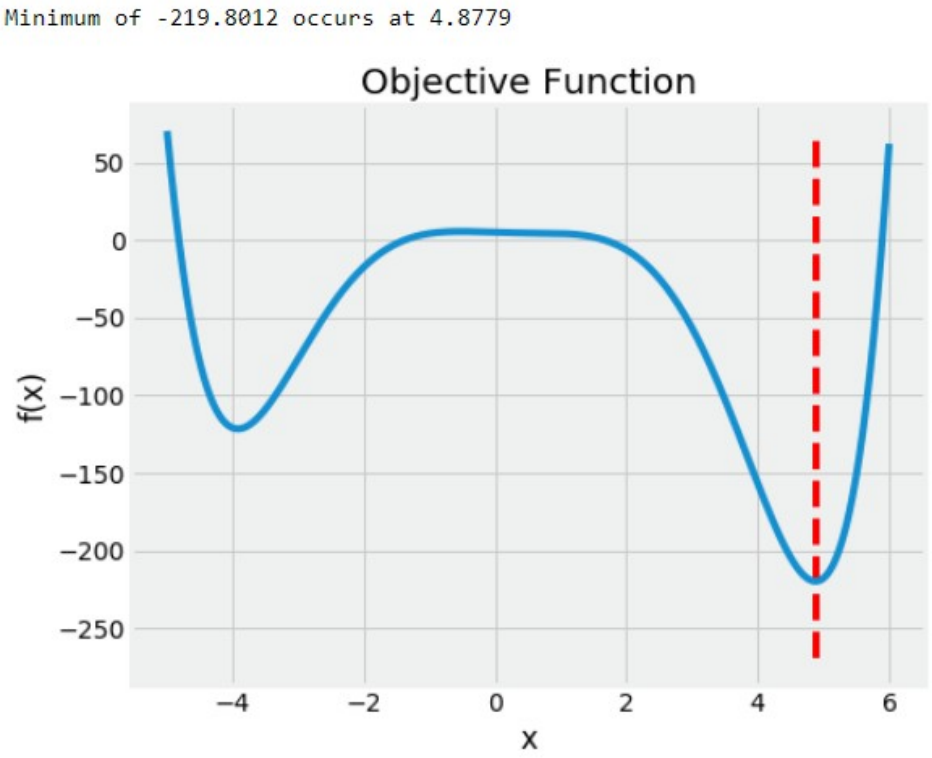
Deep learning
wykład 3



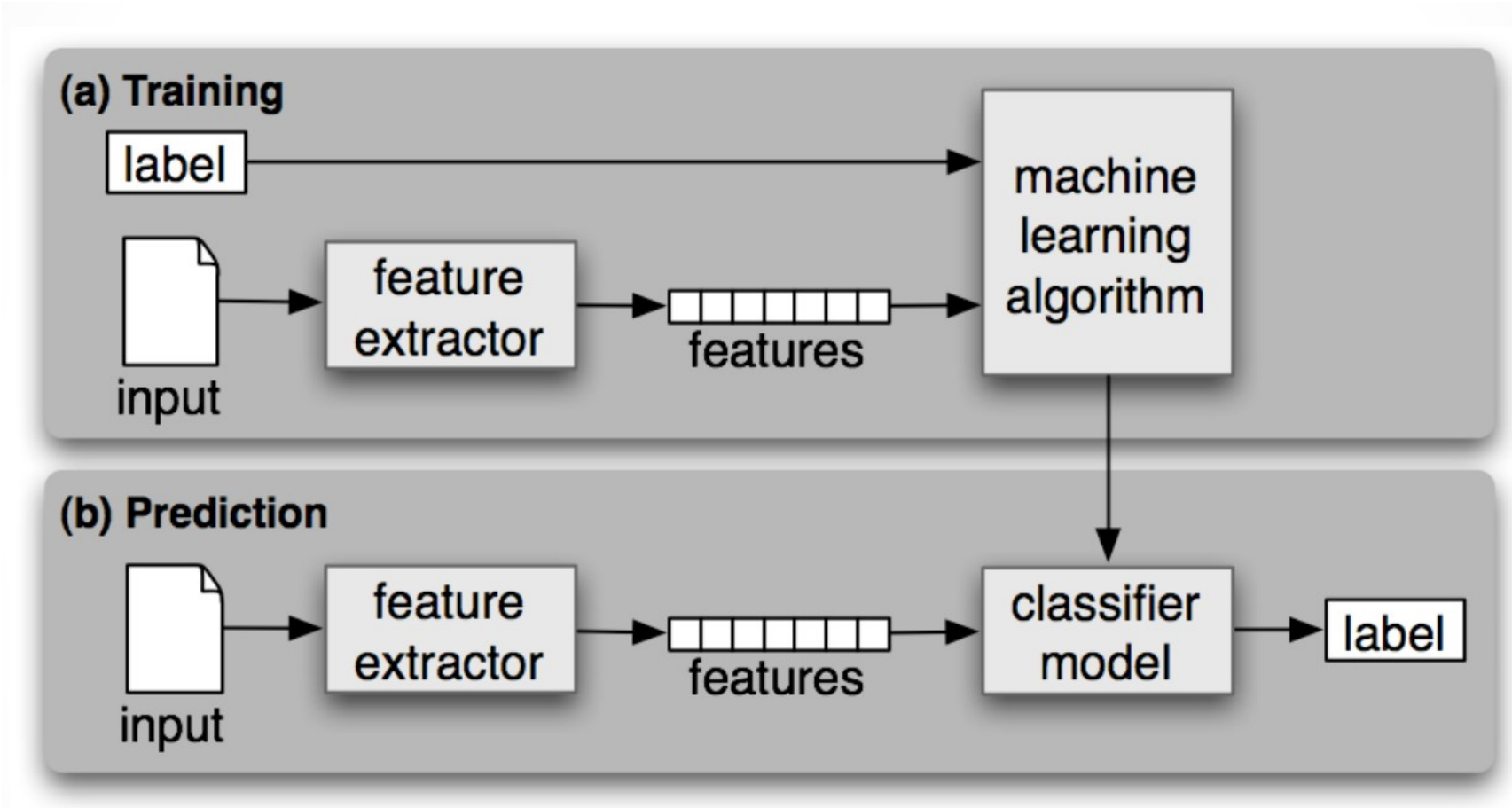
- Optymalizacja hiperparametrów

Marcin Wolter

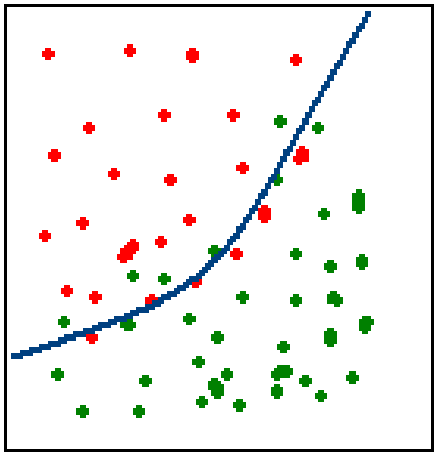
IFJ PAN

2 grudnia 2020

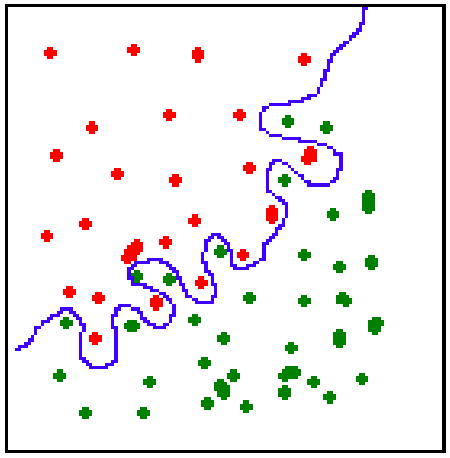
Trening nadzorowany



Przeuczenie (Overtraining)

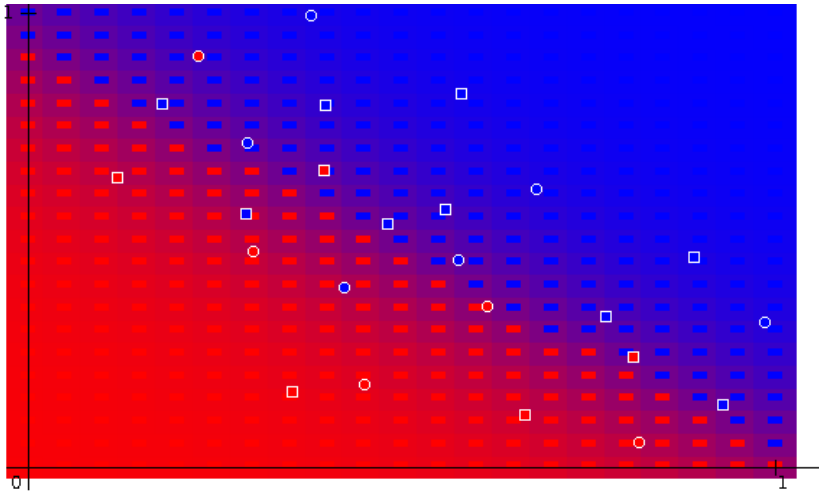


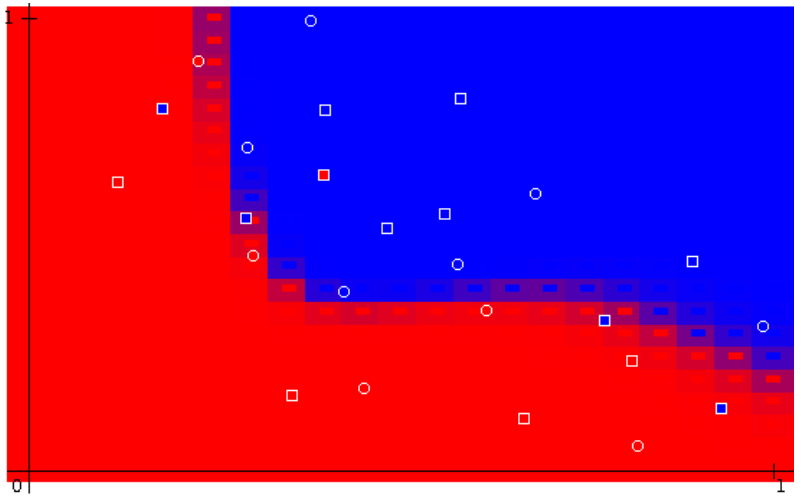
Dobrze

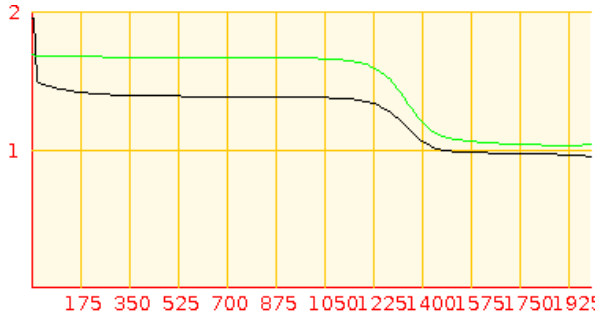


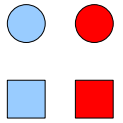
Przetrenowanie

- **Przeuczenie** – Algorytmuczy się poszczególnych przypadków, a nie ogólnych zasad.
- Efekt pojawia się we wszystkich algorytmach uczących
- Remedium – sprawdzenie na oddzielnym zbiorze danych.



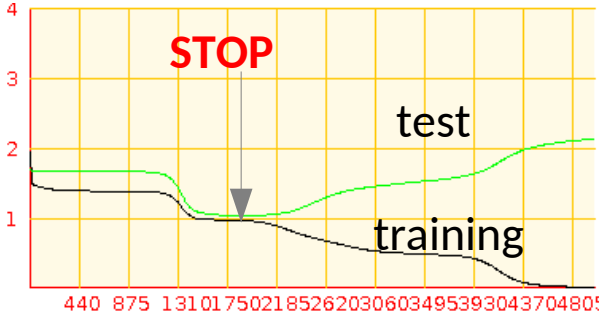






Training sample

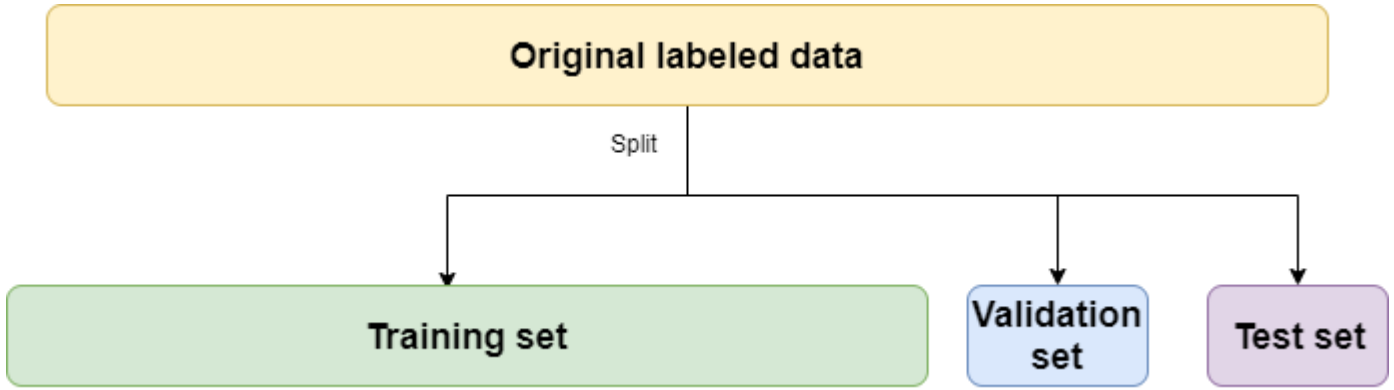
Test sample



Przykład z siecią neuronową.

Jak ternować sieć neuronową

- Jak zapobiegać **przeuczeniu** (**overtraining)**?
- Zbiory **treningowy** i **walidacyjny**
- **Uwaga:** aby zapobiegać przeuczeniu powinniśmy podzielić dane na trzy podzbiory: treningowy, walidacyjny i testowy:
  - Treningowy – do treningu
  - Walidacyjny – sprawdzamy w trakcie treningu działanie sieci. Na tej podstawie dobieramy hiperparametry (np. liczba warstw, liczba neuronów, funkcja aktywacji itp.)
  - Testowy – sprawdzamy jak działa wytrenowany algorytm



Walidacja

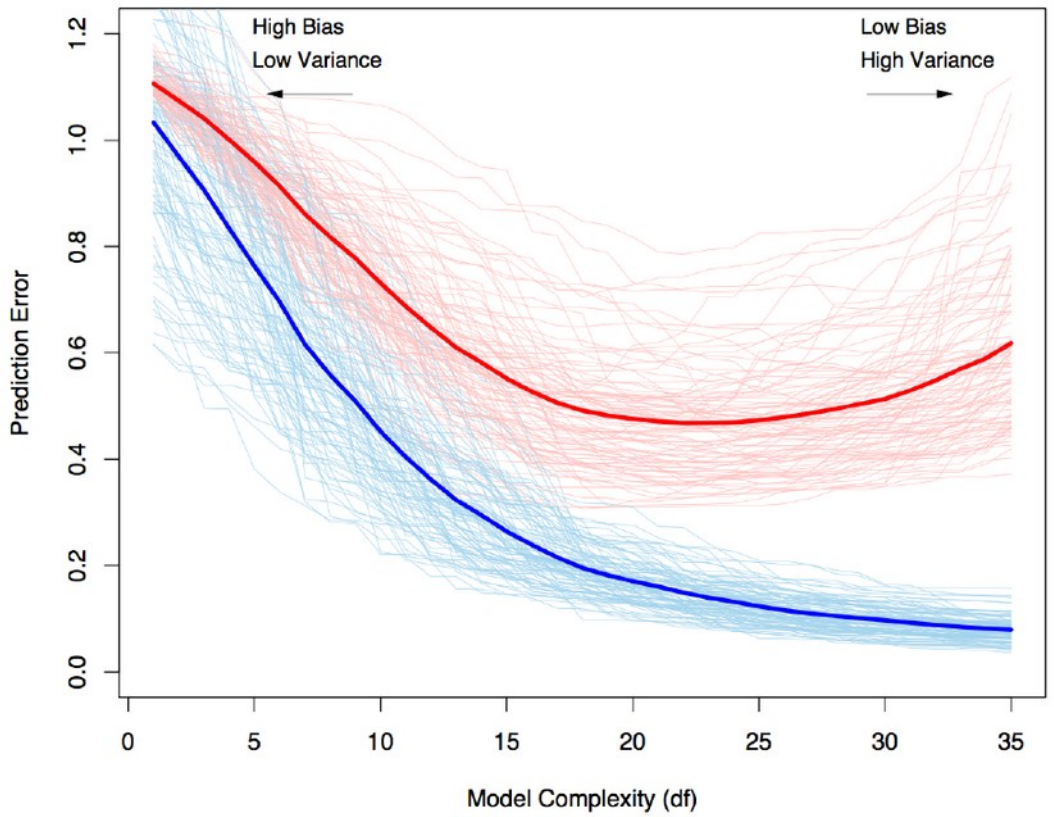


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

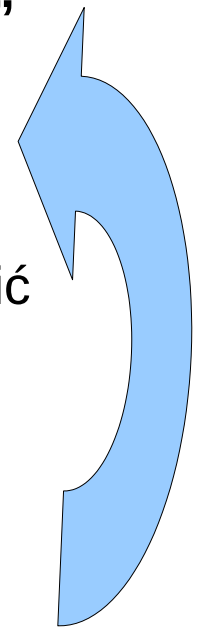
Source: Elements of Statistical Learning

Optymalizacja hiperparametrów

- Prawie każda metoda uczenia maszynowego ma jakieś swobodne hiperparametry (struktura sieci neuronowej, liczba drzew i ich głębokość w BDT itp).
- Musimy je zoptymalizować dla naszego problemu
- **Zadanie: dla danego zbioru danych znaleźć zestaw hiperparametrów taki, aby zminimalizować błąd.**
- Wygląda jak typowy problem minimalizacji, ale:
  - Każdy pomiar jest kosztowny (CPU!!!)
  - Duży szum
  - Możemy policzyć wartość minimalizowanej funkcji (czyli nasz błąd), ale nie potrafimy łatwo znaleźć pochodnej.

Optymalizacja

- Jak optymalizować:
  - „Grid search” – skanowanie po wszystkich możliwych wartościach parametrów
  - „Random search” – podobnie...
  - Jakiś sposób znajdowania minimum
- Popularna metoda – optymalizacja bayesowska **„bayesian optimization”**
  - "A priori,, model prawdopodobieństwa
  - Weź „a priori” początkowe wartości parametrów
  - Znajdź, dla jakiego punktu w przestrzeni parametrów możesz poprawić swój model
  - Znajdź błąd dla tego punktu
  - Znajdź „a posteriori” rozkład prawdopodobieństwa
  - Powtarzaj



Jak to działa w praktyce?

- Dopasowanie prostej

  $y(x, w) = w_0 + w_1 x$ fit do danych.

1) Gaussowski prior, nie znamy jeszcze danych

2) Pierwszy punkt z danych. Znajdujemy na tej podstawie prawdopodobieństwo (rysunek po lewej) i mnożymy: prior*likelihood. Otrzymujemy prawdopodobieństwo "a posteriori" (po prawej).

3) Dodajemy drugi punkt i powtarzamy procedurę.

4) Dodajemy jeden po drugim punkty z danych.

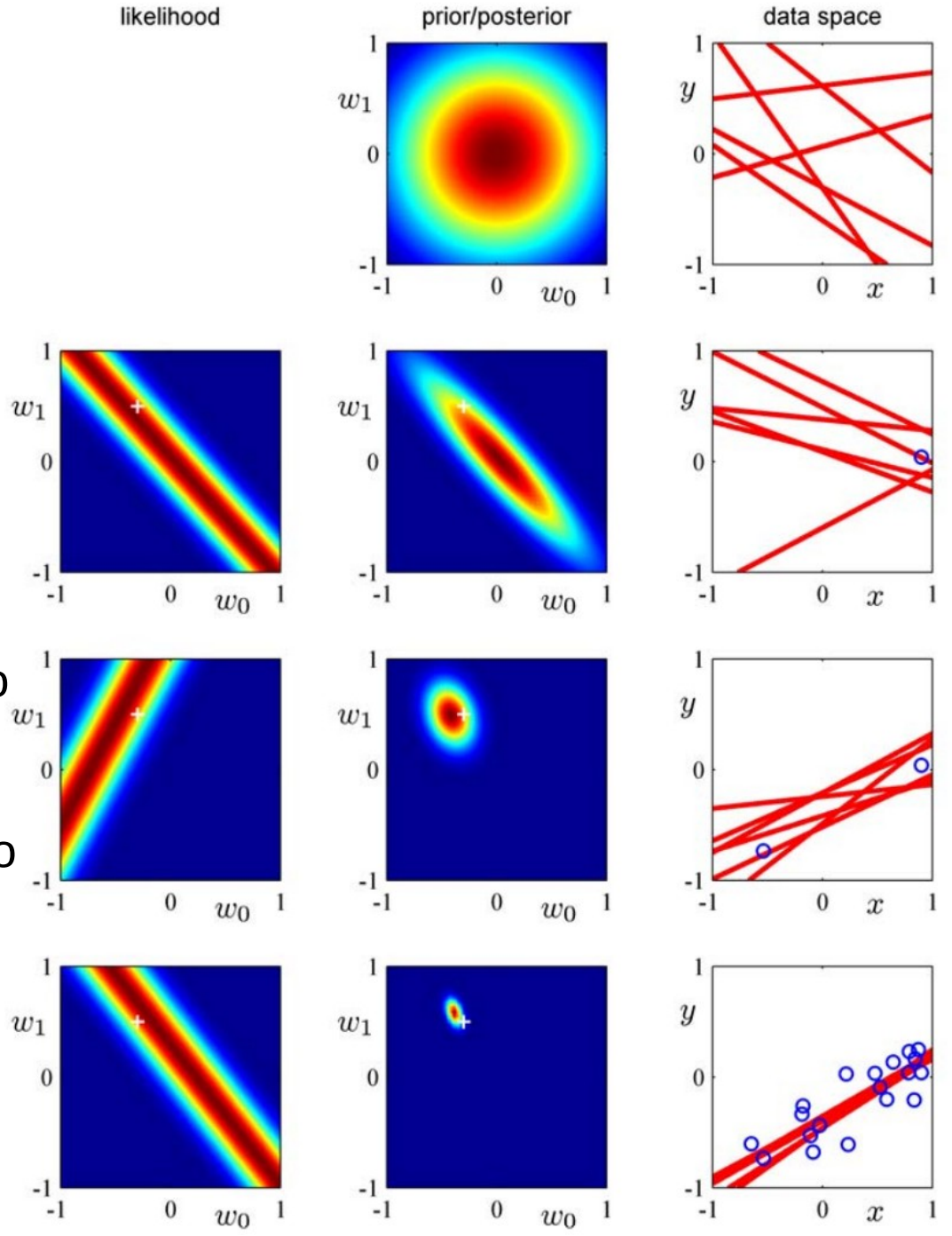
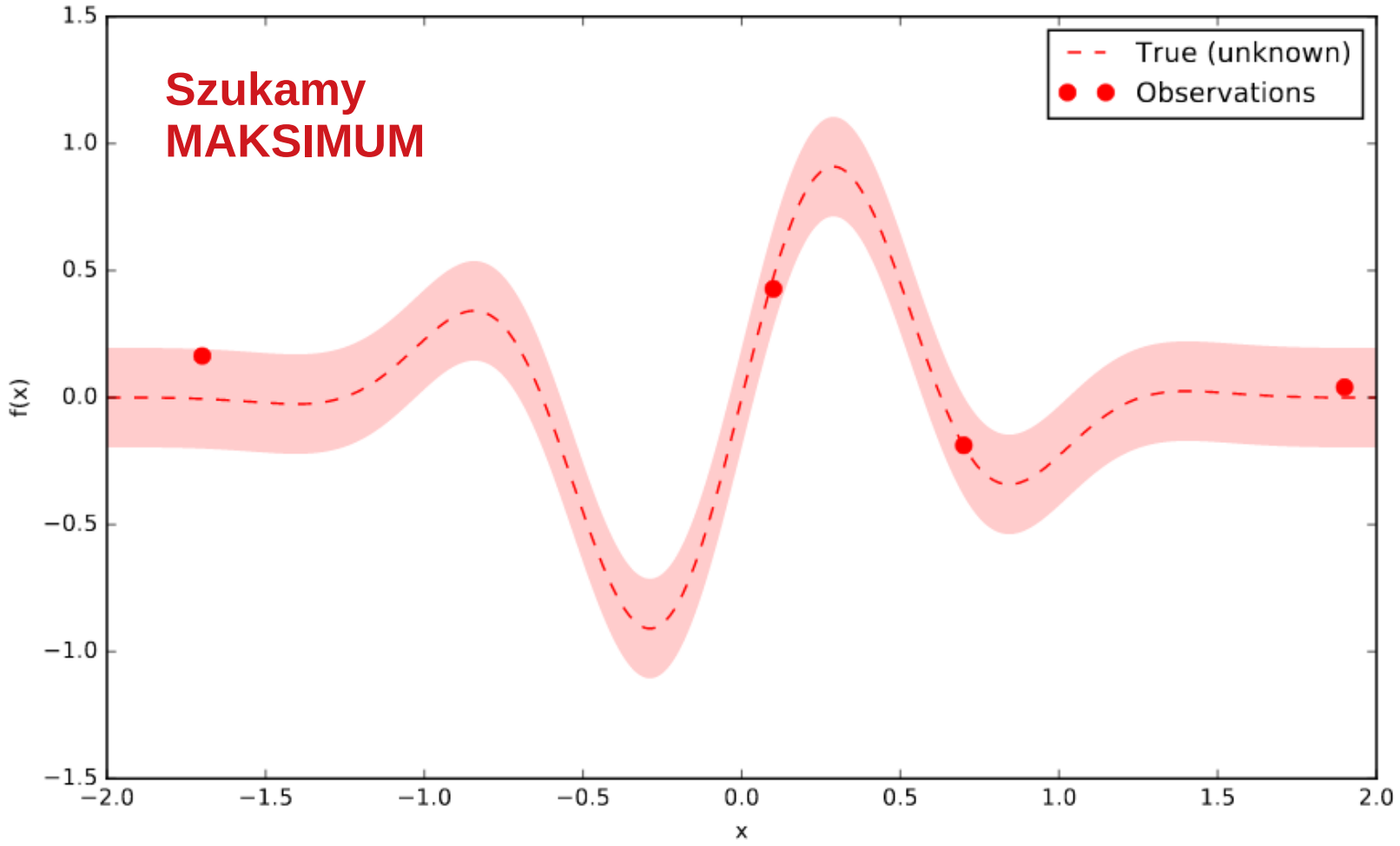


Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. A detailed description of this figure is given in the text.

Punkt startowy

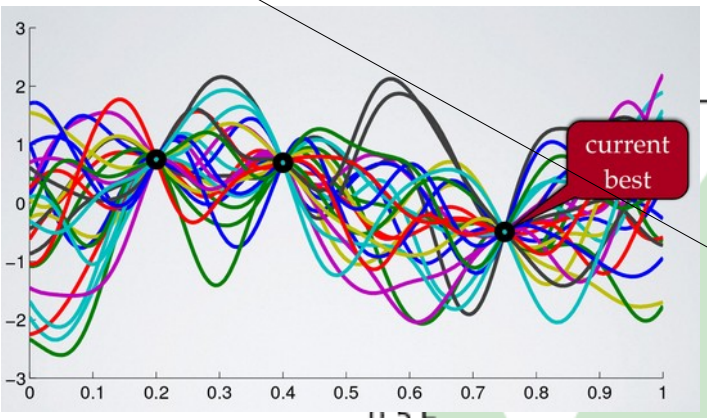


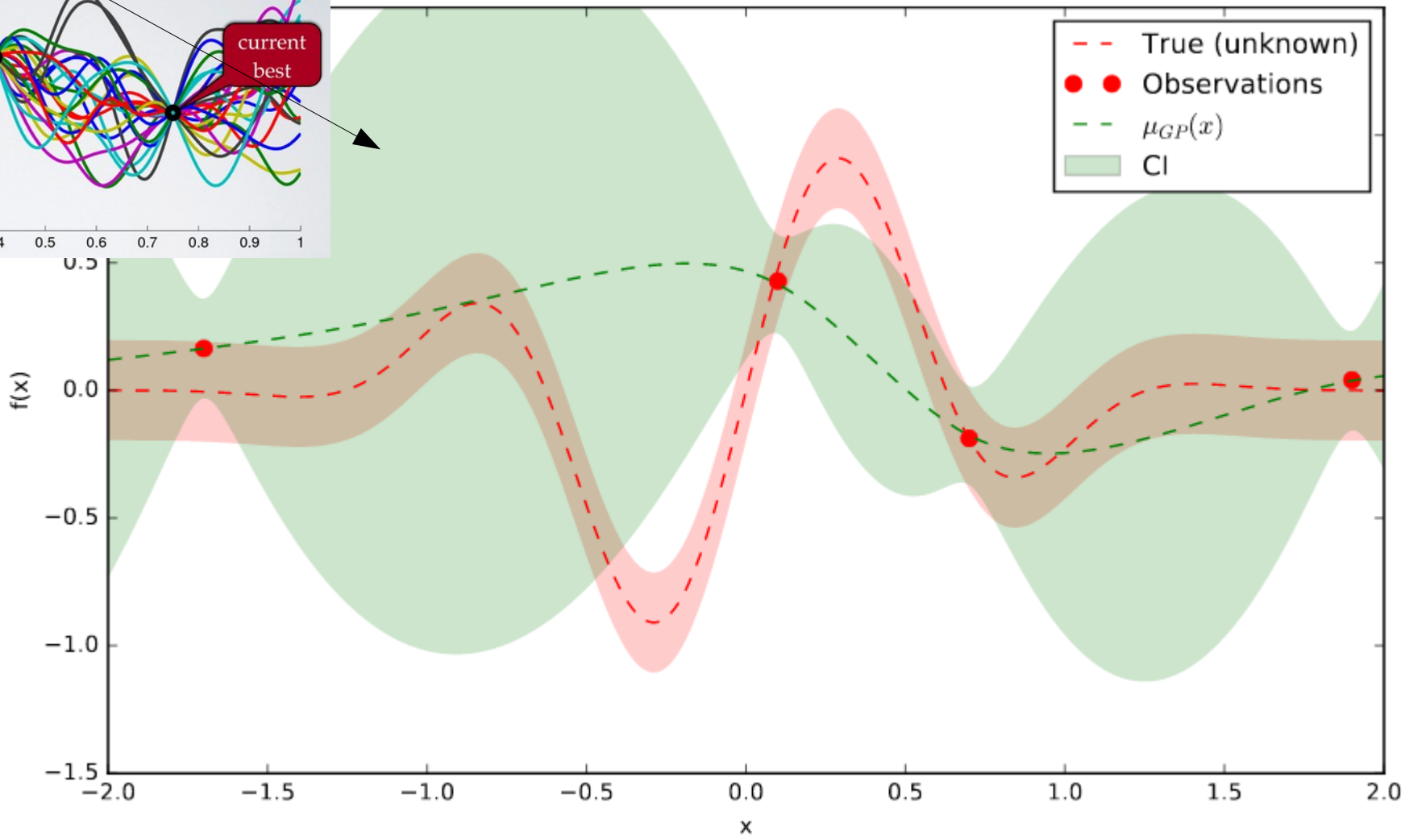
Nieznana funkcja (z szumem), cztery pomiary. Gdzie dokonać następnego, kosztownego pomiaru?

https://www.iro.umontreal.ca/~bengioy/cifar/NCAP2014-summerschool/slides/Ryan_adams_140814_bayesopt_ncap.pdf

Wiele funkcji

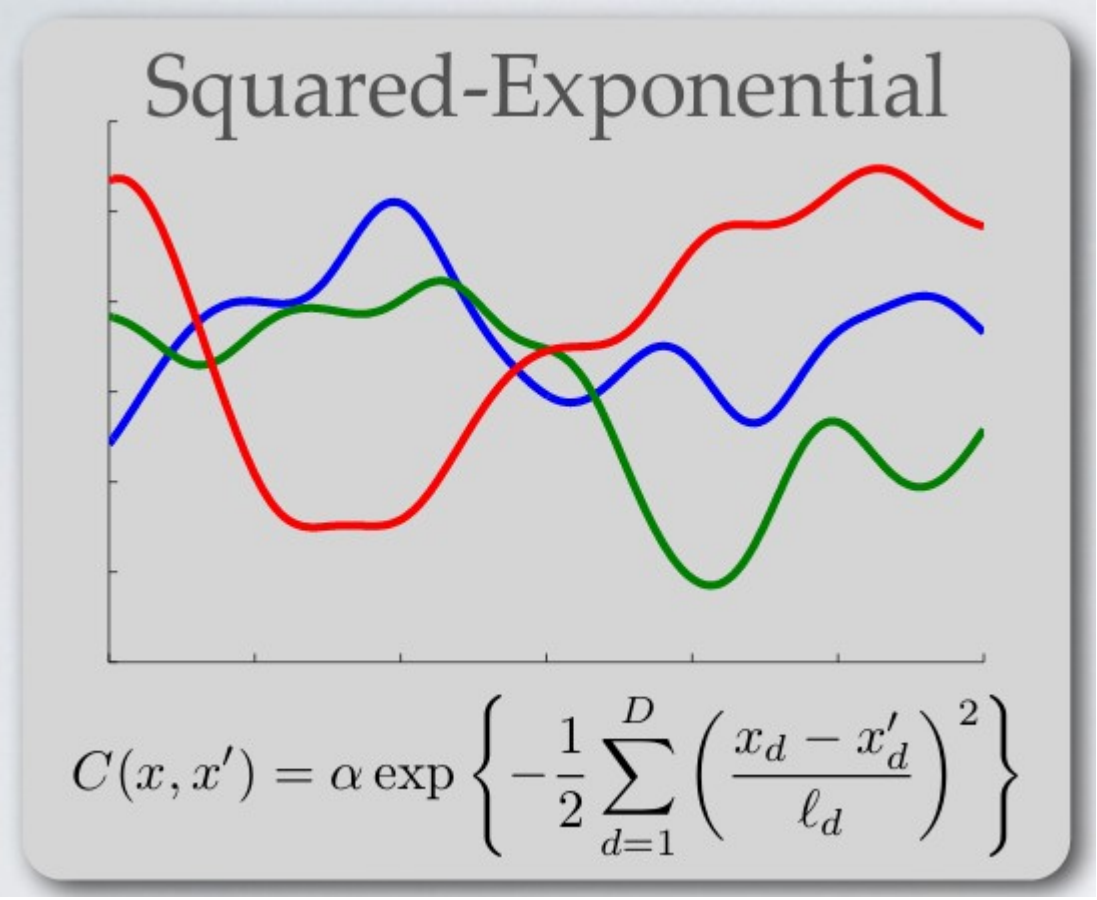
Rozkład „a posteriori”





Rozkład możliwych funkcji "a posteriori". Te funkcje mogą generować obserwowane punkty.

Funkcje a posteriori – Gaussian Processes (GP)



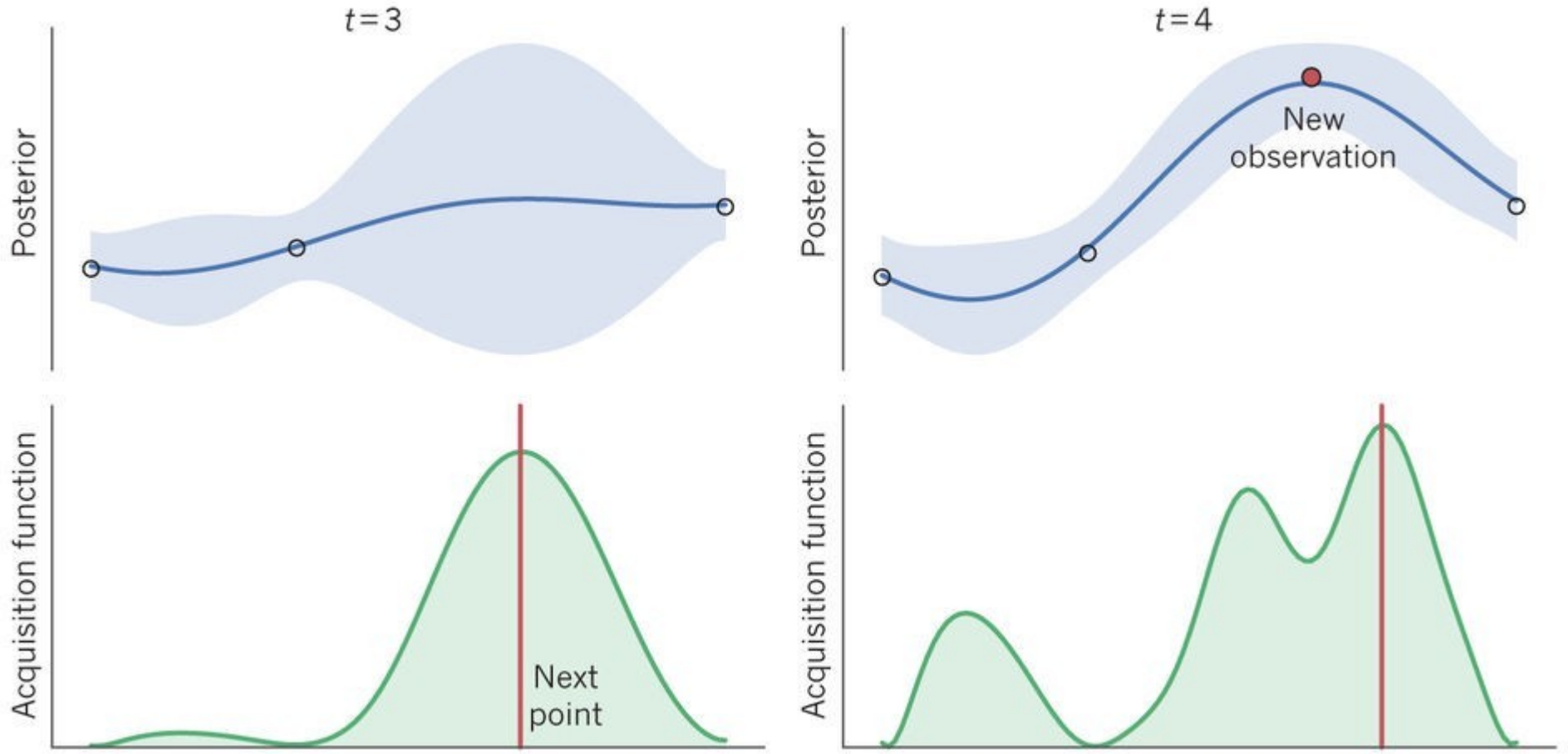
Funkcje musimy jakoś parametryzować, np. Mogą być sumą funkcji Gaussa.

Tzw. "acquisition function”

- A posteriori GP (Gaussian Processes) daje średnią z funkcji $\mu(x)$ i ich oczekiwaną wariację $\sigma^2(x)$.
  - **Exploration** – szukamy gdzie jest największa wariacja
  - **Exploitation** – szukamy największej/najmniejszej wartości średniej $\mu(x)$
- Nasz algorytm musi brać pod uwagę obie te rzeczy

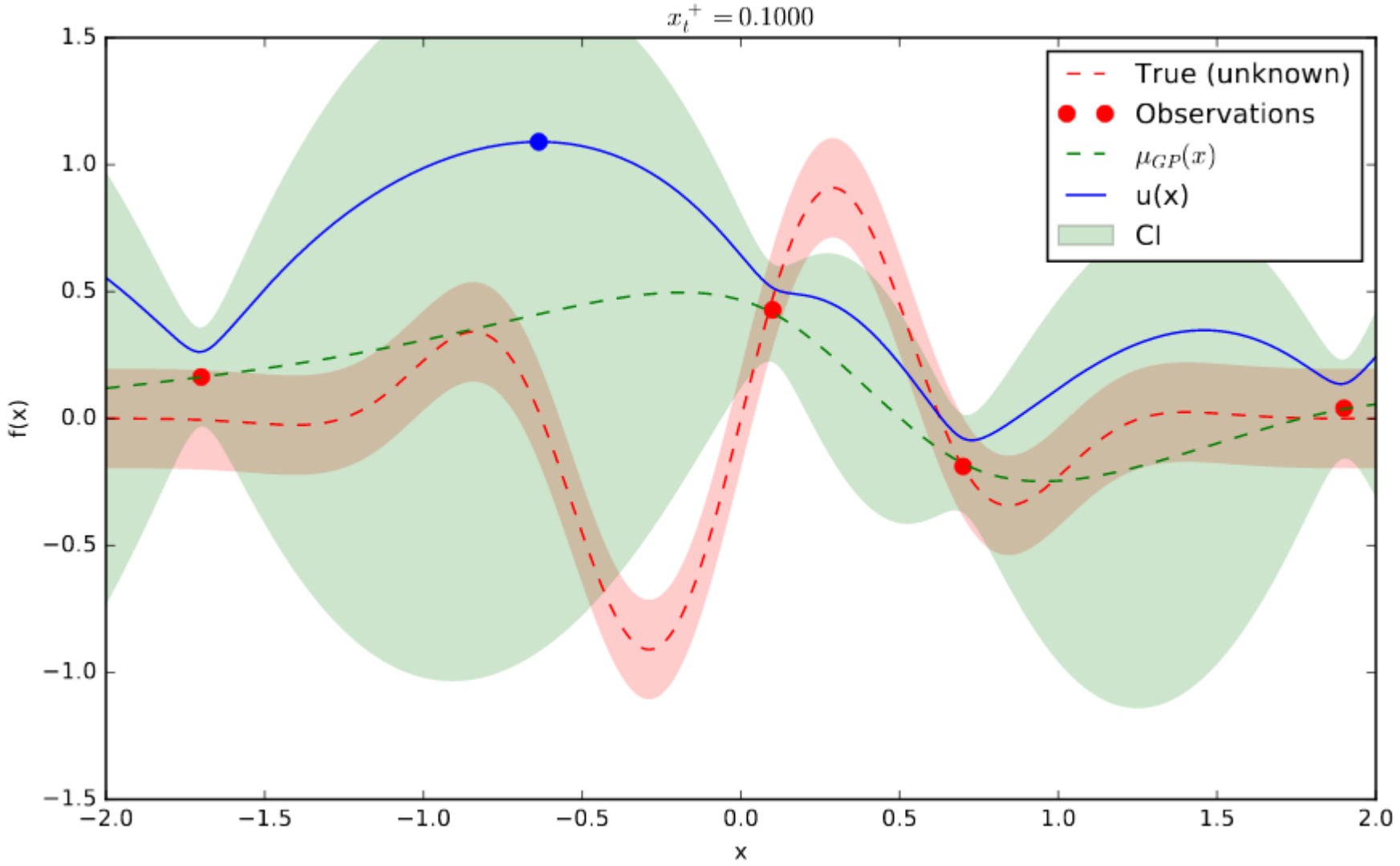
Gdzie następny punkt?

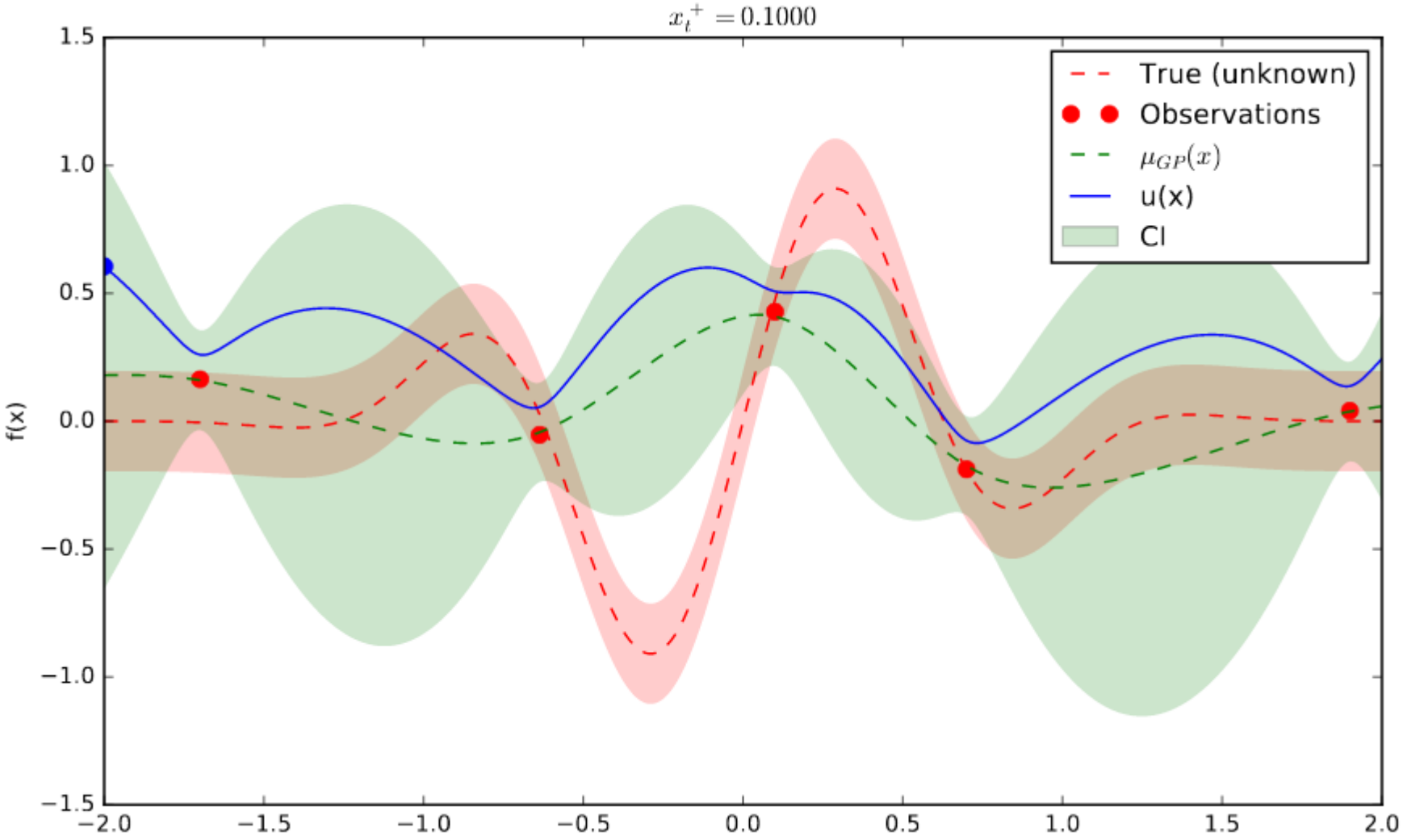
- Nasz nowy punkt point(x) musi być kompromisem pomiędzy wysoką średnią (exploitation) i dużą wariancją (exploration).



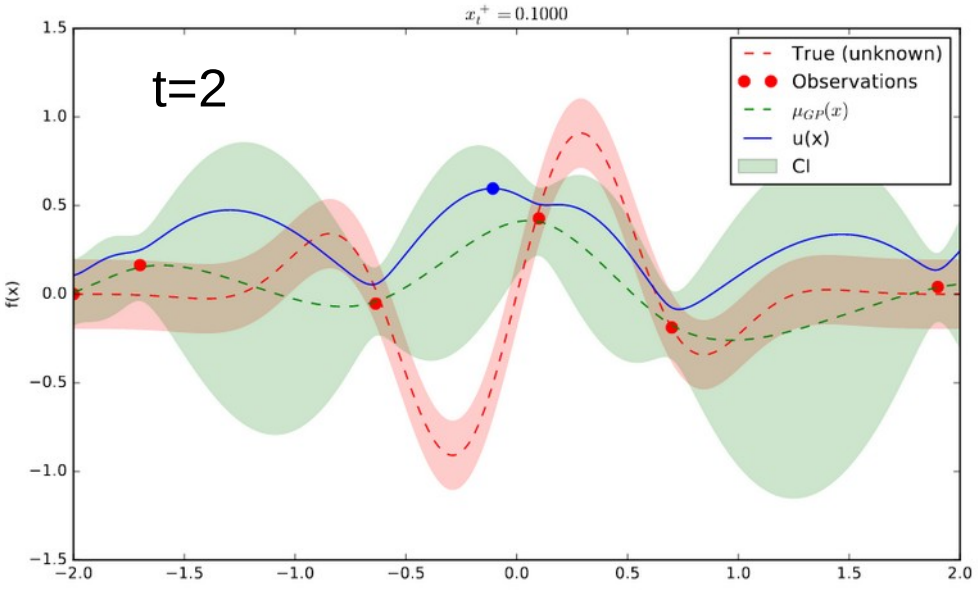
Wybieram następny punkt

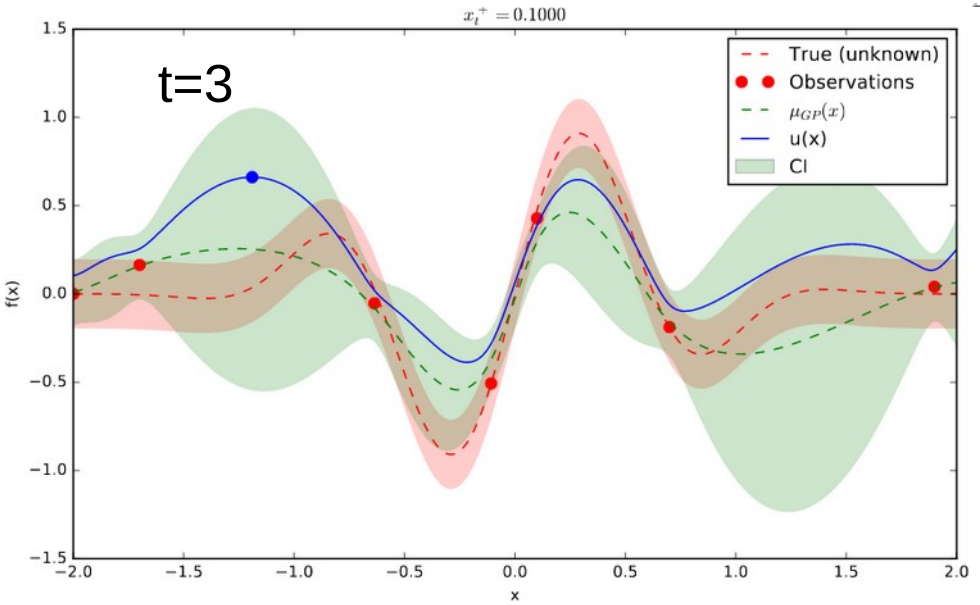
u(x) – acqusition function
(szukamy maksimum)

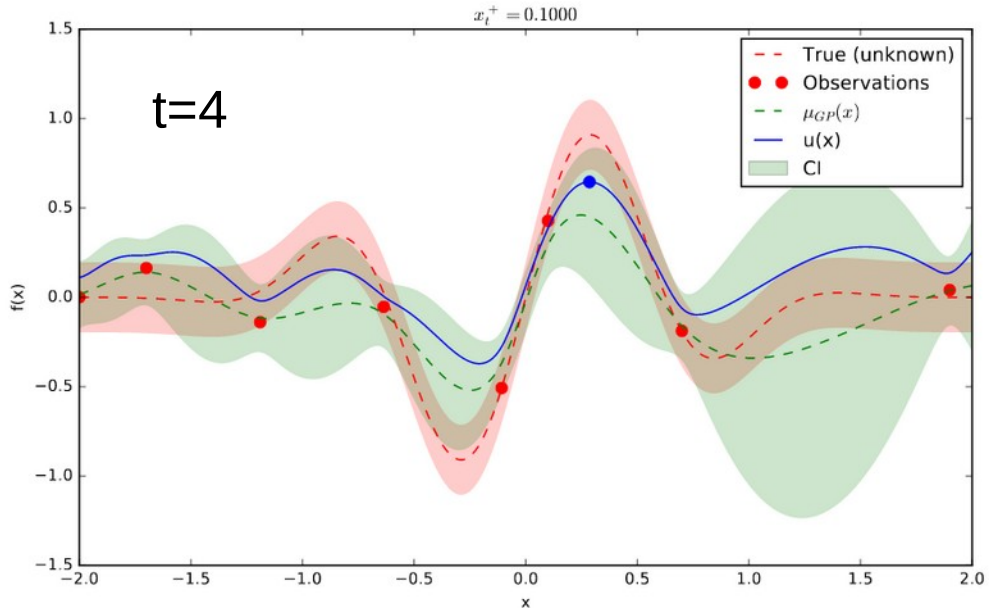


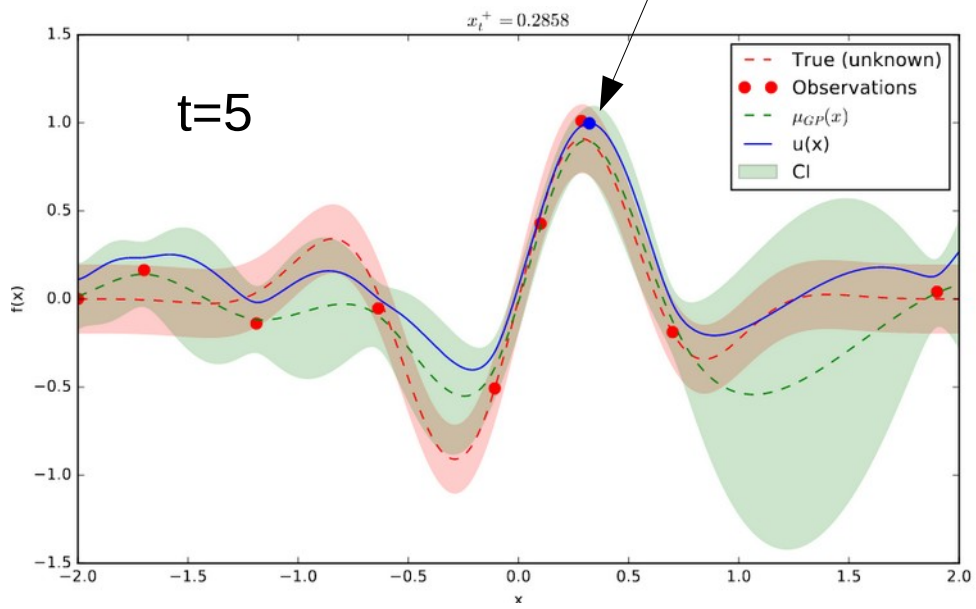


Dokonujemy próbkowania i powtarzamy procedurę...


$x_t^+ = 0.1000$
t=2
True (unknown)
Observations
$\mu_{GP}(x)$
u(x)
CI
f(x)


$x_t^+ = 0.1000$
t=3
True (unknown)
Observations
$\mu_{GP}(x)$
u(x)
CI
f(x)
x


$x_t^+ = 0.1000$
t=4
True (unknown)
Observations
$\mu_{GP}(x)$
u(x)
CI
f(x)
x


MAKSIMUM
$x_t^+ = 0.2858$
t=5
True (unknown)
Observations
$\mu_{GP}(x)$
u(x)
CI
f(x)
x

Ograniczenia

- Optymalizacja bayesowska zależy od wybranych parametrów
  - "acquisition function”
  - Wybranych "priorów"...
- Jest sekwencyjna
- Implementacje:
  - Tree of Parzen Estimators (TPE) używane przez HyperOpt package https://github.com/hyperopt/hyperopt).
  - Pakiet OPTUNA https://optuna.org/ Dość rozbudowany, zawiera algorytmy grid search, random sampling,TPE i Covariance Matrix Adaptation Evolution Strategy CMA-ES
  - Wiele innych...

OPTUNA vs. HYPEROPT comparison:
https://neptune.ai/blog/optuna-vs-hyperopt

Przykłady

- Prosty przykład HYPEROPT & OPTUNA:

  https://github.com/marcinwolter/DeepLearning_2020/blob/main/hyperopt_optuna_demo.ipynb

- Optymalizacja czytania liter MNIST z HYPEROPT oraz OPTUNA:

  https://github.com/marcinwolter/DeepLearning_2020/blob/main/mnist_mlp_minimal_hyperopt.ipynb

  https://github.com/marcinwolter/DeepLearning_2020/blob/main/mnist_mlp_minimal_hyperopt.ipynb

Artykuły

- Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599.
- Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. (2016). Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE, 104(1):148–175.
- Nice tutorial:

  https://www.iro.umontreal.ca/~bengioy/cifar/NCAP2014-summerschool/slides/Ryan_adams_140814_bayesopt_ncap.pdf