

# Numerical Analysis in DROP

**v5.73** 24 September 2022

# Introduction

### Framework Glossary

1. Hyperspace Search: Hyperspace search is a search to determine whether the entity is inside the zone of a range, e.g., bracketing search.
2. Hyperpoint Search: Hyperpoint searches are hard searches that search for an exact specific point (to within an appropriately established tolerance).
3. Iterate Nodes: This is the set of the traveled nodes (variate/Objective Function ordered pairs) that contain the trajectory traveled.
4. Iteration Search Primitives: The set of variate iteration routines that generate the subsequent iterate nodes.
5. Compound iterator search scheme: Search schemes where the primitive iteration routine to be invoked at each iteration are evaluated.
6. RunMap: Map that holds the program state at the end of each iteration, in the generic case, this is composed of the Wengert iterate node list, along with the corresponding root finder state.
7. Cost of Primitive (cop): This is the cost of invocation of a single variate iterator primitive.

### Document Layout

1. Base Framework
2. Search Initialization
   1. Bracketing
   2. Objective Function Failure
   3. Bracketing Start Initialization
   4. Open Search Initialization
   5. Search/Bracketing Initializer Customization Heuristics
3. Numerical Challenges in Search
4. Variate Iteration
5. Open Search Methods
   1. Newton’s Method
6. Closed Search Methods
   1. Secant
   2. Bracketing Iterative Search
   3. Univariate Iterator Primitive
      1. Bisection
      2. False Position
      3. Inverse Quadratic
      4. Ridder’s
   4. Univariate Compound Iterator
      1. Brent’s Method
      2. Zheng’s Method
7. Polynomial Root Search
8. References
9. Figures
10. Fixed Point Search Software Components
    1. Execution Initialization
    2. Bracketing
    3. Execution Customization
    4. Fixed Point Search
    5. Variate Iteration
    6. Initialization Heuristics

### Framework

1. The root search given an objective function and its goal is achieved by iteratively evolving the variate, and involves the following steps:

* Search initialization and root reachability determination: Searched is kicked off by spawning a root variate iterator for the search initialization process (described in detail in the next section).
* Absolute Tolerance Determination.
* Root Search Iteration: The root is searched iteratively according to the following steps:

1. The iterator progressively reduces the bracket width.
2. Successive iteration occurs using either a single primitive (e.g., using the bisection primitive), or using a selector scheme that picks the primitives for each step (e.g., Brent’s method).
3. For Open Method, instead of 1 and 2, the routine drives towards convergence iteratively.

* Search Termination Detection: The search termination occurs typically based on the following:
* Proximity to the Objective Function Goal
* Convergence on the variate
* Exhaustion if the number of iterations

1. The flow behind these steps is illustrated in Figure 1.
2. The “Flow Control Variate” in root search is the “Objective Function Distance to Goal” Metric.

### Search Initialization

1. Broadly speaking, root finding approaches can be divided into a) those that bracket roots before they solve for them, and b) those that don’t need to bracket, opting instead to pick a suitable starting point.
2. Depending upon the whether the search is a bracketing or an open method, the search initialization does one the following:
   * Determine the root brackets for bracketing methods
   * Locate root convergence zone for open methods
3. Initialization begins by a search for the starting zone. A suitable starting point/zone is determined where, by an appropriate choice for the iterator, you are expected to reach the fixed-point target within a sufficient degree of reliability. Very general-purpose heuristics often help determine the search start zone.
4. Both bracketing and open initializers are hyperspace searches, since they search for something “IN”, not “AT”.

**Bracketing**

1. Bracketing is the process of localizing the fixed point to within a target zone with the least required number of Objective Function calculations. Steps are:

* Determine a valid bracketing search start
* Choose a suitable bracket expansion
* Limit attention to where the Objective Function is defined (more on this below).

1. Figure 2 shows the flow for the Bracketing routine.
2. Bracketing methods require that the initial search interval bracket the root (i.e. the function values at interval end points have opposite signs).
3. Bracketing traps the fixed point between two variate values, and uses the intermediate value theorem and the continuity of the Objective Function to guarantee the presence/existence of the fixed point between them.
4. Unless the objective function is discontinuous, bracketing methods guarantee convergence (although may not be within the specified iteration limit).
5. Typically, they do not require the objective function to be differentiable.
6. Bracketing iteration primitives’ convergence is usually linear to super-linear.
7. Bracketing methods preserve bracketing throughout computation and allow user to specify which side of the convergence interval to select as the root.
8. It is also possible to force a side selection after a root has been found, for example, in sequential search, to find the next root.
9. Generic root bracketing methods that treat the objective function as a black box will be slower than targeted ones – so much so that they can constitute the bulk of the time for root search. This is because, to accommodate generic robustness coupled with root-pathology avoidance (oscillating bracket pairs etc.), these methods have to perform a full variate space sweep without any assumptions regarding the location of the roots (despite this most, bracketing algorithms cannot guarantee isolation of root intervals). For instance, naïve examination of the Objective Function’s “sign-flips” alone can be misleading, especially if you bracket fixed-points with even numbered multiplicity within the brackets. Thus, some ways of analyzing the Black Box functions (or even the closed form Objective Functions) are needed to better target/customize the bracketing search (of course, parsimony in invoking the number of objective function calls is the main limitation).
10. Soft Bracketing Zone: One common scenario encountered during bracketing is the existence of a soft preferred bracketing zone, one edge of which serves as a “natural edge”. In this case, the bracketing run needs to be positioned to be able to seek out starting variate inside soft zone in the direction AWAY from the natural edge.
11. The first step is to determine a valid bracketing search start. One advantage with univariate root finding is that objective function range validity maybe established using an exhaustive variate scanner search without worrying about combinatorial explosion.

**Objective Function Failure**

1. Objective Function may fail evaluation at the specified variate for the following reason:

* + Objective Function is not defined at the specified variate.
  + Objective Function evaluates to a complex number.
  + Objective Function evaluation produces NaN/Infinity/Under-flow/Over-flow errors.
  + In such situations, the following steps are used to steer the variate to a valid zone.

1. Objective Function undefined at the Bracketing Candidate Variate: If the Objective Function is undefined at the starting variate, the starting variate is expanded using the variate space scanner algorithm described above. If the objective Function like what is seen in Figure 3, a valid starting variate will eventually be encountered.
2. Objective Function not defined at any of the Candidate Variates: The risk is that the situation in Figure 4 may be encountered, where the variate space scanner iterator “jumps over” the range over which the objective function is defined. This could be because the objective function may have become complex. In this case, remember that an even power of the objective function also has the same roots as the objective function itself. Thus, solving for an even power of the objective function (like the square) – or even bracketing for it – may help.

**Bracketing Start Initialization**

1. Figure 5 shows the flow behind a general-purpose bracket start locator.
2. Once the starting variate search is successful, and the objective function validity is range-bound, then use an algorithm like bisection to bracket the root (as shown in Figure 6 below).
3. However, if the objective function runs out of its validity range under the variate scanner scheme, the following steps need to be undertaken:

* If the left bracketing candidate fails, bracketing is done towards the right using the last known working left-most bracketing candidate as the “left edge”.
* Likewise, if the right bracketing candidate fails, bracketing is done towards the left using the last known working right-most bracketing candidate as the “right edge”.

1. The final step is to trim the variate zone. Using the variate space scanner algorithm, and the mapped variate/Objective Function evaluations, the tightest bracketing zones are extracted (Figure 7).

### Open Search Initialization

1. Non-bracketing methods use a suitable starting point to kick off the root search. As is obvious, the chosen starting point can be critical in determining the fate of the search. In particular, it should be within the zone of convergence of the fixed-point root to guarantee convergence. This means that specialized methods are necessary to determine zone of convergence.
2. When the objective function is differentiable, the non-bracketing root finder often may make use of that to achieve quadratic or higher speed of convergence. If the non-bracketing root finder cannot/does not use the objective function’s differentiability, convergence ends up being linear to super-linear.
3. The typical steps for determining the open method starting variate are:

* Find a variate that is proximal to the fixed point
* Verify that it satisfies the convergence heuristic

1. Bracketing followed by a choice of an appropriate primitive variate (such as bisection/secant) satisfies both, thus could be a good starting point for open method searches like Newton’s method.
2. Depending upon the structure of the Objective Function, in certain cases the chain rule can be invoked to ease the construction of the derivative – esp. in situations where the sensitivity to inputs are too high/low.

### Search/Bracketing Initializer Heuristic Customization

1. Specific Bracketing Control Parameters
2. Left/Right Soft Bracketing Start Hints: The other components may be used from the bracketing control parameters.
3. Mid Soft Bracketing Start Hint: The other components may be used from the bracketing control parameters.
4. Floor/Ceiling Hard Bracketing Edges: The other components may be used from the bracketing control parameters.
5. Left/Right Hard Search Boundaries: In this case, no bracketing is done – brackets are used to verify the roots, search then starts directly.

### Numerical Challenges in Search

1. Bit Cancellation
2. Ill-conditioning (e.g., see high order polynomial roots)
3. "domains of indeterminacy" – existence of sizeable intervals around which the objective function hovers near the target
4. Continuous, but abrupt changes (e.g., near-delta Gaussian objection function)
5. Under-flow/over-flow/round-off errors
6. root multiplicity (e.g., in the context of polynomial roots)
7. Typical solution is to transform the objective function to a better conditioned function – insight into the behavior of the objective can be used to devise targeted solutions.

**Variate Iteration**

where is the ith variate and is the root finder state after the ith iteration.

1. Iterate nodes as Wengert variables: Unrolling the traveled iterate nodes during the forward accumulation process, as a Wengert list, is a proxy to the execution time, and may assist in targeted pre-accumulation and check-pointing.

## Cognition Techniques of Mathematical Functions:

* + Wengert Variate Analysis => Collection of the Wengert variates helps build and consolidate the Objective Function behavior from the variate iterate Wengert nodes – to build a behavioral picture of the Objective Function.
  + Objective Function Neighborhood Behavior => With every Wengert variable, calculation of the set of forward sensitivities and the reverse Jacobians builds a local picture of the Objective Function without having to evaluate it.

1. Check pointing: Currently implemented using a roving variate/OF iterate node “RunMap”; this is also used to check circularity in the iteration process.
2. Compound Iterator RunMap: For compound iterations, the iteration circularity is determined the doublet , so the Wengert RunMap is really a doublet Multi-Map.
3. Hyperpoint univariate fixed-point search proximity criterion: For hyperpoint checks, the search termination check needs to explicitly accommodate a “proximity to target” metric. This may not be then case for hyperspace checks.
4. Regime crossover indicator: On one side the crossover, the variate is within the fast convergence zone, so you may use faster Open techniques like the Newton’s methods. On the other side, continue using the bracketing techniques.
   1. Fast side of the crossover must be customizable (including other Halley’s method variants); robust side should also be customizable (say False Position).
5. Crossover indicator determination: Need to develop targeted heuristics needed to determine the crossover indicator.
   * Entity that determines the crossover indicator may be determined from the relative variate shift change and the relative objective function change .
6. Types of bracketing primitives:

* Bracket narrower primitives (Bisection, false position), and interpolator primitives (Quadratic, Ridder).
* Primitive’s COP determinants: Expressed in terms of characteristic compute units.
  1. Number of objective function evaluation (generally expensive).
  2. Number of variate iterator steps needed.
  3. Number of objective function invocation per a given variate iteration step.
* Bracket narrower primitives => Un-informed iteration primitives, low invocation cost (usually single objective function evaluation), but low search targeting quality, and high COP.
* Interpolator primitives => Informed iteration primitives, higher invocation cost (multiple objective function evaluations, usually 2), better search targeting quality, and lower COP.

1. Pre-OF Evaluation Compound Heuristic: Heuristic compound variates are less informed, but rely heavily on heuristics to extract the subsequent iterator, i.e., pre-OF evaluation heuristics try to guide the evolution without invoking the expensive OF evaluations (e.g., Brent, Zheng).
2. OF Evaluation Compound Heuristic: These compound heuristics use the OF evaluations as part of the heuristics algorithm to establish the next variate => better informed

**Open Search Method: Newton’s Method**

1. Newton’s method uses the objective function and its derivative to iteratively evaluate the root.
2. Given a well-behaved function and its derivative defined over real , the algorithm starts with an initial guess of for the root.
3. First iteration yields
4. This is repeated in

till a value that is convergent enough is obtained.

1. If is a simple root (root with multiplicity ), and

and

respectively, then for sufficiently large , the convergence is quadratic:

1. Newton’s method only works when has continuous derivatives in the root neighborhood.
2. When analytical derivatives are hard to compute, calculate slope through nearby points, but convergence tends to be linear (like secant).
3. If the first derivative is not well behaved/does not exit/undefined in the neighborhood of a particular root, the method may overshoot, and diverge from that root.
4. If a stationary point of the function is encountered, the derivative is zero and the method will fail due to division by zero.
5. The stationary point can be encountered at the initial or any of the other iterative points.
6. Even if the derivative is small but not zero, the next iteration will be a far worse approximation.
7. A large error in the initial estimate can contribute to non-convergence of the algorithm (owing to the fact that the zone is outside of the neighborhood convergence zone).
8. If is a root with multiplicity

then for sufficiently large , the convergence becomes linear, i.e.,

1. When there are two or more roots that are close together then it may take many iterations before the iterates get close enough to one of them for the quadratic convergence to be apparent.
2. However, if the multiplicity of the root is known, one can use the following modified algorithm that preserves the quadratic convergence rate (equivalent to using successive over-relaxation)
3. The algorithm estimates after carrying out one or two iterations, and then use that value to increase the rate of convergence. Alternatively, the modified Newton’s method may also be used:
4. It is easy to show that if

and

the convergence in the neighborhood becomes linear. Further, if

and

convergence becomes cubic.

1. One way of determining the neighborhood of the root. Define

where is a fixed point of , i.e.

is a positive constant,

and

1. One sufficient condition for to initialize a convergent sequence , which converges to the root

of

is that

and that be chosen so that

1. It is easy to show that under specific choices for the starting variate, Newton’s method can fall into a basin of attraction. These are segments of the real number line such that within each region iteration from any point leads to one particular root - can be infinite in number and arbitrarily small. Also, the starting or the intermediate point can enter a cycle - the -cycle can be stable, or the behavior of the sequence can be very complex (forming a Newton fractal).
2. Newton's method for optimization is equivalent to iteratively maximizing a local quadratic approximation to the objective function. But some functions are not approximated well by quadratic, leading to slow convergence, and some have turning points where the curvature changes sign, leading to failure. Approaches to fix this use a more appropriate choice of local approximation than quadratic, based on the type of function we are optimizing. Next section demonstrates three such generalized Newton rules. Like Newton's method, they only involve the first two derivatives of the function, yet converge faster and fail less often.
3. One significant advantage of Newton’s method is that it can be readily generalized to higher dimensions.
4. Also, Newton’s method calculates the Jacobian automatically as part of the calibration process, owing to the reliance on derivatives – in particular, automatic differentiation techniques can be effectively put to use.

**Closed Search Methods**

**Secant**

1. Secant method results on the replacement of the derivative in the Newton’s method with a secant-based finite difference slope.
2. Convergence for the secant method is slower than the Newton’s method (approx. order is 1.6); however, the secant method does not require the objective function to be explicitly differentiable.
3. It also tends to be less robust than the popular bracketing methods.

**Bracketing Iterative Search**

1. Bracketing iterative root searches attempt to progressively narrow the brackets and to discover the root within.
2. The first set discusses the goal search univariate iterator primitives that are commonly used to iterate through the variate.
3. These goal search iterator primitives continue generating a new pair of iteration nodes (just like their bracketing search initialization counter-parts).
4. Certain iterator primitives carry bigger “local” cost, i.e., cost inside a single iteration, but may reduce global cost, e.g., by reducing the number iterations due to faster convergence.
5. Further, certain primitives tend to be inherently more robust, i.e., given enough iteration, they will find the root within – although they may not be fast.
6. Finally, the case of compound iterator search schemes, search schemes where the primitive iteration routine to be invoked at each iteration is evaluated on-the-fly, are discussed.
7. Iterative searches that maintain extended state across searches pay a price in terms of scalability – the price depending on the nature and the amount of state held (e.g., Brent’s method carries iteration selection state, whereas Zheng’s does not).

**Univariate Iterator Primitive: Bisection**

1. Bisection starts by determining a pair of root brackets and .
2. It iteratively calculates at

then uses to replace either or , depending on the sign. It eventually stops when has attained the desired tolerance.

1. Bisection relies on being continuous within the brackets.
2. While the method is simple to implement and reliable (it is a fallback for less reliable ones), the convergence is slow, producing a single bit of accuracy with each iteration.

**Univariate Iterator Primitive: False Position**

1. False position works the same as bisection, except that the evaluation point is linearly interpolated; is computed at

where and have opposite signs. This holds obvious similarities with the secant method.

1. False position method also requires that be continuous within the brackets.
2. It is simple enough, more robust than secant and faster than bisection, but convergence is still linear to super-linear.
3. Given that the linear interpolation of the false position method is a first-degree approximation of the objective function within the brackets, quadratic approximation using Lagrange interpolation may be attempted as

where we use the three iterates, , and , with their function values, , and .

1. This reduces the number of iterations at the expense of the function point calculations.
2. Using higher order polynomial fit for the objective function inside the bracket does not always produce roots faster or better, since it may result in spurious inflections (e.g., Runge’s phenomenon).
3. Further, quadratic or higher fits may also cause complex roots.

**Univariate Iterator Primitive: Inverse Quadratic**

1. Performing a fit of the inverse instead of avoids the quadratic interpolation problem above. Using the same symbols as above, the inverse can be computed as



1. Convergence is faster than secant, but poor when iterates not close to the root, e.g., if two of the function values , and coincide, the algorithm fails.

**Univariate iterator primitive: Ridder’s**

* 1. Ridders' method is a variant on the false position method that uses exponential function to successively approximate a root of .
  2. Given the bracketing variates, and , which are on two different sides of the root being sought, the method evaluates at
  3. It extracts exponential factor such that forms a straight line across , , and . A revised (named )is calculated from

1. Ridder’s method is simpler than Brent's method, and has been claimed to perform about the same.
2. However, the presence of the square root can render it unstable for many of the reasons discussed above.

**Univariate compound iterator: Brent and Zheng**

1. Brent’s predecessor method first combined bisection, secant, and inverse quadratic to produce the optimal root search for the next iteration.
2. Starting with the bracket points and , two provisional values for the next iterate are computed; the first given by the secant method

and the second by bisection

1. If lies between and , it becomes the next iterate , otherwise the is the next iterate.
2. Then, the value of the new contra-point is chosen such that and have opposite signs.
3. Finally, if

then is probably a better guess for the solution than , and hence the values of and are exchanged.

1. To improve convergence, Brent’s method requires that two inequalities must be simultaneously satisfied.
2. Given a specific numerical tolerance , if the previous step used the bisection method, and if

the bisection method is performed and its result used for the next iteration. If the previous step used interpolation, the check becomes

1. If the previous step used bisection, if

then secant is used; otherwise the bisection used for the next iteration. If the previous step performed interpolation

is checked instead.

1. Finally, since Brent's method uses inverse quadratic interpolation, has to lie between and .
2. Brent’s algorithm uses three points for the next inverse quadratic interpolation, or secant rule, based upon the criterion specified above.
3. One simplification to the Brent’s method adds one more evaluation for the function at the middle point before the interpolation.
4. This simplification reduces the times for the conditional evaluation and reduces the interval of convergence.
5. Convergence is better than Brent’s, and as fast and simple as Ridder’s.

### Polynomial Root Search

1. This section carries out a brief treatment of computing roots for polynomials.
2. While closed form solutions are available for polynomials up to degree 4, they may not be stable numerically.
3. Popular techniques such as Sturm’s theorem and Descartes’ rule of signs are used for locating and separating real roots.
4. Modern methods such as VCA and the more powerful VAS use these with Bisection/Newton methods – these methods are used in Maple/Mathematica.
5. Since the eigenvalues of the companion matrix to a polynomial correspond to the polynomial’s roots, common fast/robust methods used to find them may also be used.
6. A number of caveats apply specifically to polynomial root searches, e.g., Wilkinson’s polynomial shows why high precision is needed when computing the roots – proximal/other ill-conditioned behavior may occur.
7. Finally, special ways exist to identify/extract multiplicity in polynomial roots – they use the fact that and share the root, and by figuring out their GCD.

# Meta-heuristics

### Introduction

1. Definition: Meta-heuristic is a higher-level procedure or heuristic designed to find, generate, or select a lower level procedure or heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity (Bianchi, Dorigo, Gambardella, and Gutjahr (2009)).
2. Applicability: Meta-heuristics techniques make only a few assumptions about the optimization problem being addressed, so are usable across a variety of problem (Blum and Roli (2003)).
3. Underpinning Philosophy: Many kinds of meta-heuristics implement some kind of stochastic optimization, so the solution depends upon the random variables being generated. As such it does not guarantee that a globally optimal solution can be found over some class of problems.
4. Search Strategy: By searching over a large set of feasible solutions meta-heuristics often finds good solutions with less computation effort than other algorithms, iterative methods, or simple heuristics (see Glover and Kochenberger (2003), Goldberg (2003), Talbi (2009)).
5. Literature: While theoretical results are available (typically on convergence and the possibility of locating global optimum, see Blum and Roli (2003)), most results on meta-heuristics are experimental, describing empirical results based on computer experiments with the algorithms.
   * While high quality research exists (e.g., Sorensen (2013)), enormously numerous meta-heuristics algorithms published as novel/practical have been of flawed quality – often arising out of vagueness, lack of conceptual elaboration, and ignorance of previous literature (Meta-heuristics (Wiki)).

### Properties and Classification

1. Properties: This comes from Blum and Roli (2003):
   1. Meta-heuristics are strategies that guide the search process.
   2. The goal is to efficiently explore the search space in order to find near-optimal solutions.
   3. Techniques that constitute meta-heuristics range from simple local search procedures to complex learning processes.
   4. Meta-heuristic algorithms are approximate and usually non-deterministic.
   5. Meta-heuristics are not problem-specific.
2. Classification: These are taken from Blum and Roli (2003) and from Bianchi, Dorigo, Gambardella, and Gutjahr (2009):
   1. Classification based on the type of the search strategy
   2. Classification off-of single solution search vs. population-based searches
   3. Classification off-of hybrid/parallel heuristics

**Meta-heuristics Techniques**

1. Simple Local Search Improvements: In this family of techniques, the search strategy employed is an improvement on simple local search algorithms; examples include simulated annealing, tabu search, iterated local search, variable neighborhood search, and GRASP (Blum and Roli (2003)).
2. Search Improvements with Learning Strategies: The other type of search strategy has a learning component to the search; meta-heuristics of this type include ant colony optimization, evolutionary computation, and genetic algorithms (Blum and Roli (2003)).
3. Single Solution Searches: These focus on modifying and improving a single candidate solution; single solution meta-heuristics include iterated local search, simulated annealing, variable neighborhood search, and tabu search (Talbi (2009)).
4. Population based Searches: Population based searches maintain and improve multiple candidate solutions using population characteristics to guide the search. These meta-heuristics include evolutionary computation, genetic algorithms, and particle swarm optimization (Talbi (2009)).
5. Swarm Intelligence: Swarm intelligence is the collective behavior of de-centralized, self-organized agents in a particle or a swarm. Ant colony optimization (Dorigo (1992)), particle swarm optimization (Talbi (2009)), artificial bee colony (Karaboga (2010)) are all example algorithms of swarm intelligence.
6. Hybrid meta-heuristic: These combine meta-heuristics with other optimization approaches (these could come from e.g., mathematical programming, constraint programming, machine learning etc.). Components of the hybrid meta-heuristic run concurrently and exchange information to guide the search.
7. Parallel meta-heuristic: This employs parallel programming techniques to run multiple meta-heuristics searches in parallel; these may range from simple distributed schemes to concurrent search runs that interact to improve the overall solution.

**Meta-heuristics Techniques in Combinatorial Problems**

1. Combinatorial Optimization Problems: In combinatorial optimization, an optimal solution is sought over a discrete search space. Typically, the search-space of the candidate solution goes faster than exponentially as the problem-size increases, making an exhaustive search for the optimal solution infeasible (e.g., the Travelling Salesman Problem (TSP)).
2. Nature and Types of Combinatorial Problems: Multi-dimensional combinatorial problems (e.g., engineering design problems such as form-finding and behavior-finding (Tomoiaga, Chandris, Sumper, Sudria-Andrieu, and Villafafila-Robles (2013)) suffer from the usual curse of dimensionality, making them infeasible to analytical or exhaustive-search methods.
3. Meta-heuristics applied to Combinatorial Optimization: Popular meta-heuristic algorithms for combinatorial problems include genetic algorithms (Holland (1975)), scatter search (Glover (1977)), simulated annealing (Kirkpatrick, Gelatt, and Vecchi (1983)), and tabu search (Glover (1986)).

**Key Meta-heuristics Historical Milestones**

1. Contributions: Many different meta-heuristics are in existence, and new variants are being continually developed. The following key milestone contributions have been extracted from Meta-heuristics (Wiki).
2. 1950s:
   1. Robbins and Munro (1951) work on stochastic optimization methods.
   2. Barricelli (1954) carries out the first simulation of the evolutionary process and uses it on general optimization problems.
3. 1960s:
   1. Rastrigin (1963) proposes random search.
   2. Matyas (1965) proposes random optimization.
   3. Nelder and Mead (1965) propose a simplex heuristic, which later shown to converge to non-stationary points on some problems.
   4. Fogel, Owens, and Walsh (1966) propose evolutionary programming.
4. 1970s:
   1. Hastings (1970) proposes the Metropolis-Hastings algorithm.
   2. Cavicchio (1970) proposes adaptation of the control parameters for an optimizer.
   3. Kernighan and Lin (1970) propose a graph-partitioning method that is related to variable-depth search and prohibition based tabu search.
   4. Holland (1975) proposes genetic algorithm.
   5. Glover (1977) proposes scatter search.
   6. Mercer and Sampson (1978) propose a meta-plan for tuning the optimizer’s parameters by using another optimizer.
5. 1980s:
   1. Smith (1980) describes genetic programming.
   2. Kirkpatrick, Gelatt, and Vecchi (1983) propose simulated annealing.
   3. Glover (1986) proposes tabu search, along with the first mention of the word meta-heuristic (Yang (2011)).
   4. Moscato (1989) proposes memetic algorithms.
6. 1990s:
   1. Dorigo (1992) introduces ant colony optimization in his PhD thesis.
   2. Wolpert and MacReady (1995) prove the no free lunch theorems (these are later extended by Droste, Jansen, and Wegener (2002), Igel and Toussaint (2003), and Auger and Teytaud (2010)).

# References

* Auger, A., and O. Teytaud (2010): Continuous Lunches are Free Plus the Design of Optimal Optimization Algorithms *Algorithmica* **57 (1)** 121-146
* Barricelli, N. A (1954): Esempi Numerici di Processi di Evoluzione *Methodos* 45-68
* Bianchi, L., M. Dorigo, L. M. Gambardella, and W. J. Gutjahr (2009): A Survey on Metaheuristics for Stochastic Combinatorial Optimization *Natural Computing: An International Journal* **8 (2)** 239-287
* Blum, C., and A. Roli (2003): Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison *ACM Computing Surveys* **35 (3)** 268-308
* Cavicchio, D. J. (1970): *Adaptive Search using Simulated Evolution* Technical Report **Computer and Communication Sciences Department, University of Michigan**
* Dorigo, M (1992): *Optimization, Learning, and Natural Algorithms* PhD Thesis **Politecnico di Milano**
* Droste, S., T. Jansen, and I. Wegener (2002): Optimization with Randomized Search Heuristics – The (A)NFL Theorem, Realistic Scenarios, and Difficult Functions *Theoretical Computer Science* **287 (1)** 131-144
* Fogel, L., A. J. Owens, and M. J. Walsh (1966): *Artificial Intelligence Through Simulated Evolution* **Wiley**
* Glover, F. (1977): Heuristics for Integer Programming using Surrogate Constraints *Decision Sciences* **8 (1)** 156-166
* Glover, F. (1986): Future Paths for Integer Programming and Links to Artificial Intelligence *Computers and Operations Research* **13 (5)** 533-549
* Glover, F., and G. A. Kochenberger (2003): *Handbook of Metaheuristics* **57 Springer International Series in Operations Research and Management Science**.
* Goldberg, D. E. (1989): *Genetic Algorithms in Search, Optimization, and Machine Learning* **Kluwer Academic Publishers**
* Hastings, W. K. (1970): Monte Carlo Sampling Methods using Markov Chains and their Applications *Biometrika* **57 (1)** 97-109
* Holland, J. H. (1975): *Adaptation in Natural and Artificial Systems* **University of Michigan Press**
* Igel, C., and M. Toussaint (2003): On Classes of Functions for which the No Free Lunch Results hold *Information Processing Letters* **86 (6)** 317-321
* Karaboga, D. (2010): Artificial Bee Colony Algorithm *Scholarpedia* **5 (3)** 6915
* Kernighan, B. W., and S. Lin (1970): An Efficient Heuristic Procedure for Partitioning Graphs *Bell System Technical Journal* **49 (2)** 291-307
* Kirkpatrick, S., C. D. Gelatt Jr., M. P. Vecchi (1983): Optimization by Simulated Annealing *Science* **220 (4598)** 671-680
* Matyas, J. (1965): Random Optimization *Automation and Remote Control* **26 (2)** 246-253
* Mercer, R. E., and J. R. Sampson (1978): Adaptive Search using a Reproductive Meta-plan *Kybernetes* **7 (3)** 215-228
* Moscato, P. (1989): *On Evolution, Search, Optimization, Genetic Algorithms, and Martial Arts: Towards Memetic Algorithms* Report 826 **Caltech Concurrent Computation Program**
* Nelder, J. A., and R. Mead (1965): A Simplex Method for Function Minimization *Computer Journal* **7** 308-313
* Rastrigin, L. A. (1963): The Convergence of Random Search Method in the Extremal Control of a many Parameter System *Automation and Remote Control* **24 (10)** 1337-1342
* Robbins, S., and H. Munro (1951): A Stochastic Approximation Method *Annals of Mathematical Statistics* **22 (3)** 400-407
* Smith, S. F. (1980): *A Learning System based on Genetic Adaptive Algorithms* PhD Thesis **University of Pittsburgh**
* Sorensen, K. (2013): [*Metaheuristics—the metaphor exposed*](http://onlinelibrary.wiley.com/doi/10.1111/itor.12001/abstract)
* Talbi, E. G. (2009): *Metaheuristics: From Design to Implementation* **Wiley**
* Tomoiaga, B., M. Chandris, A. Sumper, A. Sudria-Andrieu, and R. Villafafila-Robles (2013): Pareto Optimal Reconfiguration of Power Distribution Systems using a Genetic Algorithm based on NSGA-II *Energies* **6 (3)** 1439-1455
* Wolpert, D. H., and W. G. MacReady (1995): *No Free Lunch Theorems for Search* Technical Report SFI-TR-95-02-010 **Santa Fe Institute**
* Yang, X. S. (2011): Metaheuristic Optimization *Scholarpedia* **6 (8)** 11472

**Multi-variate Analysis**

1. Mean/Variance Location Dependence: The mean is sensitive to both translation and rotation, unless the distribution is mean centered. The variance along a given fixed direction, however, is not sensitive to rotation of the basis.

* This also implies that the maximal/minimal variances are invariant to representational basis changes. They are, however, sensitive to scaling, though, as PCA itself is.

1. Dimensional Independence vs. Dimensional Realization Independence: Dimensions are distinct (pressure, temperature etc.), but the realizations in those dimensions need not be. Therefore, correlated unit vectors only apply to actual realizations (and they are NOT scale invariant).
2. Orthogonal Data Set in the Native Basis Representation: If a data set is orthogonal under a given basis, then

(See Figure 1).

1. Non-orthogonal Data Set in the Native Basis Representation: In this case, the native representation basis results in

(See Figure 2). Thus, an orthogonalization operation needs to be performed such that under the new basis

(See Figure 3).

1. Orthogonalization as Principal Components Extraction: As can be seen from figures 2 and 3, under the new schematic axes

Further these correspond to the principal components, i.e., orthogonal components where the variance is an extremum.

1. Orthogonalized Representation: Upshot of all these is that, if the representation basis is structured such that

for all

then these representations automatically correspond to principal components as well.

1. Full Rank Matrix: This is simply an alternate term for multi-collinear matrix.

## Parallels between Vector Calculus and Statistical Distribution Analysis

1. DOT PRODUCT => The notion of dot product is analogous to covariance operation in statistical analysis, i.e., DOT PRODUCT is

if

and covariance is

if and are orthogonal to each other.

1. Distance Metric => Vector Euclidean distance is is equivalent to the variance . Further, extremizing the Euclidean/Frobenius distance is analogous to variance minimization/maximization techniques.

## Linear Systems Analysis and Transformation

## Matrix Transforms

1. Co-ordinate Rotation and Translation:

where is the rotating transformer, and is the translator.

1. Scaling vs. Rotation: Say

If

and

it becomes pure scaling.

and

produces differential elongation/compression and

or

results in rotation.

1. Uses of Gaussian Elimination:

* Linearization
* Orthogonalization
* Inversion
* Diagonalization
* Lower/Upper Triangle Decomposition (LU Decomposition)
* Independent Component Extraction
* Principal Component Analysis

1. Diagonal Identity Matrix Conception: Given a matrix what matrix should it be transformed by to get a diagonal identity matrix, i.e.

Answer is

Thus, diagonalization is also an inversion operation.

Diagonalization, of course, is unique only to a diagonal entry, whereas inversion corresponds to a specific choice of the diagonal entry.

## Systems of Linear Equations

1. Importance of Diagonal Dominance in Gauss-Seidel: Is diagonal dominance important because the dominant diagonal’s contribution to the RHS drives the given equation’s value, and therefore the iterative accuracy?
2. Eigenization Square Matrix Inversion Conceptualization: Given a source matrix

and an initialized target inverse identity matrix

achieve a suitable set of transformations simultaneously on and to eventually make

so that the corresponding

becomes the inverse.

1. Valid Inversion Rules: These are fairly straight forward application of the Gaussian elimination scheme:

* Scale a single row by a constant => scale the corresponding row by the same constant.
* Add/subtract any pair of rows from each other => add/subtract the same pair of rows from each other.

1. Matrix Inversion using Gaussian Elimination:
   * Scan across the diagonal entries.
   * Scan through the rows corresponding to each diagonal entry.
   * If a given row entry call value is zero, or its index corresponds to that of a diagonal, skip.
   * Calculate the as
   * Scan all the cells in the column of the current cell.
   * Apply the product to each entry in the current working column of the source matrix.
   * Do the same as above to the inverse matrix.
   * Subtract the entries corresponding to the designated diagonal column from the working column to make the current entry zero.
   * Do the same as above to the inverse matrix as well.
   * After the completion of all such diagonal scans, row scans, and working column scans, re-scan the diagonal again.
   * Scale down each entry of the source matrix by itself, so that the source matrix entries now constitute an identity () matrix.
   * Do the same as above to the inverse matrix as well.
2. Non-invertible Coefficient Matrix, but Solution exists: For unprocessed coefficient matrices, certain conditions (such as zero diagonal entries) may cause the coefficient matrix to be technically non-invertible, but that does not automatically mean that the system is unsolvable – a simply re-casting of the basic linear system set may be all that is required.
3. Linear Basis Re-arrangement: Sometimes, a singular coefficient matrix (with zero determinant, therefore non-invertible) may be re-arranged to create an invertible coefficient matrix. After inversion, it can be re-structured again to extract the inverse (which is just a coefficient Jacobian).
4. Rows/Columns as “Preferred Linear Basis Sequence” for Matrix Manipulation: Consider the solution to

where and are columns. In this case, the notion of constraint linear representation is maintained exclusively in rows. Therefore, all elimination/scaling basis operations need to be applied on that basis. In considering the solution to

where and are rows, columns now become the preferred linear basis sequence.

1. Regularization before Gauss Elimination: Before the Gauss Elimination can process, we need to diagonalize the matrix. Row swapping is a more robust way to diagonalize than row accumulation due to a couple of reasons:
   * Matrix Row Swap vs. Row Cumulate => In all cases, row swap can be transformed into row accumulation. When inverting, however, the row swapping AND accumulation should both be done on both the SOURCE and the TARGET matrices.
   * From a core linear operation set point-of-view, row accumulation is the inverse of the eventual of Gauss Elimination, so the danger is that the diagonalization gains of row swap may be undone during the intermediate stages of Gauss Elimination.
   * Even more important is that row swapping simply retains the original information, by just re-arranging the row set.
2. Row Swapping Caveats:

* Always swap rows by retaining the directionality of the scan AND by retaining the scan initial node preceding the swap (to choose the pivot). One way to do this is by starting the scan at - or from

if the edge has been reached - AND always keeping the scan sequence forward/backward.

* Also ensure that the target swapped row is “valid”, i.e., it’s post-swapped diagonal entry should be non-zero. If this cannot be achieved through the scan, then that is an error condition.

1. Diagonalization/Inversion Algorithm: Work in terms of an intermediate transform variate produced by re-arranging the original coefficient matrix and such that the in

is now invertible. The following would be the steps:

* + First, re-arrange the equation system set to identify a suitable pair and such that is invertible, and is estimated from
  + The re-arranging linear operation set will produce such that

## More General Inverse Transformation Re-formulation: Given the coefficient matrix , the set of unknown variables , and the RHS , , , and the corresponding and are given as

## and

## On transformation (i.e., adding row to row ), you get

## and therefore

## along with

## This clearly implies that

## or

## Orthogonalization

1. 2D Orthogonalization: In 2D, you need to fix one 1D orthogonal axis to be able to orthogonalize the other.
2. 2D Equation System:

and

This has 4 unknowns, so one solution for this is as follows: Fix

and

This results in 2 unknowns. Setting

and

you get

and

resulting in

and

Thus, all unknowns are determined.

1. nD Orthogonalization:

* Number of diagonal entries =>
* Number of non-diagonal entries =>
* Net Number of equations => Number of diagonal entries + Number of non-diagonal entries =>

1. nD Unknowns Analysis:

* Number of Unknowns => .
* Fix the first row, and the number of unknowns becomes => .
* Fixing the row takes off one equation, so the number of equations =>
* Number of equations = Number of Unknowns =>

results in

## Gaussian Elimination

1. n-D Gaussian Elimination: What does it work? If you fix a row, you can rotate the other rows to eliminate dependence on an ordinate of the fixed row.
2. Row Fixation as a Basis Choice: Row elimination does not automatically make the matrix diagonal or orthogonalize it – all it does is to eliminate dependence on a stochastic variate.
3. Number of Elimination Rotations: The first row fixation results in rotations, the second row fixation results in rotations, and so on. Thus, the total number of rotating transformations is (i.e., the same as the number of unknowns seen earlier). The result of these transformations is a lower/upper triangular matrix.
4. Final Reversing Sweep: An additional reverse sweep would eliminate similar column dependencies as well – resulting in another rotation choices.
5. Number of Sweep Operations: Total result of all the rotations and their corresponding choices =>

**Rayleigh Quotient Iteration**

**Introduction**

1. Idea behind Rayleigh Quotient Iteration: **Rayleigh Quotient Iteration** is an eigenvalue algorithm that extends the idea of inverse iteration by using the Rayleigh Quotient to obtain increasingly accurate eigenvalue estimates (Wiki - Rayleigh Quotient Iteration (2018)).
2. Progressive Navigation towards “True” Selection: Rayleigh Quotient Iteration is an iterative method, i.e., it delivers a sequence of approximate solutions that converge to a true solution in the limit. This is true for all algorithms that compute eigenvalues; since the eigenvalues can be irrational numbers, there can be no general method for computing them in a finite number of steps.
3. Practicality of the Iterative Approach: Very rapid convergence is guaranteed and no more than a few iterations are needed in practice to obtain a reasonable solution.
4. Cubic Convergence for Typical Matrices: The Rayleigh Quotient algorithm converges cubically for Hermitian or symmetric matrices, given an initial vector that is sufficiently close to an eigenvector of the matrix that is being analyzed.

**The Algorithm**

1. Update Eigenvalue using Rayleigh Quotient: The algorithm is very similar to inverse iteration, but replaces the estimated eigenvalues at the end of each iteration with the Rayleigh Quotient.
2. Initial Guess for Eigenvalue/Eigenvector: Begin by choosing a value as an initial eigenvalue guess for the Hermitian matrix . An initial vector must also be supplied as the initial eigenvector guess.
3. Iterative Eigenvalue and Eigenvector: Calculate the subsequent approximation of the eigenvector by

where is the identity matrix, and set the next approximation of the eigenvalue to the Rayleigh quotient of the current iteration equal to

1. Deflation Techniques for Successive Eigenvalues: To compute more than one eigenvalue the algorithm can be combined with a deflation technique.
2. Treatment for Very Small Matrices: Note that for very small matrices it is beneficial to replace the matrix inverse with the adjugate, which will yield the same iteration because it is equal to the inverse to an irrelevant scale – specifically, the inverse of the determinant.
3. Advantages of using the Adjugate: The adjugate is easier to compute explicitly than the inverse – though the inverse is easier to apply to a vector for problems that aren’t small – and is more numerically sound because it remains well-defined as the eigenvalue changes.

**References**

* [Wikipedia – Rayleigh Quotient Iteration (2018)](https://en.wikipedia.org/wiki/Rayleigh_quotient_iteration)

**Power Iteration**

**Introduction**

1. Power Iteration – Problem Statement/Definition: In mathematics, **power iteration** – also known as the *power method* – is an eigenvalue algorithm. Given a diagonalizable matrix , the algorithm will produce a number , which is the greatest – in absolute value – eigenvalue of , and a non-zero vector , the corresponding eigenvector of , such that

The algorithm is also known as the von Mises iteration (von Mises and Pollazek-Geiringer (1929)).

1. Characteristics of the Power Iteration Algorithm: Power iteration is a very simple algorithm, but it may converge slowly. It does not compute a matrix decomposition, and hence can be used when is a very large sparse matrix.

**The Method**

1. Starting Choice for Principal Eigenvector: The power iteration method starts with a vector , which may be either an approximation to the dominant eigenvector or a random factor.
2. Recurrence Relation for Power Iteration: The method is described by the recurrence relation

So, at every iteration, the vector is mulplied by the matrix and normalized.

1. Necessary Condition for Eigen-component Convergence: If one assumes that has an eigenvalue that is greater in magnitude than its other eigenvalues and that the starting vector has a non-zero component in the direction of an eigenvector associated with the dominant eigenvalue, then a sub-sequence converges to an eigenvector associated with the dominant eigenvector.
2. Recast of using Phase Shift: Without the two assumptions above the sequence does not necessarily converge. In this sequence

where is the eigenvector associated with the dominant eigenvalue, and

The presence of the term implies that does not converge unless

1. Convergence to the Dominant Eigenvalue: Under the two assumptions listed above, the sequence defined by

converges to the dominant eigenvalue.

1. Eigenvector and Eigenvalue Sequences: The vector is the associated eigenvector. Ideally one should use the Rayleigh Quotient in order to get the associated eigenvalue.
2. Spectral Radius using the Rayleigh Quotient: The method can also be used to calculate the spectral radius – the largest eigenvalue of a matrix – by computing the Rayleigh Quotient

**Analysis**

1. Decomposition to Jordan Canonical Form: Let be decomposed into its Jordan Canonical Form

where the first column of is an eigenvector of corresponding to the dominant eigenvalue .

1. The Principal Component Jordan Block: Since the dominant eigenvalue of is unique, the first Jordan block of is the matrix , where is the largest eigenvalue of in magnitude.
2. Initializing the Principal Component Eigenvector: The starting vector can be written as a linear combination of the columns of :

By assumption, has a non-zero component in the direction of the dominant eigenvalue, so

1. kth Recurrence of the Initial Vector: The computationally recurrence relation for can be written as:

where the expression is amenable to the analysis below.

1. Principal Component Dependence of :
2. Scaling Down the Principal Eigenvector: As

the expression above simplifies to

1. Limit of as : The limit follows from the fact that the eigenvalue of is less than in magnitude, so

as

1. Expansion of Higher Order Eigen Terms: It follows that

as

1. Reduction of for : Using this fact, can be written in a form that emphasizes its relationship with when is large:

where

and

as

1. Uniqueness of the Sequence: The sequence is bounded, so it contains a convergent sub-sequence. Note that the eigenvector corresponding to the dominant eigenvalue is unique only upto a scaler, so although the sequence may not converge, is nearly an eigenvector of for large .
2. Alternative Proof of the Sequence Convergence: Alternatively, if is diagonalizable, then the following proof yields the same result. Let be the eigenvalues – counted with multiplicity – of , and let be the corresponding eigenvectors. Suppose that is the dominant eigenvalue, so that

for

1. Suitable Choice for the Initial Vector: As seen earlier, the initial vector is written as

If is chosen randomly – with uniform probability – the

with probability

1. Power Iteration over Initial Eigenvector: Now

as long as

for

1. Convergence to the Principal Eigenvector: On the other hand,

Therefore converges to a multiple of the eigenvector

1. Convergence Ratio of Power Iteration: The convergence ratio is geometric with the ratio where denotes the second principal eigenvalue.
2. Consequence of Close Principal Eigenvectors: Thus, the method converges slowly if there is an eigenvalue close in magnitude to the dominant eigenvalue.

**Applications**

1. Identification of the Dominant Eigenvector/Eigenvalue: Although the power iteration method approximates only one eigenvalue of a matrix, it remains useful for several computational problems.
2. Use in Google and Twitter: For instance, Google uses it to calculate the PageRank of documents in their search engine (Ipsen and Wills (2005)), and Twitter uses it to show users recommendations of who to follow (Gupta, Goel, Lin, Sharma, Wang, and Zadeh (2013)).
3. Space Advantage of Power Iteration: The power iteration is especially suitable for sparse matrices, such as the web matrix, or as the matrix-free method that does not require storing the coefficient matrix explicitly, but can instead access a function evaluating matrix-vector products .
4. Power Iteration vs. Arnoldi Method: For non-symmetric matrices that are well-conditioned, the power iteration method can outperform the more complex Arnoldi method.
5. Power Iteration vs. Lanczos/LOBPCG: For symmetric matrices, the power iteration method is rarely used, since its convergence speed can be easily increased without sacrificing the smaller cost per iteration, e.g., Lanczos iteration and LOBPCG.
6. Variation in the Method - Inverse Iteration: Some of the more advanced algorithms can be understood as variations of the power iteration method. For instance, the inverse iteration method applies power iteration to the matrix .
7. Sub-space of Eigenvalues - Krylov Subspace: Other algorithms look at the whole subspace generated by the vectors . This subspace is known as the Krylov subspace. It can be computed by Arnoldi iteration or Lanczos iteration.

**References**

* Gupta, P., A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh (2013): [WTF – The Who-To-Follow Service at Twitter](https://dl.acm.org/citation.cfm?id=2488433)
* Ipsen, I., and R. Wills (2005): [Analysis and Computation of Google’s PageRank](http://www4.ncsu.edu/~ipsen/ps/slides_householderXVI.pdf)
* Von Mises, R., and H. Pollaczek-Geiringer (1929): Praktische Verfahren der Gleichingsauflosung *Zeitschrift fur Andewandte Mathematik und Mechanik* **9** 152-164

**Sylvester’s Formula**

**Overview**

1. Analytical Expression of Matrix Function: *Sylvester’s Formula*, *Sylvester’s Matrix Theorem*, or *Lagrange-Sylvester interpolation* expresses an analytic function of a matrix as a polynomial in , in terms of the eigenvectors and eigenvalues of (Claerbout (1985), Horn and Johnson (1991), Wikipedia (2019))).
2. In Terms of Eigenvalues and Eigenvectors: It states that (Sylvester (1883))

where the are the eigenvalues of , and the matrices

are the corresponding Frobenius covariants of , which are the projection matrix Lagrange polynomials of .

**Conditions**

Sylvester’s formula applies for any diagonalizable matrix with distinct eigenvalues and any function defined on some subset of complex numbers such that is well defined. The last condition means that every eigenvalue is in the domain of , and that every eigenvalue with multiplicity

is in the interior of the domain with being times differentiable at (Horn and Johnson (1991)).

**Generalization**

1. Buchheim Extension Based on Hermite Polynomials: Sylvester’s formula is only valid for diagonalizable matrices; an extension due to Buchheim (1884), based on Hermite interpolating polynomials, covers the general case:

where

1. Concise Expression of Schwerdtfeger: A further concise form is given by Schwerdtfeger (1938) as

**References**

* Buchheim, A. (1884): On the Theory of Matrices *Proceedings of the London Mathematical Society* **1-16 (1)** 63-82
* Claerbout, J. F. (1985): *Fundamentals of Geophysical Data Processing* **Blackwell Scientific Publishing**
* Horn, R. A., and C. R. Johnson (1985): *Topics in Matrix Analysis* **Cambridge University Press**
* Sylvester, J. J. (1883): On the Equation to the Secular Inequalities in the Planetary Theory *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **16 (100)** 267-269
* Wikipedia (2019): [Sylvester's Formula](https://en.wikipedia.org/wiki/Sylvester%27s_formula)

**Numerical Integration**

**Introduction and Overview**

1. Numerical Estimate of Definite Integrals: In numerical analysis, **numerical integration** comprises a broad family of algorithms for calculating the numerical value of a definite integral, and by extension, the term is also used sometimes to describe the numerical solution to differential equations.
2. Chapter Focus on Definite Integrals: This chapter focuses on the calculation of definite integrals.
3. Numerical Quadrature: One-Dimensional Integrals: The term **numerical quadrature** – often abbreviated to *quadrature* – is more or less a synonym for **numerical integration**, especially as applied to one-dimensional integrals (Wikipedia (2019)).
4. Curbature: Multi-Dimensional Numerical Integrals: Some authors refer to numerical integration over more than one dimension as *curbature*; others take *quadrature* to include higher dimensional integration.
5. Mathematical Specification of the Definite Integral: The basic problem in numerical integration is to compute an approximate solution to a definite integral to a given degree of accuracy.
6. Accurately Estimating the Definite Integral: If is a smooth function integrated over a small number of dimensions, and the domain of the integration is bounded, there are many methods for approximating the integral to the desired precision.

**Reasons for Numerical Integration**

1. Discrete Sampling of the Integrand: There are several reasons for carrying out numerical integration. For instance, the integrand may only be known at certain points, such as those obtained by sampling. Some embedded systems and other software applications may need numerical integration for this reason.
2. Anti-derivative not an Elementary Function: An expression for the integrand may be known, but it may be difficult or impossible to find an anti-derivative that is an elementary function. An example of such an integrand is

the anti-derivative of which – the error function times a constant – cannot be written in an elementary form.

1. Series Approximation of the Anti-derivative: It may be possible to find an anti-derivative symbolically, but it may be easier to compute a numerical approximation than to compute the anti-derivative. That may be the case if the anti-derivative is computed as an infinite derivative or product, or it its evaluation requires a special function that is not available.

**Methods for One-Dimensional Integrals**

1. Combining Evaluations of the Integrand: Numerical integration methods can generally be described as combining the evaluations of the integrand to get an approximation to the integral.
2. Weighted Sum at the Integration Points: The integrand is evaluated at a finite set of points called the *integration points* and a weighted sum of these values is used to approximate the integral. The integration points and the weights depend on the specified method used and the accuracy required from the approximation.
3. Approximation Error of the Method: An important part of the analysis of any numerical integration method is to study the behavior of the approximation error as a function of the number of integrand evaluations. A method that yields a small error for a small number of evaluations widely considered superior.
4. Reduced Number of Integrand Evaluations: Reducing the number of operations of the integrand reduces the number of arithmetic operations involved, and therefore reduces the total round-off error. Also, each evaluation takes time, and the integrand may be arbitrarily complicated.
5. Conditions for Brute Force Integration: A brute force kind of numerical integration can be done if the integrand is reasonably well-behaved, i.e., it is piece-wise continuous and of bounded variation, by evaluating the integrand with very small increments.

**Quadrature Rules Based on Interpolating Functions**

1. Error of Integrating Interpolation Polynomials: A large class of quadrature rules can be derived by constructing interpolating functions that are easy to integrate. Typically, these interpolating functions are polynomials. In practice, some of these polynomials of very high degree tend to oscillate very wildly, only polynomials of low degree are used, typically linear and quadratic.
2. Mid-Point Rule: Zero Degree Polynomial: The simplest function of this type is to let the interpolating function be a constant function – a polynomial of degree – that passes through the point This is the *mid-point rule* of the *rectangle rule*:
3. Trapezoidal Rule: First Degree Polynomial: The interpolating function may be a straight line – an affine function, i.e., a polynomial of degree one – passing through the points and . This is called the *trapezoidal rule*:
4. Sub division of the Integration Intervals: For either one of these rules, a more accurate representation can be made by breaking up the interval into some number of sub-intervals, computing an approximation for each sub-interval, then adding up all the results. This is called *composite rule*, *extended rule*, or *iterated rule*.
5. Example: Composite Trapezoidal Rule: For example, the composite trapezoidal rule can be stated as

where the sub-intervals have the form

with

and

Here the sub-intervals used have the same length , but one could use intervals of varying length

1. Definition of Newton-Cotes Formula: Interpolation with polynomials evaluated at equally spaced points in yields the Newton-Cotes formula, of which the rectangle rule and the trapezoidal rule are examples. Simpson’s rule, which is based on a polynomial of order , is also a Newton-Cotes formula.
2. Quadrature Rules with Nesting Property: Quadrature rules with equally spaced points have the very convenient property of *nesting*. The corresponding rule with each interval sub-divided includes all the current points, so those integrand points can be re-used.
3. Integrand with Variable Interpolant Spaces: If one allows the intervals between the interpolation points to vary, one finds another group of quadrature formulas, such as the Gaussian quadrature formula.
4. Improved Accuracy with Gaussian Quadrature: Gaussian quadrature rule is typically more accurate than a Newton-Cotes rule, which requires the same number of function evaluations, if the integrand is smooth, i.e., if it is sufficiently differentiable.
5. Other Varying Interval Quadrature Rules: Other quadrature methods with varying intervals include Clenshaw-Curtis quadrature methods – also called Fejer quadrature – and these do nest.
6. Nestability of Gaussian Quadrature Rules: Gaussian quadrature rules do not nest, but the related Gauss-Kronrod quadrature formulas do.

**Generalized Mid-Point Rule Formulation**

1. Generalized Mid-Point Rule Expression: A generalized mid-point rule formula is given by

or

1. Example Expression for Inverse Tangent: For example, substituting

and

in the generalized mid-point rule formula, one obtains the equation of the inverse tangent as

where

is the imaginary unit, and

1. Eliminating the Series Odd Terms: Since at each odd the numerator of the integrand becomes

the generalized mid-point rule formula can be re-organized as

1. Transforming Limits into : For a function defined over the interval its integral is

Therefore, the generalized mid-point integration formula above can be applied assuming that

**Adaptive Algorithms**

1. Lack of Suitable Derivative Points: If does not sufficient derivatives at all points, or if the derivatives become large, the generalized mid-point quadrature is often insufficient. In such cases, the adaptive algorithm similar to the one outlined in Wikipedia (2019) will perform better.
2. Estimation of the Quadrature Error: Some details of the algorithm require careful thought. For many cases, estimating the error from quadrature over an interval for the function is not obvious. One popular solution is to use two different rules for the quadrature, and use their difference as an estimate of the error from the quadrature.
3. Too Large Versus Small Errors: The other problem is deciding what *too large* or *very small* signify.
4. Local Criterion for *Too Large*: A *local* criterion for *too large* is that the quadrature error should not be larger than , where , a real number, is the tolerance one wishes to set for the global error. However, if is too tiny, it may not be worthwhile to make it even smaller even if the quadrature error is apparently large.
5. Global Criterion for *Too Large*: A *global* criterion is that the sum of the errors on all the intervals should be less than .
6. *A-Posteriori* Type of Error Analysis: This type of error analysis is typically called *a-posteriori* since the error is computed after having computed the approximation.
7. Forsythe Heuristics for Adaptive Quadrature: Heuristics for adaptive quadrature are discussed in Forsythe, Malcolm, and Moler (1977).

**Extrapolation Methods**

1. Error Dependence on Evaluation Point Count: The accuracy of a quadrature rule of the Newton-Cotes type is generally a function of the number of evaluation points. The result is usually more accurate as the number of evaluation points increases, or, equivalently, the width of the step size between the points decreases.
2. Error Dependence on Step Width: It is natural to ask what the result would be if the step size were allowed to approach zero. This can be answered by extrapolating the result from two or more non-zero step sizes, using series acceleration methods such as Richardson extrapolation.
3. Details of the Extrapolation Methods: The extrapolation function may be either a polynomial or a rational function. Extrapolation methods are described in more detail by Stoer and Bulirsch (1980) and are implemented in many of the routines in the QUADPACK library.

**A Priori Conservative Error Estimation**

1. Bounded First Derivative over : Let have a bounded first derivative over , i. e.,

The mean-value theorem for where

gives

for some

depending on .

1. Integrating Over on Both Sides: On integrating in from to on both sides and taking the absolute values, one obtains
2. Approximating the Right-Hand Side: The integral on the right-hand side can be further approximated by bringing the absolute value into the integrand, and replacing the term by an upper bound:

where the supremum has been used for the approximation.

1. Corresponding Approximation Applied to the LHS: Hence, if the integral is approximated by the quadrature , the error is no greater than the right-hand side above.
2. Converting the RHS into a Riemann Sum: This can be converted into an error analysis for the Riemann sum, giving an upper bound of

for the error term of that particular approximation. Note that this precisely is the error obtained for the example

1. Strict Upper Bounds on the Error: Using more derivatives, and by tweaking the quadrature, a similar analysis can be done using a Taylor series, using a partial sum with a remainder term, for . This analysis gives a strict upper bound on the error, if the derivatives of are available.
2. Algorithmic Proofs and Verified Calculations: The integration method can be combined with interval arithmetic to produce computer proofs and *verified* calculations.

**Integrals Over Infinite Intervals**

1. Standard Techniques for Unbounded Intervals: Several methods exist for approximate integration over unbounded intervals. The standard technique involves specially derived quadrature rules, such as Gauss-Hermitian quadrature for integrals on the whole real line, and Gauss-Laguerre quadrature for the integrals on the positive reals (Leader (2004)).
2. Changing Variables to Bounded Intervals: Monte Carlo methods can be used, or a change of variables to a finite interval can be applied; e.g., for the whole line, one could use

and for semi-infinite intervals one could use

as possible transformations.

**Multi-dimensional Integrals**

1. Fubini’s Theorem: Curse of Dimensionality: The quadrature rules discussed so far are al designed to compute one-dimensional integrals. To compute integrals in multiple dimensions, one approach is to phrase the multiple integrals as repeated one-dimensional integrals by applying the Fubini’s theorem – the tensor product rule. This approach requires the function evaluations to grow exponentially as the number of dimensions increases. Three methods described below are known to overcome this so-called *curse of dimensionality*.
2. Multi-dimensional Cubature Integration Rules: A great many additional techniques for forming multi-dimensional cubature integration rules for a variety of weighting functions are given in Stroud (1971).

**Monte Carlo**

1. Potential Accuracy Improvement from MC: Monte Carlo and quasi-Monte Carlo methods are easy to apply to multi-dimensional integrals. They may yield greater accuracy for the same number of function evaluations than repeated integrations using one-dimensional methods.
2. Markov Chain Monte Carlo Algorithms: A large class of useful Monte Carlo methods are the so-called Markov Chain Monte Carlo algorithms, which include the Metropolis-Hastings algorithms and Gibbs sampling.

**Sparse Grids**

Sparse grids were originally developed by Smolyak for the quadrature of high-dimensional functions. The method is always based on a one-dimensional quadrature rule, but performs a more sophisticated combination of univariate results. However, whereas the tensor product rule guarantees that the weight of all the quadrature points will be positive, Smolyak’s rule does not guarantee that the weights will be positive.

**Bayesian Quadrature**

Bayesian quadrature is a statistical approach to the numerical problem of computing integrals and falls under the field of probabilistic numerics. It can provide a full handling of the uncertainty over the range of the integral expressed as a Gaussian process posterior variance. It is also known to provide very fast convergence rates which can be up to exponential in the number of quadrature points (Briol, Oates, Girolami, and Osborne (2015)).

**Connections to Differential Equations**

1. The ODE Initial Value Problem: The problem of evaluating the integral

can be reduced to an initial value problem for an ordinary differential equation by applying the first part of the fundamental theorem of calculus.

1. Differential Form of the ODE: By differentiating both sides of the above with respect to the argument , it can be seen that the function satisfies
2. Applying the ODE Solution Schemes: Methods for ordinary differential equations, such as Runge-Kutta schemes, can be applied to the re-stated problem and thus be used to evaluate the integral. For instance, the standard fourth order Runge-Kutta method applied to the differential equation yields the Simpson’s rule from above.
3. Separating the Independent/Dependent Variables: The differential equation

has a special form; the right-hand side contains only the dependent variable (here ) and not the independent variable (here . This simplifies the theory and the algorithms considerably.

**References**

* Briol, F. X., C. J. Oates, M. Girolami, and M. A. Osborne (2015): [Frank-Wolfe Bayesian Quadrature: Probabilistic Integration with Theoretical Guarantees](https://arxiv.org/abs/1506.02681) **arXiv**
* Forsythe, G. E., M. A. Malcolm, and C. B. Moler (1977): *Computer Methods for Mathematical Computation* **Prentice Hall** Englewood Cliffs NJ
* Leader, J. J. (2004): *Numerical Analysis and Scientific Computation* **Addison Wesley**
* Stoer, J., and R. Bulirsch (1980): *Introduction to Numerical Analysis* **Springer-Verlag** New York
* Stroud, A. H. (1971): *Approximate Calculation of Multiple Integrals* **Prentice Hall** Englewood Cliffs NJ
* Wikipedia (2019): [Numerical Integration](https://en.wikipedia.org/wiki/Numerical_integration)

**Gaussian Quadrature**

**Introduction and Overview**

1. Quadrature Rule in Numerical Analysis: In numerical analysis, a **quadrature rule** is the approximation of the definite integral of a function, usually stated as a weighted sum of the function values at the specified points within the domain of integration (Wikipedia (2019)).
2. n-Point Gaussian Quadrature Rule: An n-point **Gaussian Quadrature Rule**, named after Carl Friedrich Gauss, is a quadrature rule constructed to yield the exact result for polynomials of degree or less using a suitable choice of nodes and weights for
3. Gaussian Quadrature Nodes and Weights: The most common domain of integration for such a rule is taken as , so the rule can be stated as

which is exact for polynomials of degree or less. This exact rule is known as the Gauss-Legendre quadrature rule.

1. Approximating by a Polynomial: The quadrature rule will only be an approximation to the integral above if is well approximated by a polynomial of degree or less in
2. Integrands with End-Point Singularities: The Gauss-Legendre quadrature rule is not typically used for integrable functions with end-point singularities.
3. Alternative Specification of the Quadrature Rules: Instead, of the integrand can be written as

where is well-approximated by a low-degree polynomial, then the alternative nodes and weights will usually give more accurate quadrature rules.

1. The Gauss-Jacobi Quadrature Rules: These are known as Gauss-Jacobi quadrature rules, i.e.,
2. Gauss-Chebyshev Quadrature Weights: Common weights include - referred to as Chebyshev-Gauss – and .
3. Gauss-Laguerre and Gauss-Hermite Quadrature Rules: One may also want to integrate over semi-infinite intervals – the Gauss-Laguerre quadrature – or infinite intervals – the Gauss-Hermite quadrature.
4. Quadrature Nodes as Roots of Orthogonal Polynomials: It can be shown (Stoer and Bulirsch (2002), Press, Teukolsky, Vetterling, and Flannery (2007)) that the quadrature nodes are the roots of a polynomial belonging to a class of orthogonal polynomials, i.e., they belong to the class that is orthogonal with respect to a weighted inner-product. This is a key observation for computing Gauss quadrature nodes and weights.

**Gauss-Legendre Quadrature**

1. Legendre Polynomials as Associated Orthogonals: For the simplest integration problem stated above, i.e., where is well-approximated by polynomials in , the associated orthogonal polynomials are Legendre polynomials, denoted by .
2. Weights of the Legendre Terms: With polynomial normalized to give

the Gauss node, , is the root of , and the weights are given by the formula (Abramowitz and Stegun (2007))

1. Low Order Nodes and Weights: Some low order quadrature rules are tabulated below over . The next section contains other intervals.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Points** | **Points** | **Approximate** | **Weight** | **Approximate** |
| 1 | 0 | 0 | 2 | 2 |
| 2 |  |  | 1 | 1 |
| 3 | 0 | 0 |  |  |
|  |  |  | 0.555556 |
| 4 |  |  |  |  |
|  |  |  |  |
| 5 | 0 | 0 |  |  |
|  |  |  |  |
|  |  |  |  |

**Change of Interval**

1. Interval Change - to : The integral over must be changed to the integral over before applying the Gaussian quadrature rule. The change of interval can be done in the following way:
2. Application of Gaussian Quadrature Rules: Applying the Gaussian quadrature rule then results in the following approximation:

**Other Forms**

1. Generalization of the Integrand Quadrature: The integration problem can be expressed in a slightly more general way by introducing a positive weight function into the integrand, and allowing an interval other than . That is, the problem is to calculate for some choices of , , , and .
2. Parameter Choices for the Quadrature Generation: For

and

the problem is the same as the one considered above. Other choices lead to other integration rules, some of which are tabulated below.

|  |  |  |
| --- | --- | --- |
| **Interval** |  | **Orthogonal Polynomials** |
|  |  | Legendre Polynomials |
|  |  | Jacobi Polynomials |
|  |  | Chebyshev Polynomials (First Kind) |
|  |  | Chebyshev Polynomials (Second Kind) |
|  |  | Laguerre Polynomials |
|  |  | Generalized Laguerre Polynomials |
|  |  | Hermite Polynomials |

**Fundamental Theorem**

1. n-Degree Polynomial Driving the Quadrature: Let be a non-trivial polynomial of degree such that

for all

1. Nodes as Roots of the Polynomial: If the nodes are picked to be the zeros of , then there exist weights which make the Gauss quadrature computed integral exact for polynomials of degree or less. Furthermore, all these nodes lie in the open interval (Stoer and Bulirsch (2002)).
2. Chosen as the Orthogonal Polynomial: The polynomial is said to be an orthogonal polynomial of degree associated with the weight function . It is unique to a constant normalization factor.
3. Decomposing into Orthogonal Polynomials: The idea underlying the proof is that, because of its sufficiently low degree, can be divided by to produce a quotient strictly lower than , and a remainder of still lower degree, so that both will be orthogonal to , by the defining property of . Thus,
4. Quadrature Over Reduced Degree Polynomial: Because of the choice of nodes , the corresponding relation

also holds. The exactness of the computed integral then follows from the exactness of , i.e., for polynomials of degree or less.

**General Formula for the Weights**

1. Generic Expression for the Quadrature Weights: The weights can be expressed as

where is the coefficient of in .

1. Lagrange Polynomial Form for : To prove this, note that using the Lagrange interpolation, one can express in terms of as

because has a degree less than and is thus fixed by the value it attains at different points.

1. Re-evaluation of the Quadrature: Multiplying both sides by and integrating from to yields
2. Quadrature Weights Using Lagrange Polynomials: The weights are thus given by
3. Quadrature Weights Using Orthogonal Polynomials: This integral expression for can be expressed in terms of the orthogonal polynomials and as follows.
4. Orthogonal Polynomial from Lagrange Numerator: One can write

where is the coefficient of in .

1. Orthogonal Derivative from Lagrange Denominator: Taking the limit of to yields, using L’Hopital’s rule
2. Weights from Polynomials and Derivatives: The integral expression for the weights can thus be written as
3. Reducing the Degree of Integrand: In the integrand, writing

yields

provided

because is a polynomial of degree which is then orthogonal to .

1. Product of Lower Degree Polynomials: So, if is a polynomial of at most degree , one has
2. as Degree Polynomial: The integral on the right-hand side can be evaluated for

as follows. Because is a polynomial of degree , one has

where is a polynomial of degree .

1. Polynomial in the Weight Integral: Since is orthogonal to , one has
2. in Terms of : One can then write

The term in the brackets is a polynomial of degree , which is therefore orthogonal to .

1. Weight Integral in Terms of : The integral can thus be written as
2. Recovery of the Postulated Expression: According to

the weights are obtained by dividing the weight integral above by , and that yields the expression

1. Alternate Expression Using the Term: can also be expressed in terms of the orthogonal polynomials and . In a 3-term recurrence relation (see below)

the term vanishes, so in

can be replaced by .

**Proof that the Weights are Positive**

1. Remainder – Lagrange Squared Term Polynomial: Consider the following polynomial of degree

where, as above, the are the roots of the polynomial .

1. Gaussian Quadrature for the above Polynomial: Clearly

Since the degree is less than , the Gaussian quadrature formula involving the weights and the nodes obtained from applies.

1. Weights Have to be Positive: Since

for not equal to , one has

Since both and are non-negative functions, it follows that

**Computation of Gaussian Quadrature Rules**

There are many algorithms for computing the nodes and the weights of Gaussian quadrature rules. The popular are the Golub-Welsch algorithm requiring operations, Newton’s method for solving

using the three-term recurrence relation for evaluation requiring operations, and asymptotic formulas for large requiring operations.

**Recurrence Relation**

1. Recurrence Relation for Orthogonal Polynomials: Orthogonal polynomials with

for

for a scalar product ,

and leading coefficient one – i.e., monic orthogonal polynomials – satisfy the recurrence relation

where the scalar product is defined as

for

where is the upper bound on the degree – which can be taken to infinity – and where

1. Proof of Recurrence using Induction: First of all, the polynomials defined by the recurrence relation starting with

have a leading coefficient of one and the correct degree (i.e. ). Setting the starting polynomial by , the orthogonality of can be demonstrated by induction.

1. Proof for the First Term: For

one has

1. Proof for an Arbitrary Term: Now, if are orthogonal, so is , because in

all scalar products vanish except for the first one and the one where meets the same orthogonal polynomial. Therefore,

1. Reduction to Three Term Recurrence: However, if the scalar product satisfies

- which is the case for Gaussian Quadrature – the above recurrence relation reduces to a three-term recurrence relation.

1. Serial Zeroing of Recurrence Terms: For

is a polynomial of degree less than or equal to . On the other hand, is orthogonal to every polynomial of degree less than or equal to . Therefore, one has

and

for

1. Full Form of the Recurrence Relation: The recurrence relation then simplifies to

or, with the convention

where

where the final step occurs because

since differs from by a degree less than .

**The Golub-Welsch Algorithm**

1. Three-Term Jacobi Recurrence Matrix: The three-term recurrence relation can be written in the matrix form

where

is the standard basis vector, i.e.

and is the so-called Jacobi matrix.

1. The Golub-Welsch Algorithm: The zeroes of the polynomials upto degree , which are used as nodes for the Gaussian quadrature, can be found by computing the eigenvalues of this tri-diagonal matrix. This procedure is known as the Golub-Welsch algorithm.
2. Elements of the Tri-diagonal Matrix: For computing the weights and the nodes, it is preferable to consider the symmetric tridiagonal matrix with elements
3. Quadrature Nodes from the Eigenvalues: and are similar matrices, and therefore have the same eigenvalues – the quadrature nodes.
4. Quadrature Weights Extracted from Eigenvectors: The weights can be computed from the corresponding eigenvectors. If is a normalized eigenvector – i.e., an eigenvector with Euclidean norm equal to one – associated with the eigenvalue , the corresponding weight can be computed from the first component of this eigenvector, namely:

where is the integral of the weight function

Gil, Segura, and Temme (2007) contain further details.

**Error Estimates**

1. Stoer-Bulirsch (2002) Error Estimate: Using the analysis presented in Stoer and Bulirsch (2002), the error of a Gaussian quadrature can be stated as follows. For an integrand that has continuous derivatives,

for some in , where is the monic orthogonal polynomial of degree , and

1. Kahaner, Moler, and Nash (1989) Error Estimate: In the important special case of

one has the error estimate (Kahaner, Moler, and Nash (1989))

1. Conservative Nature of the Estimate: Stoer and Bulirsch (2002) remark that this error estimate is inconvenient in practice, since it may be difficult to estimate the order derivatives. Furthermore, the actual error may be much less that the bound established by the derivative.
2. Error Estimate Using Different Orders: Another approach is to use two Gaussian quadrature rules of different orders, and to estimate the error as the difference between these two results. For this purpose, the Gauss-Kronrod quadrature rules can be useful.

**Gauss-Kronrod Rules**

1. Sub-divided Points do not Coincide: If the interval is sub-divided, the Gaussian evaluation points of the new sub-interval never coincide with the previous evaluation points – except at zero for odd numbers – and thus the integrand must be evaluated at every point.
2. Extensions to Gauss Quadrature Rules: *Gauss-Kronrod rules* are extensions to Gauss quadrature rules generated by adding points to a -point rule in such a way that the resulting rule is of the order . This allows for computing higher-order estimates while re-using the function values of the lower-order estimates.
3. Estimation of the Quadrature Error: The difference between the Gauss quadrature rule and its Kronrod extension is often used an estimate of the approximation error.

**Gauss-Lobatto Rules**

1. Gaussian vs. Lobatto Quadrature Rules: Also known as *Lobatto quadrature* (Abramowitz and Stegun (2007)), Gauss-Lobatto quadrature is named after the Dutch mathematician Rehuel Lobatto. It is similar to the Gaussian quadrature, except for the following differences:
   1. The integration points include the end-points of the integration interval.
   2. It is accurate for polynomials up to degree , where is the number of integration points (Quatteroni, Sacco, and Saleri (2000)).
2. Gauss-Lobatto Quadrature Function Expression: Lobatto quadrature of function in an interval is
3. Gauss-Lobatto Quadrature Abscissa: The abscissae is the zero of .
4. Gauss-Lobatto Quadrature Weights:
5. Gauss-Lobatto Quadrature Error Remainder:
6. Low Order Gauss-Lobatto Quadrature Weight Sample:

|  |  |  |
| --- | --- | --- |
| **Number of Points** | **Points** | **Weights** |
| 3 |  |  |
|  |  |
| 4 |  |  |
|  |  |
| 5 |  |  |
|  |  |
|  |  |
| 6 |  |  |
|  |  |
|  |  |
| 7 |  |  |
|  |  |
|  |  |
|  |  |

1. Implementation of the Adaptive Variant: An adaptive variant of this algorithm with 2 interior nodes (Gander and Gautschi (2000)) is found in GNU Octave and in MATLAB as *quadl* and *intergate*, respectively.

**References**

* Abramowitz, M., and I. A. Stegun (2007): *Handbook of Mathematics Functions* **Dover Book on Mathematics**
* Gander, W., and W. Gautschi (2000): Adaptive Quadrature – Revisited *Bit Numerical Mathematics* **40 (1)** 84-101
* Gil, A., J. Segura, and N. M. Temme (2007): *Numerical Methods for Special Functions* **Society for Industrial and Applied Mathematics** Philadelphia
* Kahaner, D., C. Moler, and S. Nash (1989): *Numerical Methods and Software* **Prentice Hall**
* Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (2007): *Numerical Recipes: The Art of Scientific Computing 3rd Edition* **Cambridge University Press** New York
* Quatteroni, A., R. Sacco, and F. Saleri (2000): *Numerical Mathematics* **Springer Verlag** New York
* Stoer, J., and R. Bulirsch (2002): *Introduction to Numerical Analysis 3rd Edition* **Springer**
* Wikipedia (2019): [Gaussian Quadrature](https://en.wikipedia.org/wiki/Gaussian_quadrature)

**Gauss-Kronrod Quadrature**

**Introduction and Overview**

1. Adaptive Method for Numerical Integration: The *Gauss-Kronrod quadrature formula* is an adaptive method for numerical integration.
2. Improved Accuracy by Re-using Less Accurate Results: It is a variant of the Gaussian quadrature, in which the evaluation points are chosen so that an accurate approximation can be computed by re-using the information produced by the computation of a less accurate approximation (Wikipedia (2019)).
3. Multi-Order Quadrature Rules and Errors: It is an example of what is called a nested quadrature rule; for the same set of function evaluation points, it has two quadrature rules, one higher order and one lower order – the latter is called an *embedded* rule. The difference between these two approximations is used to estimate the calculation error of the integral.

**Description**

1. Numerical Approximation of Definite Integrals: The problem in numerical integration is to approximate
2. Use of n Point Gaussian Quadrature: Such integrals can be approximated, for example, by n-point Gaussian quadrature

where and are the points and the weights used to evaluate the function .

1. Consequences of Non-Matching Nodes: If the interval is sub-divided, the Gauss evaluation points of the new sub-intervals never coincide with the previous evaluation points – except at the mid-point for odd numbers of evaluation points – and thus the integrand must be evaluated at every point.
2. Node Extensions using Stieltjes Polynomials: Gauss-Kronrod formulas are extensions of the Gauss quadrature formulas generated by adding points to a point rule in such a way that the resulting rule is of order (Laurie (1997)); the corresponding Gauss rule is order . The extra points are zeros of Stieltjes polynomials.
3. Kronrod Extension as an Error Estimate: This allows for computing higher-order error estimates while re-using the function values of a lower order estimate. The difference between a Gauss quadrature rule and its Kronrod extension is used often as an estimate of the approximation error.

**Example**

1. 7 Point Gauss Plus 15 Point Kronrod: A popular example combines a 7-point Gauss rule with a 15-point Kronrod rule (Kahaner, Moler, and Nash (1989)). Because the Gauss points are incorporated into the Kronrod points, a total of only 15 function evaluations are needed.
2. on :

|  |  |
| --- | --- |
| **Gauss Nodes** | **Weights** |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Kronrod Nodes** | **Weights** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1. Use in Quadrature Error Estimate: The integral is then estimated by the Kronrod rule and the error can be estimated as .
2. Enhancements to the Quadrature Algorithm: Patterson (1968) showed how to find further extensions of this type. Monegato (1978) and Piessens, de Doncker-Kapenga, Uberhuber, and Kahaner (1983) proposed improved algorithms. Finally, the most efficient algorithm was proposed by Laurie (1997).
3. Tabulation of Quadruple Precision Coefficients: Quadruple Precision (34 decimal digits) coefficients for , , , , and others are computed and tabulated in Holoborodko (2011).

**Implementations**

Routines for Gauss-Kronrod quadrature are provided by the QUADPACK library, the GNU Scientific Library, the NAG Numerical Libraries R, the C++ Boost Library, and DROP.

**References**

* Holoborodko, P. (2011): [Gauss-Kronrod Quadrature Nodes and Weights](https://www.advanpix.com/2011/11/07/gauss-kronrod-quadrature-nodes-weights/)
* Kahaner, D., C. Moler, and S. Nash (1989): *Numerical Methods and Software* **Prentice Hall**
* Laurie, D. (1997): Calculation of Gauss-Kronrod Quadrature Rules *Mathematics of Computation* **66 (219)** 1133-1145
* Monegato, G. (1978): Some Remarks on the Construction of Extended Gaussian Quadrature Rules *Mathematics of Computation* **32 (141)** 247-252
* Patterson, T. N. L. (1968): The Optimum Addition of Points to Quadrature Formulae *Mathematics of Computation* **22 (104)** 847-856
* Piessens, R., E. de Doncker-Kapenga, C. W. Uberhuber, and D. K. Kahaner (1983): *QUADPACK – A Subroutine Package for Automatic Integration* **Springer-Verlag**
* Wikipedia (2019): [Gauss-Kronrod Quadrature Formula](https://en.wikipedia.org/wiki/Gauss%E2%80%93Kronrod_quadrature_formula)

**Gamma Distribution**

**Overview**

1. Parametrization of the Gamma Distribution: The *gamma distribution* is a two-parameter family of continuous probability distributions. The Erlang distribution, the exponential distribution, and the chi-square distribution are all special cases of the gamma distribution. There are three different parameterizations in common use:

* With a shape parameter and a scale parameter
* With a shape parameter

and an inverse scale parameter

called a rate parameter

* With a shape parameter and a mean parameter

In each of these forms, both parameters are positive and real numbers (Wikipedia (2019)).

1. As a Constrained Maximum Entropy Distribution: The gamma distribution is a maximum entropy probability distribution – both with respect to a uniform base measure and with respect to a base measure – for a random variable for which

is fixed and greater than zero, and

is fixed. Here is the digamma function (Park and Bera (2009)).

**Gamma Distribution – Central Measures Table**

|  |  |  |
| --- | --- | --- |
| **Parameters** |  |  |
| **Support** |  |  |
| **PDF** |  |  |
| **CDF** |  |  |
| **Mean** |  |  |
| **Median** | No Closed Form | No Closed Form |
| **Mode** | for | for |
| **Skewness** |  |  |
| **Excess Kurtosis** |  |  |
| **Entropy** |  |  |
| **MGF** | for | for |
| **CDF** |  |  |

**Definitions**

1. Usage of the Parametrization: The parametrization using and appeears to be more common in econometrics and other applied fields where, for example, the gamma distribution is frequently used to model waiting times. For instance, in life testing, the waiting time until death is a random variable that is frequently modeled with a gamma distribution (Hogg, McKean, and Craig (2013)).
2. Usage of the Parametrization: The parametrization with and is more common in Bayesian statistics, where the gamma distribution is used as a conjugate prior for various types of inverse scale – aka rate – parameters, such as the of the exponential or the Poisson distribution (Gopalan, Hofman, and Blei (2014)) – or, for that matter, the of the gamma distribution itself. The closely related inverse gamma distribution is used as a conjugate prior for the scale parameters, such as the variance of a normal distribution.
3. Erlang Distribution - Dependent Exponentials: If is a positive integer, then the distribution represents an Erlang distribution, i.e., the sum of independent exponentially distributed random variables, each of which has a mean of .

**Characterization Using Shape and Rate**

1. Parameterized Gamma Distribution Representation: The gamma distribution can be represented in terms of a shape parameter

and an inverse scale parameter

called a rate parameter. A random variable that is gamma distributed with shape and rate is denoted

1. Parameterized Gamma Distribution PDF: The corresponding probability distribution in the shape-rate parametrization is

for

and

for

and

Here is the gamma function. Both parametrizations are common because either can be more convenient depending on the situation.

1. Parameterized Gamma Distribution CDF: The cumulative distribution function is the regularized gamma function

where is the lower incomplete gamma function.

1. Independent Integer - Erlang Distribution: If is a positive integer, i.e., the distribution is an Erlang distribution, the cumulative distribution function has the following series expansion (Papoulis and Pillai (2002))

**Characterization using Shape and Scale**

1. Parameterized Gamma Distribution: A random variable that is gamma distributed with shape and rate is denoted
2. Parameterized Gamma Distribution PDF: The probability density function in the shape-scale parametrization is

for

and

Here is the gamma function evaluated at .

1. Parameterized Gamma Distribution CDF: The cumulative distribution function is the regularized gamma function

where is the lower incomplete gamma function.

1. Independent Integer - Erlang Distribution: It can also be expressed as follows; if is a positive integer, i.e., the distribution is an Erlang distribution (Papoulis and Pillai (2002))

**Properties – Skewness**

The skewness of the gamma distribution depends only on its shape parameter , and is equal to .

**Properties – Median Calculation**

1. Expression for the Median Value: Unlike the mode and the mean which have readily calculable formulas based on the parameters, the median does not have an easy closed form equation. The median for this distribution is defined as the value such that
2. Banneheka-Ekanayake Approximation for the Median: A formula for approximating the median for any gamma distribution, when the mean is known, has been derived based on the fact that is approximately a linear function of when

(Banneheka and Ekanayake (2009)). The approximation formula is

where

is the mean.

1. Chen and Rubin Median Bounds: A rigorous treatment of the problem of choosing asymptotic expansion and the bounds for the median of the gamma distribution was handled first by Chen and Rubin (1986), who proved that

where denote the median of the distribution.

1. Choi Series Expansion for Median: Choi (1994) later showed that the first five terms in the asymptotic expansion of the median are

by comparing the median to the Ramanujan’s function. Later it was shown that is a convex function of (Berg and Pedersen (2008)).

**Properties – Summation**

If has a distribution for

where all distributions have the same scale parameter , then

provided all are independent. Mathai (1982) and Moschopoulus (1984) treat the case where are independent but have different scale parameters. The gamma distribution exhibits infinite divisibility.

**Properties – Scaling**

If

then, for any

by moment generating functions, or equivalently

Indeed, it is known that if is an exponential random variable with rate then is an exponential random variable with rate ; the same thing is true with gamma variates.

**Properties – Exponential Family**

The gamma distribution is part of the two-parameter exponential family with natural parameters and - equivalently and - and natural statistics and . If the shape parameter is held fixed, the resulting one-parameter family of distributions is a natural exponential family.

**Properties – Logarithmic Expectation and Variance**

It can be shown that

or, equivalently,

where is the digamma function. Likewise,

where is the polygamma function. This can be derived using the exponential family formula for the moment generating function of the sufficient statistic, because one of the sufficient statistics of the gamma distribution is .

**Properties – Information Entropy**

The information entropy is

In the parameterization, the information entropy is given by

**Properties – Kullback-Liebler Divergence**

1. Parametrization of the Divergence: The Kullback-Liebler divergence – KL-divergence - of - the true distribution – from - the *approximating* distribution – is given as
2. Parametrization of the Divergence: The Kullback-Liebler divergence – KL-divergence - of - the true distribution – from - the *approximating* distribution – is given as

**Properties – Laplace Transform**

The Laplace Transform of the Gamma PDF is

**Related Distributions – General**

1. Relation between Gamma and Exponential Sums: If are independent and identically distributed random variables following an exponential distribution with rate parameter then
2. Exponential and Chi Square Distributions: If

- the shape-scale parametrization – then has an exponential distribution with the rate parameter . If

- shape-scale parametrization – then is identical to the chi-square distribution with degrees of freedom. Conversely, if

and is a positive constant, then

1. Erlang and Poisson *Arrival* Process: If is an integer, the gamma distribution is an Erlang distribution and is the probability distribution of the waiting time until the arrival in a one-dimensional Poisson process with intensity . If

then

1. Relation to Maxwell-Boltzmann Distribution: If has a Maxwell-Boltzmann distribution with parameter then
2. Relation to Generalized Gamma Distribution: If

then follows a generalized gamma distribution with parameters

and

More generally, if

then for

follows the generalized gamma distribution with parameters

and

1. Relation to Inverse Gamma Distribution: If

then

1. Ratio of Independent Random Gamma Variables: If

are independent, then

Equivalently

Alternatively, if

are independent, then

or, equivalently,

1. Distribution of the Ratio : If

and

are independently distributed, then has a beta distribution with parameters and , and is independent of , which is distributed.

1. Unit Scale Gamma Variables Sum: If

are independently distributed, then the vector , where

follows a Dirichlet distribution with parameters .

1. Large Convergence to Normal: For large , the gamma distribution converges to a normal distribution with mean

and variance

1. Conjugate Prior of Gaussian Precision: The gamma distribution is a conjugate prior for the precision of a normal distribution with known mean.
2. Multivariate Generalization of Gamma Distribution: The Wishart distribution is a multi-variate generalization of the gamma distribution, where the samples are positive-definite matrices rather than positive real numbers.
3. Other Generalizations of the Gamma Distribution: The gamma distribution is a special case of the generalized gamma distribution, the generalized integer gamma distribution, and the generalized inverse gamma distribution. Further, the gamma distribution is a member of the family of Tweedie exponential dispersion models.
4. Negative Binomial Distribution as a Discrete Analogue: Among the discrete distributions, the negative binomial distribution is sometimes considered the discrete analogue of the gamma distribution.

**Properties – Compound Gamma**

If the shape parameter of the gamma distribution in known, but the inverse-sale parameter is unknown, then a gamma distribution for the inverse-scale forms a conjugate prior. The compound distribution which results from integrating out the inverse-scale has a closed form solution, known as the compound gamma distribution (Dubey (1970)). If, instead, the shape parameter is known but the mean is unknown, with the prior of the mean being given by another gamma distribution, it then results in -distribution.

**Statistical Inference – Maximum Likelihood Parameter Estimation**

1. Likelihood Estimator – Joint Observation Setup: The likelihood function for i.i.d. observations is

from which the log-likelihood function maybe calculated as

1. Maximization across Scale Parameter Space: Finding the maximum with respect to and setting it equal to zero yields the maximum likelihood estimator of the parameter:

Substituting this into the log-likelihood function gives

1. Maximization across Shape Parameter Space: Finding the maximum with respect to by taking the derivative and setting it to zero yields
2. Iterative Root Search - Starting Point: There is no closed-form solution for . The function is numerically well-behaved, so if a numerical solution is desired, it can be found using, for example, the Newton’s method. An initial value of can be found using the method of moments, or using the approximation
3. Iterative Root Search - Variate Increment: Letting

results in a that is approximately

which is within of the correct value (Minka (2002)). An explicit form for the Newton-Raphson update for this initial guess is (Choi and Wette (1969))

**Closed-Form Estimators**

1. Consistent Estimators for Shape/Scale: Consistent closed-form estimators for and exist that are derived from the likelihood of the generalized gamma distribution (Ye and Chen (2017)). The estimate for the shape is

and the estimate for the scale is

If the rate parametrization is used, the estimate of is

1. Bias Correction for Shape/Scale: These estimators are not strictly maximum likelihood estimators, but are instead referred to a mixed-type log-moment estimators. They have, however, similar efficiency as the maximum likelihood estimators. Although these estimators are consistent, they have a small bias. A bias corrected variant of the estimator for the scale parameter is

The bias correction for the shape parameter is given as (Francisco, Ramos, and Ramos (2019))

**Bayesian Minimum Mean-Squared Error**

1. Posterior from Scale-invariant Prior: With known and unknown , the posterior density function for using the standard scale-invariant prior for is
2. Expression for Joint Probability: Denote

then

1. Integration across the Space: Integration with respect to can be carried out using a change of variables

revealing that is gamma distributed with parameters

and

1. Posterior Distribution Mean and Variance: The moments can be computed by taking the ratio of by as

which shows that the mean standard deviation estimate for the posterior estimate for is

**Bayesian Inference Conjugate Prior**

1. Gamma Distribution as Conjugate Prior: In Bayesian inference, the gamma distribution is the conjugate prior to many likelihood distributions: Poisson, exponential, normal with known mean, Pareto, gamma with known shape , inverse gamma with known shape parameter, and Gompertz with known scale parameter.
2. Conjugate Prior for the Gamma Distribution: The gamma distribution’s conjugate prior is (Fink (1997))

where is the normalizing constant, which has no closed-form solution.

1. Bayesian Update of Parameter Priors: The posterior distribution can be found by updating the parameters as follows:

where is the number of observations, and is the observation.

**Occurrence and Applications**

1. Insurance Claims and Rainfall Accumulations: The gamma distribution has been used to model the size of the insurance claims (Boland (2007)) and rainfall accumulation (Aksoy (2000)). This means that the aggregated insurance claims and the amount of rainfall accumulated in a reservoir are modeled by a gamma process – much like the exponential distribution generates a Poisson process.
2. Error in Multi-level Poisson Regression: The gamma distribution is also used to model the errors in multi-level Poisson regression models, because the combination of the Poisson distribution and the gamma distribution is a negative binomial distribution.
3. Multi-path Signal Processing: In wireless communication, the gamma distribution is used to model the multi-path fading of the signal power; the Rayleigh and the Poisson distributions are also used.
4. Age Incidence of Cancer Distribution: In oncology, the age incidence of the cancer distribution often follows the gamma distribution, where the shape and the scale parameters predict, respectively, the mean number of driver events and the time interval between them (Belikov (2017)).
5. Inter-spike Neurological Interval Distribution: In neurosciences, the gamma distribution is often used to describe the distribution of inter-spike intervals (Robson and Troy (1987), Wright, Winter, Forster, and Bleeck (2014)).
6. Copy Number in Bacterial Gene Protein Expression: In bacterial gene expression, the copy number of a constitutively expressed protein often follows a gamma distribution, where the shape and the scale parameters are, respectively, the mean number of bursts per cell cycle and the mean number of protein molecules produced by a single mRNA during its lifetime (Friedman, Cai, and Xie (2006)).
7. Signal Recognition in Genomics: In genomics, the gamma distribution has been applied in the peak calling step, i.e., in the recognition of signal, in ChIP-chip (Reiss, Facciotti, and Baliga (2008)), and ChIP-seq (Mendoza-Parra, Nowicka, van Gool, and Gronemeyer (2013)) data analysis.
8. Use as a Conjugate Prior: The gamma distribution is used as a conjugate prior in Bayesian statistics. It is the conjugate prior for the precision, i.e., the inverse of the variance, for a normal distribution. It is also the conjugate prior for the exponential distribution.

**Computational Methods – Generating Gamma Distributed Random Variables**

1. Principal Methodology behind Gamma Generation: Give the scaling property above, it is enough to generate gamma variables with

as a conversion can be done for any with simple division. Suppose one wishes to generate random variables from where is a non-negative integer and

Using the fact that the is the same as an distribution, and given that it is straightforward to generate exponential variables, it may be concluded that if is uniformly distributed on then is distributed as , i.e., using the inverse transform sampling. Using the “ addition” property of the gamma distribution, this result may be expanded as

where are all uniformly distributed on and independent. All that is left now is to generate a variable distributed as for

and apply the “ addition” property once more. This is the most difficult part.

1. Literature Coverage on Gamma Generation: Random generation of gamma variables is discussed in detail by Devroye (1986), noting that none are uniformly fast for all shape parameters. For small values of the shape parameters, the algorithms are often not valid (Devroye (1986)). For arbitrary values of the shape parameter, one can apply the Ahrens and Dieter (1982) modified acceptance-rejection method algorithm GD for

or the transformation method (Ahrens and Dieter (1974)) when

Also applicable are Marsaglia’s squeeze method (Marsaglia (1977)) and the Chang and Feast algorithm GKM3 (Chang and Feast (1979)).

1. Ahrens-Dieter Acceptance-Rejection Method: The following is a version of the Ahrens-Dieter acceptance-rejection method (Ahrens and Dieter (1982)):
   1. Generate , , and as i.i.d. uniform variables.
   2. If

then

and

Otherwise

and

* 1. If

then go to step 1.

* 1. is distributed as

1. Summary of the Ahrens-Dieter Scheme: A summary of this is

where is the integer part of , is generated via the algorithm above with

- the fractional part of - and the are all independent.

1. Devroye Critique of Ahrens-Dieter: While the approach above is technically correct, Devroye notes that it is linear in and in general not a good choice. Instead, he recommends using either rejection-based or table-based methods, depending on the context (Devroye (1986)).
2. Marsaglia’s Transformation-Rejection Method: As an example, Marsaglia’s simple transformation-rejection method relies on one normal and one uniform random number (Marsaglia and Tsang (2000)):

where is standard normal.

* 1. If

and

return

* 1. Go back to step 2

1. Acceptance Rate Dependence on : With

the above algorithm generates a gamma distributed random number that is approximately constant with in time. The acceptance rate does depend on , with an acceptance rate of , , and for values of , , and respectively. For

one can use

to boost to be usable with this method.

**References**

* Ahrens, J. H., and U. Dieter (1974): Computer Methods for Sampling from Gamma, Beta, Poisson, and Binomial Distributions *Computing* **12 (3)** 223-246
* Ahrens, J. H., and U. Dieter (1982): Generating Gamma Variates by a Modified Rejection Technique *Communications of the ACM* **25 (1)** 47-54
* Aksoy, H. (2000): Use of Gamma Distribution in Hydrological Analysis *Turkish Journal of Engineering and Environmental Science* **24** 419-428
* Banneheka, B. M. S. G., and G. E. M. U. P. D. Ekanayake (2009): A New Point Estimator for the Median of the Gamma Distribution **14** 95-103
* Belikov, A. V. (2017): The Number of Key Carcinogenic Events that can be predicted from Cancer Incidence **7 (1)** 12170
* Berg, C., and H. Pedersen (2008): [Convexity of the Median in the Gamma Distribution](https://arxiv.org/pdf/math/0609442.pdf) **arXiV**
* Boland, P. J. (2007): *Statistical and Probabilistic Methods in Actuarial Science* **Chapman and Hall CRC Press**
* Chen, J., and H. Rubin (1986): Bounds for the Difference between the Mean and the Median of Gamma and Poisson Distributions *Statistics and Probability Letters* **4 (6)** 281-283
* Cheng, R. C. H., and G. M. Feast (1979): Some Simple Gamma Variate Generators *Journal of the Royal Statistical Society C* **28 (3)** 290-295
* Choi, S. C., and R. Wette (1969): Maximum Likelihood Estimation of the Parameters of the Gamma Distribution and their Bias *Technometrics* **11 (4)** 683-690
* Choi, K. P. (1994): On the Medians of the Gamma Distributions and an Equation of Ramanujan *Proceedings of the American Mathematical Society* **121 (1)** 245-251
* Devroye, L. (1986): *Non-Uniform Random Variate Generation* **Springer-Verlag** New York
* Dubey, S. D. (1970): Compound Gamma, Beta, and F Distributions *Metrika* **16** 27-31
* Fink, D. (1997): [Compendium of Conjugate Priors](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.157.5540&rep=rep1&type=pdf)
* Francisco, L., P. L. Ramos, and E. Ramos (2019): A Note on the Bias of Closed-Form Estimators for the Gamma Distribution Derived from Likelihood Equations *American Statistician* **73 (2)** 195-199
* Friedman, N., L. Cai, and X. S. Xie (2006): Linking Stochastic Dynamics to Population Distribution: An Analytical Framework of Gene Expression *Physical Review Letters* **97** 168302
* Gopalan, P., J. M. Hofman, and D. M. Blei (2014): [Scalable Recommendation with Poisson Factorization](https://arxiv.org/pdf/1311.1704v3.pdf) **arXiV**
* Hogg, R. V., J. McKean, and A. T. Craig (2013): *Introduction to Mathematical Statistics 7th Edition* **Pearson**
* Marsaglia, G. (1977): The Squeeze Method for generating Gamma Variates *Computers and Mathematics with Applications* **3 (4)** 321-325
* Marsaglia, G., W. W. Tsang (2000): A Simple Method for Generating Gamma Variates *ACM Transactions on Mathematical Software* **26 (3)** 363-372
* Mathai, A. M. (1982): Storage Capacity of a Dam with Gamma Type Inputs *Annals of the Institute of Statistical Mathematics* **34 (3)** 591-597
* Mendoza-Parra, M. A., M. Nowicka, W. van Gool, and H, Gronemeyer (2013): [Characterizing ChIP-seq binding patterns by model-based peak shape deconvolution](https://bmcgenomics.biomedcentral.com/articles/10.1186/1471-2164-14-834)
* Minka, T. P. (2002): [Estimating a Gamma Distribution](https://tminka.github.io/papers/minka-gamma.pdf)
* Moschopoulus, P. G. (1985): The Distribution of the Sum of the Independent Random Gamma Variables *Annals of the Institute of Statistical Mathematics* **37 (3)** 541-544
* Papoulis A., and S. U. Pillai (2002): *Probability, Random Variables, and Stochastic Processes 4th Edition* **McGraw-Hill**
* Park, S. Y., and A. K. Bera (2009): Maximum Entropy Auto-regressive Conditional Heteroscedasticity Model *Journal of Econometrics* **150 (2)** 219-230
* Reiss, D. J., M. T. Facciotti, and N. S. Baliga (2008): Model-based Deconvolution of Genome-wide DNA Binding *Bioinformatics* **24 (3)** 396-403
* Robson, J. G., and J. B. Troy (1987): Nature of the Maintained Discharge of the Q, the X, and the Y Retinal Ganglion Cells of the Cat *Journal of the Optical Society of America A* **4 (12)** 2301-2307
* Wikipedia (2019): [Gamma Distribution](https://en.wikipedia.org/wiki/Gamma_distribution)
* Wright, M. C. M., I. M. Winter, J. J. Forster, and S. Bleeck (2014): Response to Best-Frequency Tone Bursts in the Ventral Cochlear Nucleus is governed by Ordered Inter-spike Interval Statistics *Hearing Research* **317** 23-32
* Ye, Z. S., and N. Chen (2017): Closed-Form Distributions for the Gamma Distribution Derived from the Likelihood Equations *American Statistician* **71 (2)** 177-181

**Chi-Squared Distribution**

**Overview**

1. Definition of the Chi-Squared Distribution: The chi-squared distribution – also called chi-squared or distribution – with degrees of freedom is the distribution of the sum of the squares of independent standard normal random variables (Wikipedia (2019)).
2. Specialization of the Gamma Distribution: The chi-squared distribution is a special case of the gamma distribution and is one of the most widely used probability distributions in inferential statistics, notably in hypothesis testing or in the construction of confidence intervals (Mood, Graybill, and Boes (1974), Johnson, Klotz, and Balakrishnan (1994), Abramowitz and Stegun (2007), National Institute of Standards and Technology (2019)).
3. Non-central Chi-Squared Distribution: When it is being distinguished from the more general non-central chi-squared distribution, this distribution is sometimes called the central chi-squared distribution.
4. Common Uses of the Distribution: The chi-squared distribution is used in the common chi-squared tests for the goodness of distribution of an observed distribution to a theoretical one, the independence of two criteria of classification of qualitative data, and in the confidence interval estimation for the population standard deviation of a normal distribution from a sample standard deviation.
5. Analysis of Variance by Ranks: Many other statistical tests also use this distribution,, such as Friedman’s analysis of variance by ranks.

**Definition**

1. Chi-Squared Distribution – Mathematical Definition: If are independent standard normal random variable, then the sum of their squares

is distributed according to the chi-squared distribution with degrees of freedom.

1. Chi-Squared Representation and Parametrization: This is usually denoted as

or

The chi-squared distribution has one parameter , a positive integer that specifies the number of degrees of freedom – the number of ‘s.

**Introduction**

1. Primary Use in Hypothesis Testing: The chi-squared distribution is used primarily in hypothesis testing.
2. Not Used in Direct Modeling: Unlike the more widely known distribution such as the normal distribution and the exponential distribution, the chi-squared distribution is not as often applied in direct modeling of natural phenomenon.
3. Typical Hypothesis Test Use Cases: It arises in the following hypothesis tests, among others:
   1. Chi-squared test of independence in contingency tables.
   2. Chi-squared test of goodness of fit of observed data to hypothetical distributions
   3. Likelihood-ratio test for nested models
   4. Log-rank test in survival analysis
   5. Cochran-Mantel-Haenszel test for stratified contingency tables
4. Other Uses of the Distribution: It is also a component of the definition of the t-distribution and F-distribution used in t-tests, analysis of variance, and regression analysis.
5. Relationship to the Normal Distribution: The primary reason that the chi-squared distribution is used extensively in hypothesis testing is its relationship to the normal distribution.
6. Test-Statistic Sample Size Behavior: Many hypothesis tests use a test statistic, such as the t-statistic in a t-test. For these hypothesis tests, as the sample size increases, the sampling distribution of the test statistic approaches a normal distribution – central limit theorem.
7. Asymptotic Test-Statistic Sampling: Because the test statistic – such as – is asymptotically normally distributed, provided that the sample size is sufficiently large, the distribution used for hypothesis testing may be approximated by a normal distribution.
8. Usage with Underlying Normal Distributions: Testing hypothesis using a normal distribution is well-understood and relatively easy. The simplest chi-squared distribution is the square of a standard normal distribution. So wherever a normal-distribution could be used for a hypothesis test, a chi-squared distribution could be used.
9. Random Draw from a Normal Distribution: Specifically, suppose that is a standard normal random variable, with mean and variance ,

A sample drawn at random from is a sample from the standard normal distribution.

1. Random Draw from Chi-Squared Distribution: Define a new random variable . To generate a random sample from , take a sample from and square the value. The distribution of the squared values is given by the random variable

The distribution of the random variable is an example of a chi-squared distribution

1. Degrees of Freedom of the Distribution: The subscript 1 indicates that this particular chi-squared is constructed from only one standard normal distribution. A chi-squared distribution constructed by squaring a single standard normal distribution is said to have 1 degree of freedom.
2. Chi-Squared Distribution Asymptotic Approach: Thus, as the sample size for a hypothesis test increases, the distribution of the test-statistic approaches a normal distribution, and the distribution of the square of the test-statistic approaches a chi-squared distribution.
3. Extreme Values under the Distribution: Just as the extreme values of the normal distribution have low probability – and give small -values – extreme values of the chi-squared distribution have low probability.
4. Likelihood Ratio Class of Tests: An additional reason that the chi-squared distribution is widely used is that it is a member of the class of likelihood ratio tests LRT (Westfall (2013)).
5. Neyman-Pearson Lemma: LRT’s have several desirable properties; in particular, LRT’s commonly provide highest power to reject NULL hypothesis.
6. Advantages of using t-distribution: However, the normal and the chi-squared distribution are valid only asymptotically. For this reason, it is better to use the t-distribution rather than the normal approximation or the chi-squared approximation for small sample size.
7. Power of Exact Binomial Test: Similarly, in the analysis of contingency tables, the chi-squared approximation will be poor for small sample size, and it is preferable to use Fisher’s exact test. Ramsey (1988) shows that the exact binomial test is always more powerful than the normal approximation.
8. Binomial, Normal, and Chi-Squared: Lancaster (1969) showed the connections between the binomial, the normal, and the chi-squared distributions as follows. De Moivre and Laplace established that a binomial distribution could be approximated by a normal distribution. Specifically, they showed the asymptotic normality of the random variable

where is the observed number of successes in trials, the probability of success is and

1. Chi-Squared Distribution Variable: Squaring both sides of the equation gives
2. Re-factoring the Chi-Squared Variable: Using

and

this equation simplifies to

1. Multivariate Generalization by Pearson: The expression on the right is of the form that Pearson would generalize to

where is the Pearson’s cumulative test statistic, which asymptotically approaches a distribution, is the number of observations of type ,

is the expected theoretical frequency of the type , asserted by the NULL hypothesis that the fraction of type in the population is , and is the number of cells in the table.

1. Reducing Binomial to Normal - : In the case of a binomial outcome – flipping a coin – the binomial distribution may be approximated by a normal distribution for sufficiently large . Because the square of a standard normal distribution is the chi-squared distribution with 1 degree of freedom, the probability of results such as 1 head in 10 trials can be approximated by either the normal or the chi-squared distribution.
2. Extension to Multiple Categorical Variables: However, many problems involve more than two possible outcomes of a binomial, and instead require 3 or more categories, which leads to the multinomial distribution.
3. Chi Squared as Approximating Multinomial Distribution: Just as de Moivre and Laplace sought for and found the normal distribution approximation to the binomial, Pearson sought for and found a multivariate normal approximation to the multinomial distribution. Pearson showed that the chi-squared distribution, the sum of multiple normal distributions, was such as approximation to the multinomial distribution (Lancaster (1969)).

**Probability Density Function**

The probability density function of the chi-squared distribution is

where denotes the gamma function, which has closed-form values for integer .

**Cumulative Distribution Function**

1. Explicit Expression for the CDF: The cumulative distribution function is

where is the lower incomplete gamma function and is the regularized gamma function.

1. Expression for the Special Case : The special case of

of this function has a simple form:

and the integer recurrence of the gamma function makes it easy to compute for other small even .

1. Tables for the CDF: Tables for the chi-squared cumulative distribution function are widely available and the function is included in many spreadsheets and all statistical packages.
2. Chernoff Bounds on CDF Tails: Letting

Chernoff bounds on the lower and the upper tails of the CDF may be obtained (Dasgupta and Gupta (2003)).

1. Chernoff Bounds for : For the cases when

– which include all of the cases when this CDF is less than half –

1. Chernoff Bounds for : The tail bound for cases when

similarly is

**Additivity**

1. Sum of Independent Chi-squared Variables: It follows from the definition of the chi-squared distribution that the sum of the independent chi-squared variables is also chi-squared distributed.
2. Additivity over Degrees of Freedom: Specifically, if are independent chi-squared variables with degrees of freedom, respectively, then

is chi-squared with degrees of freedom.

**Sample Mean**

1. Chi-Squared Distribution Sample Mean: The sample mean of i.i.d. chi-squared variables of degree is distributed according to a gamma distribution with shape and scale parameters:

where

1. Asymptotic Reduction to Normal Distribution: Asymptotically, give that for a scale parameter going to infinity, a Gamma distribution converges to a normal distribution with expectation

and variance

the sample mean converges towards

1. Asymptotics using CLT: Note that one would have obtained the same result instead invoking the central limit theorem, noting that for each chi-squared variable of degree , the expectation is and the variance is – and hence the variance of the mean is

**Entropy**

1. Expression for Differential Entropy: The differential entropy is given by

where is the digamma function.

1. Chi-squared as a MaxEnt Distribution: The chi-squared distribution is the maximum entropy probability distribution for a random variable for which

and

are fixed.

1. Log Moment of Gamma Distributions: Since the chi-squared is in the family of gamma distributions, this can be derived by substituting the appropriate values in the expectation of the moments of gamma.

**Non-central Moments**

The moments about zero of a chi-squared distribution with degrees of freedom are given (Simon (2002))

**Cumulants**

The cumulants are readily obtained by a formal power series expansion of the logarithm of the characteristic function

**Asymptotic Properties**

1. Degrees of Freedom Based Determinants of Normality: By the central limit theorem, because the chis-squared distribution is the sum of independent normal variables with finite mean and variance, it converges to a normal distribution for large . For many practical purposes, for

the distribution is sufficiently close to a normal distribution (Hunter, Box, and Hunter (1970)).

1. Speed of Convergence to Normality: Specifically, if

then as tends to infinity, the distribution of tends to a standard normal distribution. However, convergence is slow because the skewness is and the excess kurtosis is

1. Schemes for Faster Approach to Normality: The sampling distribution of converges to normality much faster than the sampling distribution of (Bartlett and Kendall (1946)), as the logarithm removes much of the asymmetry (Pillai (2016)). Other functions of the chi-squared distribution converge more rapidly to a normal distribution.
2. Chi-squared Derivative Function #1: Some examples are: If

then is approximately normally distributed with mean and unit variance (Johnson, Klotz, and Balakrishnan (1994)).

1. Chi-squared Derivative Function #2: If

then is approximately normally distributed with mean and variance . This is known as the Wilson-Hilferty transformation (Wilson and Hilferty (1931), Johnson, Klotz, and Balakrishnan (1994)).

**Relation to Other Distributions**

1. Normal Distribution: As
2. Non-central Chi-squared Distribution:

is a non-central chi-squared distribution with the non-centrality parameter

1. Double Chi-squared Distribution #1: If

then

has the double chi-squared distribution

1. Double Chi-squared Distribution #2: As a special case, if

then

has the double chi-squared distribution

1. Chi-squared Distribution as a Norm: The squared norm of standard normally distributed variables is a chi-squared distribution with degrees of freedom:
2. Gamma Distribution: If

and

then

which is a gamma distribution.

1. Chi Distribution: If

then

the chi-distribution.

1. Exponential Distribution: If

then

which is an exponential distribution.

1. Rayleigh Distribution: If

which is a Rayleigh distribution, then

1. Maxwell Distribution: If

which is a Maxwell distribution, then

1. Inverse Chi-squared Distribution: If

then

which is the inverse chi-squared distribution.

1. Type 3 Pearson Distribution: The chi-squared distribution is a special case of Type-3 Pearson distribution.
2. Beta Distribution: If

and

are independent, then

which is the beta distribution.

1. Chi-squared Distribution from Uniform: If

which is a uniform distribution, then

1. Laplace Distribution Transformation: is a transformation of the Laplace distribution. If

then

1. Generalized Normal Distribution Version #1: If follows a generalized normal distribution version 1 with parameters , , and then

(Backstrom and Fischer (2018))

1. Transformed Pareto Distribution: Chi-squared distribution is a transformation of the Pareto distribution.
2. Student t Distribution from Chi-Squared: Student-t distribution is a transformation of the chi-squared distribution. Student-t distribution can be obtained from a chi-squared distribution and a normal distribution.
3. Non-central Beta Distribution: Non-central beta distribution can be obtained as a transformation of the chi-squared distribution and non-central chi-squared distribution.
4. Non-central -t Distribution: Non-central t-distribution can be obtained from normal distribution and chi-squared distribution.
5. Multidimensional Gaussian Random Vectors: If is a dimensional Gaussian random vector with mean and rank covariance matrix , then

is chi-squared with degrees of freedom.

1. Non central Chi squared Distribution: The sum of squares of statistically independent unit variance Gaussian variables which do *not* have mean zero yields a generalization of the chi-squared distribution called the non-central chi-squared distribution.
2. Rank-reduced Chi-squared Distribution: If is a vector of i.i.d. standard normal variables and is a symmetric idempotent matrix with rank , then the quadratic form is chi-squared with degrees of freedom.
3. Special Normal Chi-squared Distribution: If is a positive semi-definite covariance matrix with strictly positive diagonal covariance entries, then for

and a random -vector independent of such that

and

it then holds that

(Pillai (2016)).

1. Relation to Non-Gaussian Distributions: Thus, the chi-squared is also naturally related to other distributions arising from the Gaussian.
2. Ratio of Independent Distributions: In particular, if is -distributed,

if

where

and

are statistically independent.

1. Sum of Correlated Distributions: If

and

are statistically independent, then

If and are not independent, then is not chi-squared distributed.

**Generalizations**

1. Construction of the Chi-squared Distributions: The chi-squared distribution is obtained as the sum of independent, zero-mean unit-variance Gaussian random variables.
2. Generalizing the Chi-squared Distribution: Generalizations of this distribution can be obtained by summing the squares of other types of Gaussian random variables. Several such distributions are described below.

**Linear Combination**

If are chi-squared random variables and

then a closed expression for the distribution is

is not known. It may be, however, approximated efficiently using the property of characteristic functions of chi-squared random variables (Bausch (2013)).

**Non-Central Chi-Squared Distribution**

The non-central chi-squared distribution is obtained from the sum of squares of independent Gaussian random variables having unit variance and *non-zero* means.

**Generalized Chi-Squared Distribution**

The generalized chi-squared distribution is obtained from the quadratic form where is a zero-means Gaussian vector having an arbitrary covariance matrix, and is an arbitrary matrix.

**Gamma, Exponential, and Related Distribution**

1. Parametrized Specialization of the Gamma Distribution: The chi-squared distribution

is a special case of gamma distribution, in that

using the rate parameterization of the gamma distribution – or

using the scale parameterization of the Gamma distribution – where is an integer.

1. Exponential Distribution from Chi-squared: Because the exponential distribution is also a special case of the gamma distribution, one also has that if

then

is an exponential distribution.

1. Erlang Specialization of the Gamma Distribution: The Erlang distribution is also a special case of the gamma distribution, and thus one also has that if

with even , then is Erlang distributed with shape parameter and scale parameter .

**Occurrence and Applications**

1. Test and Variance Estimation: The chi-squared distribution has numerous applications in inferential statistics, for instance in chi-squared tests, and in estimating variances.
2. Regression Slope and Population Mean: It enters the problem of estimating the mean of a normally distributed population and the problem of estimating the slope of a regression line via its role in the Student-t distribution.
3. Analysis of Variance using F-distribution: It enters all analysis of variance problems via its role in the F-distribution, which is the distribution of the ratio of two independent chi-squared random variables, each divided by their respective degrees of freedom.
4. Central from Gaussian Distribution: Following are some of the most common situations in which the chi-squared distribution arises from a Gaussian distributed sample. If are i.i.d. random variables, then

where

1. Variants in Normal Distribution: The box below shows some statistics based on

independent random variables that have probability distributions related to chi-squared distribution.

|  |  |
| --- | --- |
| **Name** | **Distribution** |
| Chi-squared Distribution |  |
| Non-central Chi-squared Distribution |  |
| Chi Distribution |  |
| Non-central Chi Distribution |  |

1. Use in Magnetic Resonance Imaging: The chi-squared distribution is also often encountered in magnetic resonance imaging (den Dekker and Sijbers (2014)).

**Table of Values vs. -Values**

1. Review of the -Value Definition: The -Value is the probability of observing a test statistic *at least* as extreme in the specified distribution (here chi-squared).
2. Computing the -Value from CDF: Thus, since the cumulative distribution function (CDF) for the appropriate degrees of freedom gives the probability of having obtained a value *less extreme* than this point, subtracting CDF from 1 gives the -Value.
3. Establishing Statistical Significance using -Values: A low -Value, below the chosen significance level, indicates statistical significance, i.e., sufficient evidence to reject the NULL hypothesis.
4. Typical Hypothesis Testing Significance Threshold: A significance level of 0.05 is often used as the cut-off between significant and not-significant results.
5. -Values for with 10 Degrees of Freedom: The table below gives a number of -Values matching to for the first 10 degrees of freedom (Pennsylvania State University (2016)).

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Degrees of Freedom** | **Value** | | | | | | | | | | |
| **1** | 0.004 | 0.02 | 0.06 | 0.15 | 0.46 | 1.07 | 1.84 | 2.71 | 3.84 | 6.63 | 10.830 |
| **2** | 0.100 | 0.21 | 0.45 | 0.71 | 1.39 | 2.41 | 3.22 | 4.61 | 5.99 | 9.21 | 13.820 |
| **3** | 0.350 | 0.58 | 1.01 | 1.42 | 2.37 | 3.66 | 4.64 | 6.25 | 7.81 | 11.34 | 16.270 |
| **4** | 0.710 | 1.06 | 1.65 | 2.20 | 3.36 | 4.88 | 5.99 | 7.78 | 9.49 | 13.28 | 18.470 |
| **5** | 1.140 | 1.61 | 2.34 | 3.00 | 4.35 | 6.06 | 7.29 | 9.24 | 11.07 | 15.09 | 20.520 |
| **6** | 1.630 | 2.20 | 3.07 | 3.83 | 5.35 | 7.23 | 8.56 | 10.64 | 12.49 | 16.81 | 22.460 |
| **7** | 2.170 | 2.83 | 3.82 | 4.67 | 6.35 | 8.38 | 9.80 | 12.02 | 14.07 | 18.48 | 24.320 |
| **8** | 2.730 | 3.49 | 4.59 | 5.53 | 7.34 | 9.52 | 11.03 | 13.36 | 15.51 | 20.09 | 26.120 |
| **9** | 3.320 | 4.17 | 5.38 | 6.39 | 8.34 | 10.66 | 12.24 | 14.68 | 16.92 | 21.67 | 27.880 |
| **10** | 3.940 | 4.87 | 6.18 | 7.27 | 9.34 | 11.78 | 13.44 | 15.99 | 18.31 | 23.21 | 29.590 |
| **p-Value** | **0.950** | **0.90** | **0.80** | **0.70** | **0.50** | **0.30** | **0.20** | **0.10** | **0.05** | **0.01** | **0.001** |

1. p-Value from Inverse CDF Function: The above values can also be calculated using the quantile function – also known as *inverse CDF* or *ICDF* – of the chi-squared distribution – e.g., for a p-Value of 0.05 and 7 degrees of freedom no gets 14.06714, or 14.07 as in the able above.

**Summary Expressions**

|  |  |
| --- | --- |
| **Notation** | or |
| **Parameters** | (degrees of freedom) |
| **Support** | if  otherwise |
| **Probability Density Function** |  |
| **Cumulative Density Function** |  |
| **Mean** |  |
| **Median** |  |
| **Mode** |  |
| **Variance** |  |
| **Skewness** |  |
| **Excess Kurtosis** |  |
| **Entropy** |  |
| **Moment Generating Function** | for |
| **Characteristic Function** | Sanders (2011) |
| **Probability Generating Function** | for |

**References**

* Abramowitz, M., and I. A. Stegun (2007): *Handbook of Mathematics Functions* **Dover Book on Mathematics**
* Backstrom, T., and J. Fischer (2018): Fast Randomization for Distributed Low Bit-rate Coding of Speech and Audio *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **26 (1)** 19-30
* Bartlett, M. S., and D. G. Kendall (1946): The Statistical Analysis of Variance Heterogeneity and the Logarithmic Transformation *Supplement to the Journal of the Royal Statistical Society* **8 (1)** 128-138
* Bausch, J. (2013): [On the Efficient Calculation of a Linear Combination of Chi-Square Random Variables with an Application in Counting String Vacua](https://arxiv.org/abs/1208.2691) **arXiv**
* Dasgupta, S. D. K., and A. K. Gupta (2003): An Elementary Proof of a Theorem of Johnson and Lindenstrauss *Random structures and Algorithms* **22 (1)** 60-65
* den Dekker, A. J., and j. Sijbers (2014): Data Distributions in Magnetic Resonance Images: A Review *European Journal of Medical Physics* **30 (7)** 725-741
* Hunter, W. G., G. E. P. Box, and J. S. Hunter (1978): *Statistics for Experimenters* **Wiley**
* Johnson, N. L., S. Klotz, and N. Balakrishnan (1994): *Continuous Univariate Distributions* ***1*** *2nd Edition* **John Wiley and Sons**
* Lancaster, H, O. (1969): *The Chi-Squared Distribution* **Wiley**
* Mood, A., A. F. Graybill, and D. C. Boes (1974): *Introduction to the Theory of Statistics 3rd Edition* **McGraw-Hill**
* National Institute for Standards and Technology (2019): [Chi-Square Distribution](https://www.itl.nist.gov/div898/handbook/eda/section3/eda3666.htmhttps:/dlmf.nist.gov/8)
* Pennsylvania State University (2016): [Chi-squared Distribution](http://www.r-tutor.com/elementary-statistics/probability-distributions/chi-squared-distribution)
* Pillai, N. S. (2016): An Unexpected Encounter with Cauchy and Levy *Annals of Statistics* **44 (5)** 2089-2097
* Ramsey, P. H. (1988): Evaluating the Normal Approximation to the Binomial Test *Journal of Educational Statistics* **13 (2)** 173-182
* Sanders, M. A. (2011): [Characteristic Function of the Central Chi-squared Distribution](https://web.archive.org/web/20110715091705/http:/www.planetmathematics.com/CentralChiDistr.pdf)
* Simon, M. K. (2002): *Probability Distributions involving Gaussian Random Variables* **Springer** New York, NY.
* Westfall, P. H. (2013): *Understanding Advanced Statistical Methods* **CRC Press** Boca Raton, FL
* Wikipedia (2019): [Chi-squared Distribution](https://en.wikipedia.org/wiki/Chi-squared_distribution)
* Wilson, E. B., and M. M. Hilferty (1931): The Distribution of Chi-squared *Proceedings of the National Academy of Sciences* **17 (12**) 684-688.

**Non-central Chi-Square Distribution**

**Overview**

1. Generalization of the Chi-Square Distribution: The *non-central chi-square distribution* is a generalization of the chi-square distribution (Wikipedia (2019)).
2. NULL Chi Square Distribution Tests: It often arises in the power analysis of statistical tests in which the NULL distribution – perhaps asymptotically – is a chi-square distribution; important examples of such tests are the likelihood ratio tests.

**Background**

1. Non-central Chi-square Distribution: Let be independent, normally distributed random variables with means and unit variances. Then the random variable is distributed according to the non-central chi-square distribution.
2. Non-central Chi-square Distribution Parameters: It has 2 parameters - which specifies the number of degrees of freedom, i.e., the number of , and which is related to the mean of the random variables by

is sometimes called the non-centrality parameter; some references define in other ways, such as half of the above sum, or its square root.

1. as Squared Mean: This distribution arises in multivariate statistics as a derivative of multivariate normal distributions. While the central chi-square distribution is the squared norm of a random vector with distribution, i.e., squared distance from the origin to a point taken at random from that distribution, the non-central chi-square is the squared norm of a random vector with distribution, Here is a vector of length ,

and is the identity matrix of size .

**Non-central Chi-Square Distribution Table**

|  |  |
| --- | --- |
| **Parameters** | Degrees of Freedom  Non-centrality Parameter |
| **Support** |  |
| **CDF** |  |
| **PDF** | with Marcum Q-function |
| **Mean** |  |
| **Variance** |  |
| **Skewness** |  |
| **Excess Kurtosis** |  |
| **MGF** | for |
| **CF** |  |

**Definition**

1. Non-central Chi-square PDF: The probability density function is given by

where is distributed as chi-square with degrees of freedom.

1. Intuition behind the PDF Expression: From the above representation, the non-central chi-square distribution is seen to be a Poisson-weighted mixture of central chi-square distributions. Suppose that a random variable has a Poisson distribution with mean , and the conditional distribution of given

is chi-square with degrees of freedom. Then the unconditional distribution of is non-central chi-square with degrees of freedom, and non-centrality parameter .

1. Bessel Function Based PDF Expression: Alternatively, the PDF can be written as

where is a modified Bessel function of the first kind given by

1. Hypergeometric Function Based PDF: Using the relation between the Bessel functions and the hyper-geometric functions, the PDF can also be written as (Muirhead (2005))

Siegel (1979) discusses the case

specifically – zero degrees of freedom – in which the distribution has a discrete component at zero.

**Properties – Moment Generating Function**

The moment generating function is given by

for

**Properties – Moments**

1. First Four Non-central Moments: The first few raw moments are
2. Second, Third, and Fourth Central Moments: The first few central moments are
3. Cumulant/Arbitrary Non-central Moment: The cumulant is

Hence

**Cumulative Distribution Function**

1. Explicit Expression for the CDF: Using the relation between the central and the non-central chi-square distributions, the cumulative CDF can be written as

where is the cumulative distribution function of the central chi-square distribution with degrees of freedom, which is given by

and is the lower incomplete gamma function.

1. CDF Based Marcum Q Function: The Marcum Q-function can also be used to represent the CDF (Nuttall (1975)) as

**Approximation – including for Quantiles**

1. Abdel-Aty Non-central CDF Approximation: Abdel-Aty (1954) derives – as a first approximation – a non-central Wilson-Haferty approximation. is approximately normally distributed as , i.e.,

where

which is quite accurate and well-adapting to non-centrality. Also

becomes

for

which is the central chi-square case.

1. Sankaran Family of Normal Approximations: Sankaran (1963) discusses a number of closed form approximations for the cumulative distribution function. In an earlier paper (Sankaran (1959)) he states and derives the following approximation:

where

This and the other approximations are discussed in Johnson, Klotz, and Balakrishnan (1995). For a given probability, these formulas are easily inverted to provide the corresponding approximation for , to compute the approximation quantiles.

**Derivation of the PDF**

1. Spherically Symmetric Independent Gaussian Variates: The derivation of the probability density function is done most easily by performing the following steps. First, since have unit variances, their joint distribution is spherically symmetric up to a location shift.
2. Dependence on the Squared Distribution Means: The spherical symmetry implies that the distribution of

depends only on the means through the squared length

Without loss of generality, one can therefore take

and

1. Density Contributions from Positive/Negative: The next step is to derive the density of

for the

case. Simple transformations of the random variables show that

1. Poisson Weighted Taylor Series Terms: Expand the term in a Taylor series. This gives the Poisson weighted mixture representation of the density, still for

The indices on the chi-square random variables in the series above are in this case.

1. Connection to the Central Chi Square: Finally, for the general case. It has been assumed, without loss of generality, that are standard normal, and so has a *central* chi-square distribution with degrees of freedom, independent of . Using the Poisson-weighted mixture representation for , and the fact that the sum of chi-square distributed random variables is also chi-square, completes the result. The indices in the series are

as required.

**Related Distributions**

1. Central Chi-square as a Special Case: If is chi-squared distributed as

then is also non-central chi-square distributed

1. Non central Distribution: If

and

and is independent of , then a non-central distributed variables is developed as

1. Poisson Conditioned Chi Square: If

then

1. Transformation to the Rice Distribution: If

then takes the Rice distribution with the parameter .

1. Approximation using the Normal Distribution: As shown in Muirhead (2005), if

then

in distribution as either

or

1. Independent Non central Chi Square Sum:

and

and is independent of , then

where

In general, for a finite set of

the sum of the non-central chi-square distributed random variables

has the distribution

where

This can be seen using the moment generating functions as follows:

by the independence of the random variables. It remains to plug-in the MGF for the non-central chi-square distributions into the product and compute the new MGF. Alternatively, it can be seen via the interpretation in the background section above as sum of squares of independently distributed random variables with variance of and the specified means.

1. Complex Non-central Chi-square: The complex non-central chi-square has applications in radio communications and radar systems. Let be independent scalar random variables of circular symmetry, means of and unit variances.

Then the real random variable

is distributed according to the complex non-central

where

**Transformations**

1. Sankaran Cumulant Analysis and Transformations: Sankaran (1963) discusses transformations of the form

He analyzes the expansions of the cumulants of up to the term and shows that the following choices of produce reasonable results.

makes the second cumulant of approximately independent of

makes the third cumulant of approximately independent of

makes the fourth cumulant of approximately independent of

1. Variance Stabilizing Transformation of : Also, a simpler transformation

can be used as a variance stabilizing transformation that produces a random variable with a mean

and variance . Usability of these transformations may be hampered by the need to take square roots of negative numbers.

**Use in Tolerance Intervals**

Two-sided normal regression tolerance intervals can be obtained based on the non-central chi-square distribution (Young (2010)). This enables the calculation of a statistical interval within which, with some confidence level, a specified proportion of the sampled population falls.

**References**

* Abdel-Aty, S. H. (1954): Approximate Formulae for the Percentage Points and the Probability Integral of the Non-central Distribution *Biometrika* **41 (3-4)** 538-540
* Johnson, N. L., S. Klotz, and N. Balakrishnan (1995): *Continuous Univariate Distributions* ***1*** *2nd Edition* **John Wiley and Sons**
* Muirhead, R. (2005): *Aspects of Multivariate Statistical Theory 2nd Edition* **Wiley**
* Nuttall, A. H. (1975): Some Integrals Involving the Function *IEEE Transactions on Information Theory* **21 (1)** 95-96
* Sankaran, M. (1959): On the Non-central Distribution *Biometrika* **46 (1-2)** 235-237
* Sankaran, M. (1963): Approximations to the Non-central Distribution *Biometrika* **50 (1-2)** 199-204
* Siegel, A. F. (1979): The Non-central Chi-square Distribution with Zero Degrees of Freedom and Testing for Uniformity *Biometrika* **66 (2)** 381-386
* Wikipedia (2019): [Non-central Chi-square Distribution](https://en.wikipedia.org/wiki/Noncentral_chi-squared_distribution)
* Young, D. S. (2010): tolerance: An R Package for estimating Tolerance Intervals *Journal of Statistical Software* **36 (5)** 1-39

**Householder Transformation**

**Overview**

1. Definition of the Householder Transformation: In linear algebra, a *Householder transformation* – also known as a *Household reflection* or *elementary reflector* – is a linear transformation that describes a reflection about a plane or a hyperplane containing the origin (Householder (1958), Wikipedia (2022)).
2. Analogue over General Inner Product: Its analogue over the general inner product spaces is the Householder operator.

**Definition - Transformation**

1. Normal Vector Defining Reflection Hyperplane: The reflection hyperplane can be defined by its *normal* vector, a unit vector – a vector with a length – that is orthogonal to the hyperplane.
2. Reflection of about Hyperplane: The reflection of the point about this hyperplane is the linear transformation

where is given as a column unit vector with Hermitian transpose .

**Definition – Householder Matrix**

The matrix constructed from this transformation can be expressed in terms of an outer product

known as the Householder matrix, where is the identity matrix.

**Definition – Properties**

1. Hermitian: The Householder matrix has the following properties. It is Hermitian:
2. Unitary: It is unitary:
3. Involutary: Hence, it is involutary:
4. Eigenvalues : The Householder matrix has eigenvalues . To see this, notice that if is orthogonal to the vector which was used to create the reflector, then

i.e., 1 is an eigenvalue with multiplicity , since there are independent vectors orthogonal to . Also, notice

and so is an eigenvalue with multiplicity .

1. Determinant: The determinant of a Householder reflection is -1, since the determinant of a matrix is the product of the eigenvalues, in this case one of which is -1 with the remainder being +1 – as in the previous point.

**Applications – Numerical Linear Algebra**

1. Extensive Use in Linear Algebra: Householder transformations are widely used in numerical linear algebra, for example, to annihilate the entries below the main diagonal of the matrix, to perform QR decomposition, and in the first step of the QR decomposition algorithm.
2. Use in Hessenberg/Hermitian Matrices: They are also widely used for transformation to a Hessenberg form. For symmetric or Hermitian matrices, the symmetry can be preserved, resulting in triangularization (Schabauer, Pacher, Sunderland, and Gansterer (2010)).

**Applications – Numerical Linear Algebra – QR Decomposition**

Householder reflections can be used to calculate QR decompositions by reflecting one first column of a matrix onto a multiple of a standard basis vector, calculating the transformation matrix, multiplying it with the original matrix and then recursing down the minors of that product.

**Applications – Numerical Linear Algebra – Tridiagonalization**

1. Treatment in Burden, Faires, and Burden: This procedure is presented Burden, Faires, and Burden (2015) – it just uses a slightly altered function with
2. Step #1 - Determining : In the first step, to form the Householder matrix in each step one needs to determine and , which are:
3. Step #2 – Construction of : In and , construct vector :

where

and

for each

1. Step #3 - Initial : Then compute
2. Step #4 - Iterative : Having found and computing the process is repeated for

as follows:

for each

1. Computing Tridiagonal and Symmetric Matrix: Continuing in this manner, the tridiagonal and the symmetric matrix is formed.

**Applications – Numerical Linear Algebra – Examples**

1. Applying Householder Scheme to a Matrix: In this example, taken from Burden, Faires, and Burden (2016), the given matrix is transformed to the similar tridiagonal matrix by using the Householder method.



1. First Householder Iteration: The first Householder matrix is generated as shown below:



1. Second Householder Iteration: generated above is used as shown below:



1. Final Result - Tridiagonal Symmetric Matrix: As can be seen, the final result is a tridiagonal symmetric matrix similar to the original one. The process here finished in two steps.

**Computational and Theoretical Relationship to Other Unitary Transformations**

1. Recap – Constructing a Householder Transform: The householder transformation is a reflection about a hyperplane with a unit normal vector , as stated earlier. An unitary transformation satisfies
2. Householder Transform Unit Determinant - Statement: Taking the determinant - power of the geometric mean, and trace – proportional to the arithmetic mean, of a unitary matrix reveals that its eigenvalues has unit modulus.
3. Householder Transform Unit Determinant - Proof: This can be directly:
4. Equality of Arithmetic/Geometric Means: Since arithmetic and geometric means are equal if the variables are constant, the claim of unit modulus is therefore established.
5. Givens Rotations and Householder Reflections: For the case of real-valued unitary matrices, one obtains orthogonal decompositions

It may be shown that any orthogonal matrix may be decomposed into a product of rotations called Givens Rotations, and Householder Reflections.

1. Length Invariance under Rotation/Reflection: This has an intuitive appeal since multiplication of a vector by an orthogonal matrix preserves the length of that vector, and rotations and reflections exhaust the set of real-valued geometric operations hat render invariant a vector’s length.
2. Coset Decomposition of Unitary Matrices: The Householder transformation has been shown to have a one-to-one relationship with canonical coset decomposition of unitary matrices defined in the group theory, which can be used to parametrize unitary operators in a very efficient manner (Cabrera, Strohecker, and Rabitz (2010)).
3. Advantage of Householder over Givens: Finally, it may be noted that a single Householder transform, unlike a solitary Givens transform, can act on all columns of a matrix, and as such exhibits the lowest computational cost for QR decomposition and triangularization.
4. Drawback of Householder over Givens: The penalty for this “computational optimality” is, of course, that Householder operations cannot be as deeply or as efficiently parallelized.
5. Suitability of Householder vs. Givens: As such, Householder is preferred over dense matrices on sequential machines, while Givens is preferred on sparse matrices, and/or parallel machines.

**References**

* Burden, R. L., J. D. Faires, and A. M. Burden (2015): *Numerical Analysis 10th Edition* **Cengage Learning** Boston, MA
* Cabrera, R., T. Strohecker, and H. Rabitz (2010): The Canonical Coset Decomposition of Unitary Matrices through Householder Transformations *Journal of Mathematical Physics* **51 (8)** 082101
* Householder, A. S. (1958): Unitary Transformation of a Non-singular Matrix *Journal of the ACM* **5 (4)** 339-342
* Schabauer, H., C. Pacher, A. G. Sunderland, and W. N. Gansterer (2010): Toward a Parallel Solver for Generalized Complex Symmetric Eigenvalue Problems *Procedia Computer Science* **1 (1)** 437-445
* Wikipedia (2022): [Householder Transformation](https://en.wikipedia.org/wiki/Householder_transformation)

**The Householder Transformation in Numerical Linear Algebra**

**Abstract**

1. Use of the Householder Transformation: This chapter defines the Householder transformation, then puts it to work in several ways (Kerl (2008)).
2. Geometrical Illustration of Algebraic Properties: To illustrate the usefulness of geometry to elegantly derive and prove seemingly algebraic properties if the transform;
3. Use in Determinants and Inverses: To demonstrate an application to numerical linear algebra – specifically, for matrix determinants and inverses;
4. Analysis of Roundoff Errors: To show how geometric notions of determinant and matrix norm can be used to easily understand round-off error in Householder and Gaussian-elimination methods.

**Linear Algebra**

1. Gaussian Elimination vs. Householder Method: This chapter compares and contrasts two techniques for computation of determinants and inverses of square matrices: the more familiar Gaussian elimination method, and the less familiar Householder method. It primarily uses geometric methods to do so.
2. Review of the Requisite Background: This requires some preliminaries from linear algebra, including geometric interpretations of determinant, matrix norm, and error propagation.

**Geometric Meanings of Determinant and Matrix Norm**

1. Linear Transformation - Definition of Determinant: Let

be a linear transformation – which may be thought of as a matrix with respect to a standard basis. The *determinant* of can be thought of as the *signed volume* of the image of the *unit cube*.



1. Geometric Definition of Matrix Norm: The *matrix norm* is, by definition, the *maximum extent* of the image of the *unit ball*.



1. Mathematical Formulation of Matrix Norm: The matrix norm of is defined either by

or equivalently by

1. Intuition behind the Matrix Norm: That is, either one considers the vectors in the unit ball and finds the maximum extent of their images, or we consider *all* non-zero vectors and find the maximum amount by the which they are scaled. The equivalence of these two characterizations is due to the linearity of .
2. Intuition behind the Matrix Norm: That is, either one considers the vectors in the unit ball and finds the maximum extent of their images, or we consider *all* non-zero vectors and find the maximum amount by which they are scaled. The equivalence of these two characterizations is due to the linearity of .
3. Implications of Determinant Value : The matrix with determinant preserves unsigned volume, but does not necessarily preserve norm. Of particular interest for this chapter are three kinds of matrices.
4. Rotation Matrix: A *notation matrix* is of the form

and has determinant



1. Reflection Matrix: An example of a *reflection matrix*, reflecting about the -axis

is which has determined :



1. Permutation Matrix: Another example of a reflection is a *permutation matrix*:

which has determinant :



This reflection is about the line

1. Reflection about an Arbitrary Axis: Construction of a reflection matrix about an arbitrary axis is accomplished using Householder transformations, as discussed later.
2. Shear Matrix: An example of a *shear matrix* is

which has determinant



**Computation of Determinants**

1. Computing Determinants using Cofactor Expansion: In elementary linear algebra (Friedberg, Insel, and Spence (2002)), one is, first taught to compute determinants using *cofactor expansion*. This is fine for computing determinants of ’s or ’s. However, it is ineffective for larger matrices.
2. Determinant of a Matrix: The determinant of a matrix is ; uses two multiplies and an add.
3. Determinant of a Matrix: To compute the determinant of a matrix using cofactor expansion, one works down a new column or a row. There are 3 determinants.
4. Determinant of a Matrix: To do a , there are determinants of ’s, each of which recursively takes 3 determinants of ’s.
5. Determinant of an Matrix: Continuing this pattern, it can be seen that there are on the order of multiplies and adds for an matrix. For example,

Even at a billion operations a second, this requires seconds, which is hundreds of millions of years. A better approach is needed.

1. Upper and Lower Triangular Matrices: If a matrix is *upper-triangular* or *lower-triangular*, then its determinant is the product of its diagonal entries. There are of these and one only needs to multiply them, so there are multipliers.
2. Transformation into Upper Triangular Form: The next step is to see how efficiently one can rework a given square matrix into upper-triangular form, and how this modification affects the determinant.
3. The Process of Upper Triangularization: Upper-triangularization is the process of converting certain elements of a matrix to zero, while modifying the others.
4. Pre-multiplication of by : Recall that when a matric acts by pre-multiplication on a matrix , one may think of acting on each column vector of .
5. Zeroing out Locations in Matrix: That is, the jth column of is simply the jth column of . Thus, the column vectors can be transformed to place zeros in various locations.
6. Impact of Transformation on Determinant: If one applies transformations through , then

i.e.,

1. Tracking the Determinants of Transforms: That is, all one needs to do is to keep track of the determinants of the transformations, multiply them up, and compute the determinant of the resulting upper-triangular matrix.
2. for Gaussian and Householder: It can be shown that Gaussian elimination and Householder methods for upper-triangularization take . Thus, these methods are far more efficient than naïve cofactor expansion.

**Computation of Matrix Inverses**

1. Augmented Matrix vs. Cofactor Extraction: A more efficient method than cofactor expansion is the *augmented matrix* method. Here, one places the input matrix next to an identity matrix and converts the augmented matrix into row-reduced echelon form.
2. Identity Matrix on the LHS: If it is possible to get the identity matrix on the LHS, then the inverse will be found on the RHS:
3. Row-Reduction of Augmented Matrix: When the augmented matrix is row-reduced, one applies a sequence of linear transformations to the augmented matrix. Let their product be :
4. Proof of Inverse at RHS: Then, if row reduction works, i.e., if one gets on the right-hand side, then

i.e.,

But this forces

Thus,

1. Two-Step Echelon Formation Process: Note that placing a matrix into a row reducing echelon form is a two-step process.
2. Generating the Upper Triangular Form: Putting it into the row echelon form simply means putting it into the *upper-triangular* form and dividing each row by its leading coefficient:



1. Clearing the Above-diagonal Entries:



1. Sequence of ’s to apply: The remaining question to be addressed below is what particular sequence of ’s to apply.

**Error Propagation**

1. Challenges with Poorly Designed Methods: Whenever one uses numerical techniques, on needs to realize that a poorly designed algorithm can allow error to grow enormously, producing highly incorrect results.
2. Errors Generated in Numerical Analysis: The study error in numerical analysis is discussed in detail in, for example, Burden, Faires, and Burden (2015).
3. Error Propagation under Linear Transform: Consider a true matrix at the kth step of an upper-triangularization process along with an actual matrix , where is a matrix.
4. Transformation of a Perturbed Matrix: Since one uses linear transformation, one has
5. Transformation of a Perturbed Vector: Also, since the jth column of is times the jth column of ; this further becomes
6. Extracting Vector Norm from Matrix Norm: The *vector norm* of is related to the vector norm of by the *matrix norm* of ; since

one has

1. Norm-preserving Linear Transformation Matrix: In particular, if a transformation matrix is norm-preserving, it does not increase error.
2. Norms of Gaussian/Householder Schemes: The next sections consider the norms used in the Gaussian elimination and Householder methods.

**Gaussian Elimination**

The next few sections discuss Gaussian elimination from the standpoint of concatenated linear transformations, with an emphasis on the geometrical of these transformations.

**Row Reduction using Gaussian Elimination**

1. Row Reduction of Sample Matrix: For clarity, one considers matrices of height 2. The following matrix is row-reduced:



This example illustrates the three matrices used in Gaussian elimination.

1. Row-swap Matrix: The *row-swap matrix* – a permutation matrix – has determinant -1 and norm 1:



1. Row-scaling Matrix: The *row-scaling matrix* has determinant – this example

and norm



1. Row-update Matrix: The *row-update matrix* – a shear matrix – has determinant 1 and a norm which can most easily visualize using the diagrams above:



In this example

1. Gaussian Elimination’s Magnification of Error: It is clear that Gaussian elimination magnifies error when a row-scaling matrix has a large , and/or when a row-update matrix has a large .



**Gaussian Elimination without Pivoting**

1. Naïve Application of Gaussian Elimination: Here is an example of the error that accumulates when one naively uses Gaussian elimination. The solution to the linear system

is, in the matrix form



or, in the augmented form



1. Solution to the Linear System: Clearly, the solution is



1. Perturbation of the Augmented Matrix: Perturbation to the linear system above results in solving the system



1. Gaussian Elimination with Rounding Applied: One would expect a solution close to . The Gaussian elimination with no pivoting operations – rounded to 4 and 3 decimal places, respectively – is shown below.
2. Intermediate Stages of the Matrix Transform: Each step in the picture includes the transformation matrix to the next step, along with the norm of the transformation matrix



1. Exaggeration of the Round-off Effect: Here, the round-off error dominates; the right-hand solution is very wrong, and the left-hand solution is nearly correct. There were two norm-1000 operations, contributing to the error.

**Gaussian Elimination with Pivoting**

1. Pivoting Applied on Gaussian Elimination: This section re-does the sample, but with pivoting. This means that a permutation matrix is used to place the largest absolute-value column head at the top.
2. 3/4 Decimal Place Rounding: Again, use of 4 and 3 decimal places, respectively, produces the following figure.



1. Solutions Approximately Equal and Correct: Here, both solutions are approximately equal and approximately correct. All transformations have norm of the order of 1, which in turn is the case because of the pivoting operation.
2. Pivoting - a Data-dependent Decision: The point is that successful Gaussian elimination *requires* pivoting, which is a *data-dependent decision*. Also, the *permutations* must be stored. Neither of these is a serious detriment in a carefully implemented software system.
3. Solution using Householder Transformation: The next few sections examine Householder transformations not as a *necessity*, but as an *alternative*. It turns out that they will permit somewhat simpler software implementation.

**Householder Transformations**

The next few sections the Householder is derived, and then put to use in applications.

**Geometric Construction**

1. Construction of the Reflection Matrix: One may construct a *reflection matrix* acting on which sends a chosen *axis vector*, , to its negative, and reflects all other vectors through the hyperplane perpendicular to :



1. Decomposition of into Components: A given matrix may be decomposed into its components which are parallel and perpendicular to :



1. Geometric View of the Reflection: Then, if one can subtract off from twice its parallel component , then the following reflection is obtained:



1. Formulation of the Matrix : That is, one wants

Moreover, one would like a single matrix which would transform all ’s.

1. Formulation of the Projection Vector: The projection vector must be in the *direction* of , and it must have the magnitude :
2. Unit Vector of : The following notation is used:
3. Projection Vector using Dot Product #1: It turns out – this is the central trick of the Householder transformation – that one can reformulate this expression in terms of dot products, eliminating . To do this, recall that

So

1. Square Modulus of /: Also recall that

and

1. Projection Vector using Dot Product #2: Now, one has
2. Reflection Matrix acting on : One can use this to re-write the reflection of :
3. Inner Product as a Matrix Product: Recall that the dot-product – or *inner product* – can be written as a matrix product:
4. Setting using Matrix Transform: One may think of a column vector as an matrix, and the row vector as an matrix, which is the transpose of the column vector. So
5. Outer Product as Matrix Multiplication: Now that the dot products can be expressed in terms of matrix multiplication, one may use the associativity matrix multiplication:
6. Statement of the Householder Transformation: The product is the *outer product* of with itself. It is an matrix with ijth entry . One has

from which

**Construction with Specified Source and Destination**

1. Locating the Householder Reflection Axis: The next step is, given and , one wants to find the axis of reflection through which is ’s mirror image. Since Householder transformations are norm-preserving, and have same norm. As shown below, using the head-to-tail method

might be a candidate solution.



1. Verification of Householder Reflection Axis: It may be checked easily that this is correct. If is indeed that reflection axis, then it ought to get reflected to its own negative:

**Properties of , Obtained Algebraically**

1. Algebraic Proof of Householder Properties: This section attempts to prove algebraically certain properties of the Householder transform. The next section does the same geometrically.
2. Orthogonal: i.e.

Applying on itself proves this.

1. Symmetric: is *symmetric*, i.e.

This is easily seen since

This identity is symmetric, has ijth entry

and so is symmetric as well.

1. Involutary: is *involutary*, i.e.

Same as orthogonality, due to symmetry, since

1. Determinant: No easy way to find this algebraically.
2. Matrix Norm: No easy way to find this algebraically.

**Properties of Q, Obtained Geometrically**

1. Geometric Proof of Householder Properties: In this section, certain properties of Householder matrix using geometric methods are proven. Remember that dot products, via

provides with norms as well as angles.

1. is *Involutary*: i.e.

reflects to its mirror image ; a second application of sends it back again.

1. is *Orthogonal*: i.e.

for all vectors , . Since is a reflection, it preserves norm; also, from the picture below, it’s clear that it preserves angles.



1. is *Symmetric*: i.e.

for all vectors , . This is the same geometric argument as for orthogonality, making use of involutary.

1. Determinant: Since is a reflection about the axis, leaving all the axes orthogonal to a fixed, must have determinant . That is, it turns the unit cube inside out along one axis.
2. Matrix Norm: Since is a reflection, it is length-preserving.

**Repeater Householders for Upper-Triangularization**

1. Purpose of Upper Triangular Transformation: The goal of upper triangularization is to put a matrix of the form



into the form



1. One Step as a Time: This may be one step at a time: from



to



to



1. First Step in the Transformation: The first step – for which Householder transformation will be used – is on all of the input matrix entries, with an operation which transforms the first column.
2. Second Step in the Transformation: The second step is on the submatrix obtained by omitting the top row and the left column:



1. Further Steps in the Transformation: One can keep operating on the sub-matrices until there are no more of them left. So, the problem if upper-triangularization reduces to that of modifying the first column of a matrix to have non-zero entries except the entries at the top of the column.

**Householders for Column-Zeroing**

1. Final Objective behind the Transform: The goal is to put the matrix



into the form



1. Application of on : The key insight is that when acts on by left multiplication, the jth column of is the matrix-times-vector product of times the jth column of .
2. Zeroing of First Column: Let be the first column of , and be the first column of . At this point all we know is that we want to have all zeros except for the first entry.
3. Use of the Norm-preserving Nature: Since this section is going to use Householder transformations, which are norm-preserving, one knows that

This means

1. Determining the Householder Matrix : One wants a Householder matrix which sends to and which does what it does with the rest of the matrix as a side effect.
2. Establishing the Reflection Axis : As seen above the reflection axis is

Further, since

and since the task is to compute

one can choose to have the opposite sign of to avoid the cancellation error that happens when one subtracts two nearly equal numbers.

1. Error Analysis: Since the ’s are orthogonal, they are norm-preserving, so

but

as discussed earlier.

1. No Pivoting/Data- dependent Decisions: Also, there is no pivoting involved, and thus – other than the choice of the sign of – there are no *data-dependent decisions*.

**Computation of Determinants**

1. Recap - Determinant of a Householder Matrix: Given a square matrix , one can use repeated Householder transformations to turn it into an upper-triangular matrix . As discussed in the previous sections, is the product of the diagonal entries of divided by the product of the determinants of the transformation matrix.
2. Sign of the Composite Determinant: However, as seen in the earlier sections, Householder matrices have a determinant . Thus, one just has to track whether the number of Householders are odd or even. It takes Householders to upper-triangularize an matrix:



Therefore

**Computation of Matrix Inverses**

1. Use of Householder Upper-Triangularization: Given and invertible square matrix , one can make an augmented matrix just as seen in the earlier section. The Householder upper-triangularization can be used to put the augmented matrix into the upper-triangular form, i.e., row echelon form.
2. Clearing out the Above-diagonal Entries: The rest of the work is done in clearing out the above-diagonal entries of the left-hand side of the augmented matrix, for which the scale and the shear matrices are used, as seen above.

**Rotation Matrices**

1. Connected Components of Householder Matrices: has two connected components: and . The former are rigid *rotations* with determinants ; the latter are *reflections* with determinant . Simon (1995) and Frenkel (2011) contains more information on .
2. Simplified Technique to Compute : The natural question is: If the Householder transformations provide easily understandable, readily computable elements of , then is there another equally pleasant technique to compute elements of ? In fact, there is. One can consult the literature for the Rodrigues formula, and modify that.
3. Rotation as Product of Reflections: Alternatively, one can use the fact that since determinants multiply, the product of two reflections, each having determinants -1, will have determinant +1. Thus, the product will be a rotation.
4. Product of Two Identical Reflections: If a given reflection is composed with itself, the product will be an identity, as seen earlier.
5. Product of Two Distinct Reflections: However, the product of two *distinct* reflections will be a non-trivial rotation.
6. Geometric Visualization of the Reflection Sequence: Let and be given. Since rotations are norm-preserving, one can only hope to rotate into if they have the same norm. This may be helped by computing and . Note that will lie halfway between and .



1. Rotation Resulting from Two Householders: Let

and compute . The let reflect to . Just as in an earlier section, this is a reflection about the axis

Likewise for the second reflection. One then has

**References**

* Burden, R. L., J. D. Faires, and A. M. Burden (2015): *Numerical Analysis 10th Edition* **Cengage Learning** Boston, MA
* Frankel, T. (2011): *The Geometry of Physics: An Introduction 3rd Edition* **Cambridge University Press** Cambridge, UK
* Friedberg, S. H., A. J. Insel, and L. E. Spence (2002): *Linear Algebra 4th Edition* **Pearson** London, UK
* Kerl, J. (2008): [The Householder Transformation in Numerical Linear Algebra](https://johnkerl.org/doc/hh.pdf)
* Simon, B. (1995): *Representations of Finite and Compact Groups* **American Mathematical Society** Providence, RI

**Figure #1**

**Fixed Point Search SKU Flow**

**Iterate Variate**

**Has Search Termination Been Reached?**

**Extract Starting Variate/Bracket**

**Fixed Point Search Initialization**

**NO**

**YES**

**Fixed Point Search Result**

**Figure #2**

**Bracketing SKU Flow**

**Fixed Point Trapped?**

**Trim Bracket Zone to localize Fixed Point**

**Expand Brackets to Trap Fixed Points**

**Bracket Locator Start Specifier**

**NO**

**Figure #3**

**Objective Function Undefined at the Starting Variate**

**Objective Function**

**Variate**

**Starting Variate**

**v0**

**Figure #4**

**Objective Function Undefined at any of the Candidate Variates**

**Variate i**

**Vi**

**Variate i-1**

**Vi-1**

**Variate**

**Variate Scan Jump**

**Objective Function**

**Figure #5**

**General Purpose Bracket Start Locator**

**Is the Objective Function Undefined left?**

**Span the Right Variate**

**Span the Left Variate**

**Bracket Start Locator**

**NO**

**NO**

**Is the Objective Function Undefined right?**

**YES**

**YES**

**Found Starting Point**

**Figure #6**

**Bracketing when Objective Function Validity is Range-bound**

**V0**

**Objective Function**

**Iteration 1**

**V1**

**Variate**

**Iteration i**

**Iteration i-1**

**Iteration 1**

**Vi**

**Vi-1**

**Root**

**Starting Variate**

**Figure #7**

**Objective Function Fixed Point Bracketing**

**Bracketing Iteration #5**

**Bracketing Iteration #4**

**Bracketing Iteration #3**

**Bracketing Iteration #2**

**v3**

**v2**

**v1**

**-v1**

**-v2**

**-v3**

**Variate**

**Objective Function**

**Final Brackets**

**Bracketing Iteration #1**