Teoria Sterowania

Sterowanie inteligentne

Ziemowit Dworakowski

Akademia Górniczo-Hutnicza,
Katedra Robotyki i Mechatroniki

Agenda

- Wprowadzenie
- Podejście ewolucyjne w identyfikacji
- Podejście ewolucyjne w sterowaniu adaptacyjnym
- *Reinforcement learning*
- *Model-predictive control*

Kiedy klasyczne metody nie zdają egzaminu?

Zakładaliśmy do tej pory wykorzystywanie układów:

- **Liniowych**
- **Stacjonarnych**
- **Bez dużego opóźnienia**
- **Prostych w identyfikacji**

Jak podejść do sterowania, gdy obiekt sterowania jest silnie nieliniowy, nie daje się zlinearyzować wokół punktu pracy i/lub jest niestacjonarny?

Kiedy klasyczne metody nie zdają egzaminu?

Sztuczna inteligencja

Inteligentna identyfikacja układów (w tym budowa modeli układów nieliniowych oraz nadążanie za zmianami)

Inteligentne sterowanie (w tym adaptacyjne oraz pozwalające na kontrolę układów nieliniowych)

Sterowanie rozmyte (pozwalające na wykorzystanie doświadczenia eksperta)

- Ze względu na ograniczony zakres czasu na wykładzie i laboratoriach zajmiemy się wyłącznie pokazaniem wybranych przykładów oraz przedstawieniem ogólnej zasady działania metod. Nie będziemy rozważać układów nieliniowych (skupimy się na problemach niestacjonarnych), ale ogólna zasada działania przedstawianych rozwiązań pozostałaby ta sama również w przypadku pojawiania się nieliniowości w układzie sterowania.

Teoria sterowania: sterowanie inteligentne

Sterowanie inteligentne to samodzielny moduł, oceniany osobno. W jego skład wchodzą:

4 instrukcje laboratoryjne:

- Ewolucyjna identyfikacja obiektu sterowania
- Ewolucyjna optymalizacja kontrolera PID
- Sterowanie predykcyjne z modelem opartym o sieć neuronową
- Sterowanie predykcyjne oraz optymalizacja ewolucyjna układów zmiennych w czasie

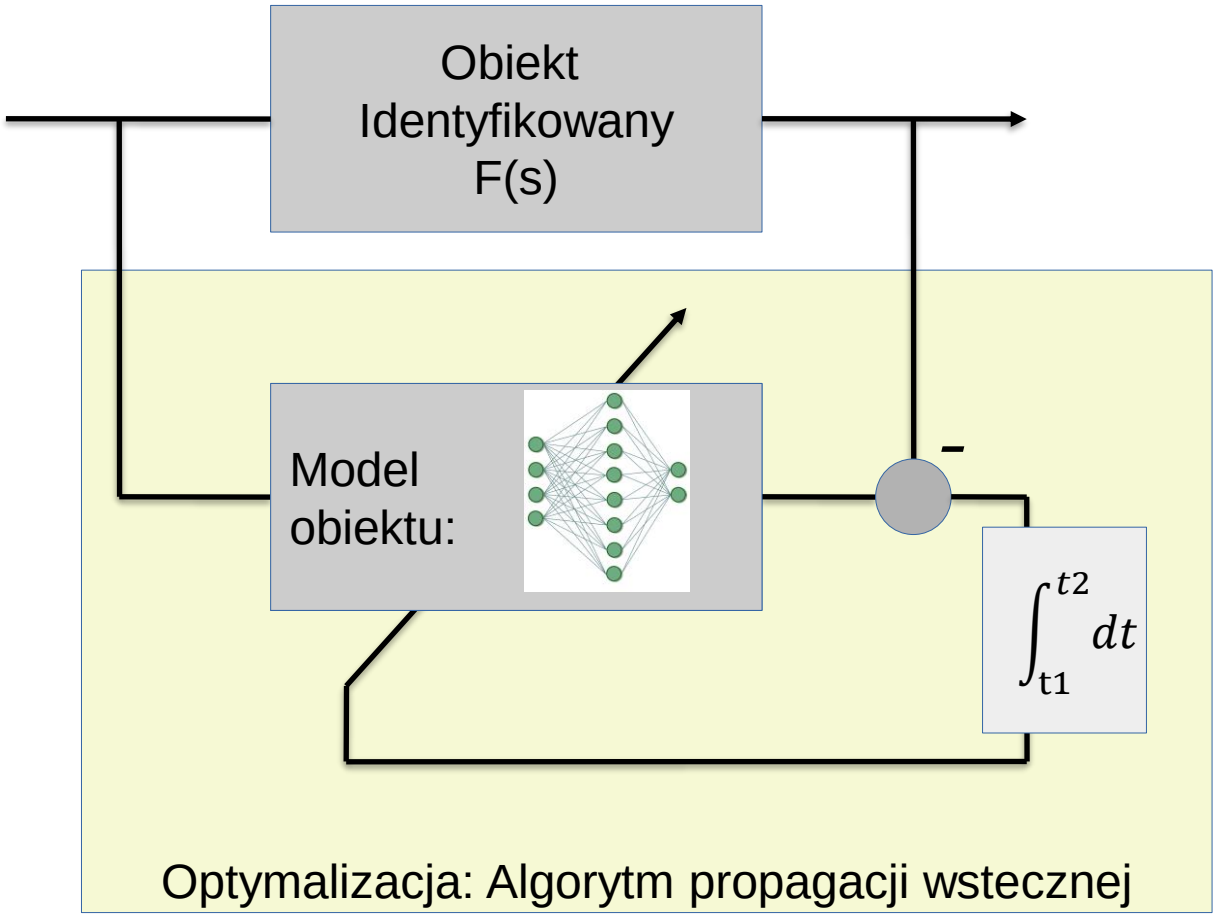
Oceniane jak na PSIWSUM/PSIWMUM: Im więcej zadań zrobionych na zajęciach, tym wyższa ocena. Instrukcje oraz materiały do zajęć do pobrania z http://galaxy.agh.edu.pl/~zdw/Materials/CT/

1 test

Na 1 zajęciach w ramach modułu. Możliwy do poprawy 2 razy. Wymagany zakres wiedzy obejmuje niniejszy wykład.

Ocena końcowa z modułu stanowi średnią ważoną ocen z instrukcji z wagą 1 oraz testu z wagą 2. Aby uzyskać zaliczenie modułu, wszystkie oceny muszą być pozytywne.

Identyfikacja neuronowa



+ Możliwość identyfikacji nieliniowości

+ Obiekt identyfikowany nie musi być wyłączany z użytkowania

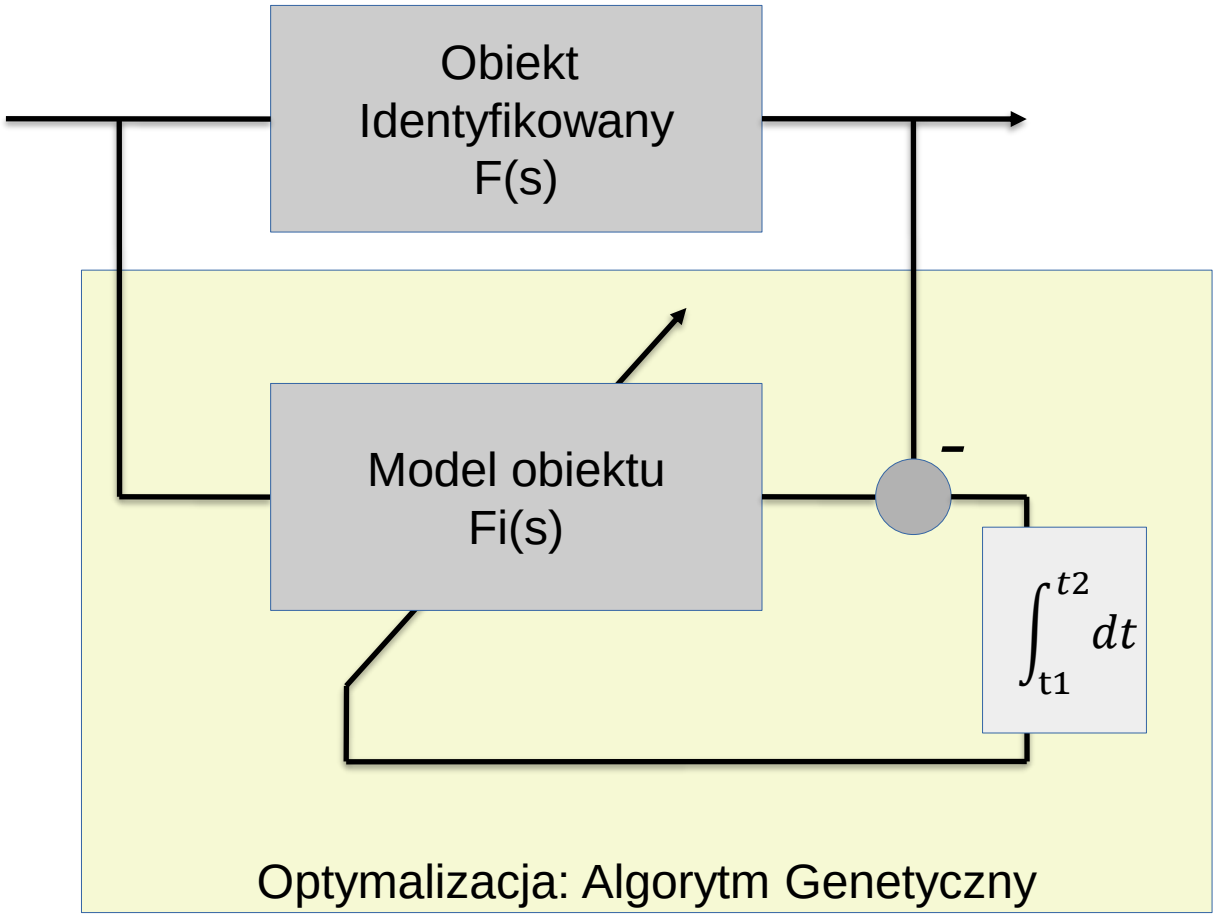
+ Szybka nauka

+ Możliwość łatwej adaptacji *(cykliczne douczanie sieci)*

- Reprezentacja nie jest intuicyjna

- Brak możliwości dalszego badania zidentyfikowanego obiektu (np. stabilności)

Identyfikacja ewolucyjna



+ Zachowujemy intuicyjną reprezentację *(w tym możliwość jej dalszych badań)*

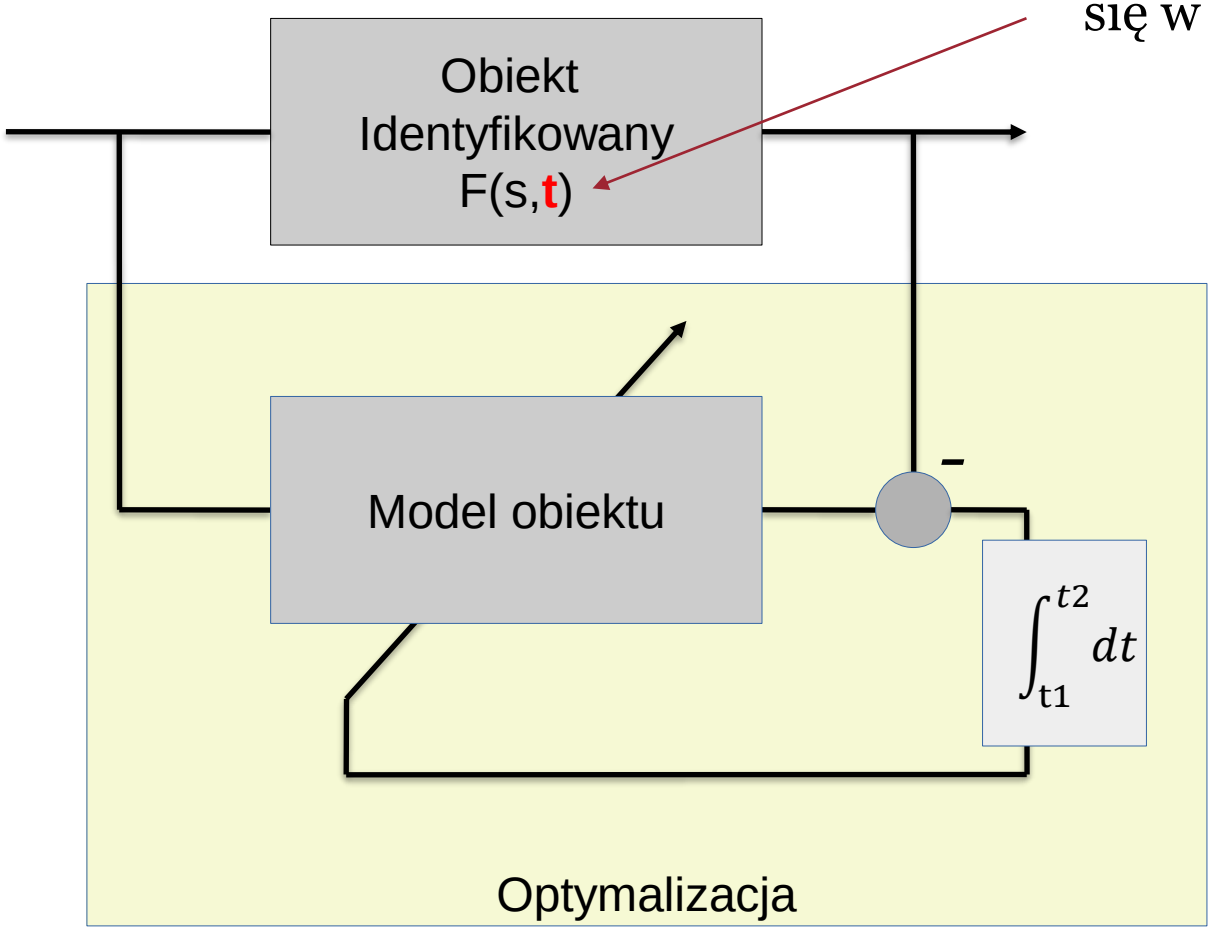
+ Obiekt identyfikowany nie musi być wyłączany z użytkowania

+ Możliwość ~~łatwej~~ adaptacji

- Problem złożony obliczeniowo

- Konfiguracja AG jest trudna

Identyfikacja w problemach niestacjonarnych

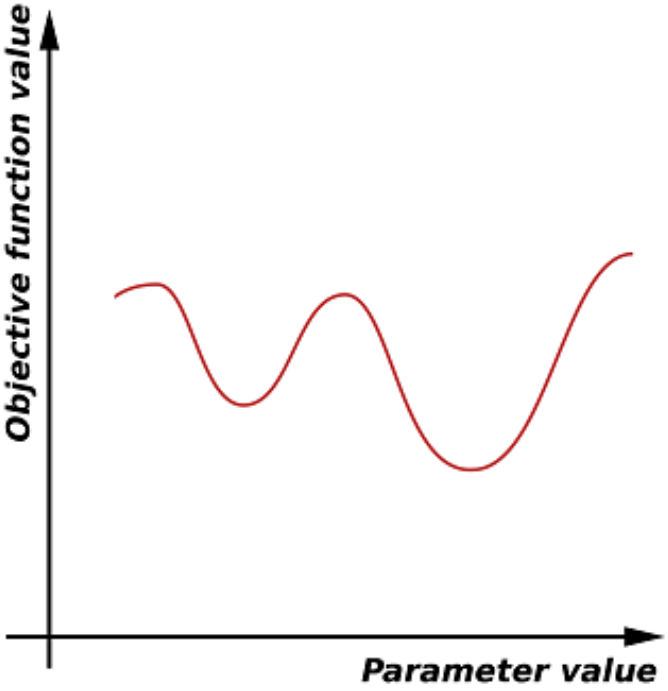


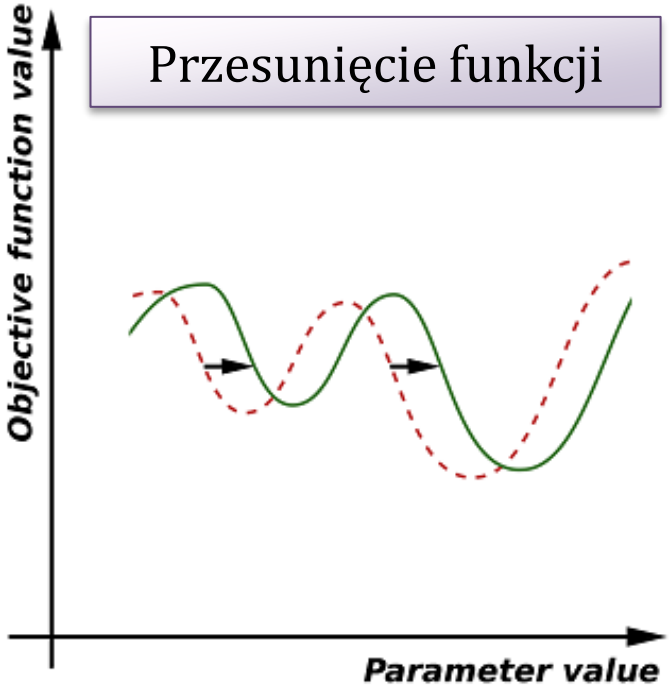
Co w przypadku, gdy obiekt zmienia się w czasie?

Reprezentacja powinna pozwalać na łatwą adaptację do nowych warunków (zmiany są zazwyczaj płynne, nie ma potrzeby identyfikowania „od zera”)

Optymalizacja adaptacyjna

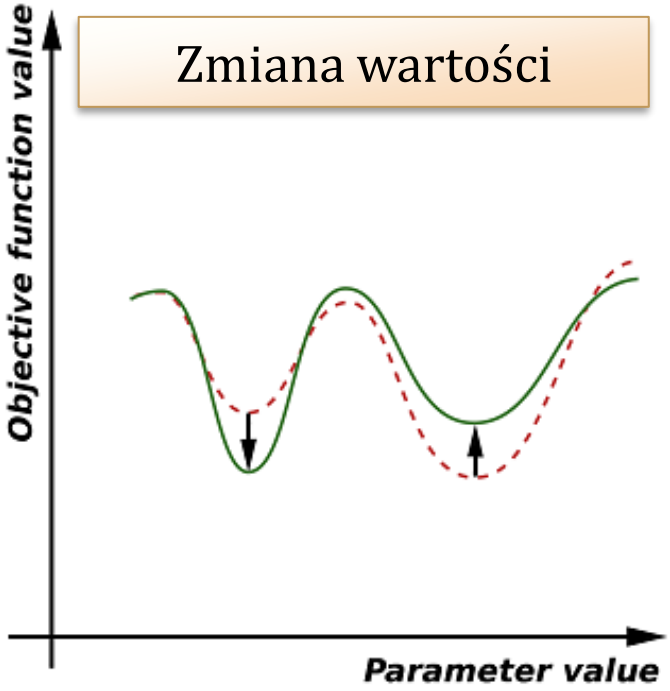
Mamy dwa rodzaje zmian funkcji celu
– zazwyczaj występujące jednocześnie, w różnym stopniu:





Przesunięcie funkcji

Częste poprawki
(np. z wykorzystaniem algorytmu gradientowego startując od poprzedniego minimum globalnego)



Zmiana wartości

Dobrze by było śledzić położenie wielu minimów lokalnych jednocześnie…

Optymalizacja adaptacyjna

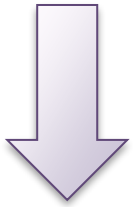
Wymagania:

- AG powinien utrzymać populację w wielu minimach jednocześnie → Spora populacja, niski nacisk selektywny, być może dodatkowe podtrzymanie różnorodności
- AG powinien nadążać za zmianami położeń minimów → Krok mutacji nigdy nie zbliża się do zera. Po początkowym przeszukaniu przestrzeni i uzyskaniu konwergencji krok pozostaje na stałym poziomie
- AG powinien zachowywać zdolności eksploracyjne przez cały czas trwania optymalizacji (zdolność do znajdowania nowych minimów) → Część populacji jest generowana losowo lub też mutowana z zastosowaniem relatywnie dużego kroku mutacji

Alternatywa: Elastyczny genom

Jeśli osobnik znajduje się **w pobliżu** minimum lokalnego,
tylko **mały** zakres mutacji umożliwi poprawę rezultatu

Jeśli osobnik znajduje się **daleko** od minimum lokalnego,
tylko **duży** zakres mutacji umożliwi poprawę rezultatu



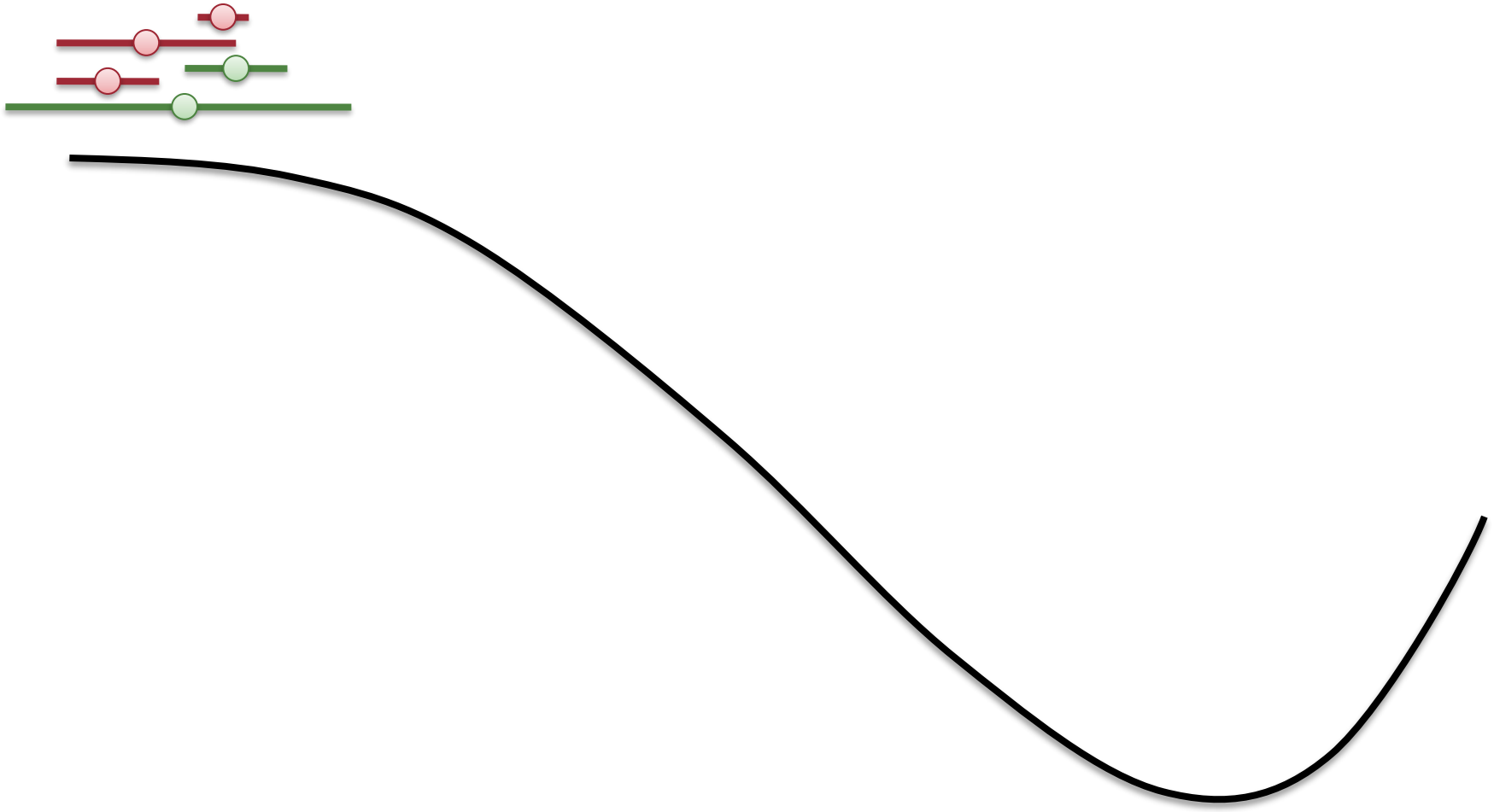
Odwróćmy wnioskowanie…

Skoro osobnika wygenerowano **małym** zakresem mutacji a poprawił rezultat,
znajduje się **w pobliżu** minimum lokalnego

Skoro osobnika wygenerowano **dużym** zakresem mutacji a poprawił rezultat,
znajduje się **daleko** od minimum lokalnego

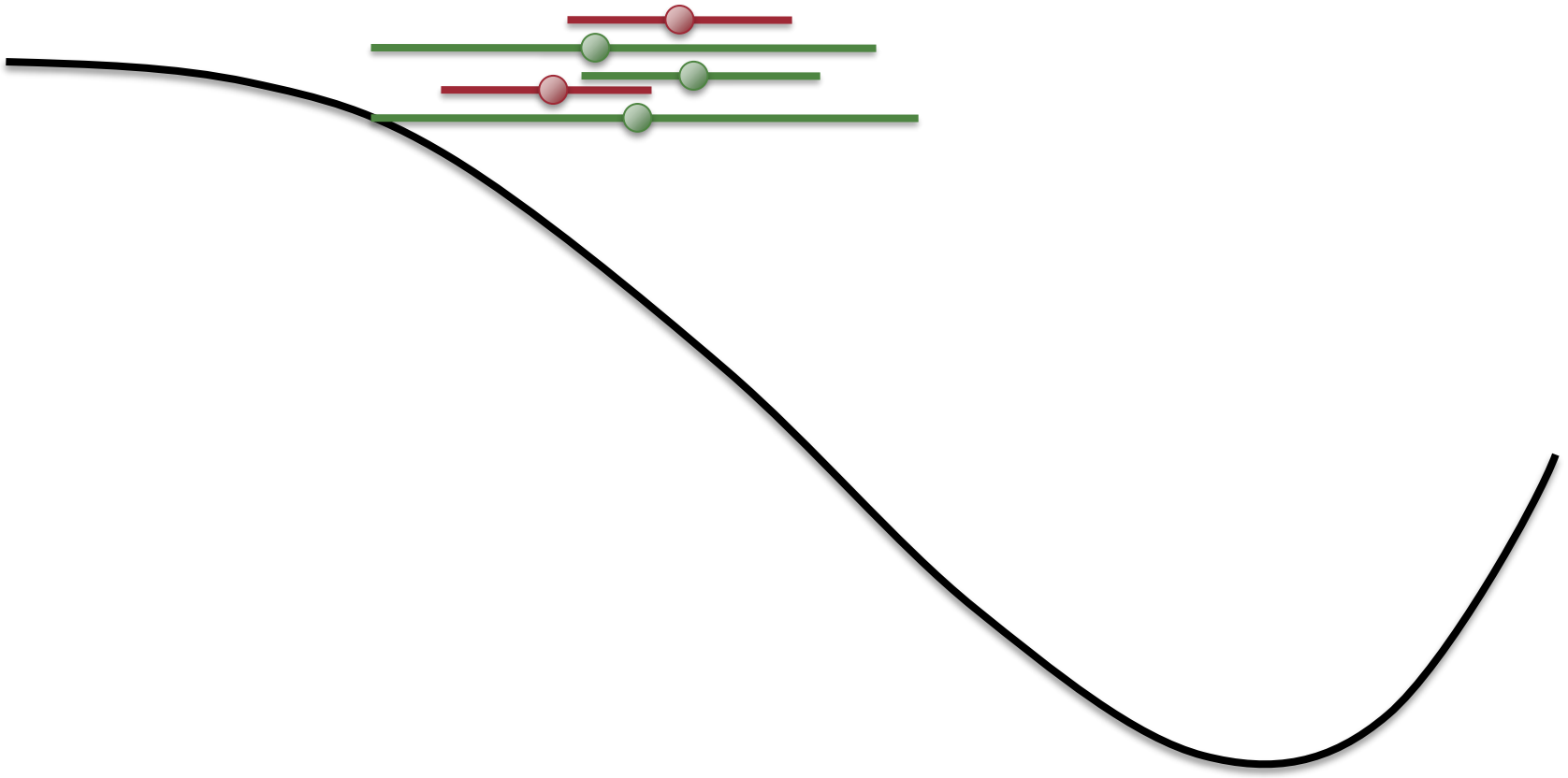
Elastyczny genom

Niech zakres mutacji nie będzie zależny od czasu,
ale znajdzie się w genomie, jako kolejny gen



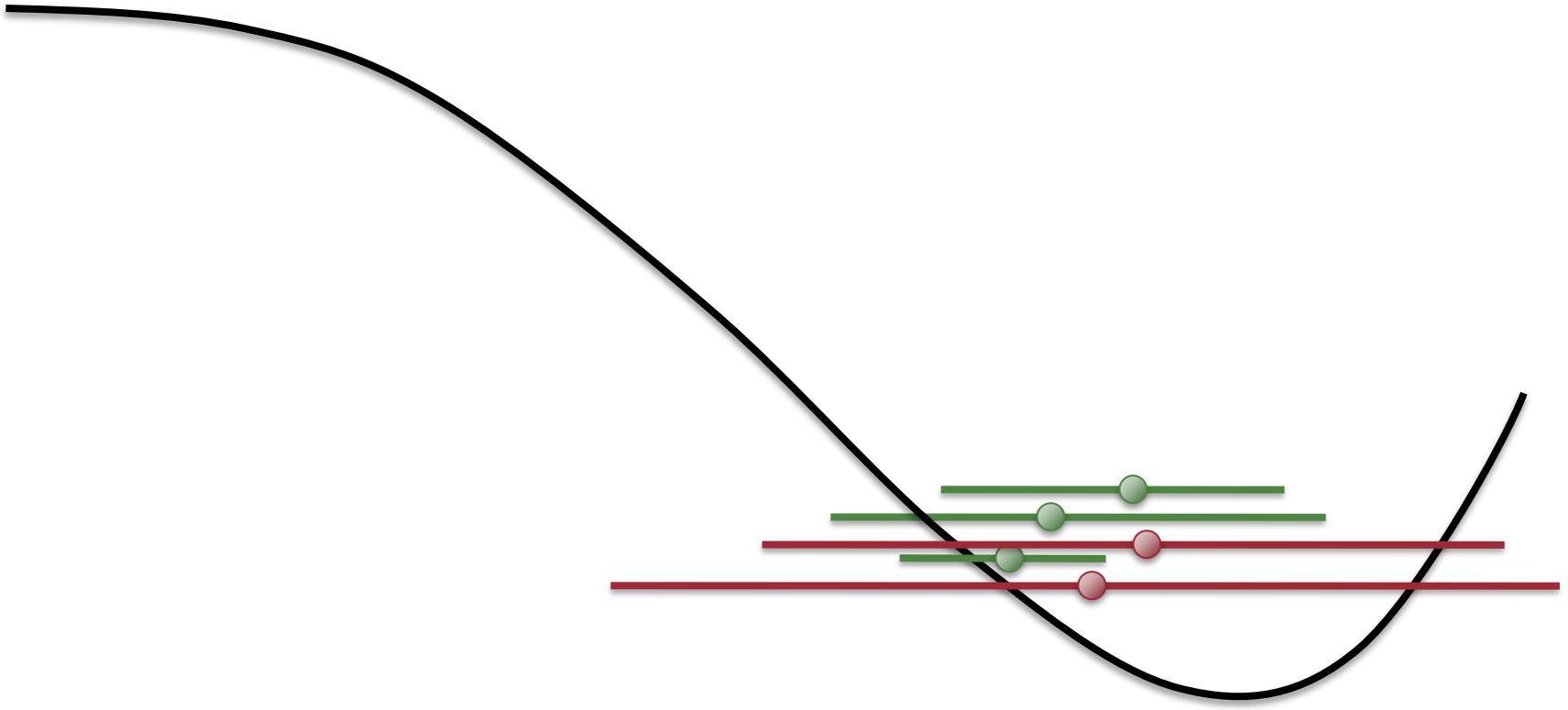
Elastyczny genom

Niech zakres mutacji nie będzie zależny od czasu,
ale znajdzie się w genomie, jako kolejny gen



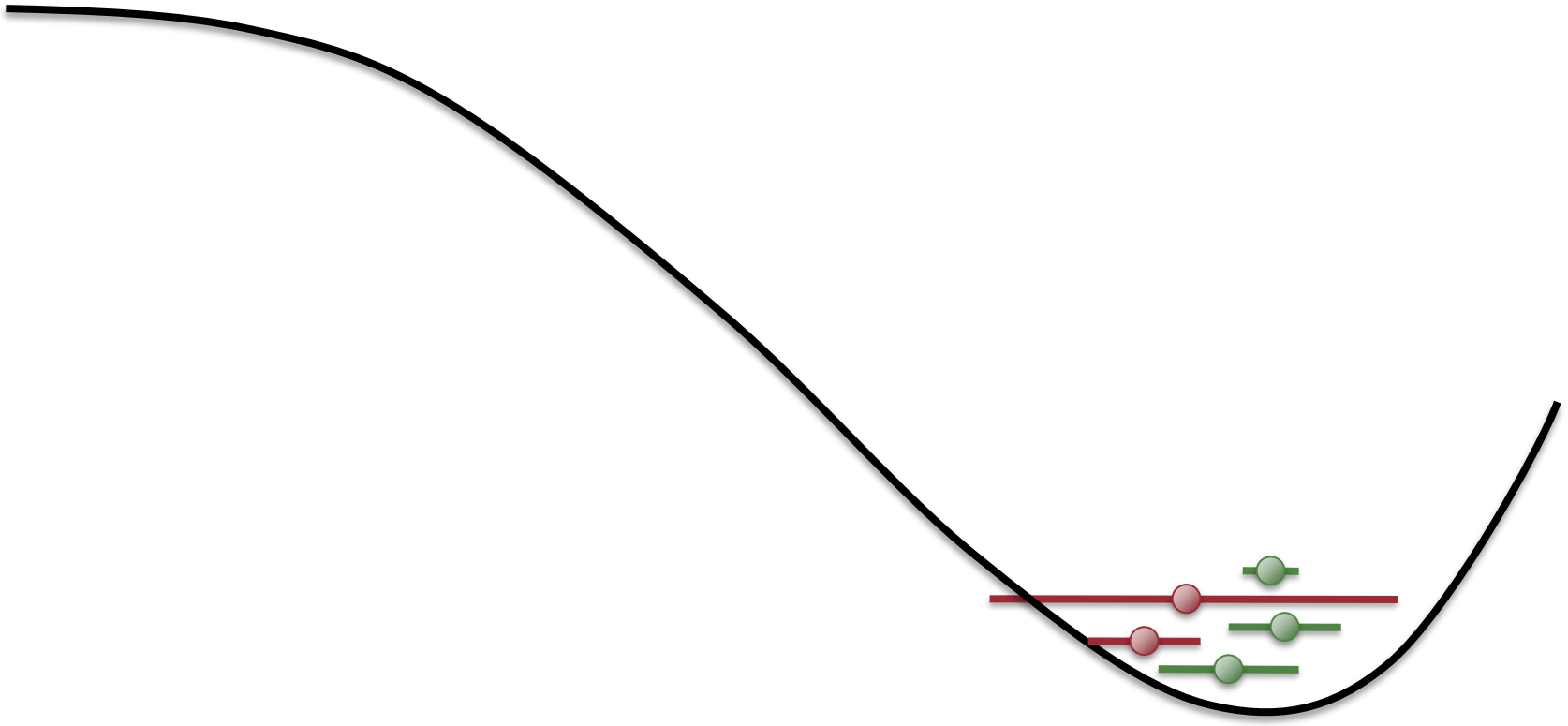
Elastyczny genom

Niech zakres mutacji nie będzie zależny od czasu,
ale znajdzie się w genomie, jako kolejny gen



Elastyczny genom

Niech zakres mutacji nie będzie zależny od czasu,
ale znajdzie się w genomie, jako kolejny gen



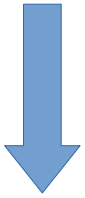
Optymalizacja adaptacyjna

Jak budować genotyp?

Wiedza kontekstowa: co wiemy o obiekcie? Czy dopuszczamy obiekty niestabilne? Czy znamy rząd układu albo ogólną postać transmitancji?

Czy jesteśmy w stanie wygenerować początkową populację tak, by nie było konieczności odrzucania obiektów nie spełniających oczywistych wymagań?

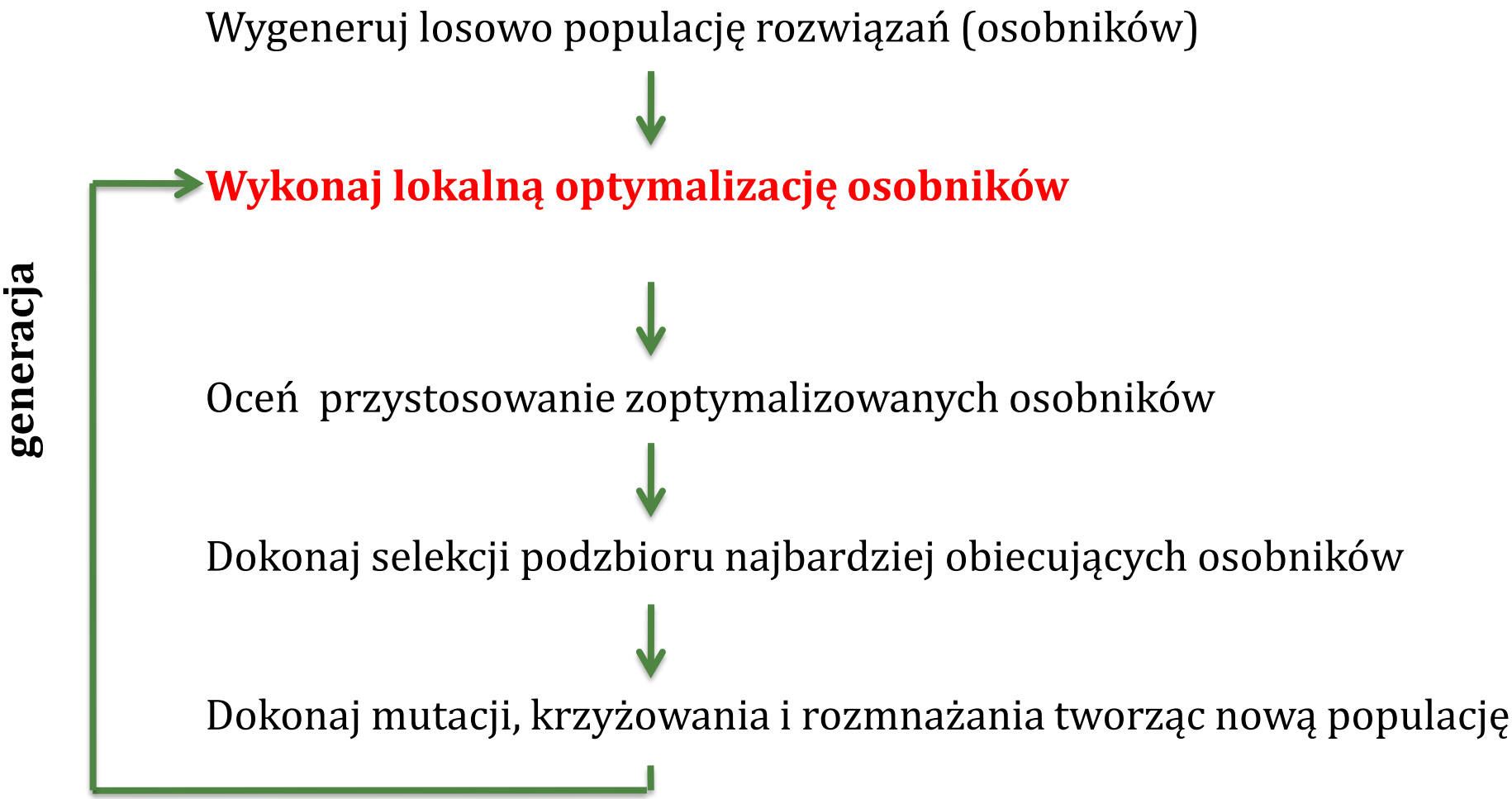
A może umożliwić AG zbudowanie własnej reprezentacji (również ewolucyjnie?)



Struktura genotypu poddawana mutacji i krzyżowaniu

Algorytm memetyczny (hybrydowy, lamarkowski)

Podstawowy algorytm memetyczny (hybrydowy, lamarkowski)



Powtarzaj dopóki nie skończy się czas lub algorytm nie przestanie poprawiać rozwiązań

Algorytm memetyczny (hybrydowy, lamarkowski)

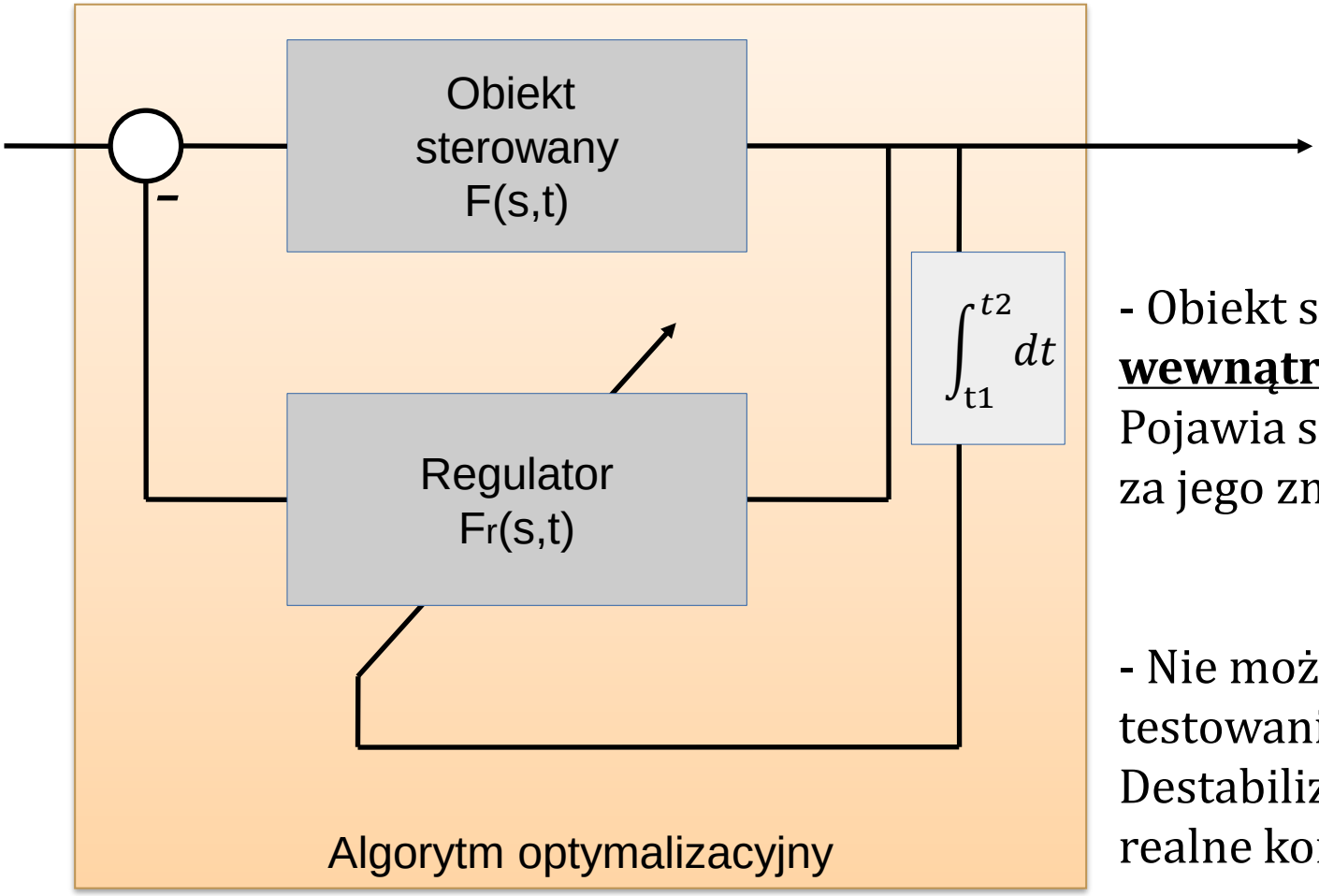
Podstawowy algorytm memetyczny (hybrydowy, lamarkowski)

Zalety:
- Wymagana mniejsza populacja
- Nadążanie za zmianami funkcji celu staje się dużo prostsze

Wady:
- Jeszcze większa ilość metaparametrów do konfiguracji
- Skomplikowany w implementacji
- Zazwyczaj wymaga więcej mocy obliczeniowej od klasycznego AG

Sterowanie adaptacyjne

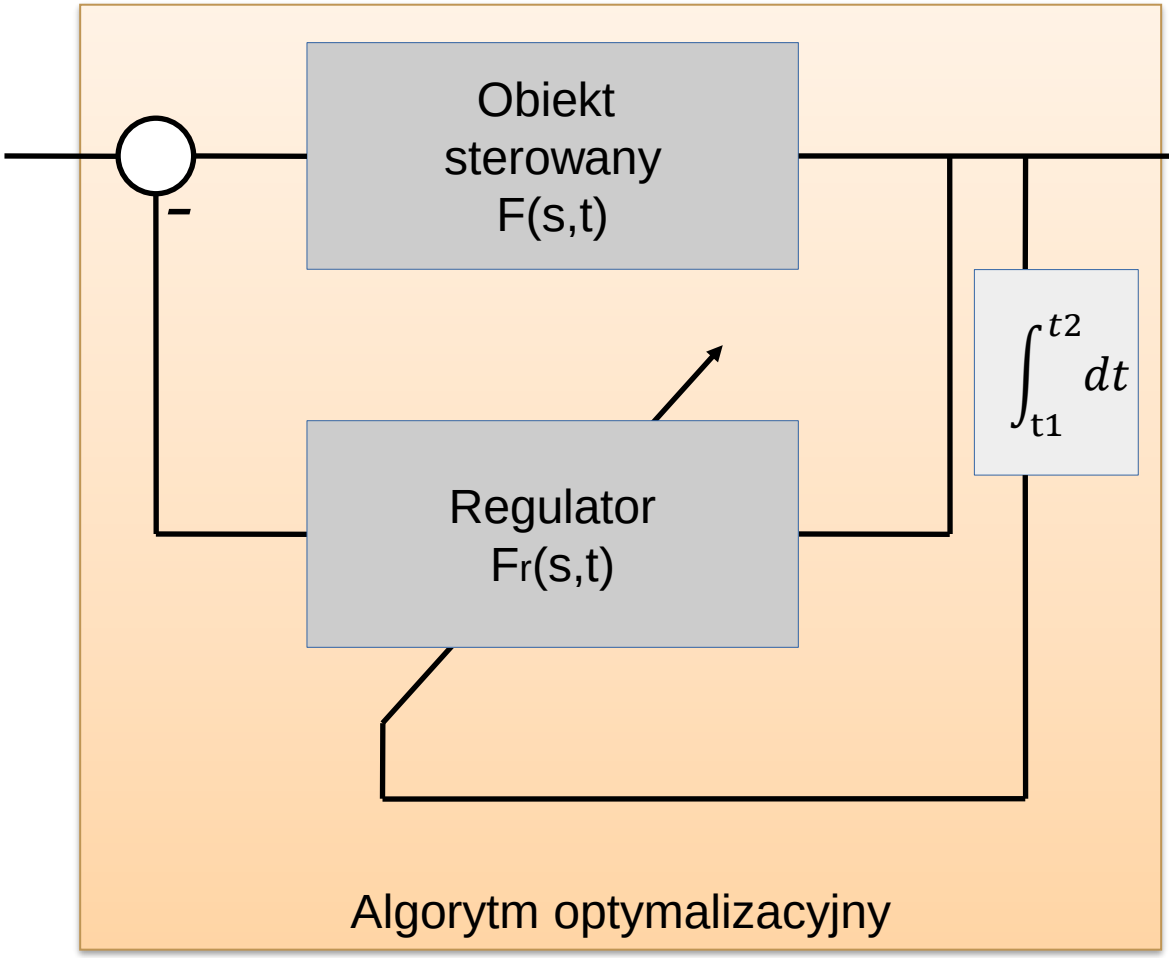
Dlaczego tym razem nie będzie już tak łatwo?



- Obiekt sterowany znajduje się **wewnątrz** pętli optymalizacyjnej: Pojawia się konieczność nadążania za jego zmianami dużo szybciej

- Nie możemy pozwolić sobie na testowanie dowolnych parametrów: Destabilizacja obiektu będzie miała realne konsekwencje.

Dlaczego tym razem nie będzie już tak łatwo?



Możliwe podejścia:

- Nie adaptujmy w ogóle (sterowanie odporne)

- Metody klasyczne (nie wykorzystujące AI)

- Metody bazujące na ewolucyjnej adaptacji prostych regulatorów (np. PID)

- Adaptacyjne sterowanie predykcyjne

- …

Ewolucja regulatorów

- Jako punkt wyjściowy potrzebny jest „w miarę dobry” regulator – np. PID (aby zapewnić poprawną regulację od samego początku)

- Obiekt musi być „łatwy w regulacji” – zmiany obiektu muszą być powolne aby zapewnić możliwość nadążenia za nimi

- Funkcja celu nie powinna posiadać wielu minimów lokalnych (przejście od dowolnego regulatora początkowego do położenia optymalnego powinno być możliwe na drodze ciągłych niewielkich poprawek)

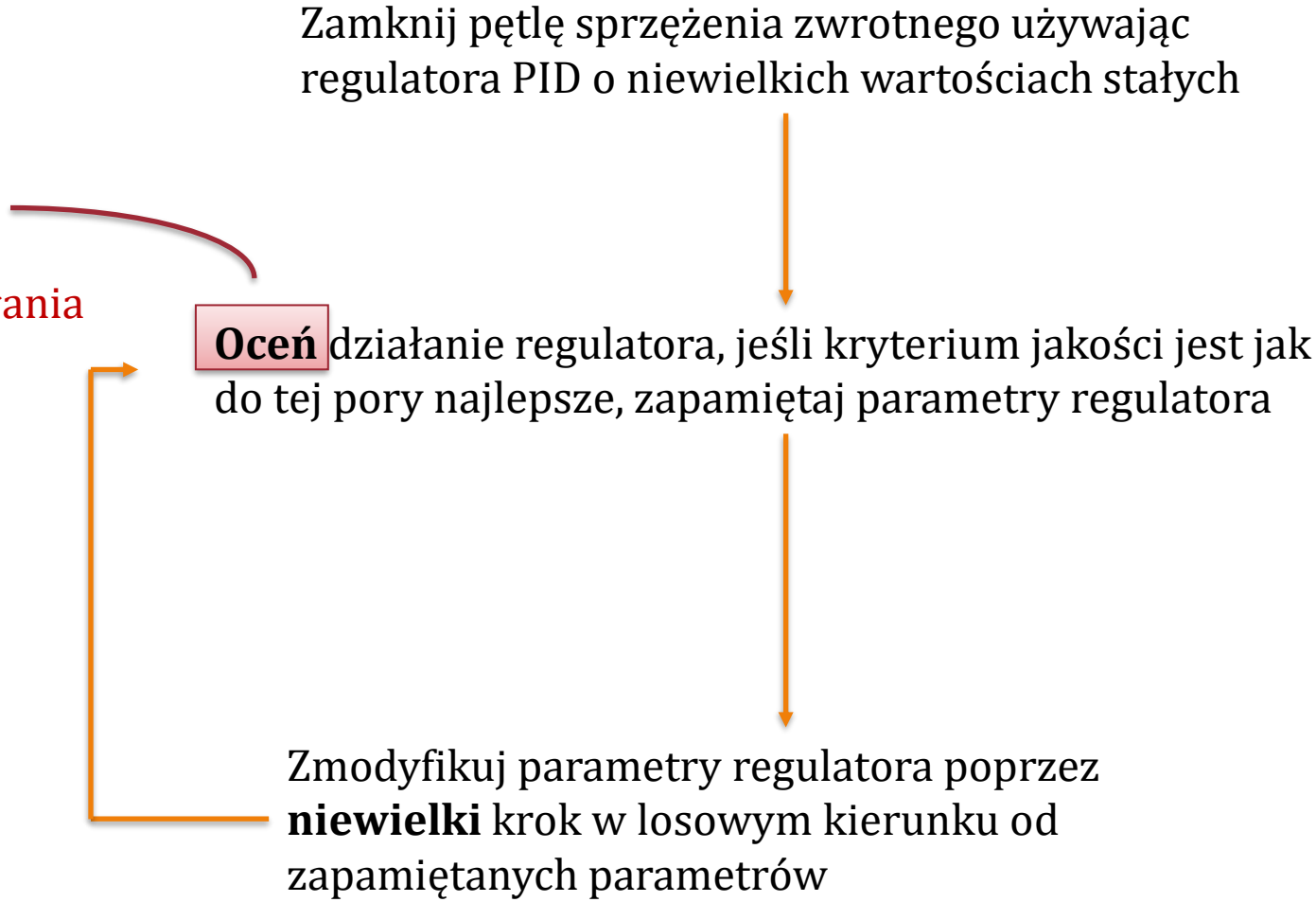
- Obiekt musi być stabilny (tj. odłączenie regulatora nie powinno stanowić zagrożenia dla obiektu)

Rozwiązanie na bazie algorytmu 1+1

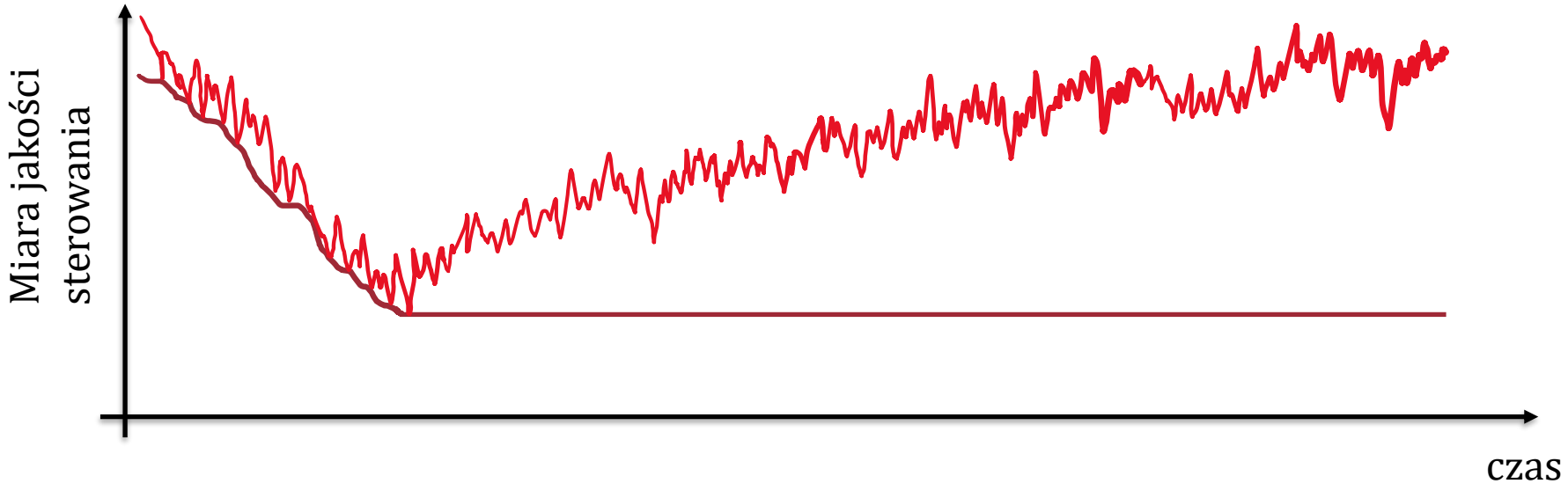
Idealnie: poprzez testowy, znany sygnał (zawsze ten sam)

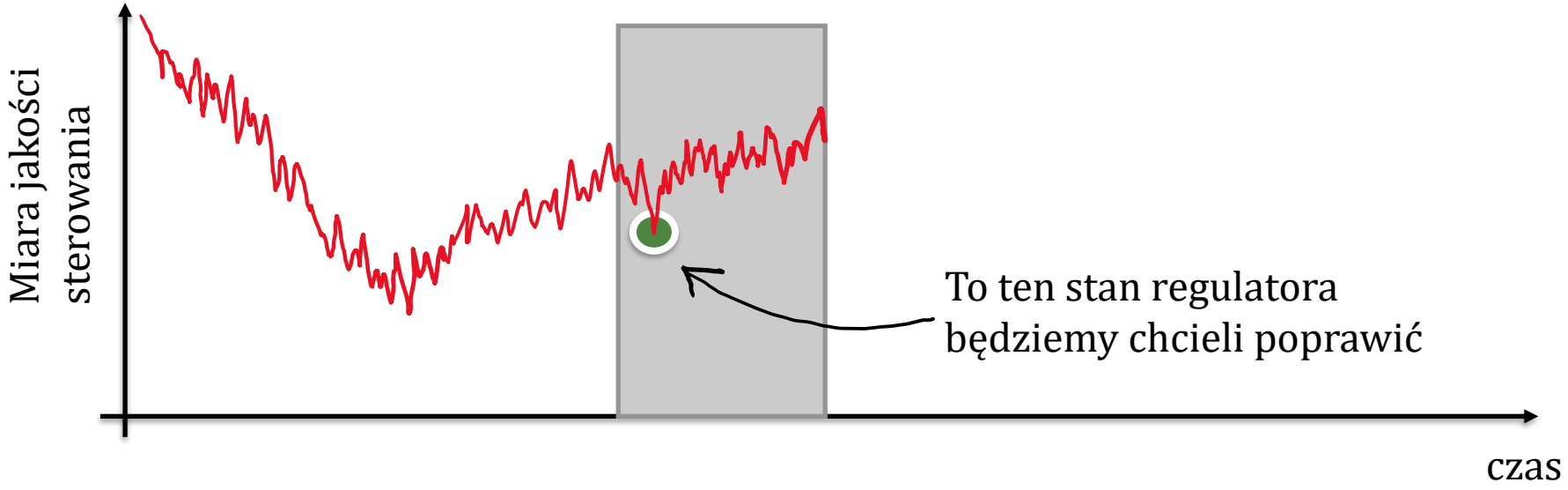
Dobrze: poprzez porównanie sygnałów wejściowych do wyjściowych

Gorzej: poprzez uśrednienie działania systemu w długim czasie



Rozwiązanie na bazie algorytmu 1+1





Rozwiązanie na bazie algorytmu 1+1

Idealnie: poprzez testowy, znany sygnał (zawsze ten sam)

Dobrze: poprzez porównanie sygnałów wejściowych do wyjściowych

Gorzej: poprzez uśrednienie działania systemu w długim czasie

Rozwiązanie na bazie algorytmu 1+1

+ Algorytm 1+1 (w przeciwieństwie do pełnego AG) zapewnia na tyle szybką konwergencję, że w korzystnych warunkach może nadążyć za zmianami obiektu

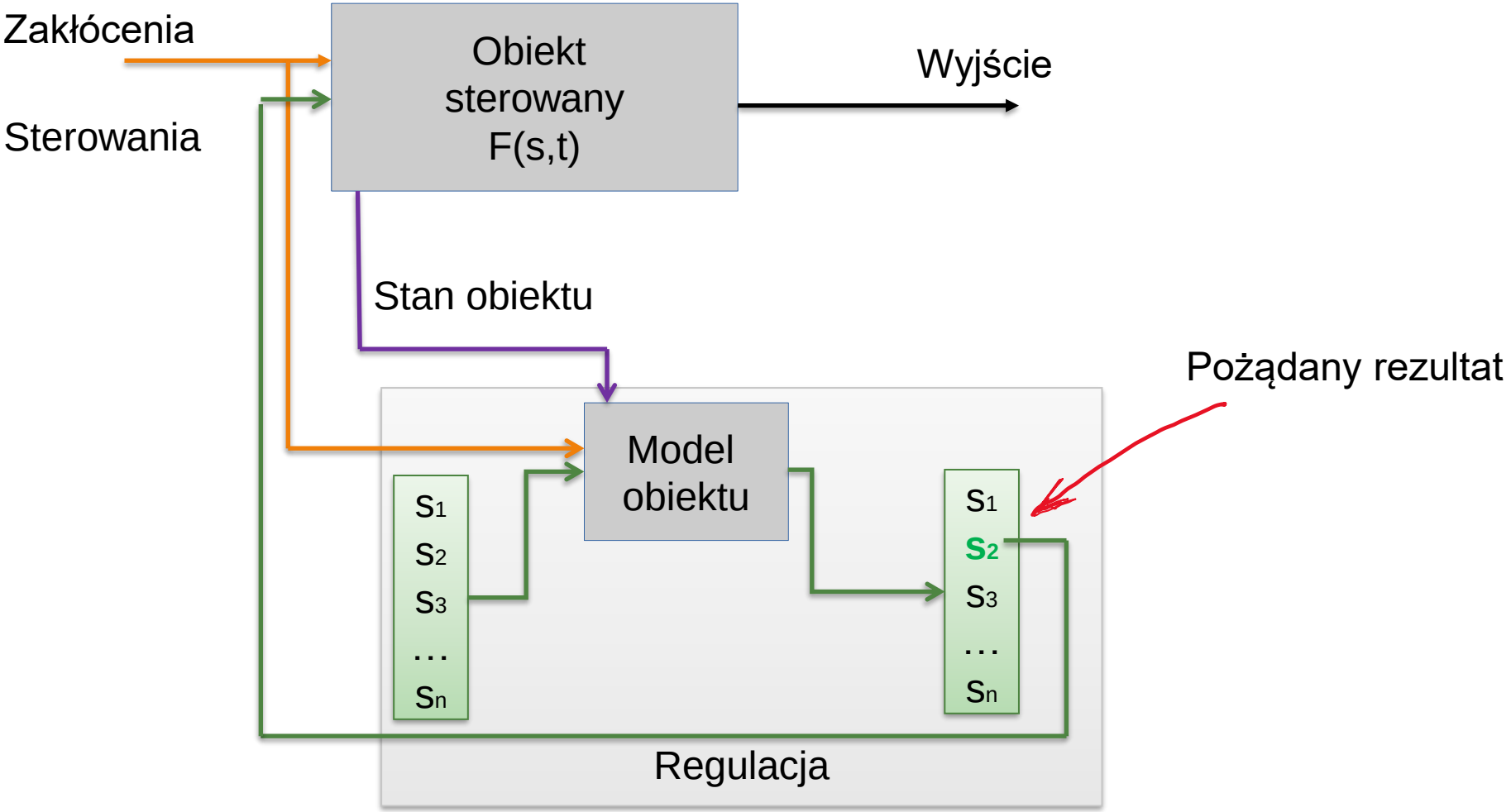
ALE:

- Konieczność uśredniania wskazań (jeśli nie mamy możliwości pełnego monitorowania wejść) powoduje znaczne spowolnienie konwergencji

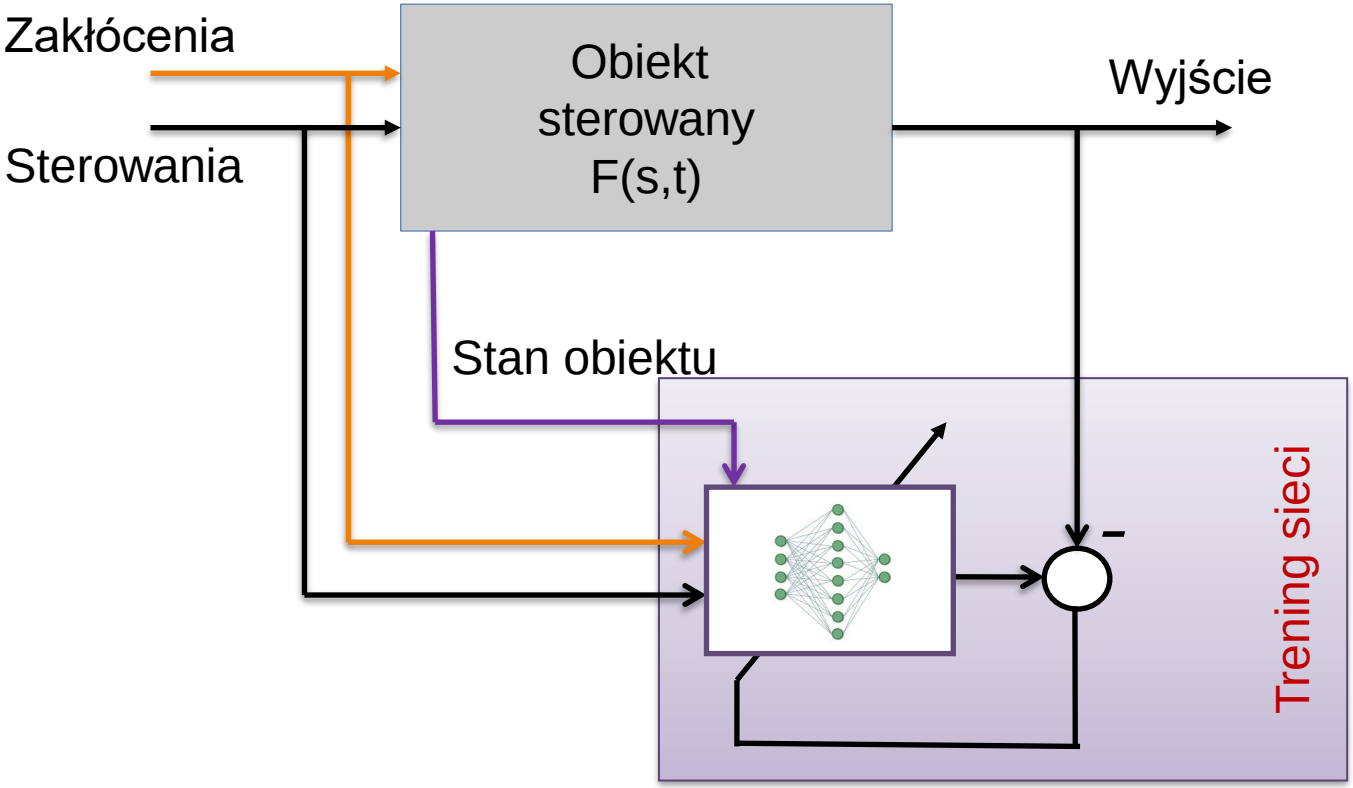
- Im szybciej obiekt się zmienia, tym bardziej trzeba skrócić okno z którego wybierane jest rozwiązanie do modyfikacji: możliwe dalsze spowolnienie konwergencji

Neuronowe sterowanie predykcyjne

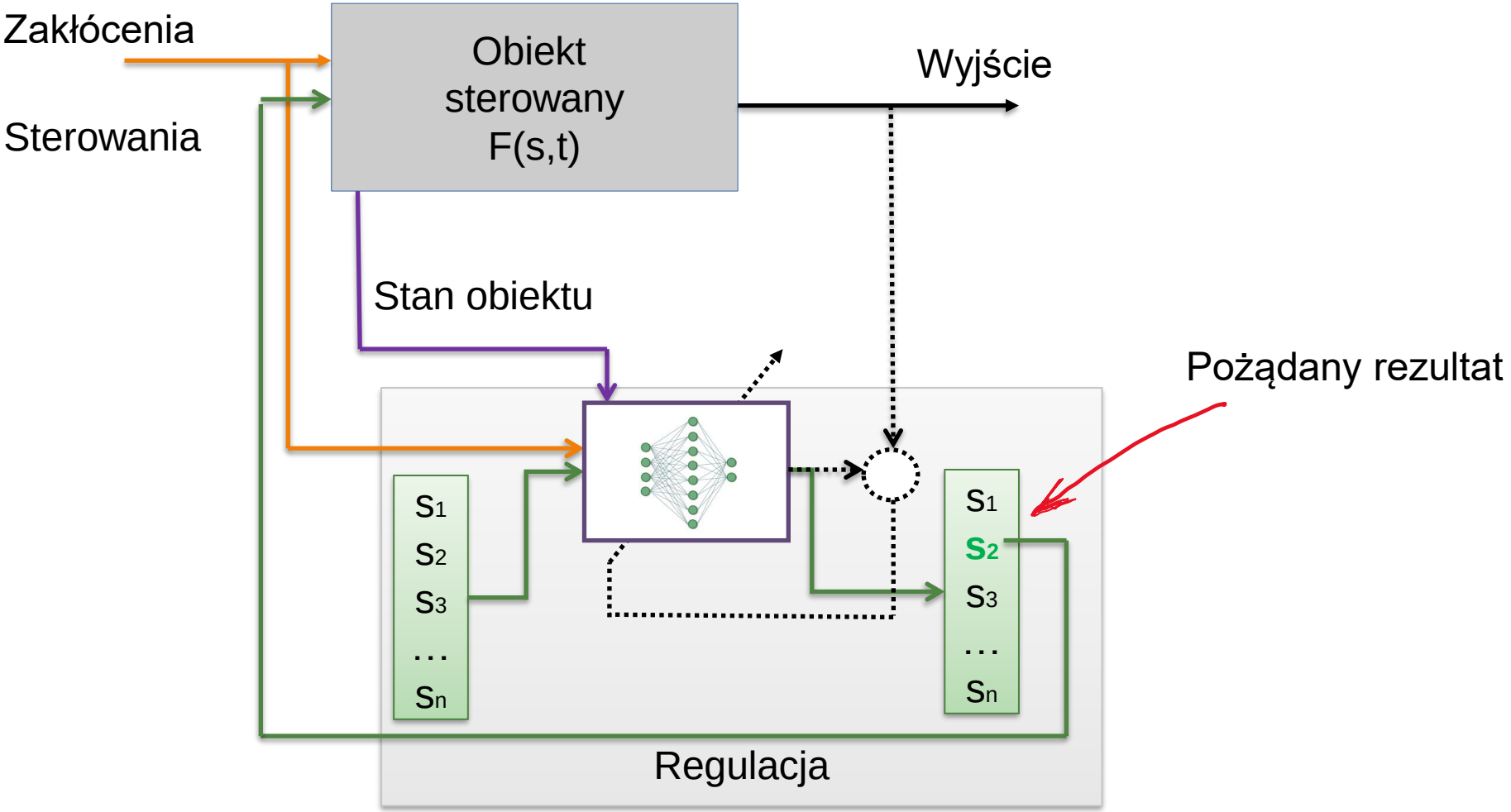
Ogólna koncepcja sterowania predykcyjnego:



Neuronowe sterowanie predykcyjne



Neuronowe sterowanie predykcyjne



Neuronowe sterowanie predykcyjne

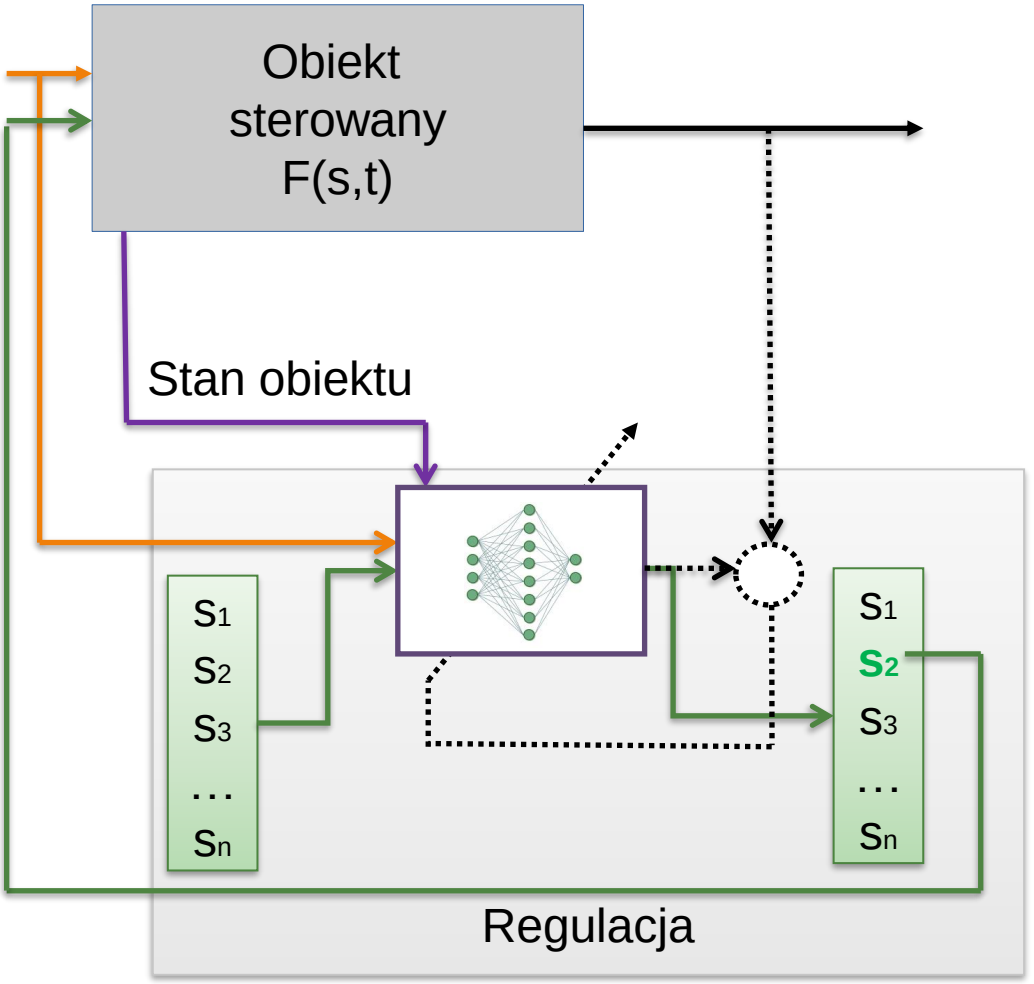
1. Uczymy sieć neuronową do przewidywania zachowania układu w zależności od obserwowanych sterowań, pozostałych wejść oraz stanu układu

2. Podajemy do sieci neuronowej różne możliwe sterowania wybierając ostatecznie to, które minimalizuje wybrane kryterium jakościowe dla danego stanu układu oraz pozostałych wejść

3. Używamy wybranego sterowania w rzeczywistym układzie

4. Co jakiś czas dotrenowujemy sieć na podstawie nowo zebranych danych

Neuronowe sterowanie predykcyjne



+ Szybka adaptacja

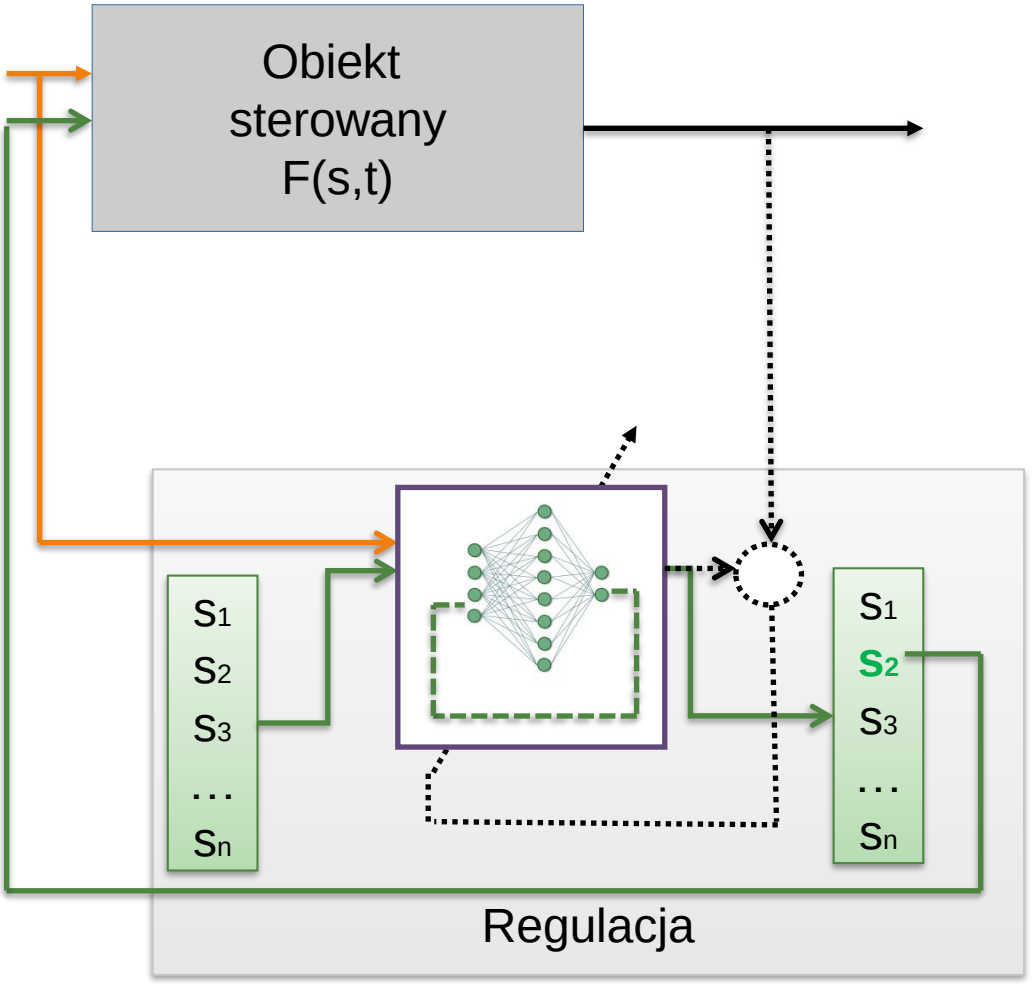
+ Duża skuteczność w przypadku systemów nieliniowych

- Konieczność monitorowania stanu obiektu

- Brak gwarancji stabilności układu

- Brak możliwości analizy zależności długoterminowych

Zastosowanie sieci rekurencyjnych



+ Szybka adaptacja

+ Duża skuteczność w przypadku systemów nieliniowych

~~- Konieczność monitorowania stanu obiektu~~

- Brak gwarancji stabilności układu

~~- Brak możliwości analizy zależności długoterminowych~~

Teoria Sterowania

Sterowanie inteligentne

1) **Jak działa genetyczna identyfikacja układu sterowania?**

1) **Jak dostosować działanie AG do optymalizacji problemów zmiennych w czasie (optymalizacji adaptacyjnej)?**

1) **Jak działa adaptacyjna optymalizacja sterownika PID (np. z zastosowaniem metody 1+1?**

1) **Jak działa neuronowe sterowanie predykcyjne?**