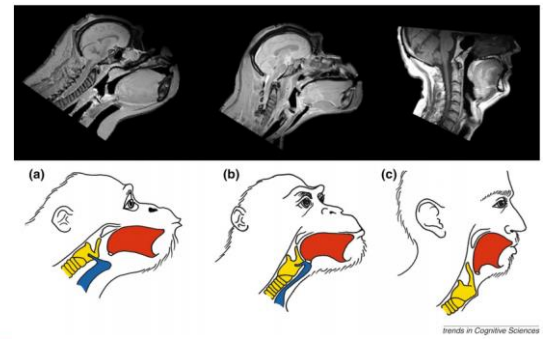
AGH University of Science and Technology

# Technologia Mowy

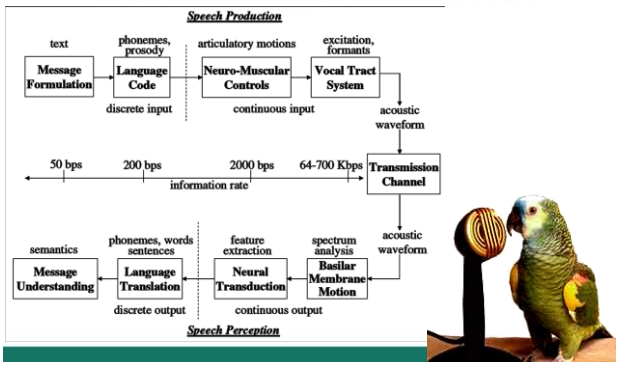
**Wykład**

**Jakub Gałka, Katedra Elektroniki © 2011-2016**
jgalka@agh.edu.pl

## Pochodzenie mowy - ewolucja




## Mowa a informacja




## Komunikacja - Informacja

**XXI wiek – czy to wiek danych, czy wiek informacji?**

## Mowa a technologia

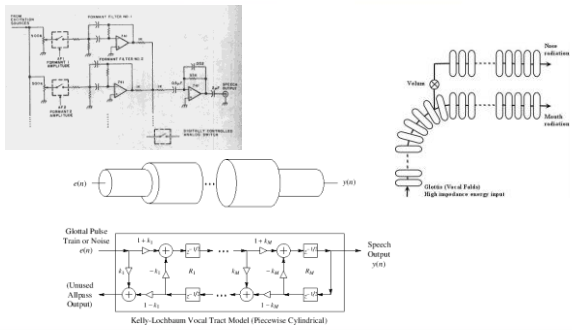
## Mowa a technologia

- Tranzystor i Telekomunikacja
- Rozwój techniki i technologii
- Maszyny cyfrowe
- Przetwarzanie sygnałów i w konsekwencji mowy (Shannon, Viterbi, Markow)



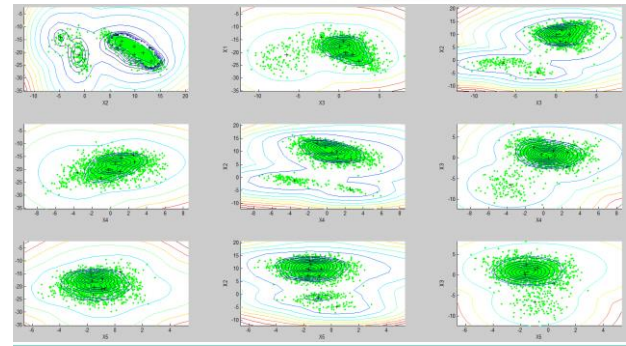
- Telefonia cyfrowa – komórkowa

## Modele traktu głosowego (Voder, 1939, http://www.youtube.com/watch?v=e5gQBei-z-c)



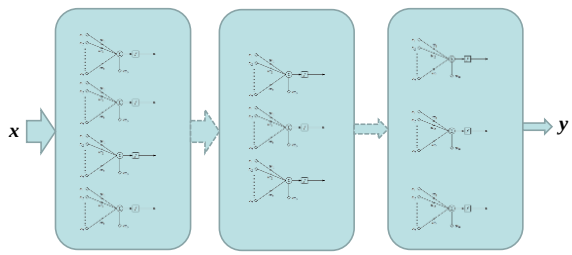

Rys.

## Do niedawna - *status-quo* DSP i modelowanie statystyczne



## Sieci neuronowe

Frank Rosenblatt (1928-1971)

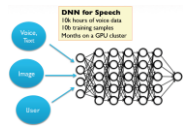



## Stabilizacja (stagnacja?) technologii

- Przetwarzanie
- Modelowanie statystyczne
- Konwencjonalny Machine Learning
- Zastosowania domenowe (specjalizowane)

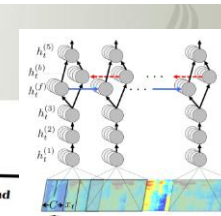
## Nowe nadzieje, nowe środowisko społeczno-techniczne

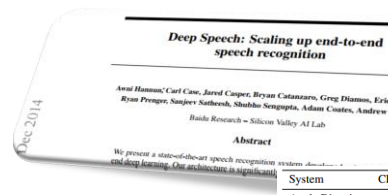
- IoT, BIG DATA
- *Crowdsourcing*
- WEB 2.0
- *Deep Learning*


- **Kolejny przełom ?**

## Kolejny przełom ?

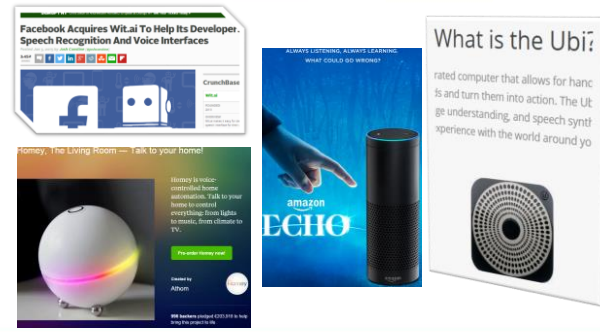





| System | Clean (94) | Noisy (82) | Combined (176) |
|---|---|---|---|
| Apple Dictation | 14.24 | 43.76 | 26.73 |
| Bing Speech | 11.73 | 36.12 | 22.05 |
| Google API | 6.64 | 30.47 | 16.72 |
| wit.ai | 7.94 | 35.06 | 19.41 |
| **Deep Speech** | **6.56** | **19.06** | **11.85** |

## Postępy


## Sukcesy, klapy, Startup'y, korporacje

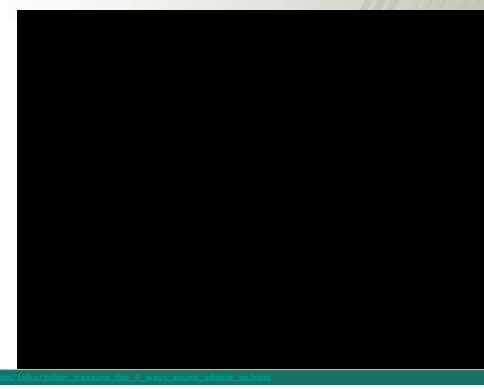



## Podstawowe oznaczenia

- $x$ – zmienna skalarna
- $\boldsymbol{x}$ – zmienna wektorowa
- $X$ – stała, macierz
- $\boldsymbol{X}$ – zbiór
- $x_i$, $x(i)$, $X(i,j)$ – element wektora, macierzy
- $\alpha$, $\boldsymbol{\beta}$, … – parametr skalarny, wektorowy
- $y=f(x)$, $\boldsymbol{z}=\boldsymbol{g}(\boldsymbol{x})$ – funkcje
- $\boldsymbol{x}^T$, $X^T$ – transpozycja
- itd.

## Dźwięki i ich postrzeganie

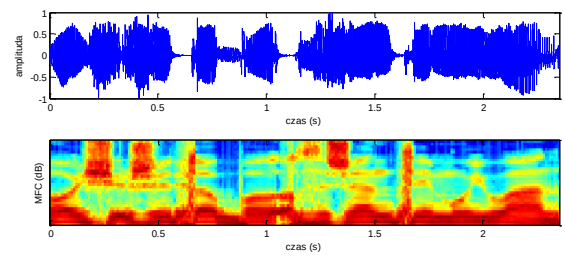
http://www.ted.com/talks/julian_treasure_the_4_ways_sound_affects_us.html



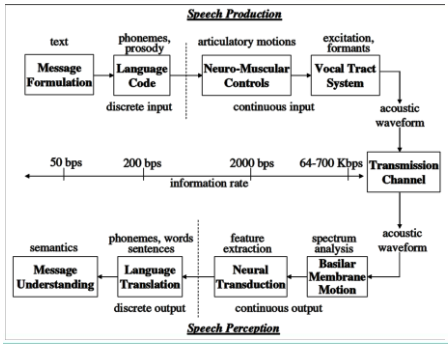
## Mowa

- Szybkie i precyzyjne przekazywanie informacji – rozwój gatunku
- Proces złożony / wielopoziomowy
- Ewolucja mowy – optymalizacja
  http://uvafon.hum.uva.nl/bart/papers/deBoerEncyclopedia2006.pdf
- Pragmatyzm (środowisko)
- Anatomia i fizjologia
- Komunikaty, informacje, pojęcia, abstrakty, myślenie
- Wspólna wiedza

## Czym jest mowa

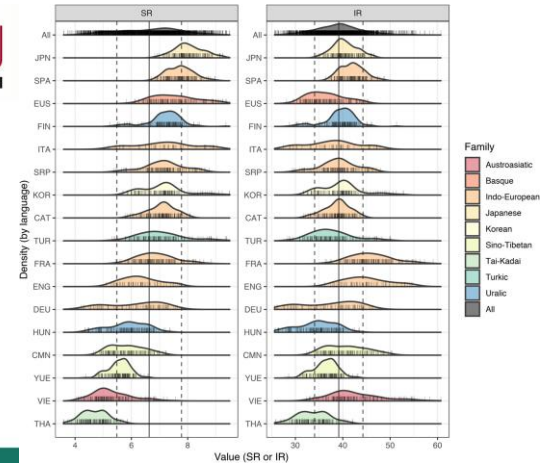


## Zjawisko mowy – proces komunikacji



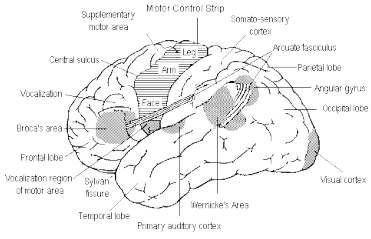

| |
|---|
| myśl |
| wypowiedź |
| zdania |
| słowa |
| sylaby |
| głoski / fonemy |

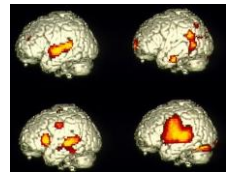
Rys. L. Rabiner




## Ośrodki mowy w mózgu

- Obszary Broc'a i Wernicke'go

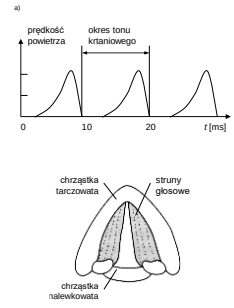


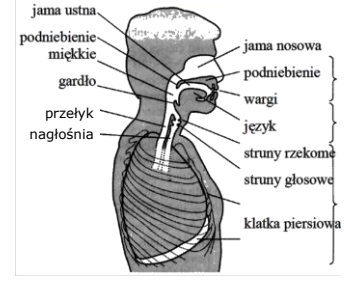



- Więcej ciekawostek: www.talkingbrains.org

Rys.

## Aparat mowy

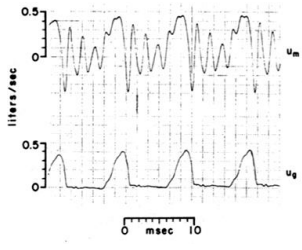



Rys. I, – Wikimedia Commons,

## Aparat mowy – struny głosowe

| Głos | Rodzaj | [Hz] | |
|---|---|---|---|
| | | Od | Do |
| bas | męski | 80 | 320 |
| baryton | męski | 100 | 400 |
| tenor | męski | 120 | 480 |
| alt | żeński | 160 | 640 |
| mezzosopran | żeński | 200 | 800 |
| sopran | żeński | 240 | 960 |

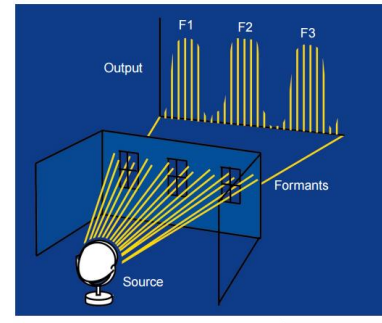



Struny głosowe (endoskop): http://www.youtube.com/watch?v=Z_ZGqn1tZn8

Rys. I, – Wikimedia Commons,

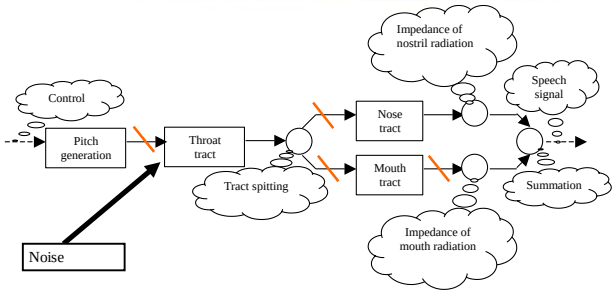
## Dziwięk mowy

- Analiza widma

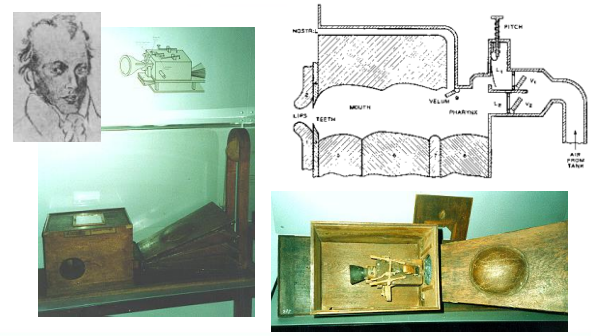



Trends in Cognitive Sciences

## Trakt głosowy




## Modele traktu głosowego
## Von Kempelen (1791), Riesz (1937)



Rys.

## Modele traktu głosowego

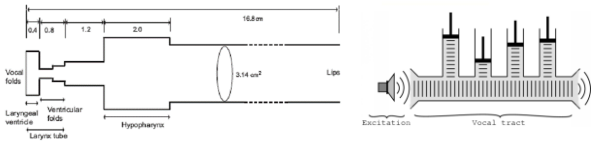
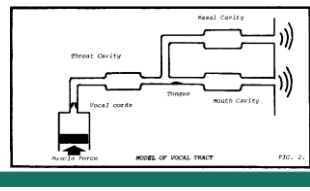
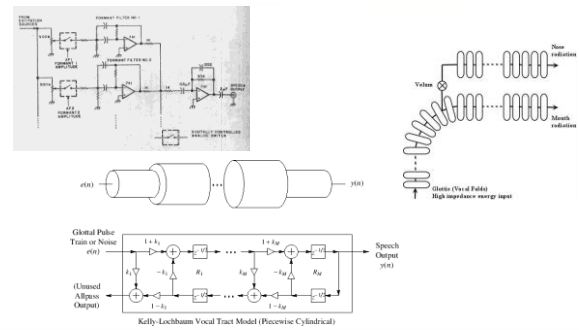



Figure 3: Sketch of the vocal tract model

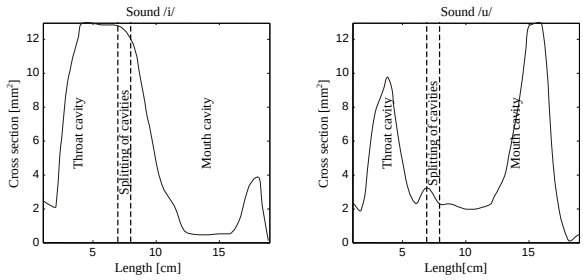



## Modele traktu głosowego
## (Voder, 1939, http://www.youtube.com/watch?v=e5gQBei-z-c)

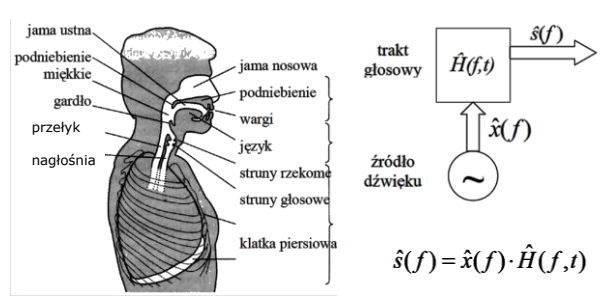



Rys.

## Profil traktu głosowego

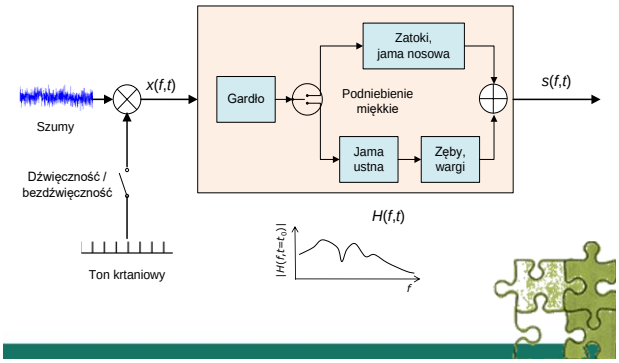



## Model sygnałowy źródło-filtr



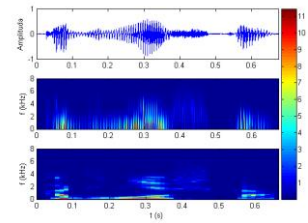

Rys. R. Tadeusiewicz, *Sygnał Mowy*

## Model źródło-filtr, c. d.

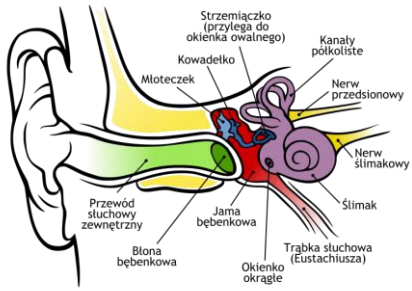



## Akustyka mowy

- Dopasowany system komunikacji
- Fizyka, teoria informacji,…
- Zakres częstotliwości:
  - PCM 4000Hz
  - Std. 8000Hz
  - CD 22050Hz
- Dynamika
  - min. 3, 4 bity
  - 6.5*3 ~ 20 dB



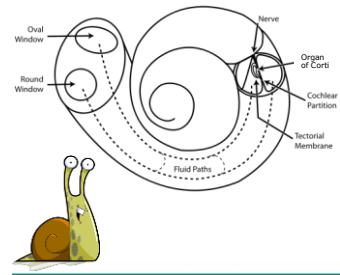
## Narząd słuchu

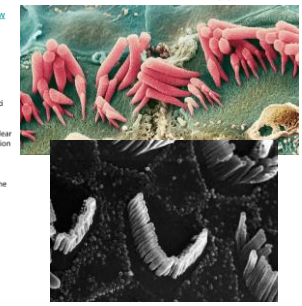



Rys. http://wupa.wustl.edu/record_archive/1998/10-14-98/people.html Wikimedia Commons, Chittka L, Brockmann

## Ślimak, narząd Cortiego

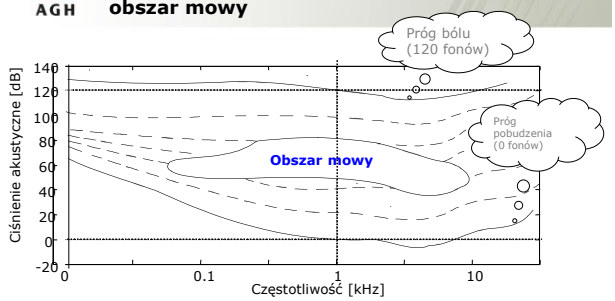
Aparat słuchowy człowieka: http://www.youtube.com/watch?v=7O-adw-HyrQ
Narząd Cortiego - zasada działania: http://www.youtube.com/watch?v=xMUI5CCoW6Y (_)
Przykład: http://www.youtube.com/watch?v=dyenMluFaUw



Rys. Wikimedia Commons, www.sciencephotolibrary.com

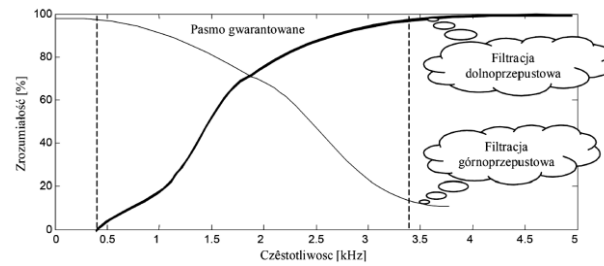
## Krzywe izofoniczne Fletchera-Munsona obszar mowy




fragment siatki obiektywnej skali decybelowej,
subiektywna skala fonowa - krzywe izofoniczne
próg pobudzenia i próg bólu

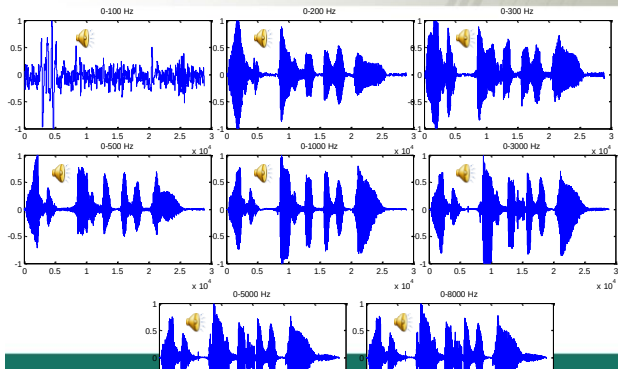
Rys. M. Kępiński

## Pasmo a zrozumiałość

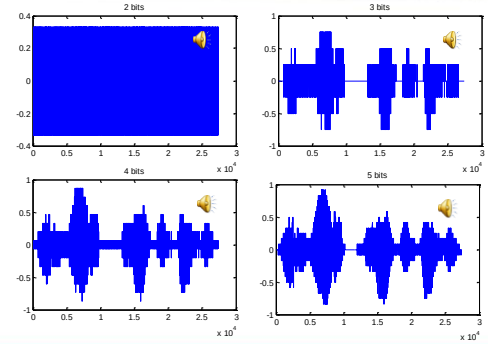



Rys. M. Kępiński

## Pasmo sygnału - przykłady

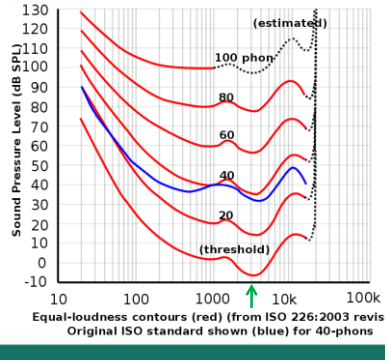



## Liczba bitów - przykłady




## Krzywe izofoniczne Fletchera-Munsona ISO 226:2003

0 dB
-10 dB
-10 dB + *Loudness/Contour* EQ




Rys. Lindosland, Wikimedia Commons

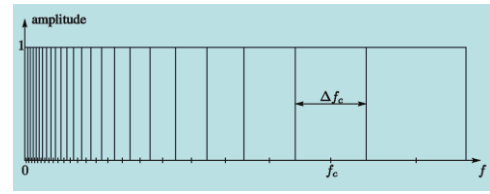
## Rozdzielczość częstotliwościowa

- Jednostka JND (ang. *Just Noticeable Difference*)

$$2 \cdot \Delta f \approx f_{ref,Hz} \cdot 0{,}7\%$$

- Pasma krytyczne (Barkhausen) – skala Bark (Zwicker, 1961)

$$\Delta f_{Bark}(f_{Hz}) = 25 + 75\left(1 + 1.4\left(\frac{f_{Hz}}{1000}\right)^2\right)^{0.69} \qquad f_{Bark}(f_{Hz}) = 13\arctan\left(\frac{0.76\, f_{Hz}}{1000}\right) + 3.5\arctan\left(\frac{f_{Hz}^2}{7500^2}\right)$$




## Rozdzielczość częstotliwościowa

- Jednostka JND (ang. *Just Noticeable Difference*)

$$2 \cdot \Delta f \approx f_{ref,Hz} \cdot 0{,}7\%$$

- Pasma krytyczne (Barkhausen) – skala Bark (Zwicker, 1961)

$$\Delta f_{Bark}(f_{Hz}) = 25 + 75\left(1 + 1.4\left(\frac{f_{Hz}}{1000}\right)^2\right)^{0.69} \qquad f_{Bark}(f_{Hz}) = 13\arctan\left(\frac{0.76\, f_{Hz}}{1000}\right) + 3.5\arctan\left(\frac{f_{Hz}^2}{7500^2}\right)$$

- ERB (ang. *Equivalent Rectangular Bandwidth*)

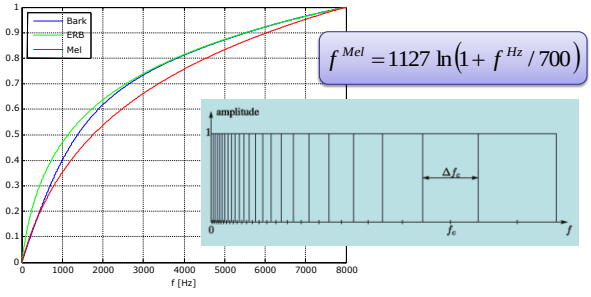
$$ERB(f_{Hz}) = 21.4 \cdot \log_{10}\left(\frac{4.37\, f_{Hz}}{1000} + 1\right)$$

- Mel (Stevens, Volkman, Newman, 1937)

$$f^{Mel} = 1127.01048 \ln(1 + f / 700)$$

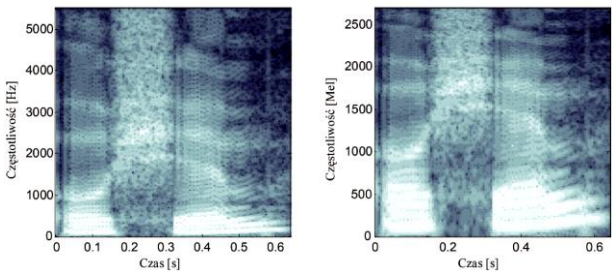
- Więcej o transformacjach: https://ccrma.stanford.edu/~jos/bbt/bbt.pdf
- *A perceptual space that can explain the robustness of bio-acoustic communication* (53 min.):
  http://video.google.com/videoplay?docid=1131332207960335088

## Melowa psychoakustyczna skala częstotliwości




$$f^{Mel} = 1127 \ln(1 + f^{Hz} / 700)$$

## Spektrogram w skali melowej




Rys. M. Kępiński

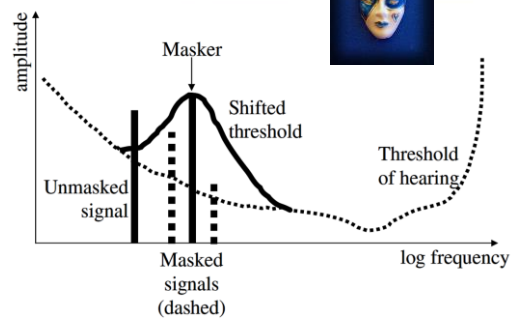
## Rozdzielczość częstotliwościowa przykłady (JND=*f**0.7%)

| *f* | | | |
|---|---|---|---|
| 200 Hz | | | |
| 500 Hz | | | |
| 2000 Hz | | | |

## Rozdzielczość częstotliwościowa przykłady (JND=*f**0.7%)

| *f* / *df* | 0.5 JND | 1 JND | 2 JND |
|---|---|---|---|
| 200 Hz | | | |
| 500 Hz | | | |
| 2000 Hz | | | |

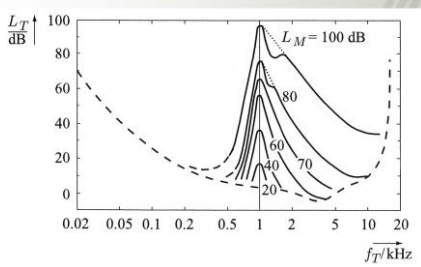
## Maskowanie




66

Rys. L. Rabiner, Introduction to Digital Speech Processing, NOW 2007

## Maskowanie tonów




Level $L_T$ of a sinusoidal test tone of frequency $f_T$ masked by
—— a sinusoidal masker with $f_M = 1$ kHz
..... narrowband noise with level $L_M$ and with critical bandwidth, centered at $f_M = f_c = 1$ kHz ([Zwicker, Fastl 1999])

Rys. P. Vary, R. Martin, Digital Speech Transmission, Wiley 2005

## Maskowanie w czasie

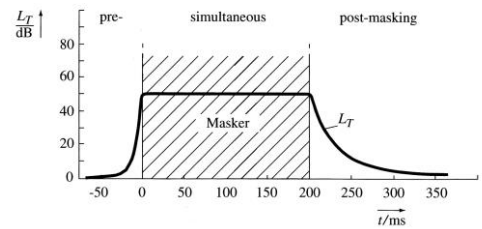
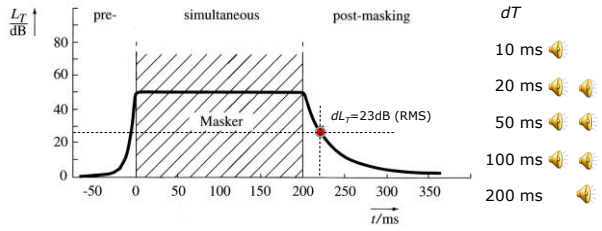



Figure 2.22: Pre- and post-masking: necessary level $L_T$ for the audibility of a sinusoidal burst (test tone) masked by wideband noise (adapted from [Zwicker, Fastl 1999])

Rys. P. Vary, R. Martin, Digital Speech Transmission, Wiley 2005

## Maskowanie w czasie - przykłady




*dT*

10 ms

20 ms

50 ms

100 ms

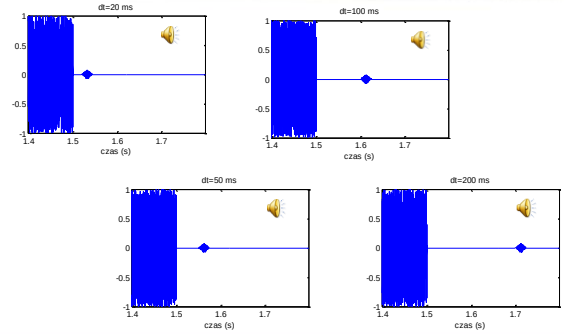
200 ms

Figure 2.22: Pre- and post-masking: necessary level $L_T$ for the audibility of a sinusoidal burst (test tone) masked by wideband noise (adapted from [Zwicker, Fastl 1999])

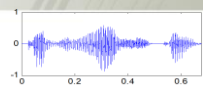
Rys. P. Vary, R. Martin, Digital Speech Transmission, Wiley 2005

## Maskowanie szumem – przykłady

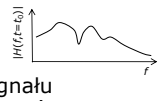
$dL_T$=23dB (RMS)



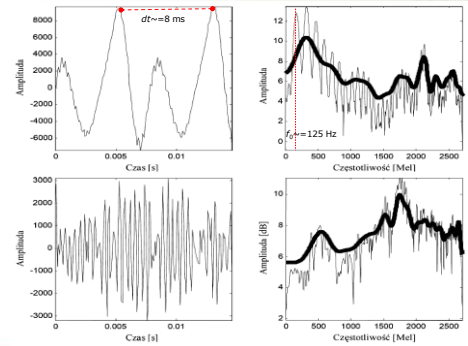

## Sygnał mowy



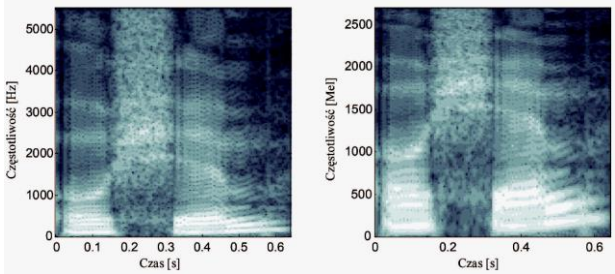
- Sygnał czasowo-częstotliwościowy
- Zmienna moc i widmo chwilowe
- Zakres(*) częstotliwości 50<f<8kHz
- Niestacjonarny globalnie
- *Quasi-stacjonarny* w porcjach 10-20ms
- Dźwięczny – bezdźwięczny – impulsowy
- Częstotliwość podstawowa – $f_0$
- Formanty: $f_1$, $f_2$, $f_3$, ($f_4$)
- Częstotliwościowa zmienność własna sygnału (np. pozycje formantów) odpowiada zdolności rozdzielczej słuchu (Mel) !!!




## Widmo - ton krtaniowy i formanty /i/, /ś/

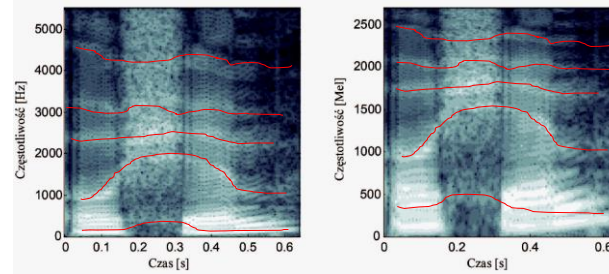



## Spektrogramy i formanty




Rys. M. Kępiński

## Spektrogramy i formanty




Rys. M. Kępiński

## Formanty – średnie częstotliwości (Hz)

**w kontekstach**

| FEMALE | | | | | |
|---|---|---|---|---|---|
| VOWEL | FORMANT | | | | |
| | F1 | F2 | F3 | F4 | F5 |
| [i] | 361 | 2752 | 3431 | 4304 | 5178 |
| [i] | 484 | 2077 | 2933 | 4423 | 5136 |
| [e] | 599 | 2140 | 2966 | 4349 | 5250 |
| [a] | 821 | 1627 | 2710 | 4232 | 5240 |
| [o] | 619 | 1194 | 2738 | 4091 | 5165 |
| [u] | 418 | 963 | 2810 | 4278 | 5045 |
| MALE | | | | | |
| VOWEL | FORMANT | | | | |
| | F1 | F2 | F3 | F4 | F5 |
| [i] | 282 | 2096 | 2760 | 3338 | 3887 |
| [i] | 388 | 1742 | 2388 | 3355 | 3889 |
| [e] | 474 | 1750 | 2416 | 3357 | 3923 |
| [a] | 620 | 1315 | 2243 | 3379 | 3942 |
| [o] | 491 | 1025 | 2288 | 3323 | 3812 |
| [u] | 354 | 918 | 2156 | 3216 | 3708 |

**izolowane**

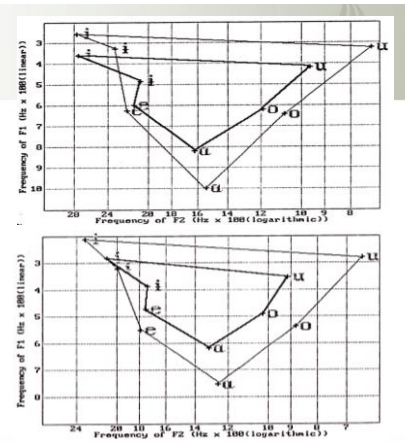
| FEMALE | | | | |
|---|---|---|---|---|
| VOWEL | FORMANTS | | | |
| | F1 | F2 | F3 | F4 |
| [i] | 247 | 2775 | 3510 | 4155 |
| [i] | 329 | 2335 | 3041 | 4214 |
| [e] | 625 | 2210 | 2979 | 3964 |
| [a] | 999 | 1545 | 2756 | 3951 |
| [o] | 641 | 1080 | 2793 | 3885 |
| [u] | 324 | 731 | 3140 | 4001 |
| MALE | | | | |
| VOWEL | FORMANTS | | | |
| | F1 | F2 | F3 | F4 |
| [i] | 213 | 2316 | 2971 | 3542 |
| [i] | 323 | 1996 | 2821 | 3422 |
| [e] | 553 | 1794 | 2561 | 3408 |
| [a] | 753 | 1261 | 2513 | 3422 |
| [o] | 538 | 883 | 2571 | 3326 |
| [u] | 280 | 652 | 2459 | 3232 |

Tab. J. Kleśta, *Allophonic variation of Polish Vowels*, w: Studia Phonetica Posnaniensia, vol. 6, Poznań 2000

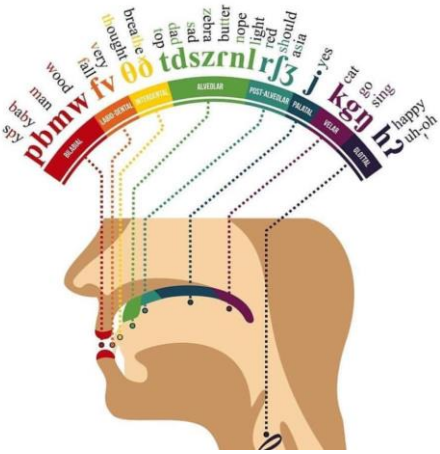
**Średnie pozycje formantów**

Kobiety

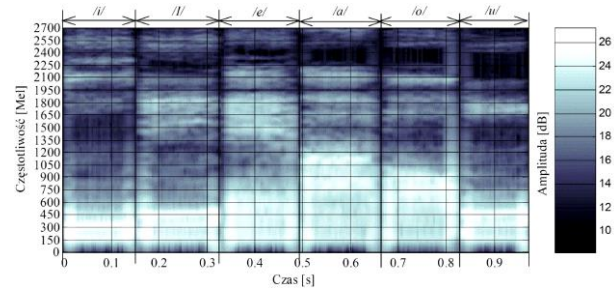
Mężczyźni



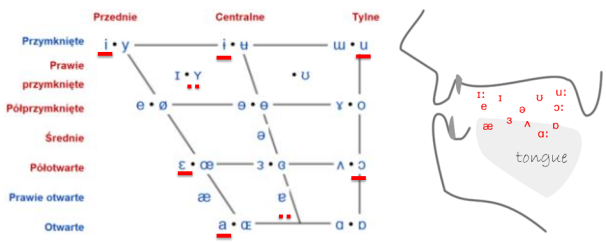

Tab. J. Kleśta, *Allophonic variation of Polish Vowels*, w: Studia Phonetica Posnaniensia, vol. 6, Poznań 2000




## Przykłady formantów widma szeregu samogłosek polskich




## Alfabet IPA – samogłoski




Źr.: Wikipedia [rev.],

## Przetwarzanie sygnału mowy

- Rejestracja sygnału
  - mikrofon, mikrofony
  - wzmacniacz, kompresja dynamiki ($L$>30dB)
  - przetwornik A/C z filtrem przeciwaliasingowym ($F_s$>8kHz, $Q$>8bit) $s(n)=s_a(t_0+n*dt)$, $n$=0, 1, 2, …
- Normalizacja …
- Redukcja szumu …
- Kształtowanie charakterystyki
  - kompresja dynamiki, kompandancja
  - equalizacja, preemfaza
- Detekcja – energia, charakterystyka, …
- Kodowanie, kompresja, parametryzacja, analiza, rozpoznawanie, synteza,…

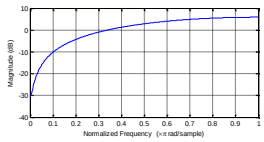
## Preemfaza

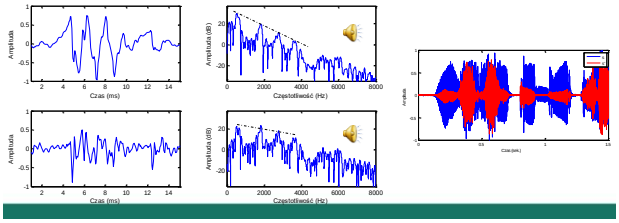
- Filtracja górnoprzepustowa FIR

$$H(z)=1-bz^{-1}$$

$$0.92<b<0.98$$

- Kompensacja charakterystyki emisji głosu
- Zwiększenie istotności wyższych formantów



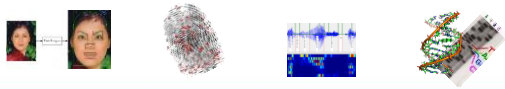




## Parametryzacja

- Zapis rozpoznawanych obiektów w postaci ich liczbowych identyfikatorów
- Przetwarzanie wstępne sygnału (*pre-processing,* filtracja, normalizacja, redukcja szumu itp…)
- Wybór i zdefiniowanie wielkości charakterystycznych dla danej klasy obiektów
- Ekstrakcja cech i ich agregacja w postaci wektora
- Przetwarzanie wektorów cech (*post-processing*)
  - *VarNorm*, *SoftMax*, PCA, SVD, …
- Transformacja, zmniejszenie wymiarowości, bez straty informacji

## Ekstrakcja cech

- Redukcja redundancji informacyjnej zapisu danego obiektu, przy zachowaniu jego cech istotnych z punktu widzenia możliwości dyskryminacyjnych.
- Przekleństwo wymiarowości: duża liczba cech wymaga dużej ilości danych ($N^L$). Proporcja $N/L>30$
- Wybór dokonany przez eksperta
- Metody automatyczne
  - Maksymalizacja zdolności dyskryminacyjnej
  - Minimalizacja błędu klasyfikacji
  - Maksymalizacja odległości międzyklasowej
  - Minimalizacja rozmiaru klas w przestrzeni
  - Analiza wariancji, statystyczne testy istotności, *test t*,…



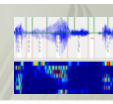
## Cechy sygnału mowy

- Prozodia
- Energia
- Widmowe
- Cepstralne
- Formantowe
- Artykulacyjne
- Czasowe
- Inne temporalne
- Supersegmentalne
- Entropowe
- Statystyczne

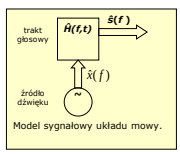
Metody parametryzacji:

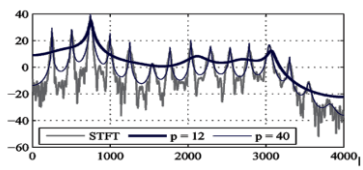
MFC, **MFCC**, LPC, **PLP**, RASTA, TRAPS, PWT, WPT, VQ, …

## Ekstrakcja cech sygnału mowy



- Zastosowanie: rozpoznawanie mowy, rozpoznawanie mówców (PLP, MFCC), kodowanie i kompresja mowy (CELP), diagnostyka, przetwarzanie (*vocoder*), …
- Mowa to zjawisko czasowo-częstotliwościowe, więc
- Analiza częstotliwościowa (STFT, FFT, Falki, LP) w krótkich (ok. 20-30 ms) okresach czasu (ramkach).
- Sygnał mowy:



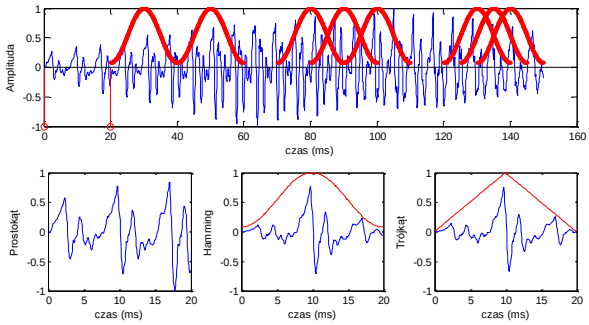

## Ramkowanie

- Konieczne zgromadzenie danych przed ich analizą
- Długość bufora / ramki
  - Zbyt długa – sygnał przestaje być stacjonarny
  - Zbyt krótka
    - problem w analizie niskich częstotliwości
    - brak okresu krtaniowego w ramce
    - niemożność ekstrakcji cech toru głosowego
    - silne efekty brzegowe
- Długość zazwyczaj 15-25 ms
- Zakładka (ang. *overlapp*, *frame-shift*): 1, ½, 3/4, 1/3, … ramki
- Okienkowanie (zmniejsza efekty brzegowe):

  Hamming, Gauss, trójkąt

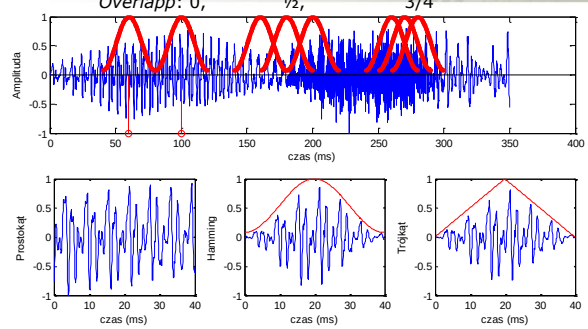
## Ramkowanie, 20ms

*Shift*: 1, ½, ¼
*Overlapp*: 0, ½, 3/4



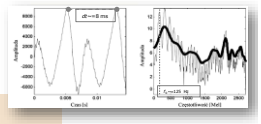

## Ramkowanie 40ms

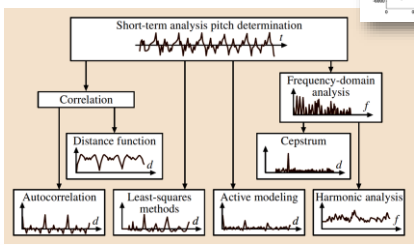
*Shift*: 1, ½, ¼
*Overlapp*: 0, ½, 3/4




## Ekstrakcja tonu krtaniowego $F_0$

- Zakres częstotliwości
  80-1000 Hz, (częściej 100-500 Hz)



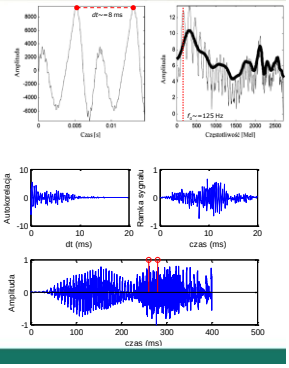



## Ekstrakcja tonu krtaniowego $F_0$

- Zakres częstotliwości
  80-1000 Hz, (częściej 100-500 Hz)
- Autokorelacja

$$s_w(m) = w(n-m)s(m)$$

$$r(l) = \sum_m s_w(m+l)s_w(m)$$



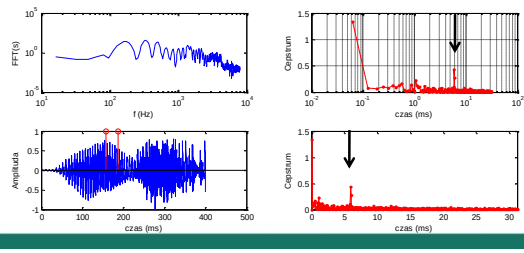

## Ekstrakcja tonu krtaniowego $F_0$

- *Cepstrum*

$$s_w^c = F^{-1}\left(\log\left(\left|F\left(s_w\right)\right|\right)\right)$$

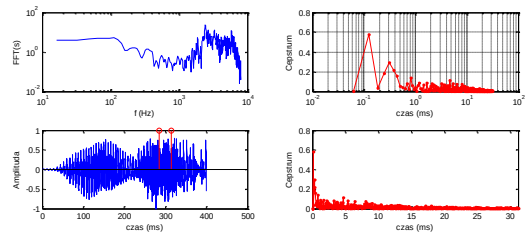
$$\hat{s}(f,t) = \hat{x}(f,t)\cdot\hat{H}(f,t)$$

$$\xrightarrow{\log} \hat{s}^c(f,t) = \hat{x}^c(f,t) + \hat{H}^c(f,t)$$



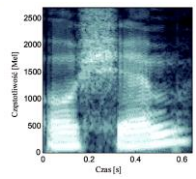

## Ekstrakcja tonu krtaniowego $F_0$

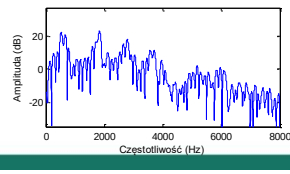
- *Cepstrum* – brak pobudzenia krtaniowego



## Ekstrakcja formantów

- Zakres częstotliwości:
  200Hz – 4000Hz
- skala Mel
- FFT – wygładzanie widma amplitudowego
- Predykcja liniowa
  – modelowanie traktu głosowego
- Inne modele
  – Statystyczne
  – Filtracyjne (*Z*)
  – AI
  – …




## Predykcja Liniowa

LP, ang. *Linear Prediction*

- Metoda modelowania cech traktu głosowego
- Umożliwia ekstrakcję formantów
- Zakładamy model sygnału $s(n)=\sum_{k=1}^{K} a(k)s(n-k)+\underline{Ge(n)}$
- Idea predykcji
  – ze znanych próbek sygnału wyznacz kolejne $\tilde{s}(n)=\sum_{k=1}^{K}\alpha(k)s(n-k)$
  – błąd predykcji $d(n)=s(n)-\tilde{s}(n)=s(n)-\sum_{k=1}^{K}\alpha(k)s(n-k)$
  – to odpowiedź FIR o transmitancji

$$A(z)=1-\sum_{k=1}^{K}\alpha(k)z^{-k} \quad \text{więc} \quad H_{t_0}(z)=\frac{G}{1-\sum_{k=1}^{K}\alpha(k)z^{-k}}$$

## Predykcja liniowa

wyznaczanie współczynników (LPC)

- Szukamy $\boldsymbol{d}\equiv G\boldsymbol{e}\Leftrightarrow\boldsymbol{\alpha}\equiv\boldsymbol{a}$
- Kryterium najmniejszych kwadratów

$$J=\left\|\left\{s_w(m)-\sum_{k=1}^{K}\alpha(k)s_w(m-k)\right\}_m\right\|^2 \quad \overset{\min_{\alpha} J}{\Rightarrow} \quad \boldsymbol{\Phi}_{i,k}\boldsymbol{\alpha}=\boldsymbol{\psi}$$

- Metoda kowariancji
  (niewygodna, rozkład macierzy kowariancji)
- Metoda autokorelacji
  – Stosujemy okienkowanie (określone granice estymacji)
  – Ustalone *K* $\quad J=\sum_{m=m_0-M}^{m_0+M+K} d^2(m)$

## Predykcja liniowa

wyznaczanie współczynników

Macierz

$$\boldsymbol{\Phi}_{i,k}:\ \varphi(i,k)=\sum_{m=m_0-M-i}^{m_0+M-i}s_w(m)s_w(m+i-k)=\sum_{m=m_0-M-k}^{m_0+M-k}s_w(m)s_w(m+k-i)\ \Rightarrow\ \varphi^T=\varphi$$

Więc jest to wektor autokorelacji

$$\varphi(i,k)=\varphi(|i-k|)\ \Rightarrow\ \varphi(k)=\varphi(-k)=\sum_m s_w(m)s_w(m+k)$$

Po wstawieniu do warunku min{*J*}, otrzymujemy:

$$\sum_{k=1}^{K}\alpha(k)\varphi(|i-k|)=\varphi(i),\quad i=1,2,\ldots,K$$

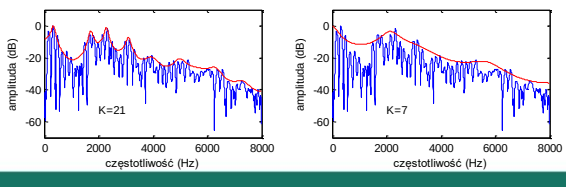
czyli

$$\begin{bmatrix}\varphi(0) & \varphi(1) & \cdots & \varphi(K-1)\\ \varphi(1) & \varphi(0) & \cdots & \varphi(K-2)\\ \cdots & \cdots & \cdots & \cdots\\ \varphi(K-1) & \varphi(K-2) & \cdots & \varphi(0)\end{bmatrix}\begin{bmatrix}\alpha(1)\\ \cdots\\ \cdots\\ \alpha(K)\end{bmatrix}=\begin{bmatrix}\varphi(1)\\ \cdots\\ \cdots\\ \varphi(K)\end{bmatrix}$$

## Charakterystyka filtru predykcyjnego

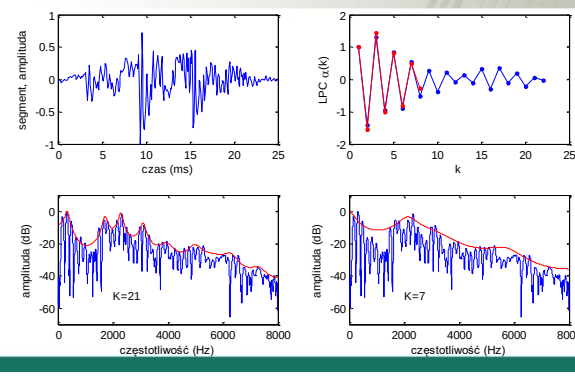
$$|H(f/f_s)|=\left|\frac{G}{1-\sum_{k=1}^{K}\alpha(k)e^{-2\pi jf/f_s}}\right|$$

- Pozwala wyznaczyć obwiednię widma segmentu
- Stała obwiedni (dobroć) zależy od *K* - rzędu predyktora
- Maksima lokalne obwiedni – formanty

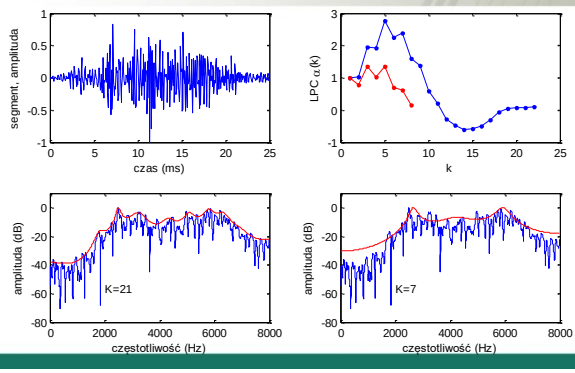



## LPC

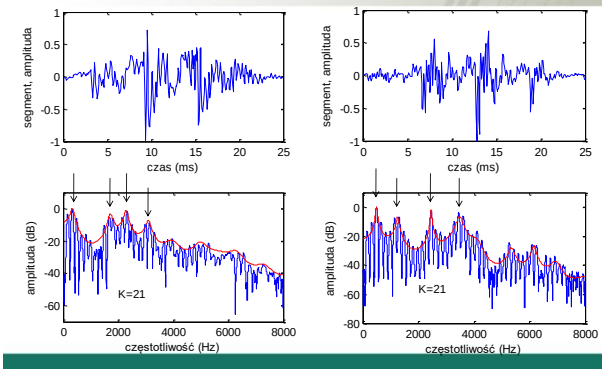
przykłady – głoska dźwięczna



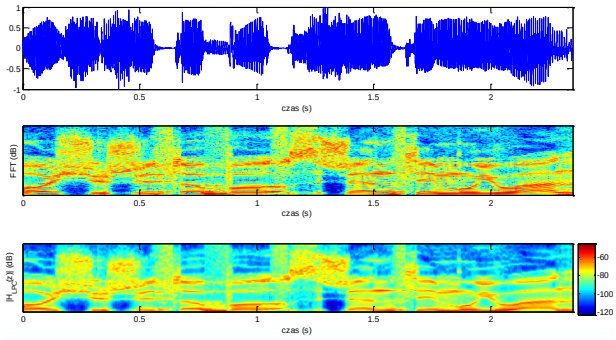
## LPC
przykłady - głoska bezdźwięczna



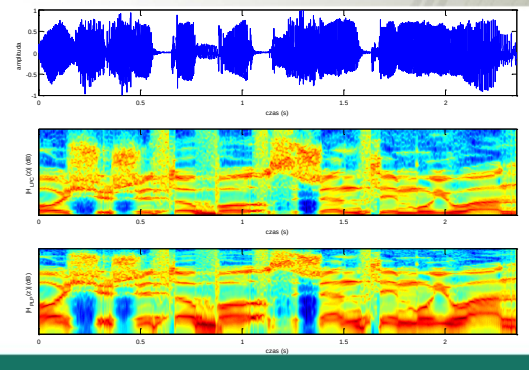

## LPC
przykłady - formanty



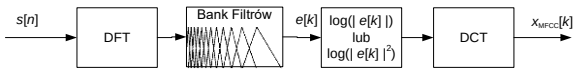

## FFT vs LPC
*„Woź żyto bo zżęty mój łan, (…)”*

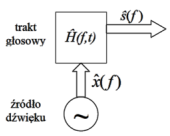



## PLP (ang. *Perceptual Linear Prediction*)
przykład – widmo LP w skali melowej
*„Woź żyto bo zżęty mój łan, (…)”*




## Schemat wyznaczania MFCC (*Mel-frequency cepstral coefficients*)

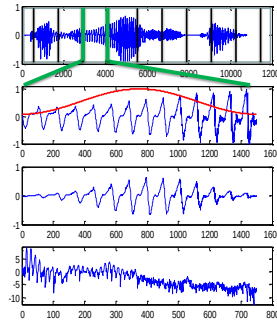



$$c[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log\left|\hat{s}(e^{j\omega})\right| e^{j\omega n} d\omega$$

$$\hat{s}(f) = \hat{x}(f)\cdot\hat{H}(f,t)$$

## Schemat wyznaczania MFCC (*Mel-frequency cepstral coefficients*)
**kroki 1-4**



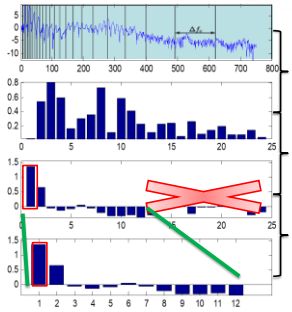
1. DSP: A/C, redukcja szumu, preemfaza (filtracja $H(z)=1-az^{-1}$, $a=0.96$), normalizacja amplitudy
2. Ramkowanie (ok. 20 ms)
3. Okienkowanie (np. okno Hamminga)

   $\boldsymbol{s}_w = \boldsymbol{s}.*\boldsymbol{w}$
4. Wyznaczenie widma amplitudowego

   $\boldsymbol{s}^* = |\mathrm{DFT}\{\boldsymbol{s}\}|$

## Schemat wyznaczania MFCC (*Mel-frequency cepstral coefficients*)

**kroki 5-8**

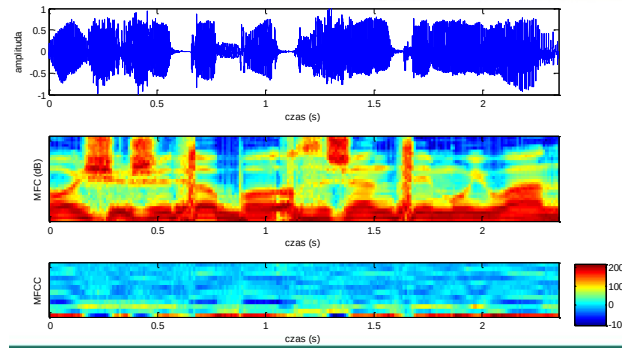


5. Filtracja bankiem filtrów melowych (np. 24 pasma) $\boldsymbol{s}'=H_{\text{mel}}\{\boldsymbol{s}^*\}$
6. Wyznaczenie (logarytmu) energii każdego pasma
   $\text{MFC}=\log(|\boldsymbol{s}'|)$
7. Transformacja kosinusowa DCT (*cepstrum*) dekorelująca
   $\text{MFCC}=\text{DCT}\{\text{MFC}\}$
8. Usunięcie niepotrzebnych elementów (*liftering*)
   $\text{MFCC}:=\text{MFCC}(\{1,2\}:12)$

## Przykład parametryzacji MFCC

(32 MFC, 12 MFCC \ $c_0$, bez preemfazy)

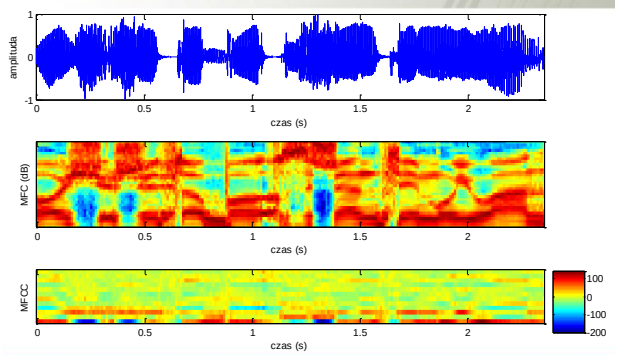
*„Woź żyto bo zżęty mój łan, (…)”*




## Przykład parametryzacji MFCC

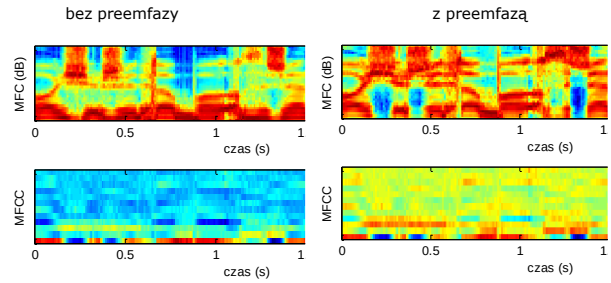
(32 MFC, 12 MFCC \ $c_0$, z preemfazą)

*„Woź żyto bo zżęty mój łan, (…)”*

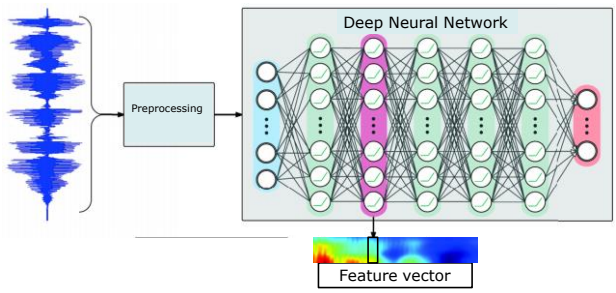



## Przykład parametryzacji MFCC

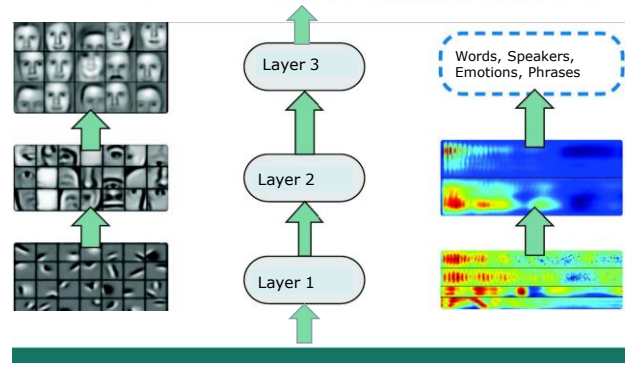
*„Woź żyto bo zżęty mój łan, (…)”*




## Bottleneck Features – Deep Neural Networks




## Bottleneck Features – Deep Neural Networks



## **Rozpoznawanie wzorców,** modelowanie, uczenie maszynowe

| Parametryzacja | Model | Ewaluacja |
|---|---|---|
| • Ekstrakcja cech<br>• Normalizacja<br>• Redukcja wymiarowości | • Architektura<br>• Trening modelu<br>• Predykcja | • Analiza<br>• Statystyki<br>• Testowanie<br>• Interpretacja |

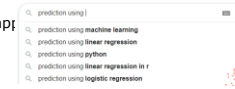
## Types of ML predictions

### Classification


Predicts category (class, label)
Known labelled input x : {c1, c2, …,}

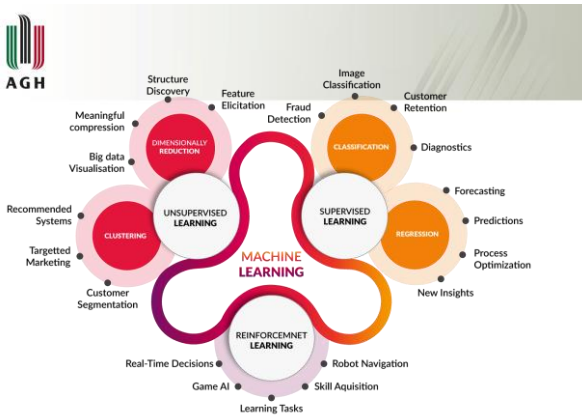
### Regression

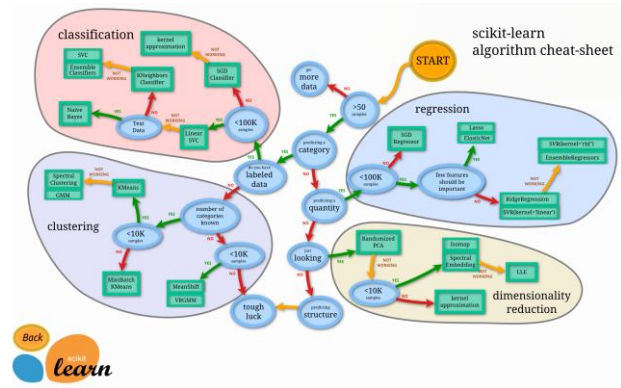
Predicts value
Known I/O value mapp
y=f(x)




### Clustering

Groups similar elements (no labels)




## Classification

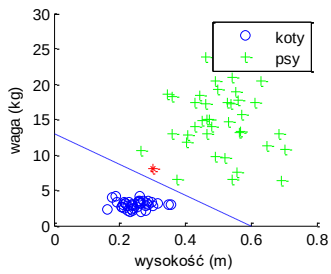

Predict known label for unseen data

***x*** *- feature vector*

$$\boldsymbol{x} = [x_1, \ldots, x_L]^T$$

$c_i$ *–class, label* from the set of *M* classes, $i$=1,…,$M$

$$\boldsymbol{x} \xrightarrow{?} c_i, \quad \bigcup_{i=1}^{M} c_i = \Omega$$




## Bayes Classifier (MAP)

MAP (*Maximum a Posteriori prob.*): we are looking for a specific label $c_i^*$

$$c_i^* = \arg\max_{c_i, i=1,\ldots,M} P(c_i \mid \boldsymbol{x})$$

That would maximize the observed probability of the class given the input ***x***.

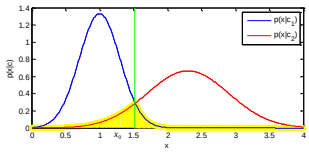
Thomas Bayes
(1702-1761)

Binary calssification: $M$=2
Class priors: $P(c_1)$, $P(c_2)$
If we knew likelihood $p(\boldsymbol{x}|c_i)$:

$$P(c_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid c_i) P(c_i)}{p(\boldsymbol{x})}$$

## Binary Naive Bayes

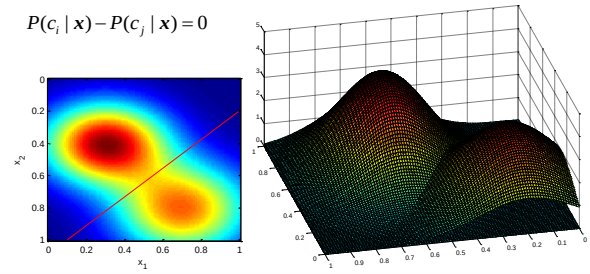
$$p(\boldsymbol{x}\,|\,c_1)P(c_1) \overset{?}{>\!<} p(\boldsymbol{x}\,|\,c_2)P(c_2)$$




## Naive Bayes – 2D

Dyscrimination Hyperplane

$$P(c_i\,|\,\boldsymbol{x}) - P(c_j\,|\,\boldsymbol{x}) = 0$$



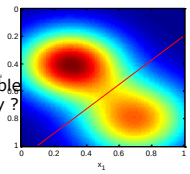

## Decision function (Fischer discriminator)

Monotonic function of MAP

$$g_i(\boldsymbol{x}) \equiv f\left(P(c_i\,|\,\boldsymbol{x})\right) \qquad \forall\, j \neq i \quad g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}) \Rightarrow \boldsymbol{x} \in c_i$$

Decision plane

$$g_{ij}(\boldsymbol{x}) \equiv g_i(\boldsymbol{x}) - g_j(\boldsymbol{x}) = 0, \quad i, j = 1, \ldots, M,\ i \neq j$$



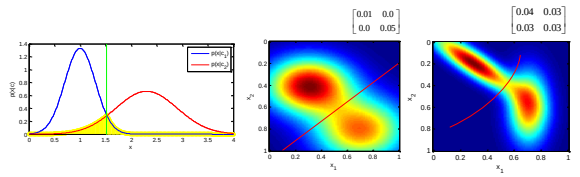

problem only ?

How to tackle an *N*-class using binary classifiers

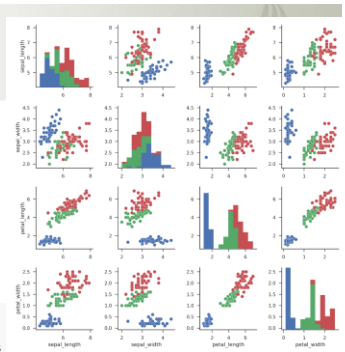
## Naive Bayes – Normal PDF

Multivariate N-PDF $p(\boldsymbol{x}\,|\,c_i) = \dfrac{1}{(2\pi)^{l/2}|\Sigma_i|^{1/2}} \exp\left(-\dfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right)$

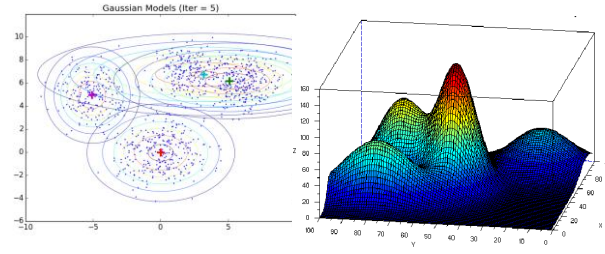
$$\boldsymbol{\mu}_i = E[\boldsymbol{x}] \qquad \Sigma_i = E\left[(\boldsymbol{x}-\boldsymbol{\mu}_i)(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\right] \qquad \boldsymbol{\Theta}_i = \{\boldsymbol{\mu}_i, \Sigma_i\}$$

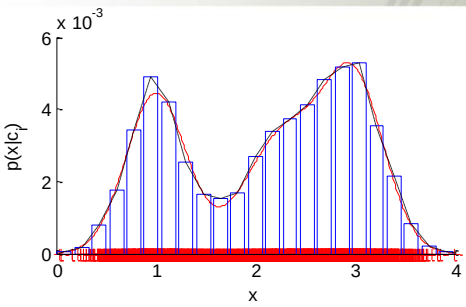







```
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> from sklearn.naive_bayes import GaussianNB
>>> gnb = GaussianNB()
>>> y_pred = gnb.fit(iris.data, iris.target).predict(iris.data)
>>> print("Number of mislabeled points out of a total %d points : %d"
... % (iris.data.shape[0],(iris.target != y_pred).sum()))
Number of mislabeled points out of a total 150 points : 6
```



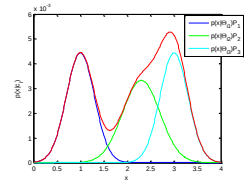



## Gaussian Mixture Model

Carl Friedrich Gauss (1777-1855)

GMM – statistical model of complex distributions

$$p(\boldsymbol{x}\,|\,c_i)=\sum_{j=1}^{J} p(\boldsymbol{x}\,|\,\boldsymbol{\Theta}_{ij})P_j$$



$$\sum_{j=1}^{J} P_j = 1, \quad \int_{x\in\boldsymbol{X}} p(\boldsymbol{x}\,|\,\boldsymbol{\Theta}_{ij})d\boldsymbol{x} = 1$$

$$\max\left\{ p^{ML} = \prod_k p(\boldsymbol{x}_k, \boldsymbol{\Theta}, P_1, \ldots, P_J) \right\}$$

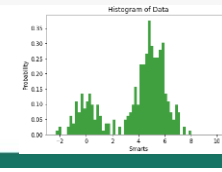
mixture.GaussianMixture

```
import numpy as np
from sklearn import mixture
np.random.seed(1)
g = mixture.GaussianMixture(n_components=3)
# Generate random observations with two modes centered on 0
# and 10 to use for training.
obs = np.concatenate((np.random.randn(100, 1), 10 + np.random.randn(300, 1), 5 + np.random.randn(200, 1)))
g.fit(obs)
l=g.predict([[0], [2], [6], [9], [10]])
print(l)
p=g.predict_proba([[0], [2], [6], [9], [10]])
print(p)
```

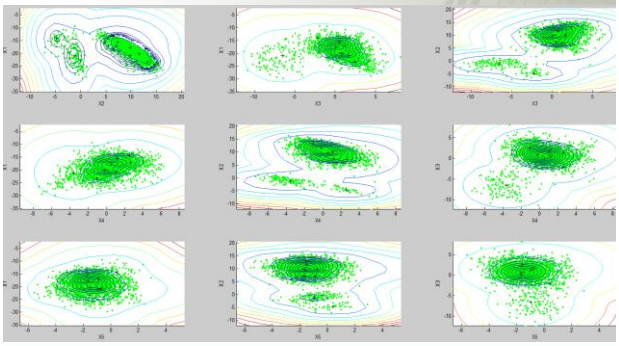
```
[1 1 2 0 0]
[[1.74450697e-23 9.99985504e-01 1.44962481e-05]
 [2.78709519e-14 8.18856673e-01 1.81143327e-01]
 [3.62624405e-04 3.43844650e-10 9.99637375e-01]
 [9.98599654e-01 3.64032460e-22 1.40034569e-03]
 [9.99984542e-01 1.86890101e-27 1.54577593e-05]]
```

```
import numpy as np
from sklearn import mixture
np.random.seed(1)
g1 = mixture.GaussianMixture(n_components=3)
g2 = mixture.GaussianMixture(n_components=3)
# Generate random observations with two modes centered on 0
# and 10 to use for training.
obs1 = np.concatenate((np.random.randn(100, 1), 5 + np.random.randn(300, 1))) # 0 & 5
obs2 = np.concatenate((2.5 + np.random.randn(100, 1), 6 + np.random.randn(200, 1))) # 2.5 & 6
g1.fit(obs1)
g2.fit(obs2)
p1=g1.score([[4]])
print("log p(x=4|g1)=",p1)
p2=g2.score([[4]])
print("log p(x=4|g2)=",p2)
p1=g1.score([[2]])
print("log p(x=2|g1)=",p1)
p2=g2.score([[2]])
print("log p(x=2|g2)=",p2)
print("What is P(g|x)=? , Bayes")
print("P(g1|x=2)=", np.exp(p1)*0.5/(np.exp(p1)*0.5+np.exp(p2)*0.5))
print("P(g2|x=2)=", np.exp(p2)*0.5/(np.exp(p1)*0.5+np.exp(p2)*0.5))
```

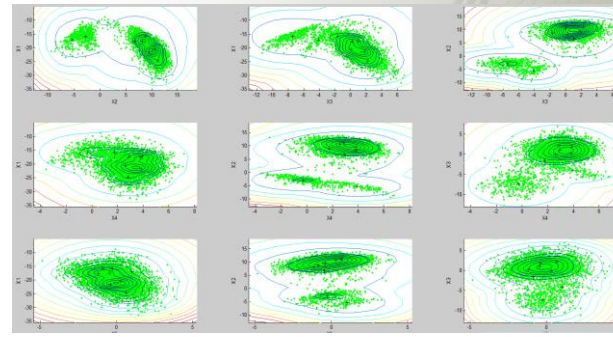
```
log p(x=4|g1)= -1.7072020753278263
log p(x=4|g2)= -2.2636967356701074
log p(x=2|g1)= -4.153287400767302
log p(x=2|g2)= -2.212724778571532
What is P(g|x)=? , Bayes
P(g1|x=2)= 0.1255860595922101
P(g2|x=2)= 0.8744139404077899
```



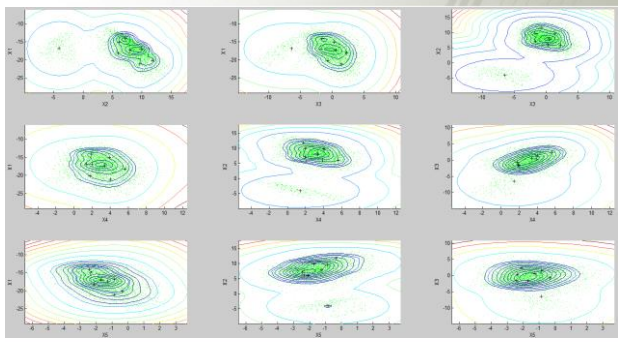

## GMM w działaniu
## Estymacja K-Means EM na DCT(log(X)), widok 3sigma, /u/, 8GMM



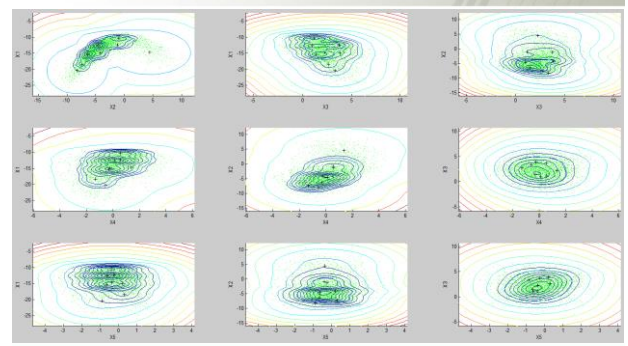
## GMM w działaniu
## Estymacja K-Means EM na DCT(log(X)), widok 3sigma, /m/, 8GMM



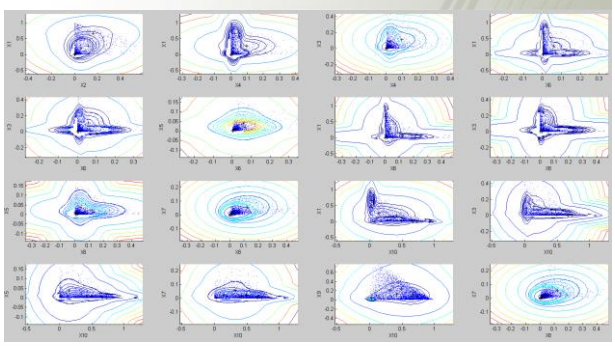
## GMM w działaniu
## Estymacja K-Means EM na DCT(log(X)), widok 3sigma, /l/, 8GMM



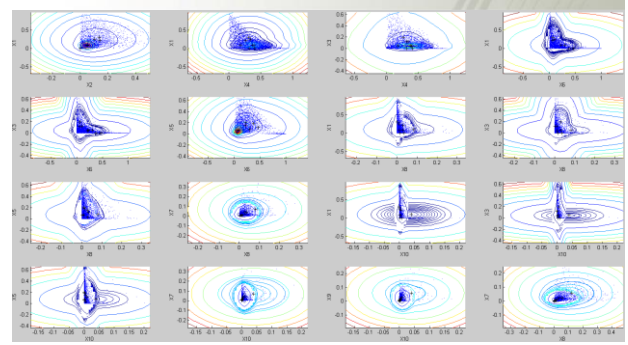
## GMM w działaniu
## Estymacja K-Means EM na DCT(log(X)), widok 3sigma, /s/, 8GMM



## GMM w działaniu
## Estymacja K-Means EM na X, widok 3sigma, /s/, 8GMM

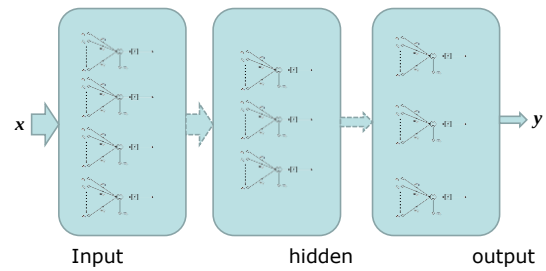


## GMM w działaniu
## Estymacja K-Means EM na X, widok 3sigma, /a/, 8GMM



## Artificial Neural Networks

## Multilayer Perceptron



## Rosenblatt perceptron

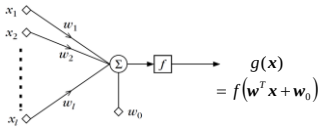
Frank Rosenblatt
(1928-1971)

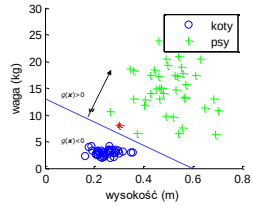
Algebraic function which defines a discriminating hyperplane

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0, \quad \boldsymbol{w} = [w(1), \ldots, w(L)]$$

$$g_{ij} = \boldsymbol{w}^T (\boldsymbol{x}_i - \boldsymbol{x}_j) = 0$$

$\boldsymbol{w}$ - hyperplane normal vector



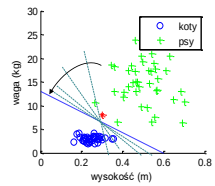


http://meta.stackoverflow.com/questions/30451/advertising-inquiry-a-platform-to-train-machine

## Neural weights estimation

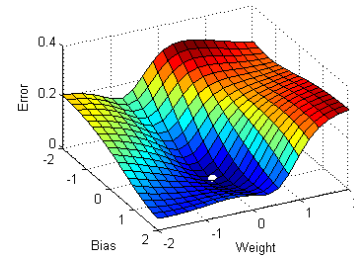
- Iterative perceptron algorithm

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \rho_t \sum_{x \in \boldsymbol{X}_{tr}} \delta_x \boldsymbol{x}, \quad \rho_t \in \Re^+, \quad \delta_x = \begin{cases} -1, & x \in c_1 \\ 1, & x \in c_2 \end{cases}$$




- Gradient descent

$$w' = w - L * dF(w,X))/dw$$



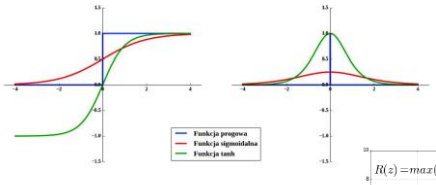

## FFN/MLP, one-hot classifier

```
>>> from sklearn.neural_network import MLPClassifier
>>> X = [[0., 0.], [1., 1.]]
>>> y = [0, 1]
>>> clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
...                     hidden_layer_sizes=(5, 2), random_state=1)
...
>>> clf.fit(X, y)
MLPClassifier(activation='relu', alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(5, 2),
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
>>> clf.predict([[2., 2.], [-1., -2.]])
array([1, 0])
```

```
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD

# Generate dummy data
import numpy as np
x_train = np.random.random((1000, 20))
y_train = keras.utils.to_categorical(np.random.randint(10, size=(1000, 1)), num_classes=10)
x_test = np.random.random((100, 20))
y_test = keras.utils.to_categorical(np.random.randint(10, size=(100, 1)), num_classes=10)

model = Sequential()
# Dense(64) is a fully-connected layer with 64 hidden units.
# in the first layer, you must specify the expected input data shape:
# here, 20-dimensional vectors.
model.add(Dense(64, activation='relu', input_dim=20))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy',
              optimizer=sgd,
              metrics=['accuracy'])

model.fit(x_train, y_train,
          epochs=20,
          batch_size=128, verbose=0)
score = model.evaluate(x_test, y_test, batch_size=128)
print(score)
```

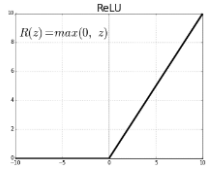
```
100/100 [==============================] - 0s 704us/step
[2.2926833629608154, 0.12999999523162842]
```

## Activation functions and derivatives






$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$$



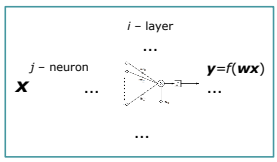

## Softmax network outputs

- Usually used as the network output activation
- Trained to produce probability estimation
- 1 example – one class only (no multivariate class problems, one-hot approach)
- Commonly trained under a log loss (or cross-entropy) criterion


$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_i^C t_i log(f(s)_i)$$

$$CE = -log\left(\frac{e^{s_p}}{\sum_j^C e^{s_j}}\right)$$

## *Backpropagation* - assumptions




$$\boldsymbol{w}_{ij}^{t+1} = \boldsymbol{w}_{ij}^{t} + \Delta \boldsymbol{w}_{ij}$$

***w*** – weights
*i* – layer (*i*=1,...,*I*)
*j*,*m* – neuron, synapse (*j*,*m*=1,...,$N_i$)
*k* – sample (*k*=1,...,*K*)
*μ* – learning rate
*J* – loss function
***y*** – expected outputs
*δ* – gradient

$$\Delta \boldsymbol{w}_{ij} = -\mu \frac{\delta J}{\delta \boldsymbol{w}_{ij}}$$

$$J = \frac{1}{2}\sum_{k=1}^{K}\sum_{m=1}^{M}\left(y_k(m) - \hat{y}_k(m)\right)^2$$

$$\Delta \boldsymbol{w}_{ij} = -\mu \sum_{k=1}^{K} \delta_{ij,k} \boldsymbol{y}_{i-1,k}$$

$$\delta_{ij,k} = \frac{\delta\left[\sum_{m=1}^{M}\left(y_k(m) - \hat{y}_k(m)\right)^2\right]}{\delta\left(\boldsymbol{w}_{ij,k} \cdot \boldsymbol{y}_{i-1,j,k}\right)}$$

Rys. [5]

## *Backpropagation* - solution

$$\boldsymbol{w}_{ij}^{t+1} = \boldsymbol{w}_{ij}^{t} - \mu \sum_{k=1}^{K} \delta_{ij,k} \boldsymbol{y}_{i-1,k}$$

$$f(x) = \frac{1}{1+\exp(-\alpha x)}$$

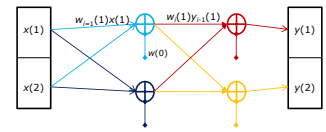
$$f'(x) = \alpha f(x)(1 - f(x))$$

Solution for *J*=*L**MSE(***y***-***y***^)

$$i = I: \quad \delta_{i=I,j,k} = \left[f\left(\boldsymbol{w}_{ij}\boldsymbol{y}_{i-1,k}\right) - \hat{y}_k(j)\right] f'\left(\boldsymbol{w}_{ij}\boldsymbol{y}_{i-1,k}\right)$$

$$i < I: \quad \delta_{i-1,j,k} = \left(\sum_{m=1}^{N_i} \delta_{im} w_{im}(j)\right) f'\left(\boldsymbol{w}_{i-1,j}\boldsymbol{y}_{i-2,k}\right)$$

***w*** – weights
*i* – layer (*i*=1,...,*I*)
*j*,*m* – neuron, synapse (*j*,*m*=1,...,$N_i$)
*k* – sample (*k*=1,...,*K*)
*μ* – learning rate
*J* – loss function
***y*** – expected outputs
*δ* – gradient




## *Backpropagation* - implementation

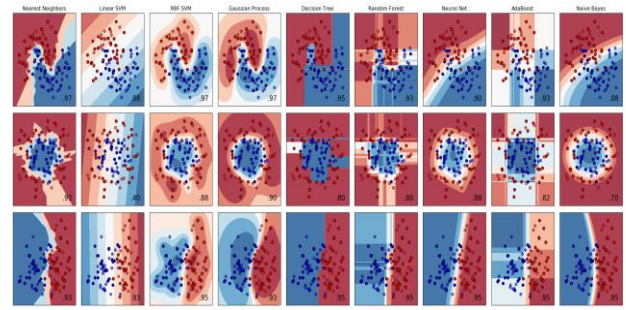
1. Inicjalizacja – wylosuj wszystkie wagi $\boldsymbol{w}^{t=0}$
2. Wyznacz odpowiedzi $\boldsymbol{y}_{i,k}=f(\boldsymbol{w}\boldsymbol{y}_{i-1,k})$ wszystkich neuronów sieci dla każdego wektora treningowego (*forward step*)
3. Oblicz funkcję kosztu *J* (krok pośredni)
4. *i=I*, Dla wszystkich wzorców (*k*) wyznacz wartość $\delta_{Ij,k}$ każdego neuronu wyjściowego (*backward step*)
5. Dla wszystkich warstw (kolejno wstecz) *i*=*I*,*I*-1,...,2 wyznacz $\delta_{i-1,\ j,k}$ wszystkich neuronów dla każdego wzorca (*k*)
6. Ustaw wszystkie wagi $$\boldsymbol{w}_{ij}^{t+1} = \boldsymbol{w}_{ij}^{t} - \mu \sum_{k=1}^{K} \delta_{ij,k} \boldsymbol{y}_{i-1,k}$$
7. t=t+1, *goto* 2, *until* $J<J_{tr}$ OR $dJ/d\boldsymbol{w}<e$ OR $t>T$

## Stochastic gradient descent

- GD Problems
  - RAM
  - Computational power
- SGD Solutions
  - Batch sample
  - Batch-size
  - Epochs

- https://losslandscape.com/gallery/

## Comparison of classifiers



## Model evaluation

- How good is our model ?
- Generalization, Robustness
- Loss Function
- Performance: Accuracy, Error rate (%)

## *N*-fold Cross-validation

$$ERR = \frac{1}{N}\sum_{n=1}^{N} ERR_n$$

1. $K=K_{trening}+K_{test}$ — $K_{test}$ $K_{trening}$
2. $K=K_{trening}+K_{test}$ — $K_{test}$ $K_{trening}$

...

... $K=K_{trening}+K_{test}$

*N* $K_{trening}$ $K_{test}$

## *Leave-one-out*

1. $K=K_{trening}+K_{te}$
2. $K_{test}=1$ $K_{trening}$
3. $K_{test}=1$ $K_{trening}$

... itd...
do *K* ...

$K_{trening}$ $K_{test}=1$

Maksymalizacja mocy zbioru testowego i treningowego jednocześnie

## Overfitting

- Train/Eval mismatch
- Data problem
- Model problem
- Optimizer problem

## How to avoid overfitting

- Better Data (quantity, quality, representativeness)
- Data augumentation
- Feature engineering
- Hyper-parameter optimization (x-val)
- Model (architecture and size change)
- Regularization L1, L2
- Batch-Normalization
- Dropout
- Shorter training