# 1 Exercise 1:

## 1.1 Stochastic Neighbor Embedding (SNE)

Stochastic Neighbor Embedding (SNE) starts by converting the high-dimensional Euclidean distances between datapoints into conditional probabilities that represent similarities.

- Consider the neighborhood around an input data point. $x_i \in \mathbb{R}^d$
- Image that we have a Gaussian distribution.
- Then the probability that $x_i$ chooses some. other datapoint $x_j$ as its neighbor is in proportion with the density under this Gaussian - A point closer to $x_i$ will be more likely one further away

$P_{j|i}$ the probability that $x_i$ chooses $x_j$ as neighbor is computed as follows:

$$P_{j|i} = \frac{\exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\left\|x_i - x_k\right\|^2 / 2\sigma_i^2\right)}$$

- $P_{i|i} = 0$
- $\sigma_i$ the chosen size of the neighborhood:
  - a low $\sigma_i$ will lead to a smaller local neighborhood with stronger probabilities
  - a high $\sigma_i$ leads to uniform weights
  - $\sigma_i$ can be set up differently for each point.
- The final distribution over pairs is symmetrized: $P_{ij} = \frac{1}{2N}\left(P_{j|i} + P_{i|j}\right)$
- Given a dataset $X = \{x_1, \cdots, x_N\} \in \mathbb{R}^D$, we define the distribution $P_{ij}$
- Goal: Finding a good embedding $Y = \{y_1, \cdots, y_N\} \in \mathbb{R}^d, \quad d << D$, and ideally $d$ should be 2 or 3 .

**Measuring and minimizing the embedding quality:**

- For all point in $Y$ we can define $Q$ in a similar way than $P$ (notice the absence of $\sigma_i$ and the assymetry):

$$Q_{ij} = \frac{\exp\left(-\left\|y_i - y_j\right\|^2\right)}{\sum_{k \neq i} \exp\left(-\left\|y_i - y_k\right\|^2\right)}$$

- We optimize $Q$ to be close to $P$ by minimizing the KL-divergence on the parameters $Y = \{y_1, \cdots, y_N\}$ !

**The KL divergence:**
Measuring the distance between two discrete distributions $P$ and $Q$ :

$$KL(Q\|P) = \sum_i Q(i) \log\left(\frac{Q(i)}{P(i)}\right)$$

KL properties

- $KL(Q\|P) \geq 0$ and is a convex function
- $KL(Q\|P) = 0$ if and only if $Q = P$
- if $P(i) = 0$ but $Q(i) > 0$, then $KL(Q\|P) = \infty$

- We have $P$ and we are looking for $Y = \{y_1, \cdots, y_N\} \in \mathbb{R}^d$ such that the distribution $Q$ we infer will minimize $L(Q) = KL(P\|Q)$ (notice the $Q$ on the right, which is uncommon).

- Note that $KL(P\|Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right) = -\sum_{ij} P_{ij}\left(Q_{ij}\right) + \text{const}$

- We can show that $\frac{\partial L}{\partial y_i} = \sum_j \left(P_{ij} - Q_{ij}\right)\left(y_i - y_j\right)$

## 1.2 t- SNE

- Student-t Probability density: $p(x) \approx \left(1 + \frac{x^2}{v}\right)^{-(v+1)/2}$
- For $v = 1$, we get: $p(x) \approx \frac{1}{1+x^2}$
- The probability of this distribution goes much more slower to zero than a Gaussian.
- This is convenient for our crowding problem. We can show that this is equivalent to averaging a Gaussian with some prior over $\sigma^2$. Then, we can redefine $Q_{ij}$ so that:

$$Q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_k \sum_{l \neq k} \left(1 + \|y_i - y_k\|^2\right)^{-1}}$$

**t-SNE Gradients:**

- - We can show that the gradients of t-SNE objective function are:

$$\frac{\partial L}{\partial y_i} = \sum_j \frac{(P_{ij} - Q_{ij})(y_i - y_j)}{1 + \|y_i - y_j\|^2}$$

- Compared with the SNE gradients: $\frac{\partial L}{\partial y_i} = \sum_j (P_{ij} - Q_{ij})(y_i - y_j)$
- Both repulse close dissimilar points and attract far similar points, but the t-SNE has a smaller attraction term that reduces crowding issues.

**t-SNE algorithm**

**Algorithm 1**: Simple version of t-Distributed Stochastic Neighbor Embedding.

**Data**: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity *Perp*,
optimization parameters: number of iterations $T$, learning rate $\eta$, momentum $\alpha(t)$.
**Result**: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
**begin**
  compute pairwise affinities $p_{j|i}$ with perplexity *Perp* (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  **for** *t=1* **to** $T$ **do**
    compute low-dimensional affinities $q_{ij}$ (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  **end**
**end**

We have seen that $\sigma_i$ is a key neighborhood parameter. For each distribution $P_{j|i}$ (depending on $\sigma_i$ ), we define the perplexity:

$$\text{Perp}\left(P_{j|i}\right) = 2^{H\left(P_{j|i}\right)} \quad \text{where} \quad H(P) = -\sum_i P_i \log\left(P_i\right)$$

**t-SNE Perplexity**

- If $P$ is uniform over $k$ elements - perplexity is $k$.
  - Smooth version of k in kNN
  - Low perplexity means small $\sigma^2$
  - High perplexity means large $\sigma^2$
- We can define the desired perplexity and set $\sigma_i$ to get that (bisection method): values between 5 and 50 usually work well.
- It is an important parameter as it allows to capture different scales in the data.

## 2 Exercise 2: (python file)

## 3 Exercise 3: (python file)

## 4 Exercise 4:

- t-SNE is an alternative dimensionality reduction algorithm tries to find a global structure
  - Low dimensional subspace
  - Can lead to local inconsistencies $->$ Far away point can become nearest neighbors
- t-SNE tries to perserve local structure
  - Low dimensional neighborhood should be the same as original neighborhood

## 5 Exercise 5: (python file)