## Sets and Functions: Sets

**1** A **set** is an unordered collection of objects. The objects in a set are called *elements*.

**2** The **cardinality** of a set is the number of elements it contains. The **empty set** $\emptyset$ is the set with no elements.

**3** If every element of $A$ is also an element of $B$, then we say $A$ is a **subset** of $B$ and write $A \subset B$. If $A \subset B$ and $B \subset A$, then we say that $A = B$.

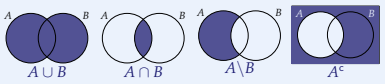
**4** Set operations:

(i) An element is in the **union** $A \cup B$ of two sets $A$ and $B$ if it is in $A$ or $B$.

(ii) An element is in the **intersection** $A \cap B$ of two sets $A$ and $B$ if it is in $A$ and $B$.

(iii) An element is in the **set difference** $A \setminus B$ if it is in $A$ but not $B$.

(iv) Given a set $\Omega$ and a set $A \subset \Omega$, the **complement** of $A$ with respect to $\Omega$ is $A^c = \Omega \setminus A$.




**5** Two sets $A$ and $B$ are **disjoint** if $A \cap B = \emptyset$ (in other words, if they have no elements in common).

**6** A **partition** of a set is a collection of nonempty disjoint subsets whose union is the whole set.

**7** The **Cartesian product** of $A$ and $B$ is

$$A \times B = \{(a, b) : a \in A \text{ and } b \in B\}.$$

**8** **(De Morgan's laws)** If $A, B \subset \Omega$, then

(i) $(A \cap B)^c = A^c \cup B^c$, and

(ii) $(A \cup B)^c = A^c \cap B^c$.

**9** A **list** is an ordered collection of finitely many objects. Lists differ from sets in that (i) order matters, (ii) repetition matters, and (iii) the cardinality is restricted.

## Sets and Functions: Functions

**1** If $A$ and $B$ are sets, then a **function** $f : A \to B$ is an assignment of some element of $B$ to each element of $A$.

**2** The set $A$ is called the **domain** of $f$ and $B$ is called the **codomain** of $f$.

**3** Given a subset $A'$ of $A$, we define the **image** of $f$—denoted $f(A')$—to be the set of elements which are mapped to from some element in $A'$.

**4** The **range** of $f$ is the image of the domain of $f$.

**5** The **composition** of two functions $f : A \to B$ and $g : B \to C$ is the function $g \circ f$ which maps $a \in A$ to $g(f(a)) \in C$.

**6** The **identity function** on a set $A$ is the function $f : A \to A$ which maps each element to itself.

**7** A function $f$ is **injective** if no two elements in the domain map to the same element in the codomain.

**8** A function $f$ is **surjective** if the range of $f$ is equal to the codomain of $f$.

**9** A function $f$ is **bijective** if it is both injective and surjective. If $f$ is bijective, then the function from $B$ to $A$ that maps $b \in B$ to the element $a \in A$ that satisfies $f(a) = b$ is called the **inverse** of $f$.

**10** If $f : A \to B$ is bijective, then the function $f^{-1} \circ f$ is equal to the identity function on $A$, and $f \circ f^{-1}$ is the identity function on $B$.

## Programming in Julia

**1** A **value** is a fundamental entity that may be manipulated by a program. Values have **types**; for example, `5` is an `Int` and `"Hello world!"` is a `String`.

**2** A **variable** is a name used to refer to a value. We can **assign** a value `5` to a variable `x` using `x = 5`.

**3** A **function** performs a particular task. You prompt a function to perform its task by **calling** it. Values supplied to a function are called **arguments**. For example, in the function call `print(1,2)`, `1` and `2` are arguments.

**4** An **operator** is a function that can be called in a special way. For example, `*` is an operator since we can call the multiplication function with the syntax `3 * 5`.

**5** A **statement** is an instruction to be executed (like `x = -3`). An **expression** is a combination of values, variables, operators, and function calls that a language interprets and **evaluates** to a value.

**6** A numerical value can be either an **integer** or a **float**. The basic operations are `+,-,*,/,^`, and expressions are evaluated according to the order of operations.

**7** Numbers can be compared using `<,>,==,≤` or `≥`.

**8** Textual data is represented using **strings**. `length(s)` returns the number of characters in `s`. The `*` operator concatenates strings.

**9** A **boolean** is a value which is either `true` or `false`. Booleans can be combined with the operators `&&` (and), `||` (or), `!` (not).

**10** Code blocks can be executed conditionally:

```
if x > 0
    "x is positive"
elseif x == 0
    "x is zero"
else
    "x is negative"
end
```

**11** Functions may be defined using the familiar math notation: `f(x,y) = 3x + 2y` or using a `function` block (`shift` is a **keyword argument**):

```
function f(x,y; shift=0)
    3x + 2y + shift
end
```

**12** The **scope** of a variable is the region in the program where it is accessible. Variables defined in the body of a function are not accessible outside the body of the function.

**13** `Array` is a compound data type for storing lists of objects. Entries of an array may be accessed with square bracket syntax using an index or using a **range** object `a:b`.

```
A = [-5,3,2,1]
A[2]
A[3:end]
```

**14** An **array comprehension** can be used to generate new arrays: `[k^2 for k=1:10 if mod(k,2) == 0]`

**15** A **dictionary** encodes a discrete function by storing input-output pairs and looking up input values when indexed. This expression returns `[0,0,1.0]`:

```
Dict("blue"=>[0,0,1.0],"red"=>[1.0,0,0])["blue"]
```

**16** A `while` loop takes a conditional expression and a body and evaluates them alternatingly until the conditional expression returns false. A `for` loop evaluates its body once for each entry in a given *iterator* (for example, a range, array, or dictionary). Each value in the iterator is assigned to a loop variable which can be referenced in the body of the loop.

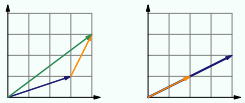
```
while x > 0
    x -= 1
end
```

```
for i=1:10
    print(i)
end
```

## Linear Algebra: Vector Spaces

**1** A **vector** in $\mathbb{R}^n$ is a column of $n$ real numbers, also written as $[v_1, \ldots, v_n]$. A vector may be depicted as an arrow from the origin in $n$-dimensional space. The **norm** of a vector $\mathbf{v}$ is the length $\sqrt{v_1^2 + \cdots + v_n^2}$ of its arrow.

**2** The fundamental vector space operations are **vector addition** and **scalar multiplication**.



**3** A **linear combination** of a list of vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ is an expression of the form

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k,$$

where $c_1, \ldots, c_k$ are real numbers. The $c$'s are called the **weights** of the linear combination.

**4** The **span** of a list $L$ of vectors is the set of all vectors which can be written as a linear combination of the vectors in $L$.

**5** A list of vectors is **linearly independent** if and only if the only linear combination which yields the zero vector is the one with all weights zero.

**6** A **vector space** is a nonempty set of vectors which is closed under the vector space operations.

**7** A list of vectors in a vector space is a **spanning list** of that vector space if every vector in the vector space can be written as a linear combination of the vectors in that list.

**8** A linearly independent spanning list of a vector space is called a **basis** of that vector space. The number of vectors in a basis of a vector space is called the **dimension** of the space.

**9** A **linear transformation** $L$ is a function from a vector space $V$ to a vector space $W$ which satisfies $L(c\mathbf{v} + \beta\mathbf{w}) = cL(\mathbf{v}) + L(\mathbf{w})$ for all $c \in \mathbb{R}$, $\mathbf{u}, \mathbf{v} \in V$. These are "flat maps": equally spaced lines are mapped to equally spaces lines or points. Examples: scaling, rotation, projection, reflection.

**10** Given two vector spaces $V$ and $W$, a basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ of $V$, and a list $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ of vectors in $W$, there exists one and only one linear transformation which maps $\mathbf{v}_1$ to $\mathbf{w}_1$, $\mathbf{v}_2$ to $\mathbf{w}_2$, and so on.

**11** The **rank** of a linear transformation from one vector space to another is the dimension of its range.

**12** The **null space** of a linear transformation is the set of vectors which are mapped to the zero vector by the linear transformation.

**13** The rank of a transformation plus the dimension of its null space is equal to the dimension of its domain (the **rank-nullity theorem**).

## Linear Algebra: Matrix Algebra

**1** The **matrix-vector product** $A\mathbf{x}$ is the linear combination of the columns of $A$ with weights given by the entries of $\mathbf{x}$.

**2** Linear transformations from $\mathbb{R}^n$ to $\mathbb{R}^m$ are in one-to-one correspondence with $m \times n$ matrices.

**3** The identity transformation corresponds to the **identity matrix**, which has entries of 1 along the diagonal and zero entries elsewhere.

**4** **Matrix multiplication** corresponds to composition of the corresponding linear transformations: $AB$ is the matrix for which $(AB)(\mathbf{x}) = A(B\mathbf{x})$ for all $\mathbf{x}$.

**5** A $m \times n$ matrix is **full rank** if its rank is equal to $\min(m, n)$

**6** $A\mathbf{x} = \mathbf{b}$ has a solution $\mathbf{x}$ if and only if $\mathbf{b}$ is in the span of the columns of $A$. If $A\mathbf{x} = \mathbf{b}$ does have a solution, then the solution is unique if and only if the columns of $A$ are linearly independent. If $A\mathbf{x} = \mathbf{b}$ does not have a solution, then there is a unique vector $\mathbf{x}$ which minimizes $|A\mathbf{x} - \mathbf{b}|^2$.

**7** If the columns of a square matrix $A$ are linearly independent, then it has a unique **inverse matrix** $A^{-1}$ with the property that $A\mathbf{x} = \mathbf{b}$ implies $\mathbf{x} = A^{-1}\mathbf{b}$ for all $\mathbf{x}$ and $\mathbf{b}$.

**8** Matrix inversion satisfies $(AB)^{-1} = B^{-1}A^{-1}$ if $A$ and $B$ are both invertible.

**9** The **transpose** $A'$ of a matrix $A$ is defined so that the rows of $A'$ are the columns of $A$ (and vice versa).

**10** The transpose is a linear operator: $(cA + B)' = cA' + B'$ if $c$ is a constant and $A$ and $B$ are matrices.

**11** The transpose distributes across matrix multiplication but with an order reversal: $(AB)' = B'A'$ if $A$ and $B$ are matrices for which $AB$ is defined.

**12** A matrix $A$ is **symmetric** if $A = A'$.

**13** A linear transformation $T$ from $\mathbb{R}^n$ to $\mathbb{R}^n$ scales all $n$-dimensional volumes by the same factor: the (absolute value of the) **determinant** of $T$.

**14** The sign of the determinant tells us whether $T$ reverses orientations.

**15** $\det AB = \det A \det B$ and $\det A^{-1} = (\det A)^{-1}$.

**16** A square matrix is invertible if and only if its determinant is nonzero.

## Linear Algebra: Orthogonality

**1** The **dot product** of two vectors in $\mathbb{R}^n$ is defined by

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n.$$

**2** $\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\|\|\mathbf{y}\| \cos\theta$, where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and $\theta$ is the angle between the vectors.

**3** $\mathbf{x} \cdot \mathbf{y} = 0$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are orthogonal.

**4** The dot product is linear: $\mathbf{x} \cdot (c\mathbf{y} + \mathbf{z}) = c\mathbf{x} \cdot \mathbf{y} + \mathbf{x} \cdot \mathbf{z}$.

**5** The **orthogonal complement** of a subspace $V \subset \mathbb{R}^n$ is the set of vectors which are orthogonal to every vector in $V$.

**6** The orthogonal complement of the span of the columns of a matrix $A$ is equal to the null space of $A'$.

**7** rank $A$ = rank $A'A$ for any matrix $A$.

**8** A list of vectors satisfying $\mathbf{v}_i \cdot \mathbf{v}_j = 0$ for $i \neq j$ is **orthogonal**. An orthogonal list of unit vectors is **orthonormal**.

**9** Every orthogonal list is linearly independent

**10** A matrix $U$ has orthonormal columns if and only if $U'U = I$. A square matrix with orthonormal columns is called **orthogonal**. An orthogonal matrix and its transpose are inverses.

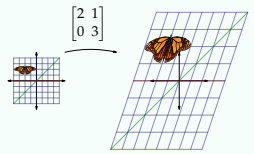
**11** Orthogonal matrices represent **rigid transformations** (ones which preserve lengths and angles).

**12** If $U$ has orthonormal columns, then $UU'$ is the matrix which represents projection onto the span of the columns of $U$.

## Linear Algebra: Spectral Analysis

**1** An **eigenvector** $\mathbf{v}$ of an $n \times n$ matrix $A$ is a nonzero vector with the property that $A\mathbf{v} = \lambda\mathbf{v}$ for some $\lambda \in \mathbb{R}$. We call $\lambda$ an **eigenvalue**.

If $\mathbf{v}$ is an eigenvector of $A$, then $A$ maps the line span($\{\mathbf{v}\}$) to itself:




**2** Eigenvectors of $A$ with distinct eigenvalues are linearly independent.

**3** Not every $n \times n$ matrix $A$ has $n$ linearly independent eigenvectors. If $A$ does have $n$ linearly independent eigenvectors, we can make a matrix $V$ with these eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_n$ as columns and get

$$AV = V\Lambda \implies A = V\Lambda V^{-1} \quad \textbf{(diagonalization of } A\textbf{)}$$

where $\Lambda$ is a diagonal matrix of eigenvalues. The action of $A$ is to scale components in the basis $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$:

$$A(c_1\mathbf{v}_1 + \cdots + c_n\mathbf{v}_n) = c_1\lambda_1\mathbf{v}_1 + \cdots + c_n\lambda_n\mathbf{v}_n$$

**4** Diagonalization allows us to apply functions to matrices by applying them to the eigenvalues: if $A = V\Lambda V^{-1}$, then $A^n = V\Lambda^n V^{-1}$, and $\sqrt{A} = V\sqrt{\Lambda}V^{-1}$.

**5** **Spectral Theorem**: if $A$ is an $n \times n$ symmetric matrix, then $A$ is orthogonally diagonalizable: there is an orthogonal matrix $V$ with columns $\mathbf{v}_1, \dots, \mathbf{v}_k$ such that

$$A = V\Lambda V' = \lambda_1\mathbf{v}_1\mathbf{v}_1' + \cdots + \lambda_n\mathbf{v}_n\mathbf{v}_n',$$

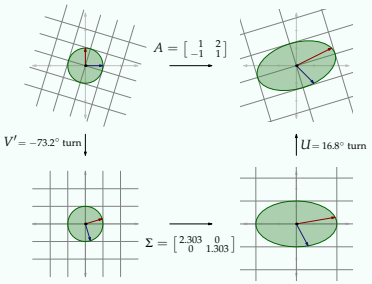
Note that $\mathbf{v}_k\mathbf{v}_k'$ projects onto the span of $\{\mathbf{v}_k\}$.

**6** A symmetric matrix is **positive semidefinite** if its eigenvalues are all nonnegative.

## Linear Algebra: SVD

**1** The **Gram matrix** $A'A$ of any $m \times n$ matrix $A$ is positive semidefinite. Furthermore, $|\sqrt{A'A}\mathbf{x}| = |A\mathbf{x}|$ for all $x \in \mathbb{R}^n$.

**2** The **singular value decomposition** is the factorization of any rectangular $m \times n$ matrix $A$ as $U\Sigma V'$, where $U$ and $V$ are orthogonal and $\Sigma$ is an $m \times n$ diagonal matrix (with diagonal entries in decreasing order).




**3** The diagonal entries of $\Sigma$ are the **singular values of** $A$, and the columns of $U$ and $V$ are called **left singular vectors** and **right singular vectors**, respectively. $A$ maps each right singular vector $\mathbf{v}_i$ to the corresponding left singular vector $\mathbf{u}_i$ scaled by $\sigma_i$.

**4** The vectors in $\mathbb{R}^n$ stretched the most by $A$ are the ones which run in the direction of the column or columns of $V$ corresponding to the greatest singular value. Same for least.

**5** For $k \geq 1$, the $k$-dimensional vector space with minimal sum of squared distances to the columns of $A$ (interpreted as points in $\mathbb{R}^m$) is the span of the first $k$ columns of $U$.

**6** The absolute value of the determinant of a square matrix is equal to the product of its singular values.

## Multivariable calculus

**1** A **sequence** of real numbers $(x_n)_{n=1}^{\infty} = x_1, x_2, \dots$ **converges** to a number $x \in \mathbb{R}$ if the distance from $x_n$ to $x$ on the number line can be made as small as desired by choosing $n$ sufficiently large. We say $\lim_{n\to\infty} x_n = x$ or $x_n \to x$.

**2** **(Squeeze theorem)** If $a_n \leq b_n \leq c_n$ for all $n \geq 1$ and if $\lim_{n\to\infty} a_n = \lim_{n\to\infty} c_n = b$, then $b_n \to b$ as $n \to \infty$.

**3** **(Comparison test)** If $\sum_{n=1}^{\infty} b_n$ converges and if $|a_n| \leq b_n$ for all $n$, then $\sum_{n=1}^{\infty} a_n$ converges.

Conversely, if $\sum_{n=1}^{\infty} b_n$ does not converge and $0 \leq b_n < a_n$, then $\sum_{n=1}^{\infty} a_n$ also does not converge.

**4** The series $\sum_{n=1}^{\infty} n^p$ converges if and only if $p < -1$. The series $\sum_{n=1}^{\infty} a^n$ converges if and only if $-1 < a < 1$.

**5** The **Taylor series**, centered at $c$, of an infinitely differentiable function $f$ is defined to be

$$f(c) + f'(c)(x-c) + \frac{f''(c)}{2!}(x-c)^2 + \frac{f'''(c)}{3!}(x-c)^3 + \cdots$$

**6** We can multiply or add Taylor series term-by-term, we can integrate or differentiate a Taylor series term-by-term, we can substitute one Taylor series into another to obtain a Taylor series for the composition.

**7** The **partial derivative** $\frac{\partial f}{\partial x}(x_0, y_0)$ of a function $f(x, y)$ at a point $(x_0, y_0)$ is the slope of the graph of $f$ in the $x$-direction at the point $(x_0, y_0)$.

**8** Given $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, we define $\partial\mathbf{f}/\partial\mathbf{x}$ to be the matrix whose $(i, j)$th entry is $\partial f_i/\partial x_j$. Then

$$\text{(i) } \frac{\partial}{\partial \mathbf{x}}(A\mathbf{x}) = A \qquad \text{(ii) } \frac{\partial}{\partial \mathbf{x}}(\mathbf{x}'A) = A'$$

$$\text{(iii) } \frac{\partial}{\partial \mathbf{x}}(\mathbf{u}'\mathbf{v}) = \mathbf{u}'\frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}'\frac{\partial \mathbf{u}}{\partial \mathbf{x}}.$$

**9** A function of two variables is **differentiable** at a point if its graph looks like a plane when you zoom in sufficiently around the point. More generally, a function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ is differentiable at $\mathbf{x}$ if it is well-approximated by its derivative near $\mathbf{x}$:

$$\lim_{\Delta\mathbf{x}\to\mathbf{0}} \frac{\mathbf{f}(\mathbf{x}+\Delta\mathbf{x}) - \left(\mathbf{f}(\mathbf{x}) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x})\Delta\mathbf{x}\right)}{|\Delta\mathbf{x}|} = 0.$$

**10** The **Hessian** $\mathcal{H}$ of $f : \mathbb{R}^n \to \mathbb{R}$ is the matrix of its second order derivatives: $\mathcal{H}_{i,j}(\mathbf{x}) = \frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j} f(\mathbf{x})$. The **quadratic approximation** of $f$ at the origin is $f(\mathbf{0}) + \frac{\partial f}{\partial \mathbf{x}}(\mathbf{0})\mathbf{x} + \frac{1}{2}\mathbf{x}'\mathcal{H}(\mathbf{0})\mathbf{x}$.

**11** Suppose that $f$ is a continuous function defined on a closed and bounded subset $D$ of $\mathbb{R}^n$. Then:

(i) $f$ realizes an absolute maximum and absolute minimum on $D$ (the **extreme value theorem**).

(ii) Any point where $f$ realizes an extremum is either a critical point—meaning that $\nabla f = 0$ or $f$ is nondifferentiable at that point—or at a point on the boundary.

(iii) **(Lagrange multipliers)** If $f$ realizes an extremum at a point on a portion of the boundary which is the level set of a differentiable function $g$ with non-vanishing gradient $\nabla g$, then either $f$ is non-differentiable at that point or the equation

$$\nabla f = \lambda \nabla g$$

is satisfied at that point, for some $\lambda \in \mathbb{R}$.

**12** If $\mathbf{r} : \mathbb{R}^1 \to \mathbb{R}^2$ and $f : \mathbb{R}^2 \to \mathbb{R}^1$, then

$$\frac{\mathrm{d}}{\mathrm{d}t}(f \circ \mathbf{r}) = \frac{\partial f}{\partial \mathbf{r}}(\mathbf{r}(t))\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t}(t). \quad \textbf{(chain rule)}$$

**13** Integrating a function is a way of totaling up its values. $\iint_D f(x, y)\,\mathrm{d}x\,\mathrm{d}y$ can be interpreted as the mass of an object occupying the region $D$ and having mass density $f(x, y)$ at each point $(x, y)$.

**14** Double integration over $D$: the bounds for the outer integral are the smallest and largest values of $y$ for any point in $D$, and the bounds for the inner integral are the smallest and largest values of $x$ for any point in a given "$y$ = constant" slice of the region.

**15** Polar integration over $D$: the outer integral bounds are the least and greatest values of $\theta$ for a point in $D$, and the inner integral bounds are the least and greatest values of $r$ for any point in $D$ along each given "$\theta$ = constant" ray. The area element is $\mathrm{d}A = r\,\mathrm{d}r\,\mathrm{d}\theta$.

## Numerical Computation: machine arithmetic

**1** Computers store numerical values as sequences of bits. The **type** of a numeric value specifies how to interpret the underlying sequence of bits as a number.

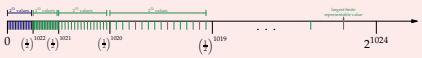
**2** The `Int64` type uses 64 bits to represent the integers from $-2^{63}$ to $2^{63} - 1$. For $0 \leq n \leq 2^{63} - 1$, we represent $n$ using its binary representation, and for $1 \leq n \leq 2^{63}$, we represent $-n$ using the binary representation of $2^{64} - n$. `Int64` arithmetic is performed modulo $2^{64}$.

**3** The `Float64` type uses 64 bits to represent real numbers. We call the first bit $\sigma$, the next 11 bits (interpreted as a binary integer) $e \in [0, 2047]$, and the final 52 bits $f \in [0, 2^{52} - 1]$. If $e \notin \{0, 2047\}$, then the number represented by $(\sigma, e, f)$ is

$$x = (-1)^{\sigma} 2^{e-1023}\left(1 + f\left(\frac{1}{2}\right)^{52}\right).$$

The representable numbers between consecutive powers of 2 are the ones obtained by 52 recursive iterations of binary subdivision. The value of $e$ indicates the powers of 2 that $x$ is between, and the value of $f$ indicates the position of $x$ between those powers of 2.

The `Float64` exponent value $e = 2047$ is reserved for `Inf` and `NaN`, while $e = 0$ is reserved for the **subnormal numbers**: $(\sigma, 0, f)$ represents $(-1)^{\sigma} f/2^{1074}$.




**4** The `BigInt` and `BigFloat` are types use an arbitrary number of bits and can handle very large numbers or very high precision. Computations are much slower than for 64-bit types (like 10 ms for `sum(1:10^6)` versus 0.5 ms).

## Numerical Computation: Error

**1** If $\hat{A}$ is an approximation for $A$, then the **relative error** is $\frac{\hat{A}-A}{A}$

**2** **Roundoff error** comes from rounding numbers to fit them into a floating point representation.

**3** **Truncation error** comes from using approximate mathematical formulas or algorithms.

**4** **Statistical error** arises from using randomness in an approximation.

**5** The **condition number** of a function measures how it stretches or compresses *relative* error. The condition number of a problem is the condition number of the map from the problem's initial data $a$ to its solution $\mathbf{S}(a)$:

$$\kappa(a) = \frac{|a|\left|\frac{\mathrm{d}}{\mathrm{d}a}\mathbf{S}(a)\right|}{|\mathbf{S}(a)|}.$$

**6** A problem is **well-conditioned** if its condition number is modest and **ill-conditioned** if the condition number is large.

**7** The condition number of $a \mapsto a^n$ is $\kappa(a) = n$. The condition number of $a \mapsto a - b$ is $\frac{a}{a-b}$, so subtracting $b$ is ill-conditioned near $b$ (**catastrophic cancellation**).

**8** The relative roundoff error between a non-extreme real number and the nearest `T`-representable value is no more than the **machine epsilon** of the floating point type `T`.

**9** An algorithm which solves a problem with error much greater than $\kappa\epsilon_{\text{mach}}$ is **unstable**. An algorithm is unstable if at least one of the steps it performs is ill-conditioned. If every step of an algorithm is well-conditioned, then the algorithm is **stable**.

**10** The condition number of an matrix $A$ is defined to be the maximum condition number of the function $\mathbf{x} \mapsto A\mathbf{x}$ over its domain: $\kappa(A) = \sigma_{\max}/\sigma_{\min}$.

## Numerical Computation: PRNGs

**1** A **pseudorandom number generator** (PRNG) is an algorithm for generating a deterministic sequence of numbers which is intended to share properties with a sequence of random numbers. The PRNG's initial value is called its **seed**.

**2** The **linear congruential generator**: fix positive integers $M$, $a$, and $c$, and consider a seed $X_0 \in \{0, 1, \dots, M-1\}$. We return the sequence $X_0, X_1, X_2, \dots$, where $X_n = \text{mod}(aX_{n-1} + c, M)$ for $n \geq 1$.

**3** The **period** of a PRNG is the minimum length of a repeating block. A long period is a desirable property of a PRNG, and a very short period is typically unacceptable.

**4** **Frequency tests** check whether blocks of terms appear with the appropriate frequency (for example, we can check whether $a_{2n} > a_{2n-1}$ for roughly half of the values of $n$).

## Numerical Computation: Automatic Differentiation

**1** A **dual number** stores the numerical values of the first *two* terms of a function's Taylor series. Differentiation by substituting dual numbers (autodiff, or AD) provides sped and machine-precision accuracy.

**2** To find the derivative of `f` with AD, every function or operation used by `f` must be dual-number-aware:

```
struct DualNumber
    v # value
    d # derivative
end
*(x::DualNumber, y::DualNumber) = # product rule
    DualNumber(x.v * y.v, x.v * y.d + x.d * y.v)
sin(x::DualNumber) = # chain rule for sine
    DualNumber(sin(x.v), cos(x.v) * x.d)
```

**3** Use packages: `ForwardDiff` for Julia; `autograd` for Python.

## Numerical Computation: Optimization

**1** **Gradient descent** seeks to minimize $f : \mathbb{R}^n \to \mathbb{R}$ by repeatedly stepping in $f$'s direction of maximum decrease. We begin with a value $\mathbf{x}_0 \in \mathbb{R}^n$ and repeatedly update using the rule $\mathbf{x}_{n+1} = \mathbf{x}_n - \epsilon\nabla f(\mathbf{x}_n)$, where $\epsilon$ is the **learning rate**. Large $\epsilon$ can lead to divergence; small $\epsilon$ to slow convergence. We fix a small number $\tau > 0$ and stop when $|\nabla f(\mathbf{x}_n)| < \tau$.

**2** A set is **convex** if it contains every line segment connecting any two points in the set. A function defined on a convex subset of $\mathbb{R}^n$ is convex if every line segment connecting two points on its graph lies on or above the graph (which is implied by an everywhere positive semidefinite Hessian). Convex function have desirable optimization properties: any local minimum of a convex function is also a global minimum.

**3** Algorithms similar to gradient descent but with usually faster convergence: **conjugate gradient, BFGS, L-BFGS**.

## Probability: Counting

**1** **Fundamental principle of counting:** If one experiment has $m$ possible outcomes, and if a second experiment has $n$ possible outcomes for each of the outcomes in the first experiment, then there are $mn$ possible outcomes for the pair of experiments.

**2** The number of ways to arrange $n$ objects in order is $n! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot n$ (read $n$ **factorial**).

**3** **Permutations**: if $S$ is a set with $n$ elements, then there are $\frac{n!}{(n-r)!}$ *ordered* $r$-tuples of distinct elements of $S$.

**4** **Combinations**: The number of $r$-element subsets of an $n$-element set is $\binom{n}{r} = \frac{n!}{r!(n-r)!}$.

## Probability: Probability Spaces

**1** Given a random experiment, the set of possible outcomes is called the **sample space** $\Omega$, like $\{(\mathtt{H},\mathtt{H}), (\mathtt{H},\mathtt{T}), (\mathtt{T},\mathtt{H}), (\mathtt{T},\mathtt{T})\}$.

**2** We associate with each outcome $\omega \in \Omega$ a **probability mass**, denoted $m(\omega)$. For example, $m((\mathtt{H},\mathtt{T})) = \frac{1}{4}$.

**3** In a random experiment, an **event** is a predicate that can be determined based on the outcome of the experiment (like "first flip turned up heads"). Mathematically, an event is a subset of $\Omega$ (like $\{(\mathtt{H},\mathtt{H}), (\mathtt{H},\mathtt{T})\}$).

**4** Basic set operations $\cup$, $\cap$, and $^c$ correspond to disjunction, conjunction, and negation of events:

(i) The event that $E$ happens or $F$ happens is $E \cup F$.
(ii) The event that $E$ happens and $F$ happens is $E \cap F$.
(iii) The event that $E$ does not happen is $E^c$.

**5** If $E$ and $F$ cannot both occur (that is, $E \cap F = \emptyset$), we say that $E$ and $F$ are **mutually exclusive** or **disjoint**.

**6** If $E$'s occurrence implies $F$'s occurrence, then $E \subset F$.

**7** The probability $\mathbb{P}(E)$ of an event $E$ is the sum of the probability masses of the outcomes in that event. The domain of $\mathbb{P}$ is $2^\Omega$, the set of all subsets of $\Omega$.

**8** The pair $(\Omega, \mathbb{P})$ is called a probability space. The fundamental probability space properties are

(i) $\mathbb{P}(\Omega) = 1$ — "something has to happen"
(ii) $\mathbb{P}(E) \geq 0$ — "probabilities are non-negative"
(iii) $\mathbb{P}(E \cup F) = \mathbb{P}(E) + \mathbb{P}(F)$ if $E$ and $F$ are mutually exclusive — "probability is additive".

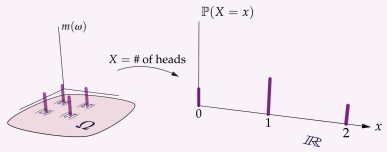
**9** Other properties which follow from the fundamental ones:

(i) $\mathbb{P}(\emptyset) = 0$
(ii) $\mathbb{P}(E^c) = 1 - \mathbb{P}(E)$
(iii) $E \subset F \implies \mathbb{P}(E) \leq \mathbb{P}(F)$ (monotonicity)
(iv) $\mathbb{P}(E \cup F) = \mathbb{P}(E) + \mathbb{P}(F) - \mathbb{P}(E \cap F)$ (principle of inclusion-exclusion).

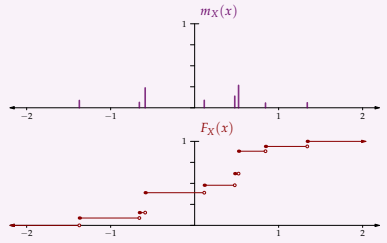
## Probability: Random Variables

**1** A **random variable** is a number which depends on the result of a random experiment (one's lottery winnings, for example). Mathematically, a random variable is a function $X$ from the sample space $\Omega$ to $\mathbb{R}$.

**2** The **distribution** of a random variable $X$ is the probability measure on $\mathbb{R}$ which maps each set $A \subset \mathbb{R}$ to $\mathbb{P}(X \in A)$. The probability mass function of the distribution of $X$ may be obtained by pushing forward the probability mass from each $\omega \in \Omega$:




**3** The **cumulative distribution function** (CDF) of a random variable $X$ is the function $F_X(x) = \mathbb{P}(X \leq x)$.




**4** The **joint distribution** of two random variables $X$ and $Y$ is the probability measure on $\mathbb{R}^2$ which maps $A \subset \mathbb{R}^2$ to $\mathbb{P}((X,Y) \in A)$. The probability mass function of the joint distribution is $m_{(X,Y)}(x,y) = \mathbb{P}(X = x \text{ and } Y = y)$.

## Probability: Conditional Probability

**1** Given a probability space $\Omega$ and an event $E \subset \Omega$, the **conditional probability measure** given $E$ is an updated probability measure on $\Omega$ which accounts for the information that the result $\omega$ of the random experiment falls in $E$:

$$\mathbb{P}(F \mid E) = \frac{\mathbb{P}(F \cap E)}{\mathbb{P}(E)}$$

**2** The conditional probability mass function of $Y$ given $\{X = x\}$ is $m_{Y \mid X=x}(y) = m_{X,Y}(x,y)/m_X(x)$.

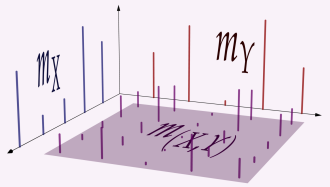
**3** **Bayes' theorem** tells us how to update beliefs in light of new evidence. It relates the conditional probabilities $\mathbb{P}(A \mid E)$ and $\mathbb{P}(E \mid A)$:

$$\mathbb{P}(A \mid E) = \frac{\mathbb{P}(E \mid A)\mathbb{P}(A)}{\mathbb{P}(E)} = \frac{\mathbb{P}(E \mid A)\mathbb{P}(A)}{\mathbb{P}(E \mid A)\mathbb{P}(A) + \mathbb{P}(E \mid A^c)\mathbb{P}(A^c)}.$$

**4** Two events $E$ and $F$ are **independent** if $\mathbb{P}(E \cap F) = \mathbb{P}(E)\mathbb{P}(F)$.

**5** Two random variables $X$ and $Y$ are **independent** if the every pair of events of the form $\{X \in A\}$ and $\{Y \in B\}$ are independent, where $A \subset \mathbb{R}$ and $B \subset \mathbb{R}$.

**6** The PMF of the joint distribution of a pair of independent random variables factors as $m_{X,Y}(x,y) = m_X(x)m_Y(y)$:




## Probability: Expectation and Variance

**1** The **expectation** $\mathbb{E}[X]$ (or **mean** $\mu_X$) of a random variable $X$ is the *probability-weighted average of* $X$:

$$\mathbb{E}[X] = \sum_{\omega \in \Omega} X(\omega)m(\omega)$$

**2** The expectation $\mathbb{E}[X]$ may be thought of as the value of a random game with payout $X$, or as the long-run average of $X$ over many independent runs of the underlying experiment. The **Monte Carlo** approximation of $\mathbb{E}[X]$ is obtained by simulating the experiment many times and averaging the value of $X$.

**3** The expectation is the center of mass of the distribution of $X$:



**4** The expectation of a function of a discrete random variable (or two random variables) may be expressed in terms of the PMF $m_X$ of the distribution of $X$ (or the PMF $m_{(X,Y)}$ of the joint distribution of $X$ and $Y$):

$$\mathbb{E}[g(X)] = \sum_{x \in \mathbb{R}} g(x)m_X(x)$$
$$\mathbb{E}[g(X,Y)] = \sum_{(x,y) \in \mathbb{R}^2} g(x,y)m_{(X,Y)}(x,y).$$

**5** Expectation is **linear**: if $c \in \mathbb{R}$ and $X$ and $Y$ are random variables defined on the same probability space, then

$$\mathbb{E}[cX + Y] = c\,\mathbb{E}[X] + \mathbb{E}[Y]$$

**6** The **variance** of a random variable is its average squared deviation from its mean. The variance measures how spread out the distribution of $X$ is. The **standard deviation** $\sigma(X)$ is the square root of the variance.

**7** Variance satisfies the properties, if $X$ and $Y$ are independent random variables and $a \in \mathbb{R}$:

$$\operatorname{Var}(aX) = a^2 \operatorname{Var} X$$
$$\operatorname{Var}(X + Y) = \operatorname{Var}(X) + \operatorname{Var}(Y)$$

**8** The **covariance** of two random variables $X$ and $Y$ is the expected product of their deviations from their respective means $\mu_X = \mathbb{E}[X]$ and $\mu_Y = \mathbb{E}[Y]$:

$$\operatorname{Cov}(X,Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y].$$

**9** The covariance of two independent random variables is zero, but zero covariance does not imply independence.

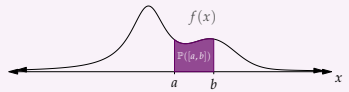
**10** The **correlation** of two random variables is their normalized covariance:

$$\operatorname{Corr}(X,Y) = \frac{\operatorname{Cov}(X,Y)}{\sigma(X)\sigma(Y)} \in [-1,1]$$

**11** The **covariance matrix** of a vector $\mathbf{X} = [X_1, \ldots, X_n]$ of random variables defined on the same probability space is defined to be the matrix $\Sigma$ whose $(i,j)$th entry is equal to $\operatorname{Cov}(X_i, X_j)$. If $\mathbb{E}[\mathbf{X}] = \mathbf{0}$, then $\Sigma = \mathbb{E}[\mathbf{X}\mathbf{X}']$.

## Probability: Continuous Distributions

**1** If $\Omega \subset \mathbb{R}^n$ and $\mathbb{P}(A) = \int_A f$, where $f \geq 0$ and $\int_{\mathbb{R}^n} f = 1$, then we call $(\Omega, \mathbb{P})$ a **continuous probability space**.




**2** The function $f$ is called a **density**, because it measures the amount of probability mass per unit volume at each point (2D volume = area, 1D volume = length).

**3** If $(X,Y)$ is a pair of random variables whose joint distribution has density $f_{X,Y} : \mathbb{R}^2 \to \mathbb{R}$, then the conditional distribution of $Y$ given the event $\{X = x\}$ has density $f_{Y \mid X=x}$ defined by

$$f_{Y \mid \{X=x\}}(y) = \frac{f_{X,Y}(x,y)}{f_X(x)},$$

where $f_X(x) = \int_{-\infty}^{\infty} f(x,y)\,\mathrm{d}y$ is the PDF of $X$.

**4** If a random variable $X$ has density $f_X$ on $\mathbb{R}$, then

$$\mathbb{E}[g(X)] = \int_{\mathbb{R}} g(x)f_X(x)\,\mathrm{d}x.$$

**5** **CDF sampling**: $F^{-1}(U)$ has CDF $F$ if $f_U = \mathbf{1}_{[0,1]}$.

## Probability: Conditional Expectation

**1** The **conditional expectation** of a random variable given an event is the expectation of the random variable calculated with respect to the conditional probability measure given that event: if $(X,Y)$ has PMF $m_{X,Y}$, then

$$\mathbb{E}[Y \mid X = x] = \sum_{y \in \mathbb{R}} y m_{Y \mid X=x}(y),$$

where $m_{Y \mid X=x}(y) = \frac{m_{X,Y}(x,y)}{m_X(x)}$. If $(X,Y)$ has pdf $f_{X,Y}$, then

$$\mathbb{E}[Y \mid X = x] = \int_{\mathbb{R}} y f_{Y \mid X=x}(y)\,\mathrm{d}y.$$

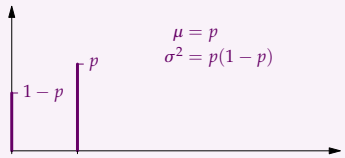
**2** The conditional expectation of a random variable $Y$ given another random variable $X$ is obtained by substituting $X$ for $x$ in the expression for the conditional expectation of $Y$ given $X = x$. Thus $\mathbb{E}[Y \mid X]$ is a random variable.

**3** If $X$ and $Y$ are independent, then $\mathbb{E}[Y \mid X] = \mathbb{E}[Y]$. If $Z$ is a function of $X$, then $\mathbb{E}[ZY \mid X] = Z\mathbb{E}[Y \mid X]$.

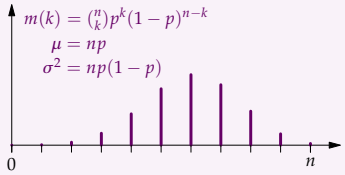
**4** The **law of iterated expectation**: $\mathbb{E}[\mathbb{E}[Y \mid X]] = \mathbb{E}[Y]$.

## Probability: Common Distributions

**1** **Bernoulli** (Ber($p$)): A weighted coin flip.

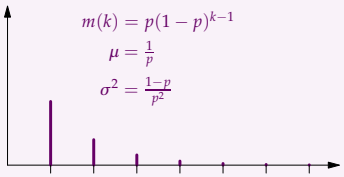



**2** **Binomial** (Bin($n,p$)): A sum of $n$ independent Ber($p$)'s.



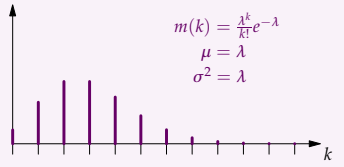
**3** **Geometric** (Geom($p$)): Time to first success (1) in a sequence of independent Ber($p$)'s.

$$m(k) = p(1-p)^{k-1}$$
$$\mu = \tfrac{1}{p}$$
$$\sigma^2 = \tfrac{1-p}{p^2}$$



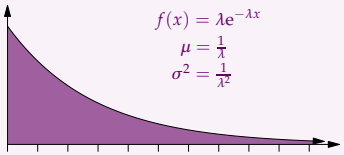
**4** **Poisson distribution** (Poiss($\lambda$)): Limit as $n \to \infty$ of Binomial($n, \frac{\lambda}{n}$).

$$m(k) = \tfrac{\lambda^k}{k!} e^{-\lambda}$$
$$\mu = \lambda$$
$$\sigma^2 = \lambda$$



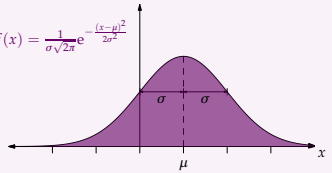

**5** **Exponential distribution** (Exp($\lambda$)): Limit as $n \to \infty$ of distribution of $1/n$ times a Geometric($\lambda/n$).

$$f(x) = \lambda e^{-\lambda x}$$
$$\mu = \tfrac{1}{\lambda}$$
$$\sigma^2 = \tfrac{1}{\lambda^2}$$



**6** **Normal distribution** ($\mathcal{N}(\mu, \sigma^2)$): Limit as $n \to \infty$ of the distribution of $\frac{X_1+X_2+\cdots+X_n}{\sqrt{n}}$, for any independent sequence $X_1, \ldots, X_n$ of identically distributed random variables (i.i.d.) with $\mathbb{E}[X_1] = \mu$ and $\mathrm{Var}(X_1) = \sigma^2 < \infty$ (see Central Limit Theorem).

$$f(x) = \tfrac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

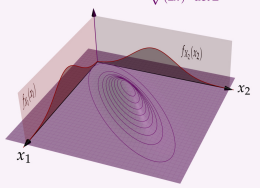



**7** **Multivariate normal distribution** ($\mathcal{N}(\mathbf{0}, \Sigma)$): if $\boldsymbol{Z} = (Z_1, Z_2, \ldots, Z_n)$ is a vector of independent $\mathcal{N}(0,1)$'s, $A$ is an $m \times n$ matrix of constants, and $\mu \in \mathbb{R}^m$, then the vector

$$\boldsymbol{X} = A\boldsymbol{Z} + \mu$$

is **multivariate normal**. The covariance matrix of $\mathbf{X}$ is $\Sigma = AA'$.

$$f(\mathbf{x}) = \tfrac{1}{\sqrt{(2\pi)^n \det \Sigma}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})'\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$




## Probability: Central Limit Theorem

**1** A sequence of random variables $X_1, X_2, \ldots$, **converges in probability** to $X$ if $\mathbb{P}(|X_n - X| > \epsilon) \to 0$ as $n \to \infty$, for any $\epsilon > 0$.

**2** A sequence $\nu_1, \nu_2, \ldots$ of probability measures on $\mathbb{R}^n$ converges to a probability measure $\nu$ if $\nu_n(A) \to \nu(A)$ for every set $A \subset \mathbb{R}^n$ with the property that $\nu(\partial A) = 0$ (intuitively, two measures are close if they put approximately the same amount of mass in approximately the same places). We say $X_n$ **converges in distribution** to $\nu$ if the distribution of $X_n$ converges to $\nu$.

**3** **Chebyshev's inequality**: if $X$ is a random variable with variance $\sigma^2 < \infty$, then $X$ differs from its mean by more than $k$ standard deviations with probability at most $k^{-2}$:

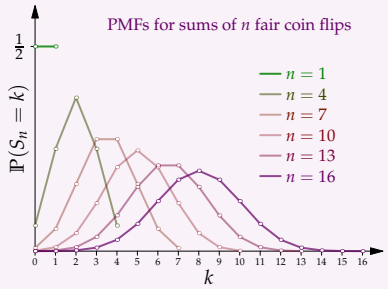
$$\mathbb{P}(|X - \mathbb{E}[X]| > k\sigma) \leq \frac{1}{k^2}$$

**4** **Law of large numbers**: if $X_1, X_2, \ldots$ is a sequence of independent observations from a finite-variance distribution with mean $\mu$, then the sequence's running average converges in probability to $\mu$: for all $\epsilon > 0$,

$$\mathbb{P}\left(\frac{X_1 + \cdots + X_n}{n} \notin [\mu - \epsilon, \mu + \epsilon]\right) \to 0,$$

as $n \to \infty$.

**5** The PDF of a sum of $n$ independent observations from a finite-variance distribution looks increasingly bell-shaped as $n$ increases, *regardless of the distribution being sampled from.*




**6** We define the **standardized running sum** of $X_1, X_2, \ldots$ to have zero mean and unit variance for all $n \geq 1$:

$$S_n^* = \frac{X_1 + X_2 + \cdots + X_n - n\mu}{\sigma\sqrt{n}}$$

**7** **Central limit theorem**: the sequence of standardized sums of an i.i.d. sequence of finite-variance random variables converges in distribution to $\mathcal{N}(0,1)$: for any interval $[a, b]$, we have

$$\mathbb{P}(S_n^* \in [a, b]) \to \int_a^b \frac{1}{\sqrt{2\pi}} e^{-t^2/2}\, dt$$

as $n \to \infty$.

**8** **Multivariate central limit theorem**: If $\mathbf{X}_1, \mathbf{X}_2, \ldots$ is a sequence of independent random vectors whose common distribution has mean $\mu$ and covariance matrix $\Sigma$, then

$$\frac{\mathbf{X}_1 + \mathbf{X}_2 + \cdots + \mathbf{X}_n - n\mu}{\sqrt{n}}$$

converges in distribution to $\mathcal{N}(\mathbf{0}, \Sigma)$.

**9** The central limit theorem explains the ubiquity of the normal distribution in statistics: many random quantities may be realized as a sum of a multitude of independent contributions.

## Statistics: Point estimation

**1** The central problem of statistics is to make inferences about a population or data-generating process based on the information in a finite sample drawn from the population.

**2** **Parametric estimation** involves an assumption that the distribution of the data-generating process comes from a family of distributions parameterized by finitely many real numbers, while **nonparametric estimation** does not. *Examples: An estimator that assumes the data are normally distributed is parametric, while histograms are nonparametric.*

**3** **Point estimation** is the inference of a single real-valued feature of the distribution of the data-generating process (such as its mean, variance, or median).

**4** A **statistical functional** is any function $T$ from the set of distributions to $[-\infty, \infty]$. An **estimator** $\widehat{\theta}$ is a random variable defined in terms of $n$ i.i.d. random variables, the purpose of which is to approximate some statistical functional of the random variables' common distribution. *Example: Suppose that $T(\nu)$ = the mean of $\nu$, and that $\widehat{\theta} = (X_1 + \cdots + X_n)/n$.*

**5** The empirical measure $\widehat{\nu}$ of $X_1, \ldots, X_n$ is the probability measure which assigns mass $\frac{1}{n}$ to each sample's location. The **plug-in estimator** of $\theta = T(\nu)$ is obtained by applying $T$ to the empirical measure: $\widehat{\theta} = T(\widehat{\nu})$.

**6** Given a distribution $\nu$ and a statistical functional $T$, let $\theta = T(\nu)$. The **bias** of an estimator of $\theta$ is the difference between the estimator's expected value and $\theta$. *Example: The expectation of the sample mean $\widehat{\theta} = (X_1 + \cdots + X_n)/n$ is $\mathbb{E}(X_1 + \cdots + X_n)/n = \mathbb{E}[\nu]$, so the bias of the sample mean is zero.*

**7** The **standard error** $\mathrm{se}(\widehat{\theta})$ of an estimator $\widehat{\theta}$ is its standard deviation.

**8** An estimator is **consistent** if $\widehat{\theta} \to \theta$ in probability as $n \to \infty$.

**9** The **mean squared error** of an estimator is defined to be

$$\mathrm{MSE}(\theta) = \mathbb{E}[(\widehat{\theta} - \theta)^2].$$

**10** MSE is equal to variance plus squared bias. Therefore, MSE converges to zero as the sample size goes to $\infty$ if and only if variance and bias both converge to zero.

## Statistics: Confidence intervals

**1** Consider an unknown probability distribution $\nu$ from which we get $n$ independent observations $X_1, \ldots, X_n$, and suppose that $\theta$ is the value of some statistical functional of $\nu$. A **confidence interval** for $\theta$ is an interval-valued function of the sample data $X_1, \ldots, X_n$. A confidence interval has **confidence level** $1 - \alpha$ if it contains $\theta$ with probability at least $1 - \alpha$.

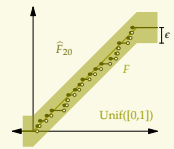
**2** If $\widehat{\theta}$ is unbiased, then $\left(\widehat{\theta} - k\,\mathrm{se}(\widehat{\theta}), \widehat{\theta} + k\,\mathrm{se}(\widehat{\theta})\right)$ is a $1 - \frac{1}{k^2}$ confidence interval, by Chebyshev's inequality.

**3** If $\widehat{\theta}$ is unbiased and approximately normally distributed, then $\left(\widehat{\theta} - 1.96\,\mathrm{se}(\widehat{\theta}), \widehat{\theta} + 1.96\,\mathrm{se}(\widehat{\theta})\right)$ is an approximate 95% confidence interval, since 95% of the mass of the standard normal distribution is in the interval $[-1.96, 1.96]$.

**4** Let $I \subset \mathbb{R}$, and suppose that $T$ is a function from the set of distributions to the set of real-valued functions on $I$. A $1 - \alpha$ **confidence band** for $T(\nu)$ is pair of random functions $y_{\min}$ and $y_{\max}$ from $I$ to $\mathbb{R}$ defined in terms of $n$ independent observations from $\nu$ and having $y_{\min} \leq T(\nu) \leq y_{\max}$ everywhere on $I$ with probability at least $1 - \alpha$.

## Statistics: Empirical CDF convergence

**1** Statistics is predicated on the idea that a distribution is well-approximated by independent observations therefrom. The **Glivenko-Cantelli theorem** is one formalization of this idea: If $F$ is the CDF of a distribution $\nu$ and $\widehat{F}_n$ is the CDF of the empirical distribution $\widehat{\nu}_n$ of $n$ observations from $\nu$, then $F_n$ converges to $F$ along the whole number line:




$$\max_{x \in \mathbb{R}} |F(x) - \widehat{F}_n(x)| \to 0 \quad \text{as } n \to \infty,$$

in probability.

**2** The **Dvoretzky-Kiefer-Wolfowitz inequality** (DKW) says that the graph of $\widehat{F}_n$ lies in the $\epsilon$-band around the graph of $F$ with probability at least $1 - 2e^{-2n\epsilon^2}$.

## Statistics: Bootstrapping

**1** **Bootstrapping** is the use of simulation to approximate the value of the plug-in estimator of a statistical functional which is expressed in terms of independent observations from $\nu$.

*Example: if $\theta = T(\nu)$ is the variance of the median of 3 independent observations from $\nu$, then the bootstrap estimate of $\theta$ is obtained as a Monte Carlo approximation of $T(\widehat{\nu})$: we sample 3 times (with replacement) from $\{X_1, \ldots, X_n\}$, record the median, repeat $B$ times for $B$ large, and take the sample variance of the resulting list of $B$ numbers.*

**2** The bootstrap approximation of $T(\widehat{\nu})$ may be made as close to $T(\widehat{\nu})$ as desired by choosing $B$ large enough. The difference between $T(\nu)$ and $T(\widehat{\nu})$ is likely to be small if $n$ is large (that is, if many observations from $\nu$ are available).

**3** The bootstrap is useful for computing standard errors, since the standard error of an estimator is often infeasible to compute analytically but conducive to Monte Carlo approximation.

## Statistics: Maximum likelihood estimation

**1** Maximum likelihood estimation is a general approach for proposing an estimator. Consider a parametric family $\{f_{\boldsymbol{\theta}}(x) : \boldsymbol{\theta} \in \mathbb{R}^d\}$ of PDFs or PMFs. Given $\mathbf{x} \in \mathbb{R}^n$, the **likelihood** $\mathcal{L}_{\mathbf{x}} : \mathbb{R}^d \to \mathbb{R}$ is defined by

$$\mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta}) = f_{\boldsymbol{\theta}}(x_1) f_{\boldsymbol{\theta}}(x_2) \cdots f_{\boldsymbol{\theta}}(x_n).$$

If $\mathbf{X}$ is a vector of $n$ independent observations drawn from $f_{\boldsymbol{\theta}}(x)$, then $\mathcal{L}_{\mathbf{X}}(\boldsymbol{\theta})$ is small or zero when $\boldsymbol{\theta}$ is not in accordance with the observed data.

*Example: Suppose $x \mapsto f(x;\theta)$ is the density of a uniform random variable on $[0, \theta]$. We observe four observations drawn from this distribution: 1.41, 2.45, 6.12, and 4.9. Then the likelihood of $\theta = 5$ is zero, and the likelihood of $\theta = 10^6$ is very small.*

**2** The **maximum likelihood estimator** is

$$\widehat{\boldsymbol{\theta}}_{\mathrm{MLE}} = \mathrm{argmax}_{\boldsymbol{\theta} \in \mathbb{R}^d} \mathcal{L}_{\mathbf{X}}(\boldsymbol{\theta}).$$

Equivalently, $\widehat{\boldsymbol{\theta}}_{\mathrm{MLE}} = \mathrm{argmax}_{\boldsymbol{\theta} \in \mathbb{R}^d} \ell_{\mathbf{X}}(\boldsymbol{\theta})$, where $\ell_{\mathbf{x}}(\boldsymbol{\theta})$ denotes the logarithm of $\mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta})$.

*Example: Suppose that $x \mapsto f(x; \mu, \sigma^2)$ is the normal density with mean $\mu$ and variance $\sigma^2$. Then the maximum likelihood estimator is the minimizer of the log-likelihood*

$$-\frac{n}{2}\log 2\pi - n \log \sigma - \frac{(X_1 - \mu)^2}{2\sigma^2} - \cdots - \frac{(X_n - \mu)^2}{2\sigma^2}$$

*Setting the derivatives with respect to $\mu$ and $\sigma^2$ equal to zero, we find $\mu = \overline{X} = \frac{1}{n}(X_1 + \cdots + X_n)$ and $\sigma^2 = \frac{1}{n}((X_1 - \overline{X})^2 +$*

$\cdots + (X_n - \overline{X})^2)$. *So the maximum likelihood estimators agree with the plug-in estimators.*

3 MLE enjoys several nice properties: under certain regularity conditions, we have (stated for $\theta \in \mathbb{R}^1$):

(i) **Consistency**: $\mathbb{E}[(\widehat{\theta} - \theta)^2] \to 0$ as the number of observations goes to $\infty$.

(ii) **Asymptotic normality**: $(\widehat{\theta} - \theta)/\sqrt{\operatorname{Var}\widehat{\theta}}$ converges to $\mathcal{N}(0,1)$ as the number of observations goes to $\infty$.

(iii) **Asymptotic optimality**: the MSE of the MLE converges to 0 approximately as fast as the MSE of any other consistent estimator.

4 Potential difficulties with the MLE:

(i) **Computational challenges**. It might be hard to work out where the maximum of the likelihood occurs, either analytically or numerically.

(ii) **Misspecification**. The MLE may be inaccurate if the distribution of the observations is not in the specified parametric family.

(iii) **Unbounded likelihood**. If the likelihood function is not bounded, then $\widehat{\theta}$ is not well-defined.

## Statistics: Hypothesis testing

1 **Hypothesis testing** is a disciplined framework for adjudicating whether observed data do not support a given hypothesis.

2 Consider an unknown distribution from which we will sample $n$ observations $X_1, \ldots X_n$.

(i) We state a hypothesis $H_0$–called the **null hypothesis**–about the distribution.

(ii) We come up with a **test statistic** $T$, which is a function of the data $X_1, \ldots X_n$, for which we can evaluate the distribution of $T$ assuming the null hypothesis.

(iii) We give an **alternative hypothesis** $H_a$ under which $T$ is expected to be significantly different from its value under $H_0$.

(iv) We give a significance level $\alpha$ (like 5% or 1%), and based on $H_a$ we determine a set of values for $T$—called the *critical region*—which $T$ would be in with probability at most $\alpha$ under the null hypothesis.

(v) **After setting $H_0$, $H_a$, $\alpha$, $T$, and the critical region**, we run the experiment, evaluate $T$ on the sample we get, and record the result as $t_{\text{obs}}$.

(vi) If $t_{\text{obs}}$ falls in the critical region, we reject the null hypothesis. The corresponding ***p*-value** is defined to be the minimum $\alpha$-value which would have resulted in rejecting the null hypothesis, with the critical region chosen in the same way*.

*Example: Muriel Bristol claims that she can tell by taste whether the tea or the milk was poured into the cup first. She is given eight cups of tea, four poured milk-first and four poured tea-first.*

*We posit a null hypothesis that she isn't able to discern the pouring method, under which the number of cups identified correctly is 4 with probability $1/\binom{8}{4} \approx 1.4\%$ and at least 3 with probability $17/70 \approx 24\%$. Therefore, at the 5% significance level, only a correct identification of all the cups would give us grounds to reject the null hypothesis. The p-value in that case would be 1.4%.*

3 Failure to reject the null hypothesis is not necessarily evidence *for* the null hypothesis. The **power** of a hypothesis test is the conditional probability of rejecting the null hypothesis given that the alternative hypothesis is true. A $p$-value may be high either because the null hypothesis is true or because the test has low power.

4 The **Wald test** is based on the normal approximation. Consider a null hypothesis $\theta = 0$ and the alternative hypothesis $\theta \neq 0$, and suppose that $\widehat{\theta}$ is approximately normally distributed. The Wald test rejects the null hypothesis at the 5% significance level if $|\widehat{\theta}| > 1.96\,\text{se}(\widehat{\theta})$.

5 The **random permutation test** is applicable when the null hypothesis is that the mean of a given random variable is equal for two populations.

(i) We compute the difference between the sample means for the two groups.

(ii) We randomly re-assign the group labels and compute the resulting sample mean differences. Repeat many times.

(iii) We check where the original difference falls in the sorted list of re-sampled differences.

*Example: Suppose the heights of the Romero sons are 72, 69, 68, and 66 inches, and the heights of the Larsen sons are 70, 65, and 64 inches. Consider the null hypothesis that the expected heights are the same for the two families, and the alternative hypothesis that the Romero sons are taller on average (with $\alpha = 5\%$). We find that the sample mean difference of about 2.4 inches is larger than 88.5% of the mean differences obtained by resampling many times. Since 88.5% < 95%, we retain the null hypothesis.*

6 If we conduct many hypothesis tests, then the probability of obtaining some false rejections is high (`xkcd.com/882`). This is called the **multiple testing problem**. The **Bonferroni method** is to reject the null hypothesis only for those tests whose $p$-values are less than $\alpha$ divided by the number of hypothesis tests being run. This ensures that the probability of having even one false rejection is less than $\alpha$, so it is very conservative.

## Statistical Learning: Theory

1 **Statistical learning**: Given some observations from a probability space with an unknown probability measure, we seek to draw conclusions about the measure.

2 **Supervised learning**: $(\mathbf{X}, Y)$ is drawn from an unknown probability measure $\mathbb{P}$ on a product space $\mathcal{X} \times \mathcal{Y}$, and we aim to predict $Y$ given $\mathbf{X}$, based on a i.i.d. collection of observations from $\mathbb{P}$ (the **training data**).

*Example: $\mathbf{X} = [X_1, X_2]$, where $X_1$ is the color of a banana, $X_2$ is the weight of the banana, and $Y$ is a measure of deliciousness. Values of $X_1, X_2$, and $Y$ are recorded for many bananas, and they are used to predict $Y$ for other bananas whose $\mathbf{X}$ values are known.*

3 We call the components of $\mathbf{X}$ *features*, *predictors*, or *input variables*, and we call $Y$ the *response variable* or *output variable*.

4 A supervised learning problem is a **regression** problem if $Y$ is quantitative ($\mathcal{Y} \subset \mathbb{R}$) and a **classification** problem if $\mathcal{Y}$ is a set of labels.

5 To choose a prediction function $h : \mathcal{X} \to \mathcal{Y}$, we specify a

(i) a space $\mathcal{H}$ of candidate functions, and

(ii) a **loss** (or *risk*) **functional** $L$ from $\mathcal{H}$ to $\mathbb{R}$.

The **target function** is $\operatorname{argmin}_{h \in \mathcal{H}} L(h)$.

6 If the loss functional for a regression problem is

$$L(h) = \mathbb{E}[(h(X) - Y)^2]$$

and $\mathcal{H}$ contains $r(\mathbf{x}) = \mathbb{E}[Y \mid \mathbf{X} = \mathbf{x}]$, then $r$ is the target function. If the loss functional for a classification problem is

$$L(h) = \mathbb{E}\left[\mathbf{1}_{\{h(X) \neq Y\}}\right],$$

and $\mathcal{H}$ contains $G(\mathbf{x}) = \operatorname{argmax}_c \mathbb{P}(Y = c \mid \mathbf{X} = \mathbf{x})$, then $G$ is the target function.

7 Since $\mathbb{P}$ is unknown, we must approximate the target function with a function $\widehat{h}$ whose values can be computed from the training data. A **learner** is a function which takes a set of training data and returns a prediction function $\widehat{h}$.

8 The **empirical probability measure** on $\mathcal{X} \times \mathcal{Y}$ is the measure which assigns a probability mass of $\frac{1}{n}$ to the location of each training sample $(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_n, Y_n)$. The **empirical risk** of a candidate function $h$ is the risk functional evaluated with respect to the empirical measure of the training data. The **empirical risk minimizer** (ERM) is the function which minimizes empirical risk.

9 **Generalization error** (or *test* error) is the difference between empirical risk and the actual value of the risk functional.

10 The ERM can **overfit**, meaning that test error and $L(\widehat{h})$ are large despite small empirical risk.

*Example: if $\mathcal{H}$ is the space of polynomials and no two training observations have the same $\mathbf{x}$ values, then then there are functions in $\mathcal{H}$ which have zero empirical risk.*


11 Mitigate overfitting with **inductive bias**:

(i) Use a restrictive class $\mathcal{H}$ of candidate functions.

(ii) **Regularize**: add a term to the loss functional which penalizes complexity.

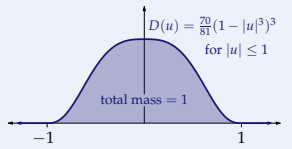
12 Inductive bias can lead to **underfitting**: relevant relations are missed, so both training and test error are larger than necessary. The tension between the costs of high inductive bias and the costs of low inductive bias is called the **bias-complexity** (or *bias-variance*) **tradeoff**.

13 **No-free-lunch theorem**: all learners are equal on average (over all possible problems), so inductive bias appropriate to a given type of problem is essential to have an effective learner for that type of problem.

## Statistical Learning: Kernel density estimation

1 Given $n$ observations $X_1, \ldots, X_n$ from a distribution with density $f$ on $\mathbb{R}$, we can estimate the PDF of the distribution by placing $1/n$ units of probability mass in a small pile around each sample.

2 We choose a **kernel** function for the shape of each pile:




3 The width of each pile is specified by a **bandwidth** $\lambda$: $D_\lambda(u) = \frac{1}{\lambda} D\left(\frac{u}{\lambda}\right)$.

4 The **kernel density estimator** with bandwidth $\lambda$ is the sum of the piles at each sample:

$$\widehat{f}_\lambda(x) = \frac{1}{n} \sum_{i=1}^{n} D_\lambda(x - X_i).$$

5 To choose a suitable bandwidth, we seek to minimize the **integrated squared error** (ISE) $L(f) = \int_{\mathbb{R}} (f - \widehat{f})^2$.

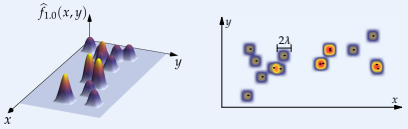
6 We approximate the minimizer of $L$ with the minimizer of the **cross-validation loss estimator**

$$J(f) = \int_{\mathbb{R}} \widehat{f}_\lambda^2 - \frac{2}{n} \sum_{i=1}^{n} \widehat{f}_\lambda^{(-i)}(X_i),$$

where $\widehat{f}_\lambda^{(-i)}$ is the KDE with the $i$th sample omitted.

7 If $f$ is a density on $\mathbb{R}^2$, then we use the KDE

$$\widehat{f}_\lambda(x, y) = \frac{1}{n} \sum_{i=1}^{n} D_\lambda(x - X_i) D_\lambda(y - Y_i).$$




8 **Stone's theorem** says that the ratio of the CV ISE to the optimal-$\lambda$ ISE converges to 1 in probability as $n \to \infty$. Also, the optimal $\lambda$ goes to 0 like $\frac{1}{n^{1/5}}$, and the minimal ISE goes to 0 like $\frac{1}{n^{4/5}}$.

9 The **Nadaraya-Watson** nonparametric regression estimator $\widehat{r}(x)$ computes $\mathbb{E}[Y \mid X = x]$ with respect to the estimated density $\widehat{f}_\lambda$. Equivalently, we average the $Y_i$'s, weighted according to horizontal distance from $x$:

$$\sum_{i=1}^{n} Y_i D(x - X_i) \Big/ \sum_{i=1}^{n} D(x - X_i).$$

## Statistical Learning: Parametric regression

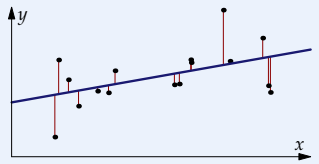
1 **Parametric regression** uses a family $\mathcal{H}$ of candidate functions which is indexed by finitely many parameters.

2 **Linear regression** uses the set of affine functions: $\mathcal{H} = \{\mathbf{x} \mapsto \beta_0 + [\beta_1, \ldots, \beta_p] \cdot \mathbf{x} : \beta_0, \ldots \beta_p \in \mathbb{R}\}$.

3 We choose the parameters to minimize a risk function, customarily the **residual sum of squares**:

$$\text{RSS}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \beta_0 - \boldsymbol{\beta} \cdot \mathbf{x}_i)^2 = |\mathbf{y} - X\boldsymbol{\beta}|^2,$$

where $\mathbf{y} = [y_1, \ldots, y_n]$, $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_p]$, and $X$ is an $n \times (p+1)$ matrix whose $i$th row is a 1 followed by the components of $\mathbf{x}_i$.




4 The RSS minimizer is $\boldsymbol{\beta} = (X'X)^{-1}X'Y$.

5 We can use the linear regression framework to do **polynomial regression**, since a polynomial is linear in its coefficients: we supplement the list of features with products of the original regressors.

6 Regularizing linear regression by penalizing the sum of the squares of the regression coefficients is called **ridge** regression, while penalizing the sum of the absolute values of the coefficients is called **lasso** regression.

DATA 1010 · Brown University · Samuel S. Watson

## Statistical Learning: Optimal classification

**1** Consider a classification problem with feature set $\mathcal{X}$ and class set $\mathcal{Y}$. For each $y \in \mathcal{Y}$, we define $p_y = \mathbb{P}(Y = y)$ and let $f_y$ be the conditional PMF or PDF of $\mathbf{X}$ given $\{Y = y\}$ ($y$'s **class conditional distribution**).

**2** Given a prediction function (or **classifier**) $h$ and an enumeration of the elements of $\mathcal{Y}$ as $\{y_1, y_2, \ldots, y_{|\mathcal{Y}|}\}$, we define the (normalized) **confusion matrix** of $h$ to be the $|\mathcal{Y}| \times |\mathcal{Y}|$ matrix whose $(i,j)$th entry is $\mathbb{P}(h(\mathbf{X}) = y_i \,|\, Y = y_j)$.

**3** If $\mathcal{Y} = \{-,+\}$, the conditional probability of correct classification given a positive sample is the **detection rate** (DR), while the conditional probability of incorrect classification given a negative sample is the **false alarm rate** (FAR).

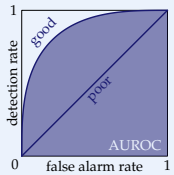
**4** The **precision** of a classifier is the conditional probability that a sample is positive given that the classifier predicts positive, and **recall** is a synonym of detection rate.

**5** The Bayes classifier $G(\mathbf{x}) = \operatorname{argmax}_y p_y f_y(\mathbf{x})$ minimizes the misclassification probability but gives equal weight to both types of misclassification.

**6** The **likelihood ratio test** generalizes the Bayes classifier by allowing a variable tradeoff between false-alarm rate and detection rate: given $t > 0$, we say $h_t(\mathbf{x}) = -1$ if $f_+(\mathbf{x})/f_-(\mathbf{x}) < t$ and $h_t(\mathbf{x}) = 1$ otherwise.

**7** The **Neyman-Pearson lemma** says that no classifier does better on both false alarm rate and detection rate than $h_t$.

**8** The **receiver operating characteristic** of $h_t$ is the curve $\{(\mathrm{FAR}(h_t), \mathrm{DR}(h_t)) : t \in [0,\infty]\}$. The AUROC (area under the ROC) is close to 1 for an excellent classifier and close to $\frac{1}{2}$ for a worthless one. NP says that no classifier is above the ROC. We choose a point on the ROC curve based on context-specific considerations.




## Statistical Learning: QDA, LDA, Naive Bayes

**1** **Quadratic discriminant analysis** (QDA) is a classification algorithm which uses the training data to estimate the mean $\boldsymbol{\mu}_y$ and covariance matrix $\Sigma_y$ of each class conditional distribution:

$$\widehat{\boldsymbol{\mu}}_y = \mathrm{mean}(\{\mathbf{x}_i : y_i = y\})$$

$$\widehat{\Sigma}_y = \mathrm{mean}(\{(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_y)(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_y)' : y_i = y\}).$$

Each distribution is assumed to be multivariate normal $(\mathcal{N}(\widehat{\boldsymbol{\mu}}_y, \widehat{\Sigma}_y))$ and the classifier $h(y) = \operatorname{argmax}_y \widehat{p}_y \widehat{f}_y(\mathbf{x})$ is proposed (where $\{\widehat{p}_y : y \in \mathcal{Y}\}$ are the class proportions from the training data).

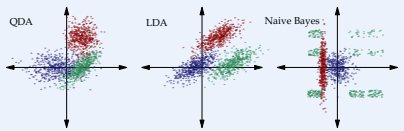
**2** **Linear discriminant analysis** (LDA) is the same as QDA except the class covariance matrices are assumed to be equal and are estimated using all of the data, not just class-specific observations.

**3** QDA and LDA are so named because they yield class prediction boundaries which are quadric surfaces and hyperplanes, respectively.

**4** A **Naive Bayes** classifier assumes that the features are conditionally independent given $Y$:

$$f_y(x_1, \ldots, x_p) = f_{y,1}(x_1) \cdots f_{y,d}(x_p),$$

for some $f_{y,1}, \ldots f_{y,p}$. Assumption-satisfying examples:



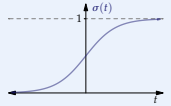

## Statistical Learning: Logistic regression

**1** **Logistic regression** for binary classification estimates $r(\mathbf{x}) = \mathbb{P}(Y = 1 \,|\, X = x)$ as a *logistic function* of a linear function of $\mathbf{x}$:

$$\widehat{r}(\mathbf{x}) = \sigma(\boldsymbol{\beta} \cdot \mathbf{x} + \alpha),$$

where

$$\sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}}.$$




**2** We choose $\alpha$ and $\beta_1, \ldots, \beta_p$ to minimize the risk

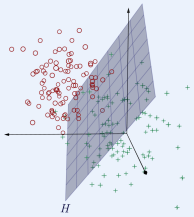
$$L(r) = \sum_{i=1}^{n} \left[ y_i \log \frac{1}{r(x_i)} + (1 - y_i) \log \frac{1}{1 - r(x_i)} \right],$$

which applies large penalty if $y_i = 1$ and $r(x_i)$ is close to zero or if $y_i = 0$ and $r(x_i)$ is close to 1.

**3** $L$ is convex, so it can be reliably minimized using numerical optimization algorithms.

## Statistical Learning: Support vector machines

**1** A **support vector machine** (SVM) chooses a hyperplane $H \subset \mathbb{R}^p$ and predicts classification ($\mathcal{Y} = \{-1, +1\}$) based on which side of $H$ the feature vector $\mathbf{x}$ lies on.



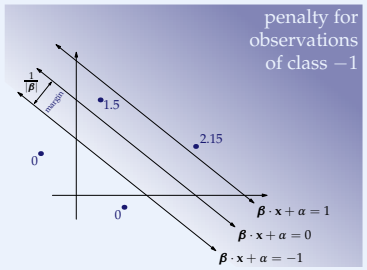

**2** $\mathbf{x} \mapsto \mathrm{sgn}(\boldsymbol{\beta} \cdot \mathbf{x} + \alpha)$ is the prediction function, where $H = \{\mathbf{x} \in \mathbb{R}^p : \boldsymbol{\beta} \cdot \mathbf{x} + \alpha = 0\}$.

**3** We train the SVM with the risk

$$L(\boldsymbol{\beta}, \alpha) = \lambda |\boldsymbol{\beta}|^2 + \frac{1}{n} \sum_{i=1}^{n} \left[1 - y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + \alpha)\right]_+$$

where $[u]_+$ denotes $\max(0, u)$, the positive part of $u$.

**4** The parameters $\boldsymbol{\beta}$ and $\alpha$ encode both $H$ and the a distance—called the **margin**—from $H$ to a parallel hyperplane where we begin penalizing for lack of decisively correct classification. The margin is $1/|\boldsymbol{\beta}|$ (and can be adjusted without changing $H$ by scaling $\boldsymbol{\beta}$ and $\alpha$).




**5** If $\lambda$ is small, then the optimization prioritizes the correctness term and uses a small margin if necessary. If $\lambda$ is large, the optimization must minimize a large-margin incorrectness penalty. A value for $\lambda$ may be chosen by cross-validation.

**6** Applying a function $\phi$ to map the feature vectors to a higher dimensional allows us to find nonlinear separating surfaces in the original feature space. The function $K = (\mathbf{x}, \mathbf{y}) \mapsto \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ is called the *kernel* associated with the transformation $\phi$. Finding the separating hyperplane in the higher dimensional space comes down to finding a vector $\boldsymbol{\eta}$ solving the problem (with $\odot$ and $\preccurlyeq$ as pointwise operations, $\mathbf{1}$ a vector of ones, and $\mathcal{K}_{i,j} = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$)

$$\text{minimize} \quad \frac{1}{2}(\boldsymbol{\eta} \odot \mathbf{y})' \mathcal{K}(\boldsymbol{\eta} \odot \mathbf{y}) - \mathbf{1}'\boldsymbol{\eta}$$

$$\text{subject to} \quad 0 \preccurlyeq \boldsymbol{\eta} \preccurlyeq C \text{ and } \boldsymbol{\eta}'\mathbf{y} = 0.$$

The prediction vector for an $n_{\text{test}} \times n$ feature matrix $X_{\text{test}}$ is

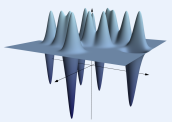
$$\mathcal{K}_{\text{test}}(\widehat{\boldsymbol{\eta}} \odot \mathbf{y}) + \widehat{\alpha}\mathbf{1},$$

where $\mathcal{K}_{\text{test}}$ is the $n_{\text{test}} \times n$ matrix whose $(i,j)$th entry is obtained by applying $K$ to the $i$th row of $X_{\text{test}}$ and the $j$th row of $X$, and where $\widehat{\alpha}$ is any entry of

$$\mathbf{y} - \mathcal{K}(\widehat{\boldsymbol{\eta}}' \mathbf{y})$$

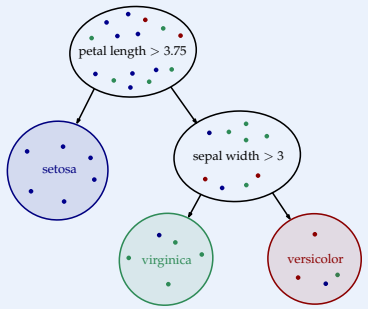
for which the corresponding entry in $\widehat{\boldsymbol{\eta}}$ is strictly between 0 and $C$. Only the support vectors ($\eta_i > 0$) need be retained.

**7** The prediction function (before taking the sign) is a constant plus a linear combination of functions of the form $\mathbf{x} \mapsto K(\mathbf{x}, \mathbf{x}_i)$, where $\mathbf{x}_i$ is the $i$th training observation. Example shown for 2D feature space with **radial basis function** kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{e}^{-|\mathbf{x}_i - \mathbf{x}_j|^2}$.



## Statistical Learning: Trees and Ensemble Methods

**1** A **decision tree** classifier maps features to response using a flowchart consisting of single-feature decisions.




**2** The **Gini impurity** of a list of classified objects is the probability that two independent random elements from the list have different classes: if $p_1, \ldots, p_k$ are the class proportions, then

$$G = 1 - (p_1^2 + \cdots + p_k^2).$$

**3** **CART** is a greedy decision tree training algorithm. Each node in the tree, starting from the first node, chooses its feature and threshold so as to separate classes as much as possible: it minimizes $p_1 G_1 + p_2 G_2$, where $p_1$ and $p_2$ are the proportion of training observations that go to the two child nodes and $G_1$ and $G_2$ are the child-node Gini impurities.

**4** A **random forest** classifier takes a vote among an ensemble of random decision trees. The decision trees may be randomized by training on a with-replacement random sample of the training data (this is called **bagging**) or by randomly restricting the set of features searched for splits at each node.

**5** Another ensemble method is **boosting**: we train a model on the original data, then train another model to address the deficiencies of the first one, etc. **Adaboost** *weights* training observations for the next model in the stack based how inaccurately they're being predicted with the current model. **Gradient boosting** fits the next model on the *residuals* (the differences between response and currently predicted response).

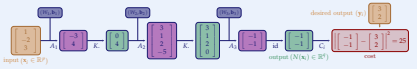
## Statistical Learning: Neural networks

**1** A **multilayer perceptron** $N : \mathbb{R}^p \to \mathbb{R}^q$ is a composition of affine transformations and componentwise applications of a function $K : \mathbb{R} \to \mathbb{R}$.

(i) We call $K$ the **activation function**. Common choices:
  (a) $u \mapsto \max(0, u)$ (rectifier, or ReLU)
  (b) $u \mapsto 1/(1 + e^{-u})$ (logistic)

(ii) *Component-wise application of* $K$ on $\mathbb{R}^t$ refers to the function $K.(x_1, \ldots, x_t) = (K(x_1), \ldots, K(x_t))$.

(iii) An affine transformation from $\mathbb{R}^t$ to $\mathbb{R}^s$ is a map of the form $A(\mathbf{x}) = W\mathbf{x} + \mathbf{b}$, where $W$ is an $s \times t$ matrix and $\mathbf{b} \in \mathbb{R}^s$. Entries of $W$ are called **weights** and entries of $\mathbf{b}$ are called **biases**.



**2** The **architecture** of a neural network is the sequence of dimensions of the domains and codomains of its affine maps. For example, a neural net with $W_1 \in \mathbb{R}^{5\times 3}$, $W_2 \in \mathbb{R}^{4\times 5}$, and $W_3 \in \mathbb{R}^{1\times 4}$ has architecture $[3, 5, 4, 1]$.

**3** Given a training sample $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, we obtain a neural net regression function by minimizing $L(N) = \sum_{i=1}^{n} C(N(\mathbf{x}_i), \mathbf{y}_i)$ where $C(\mathbf{y}, \mathbf{y}_i) = |\mathbf{y} - \mathbf{y}_i|^2$.

**4** For classification, we

(i) let $\mathbf{y}_i = [0, \ldots, 0, 1, 0, \ldots 0] \in \mathbb{R}^{|\mathcal{Y}|}$, with the location of the nonzero entry indicating class (this is called **one-hot encoding**),

(ii) replace the identity map in the diagram with the **softmax** function $\mathbf{u} \mapsto \left[\mathrm{e}^{u_j} / \left(\sum_{k=1}^{n} \mathrm{e}^{u_k}\right)\right]_{j=1}^{|\mathcal{Y}|}$, and

(iii) replace the cost function with $C(\mathbf{y}, \mathbf{y}_i) = -\log(\mathbf{y} \cdot \mathbf{y}_i)$.

**5** When the weight matrices are large, they have many parameters to tune. We use a custom optimization scheme:

(i) Start with random weights and a training input $\mathbf{x}_i$.

(ii) **Forward propagation**: apply each successive map and store the vectors at each green or purple node. The vectors stored at the green nodes are called **activations**.

(iii) **Backpropagation**: starting with the *last* green node and working left, compute the change in cost per small change in the vector at each green or purple node. By the chain rule, each such gradient is equal to the gradient computed at right-adjacent node times the derivative of map between the two nodes. The derivative of $A_j$ is $W_j$, and the derivative of $K.$ is $\mathbf{v} \mapsto \mathrm{diag}\left(\left(\frac{\mathrm{d}K}{\mathrm{d}u}\right).(\mathbf{v})\right)$.

(iv) Compute the change in cost per small change in the weights and biases at each blue node. Each such gradient is equal to the gradient stored at the next purple node times the derivative of the intervening affine map. We have $\frac{\partial(W\mathbf{x}+\mathbf{b})}{\partial \mathbf{b}} = I$ and $\mathbf{v}\frac{\partial(W\mathbf{x}+\mathbf{b})}{\partial W} = \mathbf{v}'\mathbf{x}'$.

(v) **Stochastic gradient descent**: repeat (ii)–(iv) for each sample in a *randomly chosen subset* of the training set and determine the average desired change in weights and biases to reduce the cost function. Update the weights and biases accordingly and iterate to convergence.

## Statistical Learning: Dimension reduction

**1** The goal of **dimension reduction** is to map a set of $n$ points in $\mathbb{R}^p$ to a lower-dimensional space $\mathbb{R}^k$ while retaining as much of the data's structure as possible.

DATA 1010 · Brown University · Samuel S. Watson

**2** Dimension reduction can be used as a visualization aid or as a feature pre-processing step in a machine learning model.


**3** *Structure* may be taken to mean *variation about the center*, in which case we use **principal component analysis**:

(i) Store the points' components in an $n \times p$ matrix,

(ii) de-mean each column,

(iii) compute the SVD $U\Sigma V'$ of the resulting matrix, and

(iv) let $W$ be the first $k$ columns of $V$.

Then $WW' : \mathbb{R}^p \to \mathbb{R}^p$ is the rank-$k$ projection matrix which minimizes the sum of squared projection distances of the points, and $W' : \mathbb{R}^p \to \mathbb{R}^k$ maps each point to its coordinates in that $k$-dimensional subspace (with respect to the columns of $W$).

**4** *Structure* may be taken to mean *pairwise proximity of points*, which **stochastic neighbor embedding** attempts to preserve. Given the data points $\mathbf{x}_1, \dots, \mathbf{x}_n$ and a parameter $\rho$, we define

$$P_{i,j}(\sigma) = \frac{e^{-|\mathbf{x}_i - \mathbf{x}_j|^2/(2\sigma^2)}}{\sum_{k \neq j} e^{-|\mathbf{x}_k - \mathbf{x}_j|^2/(2\sigma^2)}},$$

and for each $j$ we define $\sigma_j$ to be the solution $\sigma$ of the equation

$$2^{-\sum_{i \neq j} P_{i,j}(\sigma) \log_2 P_{i,j}(\sigma)} = \rho.$$

The values $P_{i,j} = P_{i,j}(\sigma_j)$ describe the neighborliness of each point $i$ with respect to point $j$; they sum to 1 over all points. Fixing the **perplexity** $\rho$ ensures they neither concentrate nor spread out too much. For a given choice of locations $\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n$ in $\mathbb{R}^k$, we define

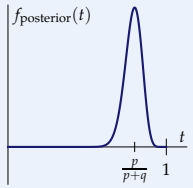
$$Q_{i,j} = \frac{(1 + |\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j|^2)^{-1}}{\sum_{k \neq j} (1 + |\tilde{\mathbf{x}}_k - \tilde{\mathbf{x}}_j|^2)^{-1}}.$$

We use gradient descent on $\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n$ to minimize

$$C(\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n) = \sum_{1 \leq i \neq j \leq n} P_{i,j} \log_2 \frac{P_{i,j}}{Q_{i,j}}.$$


## Statistical Learning: Bayesian inference

**1** In contrast to frequentist statistics, which represents model parameters as fixed and unknown, Bayesian statistics regards model parameters as random variables with a specified **prior** distribution. Observed data are used to obtain an updated **posterior** distribution, via Bayes' theorem. *Example: the prior for the heads probability of a weighted coin might be stipulated to be uniform on $[0,1]$. Then observing $p$ heads and $q$ tails results in a posterior distribution proportional to $t \mapsto t^p(1-t)^q$. Frequentist statistics would instead give a single point estimate (such as the maximum likelihood estimator $p/(p+q)$).*




**2** If the prior and posterior distributions belong to the same family of distributions, they are called **conjugate**. *Example: heads-probability distributions of the form $t \mapsto t^p(1-t)^q$ on $[0,1]$ update under coin flip observations by incrementing the exponents $p$ and $q$, so they are conjugate priors for the coin flip problem.*

**3** **Posterior is proportional to likelihood times prior**: if $X$ is the observed random variable and $\Theta$ the vector of model parameters, then

$$\overbrace{f(\theta \mid x)}^{\text{posterior}} = \frac{\overbrace{f(x \mid \theta)}^{\text{likelihood}} \overbrace{f(\theta)}^{\text{prior}}}{\underbrace{f(x)}_{\text{marginal}}}$$

where $f(\theta \mid x)$ is shorthand for the conditional density or mass function of $\Theta$ given $X = x$.

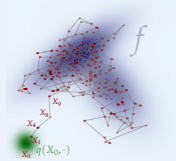
**4** Posterior distributions yield point estimates via measures of central tendency like the median or mean, as well as **Bayesian posterior intervals** (similar to confidence intervals) via their quantiles.

**5** Bayesian and frequentist statistics often yield similar results in the limit: under quite general conditions, the posterior distribution is approximately normal with mean converging to the maximum likelihood estimator as the sample size goes to $\infty$.

**6** Bayesian analysis often involves evaluating integrals. For example, the posterior mean is $\frac{\int_{\mathbb{R}^n} \theta \mathcal{L}(\theta) f(\theta)\, d\theta}{\int_{\mathbb{R}^n} \mathcal{L}(\theta) f(\theta)\, d\theta}$, where $\mathcal{L}(\theta) = f(x \mid \theta)$ is the likelihood. These integrals are often impossible to solve analytically or even approximate using exact numerical methods in the case where the parameter space is high-dimensional. One solution is to use **Monte Carlo** methods: use the identity $\int_{\mathbb{R}^n} g(x) f(x)\, dx = \mathbb{E}[g(X)]$ where $X$ is a random vector with density $f$. The expectation can be approximated by sampling repeatedly from the density $f$, using the law of large numbers.

**7** **Markov Chain Monte Carlo** (MCMC) is useful for approximating $\int_{\mathbb{R}^n} g(x) f(x)\, dx$. Metropolis-Hastings is a common class of MCMC algorithms: to sample from $f$, we:

(i) Fix a function $q : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ for which (a) $\tilde{x} \mapsto q(x, \tilde{x})$ is a probability density on $\mathbb{R}^n$, for all $x \in \mathbb{R}^n$, and (b) $q(x, \tilde{x}) = q(\tilde{x}, x)$ for all $x, \tilde{x} \in \mathbb{R}^n$.




(ii) Choose a starting point $X_0$ in some way, and sample $X_{\text{prop}}$ from the proposal density $\tilde{x} \mapsto q(X_0, \tilde{x})$.

(iii) With probability $\frac{f(X_{\text{prop}})}{f(X_0)}$ (or 1, if this ratio exceeds 1), accept the proposal and define $X_1$ to be $X_{\text{prop}}$. Otherwise, set $X_1 = X_0$.

(iv) Repeat steps (ii) and (iii) to obtain $X_2$ from $X_1$, $X_3$ from $X_2$, and so on. The resulting sequence $X_0, X_1, \dots$ has the property that the distribution of $X_n$ converges to $f$ as $n \to \infty$, as well as the property that the mean of the list $[g(X_0), g(X_1), \dots, g(X_n)]$ converges to $\int_{\mathbb{R}^n} g(x) f(x)\, dx$.

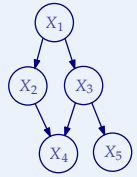
**8** Popular Metropolis algorithms:

- **Hamiltonian Monte Carlo** (HMC). Suitable for continuous variables and much faster than plain Metropolis-Hastings with a Gaussian proposal distribution. Requires the ability to differentiate the density with respect to the variables (often handled using autodiff).
- **No U-Turn Sampler** (NUTS). A common variant of HMC.
- **Particle Gibbs** (PG). Suitable for discrete variables.
- **Gibbs Sampler**. Gibbs sampling allows us to modify different variables using different samplers: we alternatingly hold one set of variables fixed while proposing a Metropolis update to the others, then hold the latter set fixed while proposing an update to the former set.

**9** One disadvantage of Bayesian statistics is the subjectivity of the prior distribution. On the other hand, when a meaningful prior is available, Bayesian statistics provides a natural way to combine that information with the observed data. Frequentist and Bayesian statistics both have strengths and weaknesses which can vary in importance depending on the details of the problem at hand.

## Statistical Learning: Graphical models

**1** A **Bayes net** is a random vector together with a directed acyclic graph (DAG) which models conditional dependence relationships among its components.



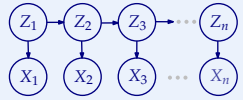

The random vector $\mathbf{X} = (X_1, \dots, X_5)$ and the graph shown make a Bayes net if the distribution of $\mathbf{X}$ factors as a product of conditional distributions as indicated by the graph connections; that is, if for all $(x_1, \dots, x_5)$, we have

$$\begin{aligned}\mathbb{P}(\mathbf{X} = (x_1, x_2, \dots, x_5)) = {} & \mathbb{P}(X_1 = x_1) \times \\ & \mathbb{P}(X_2 = x_2 \mid X_1 = x_1) \times \\ & \mathbb{P}(X_3 = x_3 \mid X_1 = x_1) \times \\ & \mathbb{P}(X_4 = x_4 \mid X_2 = x_2, X_3 = x_3) \times \\ & \mathbb{P}(X_5 = x_5 \mid X_3 = x_3)\end{aligned}$$

**2** In many Bayes net applications, only some of the random variables are observed. The others are called *hidden* or *latent* variables. This missing information presents inference challenges.

**3** A **Gaussian mixture model** (GMM) is a Bayes net consisting of a discrete random variable $Z$ and a random variable $X$ whose conditional distribution given each possible value of $Z$ is Gaussian.


**4** A **hidden Markov model** is a Bayes net consisting of a chain of random variables $Z_1, \dots, Z_n$ (called *hidden* variables), each of which is the parent of a single random variable $X_i$:




**5** Bayes net inference (drawing conclusions about the model based on observed data) can be carried out using a maximum likelihood technique called expectation-maximization (EM) or using Bayesian MCMC methods.

**6** **Expectation-Maximization** is an iterative procedure for parameter estimation in models with hidden variables: start with a random guess for the parameters and find the conditional distribution $\zeta$ of the hidden variables given the observed variables and the current parameter guess. We then treat as unknown the vector $\theta$ of all the model parameters, and we compute—with respect to the measure $\zeta$—the expected log likelihood function $Q(\theta)$. A new $\theta$ is chosen to maximize $Q$, and the two steps are iterated to convergence.

*Example. Consider a GMM with a $\{0,1\}$-valued $Z$: we have $\mathbb{P}(Z = 1) = \alpha$, and for each observation $i$ and each $j \in \{0,1\}$, the conditional distribution of $\mathbf{X}_i$ given $Z_i = j$ is normal with mean $\boldsymbol{\mu}_j$ and covariance $\Sigma_j$. All together, the parameter vector is $\theta = (\alpha, \boldsymbol{\mu}_0, \Sigma_0, \boldsymbol{\mu}_1, \Sigma_1)$. By Bayes' theorem, the conditional distribution of $Z_i$ given $\mathbf{X}_i = \mathbf{x}_i$ is Bernoulli with success probability*

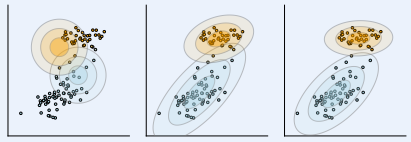
$$\pi_i = \frac{\alpha f_1(\mathbf{x}_i)}{\alpha f_1(\mathbf{x}_i) + (1-\alpha) f_0(\mathbf{x}_i)},$$

*where $f_j$ is the normal density with mean $\boldsymbol{\mu}_j$ and covariance $\Sigma_j$. Then we have*

$$\begin{aligned} Q(\theta) &= \mathbb{E}\left[\log \prod_{i=1}^n (z_i \alpha f_1(\mathbf{x}_i) + (1 - z_i)(1-\alpha) f_0(\mathbf{x}_i))\right] \\ &= \sum_{i=1}^n \pi_i [\log \alpha + \log f_1(\mathbf{x}_i)] \\ &\qquad + (1 - \pi_i)[\log(1-\alpha) + \log f_0(\mathbf{x}_i)]. \end{aligned}$$

*Optimizing, we get $\pi$-weighted counts, means, and covariance matrices for $\alpha$, $\boldsymbol{\mu}_1, \boldsymbol{\mu}_0, \Sigma_1$ and $\Sigma_0$.*

*In the EM iterations shown, membership probabilities $\pi_i$, based on current parameter estimates, are indicated by point color (E-step). These values are used as weights to update the means and covariances for the multivariate normal distributions (M-step).*



**7** A **Probabilistic Programming Language** (PPL) is a framework for describing stochastic models and performing automated Bayesian inference on them. Examples: **Stan** (a C++ library, callable from Julia/Python/R), **PyMC3**, and **Turing.jl**.

**8** A HMM example in Turing.jl (the object returned on the last line will contain estimates for the parameters):

```
using Turing
@model HMM(x) = begin
    n = length(x)
    z = tzeros(Int64, n) # hidden states
    p₁ ~ Uniform(0,1) # trans. prob. 1→1
    p₂ ~ Uniform(0,1) # trans. prob. 2→1
    P = [p₁ 1-p₁; p₂ 1-p₂] # transition matrix
    z[1] ~ Categorical([0.5,0.5]) # start 1 or 2
    x[1] ~ Normal(z[1],0.1)
    for i=2:n
        # choose next hidden state
        z[i] ~ Categorical(P[z[i-1],:])
        x[i] ~ Normal(z[i],0.1) # add noise
    end
end
# choose parameters for samplers
hmc = HMC(2, 0.001, 7, :p₁, :p₂)
pg = PG(20, 1, :z)
G = Gibbs(1000, hmc, pg)
# perform inference (assuming the vector x
# contains empirical observations)
sample(HMM(x), G)
```

## Statistics: dplyr and ggplot2

1 **dplyr** is an R package for manipulating data frames. The following functions filter rows, sort rows, select columns, add columns, group, and aggregate the columns of a grouped data frame.

```
flights %>%
  filter(month == 1, day < 5) %>%
  arrange(day, distance) %>%
  select(month, day, distance, air_time) %>%
  mutate(speed = distance / air_time * 60) %>%
  group_by(day) %>%
  summarise(avgspeed = mean(speed,na.rm=TRUE))
```

2 **ggplot2** is an R package for data visualization. Graphics are built as a sum of *layers*, which consist of a data frame, a **geom**, a **stat**, and a mapping from the data to the geom's **aesthetics** (like `x`, `y`, `color`, or `size`). The appearance of the plot can be customized with **scale**s, **coord**s, and **theme**s.

```
ggplot(data = mpg) +
    geom_smooth(mapping = aes(x = displ, y = hwy)) +
    geom_point(mapping = aes(x = displ, y = hwy,
                             color = class, alpha = cty))
```