Machine Learning (Semester 1 2025)

# Resampling & Sampling Methods

**Nina Hernitschek**
Centro de Astronomía CITEVA
Universidad de Antofagasta

May 27, 2025

# Motivation



In the previous sessions, we saw that various approaches involve **repeatedly drawing samples** from a set.
E.g.: cross-validation.

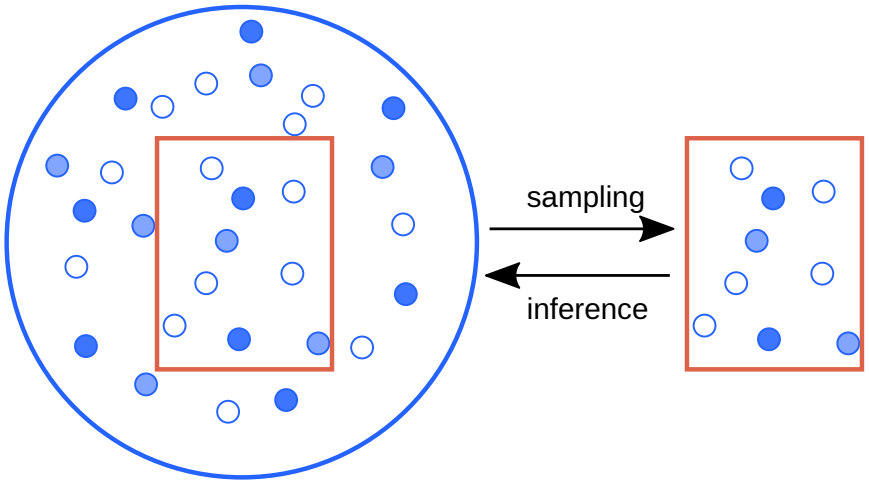
How should these samples been drawn?

# What is Resampling



Resampling is a statistical technique in which multiple new samples are drawn from a sample or from the population.

Statistics of interest (e.g. sample median) are calculated for each new sample. The distribution of new statistics can be analysed to investigate different properties (e.g., confidence intervals, the error, the bias) of the statistics.



# The Purpose of Resampling



Resampling methods are an important tool in modern statistics.

They involve repeatedly drawing samples from a training set and refitting a model on each to obtain additional information about the fitted model.

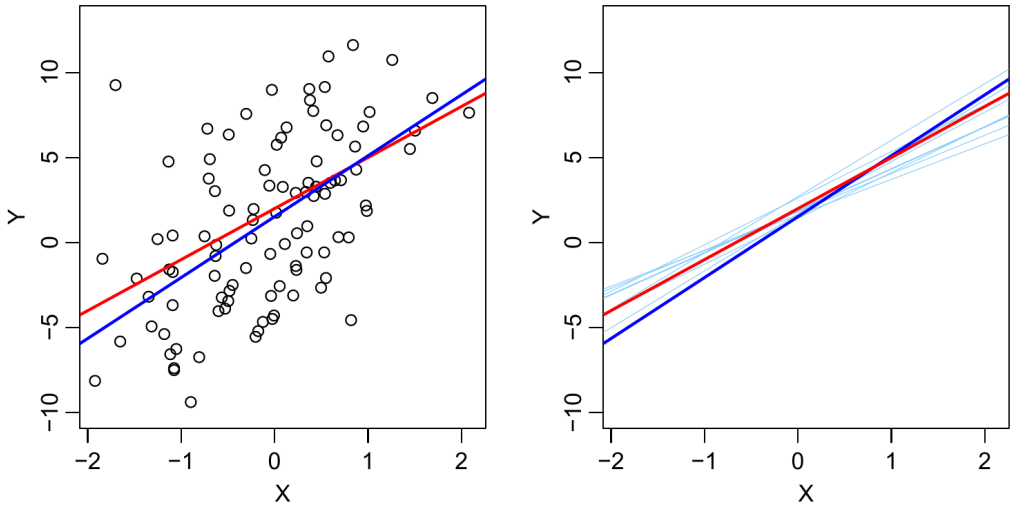
# The Purpose of Resampling



Resampling methods are an important tool in modern statistics.

They involve repeatedly drawing samples from a training set and refitting a model on each to obtain additional information about the fitted model.

**example:** To estimate the variability of a linear regression fit, we can repeatedly draw different samples from the training data, fit a linear regression to each new sample, and then examine the extent to which the resulting fits differ. This may allow us to obtain information not available from fitting the model only once using the original training sample.




Left: The red line gives the population regression line. The blue line is the estimate from one realization.
Right: Ten least squares lines are shown (light blue), each based on a separate realization drawn from the distribution.
Source: Fig. 3.3 from `https://www.statlearning.com/`

# The Purpose of Resampling



Though collecting more data would be the most obvious way to check if a model is robust, this is not always possible.

For this reason, **model validation approaches based on resampling** are developed to verify that models are robust with limited amounts of data. There are three primary model validation methods: training-validation splits (James et al., 2013), cross validation (CV) (James et al., 2013; Kohavi, 1995; Takei et al., 2021), and the bootstrap (James et al., 2013; Kohavi, 1995; Takei et al., 2021).

While drawing (or generating) samples, certain **statistics (or estimates)** of interest (e.g., the sample median) are then calculated for each of these new samples. These resulting **multiple** calculated values of the statistics are then analysed in order to investigate and estimate various properties (e.g., the sampling distribution, the error, the bias) of the statistics.

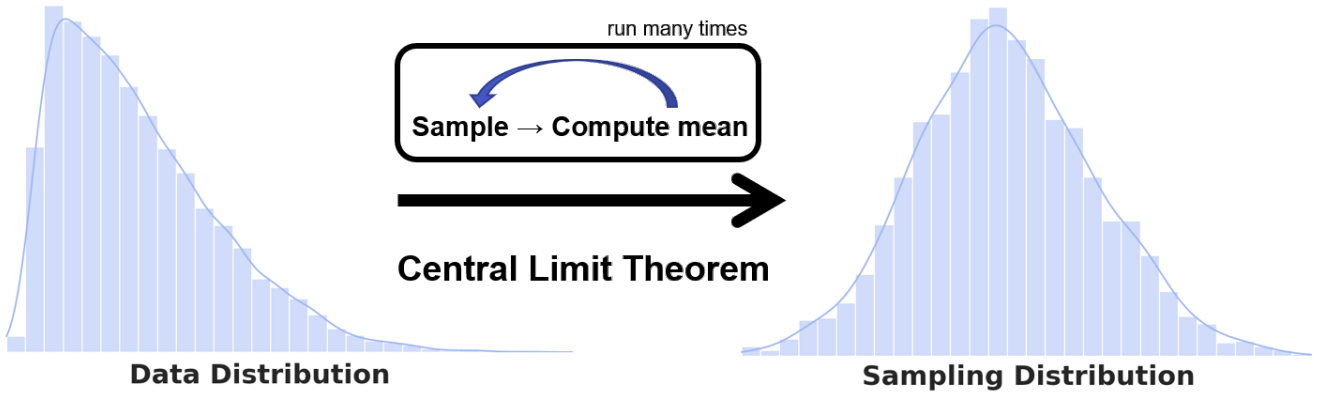
# The Purpose of Resampling



First, note that an estimator of the parameter $\theta$, which we will denote $\hat{\theta}$, is a random variable. We want to estimate $\theta$ from the data. This estimator can be e.g. mean, median, standard deviation...

Its value will (potentially) change with each new sample drawn from the population.

The possible values of and their associated probabilities are described by the **sampling distribution** of $\hat{\theta}$.



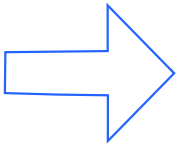
# The Purpose of Resampling



The **bias** of $\theta$ defined as the difference between the mean of the sampling distribution of $\hat{\theta}$ and $\theta$:

$$b(\hat{\theta}) = E(\hat{\theta}) - \theta$$

If we know the sampling distribution of $\hat{\theta}$, it might be possible to derive an explicit expression for $E(\hat{\theta})$. Alternatively, we *could* obtain an exact value for $E(\hat{\theta})$ by drawing every possible sample of size $n$ from the population, calculating $\hat{\theta}$ for each sample, and then averaging the result.



in most cases computationally impossible

# The Purpose of Resampling



Lastly, for a few special estimators, such as the sample mean, it is possible to analytically determine an explicit formula for the estimator's expected value (as long as certain assumptions are satisfied).

For example, we can mathematically determine that the sample mean, $\bar{X}$, is an unbiased estimator for the population mean (i.e., $b(\bar{X}) = 0$) as long as the realisations of the random variable in our sample are identical (**Central Limit Theorem**). Additionally, it was previously established that the standard error of the sample mean is

$$\sigma(\bar{X}) = \frac{\sigma}{\sqrt{N}}$$

# The Purpose of Resampling



Lastly, for a few special estimators, such as the sample mean, it is possible to analytically determine an explicit formula for the estimator's expected value (as long as certain assumptions are satisfied).

For example, we can mathematically determine that the sample mean, $\bar{X}$, is an unbiased estimator for the population mean (i.e., $b(\bar{X}) = 0$) as long as the realisations of the random variable in our sample are identical (**Central Limit Theorem**). Additionally, it was previously established that the standard error of the sample mean is

$$\sigma(\bar{X}) = \frac{\sigma}{\sqrt{N}}$$

However, for many other estimators, this is not possible.
In these cases, resampling methods provide an extremely useful approach to solving these problems.

# The Purpose of Resampling



Situations where the determination of an estimators properties is not straightforward include, but are not limited to:

**Distributional assumption violation:** Classical procedures rely on distributional assumptions regarding the population of interest. However, the population can be not well defined or the sample size is small.

# The Purpose of Resampling



Situations where the determination of an estimators properties is not straightforward include, but are not limited to:

**Distributional assumption violation:** Classical procedures rely on distributional assumptions regarding the population of interest. However, the population can be not well defined or the sample size is small.

**Non-random samples:** Certain processes of inferring population quantities require a sample to be random. However, often samples are not random: for example, questionnaires where people decide to participate, or an astronomical data set obtained from a specific region on the sky.

# The Purpose of Resampling



Situations where the determination of an estimators properties is not straightforward include, but are not limited to:

**Distributional assumption violation:** Classical procedures rely on distributional assumptions regarding the population of interest. However, the population can be not well defined or the sample size is small.

**Non-random samples:** Certain processes of inferring population quantities require a sample to be random. However, often samples are not random: for example, questionnaires where people decide to participate, or an astronomical data set obtained from a specific region on the sky.

**Small sample sizes:** Many classical methods for estimating properties of an estimator rely on the assumption of a large sample size. For smaller samples, these classical methods may result in invalid estimates.

# The Purpose of Resampling



Situations where the determination of an estimators properties is not straightforward include, but are not limited to:

**Distributional assumption violation:** Classical procedures rely on distributional assumptions regarding the population of interest. However, the population can be not well defined or the sample size is small.

**Non-random samples:** Certain processes of inferring population quantities require a sample to be random. However, often samples are not random: for example, questionnaires where people decide to participate, or an astronomical data set obtained from a specific region on the sky.

**Small sample sizes:** Many classical methods for estimating properties of an estimator rely on the assumption of a large sample size. For smaller samples, these classical methods may result in invalid estimates.

**Intractable calculations:** In some cases, either the assumed distribution or the estimator make it impossible finding explicit mathematical statements.

# The Purpose of Resampling



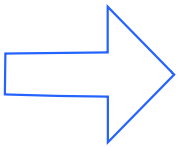
Situations where the determination of an estimators properties is not straightforward include, but are not limited to:

**Distributional assumption violation:** Classical procedures rely on distributional assumptions regarding the population of interest. However, the population can be not well defined or the sample size is small.

**Non-random samples:** Certain processes of inferring population quantities require a sample to be random. However, often samples are not random: for example, questionnaires where people decide to participate, or an astronomical data set obtained from a specific region on the sky.

**Small sample sizes:** Many classical methods for estimating properties of an estimator rely on the assumption of a large sample size. For smaller samples, these classical methods may result in invalid estimates.

**Intractable calculations:** In some cases, either the assumed distribution or the estimator make it impossible finding explicit mathematical statements.



In these situations, resampling techniques can provide a solution.

# Resampling Methods



In this lecture, we will discuss some of the most commonly used resampling methods: Jackknife, cross-validation, bootstrap and Monte Carlo.

They have important applications in statistics and machine learning:

For example, **cross-validation** is a common approach to estimate the test error associated with a given statistical learning method in order to evaluate its performance, or to select the appropriate level of flexibility.

The **bootstrap** is used in several contexts, most commonly to provide a measure of accuracy of a parameter estimate or of a given statistical learning method.

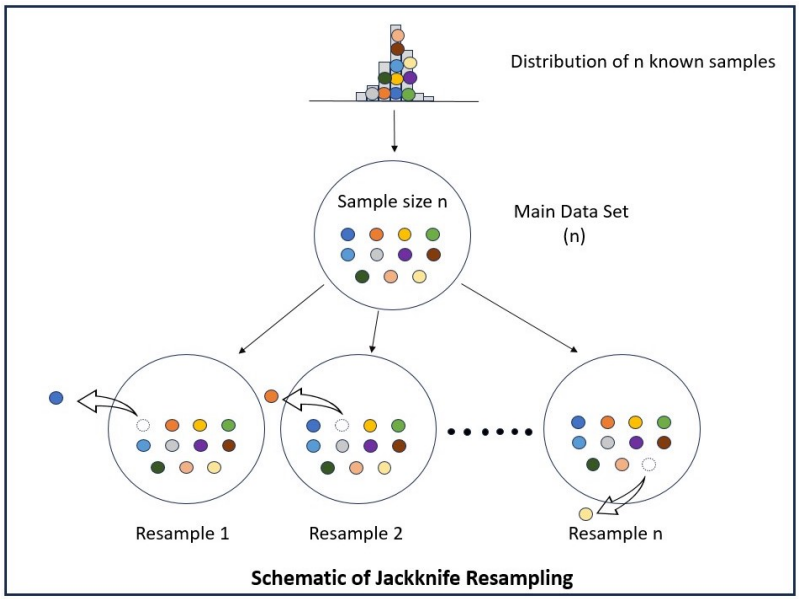
From the resampling method **Monte Carlo**, a sampling method called Markov Chain Monte Carlo is derived, a widely used optimizer for maximum likelihood approaches.

# Jackknife



M.H. Quenouille introduced the jackknife in 1949. His motivation was to construct an estimator of bias that could be used in general situations. This method is also used to provide an estimate of the variance of an estimator.

**Basic idea:** systematically recomputing the statistic estimate, leaving out one or more observations at a time from the sample set. From this new set of replicates of the statistic, an estimate for the bias and an estimate for the variance of the statistic can be calculated.



Schematic of Jackknife Resampling

# Jackknife



Denote the estimator of $\theta$ by $\hat{\theta}$, where $\hat{\theta}$ is based on the sample size $n$.

The **jackknife estimator** $\hat{\theta}_{JK}$ of $\theta$ is defined as follows:

Calculate $n$ estimators $\hat{\theta}_{(i)}$ where for each i in 1 to $n$, $\hat{\theta}_{(i)}$ is obtained using the expression defining $\hat{\theta}$ eliminating the $i$th observation so that each $\hat{\theta}_{(i)}$ is calculated with a sample of size $n-1$.

For this reason, the jackknife is often known as **leave-one-out method**.

If we now define the mean of the $\hat{\theta}_{(i)}$, $i = 1, ..., n$, as

$$\hat{\theta}_{(\bullet)} = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_{(i)},$$

the jackknife estimate of $\theta$ is

$$\hat{\theta}_{JK} = n\hat{\theta} - (n-1)\hat{\theta}_{(\bullet)}.$$

The jackknife estimate of bias is

$$b_{JK}(\hat{\theta}) = (n-1)(\hat{\theta}_{(\bullet)} - \hat{\theta}).$$

# Jackknife



In 1958, J. Tukey proposed a jackknife estimate for the variance of any sample analogue estimator $\hat{\theta}$. This can be written as

$$var_{JK}(\hat{\theta}) = \frac{(n-1)}{n} \sum_{i=1}^{n} (\hat{\theta}_i - \hat{\theta}_{(\bullet)})^2$$

and hence the jackknife estimate of the standard error of $\hat{\theta}$ is simply

$$\sigma_{JK}(\hat{\theta}) = \sqrt{var_{JK}(\hat{\theta})}$$

# Jackknife: Applications



The jackknife can be used for any estimator that is a sample analogue of a parameter.

**example:** The sample mean as an estimator of the population mean, the sample variance as an estimator of the population variance, the sample minimum as an estimator of the population minimum and so on. This definition can be extended to any population characteristic. For example, we may be interested in the parameter $\gamma$ defined as the ratio of the proportion of the population which is above the value $x$ and the proportion which is below the value $y$ (where $x < y$ and both are fixed, known numbers).

We can estimate the parameter $\hat{\gamma}$. However, without further information about the distribution of the random variable defining the population, there are no obvious theoretical results that can be used to approximate the sampling distribution of $\hat{\gamma}$, or its standard error or bias.

For such situation, the jackknife procedure is designed.

# Jackknife



Problems with this approach:

- The jackknife approach may fail when te estimator is not normally distributed.
- It may be unreliable for small datasets.
- This provides an estimated correction of bias due to the estimation method. The jackknife does not correct for a biased sample.

# Cross-Validation



We previously discussed the distinction between the **training error rate** and the **test error rate**:
The training error rate occurs during training, using the training set.
The test error results from using a statistical learning method to predict the response on a new observation - that is, a measurement that was not used in training the method.

# Cross-Validation



We previously discussed the distinction between the **training error rate** and the **test error rate**:
The training error rate occurs during training, using the training set.
The test error results from using a statistical learning method to predict the response on a new observation - that is, a measurement that was not used in training the method.

Cross-validation (CV) is a statistical method used to test a model's ability to predict unseen data (data not used to train the model). CV is useful if we have limited data when our test set is not large enough.

# Cross-Validation



We previously discussed the distinction between the **training error rate** and the **test error rate**:
The training error rate occurs during training, using the training set.
The test error results from using a statistical learning method to predict the response on a new observation - that is, a measurement that was not used in training the method.

Cross-validation (CV) is a statistical method used to test a model's ability to predict unseen data (data not used to train the model). CV is useful if we have limited data when our test set is not large enough.

**principle:**
Subsets of the data are held out for use as validation sets; a model is fit to the remaining data (a training set) and used to predict for the validation set. Averaging the quality of the predictions across the validation sets yields an overall measure of prediction accuracy. Cross-validation is employed repeatedly in building decision trees.

# Cross-Validation



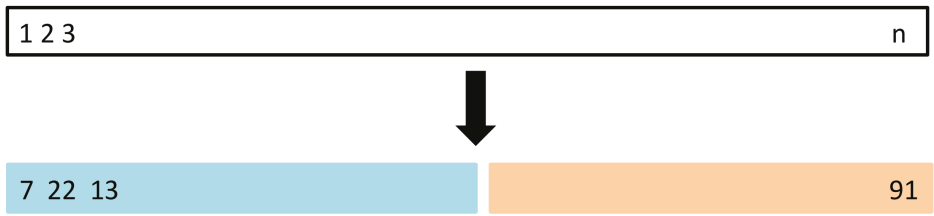
Using cross-validation avoids "self-influence". For comparison, in regression analysis methods such as linear regression, each $y$ value draws the regression line toward itself, making the prediction of that value appear more accurate than it really is. Cross-validation applied to linear regression predicts the $y$ value for each observation without using that observation.

Cross-validation is often used for deciding how many predictor variables to use in regression. Without cross-validation, adding predictors always reduces the residual sum of squares (or possibly leaves it unchanged). In contrast, the cross-validated mean-square error will tend to decrease if valuable predictors are added, but increase if worthless predictors are added.

# Cross-Validation



Suppose we want to estimate the test error associated with fitting a particular statistical learning method on a set of observations.
The validation set approach, illustrated in the figure below, is a very simple strategy for this task. It involves randomly dividing the available set of observations into two parts, a **training set** and a **validation set** (also known as a **hold-out set**).

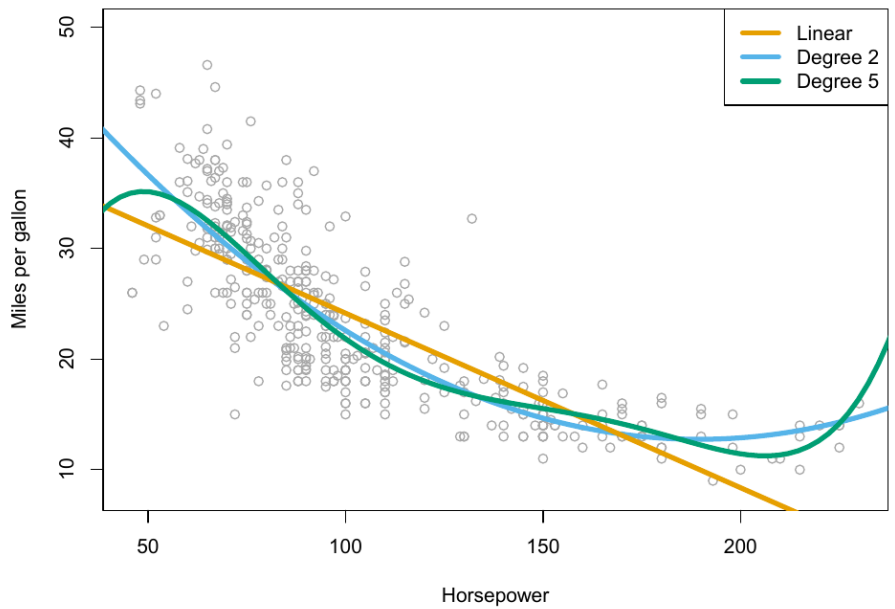



A set of $n$ observations are randomly split into a training set (shown in blue) and a validation set (shown in beige). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.
The resulting validation set error rate (typically assessed using MSE in the case of a quantitative response) provides an estimate of the test error rate.
Source: Fig. 5.1 from `https://www.statlearning.com/`

# Cross-Validation



We illustrate the validation set approach on the `Auto` data set. Recall from Lecture 3 that there appears to be a non-linear relationship between `mpg` and `horsepower`, and that a model that predicts `mpg` using `horsepower` and $\texttt{horsepower}^2$ gives better results than a model that uses only a linear term.




Source: Fig. 3.8 from `https://www.statlearning.com/`

# Cross-Validation

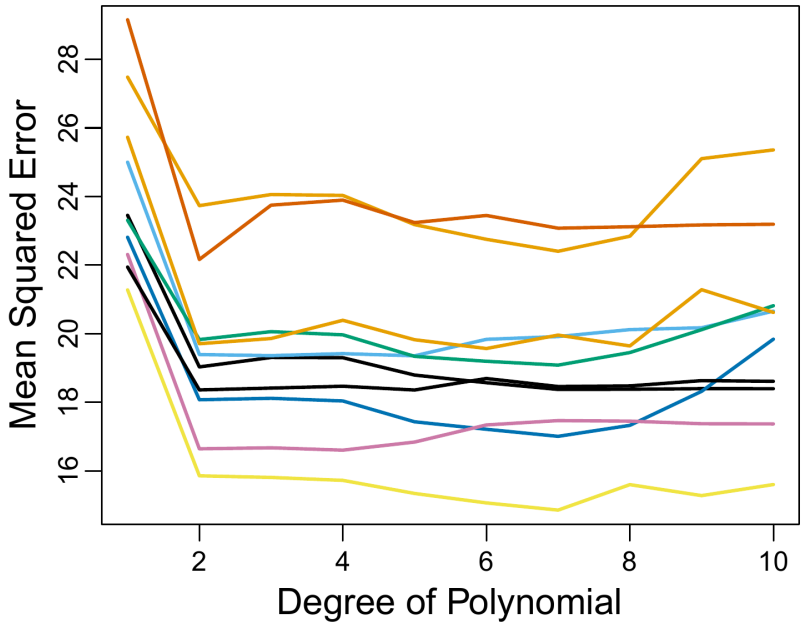


Does a cubic or higher-order fit provide even better results?
We did answer this question by looking at the p-values associated with a cubic term and higher-order polynomial terms in a linear regression. But we could also answer this question using the cross-validation method.

We randomly split the 392 observations into a training set containing 196 of the data points, and a validation set containing the remaining 196 observations. We repeat this ten times, each time using a different random 50:50 split. The test MSE is then calculated.

# Cross-Validation






Each color represents a different random split of the observations into a training set and a validation set. This illustrates the variability in the estimated test MSE that results from this approach.
Source: Fig. 5.2 from `https://www.statlearning.com/`

The validation set MSE for the quadratic fit is considerably smaller than for the linear fit. However, the validation set MSE for the cubic fit is actually slightly larger than for the quadratic fit. This implies that including a cubic term does not lead to better regression prediction over the quadratic term.

# Cross-Validation



The validation set approach as shown is conceptually simple and is easy to implement. But it has two potential drawbacks:

1. As shown in the previous figure, the validation estimate of the test error rate can be highly variable, depending on the precise selection of which observations to include in the training vs. validation set.

2. Only a subset of the observations - those that are included in the training set rather than in the validation set - are used to fit the model. Since statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

# Cross-Validation



The validation set approach as shown is conceptually simple and is easy to implement. But it has two potential drawbacks:

1. As shown in the previous figure, the validation estimate of the test error rate can be highly variable, depending on the precise selection of which observations to include in the training vs. validation set.

2. Only a subset of the observations - those that are included in the training set rather than in the validation set - are used to fit the model. Since statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

**Solution:** Carrying out a similar calculation **in turns**, taking **all results** from all subsamples into account.

# Leave-One-Out Cross-Validation



Leave-one-out cross-validation (LOOCV) uses a single observation $(x_1, y_1)$ for the validation set, and the remaining observations $\{(x_2, y_2), ..., (x_n, y_n)\}$ make up the training set.

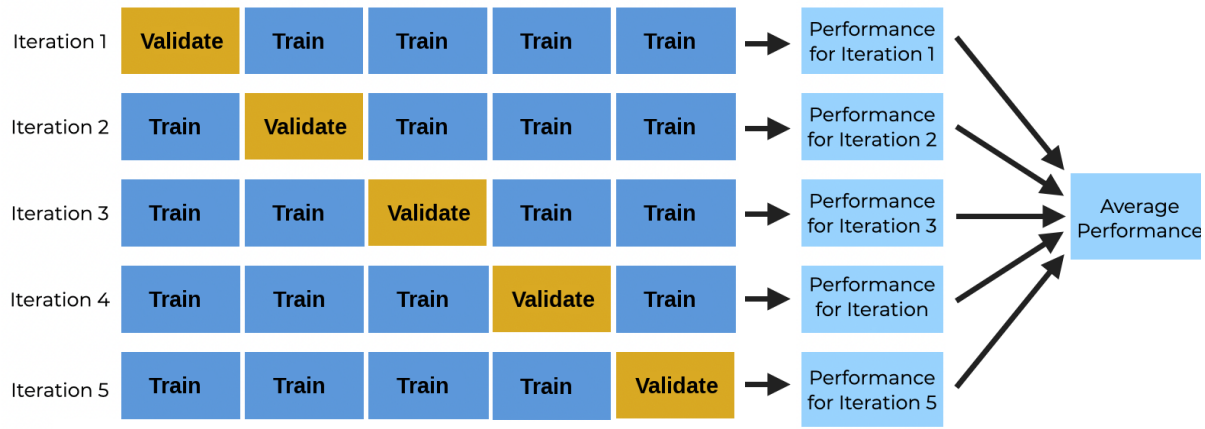
The statistical learning method is fit on the $n - 1$ training observations, and a prediction $\hat{y}_1$ is made for the excluded observation, using $x_1$.

As $(x_1, y_1)$ was not used in the fitting process, $MSE_1 = (y_1 - \hat{y}_1)^2$ provides an approximately unbiased estimate for the test error. But even though $MSE_1$ is unbiased for the test error, it is a poor estimate because it is highly variable, since it is based upon a single observation $(x_1, y_1)$.

# Leave-One-Out Cross-Validation



**Solution:** We can repeat the procedure by selecting $(x_2, y_2)$ for the validation data, training the statistical learning procedure on the $n-1$ observations $\{(x_1, y_1), (x_3, y_3), ..., (x_n, y_n)\}$ and compute $MSE_2 = (y_2 - \hat{y}_2)^2$. Repeating this $n$ times produces $n$ squared errors, $MSE_1, ..., MSE_n$.




The LOOCV estimate for the test MSE is the average of the $n$ test error estimates:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} MSE_i.$$

# Leave-One-Out Cross-Validation



LOOCV has a couple of **major advantages** over the classical validation set approach:

1. Far less bias. In LOOCV, we repeatedly fit the statistical learning method using training sets that contain $n - 1$ observations, almost as many as are in the entire data set. This is in contrast to the validation set approach, in which the training set is typically around half the size of the original data set. Consequently, the LOOCV approach tends not to overestimate the test error rate as much as the validation set approach does.

2. Consistent result. In contrast to the validation approach which will yield different results when applied repeatedly due to randomness in the training/validation set splits, performing LOOCV multiple times will always yield the same results: there is no randomness in the training/validation set splits.

# Leave-One-Out Cross-Validation



LOOCV has the potential to be **computationally expensive**, as the model has to be fit $n$ times.

When applied to least squares linear or polynomial regression, an amazing shortcut makes the cost of LOOCV the same as that of a single model fit! The following formula holds:

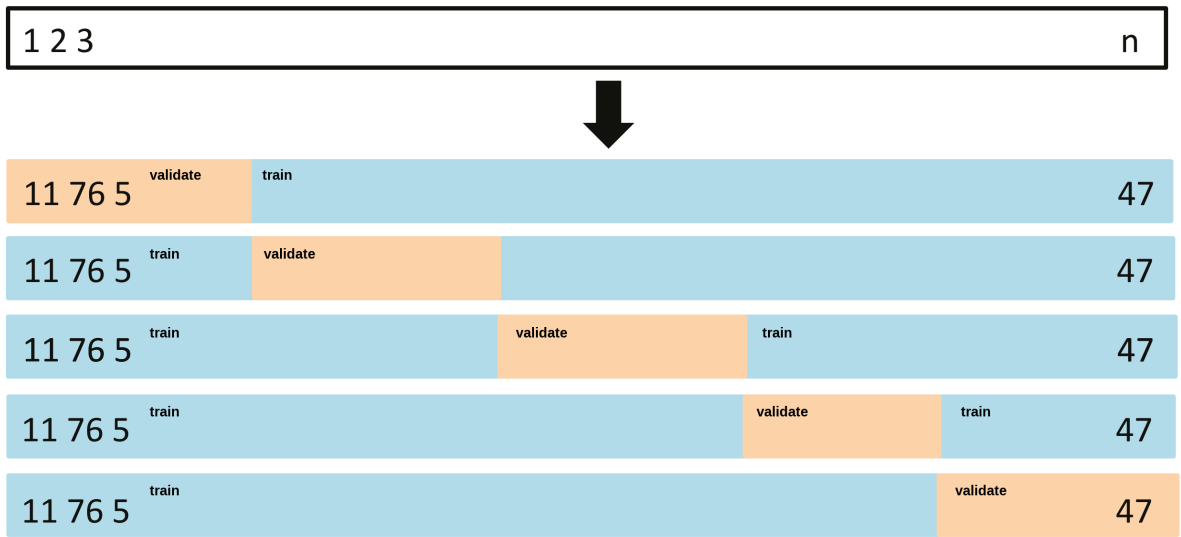
$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \left( \frac{y_i - \hat{y}_i}{1 - h_i} \right)^2 ,$$

where $\hat{y}_i$ is the $i$th fitted value from the original least squares fit, and $h_i$ is the **leverage** we have introduced before.

# $k$-Fold Cross-Validation



An alternative to LOOCV is $k$-fold CV. This approach involves randomly dividing the set of observations into $k$ groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $k-1$ folds. The mean squared error, $MSE_1$, is then computed on the observations in the held-out fold. This procedure is repeated $k$ times; each time, a different group of observations is treated as a validation set.




A schematic display of 5-fold CV.
Source: Fig. 5.5 from `https://www.statlearning.com/`

# $k$-Fold Cross-Validation



This process results in $k$ estimates of the test error, $MSE_1, MSE_2, ..., MSE_k$. The $k$-fold CV estimate is computed by averaging these values,

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i.$$

# $k$-Fold Cross-Validation



This process results in $k$ estimates of the test error, $MSE_1, MSE_2, ..., MSE_k$. The $k$-fold CV estimate is computed by averaging these values,

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i.$$

What is the advantage of using $k = 5$ or $k = 10$ rather than $k = n$?

# $k$-Fold Cross-Validation



This process results in $k$ estimates of the test error, $MSE_1, MSE_2, ..., MSE_k$. The $k$-fold CV estimate is computed by averaging these values,

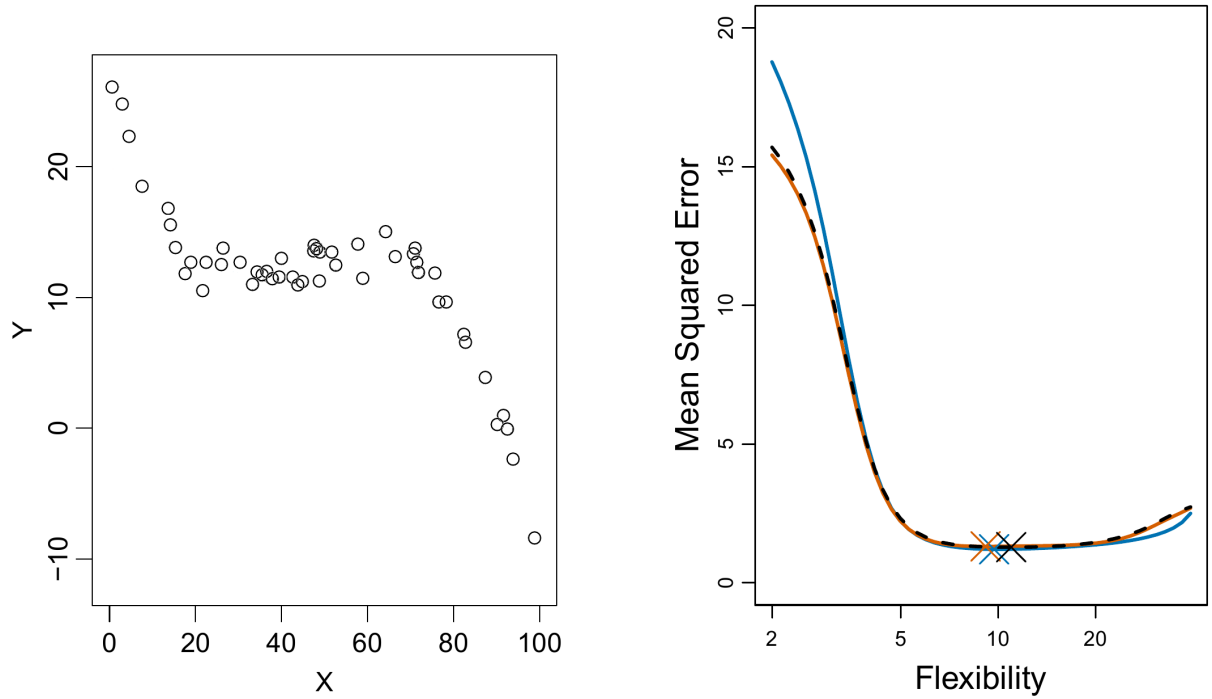
$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i.$$

What is the advantage of using $k = 5$ or $k = 10$ rather than $k = n$?

**Computational advantace:**
LOOCV requires fitting the statistical learning method $n$ times. Some statistical learning methods have computationally intensive fitting procedures, and so performing LOOCV may pose computational problems, especially if $n$ is extremely large. In contrast, performing 10-fold CV requires fitting the learning procedure only ten times.

# $k$-Fold Cross-Validation






cross-validation estimates and true test error rates that result from applying smoothing splines to a simulated data set. Left: Data set. Right: True and estimated test MSE for fitting the smoothing splines to the simulated data set. The true test MSE is shown in blue, the LOOCV estimate is shown as a black dashed line, and the 10-fold CV estimate is shown in orange. The crosses indicate the minimum of each of the MSE curves.
Based on Fig. 2.11 and Fig. 5.6 from `https://www.statlearning.com/`

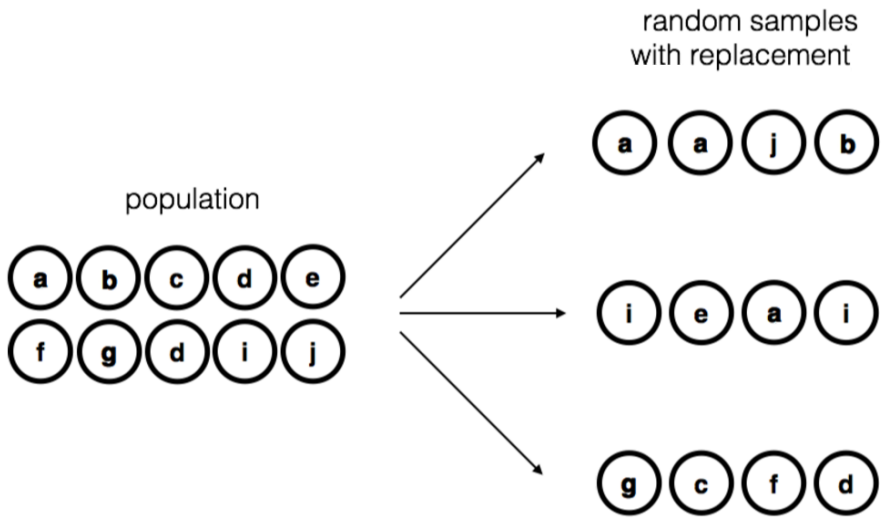
# Bootstrap



Bootstrapping is a statistical method for estimating the sampling distribution of an estimator by **sampling with replacement** from the original sample.

**Sampling with replacement:**

Selected elements are put back into the population before another element is sampled.

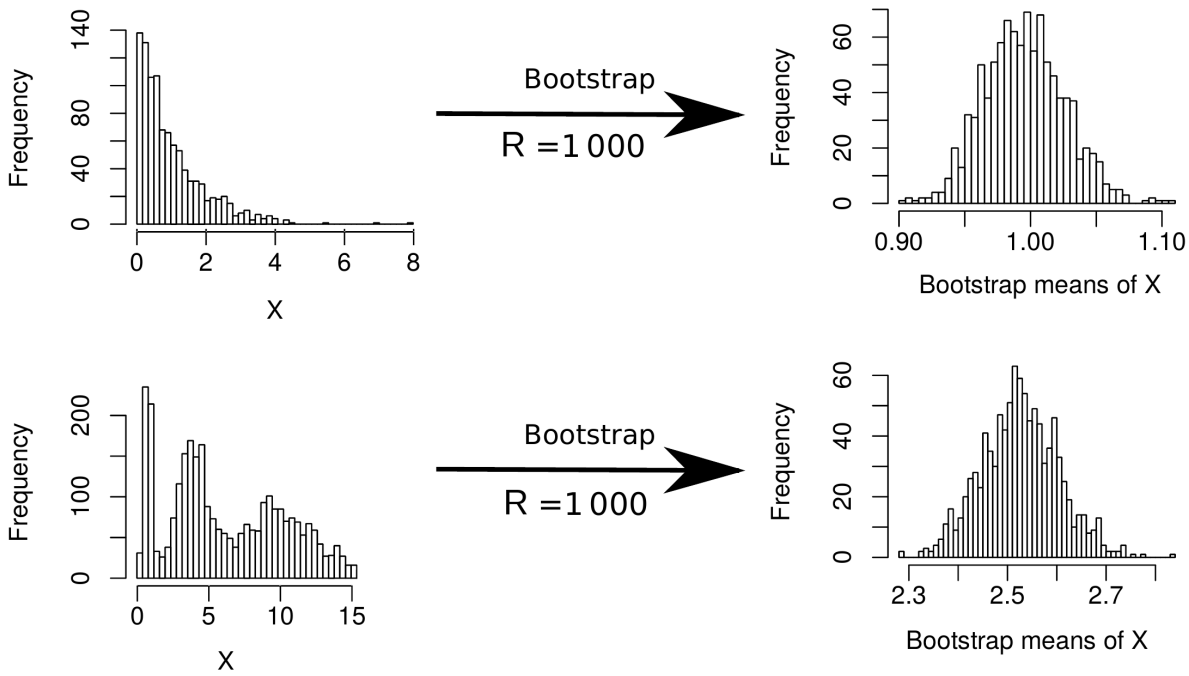


# Bootstrap



**Central limit theorem:**

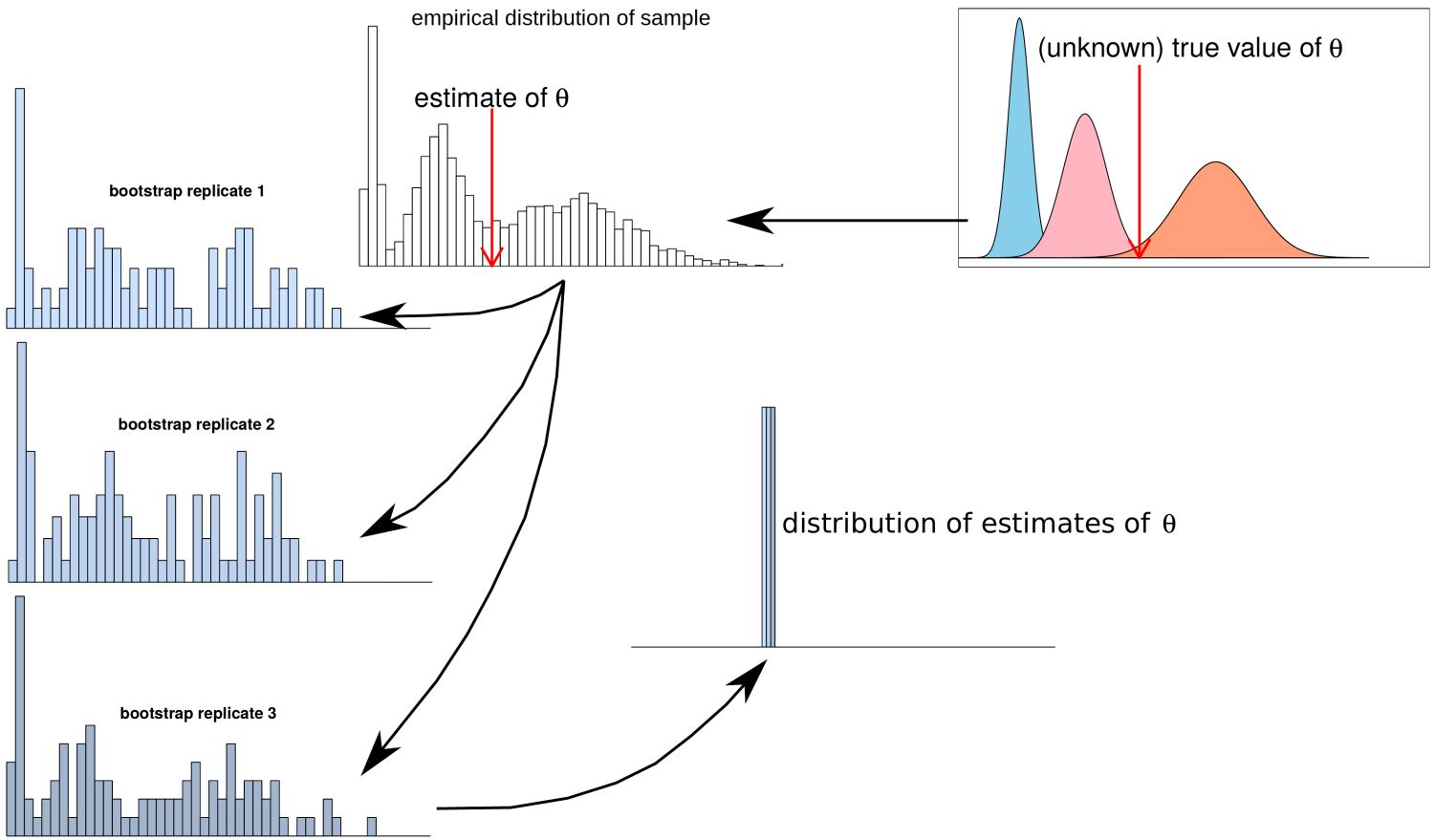
The means from a large number of independent random samples will be approximately normally distributed, regardless of the underlying distribution.



# Bootstrap



Bootstrap sampling from a distribution (a mixture of 3 normal distributions) to estimate the variance of the mean:



# Non-Parametric Bootstrap



In most cases, we will carry out the **Non-Parametric Bootstrap**:

1. Obtain an estimate $\hat{\theta}$ from the original sample values $\{x_1, .., x_n\}$.
2. For $i = 1, ..., R$, obtain an estimate $\hat{\theta}_i^*$ from a sample of size $n$ obtained by samping with replacement from the original sample values $\{x_1, .., x_n\}$.

# Non-Parametric Bootstrap



In most cases, we will carry out the **Non-Parametric Bootstrap**:

1. Obtain an estimate $\hat{\theta}$ from the original sample values $\{x_1, .., x_n\}$.

2. For $i = 1, ..., R$, obtain an estimate $\hat{\theta}_i^*$ from a sample of size $n$ obtained by samping with replacement from the original sample values $\{x_1, .., x_n\}$.

**example:** To infer the error in $\theta$, estimated from a dataset $x_1, ..., x_n$, carry out the following $R$ times (e.g. $R = 1000$):

1. Draw a *bootstrap sample* by sampling $n$ times with replacement from the sample. Call these $X_1^*, .., X_n^*$. Note that some points are represented more than once in the bootstrap samples, some once, some not at all.

2. Estimate $\theta$ from the bootstrap sample, call this $\hat{\theta}_k^*$ $(k = 1, ..., R)$.

4. When all $R$ bootstrap samples are done, the distribution of $\hat{\theta}_k^*$ estimates the distribution one would get if one were able to draw repeated samples of $n$ points from the unknown true distribution.

# Parametric Bootstrap



A variant of the above procedure, applicable when we are assume a certain pdf $f_\theta$ for the underlying population, is the parametric bootstrap:

1. Obtain an estimate $\hat{\theta}$ from the original sample values $\{x_1, .., x_n\}$.
2. For $i = 1, ..., R$, obtain an estimate $\hat{\theta}_i^*$ from a sample of size $n$ drawn from $f_\theta$.

These $R$ estimates are used to construct an **empirical** sampling distribution for $\hat{\theta}$, which can then be used to estimate any property of interest of the estimator.

For instance, a bootstrap estimate of the expectaction value of $\theta$, $E(\hat{\theta})$, $\hat{\theta}_{BS}$, is simply the mean of the $B$ bootstrap estimates. A bootstrap estimate of $\sigma(\hat{\theta})$, $\sigma_{BS}(\hat{\theta})$, is obtained by calculating the standard deviation of the $B$ bootstrapped values of $\hat{\theta}$.

# The Bootstrap Principle



The basic assumption underlying the bootstrap is that, under certain conditions, the variability of $\hat{\theta}$ around the value $\theta$ can be assessed via the variability of $\hat{\theta}_i^*$ around the value $\hat{\theta}$. This assumption is often termed the **Bootstrap Principle**.

# The Bootstrap Principle



The basic assumption underlying the bootstrap is that, under certain conditions, the variability of $\hat{\theta}$ around the value $\theta$ can be assessed via the variability of $\hat{\theta}_i^*$ around the value $\hat{\theta}$. This assumption is often termed the **Bootstrap Principle**.

In many situations, but not always, these conditions are satisfied.

# The Bootstrap Principle



The basic assumption underlying the bootstrap is that, under certain conditions, the variability of $\hat{\theta}$ around the value $\theta$ can be assessed via the variability of $\hat{\theta}_i^*$ around the value $\hat{\theta}$. This assumption is often termed the **Bootstrap Principle**.

In many situations, but not always, these conditions are satisfied.

**example:** The bootstrap cannot be applied to time series data, as the time-dependence is lost.

# Choosing $R$



Choosing $R$ is typically a trade-off between computation time and how accurately we wish to approximate the sampling distribution of $\hat{\theta}$.

This is mitigated to some extent by noting that resampling is a **parallel computational technique** (each resample can be done independently of the others).

In case parallelizing is not possible, general rule of thumb is to take $R = 1000$.

# Comparing Bootstrap with Jackknife



B. Efron introduced the concept of the bootstrap in 1979. It is conceptually simpler than the jackknife, although it usually involves more computation. To further compare the two resampling methods, the jackknife method requires computing the estimator $\hat{\theta}$ an amount of $n$ times, where each computation is based on $(n-1)$ observations.

In contrast, both the **non-parametric** and **parametric** bootstrap require us to calculate the sample analogue estimator $\hat{\theta}$ a large number of times (say, $R$), each of which is based on a sample size $n$ obtaining by either **sampling with replacement from the original data** or by **generating samples from the population inferred from the data sample**.

# Monte Carlo Cross-Validation



**Monte Carlo Cross-Validation**, also known as **Subsampling**, is an alternative method for approximating the sampling distribution of an estimator.

The two key differences to the bootstrap are:

- the resample size is smaller than the sample size and
- resampling is done without replacement.

**Sampling without replacement:**
Selected elements are not put back into the population, so cannot be sampled multiple times.

# Monte Carlo Cross-Validation



**advantage:**

Subsampling is valid under much weaker conditions compared to the bootstrap. In particular, a set of sufficient conditions is that the rate of convergence of the estimator is known and that the limiting distribution is continuous.

# Monte Carlo Cross-Validation



**advantage:**

Subsampling is valid under much weaker conditions compared to the bootstrap. In particular, a set of sufficient conditions is that the rate of convergence of the estimator is known and that the limiting distribution is continuous.

While subsampling was originally proposed for the case of independent and identically distributed (iid) data only, the methodology has been extended to cover **time series** data as well; in this case, one resamples blocks of subsequent data (to keep the time dependence) rather than individual data points.

# Markov Chain Monte Carlo



Related to the Monte Carlo Cross-Validation as a subsampling technique is the **Markov Chain Monte Carlo**, a technique to draw samples from a known probability distribution.

**Markov chain Monte Carlo (MCMC)** methods are used to study probability distributions that are too complex or too highly dimensional to study with analytic techniques alone.

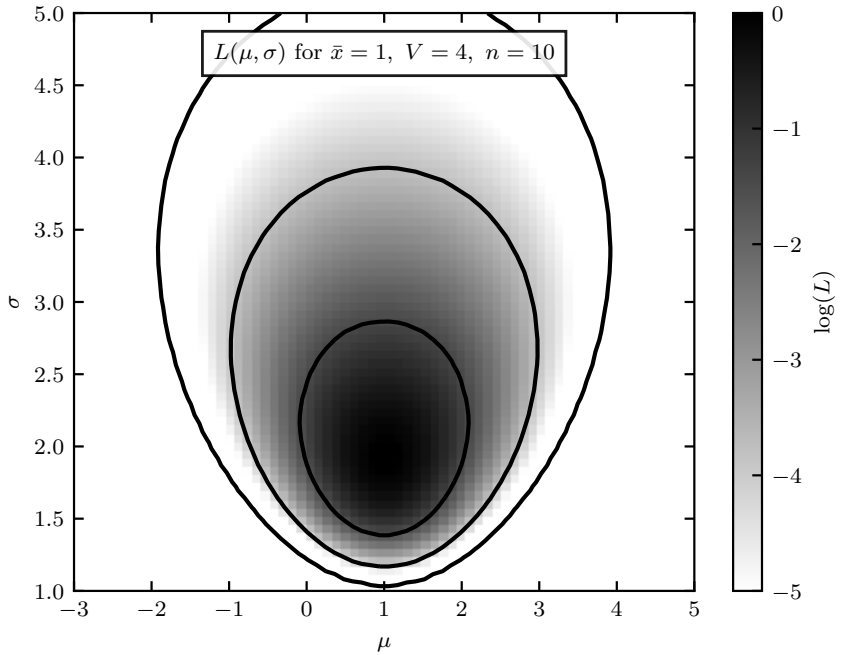
MCMC methods are used for calculating numerical approximations of multi-dimensional integrals, for example in Bayesian statistics, computational physics, computational biology and computational linguistics. A common application is **finding the maximum of a pdf** when an analytical solution is not possible, but the equation for the pdf is known.

# Monte Carlo Methods & Markov Chains



**motivation:**
estimating location and scale parameters for homoscedastic data drawn from a Gaussian distribution - two-dimensional posterior pdf for $\mu$ and $\sigma$

solution: numerically integrate the posterior pdf (easy in this case) and find its maximum by using a brute-force grid search

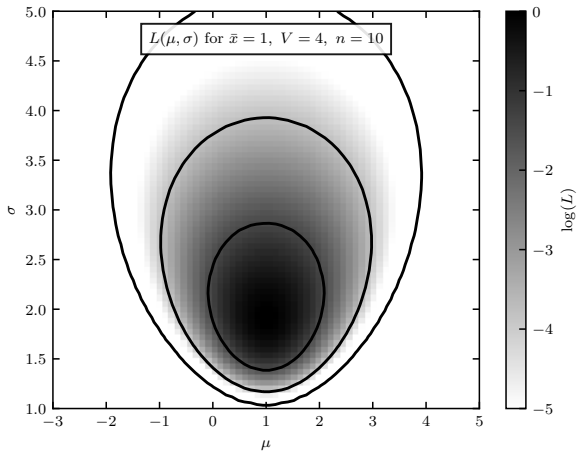


# Monte Carlo Methods & Markov Chains



**motivation:**
lecture 5: estimating location and scale parameters for homoscedastic data drawn from a Gaussian distribution - two-dimensional posterior pdf for $\mu$ and $\sigma$

solution: numerically integrate the posterior pdf (easy in this case) and find its maximum by using a brute-force grid search



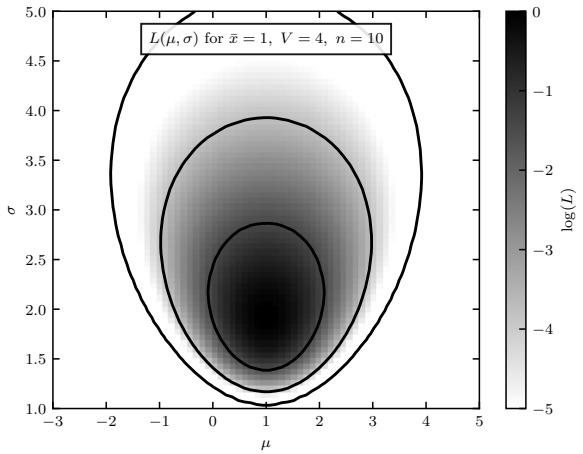

Feasible in this case: With 100 grid points per coordinate it was only $100 \times 100$ values.

# Monte Carlo Methods & Markov Chains



**motivation:**
lecture 5: estimating location and scale parameters for homoscedastic data drawn from a Gaussian distribution - two-dimensional posterior pdf for $\mu$ and $\sigma$

solution: numerically integrate the posterior pdf (easy in this case) and find its maximum by using a brute-force grid search




Feasible in this case: With 100 grid points per coordinate it was only $100 \times 100$ values.

**But what about high-dimensional parameter spaces?**

# Monte Carlo Methods & Markov Chains



**challenges:**

- high-dimensional parameter space

computing time scales exponentially with number of parameters $k$

- spiky distributions

for odd shaped posterior distributions, brute-force grid methods can either totally miss the maximum of the pdf, or the grid needs to be very fine

# Monte Carlo Methods & Markov Chains



**challenges:**

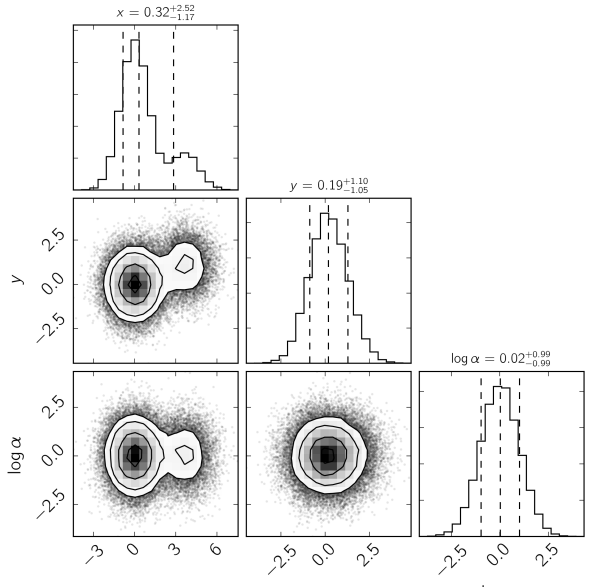
- high-dimensional parameter space

computing time scales exponentially with number of parameters $k$

- spiky distributions

for odd shaped posterior distributions, brute-force grid methods can either totally miss the maximum of the pdf, or the grid needs to be very fine

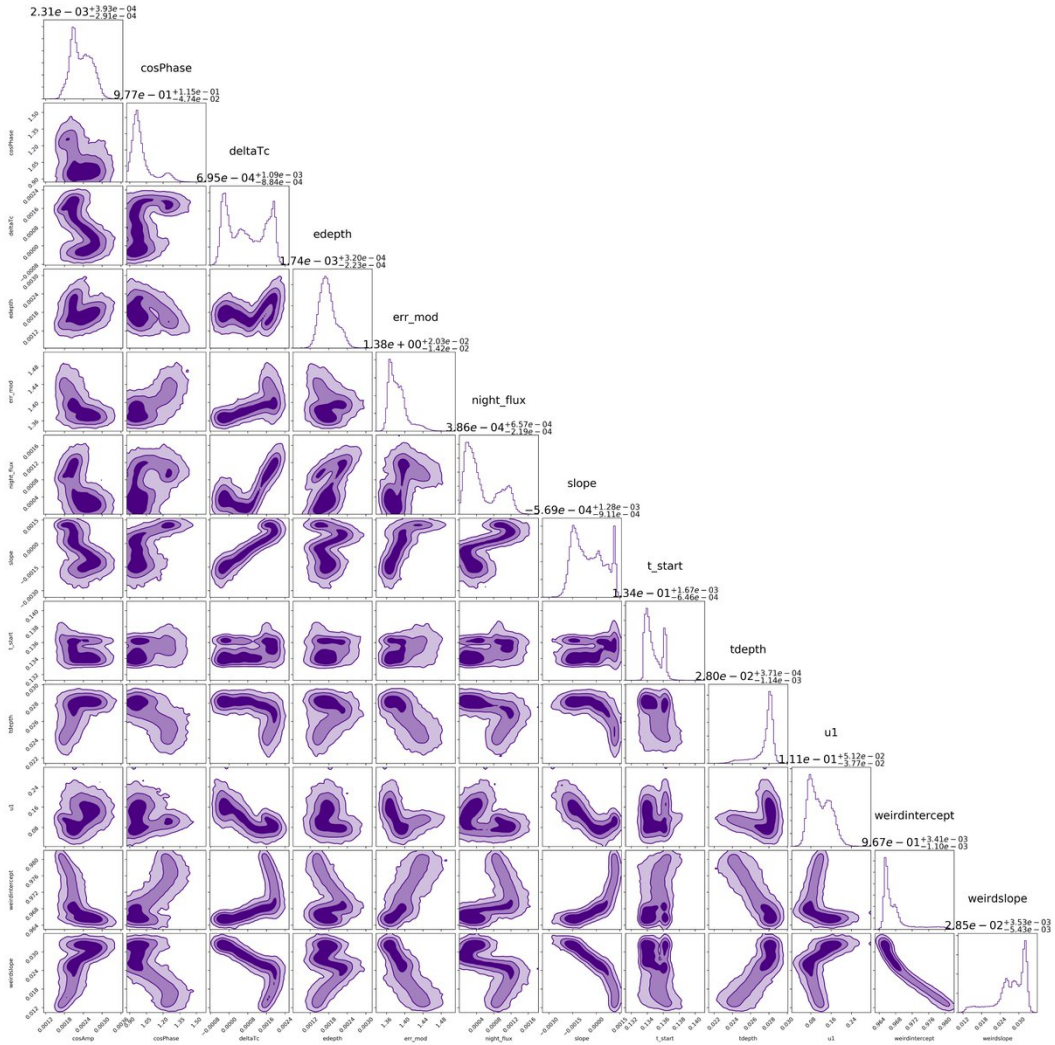
**example realistic posterior distribution (I):**



# Monte Carlo Methods & Markov Chains



**example realistic posterior distribution (II):**

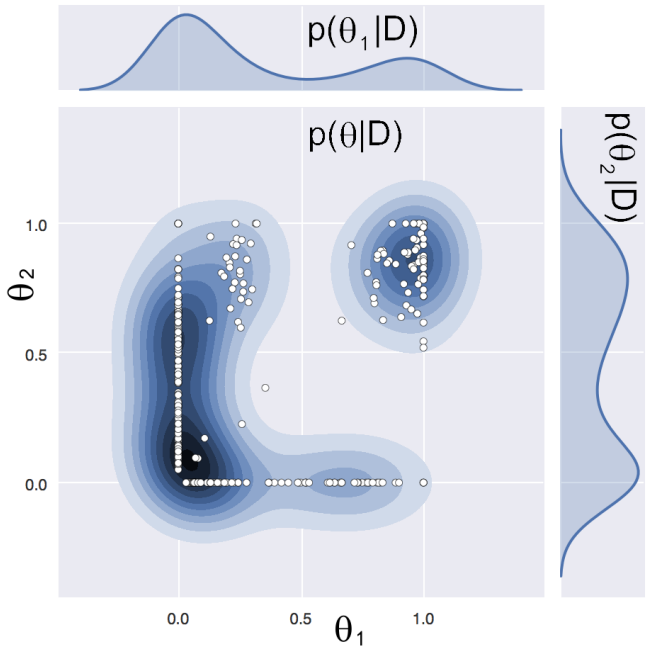


# Monte Carlo Methods & Markov Chains



**motivation:**

a better way to solve this is an algorithm that **samples the full multi-dimensional parameter space**, in a way that builds up the most sample density in regions that are closest to the maximum probability



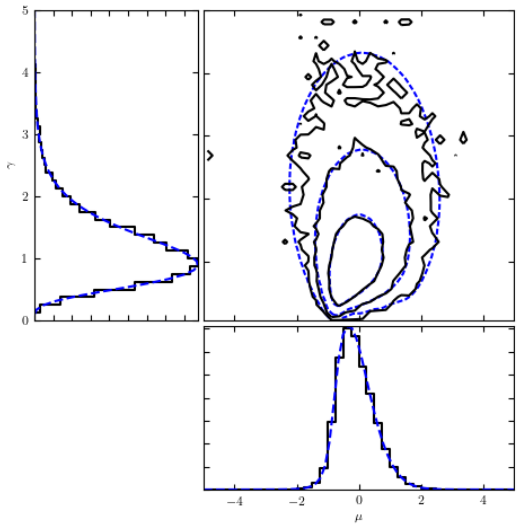
# Monte Carlo Methods & Markov Chains



**motivation:**

a better way to solve this is an algorithm that **samples the full multi-dimensional parameter space**, in a way that builds up the most sample density in regions that are closest to the maximum probability

exactly this is the **Markov Chain Monte Carlo (MCMC)** algorithm



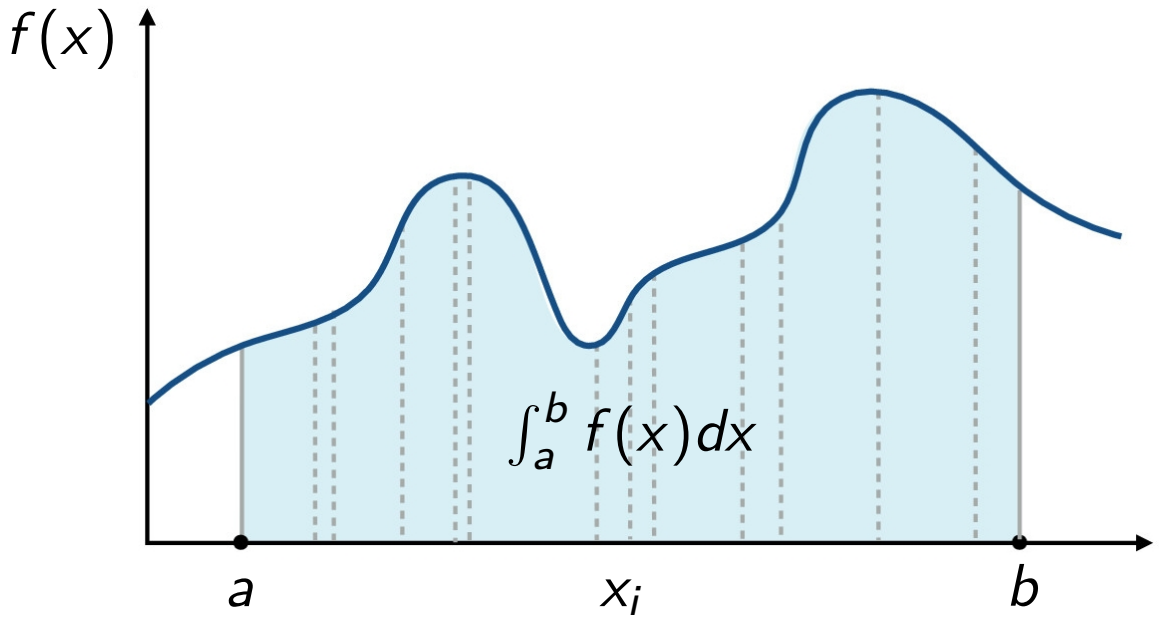

The dashed lines are the known (analytic) solution. The solid lines are from the MCMC estimate with 10,000 sample points.

# Monte Carlo Methods

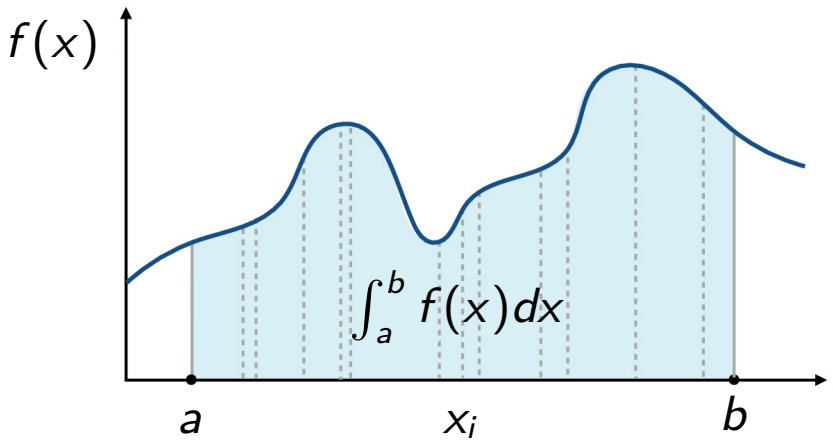


Algorithms from the family of **Monte Carlo methods** use **random sampling** to obtain a numerical result where there is no analytic result or it is difficult to compute.

**Monte Carlo integration:**
simple idea: estimate the integral of a function by averaging random samples of the function's value



**Monte Carlo integration:**
simple idea: estimate the integral of a function by averaging random samples of the function's value




define integral:
$\int_a^b f(x)dx$

random variable: $X_i \sim p(x)$
note: $p(x)$ must be nonzero for all $x$ where $f(x)$ is nonzero

**Monte Carlo estimator**

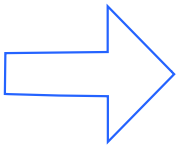
$$F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}$$

# Monte Carlo Methods



**Monte Carlo integration:**
simple idea: estimate the integral of a function by averaging random samples of the function's value

**improvement:** sample the integrand according to how much we expect it to contribute to the integral



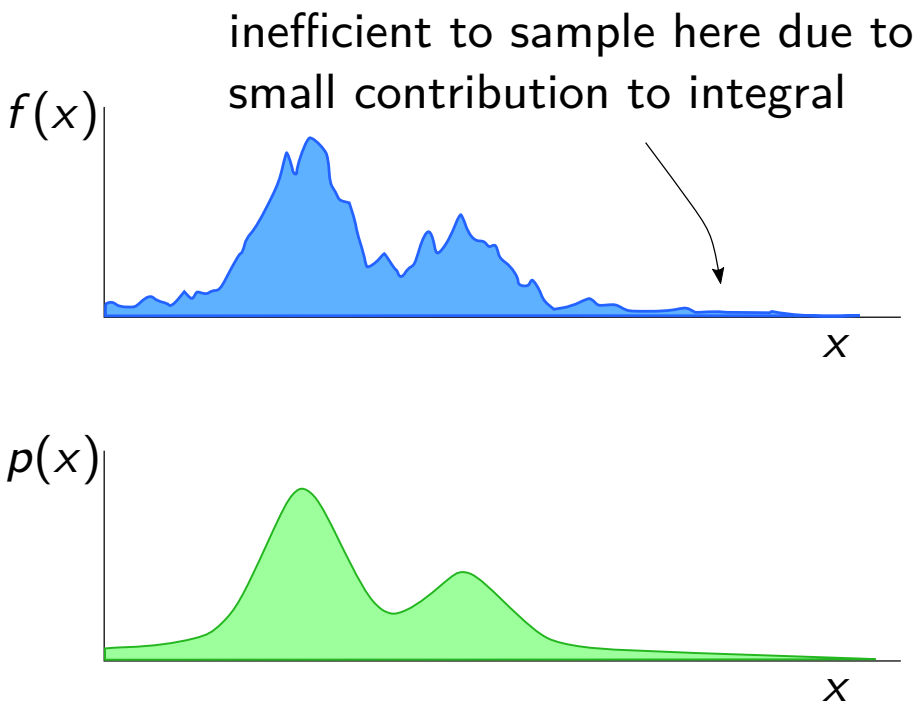
**Importance Sampling**

# Monte Carlo Methods



**Importance Sampling:**

**improvement:** sample the integrand according to how much we expect it to contribute to the integral




**Basic Monte Carlo**

$F_N = \frac{b-a}{N} \sum_{i=1}^{N} f(X_i)$
where $x_i$ are sampled uniformly

**Importance-Sampled Monte Carlo**

$F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$
where $x_i$ are sampled proportional to $p$

# Markov-Chain Monte Carlo Methods



A **Markov Chain** is defined as

a sequence of random variables where a parameter depends only on the preceding value. Such processes are **memoryless**.

# Markov-Chain Monte Carlo Methods



A **Markov Chain** is defined as

a sequence of random variables where a parameter depends only on the preceding value. Such processes are **memoryless**.

We thus have

$$p(\theta_{i+1}|\theta_i, \theta_{i-1}, \theta_{i-2}, \cdots) = p(\theta_{i+1}|\theta_i).$$

For equilibrium, or a stationary distribution of positions, it is necessary that the transition probability is symmetric:

$$p(\theta_{i+1}|\,\theta_i) = p(\theta_i|\,\theta_{i+1}).$$

This is called the **principle of detailed balance** or reversibility condition (i.e. the probability of a jump between two points does not depend on the direction of the jump).

# Markov-Chain Monte Carlo Methods



The use of Markov chains to perform Monte Carlo integration is called **Markov Chain Monte Carlo (MCMC)**.

Given such a Markov chain of length $N$ that corresponds to draws of $p(\theta)$, integrals can be estimated as

$$\int g(\theta)\, p(\theta)\, d\theta \approx \frac{1}{N} \sum_{i=1}^{N} g(\theta_i).$$

# Markov-Chain Monte Carlo Methods



The use of Markov chains to perform Monte Carlo integration is called **Markov Chain Monte Carlo (MCMC)**.

Given such a Markov chain of length $N$ that corresponds to draws of $p(\theta)$, integrals can be estimated as

$$\int g(\theta)\, p(\theta)\, d\theta \approx \frac{1}{N} \sum_{i=1}^{N} g(\theta_i).$$

To estimate the expectation value for $\theta_1$ (i.e., $g(\theta) = \theta_1$), we simply take the mean value of all $\theta_1$ in the chain.

# Markov-Chain Monte Carlo Methods



The use of Markov chains to perform Monte Carlo integration is called **Markov Chain Monte Carlo (MCMC)**.

Given such a Markov chain of length $N$ that corresponds to draws of $p(\theta)$, integrals can be estimated as

$$\int g(\theta)\, p(\theta)\, d\theta \approx \frac{1}{N} \sum_{i=1}^{N} g(\theta_i).$$

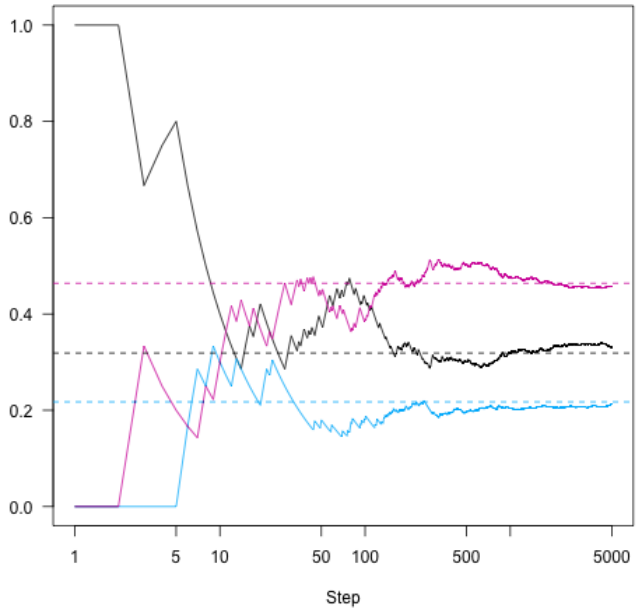
To estimate the expectation value for $\theta_1$ (i.e., $g(\theta) = \theta_1$), we simply take the mean value of all $\theta_1$ in the chain. To **visualize** the posterior pdf for

parameter $\theta_1$, marginalized over all other parameters, $\theta_2, \ldots, \theta_k$, we can construct a histogram of all $\theta_1$ values in the chain, and normalize its integral to 1.
To get an **estimate** for $\theta_1$, we find the maximum of this marginalized pdf.

# Markov-Chain Monte Carlo Methods



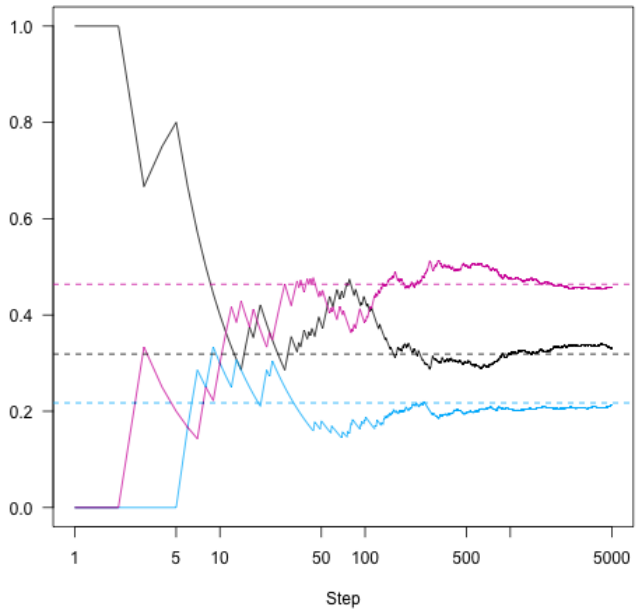
**trace plot** of a Markov Chain:



# Markov-Chain Monte Carlo Methods



**trace plot** of a Markov Chain:

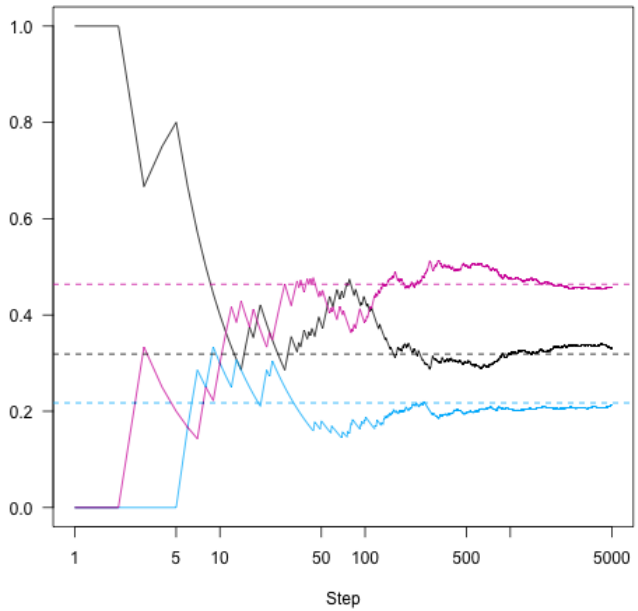



In MCMC the process must be **stationary**: the chain statistics look the same in each part of the chain.
Obviously that isn't going to be the case in the early steps of the chain (see plot).

# Markov-Chain Monte Carlo Methods



**trace plot** of a Markov Chain:




In MCMC the process must be **stationary**: the chain statistics look the same in each part of the chain.
Obviously that isn't going to be the case in the early steps of the chain (see plot).

**solution:** discard the first few steps, the ***burn-in phase***.
How many steps to discard? Make a trace plot.

# The Metropolis-Hastings Algorithm



The Metropolis-Hastings Algorithm is the most commonly used algorithm for MCMC sampling.

It adopts the following **acceptance probability** for newly proposed points $\theta^*$ to step to:

$$p_{\mathrm{acc}}(\theta_i, \theta^*) = \frac{p(\theta^*)}{p(\theta_i)},$$

where the proposed point $\theta^*$ is drawn from an arbitrary symmetric density distribution $q(\theta^* \mid \theta_i)$. Since it is symmetric, the **ratio of transition probabilities cancels out and detailed balance is ensured**. A Gaussian distribution centered on the current point $\theta_i$ is often used for $q(\theta^*|\theta_i)$.

This algorithm guarantees that the chain will reach an **equilibrium, or stationary, distribution**, and it will approximate a sample drawn from $p(\theta)$.

# The Metropolis-Hastings Algorithm



**Algorithm:** Metropolis-Hastings MCMC

**Input:** $p(\theta)$: probability distribution
$q(\theta)$: proposal distribution
$N_{\text{iter}}$: number of sample iterations

**Output:** $\{\theta_i\}$: chain of samples

**begin**

initialization $\theta_0$

**for** $i = 1, ..., N_{\text{iter}}$ **do**

sample $\theta^* \in q(\theta^*|\theta_i)$

$p_{\text{acc}} = \min\left(1, \frac{p(\theta^*)q(\theta_i|\theta^*)}{p(\theta_i)q(\theta^*|\theta_i)}\right)$

sample $u \in U[0, 1]$

**if** $u < p_{\text{acc}}$ **then**

$\theta_{i+1} = \theta^*$

**else**

$\theta_{i+1} = \theta_i$

# The Metropolis-Hastings Algorithm



**Algorithm:** Metropolis-Hastings MCMC

**Input:** $p(\theta)$: probability distribution
$q(\theta)$: proposal distribution
$N_{\text{iter}}$: number of sample iterations

**Output:** $\{\theta_i\}$: chain of samples

**begin**

initialization $\theta_0$

**for** $i = 1, ..., N_{\text{iter}}$ **do**

sample $\theta^* \in q(\theta^*|\theta_i)$

$p_{\text{acc}} = \min\left(1, \frac{p(\theta^*)q(\theta_i|\theta^*)}{p(\theta_i)q(\theta^*|\theta_i)}\right)$

sample $u \in U[0, 1]$

**if** $u < p_{\text{acc}}$ **then**

$\theta_{i+1} = \theta^*$

**else**

$\theta_{i+1} = \theta_i$

Given $\theta_i$ and $q(\theta_{i+1}|\theta_i)$, draw $\theta^*$ as proposed value for $\theta_{i+1}$.

# The Metropolis-Hastings Algorithm



**Algorithm:** Metropolis-Hastings MCMC

**Input:** $p(\theta)$: probability distribution
$q(\theta)$: proposal distribution
$N_{\text{iter}}$: number of sample iterations

**Output:** $\{\theta_i\}$: chain of samples

**begin**

initialization $\theta_0$

**for** $i = 1, ..., N_{\text{iter}}$ **do**

sample $\theta^* \in q(\theta^*|\theta_i)$

$p_{\text{acc}} = \min\left(1, \frac{p(\theta^*)q(\theta_i|\theta^*)}{p(\theta_i)q(\theta^*|\theta_i)}\right)$

sample $u \in U[0, 1]$

**if** $u < p_{\text{acc}}$ **then**

$\theta_{i+1} = \theta^*$

**else**

$\theta_{i+1} = \theta_i$

Compute acceptance probability $p_{\text{acc}}(\theta_i, \theta^*)$.

# The Metropolis-Hastings Algorithm



**Algorithm:** Metropolis-Hastings MCMC

**Input:** $p(\theta)$: probability distribution
$q(\theta)$: proposal distribution
$N_{\text{iter}}$: number of sample iterations

**Output:** $\{\theta_i\}$: chain of samples

**begin**

initialization $\theta_0$

**for** $i = 1, ..., N_{\text{iter}}$ **do**

sample $\theta^* \in q(\theta^*|\theta_i)$

$p_{\text{acc}} = \min\left(1, \frac{p(\theta^*)q(\theta_i|\theta^*)}{p(\theta_i)q(\theta^*|\theta_i)}\right)$

sample $u \in U[0, 1]$

**if** $u < p_{\text{acc}}$ **then**

$\theta_{i+1} = \theta^*$

**else**

$\theta_{i+1} = \theta_i$

Draw a random number between 0 and 1 from a uniform distribution; if it is smaller than $p_{\text{acc}}(\theta_i, \theta^*)$, then accept $\theta^*$ as $\theta_{i+1}$.
Purpose: If we only accepted points of higher probability we would find the maximum of the pdf, but not map out the full pdf.

# The Metropolis-Hastings Algorithm



**Algorithm:** Metropolis-Hastings MCMC

**Input:** $p(\theta)$: probability distribution
$q(\theta)$: proposal distribution
$N_{\text{iter}}$: number of sample iterations

**Output:** $\{\theta_i\}$: chain of samples

**begin**

initialization $\theta_0$

**for** $i = 1, ..., N_{\text{iter}}$ **do**

sample $\theta^* \in q(\theta^*|\theta_i)$

$p_{\text{acc}} = \min\left(1, \frac{p(\theta^*)q(\theta_i|\theta^*)}{p(\theta_i)q(\theta^*|\theta_i)}\right)$

sample $u \in U[0, 1]$

**if** $u < p_{\text{acc}}$ **then**

$\theta_{i+1} = \theta^*$

**else**

$\theta_{i+1} = \theta_i$

If $\theta^*$ is accepted, added it to the chain. If not, add $\theta_i$ to the chain.

# Practical MCMC Chain Checks



**check the acceptance rate:**

Some MCMC samplers give an updating estimate of the current acceptance rate of new samples. Ideally for a sampler using some form of Metropolis-Hastings, this should be somewhere between $\sim 20 - 50\%$ depending on the type of problem you're trying to solve.

- A high acceptance rate indicates that the chain is moving but might not be exploring the pdf well. This gives high acceptance rate but poor global exploration of the posterior distribution function.
- A low acceptance rate indicates that the chain is hardly moving, i.e. it's stuck in a rut or trying to jump to new points that are too far away.

# Practical MCMC Chain Checks



**check trace plots:**

Ideally, our traceplot in each parameter would be mixing well (moving across parameter space without getting stuck). This will tell you whether your chain is getting stuck or encountering inefficiencies.

# Practical MCMC Chain Checks



**check trace plots:**

Ideally, our traceplot in each parameter would be mixing well (moving across parameter space without getting stuck). This will tell you whether your chain is getting stuck or encountering inefficiencies.

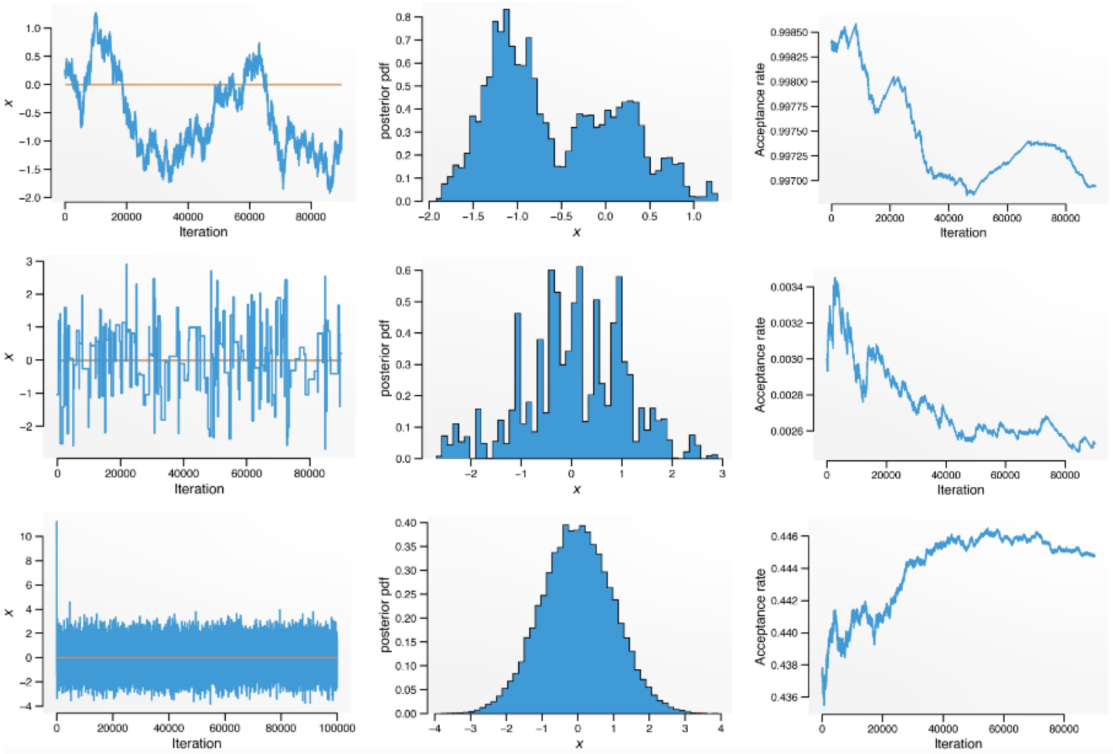
**check autocorrelation length:**

The MCMC chain with Metropolis-Hastings will not give fully-independent random samples. The next point is influenced by where the previous point was. We need to check how much to down-sample the chain so that the points lack memory and influence from others. This is given by the autocorrelation length.

# Practical MCMC Chain Checks



first column: trace plot, second column: histogram of the chain, third column: acceptance rate of newly proposed $\theta^*$




In the top row, the proposal width was too small. In the middle row, the proposal width was too big. Only the bottom row shows reasonable sampling.

# Summary



Regarding resampling and sampling techniques, despite they are extremely useful, there are several **caveats** where we must be cautious:

The most common criticisms within the statistical community include, but are not limited to:

**"Hidden" assumptions:** Every theorem, test, and procedure in statistics is built upon a set of assumptions. Resampling is no different. For instance, the use of bootstrapping to construct a confidence interval for a certain population parameter relies on the Bootstrap Principle assumption.
A potential problem with resampling is that these assumptions are not obvious or intuitive, thus bootstrapping might be applied in unappropriate situations.

# Summary



Regarding resampling and sampling techniques, despite they are extremely useful, there are several **caveats** where we must be cautious:

The most common criticisms within the statistical community include, but are not limited to:

**"Philosophical" issues:** We only have one particular sample from the population of interest. Can we really generalize based on "simulated" data, and on what grounds? Will the availability of procedures such as the bootstrap encourage a less diligent approach to data collection and sampling?

**Data quality:** If the collected sample is biased, or contains spurious observations or unusual features, conclusions based on the use of resampling techniques will be suspect. This is true because resampling makes repeated use of this data, thereby exacerbating any problems with the data. It is worth noting, however, that this is also an issue with classical statistical procedures.

# Summary



Today we have learned about **resampling and sampling techniques** that have a variety of applications, like for cross-validation or approximating integrals with MCMC.

Next time we will learn about **Classification**, where resampling and sampling techniques are of high relevance.