Continuous versus Abrupt Reorganizations

Anton Kolotilin and Hongyi Li (UNSW)

March 2021

Entangled systems

- Complicated systems: accumulate design elements over time

Entangled systems

- Complicated systems: accumulate design elements over time

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

  Fixes create problems elsewhere, necessitate further fixes, etc.

- Change may be delayed → inefficiencies persist and accumulate

* Rich terminology for inefficiencies: *cruft*, *kludges*, *technical debt*, etc

- Examples:
  - MS-DOS → Windows → Windows 95 ...
  - 1960s contracting processes at US Defense Dept
  - Public policy: tax, healthcare

This paper:

When complicated, entangled systems face continuous pressure to change,

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

Abrupt change occurs in various settings:

- radical re-engineering in organizations
- big-bang reforms of public policy
- periodic refactoring in software development

Abrupt change is often associated with technical debt ↓, functionality ↓.

- Occurs with disruptive new products: e.g., iPhone.

This paper:

When complicated, entangled systems face continuous pressure to change,

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

Abrupt change occurs in various settings:

- radical re-engineering in organizations
- big-bang reforms of public policy
- periodic refactoring in software development

Abrupt change is often associated with technical debt ↓, functionality ↓.

- Occurs with disruptive new products: e.g., iPhone.

This paper:

When complicated, entangled systems face continuous pressure to change,

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

Abrupt change occurs in various settings:

- radical re-engineering in organizations
- big-bang reforms of public policy
- periodic refactoring in software development

Abrupt change is often associated with technical debt $\downarrow$, functionality $\downarrow$.

- Occurs with disruptive new products: e.g., iPhone.

Product design example

2011: Apple releases Final Cut Pro X (to replace Final Cut Pro 7).

> *Many users have expressed their frustration with a litany of missing features in Final Cut Pro X. To begin with, there's no support for output to tape … There's no support for EDL or XML export … There's no …*
>
> *… because FCPX uses a completely re-architected underlying media handling and editing paradigm, it can't …*
>
> *— arstechnica.com*

Stylised model of entangled systems

System Design:

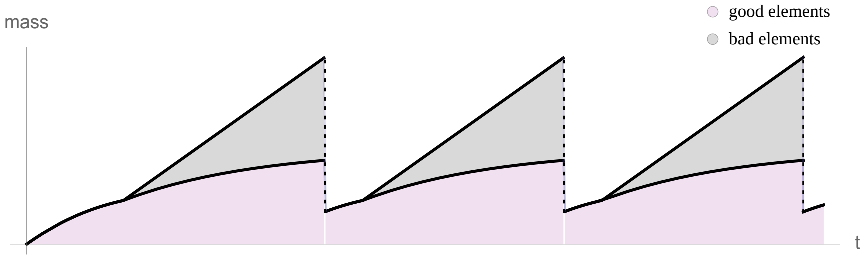
- system with continuum of elements
- designer can add and delete elements
- ‘good’ elements randomly turn ‘bad’ over time

Entanglement:

- *exogenous* directed network structure over elements
- element deleted → direct and indirect children also deleted

Main result: a preview

The (myopic) designer's optimal strategy involves episodic 'abrupt' reorganizations (iff network is sufficiently dense).



Preview of results

Why abrupt reorganization?

1. Driven by *disentanglement* effect: large one-time reorganization less disruptive than continuous, incremental reorganization.
2. Also driven by intertemporal tradeoff: patient designer will optimally cycle between 'clean' and 'dirty' designs.

Lit review

- Kludges
  - Ely 2011; Ellison and Holden 2013; Kawai, Lang and Li 2018
- Sandpile / traffic-jam models
  - Bak, Chen, Scheinkman and Woodford 1993
- Rugged landscapes
  - Kauffman 1989; Milgrom and Roberts 1992; Levinthal 1997

Road map

The Model

- Time is continuous, $t \geq 0$.
- System S_t is a continuum of infinitesimal, equal-weighted elements.
- Good elements independently turn bad with constant decay rate λ.
- Designer's flow payoff depends on masses of good vs. bad elements:

$$\pi_t = m_G(t) - c\, m_B(t).$$

The designer

At each instant t, the designer may:

- Add good elements at bounded rate $a_t \leq \alpha$ (mass per unit time).
- Choose *target set* $D(t) \subseteq S_t$ of elements for deletion.

Elements are entangled:

- *Exogenous* network of directed links between elements.
- Element x targeted → all children, grandchildren, ... also deleted.
- ∗ Notation: set D targeted → *collateral set* $C(D, S_t) \supset D$ deleted.

Continuous implementation of network formation/deletion process

- Similar to mean-field approximation of Barabási and Albert 1999.

The designer

At each instant t, the designer may:

- Add good elements at bounded rate $a_t \leq \alpha$ (mass per unit time).
- Choose *target set* $D(t) \subseteq S_t$ of elements for deletion.

Elements are entangled:

- *Exogenous* network of directed links between elements.
- Element x targeted → all children, grandchildren, ... also deleted.
- ∗ Notation: set D targeted → *collateral set* $C(D, S_t) \supset D$ deleted.

Continuous implementation of network formation/deletion process

- Similar to mean-field approximation of Barabási and Albert 1999.

The designer

At each instant t, the designer may:

- Add good elements at bounded rate $a_t \leq \alpha$ (mass per unit time).
- Choose *target set* $D(t) \subseteq S_t$ of elements for deletion.

Elements are entangled:

- *Exogenous* network of directed links between elements.
- Element x targeted → all children, grandchildren, ... also deleted.
- ∗ Notation: set D targeted → *collateral set* $C(D, S_t) \supset D$ deleted.

Continuous implementation of network formation/deletion process

- Similar to mean-field approximation of Barabási and Albert 1999.

Network: preview of key features

- Homogenous, ‘detail-free’ network;
  so, ‘big-picture’ view of system is sufficient.

- Entanglement is ‘limited’:
  each (infinitesimal) element has finite number of (infinitesimal) children + grandchildren + ...

- Entanglement is ‘non-localised’:
  as system grows, each element accumulates more links.

Network formation

Links:

- Each new element links to each existing element with probability $\kappa \cdot dm$, where dm is infinitesimal element mass;

  ($\kappa > 0$ parametrizes *entanglement*.)

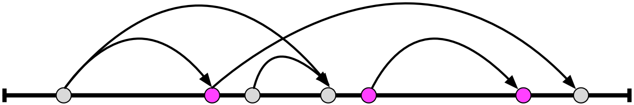
* So, each new element links to $\kappa \cdot m$ other elements (in expectation).

Directions:

- Elements are ranked [0, 1]. Links point towards lower-ranked elements.

* So, network is acyclic – which ensures 'limited entanglement'.

- Each new element is uniformly randomly assigned a rank.



Network formation

Links:

- Each new element links to each existing element with probability $\kappa \cdot dm$, where dm is infinitesimal element mass;

  ($\kappa > 0$ parametrizes *entanglement*.)

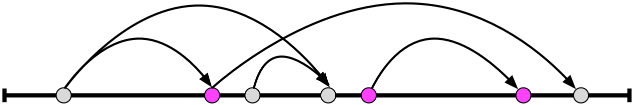
* So, each new element links to $\kappa \cdot m$ other elements (in expectation).

Directions:

- Elements are ranked $[0, 1]$. Links point towards lower-ranked elements.

* So, network is acyclic – which ensures 'limited entanglement'.

- Each new element is uniformly randomly assigned a rank.



What does the designer know?

The Designer:

- Observes the type (good or bad) of each element in S_t.
- Understands the network formation process,
  but *doesn't observe time-t network*.

* Upon deleting x, immediately observes deletion of x's descendants.

Continuous ingredients

- continuous time
- continuous space (continuum of elements)
- continuous pressure to reorganize (decay process)
- ‘small’ network frictions

$\Rrightarrow$ continuous / discontinuous reorganizations?

Simplifying the problem

Designer's time-t problem:

Given system $S(t)$ and (beliefs about) network $E(t)$,
Choose growth rate g_t and deletion set $D(t)$

to maximize

$$\int_0^\infty e^{-rt} \underbrace{\big(m_G(t) - c\, m_B(t)\big)}_{\text{flow payoff}} dt.$$

Simplifying the problem

Myopic Designer's time-t problem:

Given system $S(t)$ and (beliefs about) network $E(t)$,
Choose growth rate g_t and deletion set $D(t)$

to maximize

$$\frac{d}{dt}\underbrace{\left(m_G(t) - c\, m_B(t)\right)}_{\text{flow payoff}}.$$

The details don't matter

Following any history, the designer believes that:

- Links are uniformly randomly distributed across element-pairs.
- Each element's rank is uniformly randomly distributed.

$\Rightarrow$ So, all good elements look alike; all bad elements look alike.

In the optimal strategy,

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

$\Rightarrow$ So, designer simply chooses *how many* bad elements to target.

The details don't matter

Following any history, the designer believes that:

- Links are uniformly randomly distributed across element-pairs.
- Each element's rank is uniformly randomly distributed.

$\Rightarrow$ So, all good elements look alike; all bad elements look alike.

In the optimal strategy,

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

$\Rightarrow$ So, designer simply chooses *how many* bad elements to target.

The coefficient of Friction

Given:

- system S and network E
- target set $D \subset S$ of bad elements
- collateral set $C(D, S)$ with mass Δ_B of bad elements

The (coefficient of) *Friction*

$$F(\underbrace{m}_{\text{mass}}, \underbrace{m_G/m_B}_{\text{ratio}}, \underbrace{\Delta_B}_{\text{scale}}) = \frac{\Delta_G}{\Delta_B}$$

is the ratio of good to bad elements in $C(D, S)$.

Laws of motion

At time t, the Designer chooses

flow rates of deletion $\beta_G(t), \beta_B(t)$

discrete masses of deletion $\Delta_G(t), \Delta_B(t)$

to control the system $\left(m_G(t), m_B(t)\right)$ via

$$dm_G(t) = \underbrace{\alpha dt}_{growth} - \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_G(t)dt + \Delta_G(t)\right)}_{removal},$$

$$dm_B(t) = \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_B(t)dt + \Delta_B(t)\right)}_{removal}$$

subject to Frictional constraints

$$\frac{\Delta_G(t)}{\Delta_B(t)} = F\left(m(t), m_G(t)/m_B(t), \Delta_B(t)\right),$$

$$\frac{\beta_G(t)}{\beta_B(t)} = F\left(m(t), m_G(t)/m_B(t), 0\right).$$

Laws of motion

At time t, the Designer chooses

flow rates of deletion $\beta_G(t), \beta_B(t)$

discrete masses of deletion $\Delta_G(t), \Delta_B(t)$

to control the system $\left(m_G(t), m_B(t)\right)$ via

$$dm_G(t) = \underbrace{\alpha dt}_{growth} - \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_G(t)dt + \Delta_G(t)\right)}_{removal},$$

$$dm_B(t) = \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_B(t)dt + \Delta_B(t)\right)}_{removal}$$

subject to Frictional constraints

$$\frac{\Delta_G(t)}{\Delta_B(t)} = F\left(m(t), m_G(t)/m_B(t), \Delta_B(t)\right),$$

$$\frac{\beta_G(t)}{\beta_B(t)} = F\left(m(t), m_G(t)/m_B(t), 0\right).$$

Laws of motion

At time t, the Designer chooses

flow rates of deletion $\beta_G(t), \beta_B(t)$

discrete masses of deletion $\Delta_G(t), \Delta_B(t)$

to control the system $\left(m_G(t), m_B(t)\right)$ via

$$dm_G(t) = \underbrace{\alpha dt}_{growth} - \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_G(t)dt + \Delta_G(t)\right)}_{removal},$$

$$dm_B(t) = \underbrace{\lambda m_G(t)dt}_{decay} - \underbrace{\left(\beta_B(t)dt + \Delta_B(t)\right)}_{removal}$$

subject to Frictional constraints

$$\frac{\Delta_G(t)}{\Delta_B(t)} = F\left(m(t), m_G(t)/m_B(t), \Delta_B(t)\right),$$

$$\frac{\beta_G(t)}{\beta_B(t)} = F\left(m(t), m_G(t)/m_B(t), 0\right).$$

Friction: key property

Friction F is:

1. increasing in mass m
2. increasing in good/bad ratio m_G/m_B
3. increasing in entanglement κ
4. decreasing in scale of reorganization Δ_B

Friction: key property

Friction F is:

1. increasing in mass m
2. increasing in good/bad ratio m_G/m_B
3. increasing in entanglement κ
4. **decreasing in scale of reorganization** Δ_B

Friction: some intuition

C comprises (i) target set D and (ii) descendants D' of targets:

$$\underbrace{C}_{\text{collateral set}} = \underbrace{D}_{\text{bad elements only}} \bigcup \underbrace{D'}_{\text{random draws from } S}$$

Friction F is good/bad ratio of $C = D \cup D'$:

1. As mass m increases, D' increases in size $\Rightarrow F$ increases.
2. As ratio $\frac{m_G}{m_B}$ increases, more good elements in D' $\Rightarrow F$ increases.
3. As entanglement κ increases, D' increases in size $\Rightarrow F$ increases.
4. As scale Δ_B increases ... ?

Friction: some intuition

C comprises (i) target set D and (ii) descendants D' of targets:

$$\underbrace{C}_{\text{collateral set}} = \underbrace{D}_{\text{bad elements only}} \bigcup \underbrace{D'}_{\text{random draws from } S}$$

Friction F is good/bad ratio of $C = D \cup D'$:

1. As mass m increases, D' increases in size $\Rightarrow F$ increases.
2. As ratio $\frac{m_G}{m_B}$ increases, more good elements in D' $\Rightarrow F$ increases.
3. As entanglement κ increases, D' increases in size $\Rightarrow F$ increases.
4. **As scale Δ_B increases … ?**

How does friction F change with scale Δ_B?

As more elements deleted ($\Delta_B \uparrow$), two conflicting effects:

decontamination vs. disentanglement

(1/2): Decontamination Effect

As more elements deleted ($\varDelta_B \uparrow$),
decontamination effect *increases* friction:

$$\varDelta_B \uparrow \quad \Rightarrow \quad \underbrace{\frac{m_G - \varDelta_G}{m_B - \varDelta_B} \uparrow}_{\substack{\text{remaining elements } S \setminus C \\ \text{become `cleaner'}}} \quad \Rightarrow \underbrace{F = \frac{\varDelta_G}{\varDelta_B} \text{ tends to } \uparrow}_{\substack{D' \text{ and thus } C = D \cup D' \\ \text{also becomes `cleaner'}}}$$

C : collateral set

D : target set

D' : descendants

(2/2): Disentanglement Effect

As more elements deleted ($\Delta_B \uparrow$),
disentanglement effect *reduces* friction:

$$\Delta_B \uparrow \quad \Rightarrow \quad \underbrace{|S \setminus C| \downarrow}_{\substack{\text{remaining system } S \setminus C \\ \text{shrinks}}} \quad \Rightarrow \quad \underbrace{\frac{|D'|}{|D|}}_{\text{less `collateral damage'}} \downarrow \quad \Rightarrow \quad \underbrace{F = \frac{\Delta_G}{\Delta_B} \text{ tends to } \downarrow}_{C \text{ becomes `dirtier'}}$$

C : collateral set

D : target set

D' : descendants

How does friction F change with scale Δ_B?

$$\underbrace{\text{Disentanglement effect}}_{F \text{ tends to } \downarrow \text{ as } \Delta_B \uparrow} \quad \text{dominates} \quad \underbrace{\text{Decontamination effect}}_{F \text{ tends to } \uparrow \text{ as } \Delta_B \uparrow}$$

$$\Downarrow$$

Friction F decreases as scale Δ_B increases

Myopic designer performs full cleansing

Consider a myopic designer: i.e.,
maximizes $\frac{d}{dt}\left(m_G(t) - cm_B(t)\right)$.

Friction F decreases as scale Δ_B increases

$\Downarrow$

With myopic designer, at any instant,
whenever *any* bad elements are removed, *all* bad elements are removed.

Myopic designer performs full cleansing

Consider a myopic designer: i.e.,
maximizes $\frac{d}{dt}\left(m_G(t) - c m_B(t)\right)$.

Friction F decreases as scale Δ_B increases

$\Downarrow$

With myopic designer, at any instant,
whenever *any* bad elements are removed, *all* bad elements are removed.

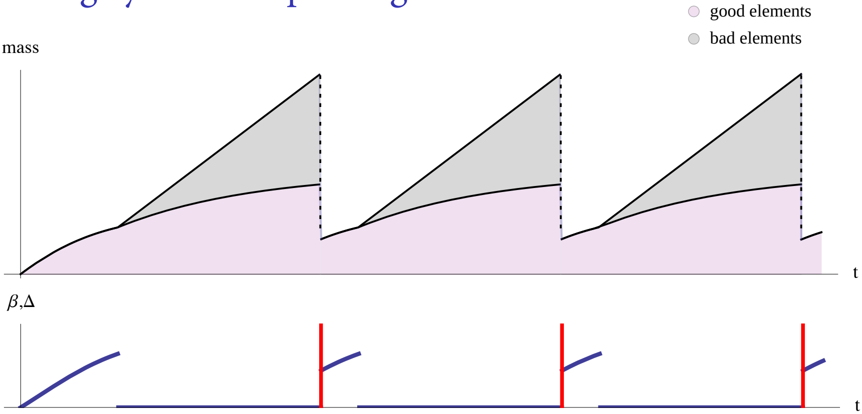
Optimal modes for a myopic designer

The myopic designer's optimal strategy is unique, and takes one of two forms:

1. a 'cleansing-cycles' mode.
2. a 'constant-cleansing' mode.

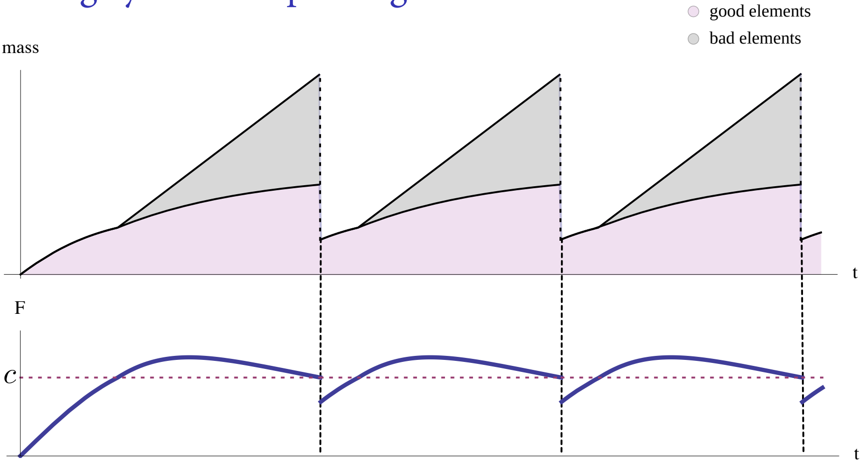
Cleansing cycles are optimal iff entanglement κ is sufficiently high

Cleansing cycles: abrupt reorganizations



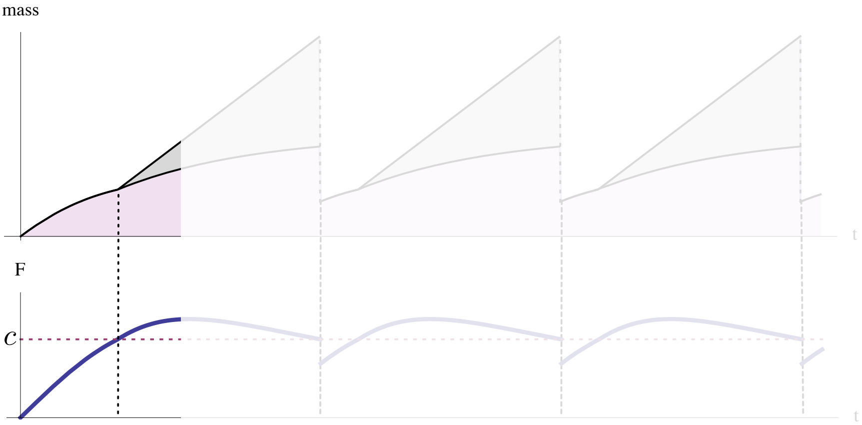
Reorganization occurs episodically:
abrupt cleansing → continuous cleansing → no cleansing → abrupt ...

Cleansing cycles: abrupt reorganizations



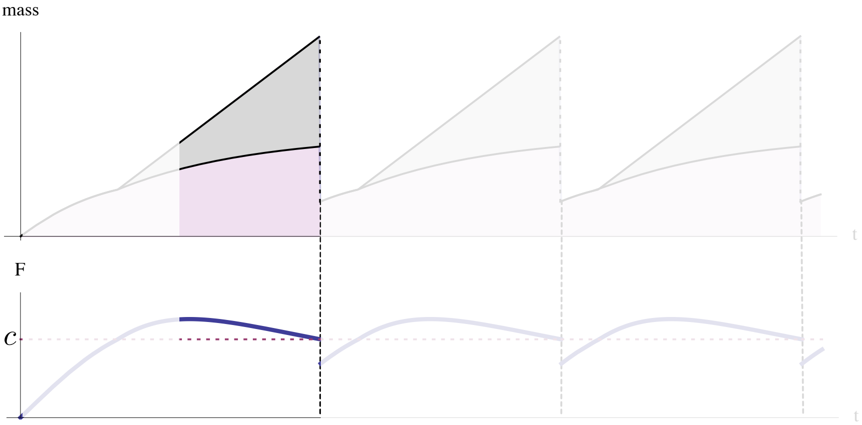
Reorganization occurs whenever friction is low: $F < c$.

Cleansing cycles: a walkthrough



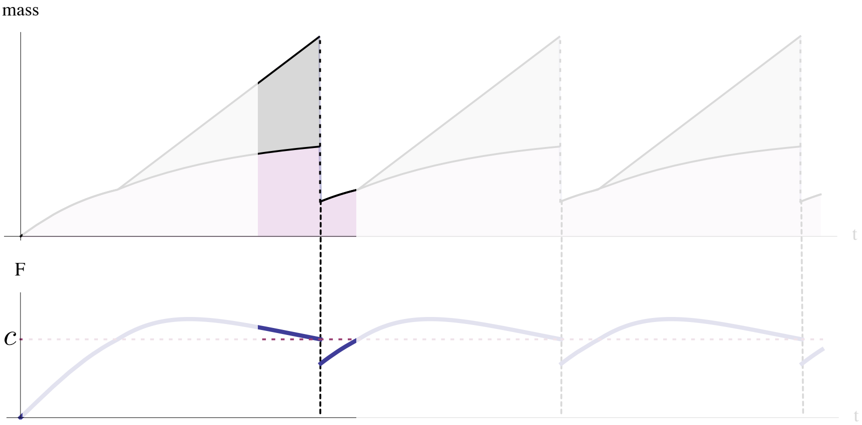
Initially, friction F increases as system grows.
When $F > c$, designer stops cleansing;
$\Rightarrow$ contamination begins (m_G/m_B decreases).

Cleansing cycles: a walkthrough



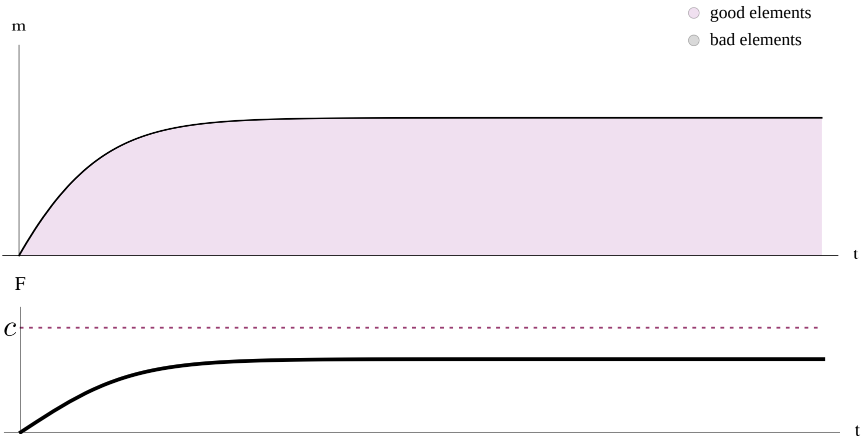
Eventually, 'contamination effect' starts to dominate;
$\Rightarrow$ friction F starts decreasing.

Cleansing cycles: a walkthrough



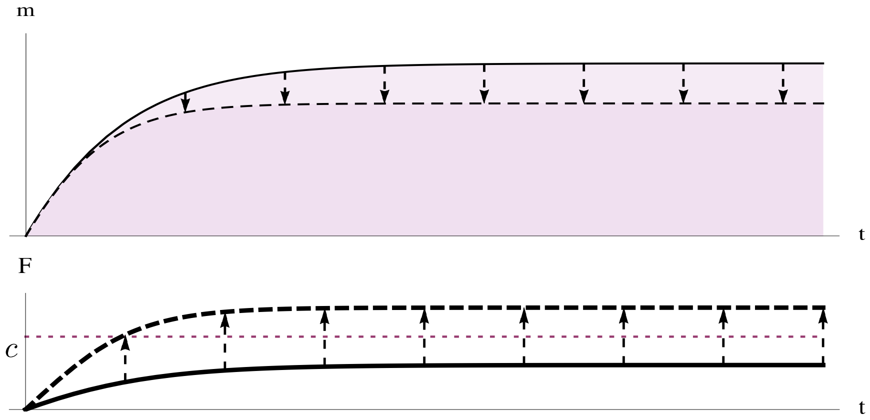
When friction $F = c$, cleansing event is triggered.
Abrupt 'full cleansing' due to disentanglement effect.

Steady-state mode



In steady-state mode: cleansing occurs constantly.

Entangled systems and abrupt reorganizations



Increase in entanglement $\kappa \rightarrow$ steady-state mode is inefficient
$\rightarrow$ cleansing cycles are optimal

Comparative statics

Cleansing Cycles are optimal iff:

1. **entanglement** (κ) **is high**.
2. burden imposed by bad elements (c) is low.
3. productivity/innovativeness of designer (α) is high.
4. rate of decay (λ) is low.

Patient designer

Suppose designer is non-myopic: maximizes

$$\int_0^\infty e^{-rt} \underbrace{\left(m_G(t) - c\, m_B(t)\right)}_{\text{flow payoff}} dt \text{ with } r < \infty.$$

Simplifying assumption: elements are densely entangled, i.e., $\kappa \to \infty$.

Dense entanglement → constant returns to scale

Consider a small target set D.

Recall: C comprises (i) target set D and (ii) descendants D' of targets,

$$\underbrace{C}_{\text{collateral set}} = \underbrace{D}_{\text{bad elements only}} \bigcup \underbrace{D'}_{\text{random draws from } S}$$

Given $\kappa \to \infty$,

- D' (vastly) outnumbers D, so ...
- C is approx. random sample of $S = (m_G, m_B)$.

At the $\kappa \to \infty$ limit, $F(m, m_G/m_B, \Delta_B) \equiv m_G/m_B$.

⇒ Friction is constant in scale Δ_B; i.e., 'disentanglement' effect is absent.

Dense entanglement → constant returns to scale

Consider a small target set D.

Recall: C comprises (i) target set D and (ii) descendants D' of targets,

$$\underbrace{C}_{\text{collateral set}} = \underbrace{D}_{\text{bad elements only}} \bigcup \underbrace{D'}_{\text{random draws from } S}$$

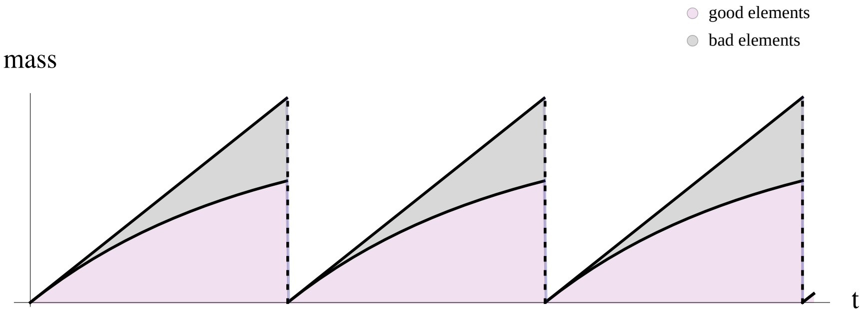
Given $\kappa \to \infty$,

- D' (vastly) outnumbers D, so ...
- C is approx. random sample of $S = (m_G, m_B)$.

At the $\kappa \to \infty$ limit, $F(m, m_G/m_B, \Delta_B) \equiv m_G/m_B$.

$\Rightarrow$ Friction is constant in scale Δ_B; i.e., 'disentanglement' effect is absent.

Dense entanglement → cleansing cycles



Proposition

With non-myopic designer and dense network:
Cleansing cycles are strictly optimal.

Why cleansing cycles (even without disentanglement)?

Intuition –

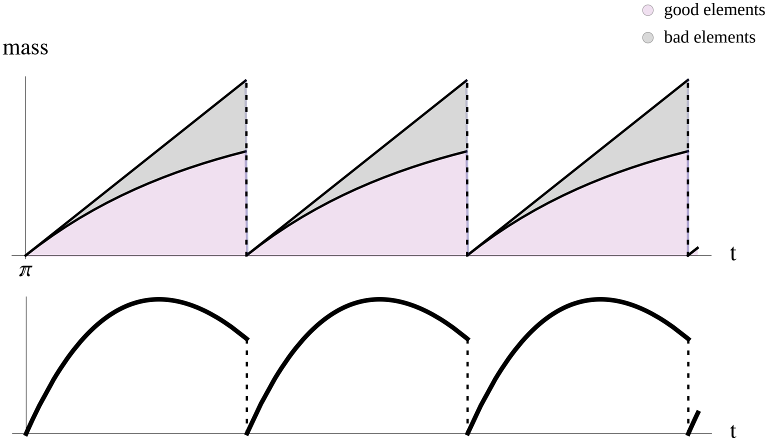
Designer has two conflicting objectives:

- Maintain ‘productive’ system (high m_G, low m_B) over time (on average).
- Reorganize cheaply, i.e., when friction $F = \frac{m_G}{m_B}$ is low.

How to reconcile these objectives?

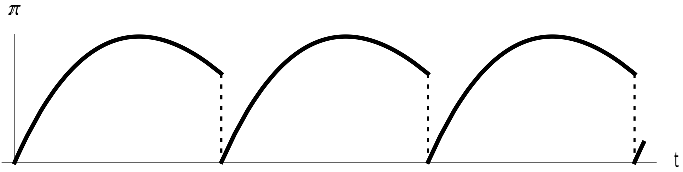
- Allow system to cycle between ‘clean’ (high F) and ‘dirty’ (low F).
- Concentrate reorganizations on times with low F.

How does reorganization affect productivity?



Reorganization: discontinuous drop in payoffs
(foreshadowed by gradual decline).

How does productivity evolve over the cycle?



For a non-myopic designer,

- Drop in payoffs from reorganization is worthwhile:
  * Following reorganization, payoff decline is reversed.

- Designer delays past peak payoff to reorganize:
  * Maximizes time spent at peak / near-peak payoffs.

Recap

1. Abrupt reorganizations iff system is highly entangled.
2. Abrupt reorganization → functionality ↓, technical debt ↓.
   - Introduction of disruptive new products: e.g., iPhone.
3. Abrupt reorganization → discrete drop in performance.
   - But performance improves rapidly afterward.
4. Reorganizations are not triggered by discrete technological shock.
   - e.g., incremental improvements in battery, touchscreen, CPU, storage tech → iPhone.

What's next

Immediate:

- Extend results to general case: $\kappa < \infty$ and $r < \infty$.

On the agenda:

- How to model different (richer) interdependency structures?
- How to endogenize entanglement? (e.g., modularization.)
- How to introduce competition / strategic interactions?