Chapter 5

Independent Component Analysis

Part II: Algorithms

max planck institut
informatik

ICA definition

- Given n observations of m random variables in matrix $\boldsymbol{X}$, find n observations of m independent components in $\boldsymbol{S}$ and m-by-m invertible mixing matrix $\boldsymbol{A}$ s.t. $\boldsymbol{X} = \boldsymbol{S}\boldsymbol{A}$
    - Components are statistically independent
    - At most one is Gaussian
    - We can assume $\boldsymbol{A}$ is orthogonal (by whitening $\boldsymbol{X}$)

Maximal non-Gaussian

Skillikorn chapter 7; Hyvärinen & Oja 2000

Central limit theorem

- Average of i.i.d. variables converges to normal distribution
  - $\sqrt{n}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - \mu\right) \xrightarrow{d} N(0, \sigma^2)$ as $n \to \infty$
- Hence $(X_1 + X_2)/2$ is "more Gaussian" than X_1 or X_2 alone
  - For i.i.d. zero-centered non-Gaussian X_1 and X_2
- Hence, we can try to find components s that are "maximally non-Gaussian"

Re-writing ICA

- Recall, in ICA $\boldsymbol{x} = \boldsymbol{sA} \Leftrightarrow \boldsymbol{s} = \boldsymbol{xA}^{-1}$
  - Hence, s_j is a linear combination of x_i
- Approximate $s_j \approx y = \boldsymbol{xb}^T$ ($\boldsymbol{b}$ to be determined)
  - Now $y = \boldsymbol{sAb}^T$ so y is a lin. comb. of $\boldsymbol{s}$
  - Let $\boldsymbol{q}^T = \boldsymbol{Ab}^T$ and write $y = \boldsymbol{xb}^T = \boldsymbol{sq}^T$

More re-writings

- Now $s_j \approx y = \boldsymbol{x}\boldsymbol{b}^T = \boldsymbol{s}\boldsymbol{q}^T$
- If $\boldsymbol{b}^T$ is a column of $\boldsymbol{A}^{-1}$, $s_j = y$ and $q_j = 1$ and $\boldsymbol{q}$ is 0 elsewhere
- CLT: $\boldsymbol{s}\boldsymbol{q}^T$ is least Gaussian when $\boldsymbol{q}$ looks correct
  - We don't know $\boldsymbol{s}$, so we can't vary $\boldsymbol{q}$
  - But we can vary $\boldsymbol{b}$ and study $\boldsymbol{x}\boldsymbol{b}^T$
- **Approach**: find $\boldsymbol{b}$ s.t. $\boldsymbol{x}\boldsymbol{b}^T$ is least Gaussian

Kurtosis

- One way to measure how Gaussian a random variable is is its **kurtosis**
  - $\text{kurt}(y) = E[(y - \mu)^4] - 3(E[(y - \mu)^2])^2$
    - $E[y] = \mu$
    - Normalized version of the fourth central moment $E[(y - \mu)^4]$
- If $y \sim N(\mu, \sigma^2)$, $\text{kurt}(y) = 0$, most other distributions have non-zero kurtosis (positive or negative)

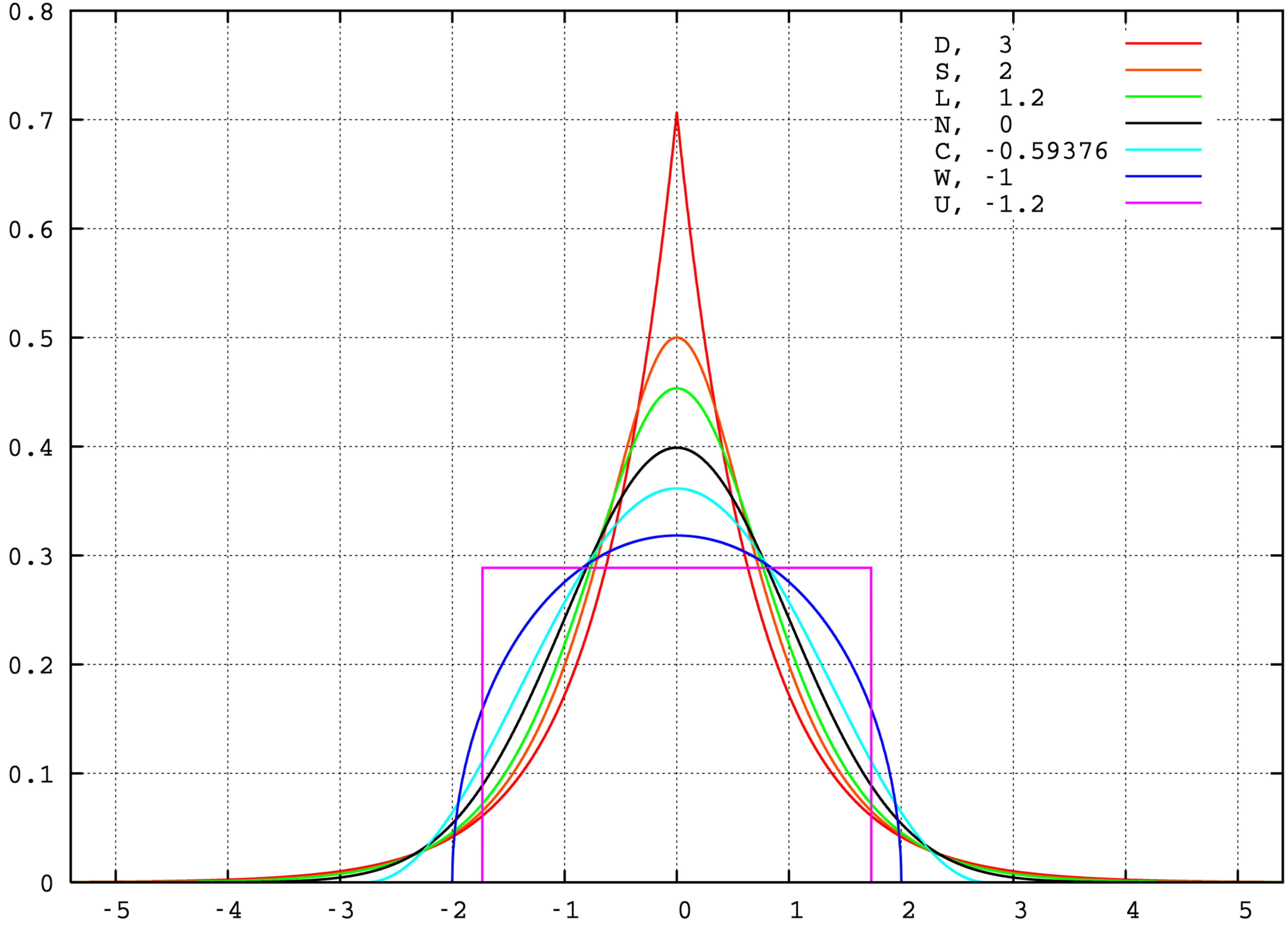
Computing with kurtosis

- If x and y are independent random variables:
  - $\text{kurt}(x + y) = \text{kurt}(x) + \text{kurt}(y)$
    - Homework
- If α is a constant:
  - $\text{kurt}(\alpha x) = \alpha^4 \text{kurt}(x)$
    - $E[(\alpha x)^4] - 3(E[(\alpha x)^2])^2 = \alpha^4 E[x^4] - \alpha^4 3(E[x^2])^2$

Sub- and super-Gaussian distributions

- Distributions with negative kurtosis are **sub-Gaussian** (or **platykurtic**)
  - Flatter than Gaussian
- Distributions with positive kurtosis are **super-Gaussian** (or **leptokurtic**)
  - Spikier than Gaussian

Examples



https://en.wikipedia.org/wiki/Kurtosis#/media/File:Standard_symmetric_pdfs.png

Negentropy

- Another measure of non-Gaussianity
- Entropy of discrete r.v. X is $H(X) = -\sum_i \Pr[X=i] \log \Pr[X=i]$
- The differential entropy of continuous random vector $\boldsymbol{x}$ with density $f(\boldsymbol{x})$ is $H(\boldsymbol{x}) = -\int f(\boldsymbol{x}) \log f(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$
  - Gaussian $\boldsymbol{x}$ has the largest entropy over all random variables of equal variance
- Negentropy is $J(\boldsymbol{x}) = H(\boldsymbol{x}_{\text{Gauss}}) - H(\boldsymbol{x})$
  - $\boldsymbol{x}_{\text{Gauss}}$ is a Gaussian r.v. of the same covariance matrix as $\boldsymbol{x}$

Approximating negentropy

- Computing the negentropy requires estimating the (unknown) pdfs
- It can be approximated as
  $J(y) \approx \sum_i k_i(\mathrm{E}[G_i(y)] - \mathrm{E}[G_i(v)])^2$
  - $v \sim N(0, 1)$, k_i are positive constants and G_i are some non-quadratic functions
    - With only one function $G(y) = y^4$, this is kurtosis
- One choice: $G_1(y) = \log(\cosh(ay))/a$, $G_2(y) = -\exp(-y^2/2)$

Back to optimization (using kurtosis)

- Recall: with two components
  $y = \boldsymbol{x}\boldsymbol{b}^{\mathrm{T}} = \boldsymbol{s}\boldsymbol{q}^{T} = q_1 s_1 + q_2 s_2$
  - s_i have unit variance
  - We want to find $\pm\boldsymbol{b} = \operatorname{argmax} |\operatorname{kurt}(\boldsymbol{x}\boldsymbol{b}^{T})|$
    - We can't determine the sign
- We want y to be either s_1 or s_2, hence
  $\mathrm{E}[y^2] = q_1^{\,2} + q_2^{\,2} = 1$

Whitening, again

- Generally, $||\boldsymbol{q}||^2 = 1$
- Recall: $\boldsymbol{Z} = \boldsymbol{U} = \boldsymbol{X}\boldsymbol{V}\boldsymbol{\Sigma}^{-1}$ is the whitened version of $\boldsymbol{X}$
- Target becomes $\pm\boldsymbol{w} = \text{argmax } |\text{kurt}(\boldsymbol{z}\boldsymbol{w}^T)|$
- Now $\|\boldsymbol{q}\|_2^2 = (\boldsymbol{w}\boldsymbol{U}^T)(\boldsymbol{U}\boldsymbol{w}^T) = \|\boldsymbol{w}\|_2^2$
  - Hence we have constraint $||\boldsymbol{w}||^2 = 1$

Gradient-based algorithm

- Gradient with kurtosis is

$$\frac{\partial|\text{kurt}(\boldsymbol{z}\boldsymbol{w}^T)|}{\partial \boldsymbol{w}} = 4\,\text{sign}(\text{kurt}(\boldsymbol{z}\boldsymbol{w}^T))\left(E[(\boldsymbol{z}\boldsymbol{w}^T)^3\boldsymbol{z}] - 3\boldsymbol{w}\,\|\boldsymbol{w}\|_2^2\right)$$

  - $\mathrm{E}[(\boldsymbol{z}\boldsymbol{w}^T)^2] = ||\boldsymbol{w}||^2$ for whitened data

- We can optimize this using standard gradient methods

  - To satisfy the constraint $||\boldsymbol{w}||^2 = 1$, we divide $\boldsymbol{w}$ with its norm after every update

FastICA for one IC and kurtosis

- Noticing that $||\boldsymbol{w}||^2 = 1$ by constraint and taking infinite step update, we get $\boldsymbol{w} \leftarrow \mathrm{E}[(\boldsymbol{z}\boldsymbol{w}^T)^3\boldsymbol{z}] - 3\boldsymbol{w}$
  - Again set $\boldsymbol{w} \leftarrow \boldsymbol{w}/||\boldsymbol{w}||$ after every update
- Expectation has naturally to be estimated
- No theoretical guarantees but works in practice

FastICA with approximations of negentropy

- Let g be the derivative of a function used to approximate the negentropy
  - $g_1(x) = G_1'(x) = \tanh(ax)$
- The general fixed-point update rule is $\boldsymbol{w} \leftarrow \mathrm{E}[g(\boldsymbol{z}\boldsymbol{w}^T)\boldsymbol{z}] - \mathrm{E}[g'(\boldsymbol{z}\boldsymbol{w}^T)]\boldsymbol{w}$

Multiple components

- So far we have found only one component
  - To find more, remember that vectors $\boldsymbol{w}_i$ are orthogonal (columns of invertible $\boldsymbol{A}$)
- General idea:
  - Find one vector $\boldsymbol{w}$
  - Find second that is orthogonal to the first one
  - Find third that is orthogonal to the two previous ones, etc.

Symmetric orthogonalization

- We can compute $\boldsymbol{w}_i$s in parallel
  - Update $\boldsymbol{w}_i$s independently
  - Run orthogonalization after every update step
    - $\boldsymbol{W} \leftarrow (\boldsymbol{W}\boldsymbol{W}^T)^{-1/2}\boldsymbol{W}$
  - Iterate until convergence

Maximum Likelihood

Skillikorn chapter 7; Hyvärinen & Oja 2000

Maximum-likelihood algorithms

- **Idea**: We are given observations $\boldsymbol{X}$ that are drawn from some parameterized family of distributions $D(\Theta)$
  - The **likelihood** of $\boldsymbol{X}$ given Θ, $L(\Theta; \boldsymbol{X}) = p_D(\boldsymbol{X}; \Theta)$, where $p_D(\cdot; \Theta)$ is the probability density function of D with parameters Θ
- In **maximum-likelihood estimation** (MLE) we try to find Θ that maximizes the likelihood given $\boldsymbol{X}$

ICA as MLE

- If $p_x(\boldsymbol{x})$ is the pdf of $\boldsymbol{x} = \boldsymbol{sA}$, then

$$p_x(\boldsymbol{x}) = p_s(\boldsymbol{s})\,|\det \boldsymbol{B}| = |\det \boldsymbol{B}| \prod_i p_i(s_i) = |\det \boldsymbol{B}| \prod_i p_i(\boldsymbol{x}\boldsymbol{b}_i^T)$$

  - here $\boldsymbol{B} = \boldsymbol{A}^{-1}$

    - In general, if $\boldsymbol{x}$ is r.v. with pdf $p_x(\boldsymbol{x})$ and $\boldsymbol{y} = \boldsymbol{Bx}$, then $p_y(\boldsymbol{y}) = p_x(\boldsymbol{Bx})|\det \boldsymbol{B}|$

- For T observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$ the log-likelihood of $\boldsymbol{B}$ given $\boldsymbol{X}$ is

$$\log L(\boldsymbol{B}; \boldsymbol{X}) = \sum_{t=1}^{T} \sum_{i=1}^{m} \log p_i(\boldsymbol{x}_t \boldsymbol{b}_i^T) + T \log |\det \boldsymbol{B}|$$

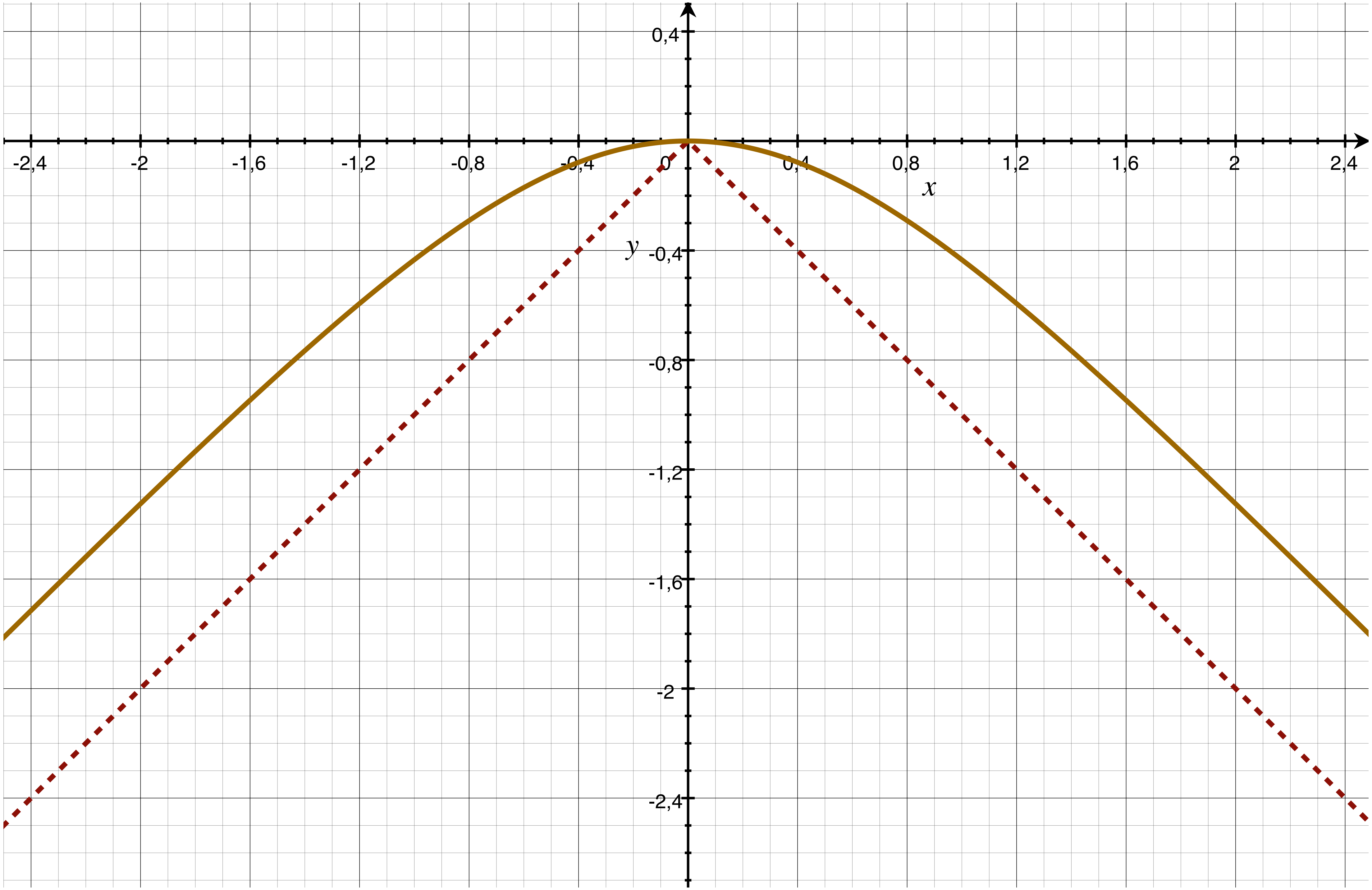
Problems with MLE

- The likelihood is expressed as a function of $\boldsymbol{B}$
- But we also need to estimate the pdfs $p_i()$
  - Non-parametric problem, infinite number of different pdfs
- Very hard problem…

If we know the pdfs

- Sometimes we know the pdfs of the components
  - We only need to estimate their parameters and ***B***
- Sometimes we know only that the pdfs are super-Gaussian (for example)
  - We can use $\log p_i(s_i) = -\log \cosh(s_i)$
    - Requires normalization

$-\log\cosh(x) \approx -|x|$



Nothing on the pdfs is known

- We might not know whether the pdfs of the components are sub- or super-Gaussian
  - It is enough to estimate which one they are!
- For super-Gaussian,
  $\log p_i^+(s_i) = \alpha_1 - 2\log\cosh(s_i)$
- For sub-Gaussian,
  $\log p_i^-(s_i) = \alpha_2 - (s_i^2/2 - \log\cosh(s_i))$

α_i are only needed to make these logs of pdfs – not in optimization

Log-likelihood gradient

- The gradient is $\frac{\partial \log L}{\partial \boldsymbol{B}} = (\boldsymbol{B}^T)^{-1} + \sum_{t=1}^{T} \boldsymbol{g}(\boldsymbol{x}_t \boldsymbol{B}^T)^T \boldsymbol{x}_t$
  - Here $\boldsymbol{g}(\boldsymbol{y}) = (g_i(y_i))_{i=1}^{n}$ with $g_i(y_i) = (\log p_i(y_i))' = p_i'(y_i)/p_i(y_i)$

Step size

- This gives us $\boldsymbol{B} \leftarrow \boldsymbol{B} + \delta((\boldsymbol{B}^T)^{-1} + \sum_t \boldsymbol{g}(\boldsymbol{x}_t\boldsymbol{B}^T)^T\boldsymbol{x}_t)$
- Multiplying from right with $\boldsymbol{B}^T\boldsymbol{B}$ and defining $\boldsymbol{y}_t = \boldsymbol{x}_t\boldsymbol{B}^T$ gives $\boldsymbol{B} \leftarrow \boldsymbol{B} + \delta(\boldsymbol{I} + \sum_t \boldsymbol{g}(\boldsymbol{y}_t)^T\boldsymbol{y}_t)\boldsymbol{B}$
  - So-called **infomax** algorithm

Setting g()

- We compute $E[-\tanh(s_i)s_i + (1 - \tanh(s_i)^2)]$
  - If positive, set $g(y) = -2\tanh(y)$
  - If negative (or zero), set $g(y) = \tanh(y) - y$
- Use current estimates of s_i

Putting it all together

- Start with random $\boldsymbol{B}$ and γ, choose learning rates δ and δ_γ
- Iterate until convergence
  - $\boldsymbol{y} \leftarrow \boldsymbol{Bx}$ and normalize $\boldsymbol{y}$ to unit variance
  - $\gamma_i \leftarrow (1 - \delta_\gamma)\gamma_{i-1} + \delta_\gamma \mathrm{E}[-\tanh(y_i)y_i + (1 - \tanh(y_i)^2)]$
    - if $\gamma_i > 0$, use super-Gaussian g; o/w sub-Gaussian g
  - $\boldsymbol{B} \leftarrow \boldsymbol{B} + \delta(\boldsymbol{I} + \sum_t \boldsymbol{g}(\boldsymbol{y}_t)^T \boldsymbol{y}_t)\boldsymbol{B}$

ICA summary

- ICA can recover independent source signals
  - if they are non-Gaussian
- Does not reduce rank
- Many applications, special case of blind source separation
- Standard algorithmic technique is to maximize non-Gaussianity of the recovered components

ICA literature

- Hyvärinen & Oja (2000): *Independent Component Analysis: Algorithms and Applications*. Neural networks 13(4), 411–430
- Hyvärinen (2013): *Independent component analysis: recent advances*. Phil. Trans. R. Soc. A 371:20110534