Table of Contents

CSC 2506 Probabilistic Learning and Reasoning

Assignment I - Q2

Tianyi Liu liuti110 1005820827

High-Dimensional Gaussians [20 pts]

In this question we will investigate how our intuition for samples from a Gaussian may break down in higher dimensions. Consider samples from a D-dimensional unit Gaussian

$x \sim \mathcal{N}(0_D, I_D)$ where~0_D indicates a column vector of~D zeros and~I_D is a $D \times D$ identity matrix.

Distance of Gaussian samples from origin

Starting with the definition of Euclidean norm, quickly show that the distance of x from the origin is $\sqrt{x^\intercal x}$

Answer: $d_{euclidean}(x) = \|x\|_2 = \sqrt{\|x\|_2^2} = \sqrt{\sum_{i=1}^{D} x_i^2} = \sqrt{x^T x}. \quad \square$

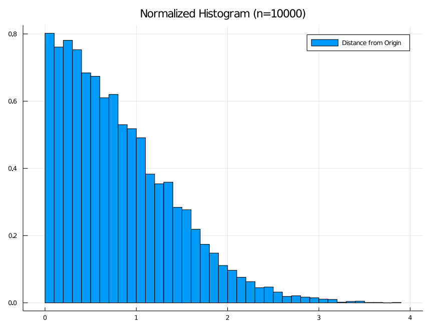
Distribution of distances of Gaussian samples from origin

In low-dimensions our intuition tells us that samples from the unit Gaussian will be near the origin.

1. Draw 10000 samples from a $D = 1$ Gaussian
2. Compute the distance of those samples from the origin.
3. Plot a normalized histogram for the distance of those samples from the origin.

Does this confirm your intuition that the samples will be near the origin?

```
using Plots
```



```
begin
    n= 10000;
    # data = randn(n);
    data = []
    for i = 1:n
        append!(data, randn())
    end
    data = convert.(Float64, data)
    distance = .√ (data .^ 2);
    histogram(distance, normalize=true, label="Distance from Origin", size=
(800,600));
    title!("Normalized Histogram (n=$n)");
end
```

Answer: Yes. When $D = 1$, the samples are near the origin.

Plot samples from distribution of distances

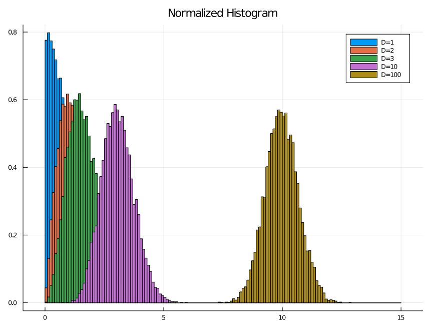
1. Draw a set of 10000 samples from $D = \{1, 2, 3, 10, 100\}$ Gaussians
2. Compute the distance of each sample from the origin
3. With all D dimensionality on a single plot, show the normalized histograms for the distribution of distance of those samples from the origin.

As the dimensionality of the Gaussian increases, what can you say about the expected distance of the samples from the Gaussian's mean (in this case, origin)?

```
using Distributions
```

```
generate_mvgaussian (generic function with 3 methods)
```

```
function generate_mvgaussian(D=1, n=10000)
    # return rand(MvNormal(D, 1), n)
    data = []
    for i = 1: (D*n)
        append!(data,randn())
    end
    return convert.(Float64, reshape(data, (D, n)))
end
```



```
begin
    ks = [1, 2, 3, 10, 100]
    datas = generate_mvgaussian.(ks, [10000]);
    hist_resolution = collect(0:0.1:15);
    for (data, dim) in zip(datas, ks)
        distance = dropdims(.√sum((data .^ 2), dims=1), dims=1);
        if dim == ks[1]
            histogram(distance, normalize=true, bins=hist_resolution,
label="D=$dim", size=(800,600));
        else
            histogram!(distance, normalize=true, bins=hist_resolution,
label="D=$dim");
        end
    end
    title!("Normalized Histogram");
    current();
end
```

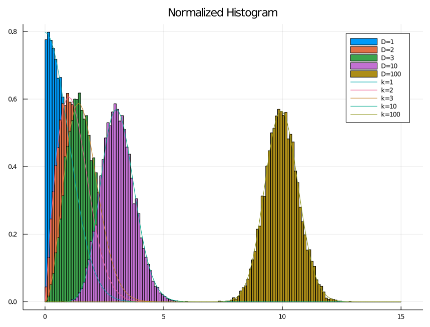
Answer: As the dimensionality of Gaussian increases, the expected distance from data point to the mean vector increases.

Plot the χ-distribution

From Wikipedia, if x_i are k independent, normally distributed random variables with means μ_i and standard deviations σ_i then the statistic $Y = \sqrt{\sum_{i=1}^{k}(\frac{x_i-\mu_i}{\sigma_i})^2}$ is distributed according to the χ-distribution]

On the previous normalized histogram, plot the probability density function (pdf) of the χ-

distribution for $k = \{1, 2, 3, 10, 100\}$.



```
begin
    using SpecialFunctions
    function chipdf(k, x)
        1 / (2 ^ ((k / 2) - 1) * gamma(k / 2)) * x ^ (k - 1) * e ^ (- x ^ 2 / 2)
    end
    for k in ks
        plot!(hist_resolution, chipdf.([k], hist_resolution),  label="k=$k");
    end
    current();
end
```

Distribution of distance between samples

Taking two samples from the D-dimensional unit Gaussian, $x_a, x_b \sim \mathcal{N}(0_D, I_D)$ how is $x_a - x_b$ distributed? Using the above result about χ-distribution, derive how $||x_a - x_b||_2$ is distributed.

(Hint: start with a $\mathcal{X}$-distributed random variable and use the change of variables formula.)

$$x_a - x_b \sim \mathcal{N}(0_D, 2I_D).$$

As given, $Y = \sqrt{\sum_{i=1}^{k}(\frac{x_i - \mu_i}{\sigma_i})^2} \sim \chi(k)$. We write $\tilde{x} \triangleq x_a - x_b$, which yields $\tilde{x} \sim \mathcal{N}(0_D, 2I_D)$.

Hence,

$$\begin{aligned} Y &= \sqrt{\sum_{i=1}^{D}\left(\frac{\tilde{x}_i}{\sqrt{2}}\right)^2} \\ &= \sqrt{\frac{1}{2}\sum_{i=1}^{D}\tilde{x}_i{}^2} \\ &= \frac{\sqrt{2}}{2}||\tilde{x}||_2 \sim \chi(D). \end{aligned}$$

Use change of variable formula, we have $g : \mathbb{R} \to \mathbb{R}$, i.e., $g(X) = \sqrt{2}X$, a scalar to scalar transformation. Hence, the resultant PDF is:

$$\begin{aligned} f_Y(y) &= f_X(g^{-1}(y))\left|\frac{d}{dy}g^{-1}(y)\right| \\ &= f_X\left(\frac{\sqrt{2}}{2}x\right)\frac{\sqrt{2}}{2}. \end{aligned}$$

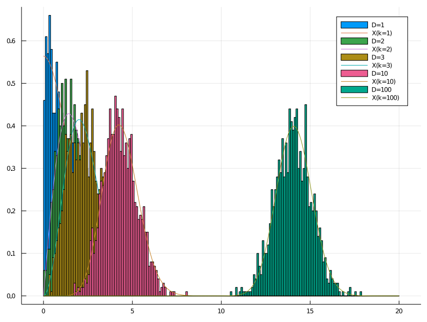
Hence, $\|x_a - x_b\|_2 \sim \frac{\sqrt{2}}{2}\chi(\frac{\sqrt{2}}{2}D)$.

Plot pdfs of distribution distances between samples

For for $D = \{1, 2, 3, 10, 100\}$. How does the distance between samples from a Gaussian behave as dimensionality increases? Confirm this by drawing two sets of 1000 samples from the D-dimensional unit Gaussian. On the plot of the χ-distribution pdfs, plot the normalized histogram of the distance between samples from the first and second set.

```
generate_mvgaussian_2 (generic function with 3 methods)
```

```
function generate_mvgaussian_2(D=1, n=10000)
    # return rand.([MvNormal(D, 1),MvNormal(D, 1)], [n,n])
    return [generate_mvgaussian(D, n), generate_mvgaussian(D, n)]
end
```



```
begin
    datas_2 = generate_mvgaussian_2.(ks, [1000]);
    hist_resolution_2 = collect(0:0.1:20);
    for (data, dim) in zip(datas_2, ks)
        distance = dropdims(.√sum(((data[1] - data[2]) .^2), dims=1), dims=1)
        if dim == ks[1]
            histogram(distance, normalize=true, bins=hist_resolution_2,
label="D=$dim", size=(800,600));
            plot!(hist_resolution_2, 1 / √2 * chipdf.([dim], 1 / √2 *
hist_resolution_2), label="X(k=$dim)");
        else
            histogram!(distance, normalize=true, bins=hist_resolution_2,
label="D=$dim");
            plot!(hist_resolution_2, 1 / √2 * chipdf.([dim], 1 / √2 *
hist_resolution_2), label="X(k=$dim)");
        end
    end
    current();
end
```

Answer: The distance between samples from a Gaussian increases as dimensionality increases. For a given data dimensionality, the distance is larger than the results we obtained from the previous question (about $\sqrt{2}$ larger).

Linear interpolation between samples

Given two samples from a gaussian $x_a, x_b \sim \mathcal{N}(0_D, I_D)$ the linear interpolation between them x_α is defined as a function of $\alpha \in [0, 1]$

$$\text{lin\_interp}(\alpha, x_a, x_b) = \alpha x_a + (1 - \alpha) x_b$$

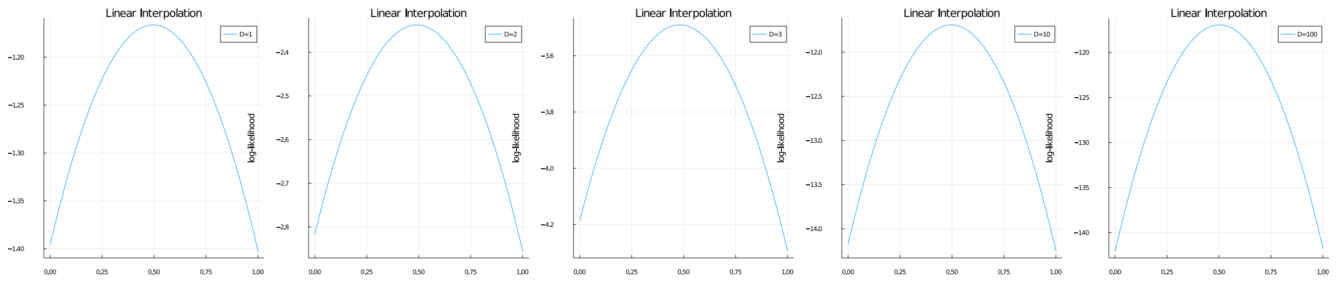
For two sets of 1000 samples from the unit gaussian in D-dimensions, plot the average log-likelihood along the linear interpolations between the pairs of samples as a function of α.

(i.e. for each pair of samples compute the log-likelihood along a linear space of interpolated points between them, $\mathcal{N}(x_\alpha | 0, I)$ for $\alpha \in [0, 1]$. Plot the average log-likelihood over all the interpolations.)

Do this for $D = \{1, 2, 3, 10, 100\}$, one plot per dimensionality. Comment on the log-likelihood under the unit Gaussian of points along the linear interpolation. Is a higher log-likelihood for the interpolated points necessarily better? Given this, is it a good idea to linearly interpolate between samples from a high dimensional Gaussian?

```
loglikelihood_mvgaussian (generic function with 1 method)
```

```julia
begin
    using LinearAlgebra
    function loglikelihood_mvgaussian(μ, Σ, x)
        # For faster matrix inversion and det
        if Σ == Matrix(I, size(μ)[1], size(μ)[1])
            Σ_inv = Σ
            Σ_det = 1
        else
            Σ_inv = inv(Σ)
            Σ_det = det(Σ)
        end
        k = size(μ)[1]
        return log((2 * π) ^ (- k / 2) * Σ_det ^ (- 0.5) * ℯ ^ (- 0.5 * (x - μ)' * 
Σ_inv * (x - μ)))
    end
end
```



```
begin
    ps_1 = [];
    n_sample = 1000;
    datas_3 = generate_mvgaussian_2.(ks, [n_sample]);
    α = collect(0:0.01:1);
    logllhd_linear = []
    for (data, dim) in zip(datas_3, ks)
        logllhd = []
        for _α in α
            interpolated_data = _α .* data[1] + (1 .- _α) .* data[2];
            lllhd = 0;
            for i in 1:n_sample
                # built-in log-likelihood
                # lllhd += loglikelihood(MvNormal(dim, 1), interpolated_data[:,i]);
                # manual implementation
                lllhd += loglikelihood_mvgaussian(zeros(dim), Matrix(I, dim, dim),
interpolated_data[:,i])
            end
            lllhd /= n_sample;
            append!(logllhd, lllhd);
        end
        if dim == ks[1]
            p = plot(α, logllhd, label="D=$dim");
        else
            p = plot(α, logllhd, label="D=$dim");
        end
        push!(logllhd_linear, logllhd);
        xlabel!("α");
        ylabel!("log-likelihood");
        title!("Linear Interpolation");
        push!(ps_1, p);
    end
    current();
    plot(ps_1[1], ps_1[2], ps_1[3], ps_1[4], ps_1[5], layout=(1,5), size=(2500,
500));
end
```

Answer: The log-likelihood of the interpolated point increases with α until 0.5, where the log-likelihood starts to decrease. The percentage-wise difference between the max value and the min value keeps increasing with D.

- A higher likelihood does not always correspond to a better interpolation results. Comparing to the starting point and end point, the likelihood of points along linear interpolation deviate a significant amount. This, in the contrary, reflects that linear interpolation won't produce *convicing* data points.
- Recall that samples from higher dimensional Gaussian are most likely to have the same ℓ_2 norm and also being orthogonal. Linear interpolation also does not preserve such relationship.
- From another prospective, the log-likelihood is always larger than end points. Since the linearly interpolated points are still comes from Gaussian, linear interpolation results in a narrower and higher PDF.

In summary, linear interpolation is not a wise choice in high dimensional Gaussian.

Polar Interpolation Between Samples

Instead we can interpolate in polar coordinates: For $\alpha \in [0, 1]$ the polar interpolation is

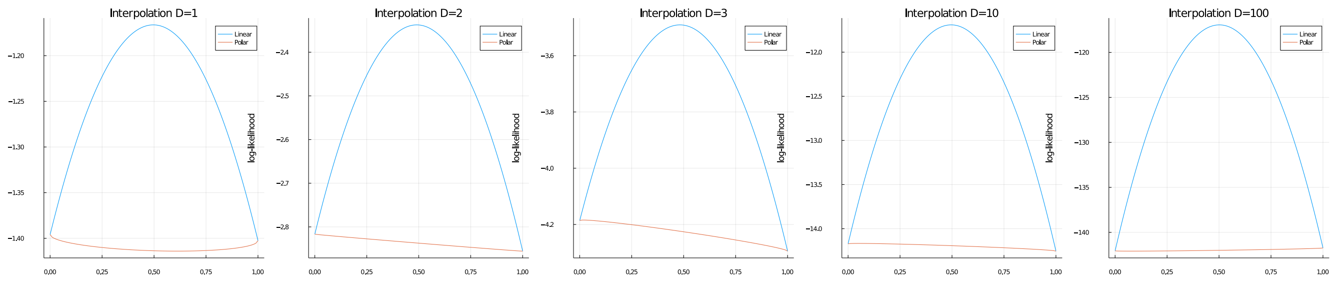
$$\text{polar\_interp}(\alpha, x_a, x_b) = \sqrt{\alpha} x_a + \sqrt{(1-\alpha)} x_b$$

This interpolates between two points while maintaining Euclidean norm.

On the same plot from the previous question, plot the probability density of the polar interpolation between pairs of samples from two sets of 1000 samples from D-dimensional unit Gaussians for $D = \{1, 2, 3, 10, 100\}$.

Comment on the log-likelihood under the unit Gaussian of points along the polar interpolation. Give an intuitive explanation for why polar interpolation is more suitable than linear interpolation for high dimensional Gaussians. (For 6. and 7. you should have one plot for each D with two curves on each).

```
begin
    logllhd_polar = []
    for (data, dim) in zip(datas_3, ks)
        logllhd = []
        for _α in α
            interpolated_data = √_α .* data[1] + √(1 .- _α) .* data[2];
            _size = size(interpolated_data);
            lllhd = 0;
            for i in 1:n_sample
                # built-in log-likelihood
                # lllhd += loglikelihood(MvNormal(dim, 1), interpolated_data[:,i]);
                # manual implementation
                lllhd += loglikelihood_mvgaussian(zeros(dim), Matrix(I, dim, dim),
interpolated_data[:,i])
            end
            lllhd /= n_sample;
            append!(logllhd, lllhd);
        end
        push!(logllhd_polar, logllhd);
    end
end
```

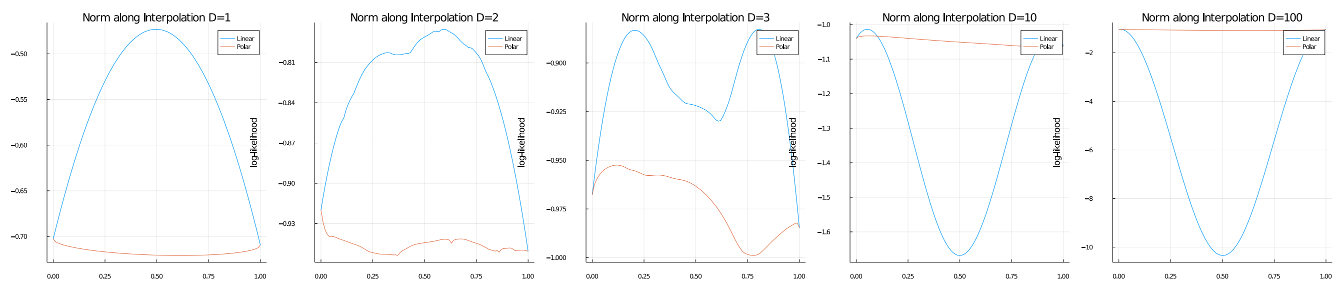


```
begin
    ps_2 = []
    for (lnr, plr, dim) in zip(logllhd_linear, logllhd_polar, ks)
        p = plot(α, lnr, label="Linear");
        plot!(p, α, plr, label="Polar");
        title!("Interpolation D=$dim");
        xlabel!("α");
        ylabel!("log-likelihood");
        push!(ps_2, p);
    end
    current();
    plot(ps_2[1], ps_2[2], ps_2[3], ps_2[4], ps_2[5], layout=(1,5), size=(2500,
500));
end
```

Answer: The plots of linear interpolation are the same as above. We notice that polar interpolation results in flatter log-likelihood curve along α. This implies that data points along polar interpolation are similar, and also more convicing in the context of probability, which is represented by the likelihood. Whereas, data points along linear interpolation deviate more than them along polar interpolation. Note that polar interpolation preserves the euclidean norm, which could explain the reason why polar interpolation produces better interpolated data points than linear interpolation. In summary, polar interpolation is more suitable than linear interpolation.

Norm along interpolation

In the previous two questions we compute the average log-likelihood of the linear and polar interpolations under the unit gaussian. Instead, consider the norm along the interpolation, $\sqrt{x_\alpha^\intercal x_\alpha}$. As we saw previously, this is distributed according to the $\mathcal{X}$-distribution. Compute and plot the average log-likelihood of the norm along the two interpolations under the the $\mathcal{X}$-distribution for $D = \{1, 2, 3, 10, 100\}$, i.e. $\mathcal{X}_D(\sqrt{x_\alpha^\intercal x_\alpha})$. There should be one plot for each D, each with two curves corresponding to log-likelihood of linear and polar interpolations. How does the log-likelihood along the linear interpolation compare to the log-likelihood of the true samples (endpoints)?



```
begin
    function loglikelihood_chi_stable(ks, x)
        return log(1 / (2 ^ ((ks / 2) - 1) * gamma(ks / 2)) * x ^ (ks - 1)) + (- x
^ 2 / 2)
    end
    ps_3 = [];
    for (data, dim) in zip(datas_3, ks)
        linear_logllhd = [];
        polar_logllhd = [];
        for _α in α
            linear_interpolate = _α .* data[1] + (1 .- _α) .* data[2];
            polar_interpolate = √_α .* data[1] + √(1 .- _α) .* data[2];
            linear_norm = .√sum(linear_interpolate[:,1:n_sample]' .*
linear_interpolate'[1:n_sample,:], dims=2)
            polar_norm = .√sum(polar_interpolate[:,1:n_sample]' .*
polar_interpolate'[1:n_sample,:], dims=2)
            linear_lllhd = sum(loglikelihood_chi_stable.(dim, linear_norm))
            polar_lllhd = sum(loglikelihood_chi_stable.(dim, polar_norm))
            linear_lllhd /= n_sample;
            polar_lllhd /= n_sample;
            append!(linear_logllhd, linear_lllhd);
            append!(polar_logllhd, polar_lllhd);
        end
        p = plot(α, linear_logllhd, label="Linear");
        plot!(α, polar_logllhd, label="Polar")
        xlabel!("α");
        ylabel!("log-likelihood");
        title!("Norm along Interpolation D=$dim");
        push!(ps_3, p);
    end
    current();
    plot(ps_3[1], ps_3[2], ps_3[3], ps_3[4], ps_3[5], layout=(1,5), size=(2500,
500));
end
```

Answer: For polar interpolation, the results we claimed in the previous question remain unchanged. Additionally, we observe a tradeoff in dimension D for linear interpolation. When D is smaller, the log-likelihood of norm is costantly larger than end points. When D is larger, the log-likelihood of norm is, conversely, smaller than end points.

This indicates that linear interpolations results in less probably samples, reflected as deviation in log-likelihood of either side, and polar interpolations produce comparably good points.