Statistics for Astronomers: Lecture 18, 2019.05.06

Prof. Sundar Srinivasan

IRyA/UNAM

Recall: Multivariate posteriors

(from Andrew Gelman et al., "Bayesian Data Analysis", 3ed.)

In most of the problems you will deal with in research,
$\vec{\boldsymbol{\theta}} = (\theta_1, \theta_2, \cdots, \theta_{N_{\rm par}})$ with $N_{\rm par} > 1$.

Definition (Joint, conditional, and marginal posteriors)

$p(\vec{\boldsymbol{\theta}}|\mathrm{data})$ – joint posterior distribution for all the parameters.

$p(\theta_1|\theta_2, \cdots, \theta_{N_{\rm par}}, \mathrm{data})$ – conditional posterior for θ_1 at fixed values of all other components of $\vec{\boldsymbol{\theta}}$ and data.

$p(\theta_1|\mathrm{data})$ – marginal posterior for θ_1, marginalised over all other parameters.

Recall: Illustration for a normal posterior

If the data is drawn from a normal distribution, and we pick non-informative priors for μ and $\ln\sigma$, then

1. For known σ^2, μ is distributed normally about the sample mean, with variance σ^2/N.
2. For known μ, σ^2 has an Inverse-χ^2 distribution with degree equal to the sample size.
3. For unknown σ^2, μ has a Student's t distribution around the sample mean.
4. For unknown μ, σ^2 has an Inverse-χ^2 distribution with degree equal to the sample size minus 1.

For the last two cases, the unknown parameter is a nuisance parameter that has been marginalised over.

Recall: Posterior predictive distribution

Given a set of observations (data) and the resulting posterior for the model ("data is drawn from a normal distribution"), predict the pdf of future data values.
For the problem discussed in this lecture,

$$p(\text{future data}|\text{data}) = \int\int d\mu\, d\sigma^2 \underbrace{p(\mu,\sigma^2|\text{data})}_{\text{joint posterior}} \overbrace{p(\text{future data}|\mu,\sigma^2,\text{data})}^{\sim\mathcal{N}(\mu,\sigma^2)}$$

To simulate this distribution, first draw μ, σ^2 from their joint pdf then draw new data values from $\mathcal{N}(\mu,\sigma^2)$.

We expect that the new data point be distributed around $\bar{x}$, the mean of the current dataset.

The expected variance is $\sigma^2 + \sigma^2/N = (1+1/N)\sigma^2$.

In fact, the posterior predictive pdf for the new data point is a Student's t distribution with location $\bar{x}$, scale $\sigma\sqrt{1+1/N}$, and degree $N-1$.

Visualising data

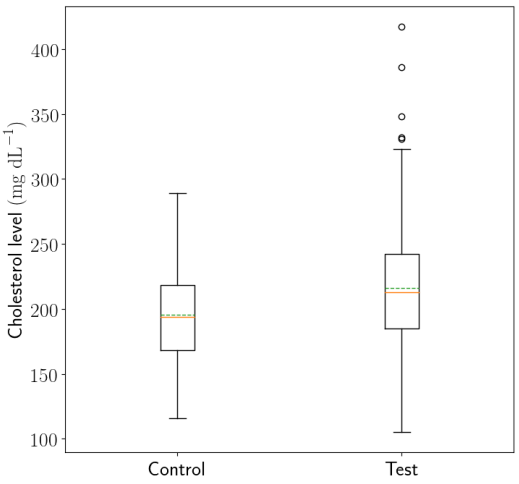
Five-number summary

Five number summary of a dataset of size N: $x_{(1)}, q_{25}, q_{50}, q_{75}, x_{(N)}$.
q_{50} = median, a robust location measure.
Interquartile range, $\mathrm{IQR} = q_{75} - q_{50}$ is a robust scale measure (for a normal distribution, $\mathrm{IQR} \approx 1.349\sigma$). The IQR encloses 50% of the sample distribution.
Compare $\bar{x}$ to q_{50} to check for skewness.
One way of visualising data, the box plot, uses the five-number summary.

Box plot or Box-and-whisker plot

A non-parametric way to visualise the data distribution, without binning.
Procedure illustrated with the blood cholesterol data from Homework #3 (https://bit.ly/2Wvy49i):

1. Identify median with a horizonal line. In addition, can show the mean with a dotted line. Compare the two → skewness.
2. Draw a box enclosing the central 50% of the data (the box edges are q_{25} and q_{75}).
3. From each box edge, extend a "whisker" of length $\frac{3}{2}\mathrm{IQR}$. These whiskers display the tails of the distribution.
4. Any data outside the box-and-whisker region are outliers and can be displayed with individual symbols.
5. Mild $\left(\frac{3}{2} \leq \frac{|x - q_{50}|}{\mathrm{IQR}} < 3\right)$ and extreme $\left(\frac{|x - q_{50}|}{\mathrm{IQR}} \geq 3\right)$ outliers can be also distinguished.



Comparing relative locations and sizes of boxes → comparing distributions.

Activity: use the blood fat data from HW#3 and `pyplot.boxplot` to replicate above plot.

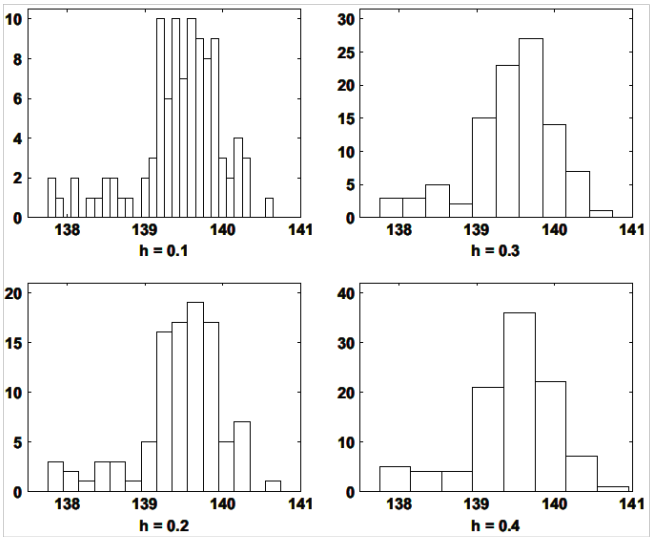
Histogram

Also non-parametric, generates a piecewise constant estimator of the underlying density distribution. Data of size N is placed into M bins of width h such that

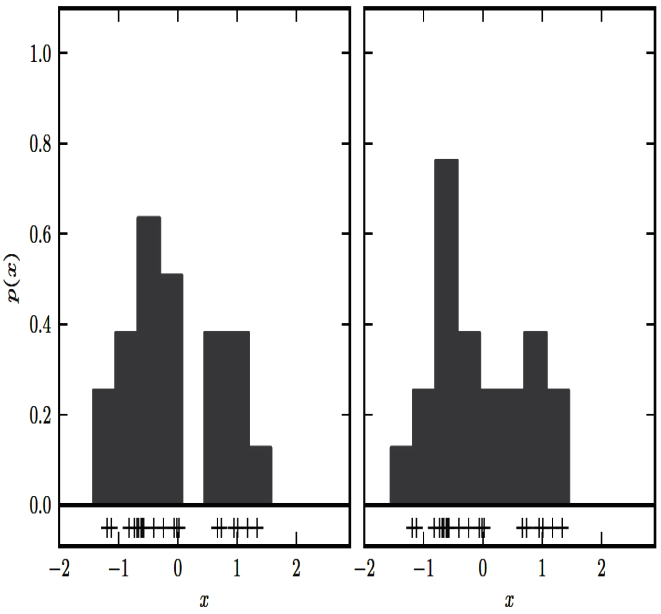
$$\hat{f}(x) = \frac{1}{hN} \sum_{i=1}^{N} \sum_{b=1}^{M} \mathbb{I}\left(\frac{|x_i - x_b|}{h} \leq 1\right) \mathbb{I}\left(\frac{|x - x_b|}{h} \leq 1\right)$$

where x_i are the data points, x_b is the central location of the b^{th} bin, and $\mathbb{I}$ is the indicator function.
Advantages: easy and quick to compute, does well for large N.
Disadvantages:
Location information for data degraded (location for all points in a bin is now center of bin).
Shape highly sensitive on bin edge and bin width.

Histogram (contd.)



Effect of binwidth.
Source: Applied Multivariate Statistical Analysis, Härdle & Simar



Effect of bin location.
Source: AstroML book

Optimal bin width for a histogram

Frequentist methods

Find the binwidth that optimises some function of deviation of the estimated density from the true density. This requires assumptions about the true density.

e.g. Assuming that the data are normally distributed, Scott's rule (Scott 1979) is $h \approx \mathrm{IQR}\ N^{-1/3}$, with s the sample standard deviation.

e.g. Allowing for some departure from normality (*viz.*), the Freedman-Diaconis rule (Freedman & Diaconis 1981) is $h = 2\ \mathrm{IQR}\ N^{-1/3}$.

Disadvantage of these methods: not sensitive to multimodal distributions.

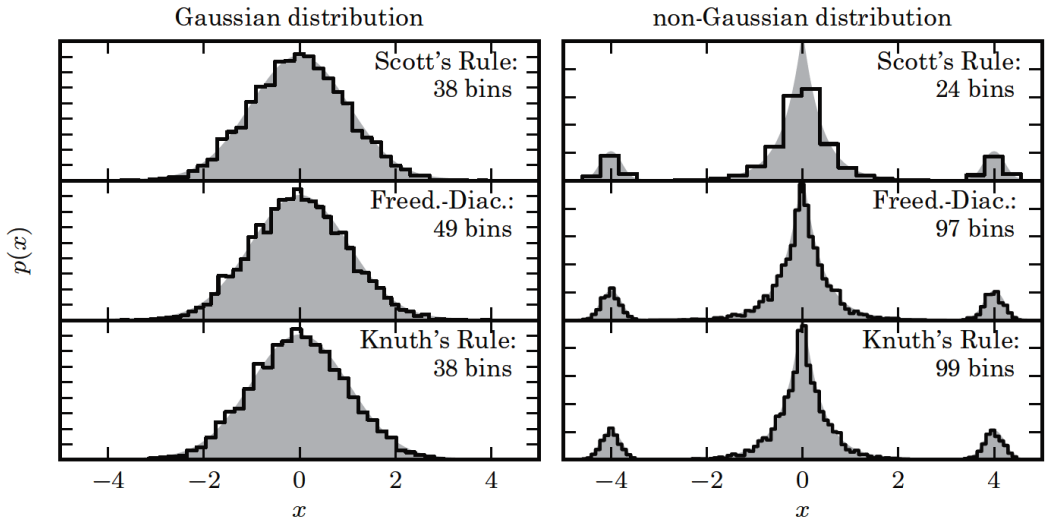
Bayesian methods:

No assumptions required about underlying distribution, can form a data likelihood and assume appropriate priors for the problem.

Knuth (2006) used a multinomial likelihood and Jeffreys priors to find the optimal h. The Bayesian method also allows the computation of the means and standard deviations of the bin heights. Good multimodal/unimodal distinction!

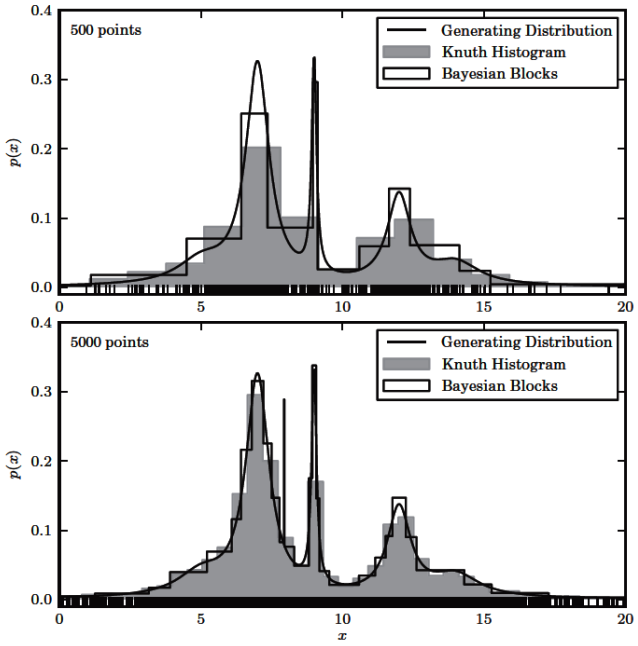
The method of Bayesian Blocks (*e.g.*, Scargle et al. 2013, applied to time-series data) designs a log-likelihood allowing for varying binsize. The explanation by Jake VanderPlas is worth a read: https://jakevdp.github.io/blog/2012/09/12/dynamic-programming-in-python/.

Comparison of optimal widths



Source: AstroML book

Bayesian methods: constant vs. variable bin width



Source: AstroML book

Bayesian blocks method better for small samples.

Kernel density estimate

Non-parametric density estimate. Recall the histogram estimator equation:

$$\hat{f}(x) = \frac{1}{hN} \sum_{i=1}^{N} \overbrace{\sum_{b=1}^{M} \mathbb{I}\left(\frac{|x_i - x_b|}{h} \leq 1\right)}^{K(u_i)} \mathbb{I}\left(\frac{|x - x_b|}{h} \leq 1\right)$$

Generalisation: replace the inner sum with a function $K(u_i)$ of $u_i = \left(\frac{x - x_i}{h}\right)$. The function $K(u)$ is called a kernel, and h is its bandwidth. $K(u)$ is evaluated at each data point x_i. Instead of each data point being treated as a delta function at its location, each data point now has a "bin" represented by the normalised function $K(u)$, and the bins are allowed to overlap with those of other data points. The estimated density is then a sum of these overlapping functions.

KDE (contd.)

There are many functions used for $K(u)$.
Some standard ones: Gaussian, top hat, Epanechnikov (quadratic in u), exponential, linear, and cosine. The Gaussian kernel is one of the most popular choices. The Epanechnikov kernel minimises the mean square error, so it is also popular.

For more, see https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/.

The influence of $K(u)$ is controlled by its bandwidth h, which must be estimated. Modern codes for computing the KDE have built-in options for this.
KDE is implemented in `Python` packages such as `Scikit-learn`, `Scipy`, and `Statsmodels`.

KDE can also be modified to handle measurement errors (see chapter 6 in AstroML book)!

	Bandwidth Selection	Available Kernels	Multi-dimension	Heterogeneous data	FFT-based computation	Tree-based computation
Scipy	Scott & Silverman	One (Gauss)	Yes	No	No	No
Statsmodels KDEUnivariate	Scott & Silverman	Seven	1D only	No	Yes	No
Statsmodels KDEMultivariate	normal reference cross-validation	Seven	Yes	Yes	No	No
Scikit-Learn	None built-in; Cross val. available	6 kernels x 12 metrics	Yes	No	No	Ball Tree or KD Tree

Summary table from Jake VanderPlas' blog.

Summary

1. If you're only interested in the general trend in your data, use box plots. They'll also immediately identify outliers!
2. Histograms are fast but bad for various reasons – their shapes depend on bin size and bin location, and they degrade the information contained in the raw data.
3. There are ways to figure out the optimum bin size – both frequentist and Bayesian. The Bayesian versions are more sensitive to multimodal distributions, and allow for the computation of the optimum bin size without as few assumptions on the underlying distribution as possible.
4. The Bayesian Blocks method allows for variable bin size! It is especially applicable for small data sizes.
5. If you're really interested in generating a function that mimics the true population distribution, use KDEs.