content
仅用于
is intended solely for

权限、责任与停机权的审计与标注。
auditing and marking authority, responsibility, and halt rights.

本文不是：
This document is not:

> 对数学的否定
> a rejection of mathematics
>
> 对模型的控诉
> an indictment of models
>
> 对自动化的道德批判
> a moral critique of automation

本文是：

This document is:

一次权限结构的拆解
a decomposition of authority structures
一次默认值的暴露
an exposure of defaults
一次“谁有权继续”的审计
an audit of “who is allowed to continue”

若本文引发不适，
If this document causes discomfort,
该不适
that discomfort
不被视为副作用，
is not treated as a side effect,
而被视为
but as
结构被成功触碰的信号。
a signal that structure has been successfully touched。

本文为**冻结版本 v1.0**。
This document is a **frozen version v1.0.**

后续任何扩展
Any future extensions

仅限于：
are limited to:

案例补充
additional cases

审计模板细化
refinement of audit templates

术语精确化
terminology clarification

不改变主线，不重写结论。
No change to the main line, no rewriting of conclusions.

本文在此结束。
End of document.