Big-Bang Reforms

Anton Kolotilin and Hongyi Li (UNSW)

June 2024

Entangled systems

- Complicated systems: accumulate design elements incrementally

Entangled systems

- Complicated systems: accumulate design elements incrementally

- Elements are interdependent (*entangled* with each other).
- Entanglements inhibit change

- Change may be delayed → inefficiencies persist and accumulate

- Examples:
  - Software: MS-DOS → Windows → Windows 95 ...
  - Public policy: tax, healthcare

This paper:

When complicated, entangled systems face continuous pressure to change,

- Should they adapt *continuously*?
- Or *abruptly* and dramatically?

Applications to various settings:

- organizations
- public policy
- software development

This paper:

When complicated, entangled systems face continuous pressure to change,

- Should they adapt *continuously*?
- Or *abruptly* and dramatically?

Applications to various settings:

- organizations
- public policy
- software development

The Model

- Time is continuous, $t \geq 0$.
- System S_t is a mass of infinitesimal ($dm \to 0$) elements.
- Each element is characterized by its:

vintage	quality	dependencies
(time of birth)	(good or bad)	(hidden position in network)

How elements work

- Designer adds and deletes elements over time.
- Each element is initially good, randomly turns bad at rate $\lambda > 0$.
- Bad elements remain bad forever (until deletion).

- Designer's flow payoff:

$$\pi_t = \underbrace{m_G(t)}_{\substack{\text{mass of}\\ \text{good elements}}} - c \underbrace{m_B(t)}_{\substack{\text{mass of}\\ \text{bad elements}}}.$$

- Myopic Designer seeks to maximize

$$\mathbb{E}\left[\frac{d\pi_t}{dt}\right].$$

How elements work

- Designer adds and deletes elements over time.
- Each element is initially good, randomly turns bad at rate $\lambda > 0$.
- Bad elements remain bad forever (until deletion).

- Designer's flow payoff:

$$\pi_t = \underbrace{m_G(t)}_{\substack{\text{mass of} \\ \text{good elements}}} - c \underbrace{m_B(t)}_{\substack{\text{mass of} \\ \text{bad elements}}} .$$

- **Myopic** Designer seeks to maximize

$$\mathbb{E}\left[\frac{d\pi_t}{dt}\right].$$

Friction

- Each newborn element randomly, immutably endowed with directed links *to* existing elements:

$$\text{each new element} \quad \underset{\text{depends on}}{\longrightarrow} \quad \text{each existing element}$$

$$\text{with probability} \quad \kappa \cdot dm.$$

- *Friction*: whenever element x is deleted,

$$\text{all dependents } (y \to x),$$
$$\text{dependents of dependents } (z \to y \to x), \text{ etc}$$
$$\text{are also instantly deleted.}$$

Control

At each instant t, the designer may:

- add new (good) elements at bounded rate $a_t \in [0, \alpha]$ (mass per unit time).
- delete any elements in S_t.
  - * all direct + indirect dependents of deleted elements also instantly deleted.
  - * no rate constraint: can delete discrete mass of elements instantly.

Designer's information set

The Designer:

- Observes the type (good/bad) and vintage $\tau \le t$ of each element in S_t.
- Understands the network formation process,
  but *doesn't observe time-t network*.

* Upon deleting element, immediately observes deletion of its dependents.

Dependency network: summary of features

- Homogenous, 'detail-free' network;
  elements are distinguished only by their (ordinal) vintage and kind.

- Entanglement is 'limited':
  no 'runaway' chain deletions.

- Entanglement is 'nonlocal':
  pairs of elements with widely differing vintages may be linked.

Smoothing out the System

At limit $dm \to 0$, time-t system is characterized by

Density $\mu_K(\tau) \geq 0$ of elements

for each vintage $\tau \in [0, t]$ and each kind $K \in \{B, G\}$

with $\int_0^t \mu_K(\tau) = m_K$.

(At the limit $dm \to 0$, system evolves deterministically.)

Smoothing out the System

At limit $dm \to 0$, time-t system is characterized by

Density $\mu_K(\tau) \geq 0$ of elements

for each vintage $\tau \in [0, t]$ and each kind $K \in \{B, G\}$

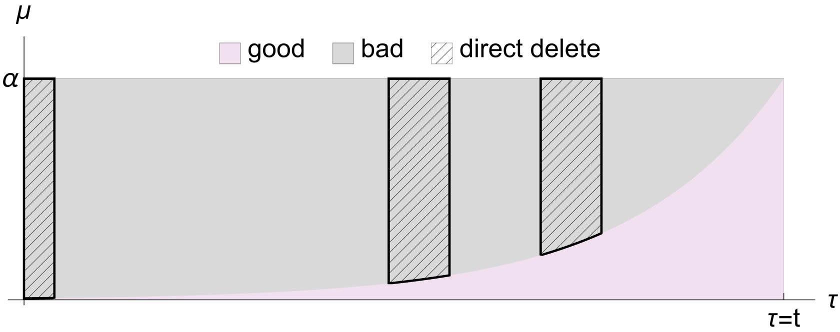
with $\int_0^t \mu_K(\tau) = m_K$.

(At the limit $dm \to 0$, system evolves deterministically.)

Preliminaries: Simple Strategies

In the optimal strategy,

- Good elements are added at maximal rate: $a(t) \equiv \alpha$.
- Only bad elements are directly deleted.



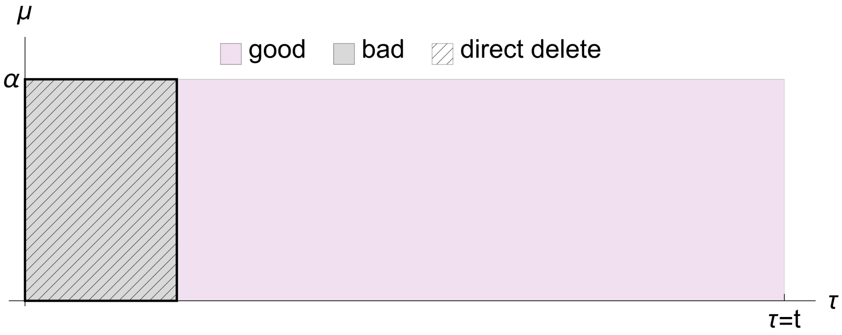
So, designer effectively chooses which vintages *of bad elements* to (directly) delete.

Preliminaries: Indirect Deletions

Recall: when some elements are directly deleted,
their dependants will immediately be *indirectly* deleted.

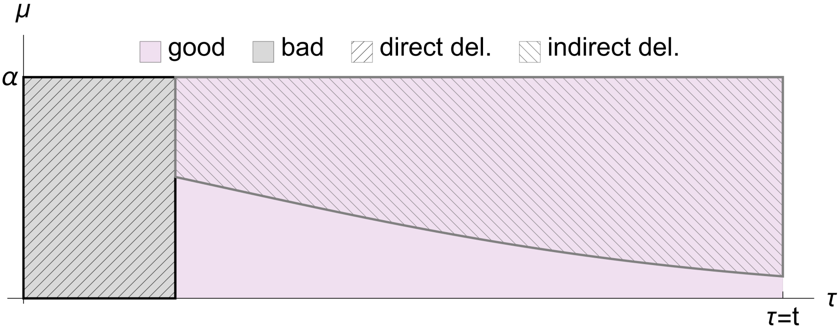
Preliminaries: Indirect Deletions

Recall: when some elements are directly deleted,
their dependants will immediately be *indirectly* deleted.



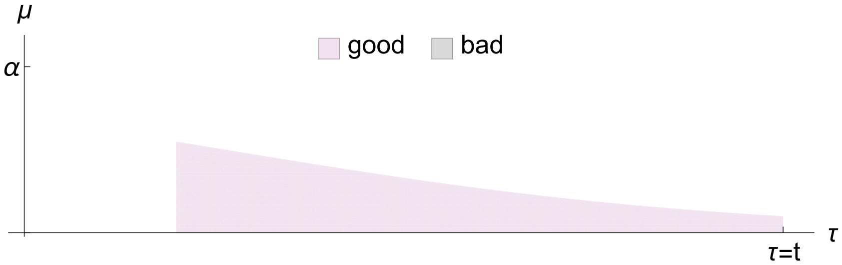
Preliminaries: Indirect Deletions

Recall: when some elements are directly deleted,
their dependants will immediately be *indirectly* deleted.



Preliminaries: Indirect Deletions

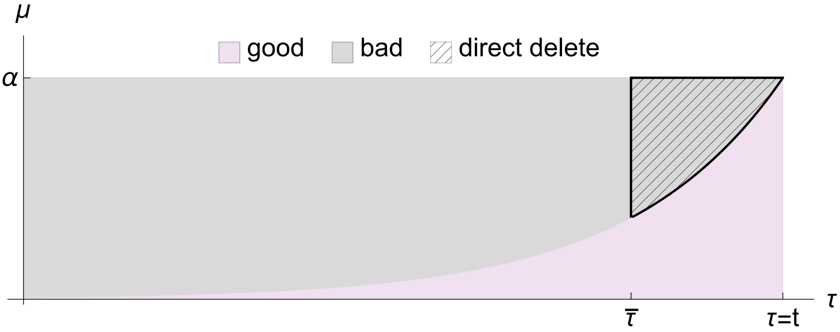
Recall: when some elements are directly deleted,
their dependants will immediately be *indirectly* deleted.



Last-In First-Out

The myopic designer optimally plays a threshold strategy $\overline{\tau}(S_t) \in [0, t]$:

at each instant t, delete all bad *elements with vintage $\geq \overline{\tau}(S_t)$.*

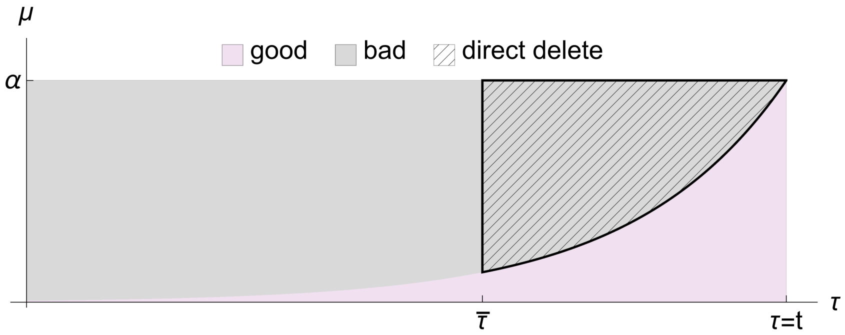


Intuition: recently added elements have fewer dependants, so are "cheap" to delete

Last-In First-Out

The myopic designer optimally plays a threshold strategy $\overline{\tau}(S_t) \in [0, t]$:

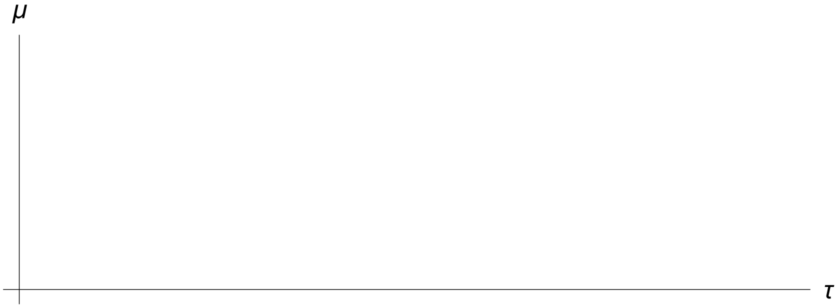
at each instant t, delete all bad *elements with vintage* $\geq \overline{\tau}(S_t)$.



Intuition: recently added elements have fewer dependants, so are "cheap" to delete

Dynamics

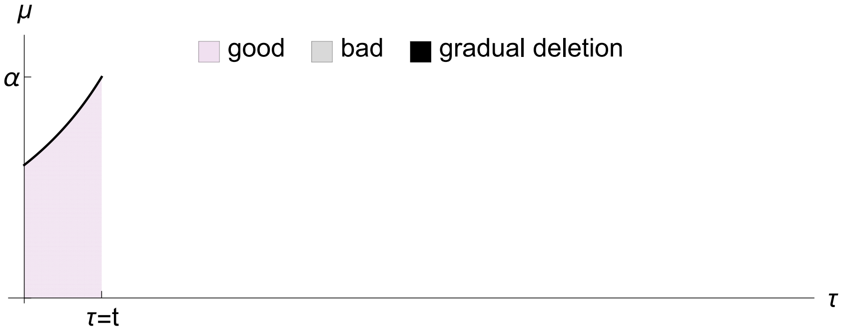
Starting from $t = 0$:



Low-Hanging Fruit

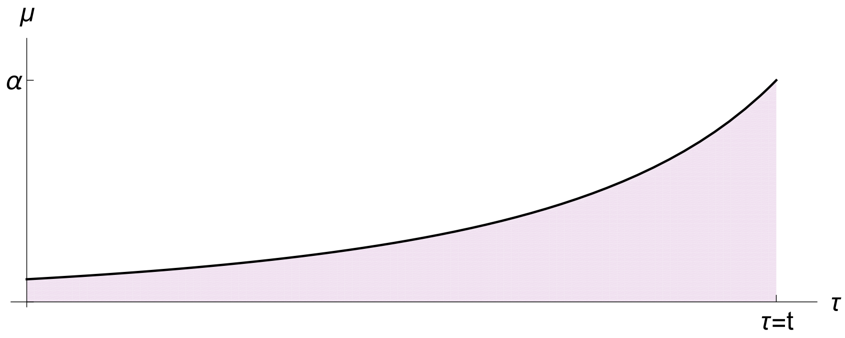
The most recent mass $\overline{m}$ of elements is constantly cleansed, where

$$\overline{m} = \log((1 + c)/\kappa).$$



Low κ: gradual reforms only

If $\kappa \leq c\lambda/\alpha$,
total mass approaches steady-state, never exceeds threshold $\overline{m}$:
system remains in gradual-cleansing mode forever.

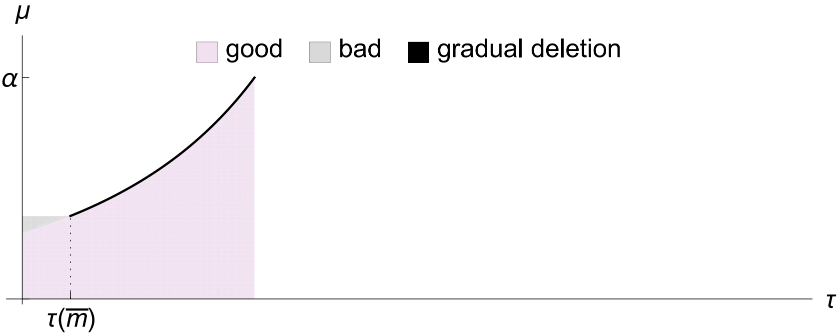


Leaking Out

If $\kappa > c\lambda/\alpha$,

Some elements get past the threshold $\overline{m}$,

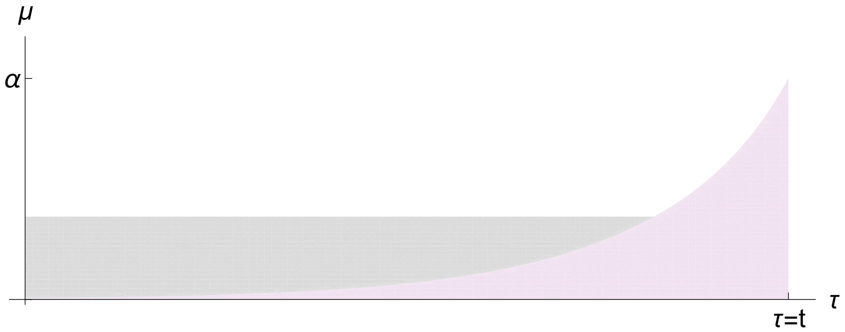
where deletions are delayed –



Big-Bang Reform

Until, after some delay,

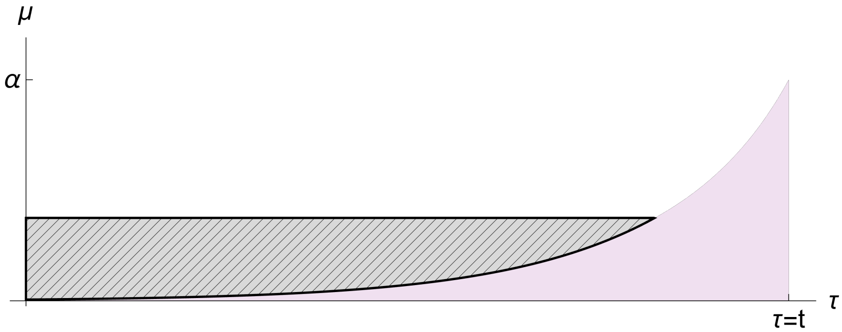
All bad elements are deleted in an instant.



Big-Bang Reform

Until, after some delay,

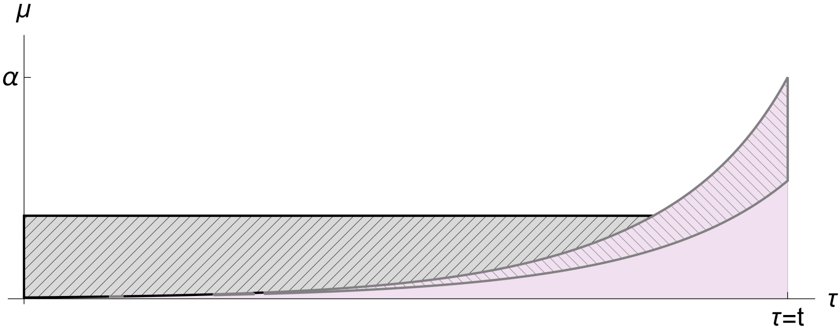
All bad elements are deleted in an instant.



Big-Bang Reform

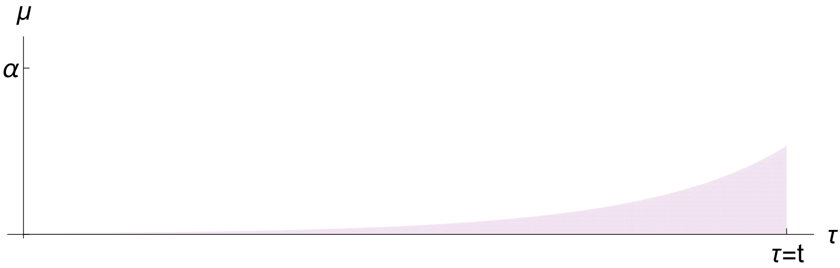
Until, after some delay,

All bad elements are deleted in an instant.



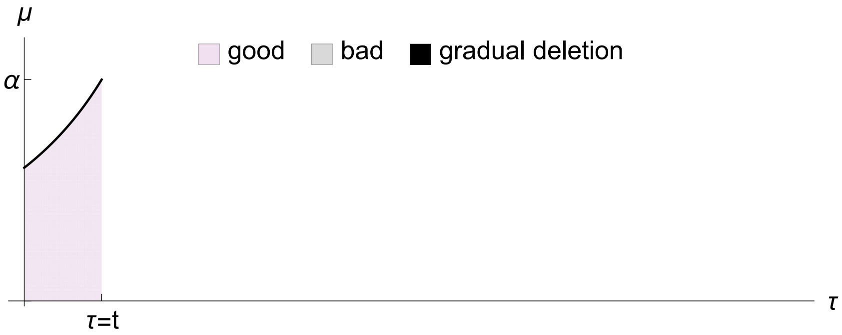
Cycles

Then, the cycle reboots.

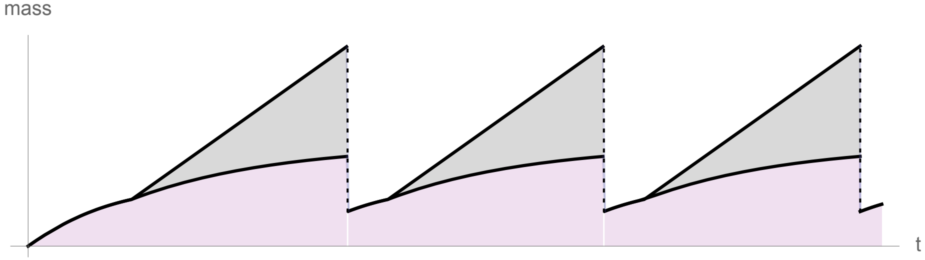


Cycles

Then, the cycle reboots.



Cycles



Optimal Dynamics

Big-Bang Reforms are optimal iff $\kappa > c\lambda/\alpha$, i.e.

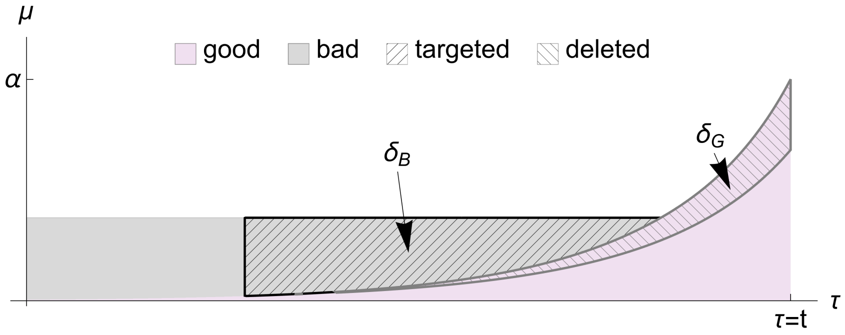
1. **entanglement** (κ) **is high**.
2. cost of retaining bad elements (c) is low.
3. productivity of designer (α) is high.
4. good elements turn bad slowly (λ is low).

Friction over the cycle.

Recall: $\pi(t) = m_G - c \cdot m_B$.

So, delete elements only when friction is low:

$$\underbrace{\delta_G/\delta_B}_{\text{friction}} \leq c.$$

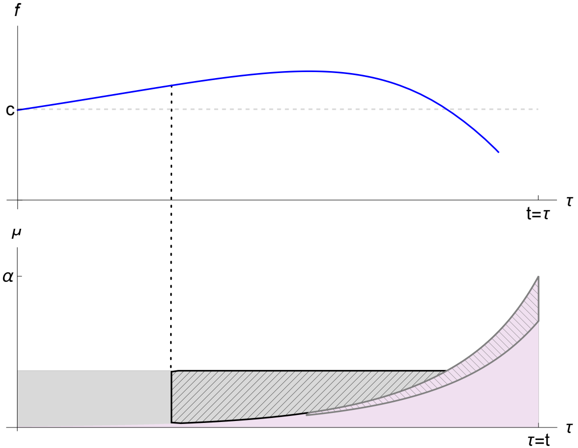


The shape of friction.

Given last-in-first-out rule,

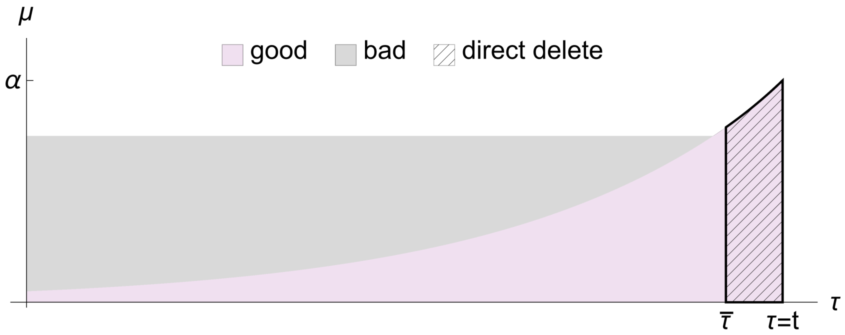
Friction is quasi-concave in scale of deletion:

Friction is low iff very few/many elements deleted.



“Excavation” effect

Suppose: at each time t, designer chooses threshold vintage $\overline{\tau}(t)$, deletes all (good + bad) elements w/ vintage $\geq \overline{\tau}(t)$.

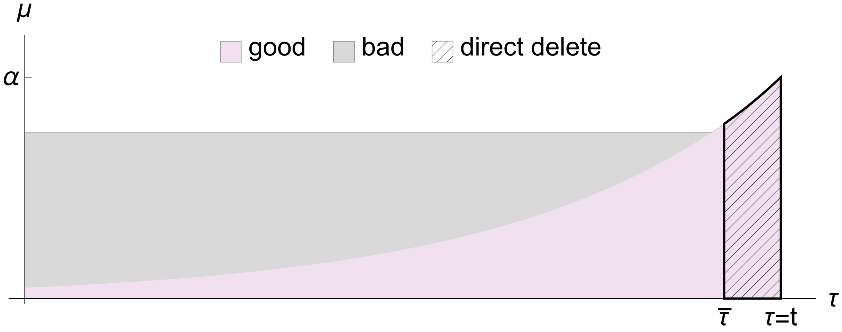


In this setting:

Marginal friction = good:bad ratio at threshold vintage

which increases with older vintages;

⇒ friction strictly decreases with scale.

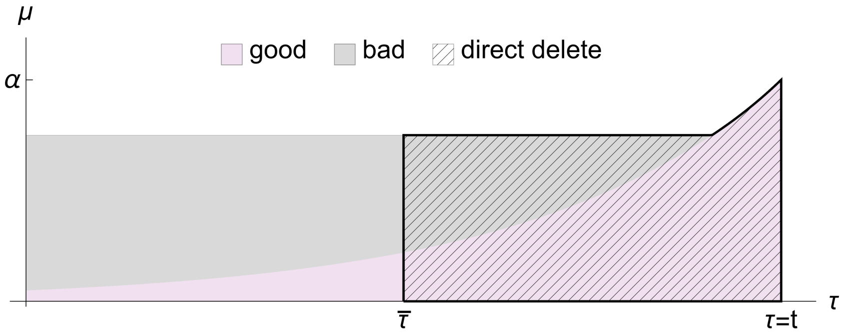


In this setting:

Marginal friction = good:bad ratio at threshold vintage

which increases with older vintages;

⇒ friction strictly decreases with scale.



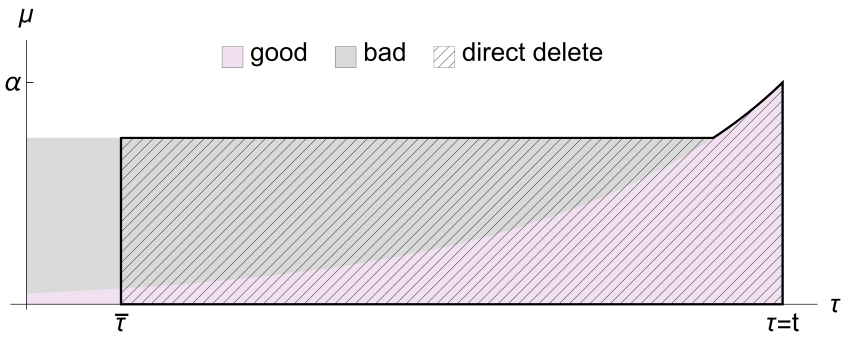
Archaeological Economics

In this setting:

Marginal friction = good:bad ratio at threshold vintage

which increases with older vintages;

$\Rightarrow$ friction strictly decreases with scale.



Untangling effect

Consider a "simple" distribution:

elements older (younger) than $\hat{\tau}$ are all bad (good).

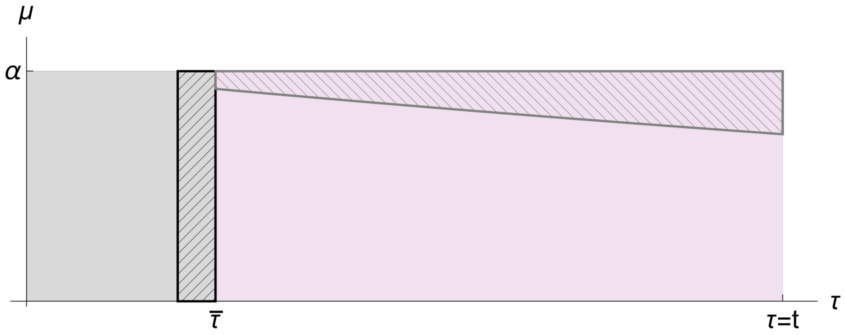
how does marginal friction change with deletion threshold $\overline{\tau} \leq \hat{\tau}$?

Untangling effect

Consider a "simple" distribution:

elements older (younger) than $\hat{\tau}$ are all bad (good).

how does marginal friction change with deletion threshold $\overline{\tau} \leq \hat{\tau}$?



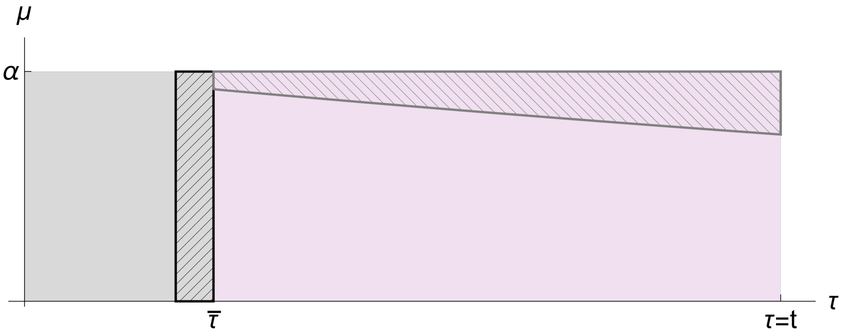
Untangling effect

As more bad elements deleted:

fewer good elements remain

$\Rightarrow$ the marginal (bad) deleted element has fewer (good) dependents

$\Rightarrow$ friction decreases with scale.



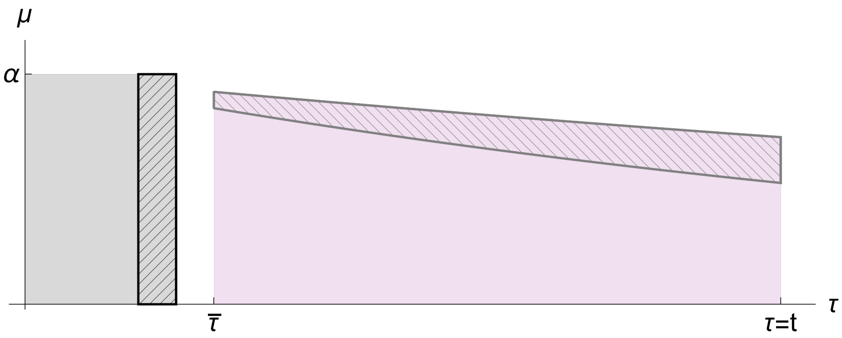
Untangling effect

As more bad elements deleted:

fewer good elements remain

⇒ the marginal (bad) deleted element has fewer (good) dependents

⇒ friction decreases with scale.



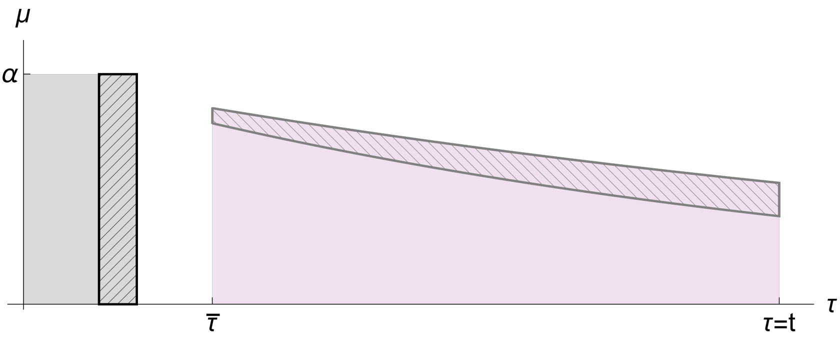
Untangling effect

As more bad elements deleted:

fewer good elements remain

⇒ the marginal (bad) deleted element has fewer (good) dependents

⇒ friction decreases with scale.



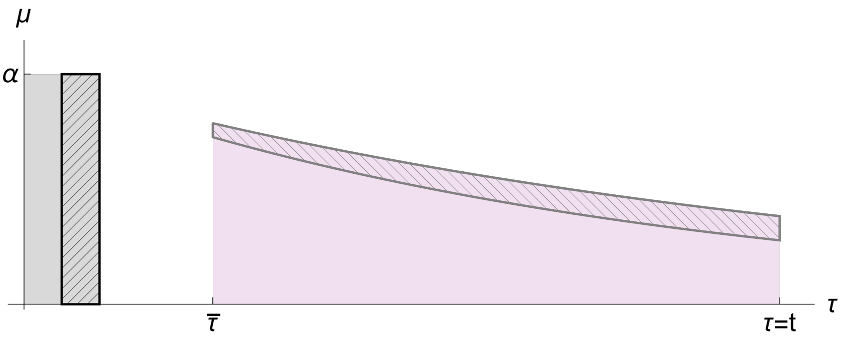
Untangling effect

As more bad elements deleted:

fewer good elements remain

⇒ the marginal (bad) deleted element has fewer (good) dependents

⇒ friction decreases with scale.



Why Big-Bang Reforms

Two forces drive big-bang reforms:

- “Excavation” effect and “untangling” effect.
  * Parallel forces: each drives big-bang reforms even even in isolation.
- With non-myopic designer, third force emerges: intertemporal tradeoffs lead, again, to big-bang reforms

Take-away points:

- Multiple “parallel” mechanisms – big-bang reforms are a relatively robust phenomenon.
- Big-bang reforms are optimal iff system is complicated (high κ).

Laws of motion

At each instant t, the Designer chooses
for each vintage τ and each kind $K \in \{B, G\}$,

$$\underbrace{\text{gradual deletions } \delta_Q(\tau, t)}_{\text{cubits / second}} \quad \text{and} \quad \underbrace{\text{jump deletions } \Delta_Q(\tau, t)}_{\text{cubits}}$$

to control the system $\left(\mu_G(\tau, t), \mu_B(\tau, t)\right)$ via

$$d\mu_G(\tau, t) = -\underbrace{\lambda\mu_G(\tau, t)dt}_{\text{decay}} - \underbrace{\beta_G(\tau, t)dt}_{\text{gradual removal}} - \underbrace{\Delta_G(\tau, t)}_{\text{jump removal}},$$

$$d\mu_B(\tau, t) = +\underbrace{\lambda\mu_B(\tau, t)dt}_{\text{decay}} - \underbrace{\beta_B(\tau, t)dt}_{\text{gradual removal}} - \underbrace{\Delta_B(\tau, t)}_{\text{jump removal}}$$

Laws of motion

At each instant t, the Designer chooses
for each vintage τ and each kind $K \in \{B, G\}$,

$$\underbrace{\text{gradual deletions } \delta_Q(\tau, t)}_{\text{cubits / second}} \quad \text{and} \quad \underbrace{\text{jump deletions } \Delta_Q(\tau, t)}_{\text{cubits}}$$

to control the system $\big(\mu_G(\tau, t), \mu_B(\tau, t)\big)$ via

$$d\mu_G(\tau, t) = -\underbrace{\lambda\mu_G(\tau, t)dt}_{\text{decay}} \quad - \quad \underbrace{\beta_G(\tau, t)dt}_{\text{gradual removal}} \quad - \quad \underbrace{\Delta_G(\tau, t)}_{\text{jump removal}},$$

$$d\mu_B(\tau, t) = +\underbrace{\lambda\mu_B(\tau, t)dt}_{\text{decay}} \quad - \quad \underbrace{\beta_B(\tau, t)dt}_{\text{gradual removal}} \quad - \quad \underbrace{\Delta_B(\tau, t)}_{\text{jump removal}}$$

Laws of motion

At each instant t, the Designer chooses
for each vintage τ and each kind $K \in \{B, G\}$,

$$\underbrace{\text{gradual deletions } \delta_Q(\tau, t)}_{\text{cubits / second}} \quad \text{and} \quad \underbrace{\text{jump deletions } \Delta_Q(\tau, t)}_{\text{cubits}}$$

subject to frictional constraints: for each $Q \in \{B, G\}$,

$$\underbrace{\frac{\beta_Q(\tau, t)}{\mu_Q(t)} \geq \kappa \cdot \frac{\partial}{\partial t} D(\tau, t)}_{\text{rate constraint}} \quad \text{and} \quad \underbrace{\frac{\Delta_Q(\tau, t)}{\mu_Q(t)} \geq f(\kappa \Delta D(\tau, t))}_{\text{jump constraint}}$$

where $f(x) = 1 - e^{-x}$ and

$$D(\tau, t) = \underbrace{\sum_{Q \in \{B,G\}} \int_0^{\tau} \left(\int_{\tau}^{t} \beta_Q(\tau, t') dt' + \sum_{t' \in [\tau, t)} \Delta_Q(\tau, t') \right) d\tau'}_{\text{mass of deleted elements older than } \tau} .$$