MI
Metody Identyfikacji

wykład #9

1. *Identyfikacja charakterystyk statycznych*
2. *Regresja liniowa*
3. *Regresja nieliniowa*
4. *Wygładzanie*
5. *Estymatory jądrowe – kernel estimation*
6. *Zadanie klasyfikacji a zadanie interpolacji*

Wstęp

- O co walczymy dokąd zmierzamy? – różne hasła → te same metody
  - Machine Learning
  - Data Mining
  - Pattern Recognition
  - Data Analysis
  - Statistics

- Automatyczna identyfikacja nieprzypadkowych struktur w danych

- Czego oczekujemy:
  - Algorytmu odpornego na wartości poboczne (odległe) i błędne założenia odnośnie modelu.
  - Algorytmu stabilnego: dobra generalizacja w przypadku nowych danych.
  - Algorytmu wydajnego obliczeniowo: wiele danych.

Motywacja

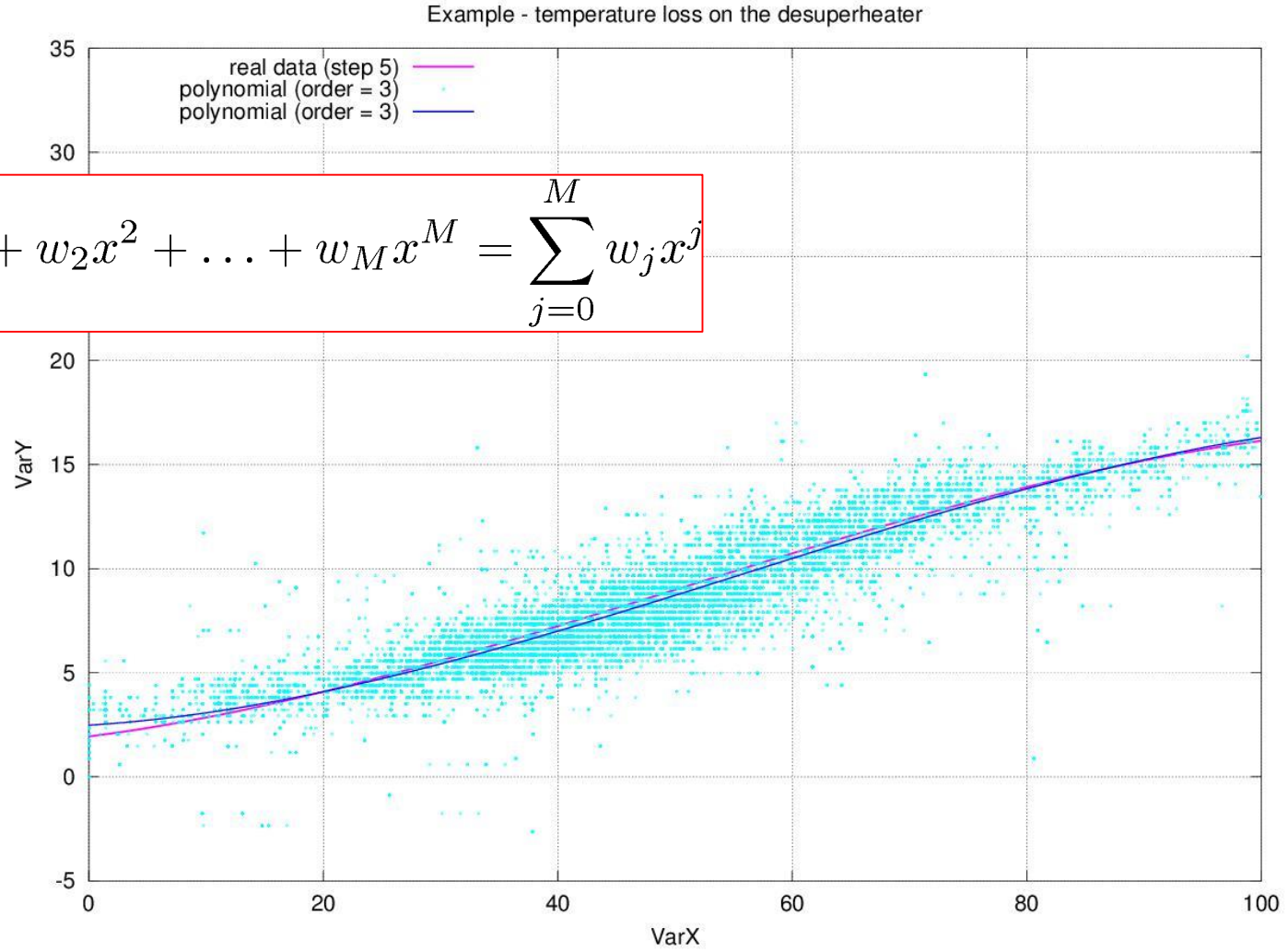
- Wszechstronność w analizie pojawiających się w danych zależności
- Możliwość predykcji zachowania bez konieczności budowania i identyfikacji modelu parametrycznego
- Jak wykryć błędne zachowanie / informację / daną analizując lokalne punkty
- Elastyczność w odtwarzaniu brakujących danych oraz w zadaniu interpolacji

Zakres

- … zaczynamy od **regresji liniowej**
- Regresja nieliniowa
- Wygładzanie
- Estymacja jądrowa

Regresja liniowa – dopasowanie funkcji wielomianowej

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

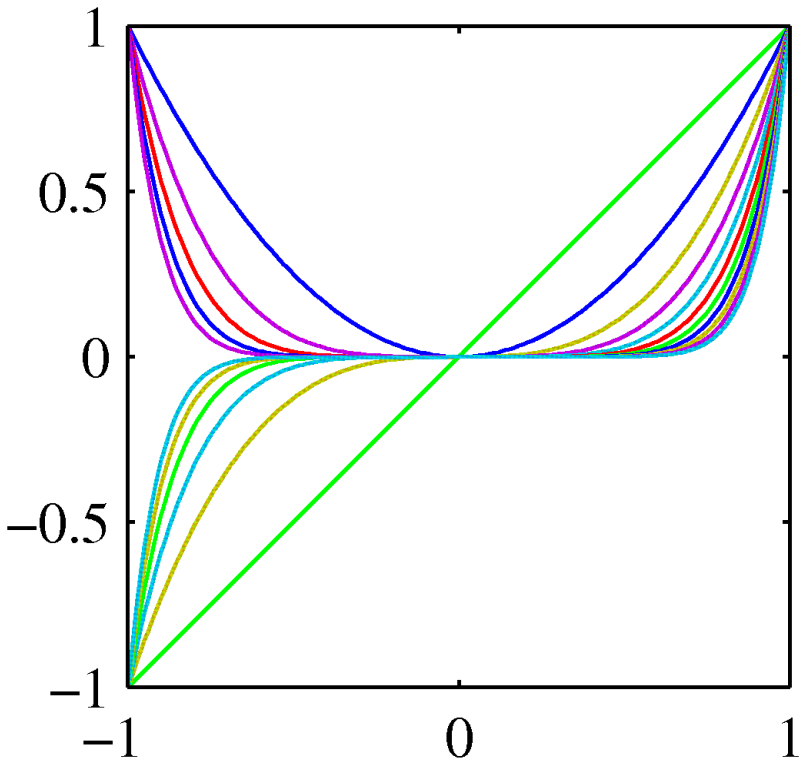


Modele regresji liniowej

- Wielomianowe funkcje bazowe:

$$\phi_j(x) = x^j.$$

- Są globalne: mała zmiana w x zmienia wszystkie funkcje bazowe.

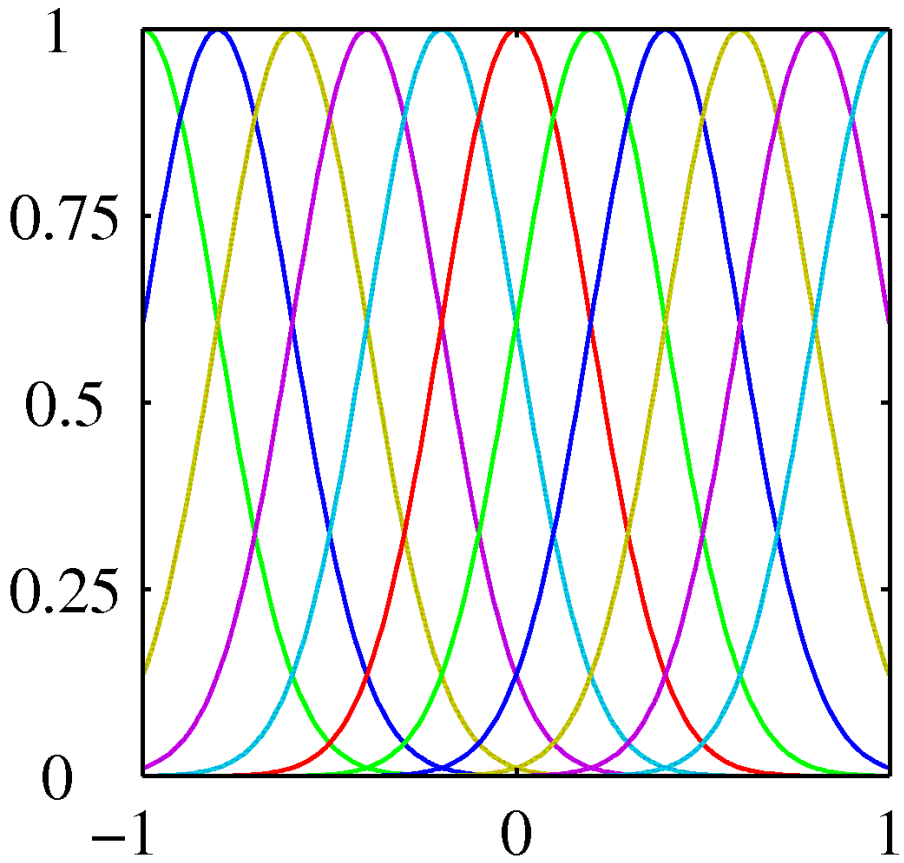


Gaussowskie funkcje bazowe

- Funkcja Gaussa:

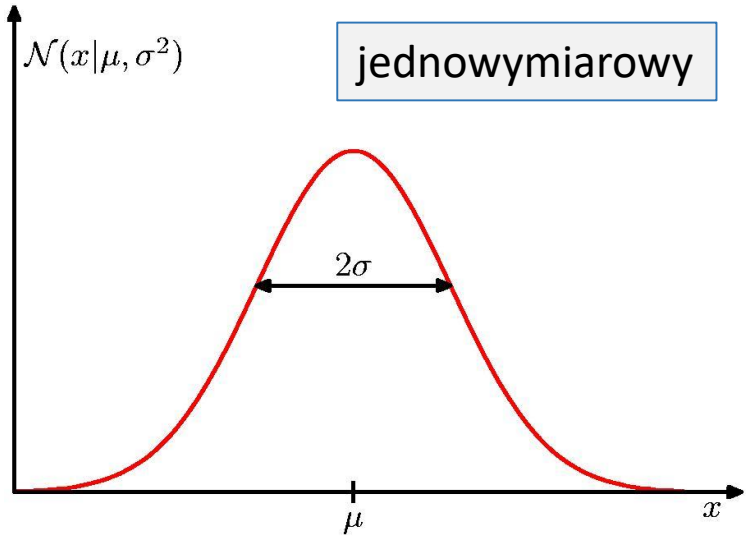
$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

- Lokalne: mała zmiana w x
  zmienia tylko okoliczne (lokalne) funkcje.
- Metoda jądrowa

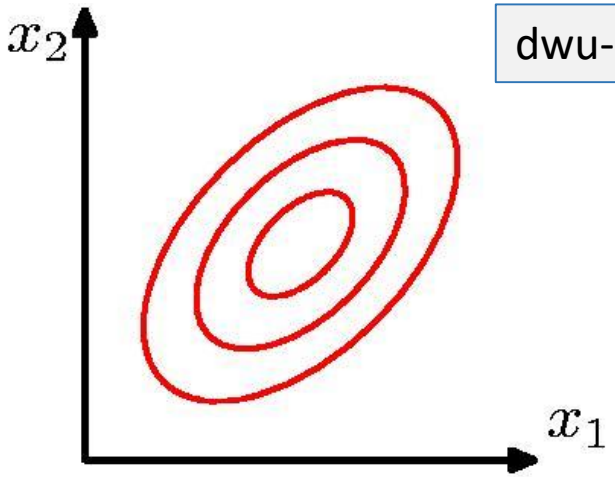


Rozkład Gaussa

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



jednowymiarowy



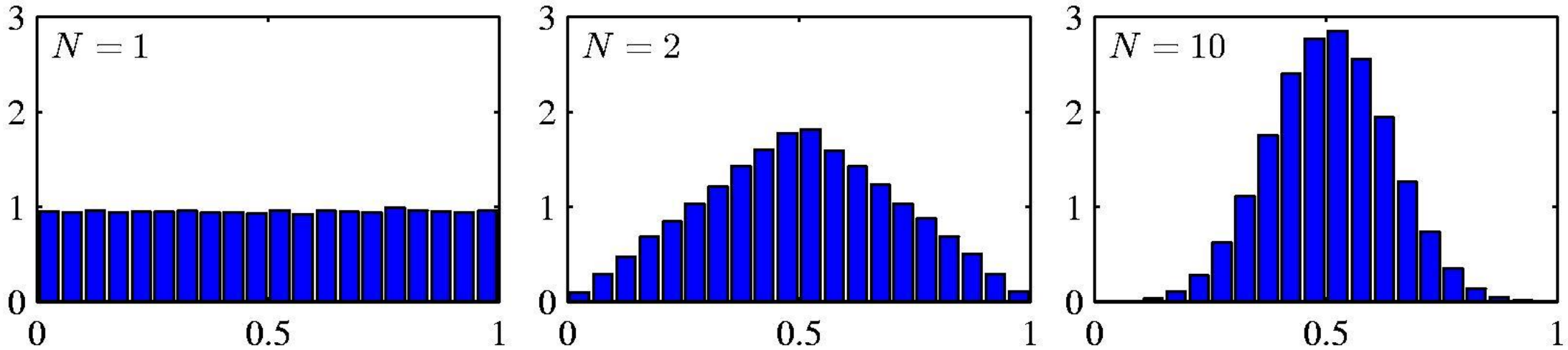
dwu-wymiarowy

wielowymiarowy ⇒

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

Centralne twierdzenie graniczne

- Jeśli zmienne są niezależnymi zmiennymi losowymi o jednakowym rozkładzie, takiej samej wartości oczekiwanej i skończonej wariancji, to zmienna losowa w postaci sumy lokalnych zmiennych losowych zbiega według rozkładu do standardowego rozkładu normalnego, gdy N rośnie do nieskończoności.



Identyfikacja (1)
Maximum Likelihood / Least Squares

- Załóżmy obserwacje funkcji deterministycznej z dodanym szumem Gaussowskim:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \qquad \text{gdzie} \qquad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

- Co jest równoważne z ,

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}).$$

- Dla zaobserwowanych wejść $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ i wyjść $\mathbf{t} = [t_1, \ldots, t_N]^{\mathrm{T}}$ otrzymujemy ocenę wiarygodności

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}).$$

Identyfikacja (2)
Maximum Likelihood / Least Squares

- Z algorytmu otrzymujemy

$$\ln p(\mathbf{t}|\mathbf{w},\beta) = \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1})$$
$$= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta E_D(\mathbf{w})$$

- gdzie

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

- jest sumą kwadratów błędów.

Identyfikacja (3)
Maximum Likelihood / Least Squares

- Wyznaczając gradient i przyrównując go do zera otrzymujemy

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = \mathbf{0}.$$

- A rozwiązując względem w otrzymujemy

$$\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t}$$

Pseudo-odwrotność, Moore-Penrose $\boldsymbol{\Phi}^{\dagger}$

- gdzie

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}.$$

Identyfikacja (4)
Maximum Likelihood / Least Squares

- Minimalizując tylko względem wyrazu wolnego w_0 widzimy, że:

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \overline{\phi_j}$$

$$= \frac{1}{N}\sum_{n=1}^{N} t_n - \sum_{j=1}^{M-1} w_j \frac{1}{N}\sum_{n=1}^{N} \phi_j(\mathbf{x}_n).$$

- We Maksymalizując względem ¯ daje:

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - \mathbf{w}_{\mathrm{ML}}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

Regularyzacja kwadratowa

- Karzemy współczynniki o wysokich wartościach

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Regularyzowana MNK (1)

- Rozważmy funkcje błędu:

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Dane + Regularyzacja

- Otrzymujemy wskaźnik jakości w postaci

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

- I rozwiązanie

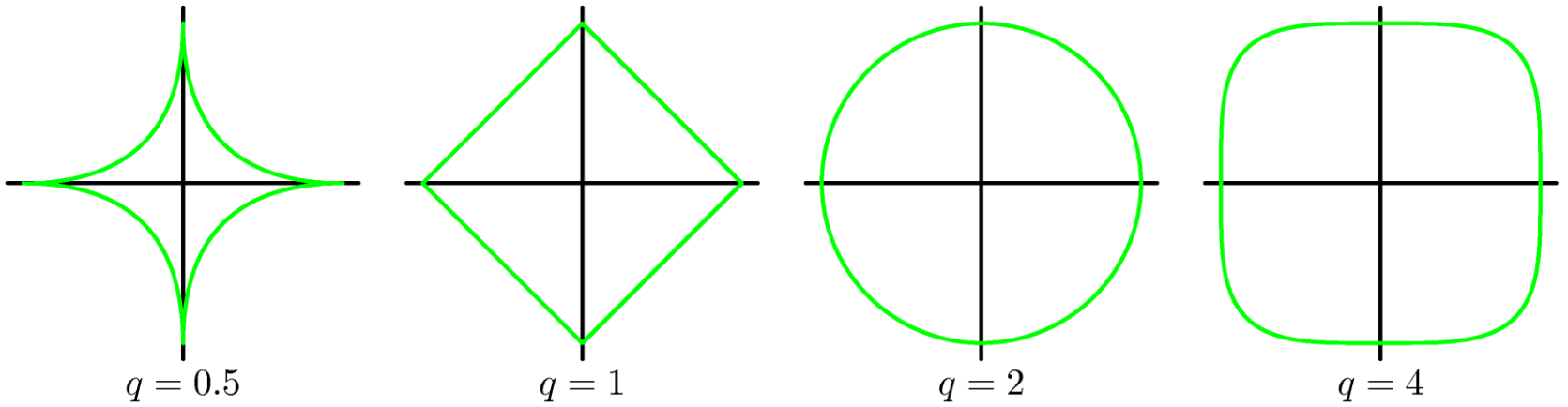
$$\mathbf{w} = \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}.$$

λ: współczynnik regularyzacji

Regularyzowana MNK (2)

- Z bardziej uogólnionym regularyzatorem otrzymujemy

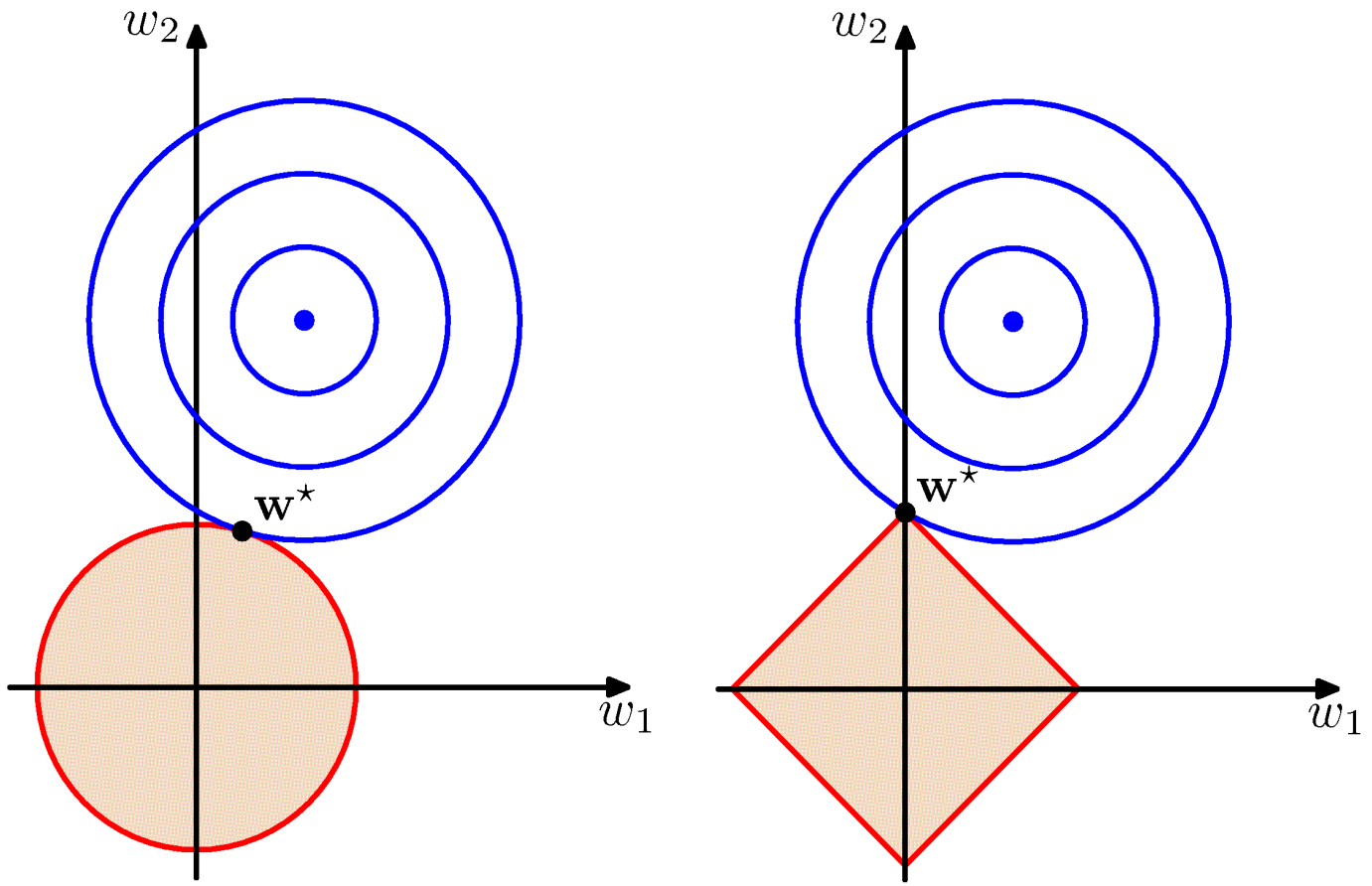
$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$



Lasso kwadratowy

Regularyzowana MNK (3)

- Lasso ma tendencje do bardziej rozproszonych rozwiązań niż regularyzator kwadratowy.



Metody kernelowe

- Podejście w oparciu o funkcje bazowe daje możliwość poszerzenia przestrzenni cech czyniąc proste metody jak regresja liniowa znacznie pojemniejszymi
- Przykład:
  - Wejście: x
  - Cechy (funkcje bazowe) $x, x^2, x^3, \sin(x), \ldots$
- Dwa potencjalne ograniczenia:
  - Wydajność obliczeniowa: jak znaleźć właściwe funkcje bazowe
  - Regularyzacja: jak uniknąć przetrenowania?
- Metody kernelowe próbują rozwiązać powyższe kwestie

Definicja jądra

- Załóżmy, że $\boldsymbol{\phi}(\mathbf{x})$ mapuje D-wymiarowy wektor wejściowy $\mathbf{x}$ na wielowymiarową (nieskończenie) przestrzeń cech
- Proste metody bazują na prostym iloczynie wektorów cech, $\boldsymbol{\phi}(\mathbf{x}_1)^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_2)$
- Dla pewnych przestrzeni można wykorzystać “kernel trick” do wyznaczenia iloczynu $\boldsymbol{\phi}(\mathbf{x}_1)^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_2)$ tylko przy użyciu wektora wejść:

$$\boldsymbol{\phi}(\mathbf{x}_1)^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_2) = k(\mathbf{x}_1, \mathbf{x}_2)$$

- $k(\mathbf{x}_1, \mathbf{x}_2)$ jest <u>jądrem</u>

- Sprawdzenie czy $k(x,y)$ jest właściwym jądrem zależy jedynie o właściwości samej funkcji jądrowej, bez konieczności weryfikacji macierzy cech

Przykłady funkcji jądrowych

- $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^{\mathrm{T}} \mathbf{x}_2$

- $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{x}_2$
  ($\boldsymbol{\Sigma}^{-1}$ symetryczna dodatnio zdefiniowana)

- $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2 / 2\sigma^2)$

- $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{1}{2} \mathbf{x}_1^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{x}_2)$
  ($\boldsymbol{\Sigma}^{-1}$ symetryczna dodatnio zdefiniowana)

- $k(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1) p(\mathbf{x}_2)$

Modularność

Metody kernelowe składają się z dwu części:

1) Wybór jądra (zadanie nietrywialne)
2) Właściwy algorytm go wykorzystujący

Modularność: możemy wykorzystać dowolny kernel z dowolnym algorytmem.

Przykładowe jądra:

$$k(x,y) = e^{(-\|x-y\|^2/c)}$$

$$k(x,y) = (< x,y > +\theta)^d$$

$$k(x,y) = \tanh(\alpha < x,y > +\theta)$$

$$k(x,y) = \frac{1}{\sqrt{\| x-y \|^2 + c^2}}$$

Przykładowe algorytmy:
- SVM (Support Vector Machine)
- Fisher discriminant analysis
- Regresja kernelowa
- kernel PCA
- kernel CCA

Proces Gaussowski

- Dla regresji liniowej: $\mathbf{y} = \mathbf{\Phi w}$
- Wykorzystując macierz Φ, wektor predykcji to $y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})$
- Jeśli (najprościej) $\mathbf{w}$: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$
- wtedy $\mathbb{E}[\mathbf{y}] = \mathbf{\Phi}\mathbb{E}[\mathbf{w}] = \mathbf{0}$

$$\mathrm{cov}[\mathbf{y}] = \mathbb{E}\left[\mathbf{y}\mathbf{y}^{\mathrm{T}}\right] = \mathbf{\Phi}\mathbb{E}\left[\mathbf{w}\mathbf{w}^{\mathrm{T}}\right]\mathbf{\Phi}^{\mathrm{T}} = \frac{1}{\alpha}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}} = \mathbf{K}$$

- $\mathbf{K}$ nazywamy macierzą Grama, gdzie

$$K_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) = \frac{1}{\alpha}\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_m)$$

- Tym samym: *korelacja dwu predykcji równa się jądru wyznaczonemu dla właściwych wejść*

$$\mathbb{E}\left[y(\mathbf{x}_n)y(\mathbf{x}_m)\right] = k(\mathbf{x}_n, \mathbf{x}_m)$$

Przykład: $k(x, x') = \exp\left(-\theta\,|x - x'|\right)$

Proces Gaussa: “uczenie” i predykcja

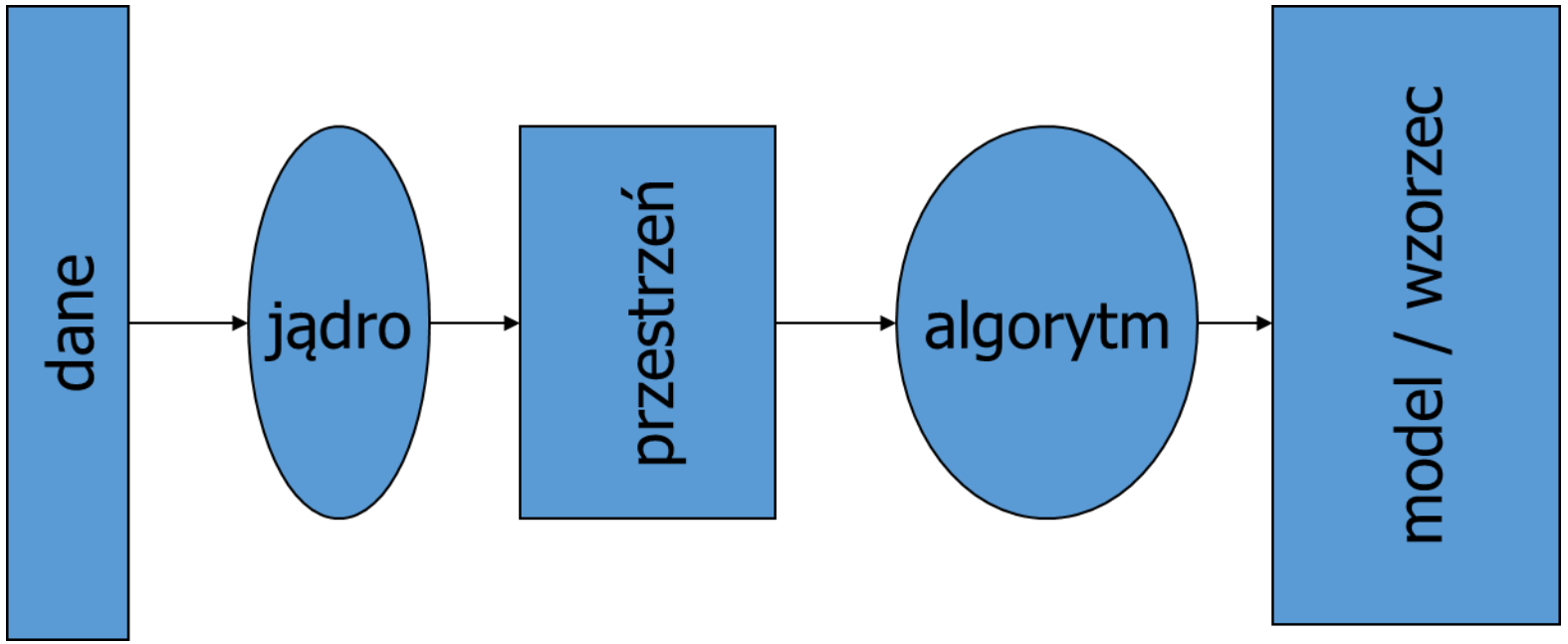
- Jak poprzednio zakładamy $p(t_n|y_n) = \mathcal{N}(t_n|y_n, \beta^{-1})$
- Wiarygodność wektora wyjściowego $p(\mathbf{t}|\mathbf{y}) = \mathcal{N}(\mathbf{t}|\mathbf{y}, \beta^{-1}\mathbf{I}_N)$
- Biorąc $p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$, otrzymujemy graniczną dystrybucję predykcji wyjścia:

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{y})p(\mathbf{y})\,\mathrm{d}\mathbf{y} = \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C})$$

- gdzie $C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}$
- Prognozy oparte są na: $p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$
- gdzie $\mathbf{C}_{N+1} = \begin{pmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^{\mathrm{T}} & c \end{pmatrix}$ $\mathbf{k}_n = k(\mathbf{x}_n, \mathbf{x}_{N+1})$ $c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$
- $p(t_{N+1}|\mathbf{t})$ jest Gaussowski z:

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{t}$$
$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{k}$$

Metody jądrowe: plug and play



Wygładzanie (interpolacja) liniowa

- Wygładzanie jądrowe opisuje funkcje wagowe $W_{ni}(x)$ poprzez funkcję gęstości K wraz z parametrem skali, który modyfikuje (poprawia) rozmiar i formę wag w okolicy x.
- Jadro K jest gładką, ograniczoną i symetryczna funkcja rzeczywistą całkowaną do 1.
- Wagi są opisane jako:

$$W_{hi}(x) = K_h(x - X_i) / \hat{f}_h(x)$$

- gdzie

$$\hat{f}_h(x) = n^{-1} \sum_{i=1}^{n} K_h(x - X_i)$$

$$K_h(u) = h^{-1} K(u/h)$$

Estymata k-NN (k-Nearest Neighbor)

- Dla k-NN, otoczenie zdefiniowane jest poprzez te zmienne X, które są pomiędzy k-najbliższymi sąsiadami *x* względem odległości euklidesowej.
- Estymator k-NN definiujemy jako:

$$\hat{m}_k(x) = n^{-1} \sum_{i=1}^{n} W_{ki}(x) Y_i$$

- gdzie $\{ W_{ki}(x) \}\, i=1, \ldots, n$ definiujemy jako zbiór indeksów

$$J_x = \{i : X_i \; i \; jest \; jedną \; z \; k \; najblizszych \; obserwacji \; x\}$$

- i

$$W_{ki}(x) = \begin{cases} n/k, if \;\; i \in J_x \\ 0 \quad inne \end{cases}$$

Estymata k-NN

- Parametr wygładzania *k* jest odpowiedzialny za gładkość estymaty (pasmo wygładzania).
- Jeśli $k > n$, estymata *k - NN* odpowiada średniej.
- Jeśli $k = 1$, obserwacje są odtworzone w X_i i dla *x* pomiędzy dwoma sąsiednimi predykowanymi zmiennymi jest odtworzone jako skok pomiędzy nimi.

Jednowymiarowe wygładzanie jądrowe

- k-NN: $\hat{f}(x) = Ave(y_i \mid x_i \in N_k(x))$
- 30-NN nie jest gładka, ponieważ $\hat{f}(x)$ jest nieciągła w x.
- Dyskretne zmiany powodują nieciągłość $\hat{f}(x)$.

Jednowymiarowe wygładzanie jądrowe

- Nadaraya-Watson Kernel średnia ważona:

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)}$$

- Jądro kwadratowe Epanechnikova:

$$K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right)$$

Lokalna regresja liniowa

- Kwestie graniczne
  - Złe uwarunkowanie na ograniczeniach ze względu na niesymetrie w okolicy
  - Przesunięcie jest usunięte przez dopasowanie liniowe

Lokalna regresja liniowa

- Lokalna ważona regresja liniowa realizuje korekcję pierwszego rzędu
- Oddzielne ważone MNK dla każdego punktu x_0:

$$\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)[y_i - \alpha(x_0) - \beta(x_0)x_i]^2$$

- aby estymować: $\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_0$
- *b(x)T=(1,x); macierz regresji B:* $N \times 2$ oraz *i*-tym wierszem *b(x)T*;

$$W_{N\times N}(x_0) = diag\big(K_\lambda(x_0, x_i)\big), i = 1, \ldots, N$$

$$\hat{f}(x_0) = b(x_0)^T (B^T W(x_0) B)^{-1} B^T W(x_0) y = \sum_{i=1}^{N} l_i(x_0) y_i$$

Lokalna regresja liniowa

- Wagi $l_i(x_0)$ łączą lokalne wygładzanie jądrowe $K_\lambda(x_0,\cdot)$ i metodę najmniejszych kwadratów – Equivalent Kernel

Lokalna regresja wielomianowa

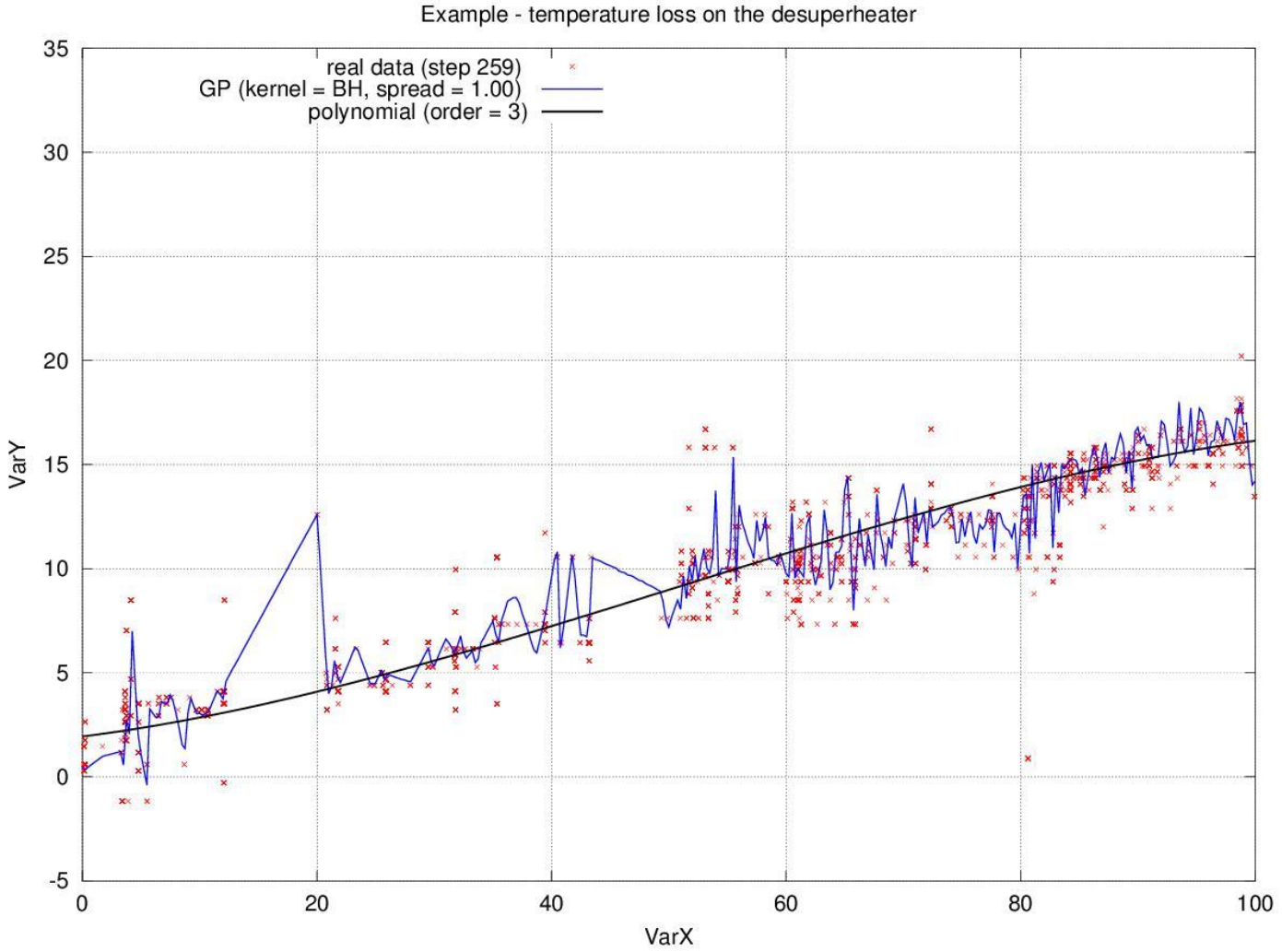
- Lokalne dopasowanie wielomianu o stopniu d

Ograniczenia stałych funkcji jądrowych

- Rozmieszczać równomiernie – przekleństwo wymiarowości.
- Może grupowanie danych
  - Metoda k-means
  - Metoda C-means
  - Sieci RBF
  - ...

Przykład #3 (1) - zawór

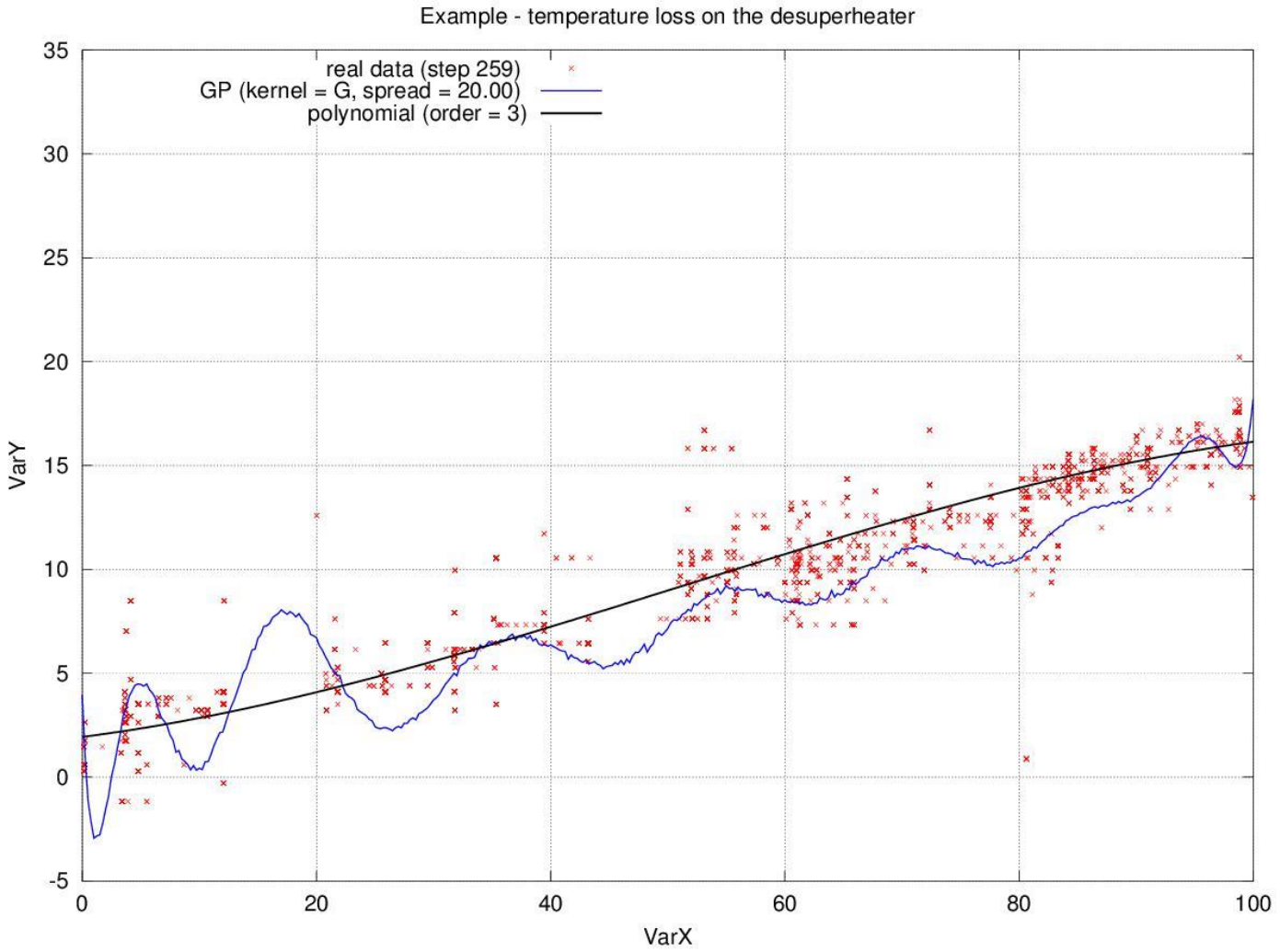
Biharmonic kernel



Przykład #3 (2) - zawór

Gaussian kernel

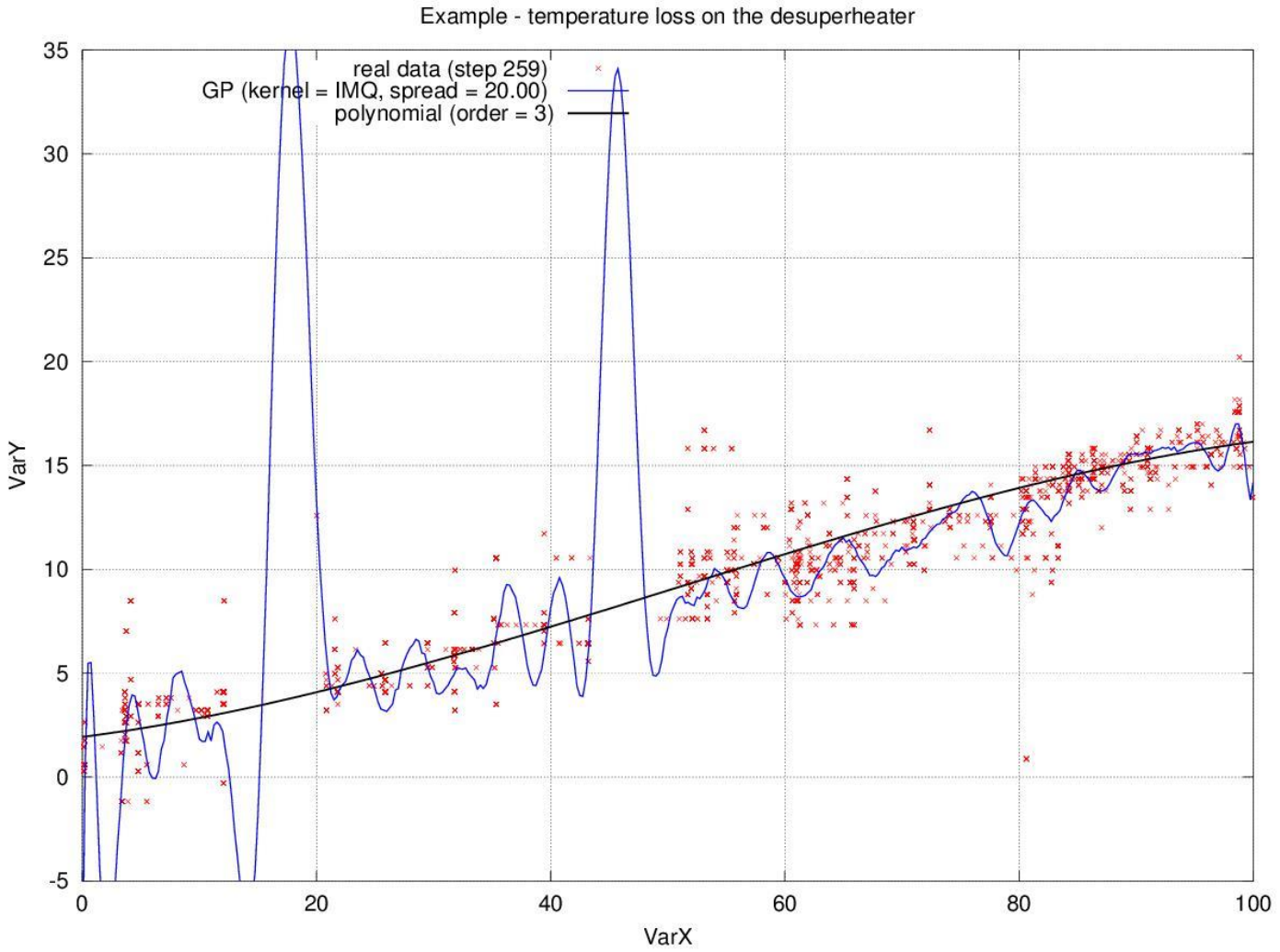
$$\phi(r) = e^{-(\varepsilon r)^2}$$



Przykład #3 (3) - zawór

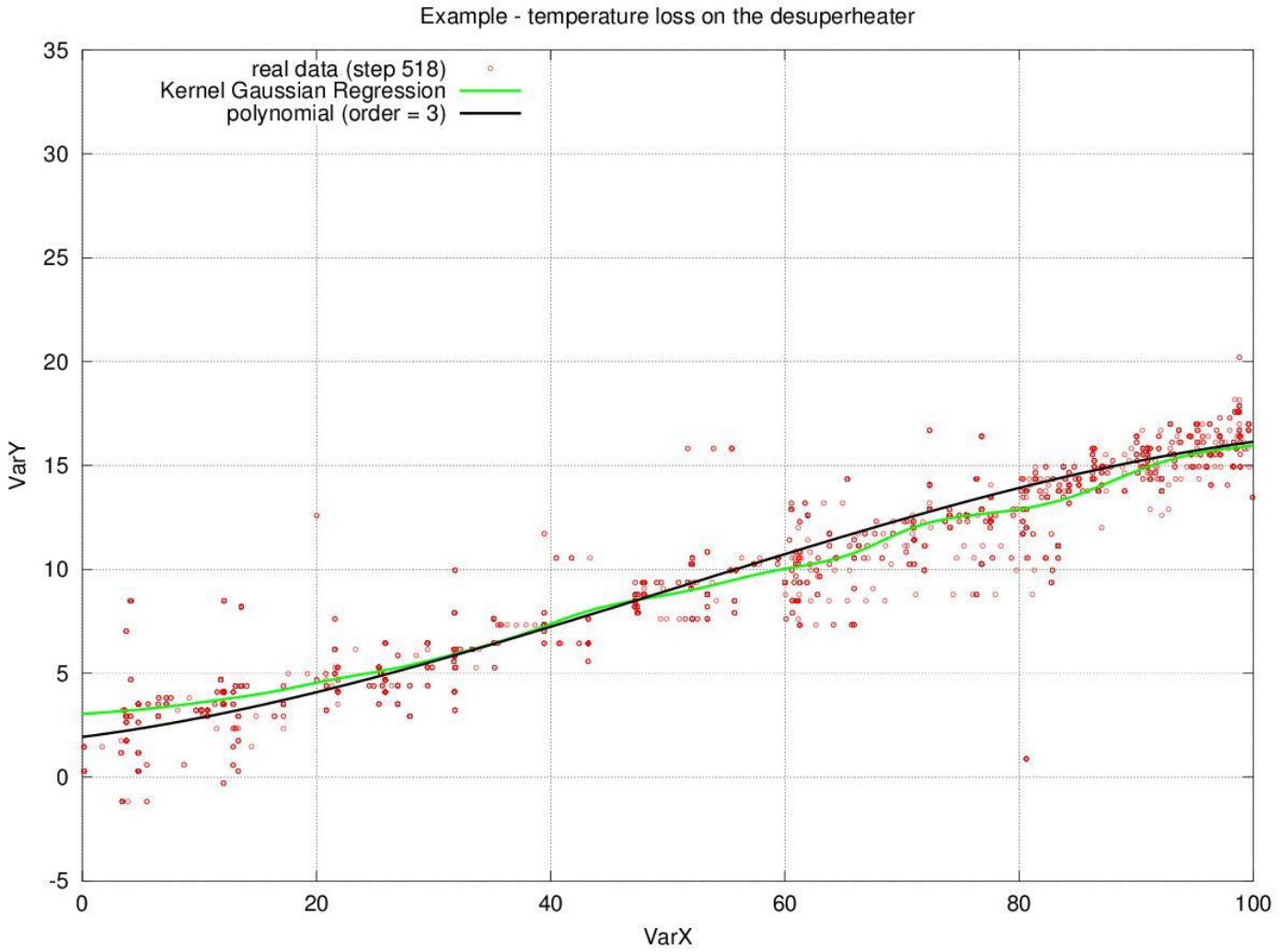
Inverse Multiquadric kernel

$$\phi(r) = \frac{1}{\sqrt{1 + (\varepsilon r)^2}}$$



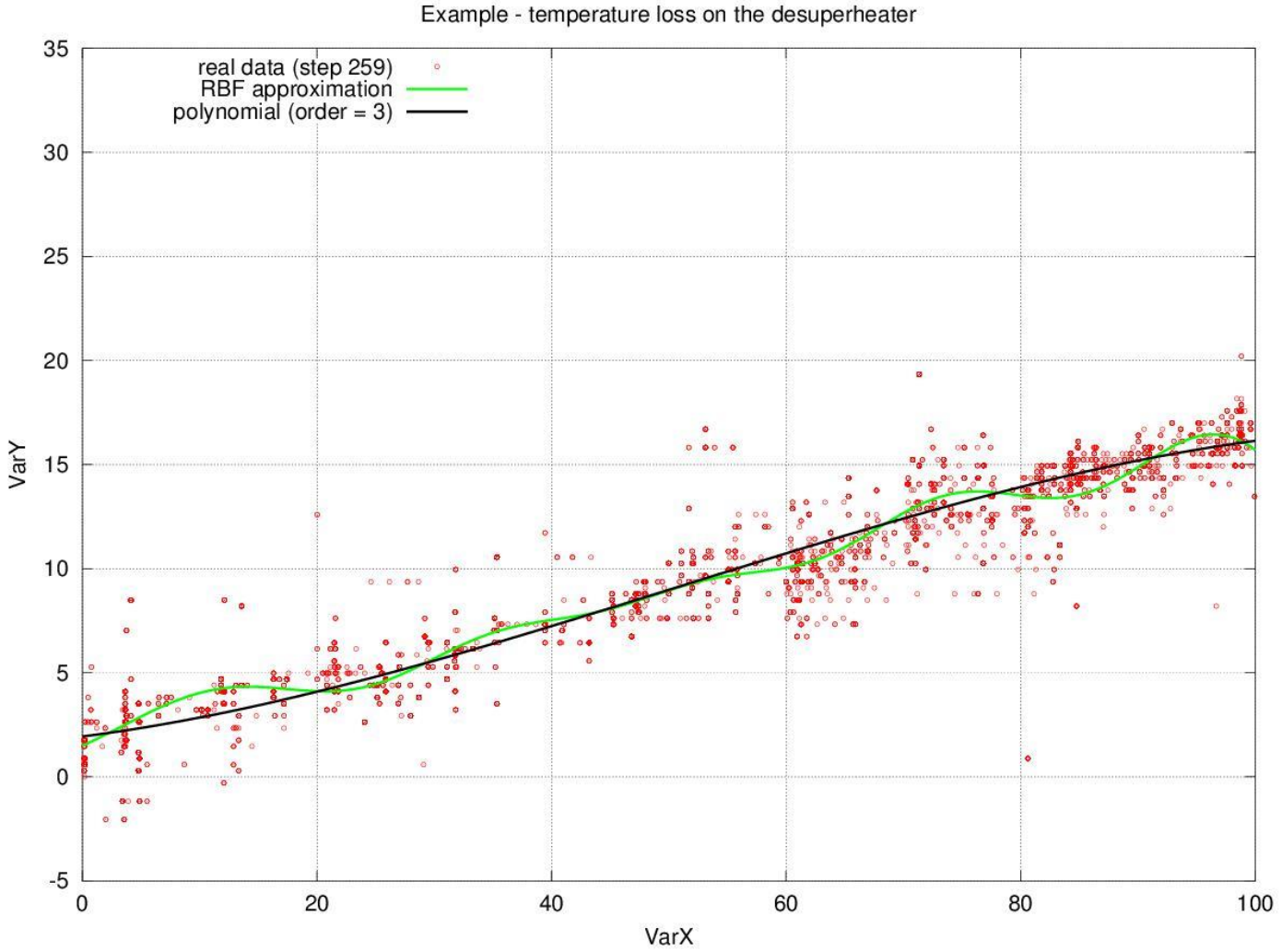
Przykład #3 (4) - zawór

KGR



Przykład #3 (5) - zawór

Radial Basis Function
multiple-running



Przykład #3 (6) - zawór

Radial Basis Function
single shot