Congestion control methods: A, B, C and D

`Method A`:

- if $Q(t) = Q^*$ then $\lambda(t+1) \leftarrow \lambda(t)$
- if $Q(t) < Q^*$ then $\lambda(t+1) \leftarrow \lambda(t) + a$
- if $Q(t) > Q^*$ then $\lambda(t+1) \leftarrow \lambda(t) - a$

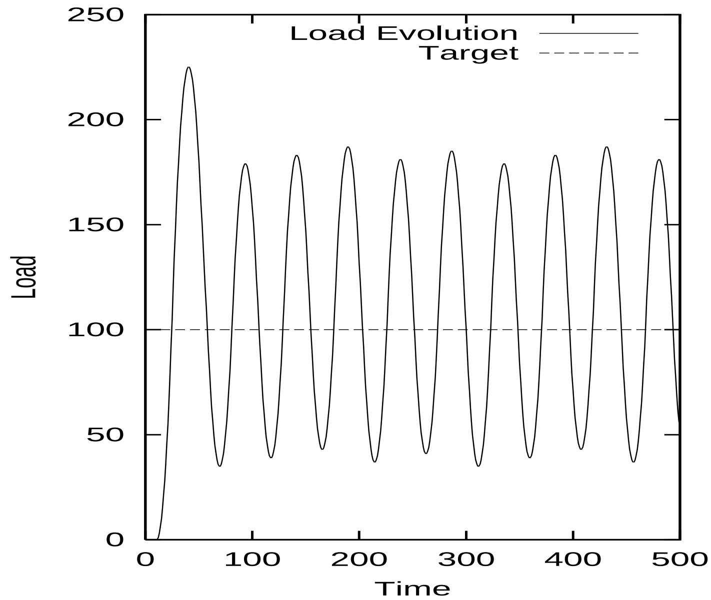
where $a > 0$ is a fixed parameter

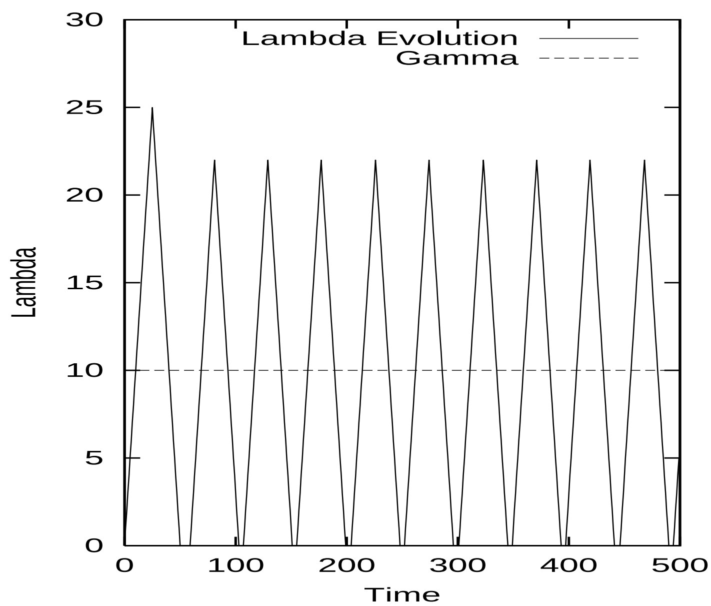
$\longrightarrow$ linear increase and linear decrease

Question: does it work?

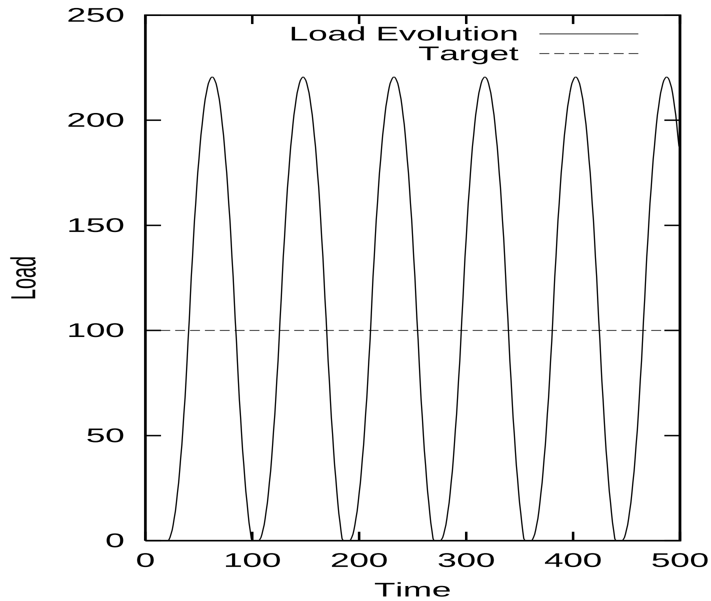
Example:

- $Q^* = 100$
- $\gamma = 10$
- $Q(0) = 0$
- $\lambda(0) = 0$
- $a = 1$

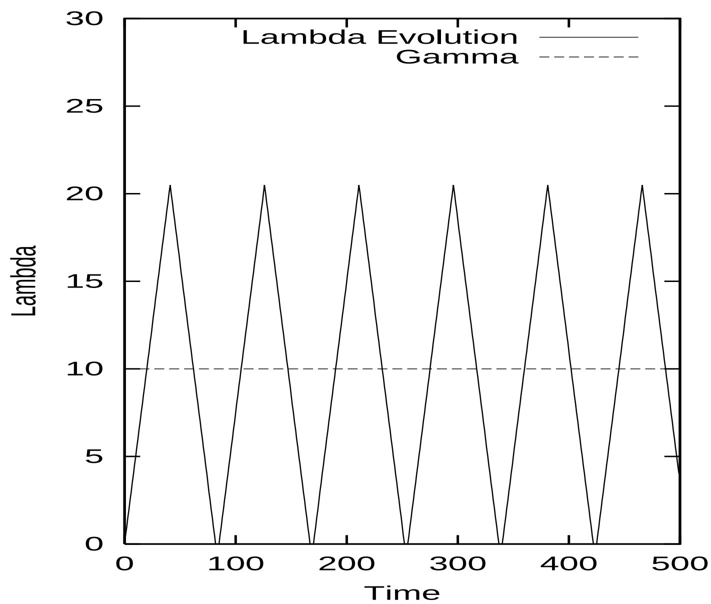

Load Evolution
Target
Load
Time
250
200
150
100
50
0
0
100
200
300
400
500


Lambda Evolution
Gamma
Lambda
Time
30
25
20
15
10
5
0
0
100
200
300
400
500

With $a = 0.5$:

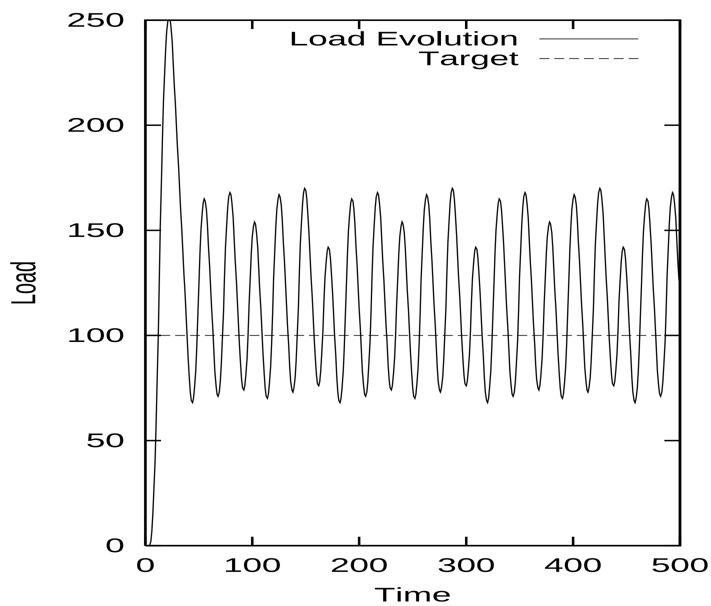


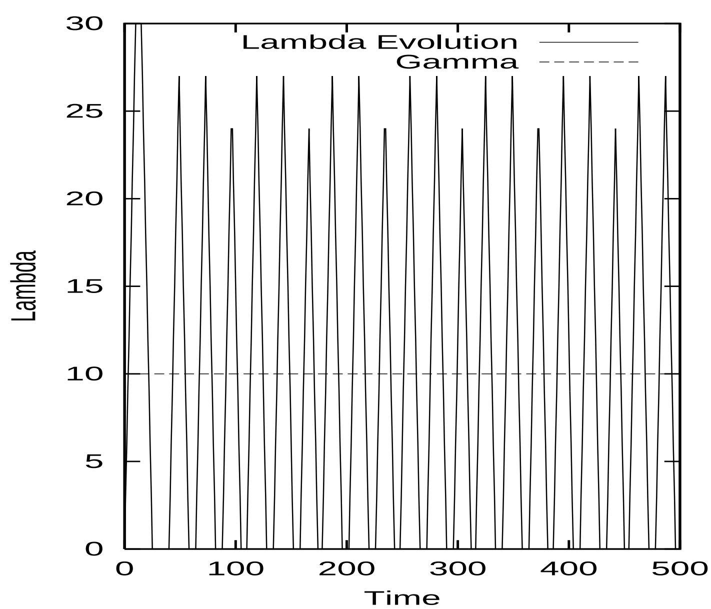
Load Evolution
Target
Load
Time
250
200
150
100
50
0
0
100
200
300
400
500



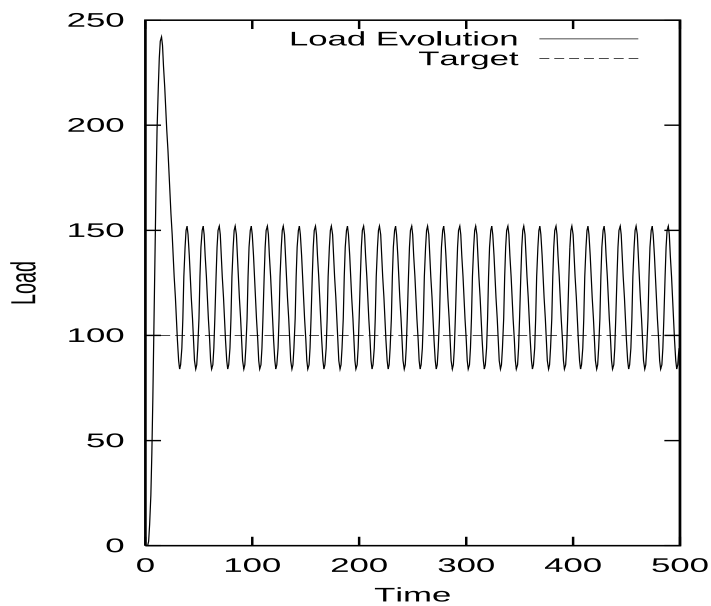
Lambda Evolution
Gamma
Lambda
Time
30
25
20
15
10
5
0
0
100
200
300
400
500

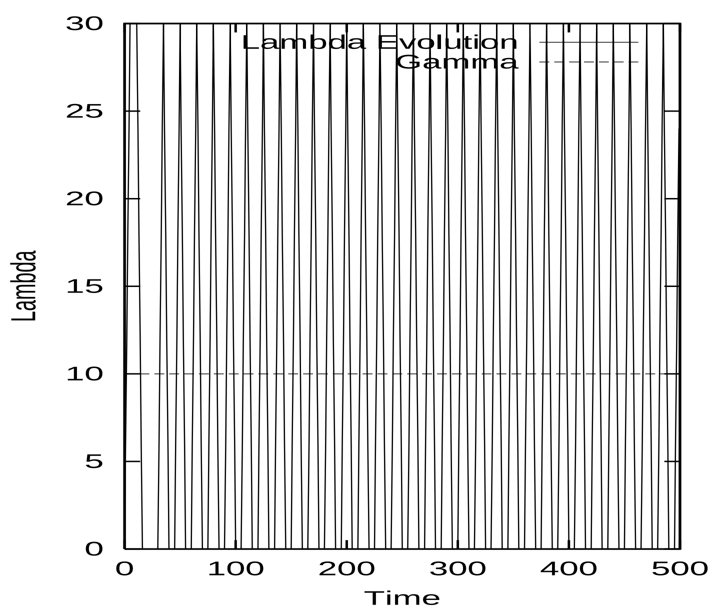
With $a = 3$:


Load Evolution
Target
Load
Time
250
200
150
100
50
0
0
100
200
300
400
500


Lambda Evolution
Gamma
Lambda
Time
30
25
20
15
10
5
0
0
100
200
300
400
500

With $a = 6$:


Load Evolution
Target
Load
Time
250
200
150
100
50
0
0
100
200
300
400
500


Lambda Evolution
Gamma
Lambda
Time
30
25
20
15
10
5
0
0
100
200
300
400
500

Remarks:

- Method A isn't that great no matter what a value is used

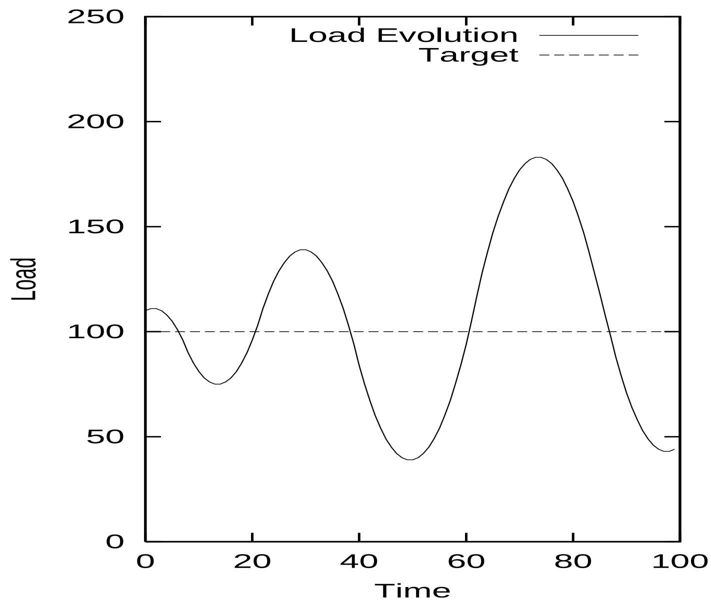
  $\rightarrow$ keeps oscillating

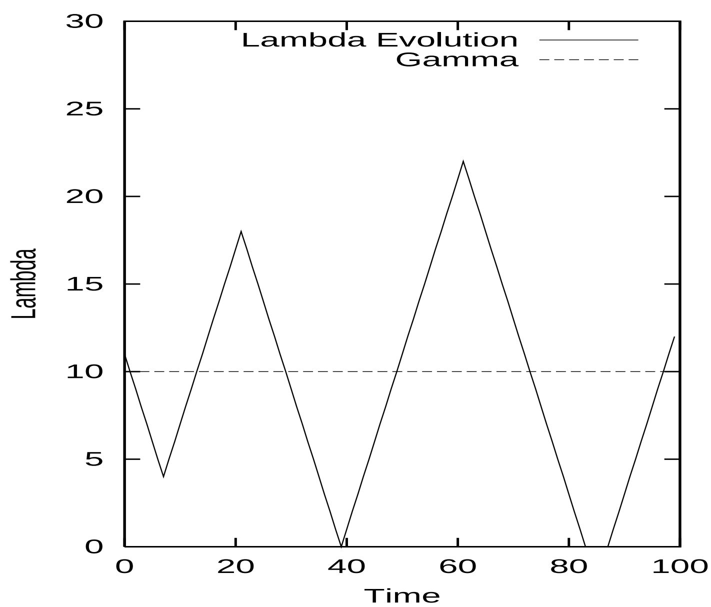
- Actually: would lead to unbounded oscillation if not for physical restriction $\lambda(t) \geq 0$ and $Q(t) \geq 0$

  $\longrightarrow$ easily seen: start from non-zero buffer

  $\longrightarrow$ e.g., $Q(0) = 110$

With $a = 1$, $Q(0) = 110$, $\lambda(0) = 11$:





`Method B`:

- if $Q(t) = Q^*$ then $\lambda(t+1) \leftarrow \lambda(t)$
- if $Q(t) < Q^*$ then $\lambda(t+1) \leftarrow \lambda(t) + a$
- if $Q(t) > Q^*$ then $\lambda(t+1) \leftarrow \delta \cdot \lambda(t)$

where $a > 0$ and $0 < \delta < 1$ are fixed parameters

Note: only decrease part differs from `Method A`.

$\longrightarrow$ linear increase with slope a

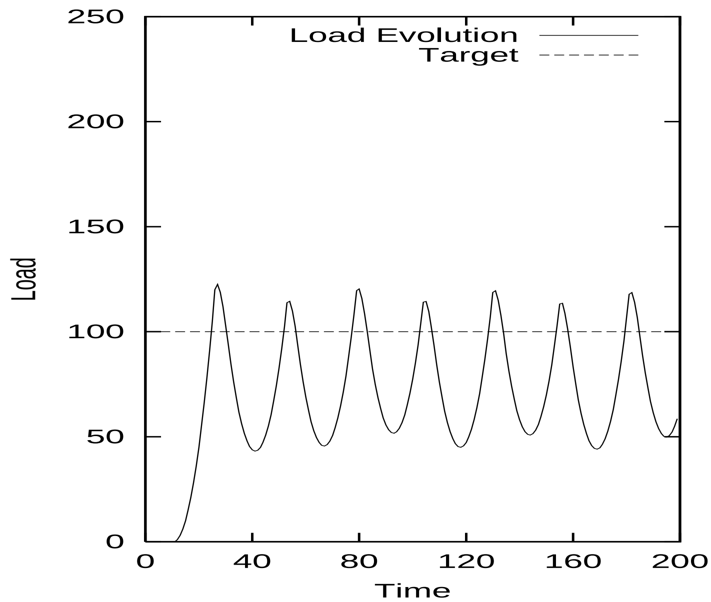
$\longrightarrow$ exponential decrease with backoff factor δ

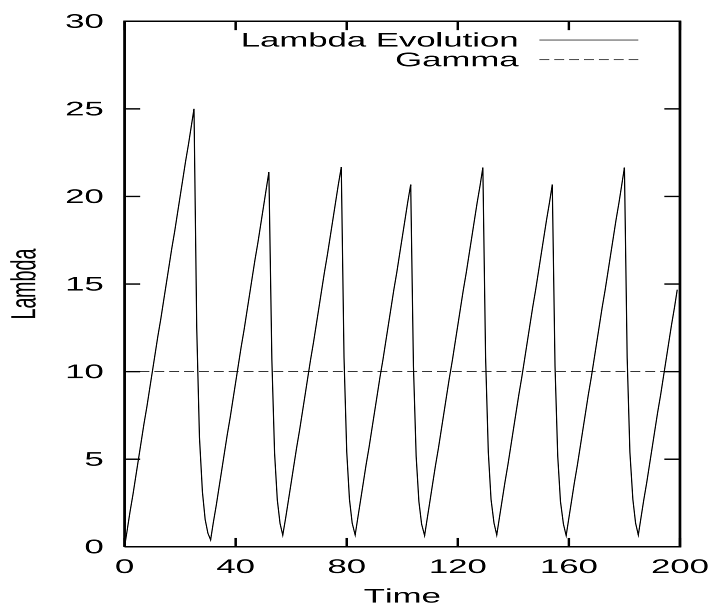
$\longrightarrow$ e.g., binary backoff in case $\delta = 1/2$

Similar to Ethernet and WLAN backoff

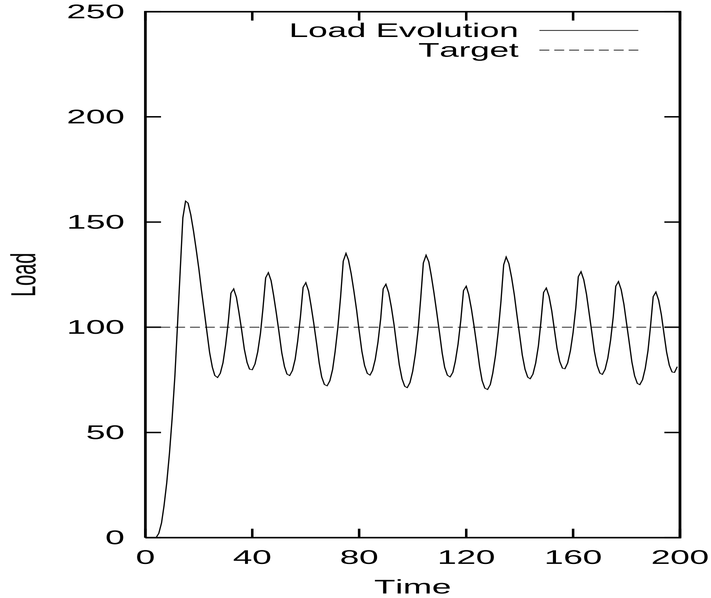
$\longrightarrow$ question: does it work?

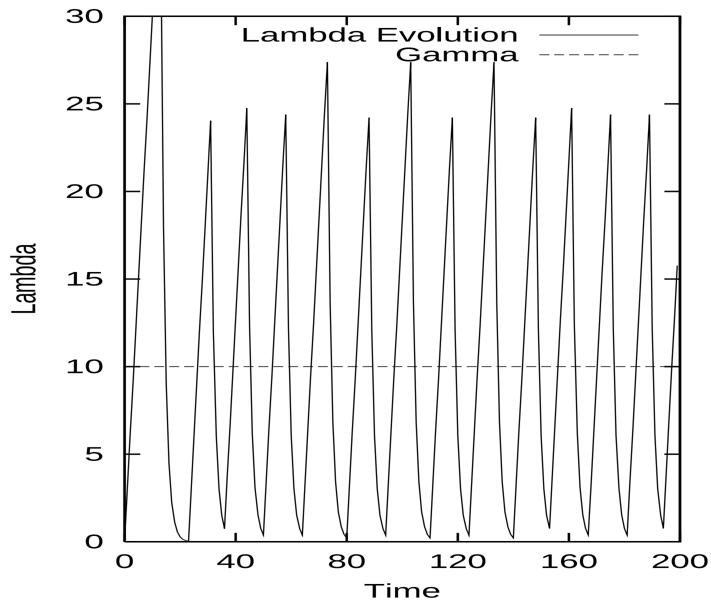
With $a = 1$, $\delta = 1/2$:



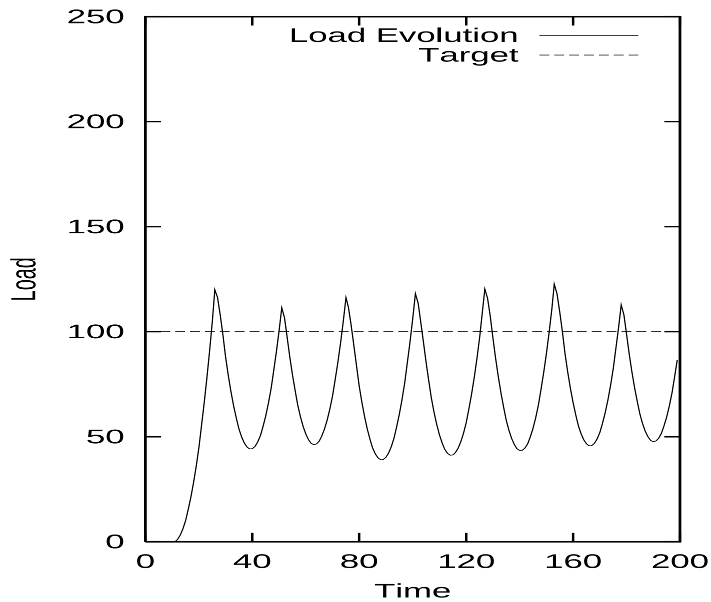


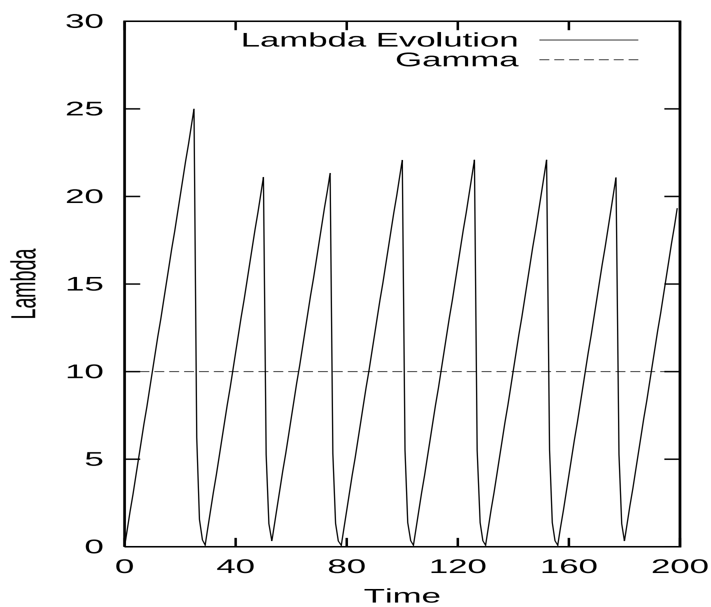
With $a = 3$, $\delta = 1/2$:



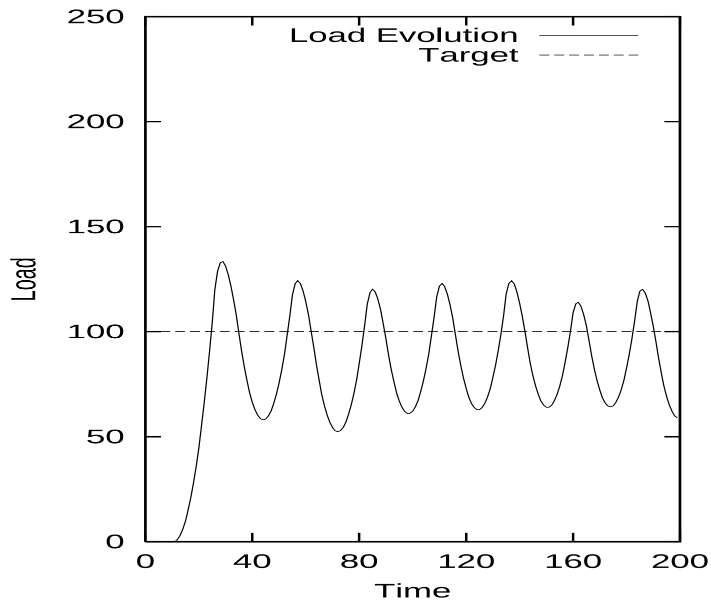


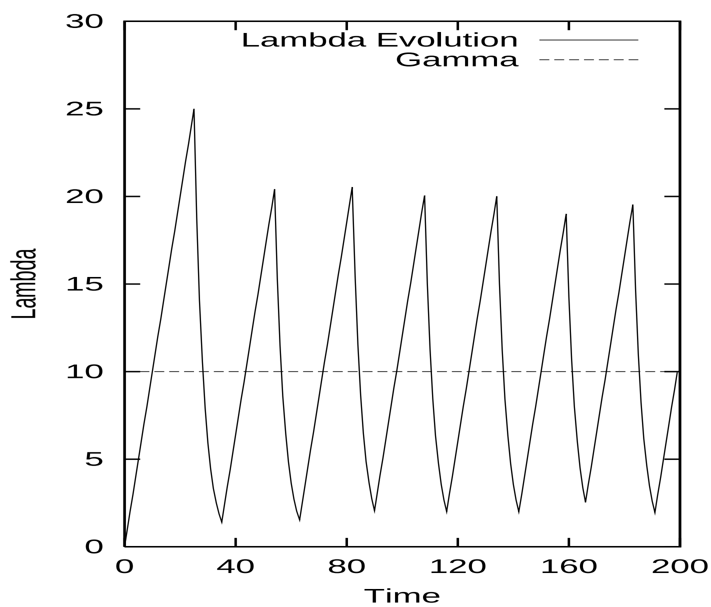
With $a = 1$, $\delta = 1/4$:





With $a = 1$, $\delta = 3/4$:





Note:

- Method B isn't that great either
- One advantage over Method A: doesn't lead to unbounded oscillation

  → note: doesn't hit "rock bottom"

  → due to asymmetry in increase vs. decrease policy

  → typical "sawtooth" pattern
- Method B is used by TCP

  → linear increase/exponential decrease

  → additive increase/multiplicative decrease (AIMD)

Question: can we do better?

⟶ what "freebie" have we not utilized yet?

`Method C`:

$$\lambda(t+1) \leftarrow \lambda(t) + \varepsilon(Q^* - Q(t))$$

where $\varepsilon > 0$ is a fixed parameter

Tries to adjust magnitude of change as a function of the gap $Q^* - Q(t)$

$\longrightarrow$ incorporate distance from target Q^*

$\longrightarrow$ before: just the sign (above/below)

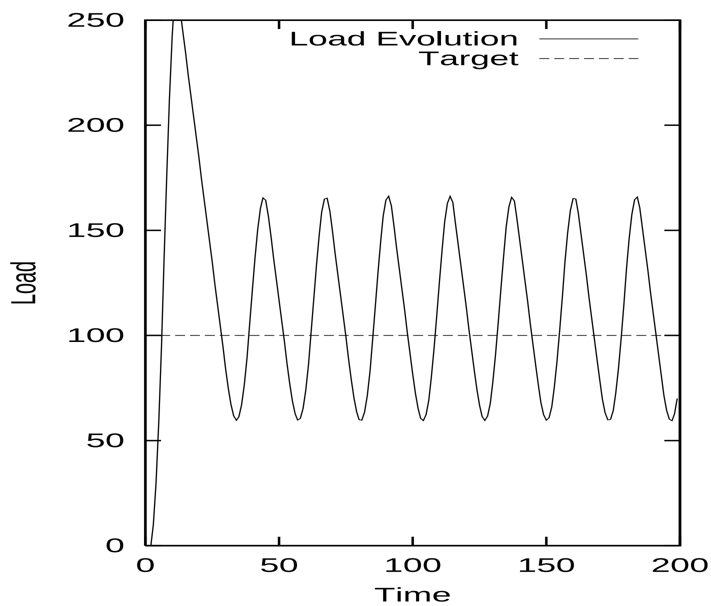
Thus:

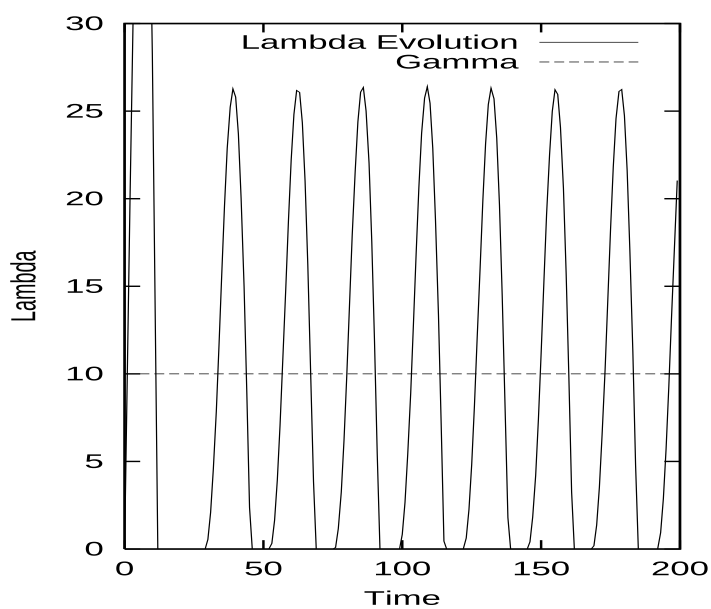
- if $Q^* - Q(t) > 0$, increase $\lambda(t)$ proportional to gap
- if $Q^* - Q(t) < 0$, decrease $\lambda(t)$ proportional to gap

Trying to be more clever...

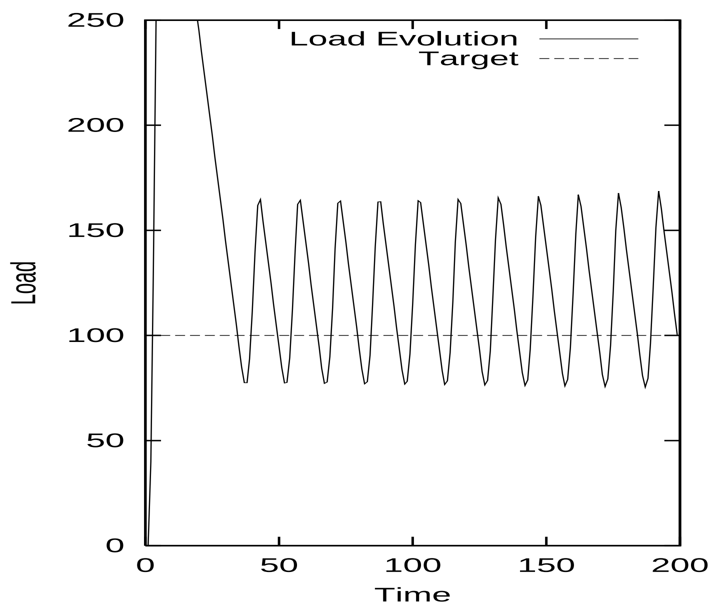
$\longrightarrow$ bottom line: is it any good?

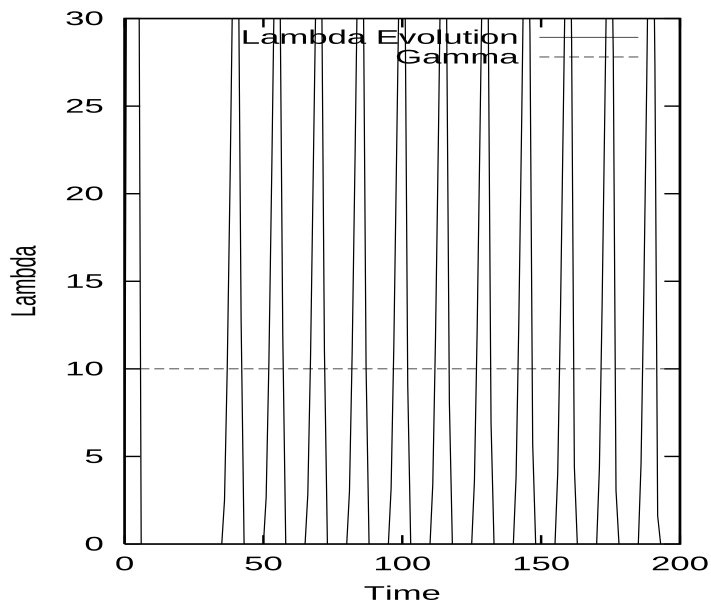
With $\varepsilon = 0.1$:





With $\varepsilon = 0.5$:





Answer: no

$\longrightarrow$ looks good

$\longrightarrow$ but looks can be deceiving

Time to try something strange

$\longrightarrow$ any (crazy) ideas?

$\longrightarrow$ good for course project (assuming it works)

`Method D`:

$$\lambda(t+1) \leftarrow \lambda(t) + \varepsilon(Q^* - Q(t)) - \beta(\lambda(t) - \gamma)$$

where $\varepsilon > 0$ and $\beta > 0$ are fixed parameters

$\longrightarrow$ odd looking modification to `Method C`

$\longrightarrow$ additional term $-\beta(\lambda(t) - \gamma)$

$\longrightarrow$ what's going on?

Sanity check: at desired operating point $Q(t) = Q^*$ and $\lambda(t) = \gamma$, nothing should move

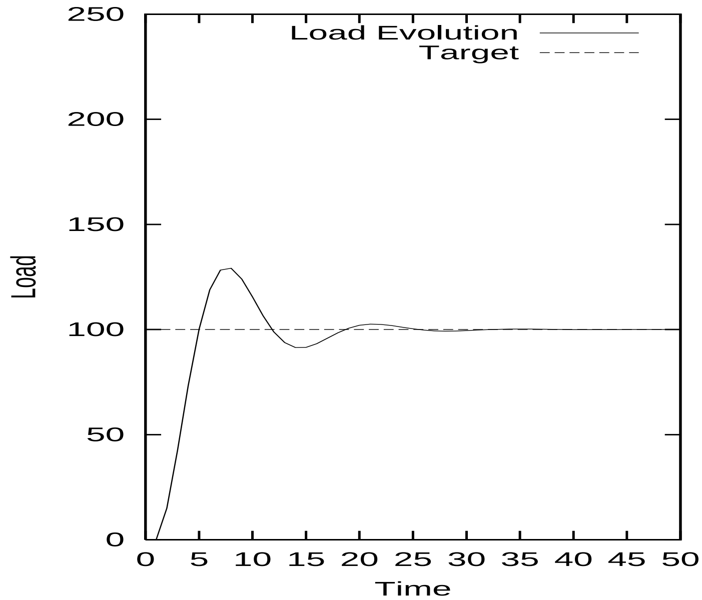
$\longrightarrow$ check with methods A, B and C

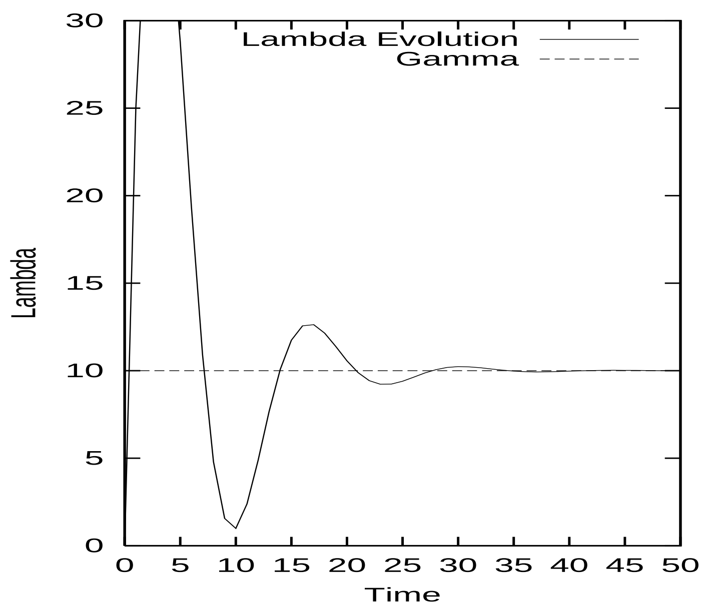
$\longrightarrow$ fixed-point property

$\longrightarrow$ what about `Method D`?

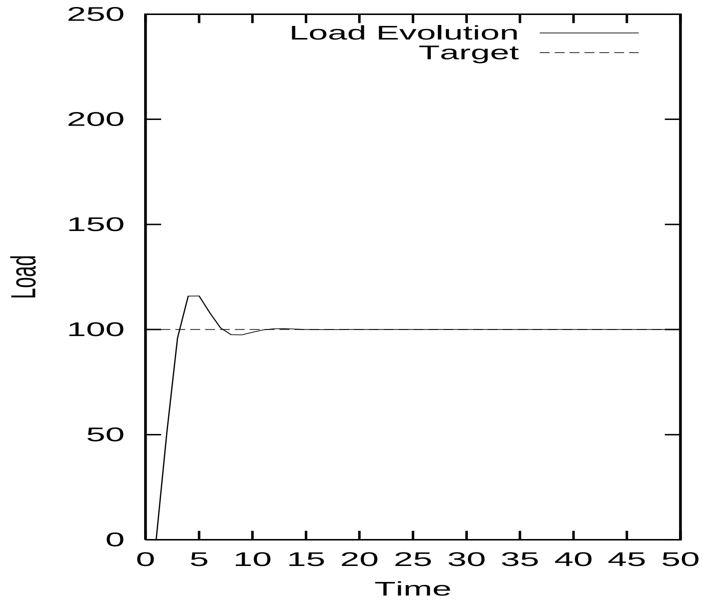
Now: does `Method D` get to the targe fixed point?

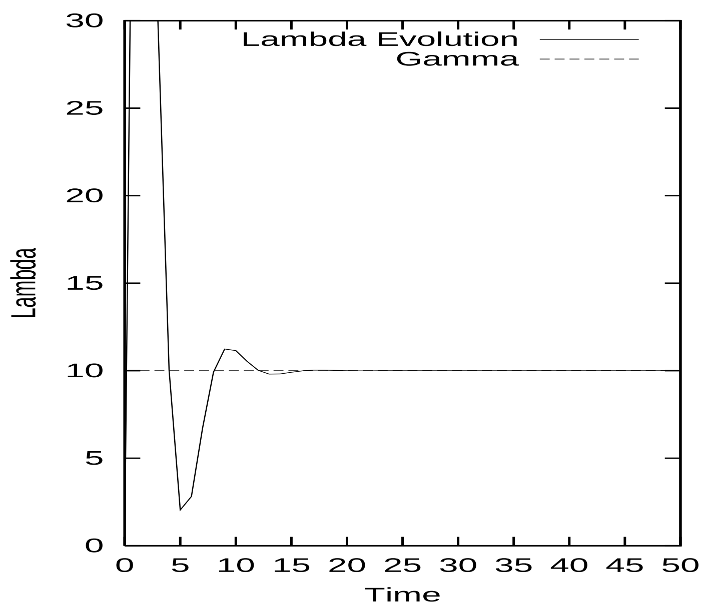
With $\varepsilon = 0.2$ and $\beta = 0.5$:



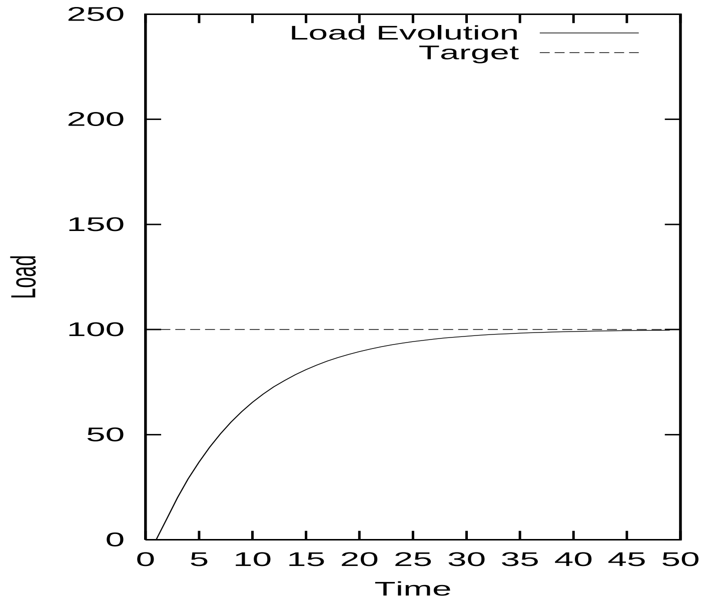


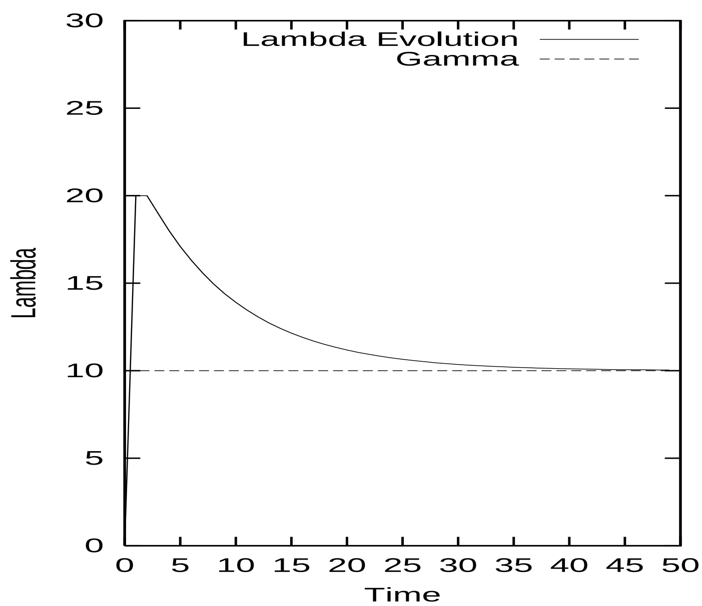
With $\varepsilon = 0.5$ and $\beta = 1.1$:





With $\varepsilon = 0.1$ and $\beta = 1.0$:





Remarks:

- Method D has desired behavior
- Is superior to Methods A, B, and C
- No unbounded oscillation
- In fact, dampening and convergence to desired operating point

  $\rightarrow$ converges to target operating point (Q^*, γ)

  $\rightarrow$ asymptotic stability

Why does it work?

What is the role of the $-\beta(\lambda(t) - \gamma)$ term in the control law:

$$\lambda(t+1) \leftarrow \lambda(t) + \varepsilon(Q^* - Q(t)) - \beta(\lambda(t) - \gamma)$$

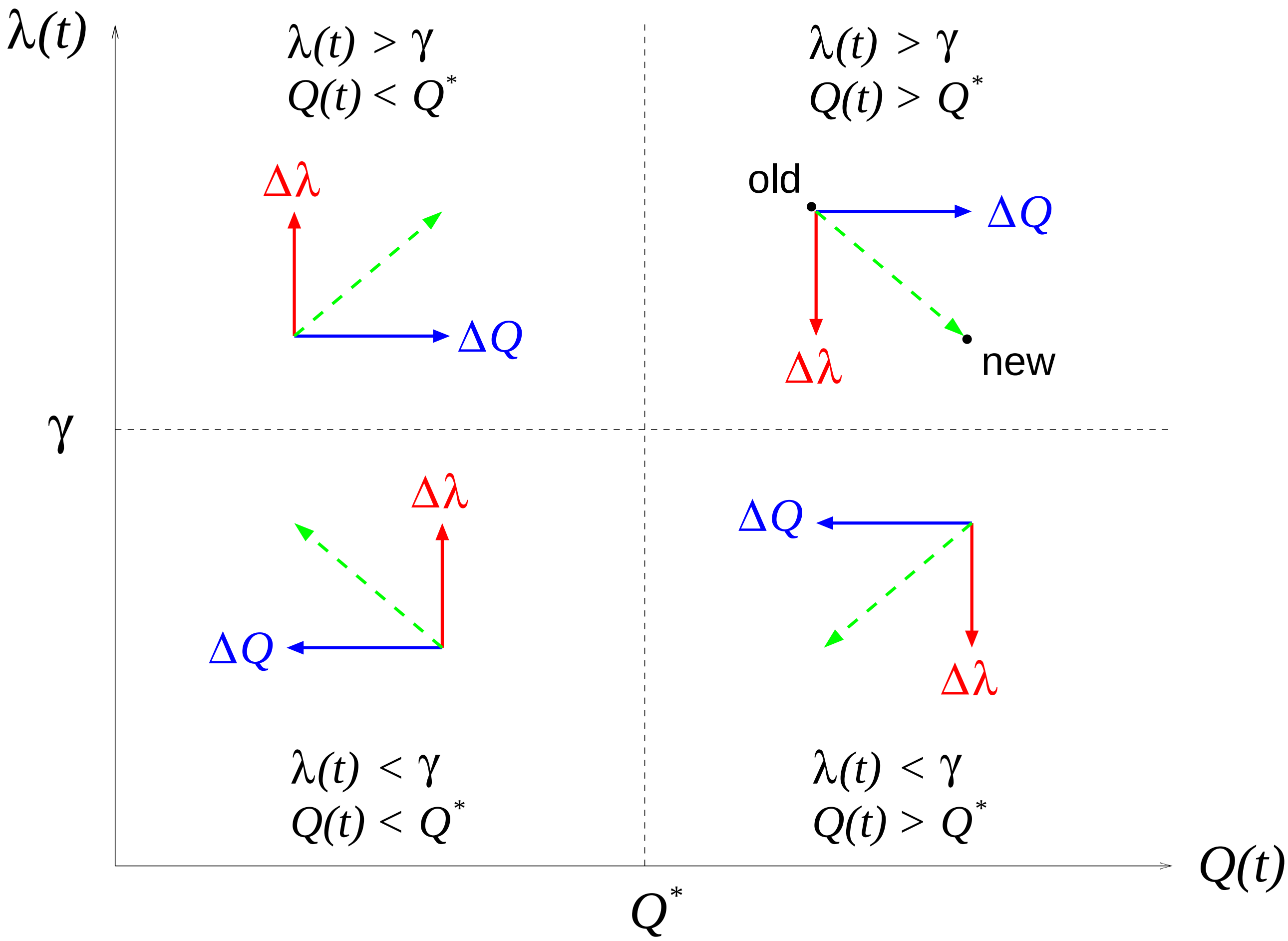
Need to look beneath the hood . . .

$\longrightarrow$???

$\longrightarrow$ intuition

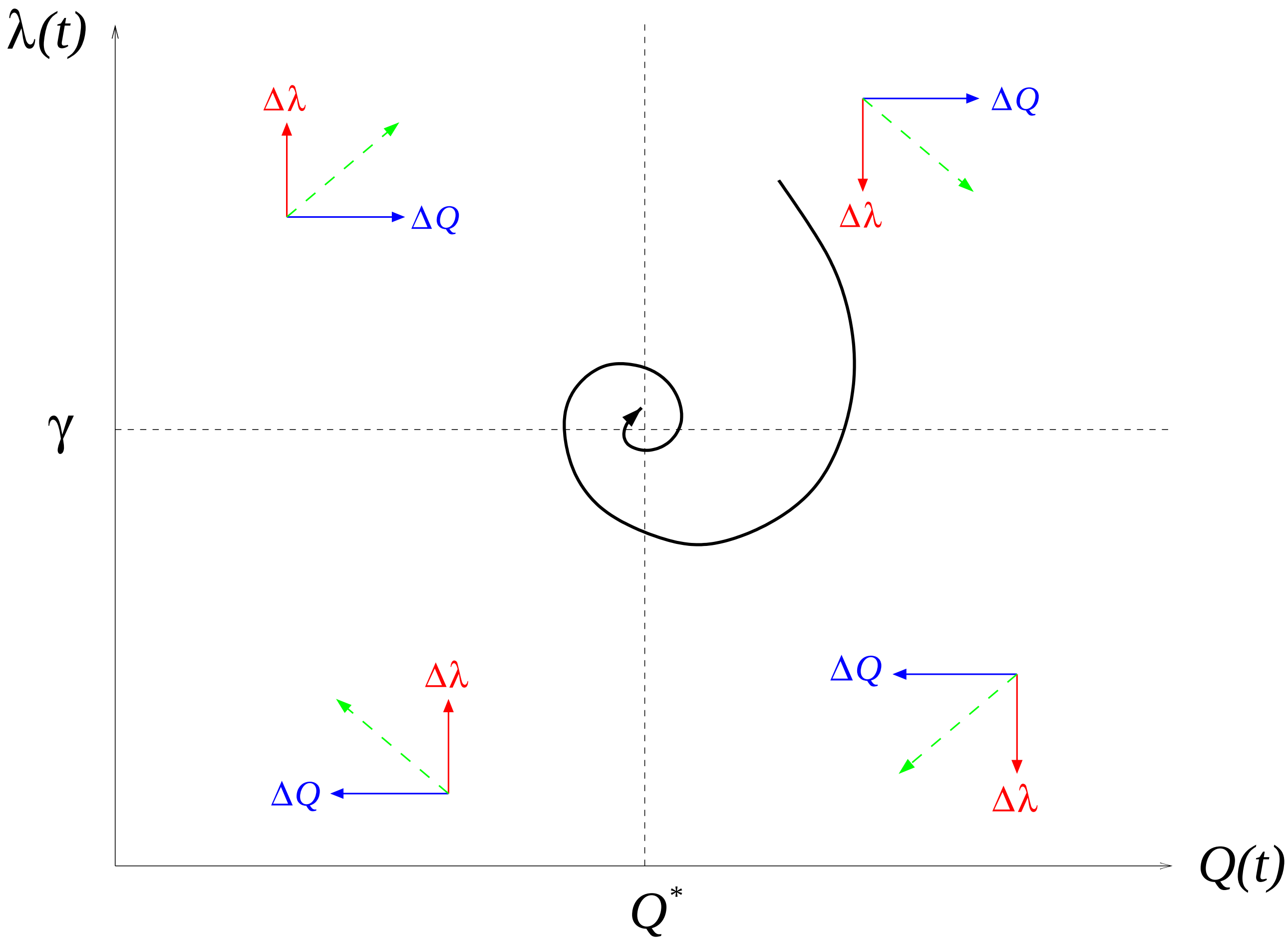
Visualize action in 2-D $(Q(t), \lambda(t))$-space:

$\longrightarrow$ phase space



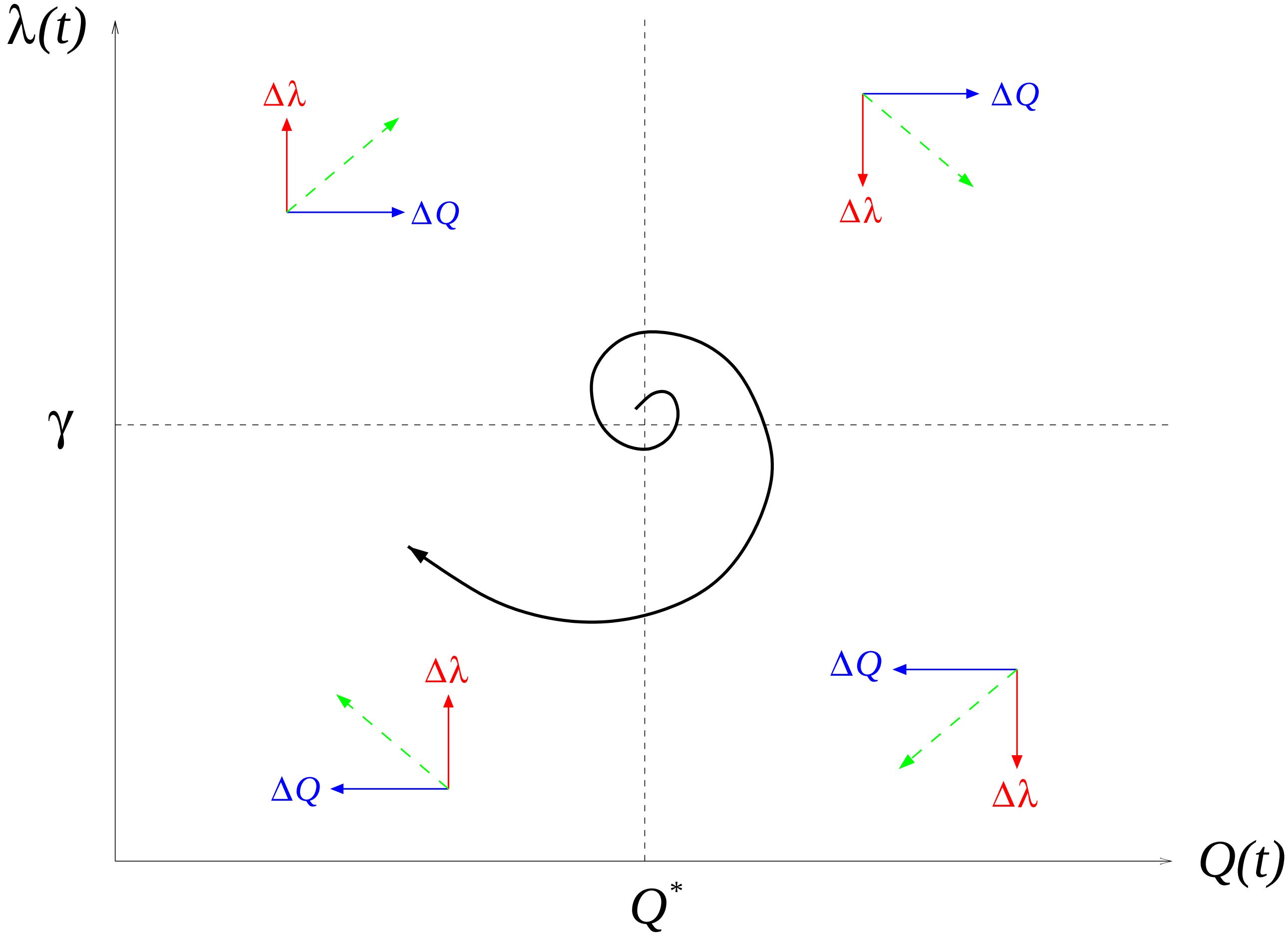
Convergent trajectory:

$\longrightarrow$ asymptotically stable & optimal



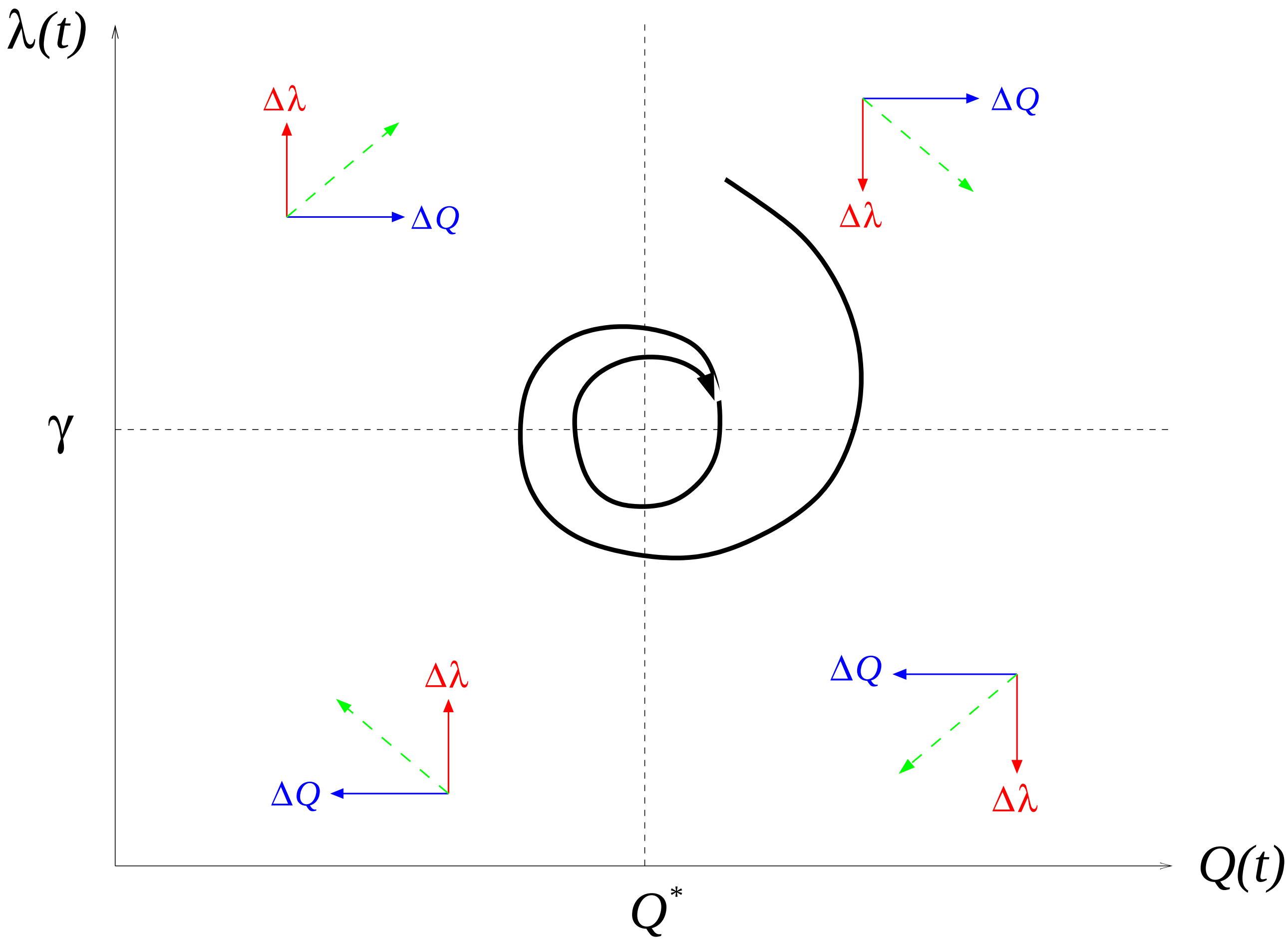
Divergent trajectory:

$\longrightarrow$ unstable



Stable (but not asymptotically so) trajectory:

$\longrightarrow$ limit cycle



Which case arises depends on the specifics of protocol actions.

For example:

- Methods A and C: divergent
- Method B: stable (but not asymptotically)

  $\rightarrow$ TCP
- Method D: asymptotically stable & optimal

  $\rightarrow$ "optimal control"

Why does Method D work:

$\longrightarrow$ overview of underlying mathematics