Numerical Analysis in DROP

v4.62 21 December 2019

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
    a. Bracketing
    b. Objective Function Failure
    c. Bracketing Start Initialization

    d. Open Search Initialization
    e. Search/Bracketing Initializer Customization Heuristics
3. Numerical Challenges in Search
4. Variate Iteration
5. Open Search Methods
    a. Newton's Method
6. Closed Search Methods
    a. Secant
    b. Bracketing Iterative Search
    c. Univariate Iterator Primitive
        i. Bisection
        ii. False Position
        iii. Inverse Quadratic
        iv. Ridder's
    d. Univariate Compound Iterator
        i. Brent's Method
        ii. Zheng's Method
7. Polynomial Root Search
8. References
9. Figures
10. Fixed Point Search Software Components
    a. Execution Initialization
    b. Bracketing
    c. Execution Customization
    d. Fixed Point Search
    e. Variate Iteration
    f. 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
2. The flow behind these steps is illustrated in Figure 1.
3. 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).
2. Figure 2 shows the flow for the Bracketing routine.
3. Bracketing methods require that the initial search interval bracket the root (i.e. the function values at interval end points have opposite signs).

4. 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.
5. Unless the objective function is discontinuous, bracketing methods guarantee convergence (although may not be within the specified iteration limit).
6. Typically, they do not require the objective function to be differentiable.
7. Bracketing iteration primitives' convergence is usually linear to super-linear.
8. Bracketing methods preserve bracketing throughout computation and allow user to specify which side of the convergence interval to select as the root.
9. 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.
10. 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).
11. 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.
12. 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.
2. 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.
3. 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”.
4. 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

4. 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.
5. 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

1. $$\nu_{i+1} = I(\nu_i, \Im_i)$$

   where ν_i is the i[th] variate and $\Im_i$ is the root finder state after the i[th] iteration.
2. 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.
3. 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.
4. Check pointing: Currently implemented using a roving variate/OF iterate node "RunMap"; this is also used to check circularity in the iteration process.
5. Compound Iterator RunMap: For compound iterations, the iteration circularity is determined the doublet $(\nu_i, \Im_i)$, so the Wengert RunMap is really a doublet Multi-Map.
6. 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.
7. 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.

   a. Fast side of the crossover must be customizable (including other Halley's method variants); robust side should also be customizable (say False Position).

8. 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 $\frac{x_{N+1}-x_N}{x_N-x_{N-1}}$ and the relative objective function change $\frac{y_{N+1}-y_N}{y_N-y_{N-1}}$.
9. 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.
     a. Number of objective function evaluation (generally expensive).
     b. Number of variate iterator steps needed.
     c. 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.
10. 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).
11. 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 f and its derivative f' to iteratively evaluate the root.
2. Given a well-behaved function f and its derivative f' defined over real x, the algorithm starts with an initial guess of x_0 for the root.
3. First iteration yields

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

4. This is repeated in

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

   till a value x_n that is convergent enough is obtained.
5. If α is a simple root (root with multiplicity 1), and

$$\epsilon_n = x_n - \alpha$$

   and

$$\epsilon_{n+1} = x_{n+1} - \alpha$$

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

$$\epsilon_{n+1} \approx \frac{1}{2}\left|\frac{f''(x_n)}{f'(x_n)}\right| {\epsilon_n}^2$$

6. Newton's method only works when f has continuous derivatives in the root neighborhood.
7. When analytical derivatives are hard to compute, calculate slope through nearby points, but convergence tends to be linear (like secant).
8. 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.
9. If a stationary point of the function is encountered, the derivative is zero and the method will fail due to division by zero.
10. The stationary point can be encountered at the initial or any of the other iterative points.
11. Even if the derivative is small but not zero, the next iteration will be a far worse approximation.
12. 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).
13. If α is a root with multiplicity

$$m > 1$$

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

$$\epsilon_{n+1} \approx \frac{m-1}{m}\epsilon_n$$

14. 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.

15. However, if the multiplicity m of the root is known, one can use the following modified algorithm that preserves the quadratic convergence rate (equivalent to using successive over-relaxation)

$$x_{n+1} = x_n - m\frac{f(x_n)}{f'(x_n)}$$

16. The algorithm estimates m 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:

$$x_{n+1} = x_n - \frac{f(x_n)f'(x_n)}{f'(x_n)f'(x_n) - f(x_n)f''(x_n)}$$

17. It is easy to show that if

$$f'(x_n) = 0$$

and

$$f''(x_n) \neq 0$$

the convergence in the neighborhood becomes linear. Further, if

$$f'(x_n) \neq 0$$

and

$$f''(x_n) = 0$$

convergence becomes cubic.

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

$$g(x) = x - \frac{f(x)}{f'(x)}$$

$$p_n = g(p_{n-1})$$

where p is a fixed point of g, i.e.

$$g \in C[a, b]$$

k is a positive constant,

$$p_0 \in C[a, b]$$

and

$$g(x) \in C[a, b] \; \forall \; x \in C[a, b]$$

19. One sufficient condition for p_0 to initialize a convergent sequence $\{p_0\}_{k=0}^{\infty}$, which converges to the root

$$x = p$$

of

$$f(x) = 0$$

is that

$$x \in (p - \delta, p + \delta)$$

and that δ be chosen so that

$$\frac{f(x_n)f''(x_n)}{f'(x_n)f'(x_n)} \leq k < 1 \; \forall \; x \in (p - \delta, p + \delta)$$

20. 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 n-cycle can be stable, or the behavior of the sequence can be very complex (forming a Newton fractal).

21. 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.
22. One significant advantage of Newton’s method is that it can be readily generalized to higher dimensions.
23. 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 a and b.
2. It iteratively calculates f at

$$c = \frac{a + b}{2}$$

   then uses c to replace either a or b, depending on the sign. It eventually stops when f has attained the desired tolerance.
3. Bisection relies on f being continuous within the brackets.
4. 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 c is linearly interpolated; f is computed at

$$c = \frac{bf(a) + af(b)}{f(a) + f(b)}$$

where $f(a)$ and $f(b)$ have opposite signs. This holds obvious similarities with the secant method.

2. False position method also requires that f be continuous within the brackets.
3. It is simple enough, more robust than secant and faster than bisection, but convergence is still linear to super-linear.
4. 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

$$f(x) = \frac{(x - x_{n-1})(x - x_n)}{(x_{n-2} - x_{n-1})(x_{n-2} - x_n)} f_{n-2} + \frac{(x - x_{n-2})(x - x_n)}{(x_{n-1} - x_{n-2})(x_{n-1} - x_n)} f_{n-1} + \frac{(x - x_{n-2})(x - x_{n-1})}{(x_n - x_{n-2})(x_n - x_{n-1})} f_n$$

where we use the three iterates, x_{n-2}, x_{n-1} and x_n, with their function values, f_{n-2}, f_{n-1} and f_n.

5. This reduces the number of iterations at the expense of the function point calculations.
6. 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).
7. Further, quadratic or higher fits may also cause complex roots.

Univariate Iterator Primitive: Inverse Quadratic

1. Performing a fit of the inverse $\frac{1}{f}$ instead of f avoids the quadratic interpolation problem above. Using the same symbols as above, the inverse can be computed as

$$\frac{1}{f(y)} = \frac{(y - f_{n-1})(y - f_n)}{(f_{n-2} - f_{n-1})(f_{n-2} - f_n)} x_{n-2} + \frac{(y - f_{n-2})(x - f_n)}{(f_{n-1} - f_{n-2})(f_{n-1} - f_n)} x_{n-1} + \frac{(y - f_{n-2})(y - f_{n-1})}{(f_n - x_{n-2})(f_n - f_{n-1})} x_n$$

2. Convergence is faster than secant, but poor when iterates not close to the root, e.g., if two of the function values f_{n-2}, f_{n-1} and f_n 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 f.
2. Given the bracketing variates, x_1 and x_2, which are on two different sides of the root being sought, the method evaluates f at

$$x_3 = \frac{x_1 + x_2}{2}$$

3. It extracts exponential factor α such that $f(x)e^{\alpha x}$ forms a straight line across $x_1, x_2,$ and x_3. A revised x_2 (named x_4) is calculated from

$$x_4 = x_3 + (x_3 - x_1)\frac{sign[f(x_1) - f(x_2)]f(x_3)}{\sqrt{f^2(x_3) - f(x_1)f(x_2)}}$$

2. Ridder's method is simpler than Brent's method, and has been claimed to perform about the same.
3. 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 a_0 and b_0, two provisional values for the next iterate are computed; the first given by the secant method

$$s = b_k - \frac{b_k - b_{k-1}}{f(b_k) - f(b_{k-1})} f(b_k)$$

   and the second by bisection

$$m = \frac{a_k + b_k}{2}$$

3. If s lies between b_k and m, it becomes the next iterate b_{k+1}, otherwise the m is the next iterate.
4. Then, the value of the new contra-point is chosen such that $f(a_{k+1})$ and $f(b_{k+1})$ have opposite signs.
5. Finally, if

$$|f(a_{k+1})| < |f(b_{k+1})|$$

   then a_{k+1} is probably a better guess for the solution than b_{k+1}, and hence the values of a_{k+1} and b_{k+1} are exchanged.
6. To improve convergence, Brent's method requires that two inequalities must be simultaneously satisfied.
   a) Given a specific numerical tolerance δ, if the previous step used the bisection method, and if

$$\delta < |b_k - b_{k-1}|$$

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

$$\delta < |b_{k-1} - b_{k-2}|$$

b) If the previous step used bisection, if

$$|s - b_k| < \frac{1}{2}|b_k - b_{k-1}|$$

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

$$|s - b_k| < \frac{1}{2}|b_{k-1} - b_{k-2}|$$

is checked instead.

7. Finally, since Brent's method uses inverse quadratic interpolation, s has to lie between $\frac{3a_k+b_k}{4}$ and b_k.
8. Brent's algorithm uses three points for the next inverse quadratic interpolation, or secant rule, based upon the criterion specified above.
9. One simplification to the Brent's method adds one more evaluation for the function at the middle point before the interpolation.
10. This simplification reduces the times for the conditional evaluation and reduces the interval of convergence.
11. 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 $f(x)$ and $f'(x)$ 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):
    a. Meta-heuristics are strategies that guide the search process.
    b. The goal is to efficiently explore the search space in order to find near-optimal solutions.
    c. Techniques that constitute meta-heuristics range from simple local search procedures to complex learning processes.
    d. Meta-heuristic algorithms are approximate and usually non-deterministic.
    e. Meta-heuristics are not problem-specific.
2. Classification: These are taken from Blum and Roli (2003) and from Bianchi, Dorigo, Gambardella, and Gutjahr (2009):
    a. Classification based on the type of the search strategy
    b. Classification off-of single solution search vs. population-based searches
    c. 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:
    a. Robbins and Munro (1951) work on stochastic optimization methods.
    b. Barricelli (1954) carries out the first simulation of the evolutionary process and uses it on general optimization problems.
3. 1960s:
    a. Rastrigin (1963) proposes random search.
    b. Matyas (1965) proposes random optimization.
    c. Nelder and Mead (1965) propose a simplex heuristic, which later shown to converge to non-stationary points on some problems.
    d. Fogel, Owens, and Walsh (1966) propose evolutionary programming.
4. 1970s:
    a. Hastings (1970) proposes the Metropolis-Hastings algorithm.
    b. Cavicchio (1970) proposes adaptation of the control parameters for an optimizer.
    c. Kernighan and Lin (1970) propose a graph-partitioning method that is related to variable-depth search and prohibition based tabu search.
    d. Holland (1975) proposes genetic algorithm.
    e. Glover (1977) proposes scatter search.
    f. Mercer and Sampson (1978) propose a meta-plan for tuning the optimizer's parameters by using another optimizer.
5. 1980s:
    a. Smith (1980) describes genetic programming.
    b. Kirkpatrick, Gelatt, and Vecchi (1983) propose simulated annealing.

   c. Glover (1986) proposes tabu search, along with the first mention of the word meta-heuristic (Yang (2011)).
   d. Moscato (1989) proposes memetic algorithms.

6. 1990s:
   a. Dorigo (1992) introduces ant colony optimization in his PhD thesis.
   b. 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*
- 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.
2. 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).
3. Orthogonal Data Set in the Native Basis Representation: If a data set is orthogonal under a given basis, then

$$\langle x_i x_j \rangle = 0$$

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

$$\langle x_i x_j \rangle \neq 0$$

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

$$\langle x_i' x_j' \rangle = 0$$

(See Figure 3).

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

$$\langle x_i' x_j' \rangle = 0$$

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

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

$$\langle x_i x_j \rangle = 0$$

for all

$$i \neq j$$

then these representations automatically correspond to principal components as well.

7. 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

$$\vec{x}_i \cdot \vec{x}_j = 0$$

if

$$\vec{x}_i \perp \vec{x}_j$$

and covariance is

$$\langle x_i x_j \rangle = 0$$

if x_i and x_j are orthogonal to each other.

2. Distance Metric => Vector Euclidean distance is $\sum[(x_1 - x_2)^2 + (y_1 - y_2)^2 + \cdots]$ is equivalent to the variance $\sum[(x_i - \mu_i)^2 + \cdots]$. 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:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix} + B$$

where A is the rotating transformer, and B is the translator.

2. Scaling vs. Rotation: Say

$$A = \begin{bmatrix} a_{11} & a_{21} \\ a_{21} & a_{22} \end{bmatrix}$$

If

$$a_{11} = a_{22} \neq 1$$

and

$$a_{12} = a_{21} = 0$$

it becomes pure scaling.

$$a_{11} \neq a_{22}$$

and

$$a_{12} = a_{21} = 0$$

produces differential elongation/compression and

$$a_{12} \neq 0$$

or

$$a_{21} \neq 0$$

results in rotation.

3. Uses of Gaussian Elimination:
   - Linearization
   - Orthogonalization
   - Inversion
   - Diagonalization
   - Lower/Upper Triangle Decomposition (LU Decomposition)
   - Independent Component Extraction
   - Principal Component Analysis
4. Diagonal Identity Matrix Conception: Given a matrix A what matrix M should it be transformed by to get a diagonal identity matrix, i.e.

$$MA = I$$

Answer is

$$M = A^{-1}$$

Thus, diagonalization is also an inversion operation.

$$Diagonalization == Orthogonalization == Inversion == ICA$$

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

$$F_{SOURCE,0} = \{f_{ij}\}_{i,j=0}^{n-1}$$

and an initialized target inverse identity matrix

$$F_{INV,0} = \{I\}_{n \times n}$$

achieve a suitable set of p transformations simultaneously on F_{SOURCE} and F_{INV} to eventually make

$$F_{SOURCE,p} = \{I\}_{n \times n}$$

so that the corresponding

$$F_{INV,p} = \{f_{INV,ij}\}_{i,j=0}^{n-1}$$

becomes the inverse.

3. Valid Inversion Rules: These are fairly straight forward application of the Gaussian elimination scheme:
   - Scale a single F_{SOURCE} row by a constant => scale the corresponding F_{INV} row by the same constant.
   - Add/subtract any pair of F_{SOURCE} rows from each other => add/subtract the same pair of F_{INV} rows from each other.
4. 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 $WorkColFactor$ as

$$WorkColFactor = \frac{DiagonalEntry}{CellValueEntry}$$

   - Scan all the cells in the column of the current cell.
   - Apply the $WorkColFactor$ 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 ($\{I\}_{n\times n}$) matrix.
   - Do the same as above to the inverse matrix as well.

5. 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.
6. 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).
7. Rows/Columns as "Preferred Linear Basis Sequence" for Matrix Manipulation: Consider the solution to

$$AX = Z$$

where X and Z 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

$$AX = Z$$

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

8. 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.

9. 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 $row + 1$ - or from

$$row = 0$$

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

$$AX = Z$$

    is now invertible. The following would be the steps:
    - First, re-arrange the equation system set to identify a suitable pair A and Z such that A is invertible, and Z is estimated from

$$AX = Z$$

    - The re-arranging linear operation set will produce A such that

$$BY = Z \Longrightarrow AX = BY \Longrightarrow X = A^{-1}BY$$

11. **More General Inverse Transformation Re-formulation:** Given the coefficient matrix $\{a_{qj}\}_{j,q=0}^{n-1}$, the set of unknown variables $\{x_j\}_{j=0}^{n-1}$, and the RHS $\{y_q\}_{q=0}^{n-1}$, y_p, y_q, and the corresponding z_p and z_q are given as

$$\sum_{j=0}^{n-1} a_{qj} x_j = y_q$$

$$\sum_{j=0}^{n-1} a_{pj} x_j = y_p$$

and

$$\{z_j = y_j\}_{j=0}^{n-1}$$

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

$$\sum_{j=0}^{n-1} (a_{pj} + a_{qj}) x_j = y_p + y_q$$

and therefore

$$\{z_j = y_j\}_{j=0;j\neq q}^{n-1}$$

along with

$$z_q = y_p + y_q$$

This clearly implies that

$$\frac{\partial z_q}{\partial y_p} = 1$$

or

$$B_{qp} = B_{qp} + 1$$

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:

$$x_1 = a_{11}s_1 + a_{12}s_2$$

and

$$x_2 = a_{21}s_1 + a_{22}s_2$$

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

$$a_{11} = 1$$

and

$$a_{12} = 0$$

This results in 2 unknowns. Setting

$$\langle {x_1}^2 \rangle = 1$$

$$\langle {x_2}^2 \rangle = 1$$

and

$$\langle x_1 x_2 \rangle = \rho$$

you get

$${a_{11}}^2 + {a_{12}}^2 = 1$$

and

$${a_{21}}^2 + {a_{22}}^2 = 1$$

resulting in

$$a_{21} = \rho$$

and

$$a_{21} = \sqrt{1 - \rho^2}$$

Thus, all unknowns are determined.

3. nD Orthogonalization:

$$\begin{bmatrix} x_0 \\ \vdots \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} a_{0,0} & \cdots & a_{0,n-1} \\ \vdots & \ddots & \vdots \\ a_{n-1,0} & \cdots & a_{n-1,n-1} \end{bmatrix} \begin{bmatrix} s_0 \\ \vdots \\ s_{n-1} \end{bmatrix}$$

- Number of diagonal entries => n
- Number of non-diagonal entries => $n^2 - n$
- Net Number of equations => Number of diagonal entries + Number of non-diagonal entries =>

$$n + \frac{n(n-1)}{2} = \frac{n(n+1)}{2}$$

4. nD Unknowns Analysis:
   - Number of Unknowns => n^2.
   - Fix the first row, and the number of unknowns becomes => $n^2 - n$.
   - Fixing the row takes off one equation, so the number of equations => $\frac{n(n+1)}{2} - 1$
   - Number of equations = Number of Unknowns =>

$$\frac{n(n+1)}{2} - 1 = n^2 - n$$

   results in

$$(n-1)(n-2) = 0$$

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 $n-1$ rotations, the second row fixation results in $n-2$ rotations, and so on. Thus, the total number of rotating transformations is $\frac{(n-1)(n-2)}{2}$ (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 $\frac{(n-1)(n-2)}{2}$ rotation choices.
5. Number of Sweep Operations: Total result of all the rotations and their corresponding choices =>

$$2 \cdot \frac{(n-1)(n-2)}{2} = (n-1)(n-2)$$

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 μ_0 as an initial eigenvalue guess for the Hermitian matrix A. An initial vector b_0 must also be supplied as the initial eigenvector guess.
3. Iterative Eigenvalue and Eigenvector: Calculate the subsequent approximation of the eigenvector b_{i+1} by

$$b_{i+1} = \frac{[A - \mu_i I]^{-1} b_i}{\|[A - \mu_i I]^{-1} b_i\|}$$

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

$$\mu_i = \frac{{b_i}^* A b_i}{{b_i}^* b_i}$$

4. Deflation Techniques for Successive Eigenvalues: To compute more than one eigenvalue the algorithm can be combined with a deflation technique.
5. 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.
6. 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)

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 A, the algorithm will produce a number λ, which is the greatest – in absolute value – eigenvalue of A, and a non-zero vector v, the corresponding eigenvector of λ, such that

$$Av = \lambda v$$

   The algorithm is also known as the von Mises iteration (von Mises and Pollazek-Geiringer (1929)).
2. 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 A is a very large sparse matrix.

The Method

1. Starting Choice for Principal Eigenvector: The power iteration method starts with a vector b_0, 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

$$b_{k+1} = \frac{Ab_k}{\|Ab_k\|}$$

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

3. Necessary Condition for Eigen-component Convergence: If one assumes that A has an eigenvalue that is greater in magnitude than its other eigenvalues and that the starting vector b_0 has a non-zero component in the direction of an eigenvector associated with the dominant eigenvalue, then a sub-sequence $\{b_k\}$ converges to an eigenvector associated with the dominant eigenvector.
4. Recast of $\{b_k\}$ using Phase Shift: Without the two assumptions above the sequence does not necessarily converge. In this sequence

$$b_k = e^{i\phi_k} v_1 + r_k$$

where v_1 is the eigenvector associated with the dominant eigenvalue, and

$$\|r_k\| \to 0$$

The presence of the term $e^{i\phi_k}$ implies that $\{b_k\}$ does not converge unless

$$e^{i\phi_k} \equiv 1$$

5. Convergence to the Dominant Eigenvalue: Under the two assumptions listed above, the sequence $\{\mu_k\}$ defined by

$$\mu_k = \frac{{b_k}^T A b_k}{{b_k}^T b_k}$$

converges to the dominant eigenvalue.

6. Eigenvector and Eigenvalue Sequences: The vector b_k is the associated eigenvector. Ideally one should use the Rayleigh Quotient in order to get the associated eigenvalue.
7. 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

$$\frac{{b_k}^T A b_k}{{b_k}^T b_k} = \frac{{b_{k+1}}^T b_k}{{b_k}^T b_k}$$

Analysis

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

$$A = VJV^{-1}$$

where the first column of V is an eigenvector of A corresponding to the dominant eigenvalue λ_1.

2. The Principal Component Jordan Block: Since the dominant eigenvalue of A is unique, the first Jordan block of J is the 1×1 matrix $[\lambda_1]$, where λ_1 is the largest eigenvalue of A in magnitude.
3. Initializing the Principal Component Eigenvector: The starting vector b_0 can be written as a linear combination of the columns of V:

$$b_0 = c_1 v_1 + c_2 v_2 + \cdots + c_n v_n$$

By assumption, b_0 has a non-zero component in the direction of the dominant eigenvalue, so

$$c_1 \neq 0$$

4. k[th] Recurrence of the Initial Vector: The computationally recurrence relation for b_{k+1} can be written as:

$$b_{k+1} = \frac{Ab_k}{\|Ab_k\|} = \frac{A^{k+1}b_0}{\|A^{k+1}b_0\|}$$

where the expression $\frac{A^{k+1}b_0}{\|A^{k+1}b_0\|}$ is amenable to the analysis below.

5. Principal Component Dependence of b_k:

$$\begin{aligned} b_k = \frac{A^k b_0}{\|A^k b_0\|} = \frac{[VJV^{-1}]^k b_0}{\|[VJV^{-1}]^k b_0\|} &= \frac{VJ^kV^{-1}b_0}{\|VJ^kV^{-1}b_0\|} \\ &= \frac{VJ^kV^{-1}(c_1v_1 + c_2v_2 + \cdots + c_nv_n)}{\|VJ^kV^{-1}(c_1v_1 + c_2v_2 + \cdots + c_nv_n)\|} \\ &= \frac{VJ^k(c_1\hat{e}_1 + c_2\hat{e}_2 + \cdots + c_n\hat{e}_n)}{\|VJ^k(c_1\hat{e}_1 + c_2\hat{e}_2 + \cdots + c_n\hat{e}_n)\|} \\ &= \left(\frac{\lambda_1}{|\lambda_1|}\right)^k \frac{c_1}{|c_1|} \frac{v_1 + \frac{1}{c_1}V\left(\frac{1}{\lambda_1}J\right)^k (c_2\hat{e}_2 + \cdots + c_n\hat{e}_n)}{\left\|v_1 + \frac{1}{c_1}V\left(\frac{1}{\lambda_1}J\right)^k (c_2\hat{e}_2 + \cdots + c_n\hat{e}_n)\right\|} \end{aligned}$$

6. Scaling Down the Principal Eigenvector: As

$$k \to \infty$$

the expression above simplifies to

$$\left(\frac{1}{\lambda_1}J\right)^k = \begin{pmatrix} [\mathbb{I}] & \cdots & & \\ \cdots & \left(\frac{1}{\lambda_1}J_2\right)^k & \cdots & \cdots \\ & \cdots \quad \cdots & \ddots & \cdots \\ & \cdots \quad \cdots & \cdots & \left(\frac{1}{\lambda_1}J_n\right)^k \end{pmatrix} \to \begin{pmatrix} [\mathbb{I}] & \cdots & \cdots & \cdots \\ \cdots & 0 & & \\ \cdots & \cdots & \ddots & \cdots \\ \cdots & \cdots & \cdots & 0 \end{pmatrix}$$

7. Limit of $\left(\frac{1}{\lambda_1}J_i\right)^k$ as $k \to \infty$: The limit follows from the fact that the eigenvalue of $\left(\frac{1}{\lambda_1}J_i\right)^k$ is less than 1 in magnitude, so

$$\left(\frac{1}{\lambda_1}J_i\right)^k \to 0$$

as

$$k \to \infty$$

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

$$\frac{1}{c_1}V\left(\frac{1}{\lambda_1}J\right)^k (c_2\hat{e}_2 + \cdots + c_n\hat{e}_n) \to 0$$

as

$$k \to \infty$$

9. Reduction of b_k for $k \to \infty$: Using this fact, b_k can be written in a form that emphasizes its relationship with v_1 when k is large:

$$b_k = \left(\frac{\lambda_1}{|\lambda_1|}\right)^k \frac{c_1}{|c_1|} \frac{v_1 + \frac{1}{c_1} V \left(\frac{1}{\lambda_1} J\right)^k (c_2 \hat{e}_2 + \cdots + c_n \hat{e}_n)}{\left\| v_1 + \frac{1}{c_1} V \left(\frac{1}{\lambda_1} J\right)^k (c_2 \hat{e}_2 + \cdots + c_n \hat{e}_n) \right\|} = e^{i\phi_k} \frac{c_1}{|c_1|} \frac{v_1}{\|v_1\|} + r_k$$

where

$$e^{i\phi_k} = \left(\frac{\lambda_1}{|\lambda_1|}\right)^k$$

and

$$\|r_k\| \to 0$$

as

$$k \to \infty$$

10. Uniqueness of the $\{b_k\}$ Sequence: The sequence $\{b_k\}$ 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 $\{b_k\}$ may not converge, b_k is nearly an eigenvector of A for large k.
11. Alternative Proof of the Sequence Convergence: Alternatively, if A is diagonalizable, then the following proof yields the same result. Let $\lambda_1, \lambda_2, \cdots, \lambda_m$ be the m eigenvalues – counted with multiplicity – of A, and let $v_1, v_2, \cdots, v_m$ be the corresponding eigenvectors. Suppose that λ_1 is the dominant eigenvalue, so that

$$|\lambda_1| > |\lambda_j|$$

for

$$j > 1$$

12. Suitable Choice for the Initial Vector: As seen earlier, the initial vector b_0 is written as

$$b_0 = c_1 v_1 + c_2 v_2 + \cdots + c_n v_n$$

If b_0 is chosen randomly – with uniform probability – the

$$c_1 \neq 0$$

with probability 1

13. Power Iteration over Initial Eigenvector: Now

$$A^k b_0 = c_1 A^k v_1 + c_2 A^k v_2 + \cdots + c_n A^k v_n = c_1 {\lambda_1}^k v_1 + c_2 {\lambda_2}^k v_2 + \cdots + c_n {\lambda_n}^k v_n$$
$$= c_1 {\lambda_1}^k \left[v_1 + \frac{c_2}{c_1} \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 + \cdots + \frac{c_n}{c_1} \left(\frac{\lambda_n}{\lambda_1} \right)^k v_n \right] \rightarrow c_1 {\lambda_1}^k v_1$$

as long as

$$\left| \frac{\lambda_j}{\lambda_1} \right| < 1$$

for

$$j > 1$$

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

$$b_k = \frac{A^k b_0}{\|A^k b_0\|}$$

Therefore b_k converges to a multiple of the eigenvector v_1

15. Convergence Ratio of Power Iteration: The convergence ratio is geometric with the ratio $\left|\frac{\lambda_2}{\lambda_1}\right|$ where λ_2 denotes the second principal eigenvalue.
16. 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 A explicitly, but can instead access a function evaluating matrix-vector products Ax.
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 A^{-1}.
7. Sub-space of Eigenvalues - Krylov Subspace: Other algorithms look at the whole subspace generated by the vectors b_k. 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
- Ipsen, I., and R. Wills (2005): Analysis and Computation of Google's PageRank
- 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 $f(A)$ of a matrix A as a polynomial in A, in terms of the eigenvectors and eigenvalues of A (Claerbout (1985), Horn and Johnson (1991), Wikipedia (2019))).
2. In Terms of Eigenvalues and Eigenvectors: It states that (Sylvester (1883))

$$f(A) = \sum_{i=1}^{k} f(\lambda_i) A_i$$

where the λ_i are the eigenvalues of A, and the matrices

$$A_i = \prod_{\substack{j=1 \\ j \neq i}}^{k} \frac{1}{\lambda_i - \lambda_j} (A - \lambda_j I)$$

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

Conditions

Sylvester's formula applies for any diagonalizable matrix A with k distinct eigenvalues $\lambda_1, \cdots, \lambda_k$ and any function f defined on some subset of complex numbers such that $f(A)$

is well defined. The last condition means that every eigenvalue λ_i is in the domain of f, and that every eigenvalue λ_i with multiplicity

$$m_i > 1$$

is in the interior of the domain with f being $m_i - 1$ times differentiable at λ_i (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:

$$f(A) = \sum_{i=1}^{s} \left[\sum_{j=0}^{n_i-1} \frac{1}{j!} \phi_i{}^{(j)}(\lambda_i)(A - \lambda_j I)^j \prod_{\substack{j=1 \\ j \neq i}}^{s} (A - \lambda_j I)^{n_j} \right]$$

   where

$$\phi_i{}^{(j)} := \frac{f(t)}{\prod_{\substack{j=1 \\ j \neq i}}^{s} (t - \lambda_j)^{n_j}}$$

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

$$f(A) = \sum_{i=1}^{s} A_i \sum_{j=0}^{n_i-1} \frac{1}{j!} f^{(j)}(\lambda_i)(A - \lambda_i I)^j$$

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

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 $\int_a^b f(x)dx$ to a given degree of accuracy.
6. Accurately Estimating the Definite Integral: If $f(x)$ 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 $f(x)$ 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

$$f(x) = e^{-x^2}$$

   the anti-derivative of which – the error function times a constant – cannot be written in an elementary form.
3. 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 0 – that passes through the point $\left[\frac{a+b}{2}, f\left(\frac{a+b}{2}\right)\right]$ This is the *mid-point rule* of the *rectangle rule*:

$$\int_a^b f(x)dx \approx (b-a)f\left(\frac{a+b}{2}\right)$$

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 $[a, f(a)]$ and $[b, f(b)]$. This is called the *trapezoidal rule*:

$$\int_a^b f(x)dx \approx (b-a)\left[\frac{f(a)+f(b)}{2}\right]$$

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 $[a, b]$ into some number n 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

$$\int_a^b f(x)dx \approx (b-a)\left[\frac{f(a)+f(b)}{2} + \sum_{k=1}^{n-1} f\left(a + k\frac{b-a}{n}\right)\right]$$

where the sub-intervals have the form

$$[a + kh, a + (k + 1)h] \subset [a, b]$$

with

$$h = \frac{b - a}{n}$$

and

$$k = 0, \cdots, n - 1$$

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

6. Definition of Newton-Cotes Formula: Interpolation with polynomials evaluated at equally spaced points in $[a, b]$ 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 2, is also a Newton-Cotes formula.
7. 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.

8. 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.
9. 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.
10. Other Varying Interval Quadrature Rules: Other quadrature methods with varying intervals include Clenshaw-Curtis quadrature methods – also called Fejer quadrature – and these do nest.
11. 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

$$\int_0^1 f(x)dx = \sum_{m=1}^{M} \sum_{n=0}^{\infty} \frac{(-1)^n + 1}{(2M)^{n+1}(n+1)!} \frac{d^n y}{dx^n}\Big\|_{x=\frac{m-\frac{1}{2}}{M}}$$

or

$$\int_0^1 f(x)dx = \underset{N \to \infty}{Limit} \sum_{m=1}^{M} \sum_{n=0}^{N} \frac{(-1)^n + 1}{(2M)^{n+1}(n+1)!} \frac{d^n y}{dx^n}\Big\|_{x=\frac{m-\frac{1}{2}}{M}}$$

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

$$M = 1$$

and

$$f(x) = \frac{\theta}{1 + \theta^2 x^2}$$

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

$$\tan^{-1} z = i \sum_{n=1}^{\infty} \frac{1}{2n-1} \left[\frac{1}{\left(1 + \frac{2i}{z}\right)^{2n-1}} - \frac{1}{\left(1 - \frac{2i}{z}\right)^{2n-1}} \right]$$

$$= 2 \sum_{n=1}^{\infty} \frac{1}{2n-1} \frac{a_n(z)}{{a_n}^2(z) + {b_n}^2(z)}$$

where

$$i = \sqrt{-1}$$

is the imaginary unit, and

$$a_1(z) = \frac{2}{z}$$

$$b_1(z) = 1$$

$$a_n(z) = a_{n-1}(z)\left[1 - \frac{4}{z^2}\right] + 4\frac{b_{n-1}(z)}{z}$$

$$b_n(z) = b_{n-1}(z)\left[1 - \frac{4}{z^2}\right] - 4\frac{a_{n-1}(z)}{z}$$

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

$$(-1)^n + 1 = 0$$

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

$$\int_0^1 f(x)dx = 2\sum_{m=1}^{M}\sum_{n=0}^{\infty}\frac{1}{(2M)^{2n+1}(2n+1)!}\frac{d^{2n}y}{dx^{2n}}\Bigg|_{x=\frac{m-\frac{1}{2}}{M}}$$

4. Transforming (a, b) Limits into $(0, 1)$: For a function $g(t)$ defined over the interval (a, b) its integral is

$$\int_a^b g(t)dt = \int_0^{b-a} g(\tau + a)d\tau = (b-a)\int_0^1 g([b-a]x + a)dx$$

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

$$f(x) = g([b-a]x + a)dx$$

Adaptive Algorithms

1. Lack of Suitable Derivative Points: If $f(x)$ 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 $f(x)$ 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 $t \cdot h$, where t, a real number, is the tolerance one wishes to set for the global error. However, if h 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 t.
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 $[a, b]$: Let f have a bounded first derivative over $[a, b]$, i. e.,

$$f \in \mathbb{C}^1(a, b)$$

The mean-value theorem for f where

$$x \in (a, b]$$

gives

$$(x - a)\frac{df(\xi_x)}{dx} = f(x) - f(a)$$

for some

$$\xi_x \in (a, x]$$

depending on x.

2. Integrating Over x on Both Sides: On integrating in x from a to b on both sides and taking the absolute values, one obtains

$$\left| \int_a^b f(x)dx - (b-a)f(a) \right| \leq \left| \int_a^b (x-a) \frac{df(v_x)}{dx} dx \right|$$

3. 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 $\frac{df}{dx}$ by an upper bound:

$$\left| \int_a^b f(x)dx - (b-a)f(a) \right| \leq \frac{(b-a)^2}{2} \sup_{a \leq x \leq b} \left| \frac{df}{dx} \right|$$

where the supremum has been used for the approximation.

4. Corresponding Approximation Applied to the LHS: Hence, if the integral $\int_a^b f(x)dx$ is approximated by the quadrature $(b-a)f(a)$, the error is no greater than the right-hand side above.
5. Converting the RHS into a Riemann Sum: This can be converted into an error analysis for the Riemann sum, giving an upper bound of

$$\frac{n^{-1}}{2} \sup_{0 \le x \le 1} \left| \frac{df}{dx} \right|$$

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

$$f(x) = x$$

6. 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 f. This analysis gives a strict upper bound on the error, if the derivatives of f are available.
7. 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

$$\int_{-\infty}^{+\infty} f(x)dx = \int_{-1}^{+1} f\left(\frac{1}{1-t^2}\right)\frac{1+t^2}{(1-t^2)^2}dt$$

and for semi-infinite intervals one could use

$$\int_{a}^{+\infty} f(x)dx = \int_{0}^{+1} f\left(a+\frac{t}{1-t}\right)\frac{1}{(1-t)^2}dt$$

$$\int_{-\infty}^{a} f(x)dx = \int_{0}^{+1} f\left(a-\frac{1-t}{t}\right)\frac{1}{t^2}dt$$

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 n (Briol, Oates, Girolami, and Osborne (2015)).

Connections to Differential Equations

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

$$F(x) = \int_{a}^{x} f(u)du$$

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

2. Differential Form of the ODE: By differentiating both sides of the above with respect to the argument x, it can be seen that the function F satisfies

$$\frac{dF(x)}{dx} = f(x)$$

$$F(a) = 0$$

3. 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.
4. Separating the Independent/Dependent Variables: The differential equation

$$F(x) = f(x)$$

has a special form; the right-hand side contains only the dependent variable (here x) and not the independent variable (here F). 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 **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

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 $2n-1$ or less using a suitable choice of nodes x_i and weights w_i for

$$i = 1, \cdots, n$$

3. Gaussian Quadrature Nodes and Weights: The most common domain of integration for such a rule is taken as $[-1, +1]$, so the rule can be stated as

$$\int_{-1}^{+1} f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

   which is exact for polynomials of degree $2n-1$ or less. This exact rule is known as the Gauss-Legendre quadrature rule.
4. Approximating $f(x)$ by a $2n-1$ Polynomial: The quadrature rule will only be an approximation to the integral above if $f(x)$ is well approximated by a polynomial of degree $2n-1$ or less in $[-1, +1]$

5. Integrands with End-Point Singularities: The Gauss-Legendre quadrature rule is not typically used for integrable functions with end-point singularities.
6. Alternative Specification of the Quadrature Rules: Instead, of the integrand can be written as

$$f(x) = (1-x)^{\alpha}(1+x)^{\beta}g(x)$$

$$\alpha, \beta > -1$$

where $g(x)$ is well-approximated by a low-degree polynomial, then the alternative nodes x_i' and weights w_i' will usually give more accurate quadrature rules.

7. The Gauss-Jacobi Quadrature Rules: These are known as Gauss-Jacobi quadrature rules, i.e.,

$$f(x) = (1-x)^{\alpha}(1+x)^{\beta}g(x) \approx \sum_{i=1}^{n} w_i' f(x_i')$$

8. Gauss-Chebyshev Quadrature Weights: Common weights include $\frac{1}{\sqrt{1-x^2}}$ - referred to as Chebyshev-Gauss – and $\sqrt{1-x^2}$.
9. 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.
10. 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 x_i 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 $f(x)$ is well-approximated by polynomials in $[-1, +1]$, the associated orthogonal polynomials are Legendre polynomials, denoted by $P_n(x)$.
2. Weights of the Legendre Terms: With n^{th} polynomial normalized to give

$$P_n(1) = 1$$

the i^{th} Gauss node, x_i, is the i^{th} root of P_n, and the weights are given by the formula (Abramowitz and Stegun (2007))

$$w_i = \frac{2}{(1 - x_i)^2 [P_n{}'(x_i)]^2}$$

3. Low Order Nodes and Weights: Some low order quadrature rules are tabulated below over $[-1, +1]$. The next section contains other intervals.

Number of Points	**Points x_i**	**Approximate x_i**	**Weight w_i**	**Approximate w_i**
1	0	0	2	2
2	$\pm\sqrt{\frac{1}{3}}$	± 0.57735	1	1
3	0	0	$\frac{8}{9}$	0.888889
	$\pm\sqrt{\frac{3}{5}}$	± 0.774597	$\frac{5}{9}$	0.555556

4	$\pm\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}$	± 0.339981	$\frac{18+\sqrt{30}}{36}$	0.652145
	$\pm\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$	± 0.861136	$\frac{18-\sqrt{30}}{36}$	0.347855
5	0	0	$\frac{128}{225}$	0.568889
	$\pm\frac{1}{3}\sqrt{5-2\sqrt{\frac{10}{7}}}$	± 0.538469	$\frac{322+13\sqrt{70}}{900}$	0.478629
	$\pm\frac{1}{3}\sqrt{5+2\sqrt{\frac{10}{7}}}$	± 0.906180	$\frac{322-13\sqrt{70}}{900}$	0.236927

Change of Interval

1. Interval Change - $[a, b]$ to $[-1, +1]$: The integral over $[a, b]$ must be changed to the integral over $[-1, +1]$ before applying the Gaussian quadrature rule. The change of interval can be done in the following way:

$$\int_a^b f(t)dt = \frac{b-a}{2}\int_{-1}^{+1} f\left(\frac{b-a}{2}x + \frac{b+a}{2}\right)dx$$

2. Application of Gaussian Quadrature Rules: Applying the Gaussian quadrature rule then results in the following approximation:

$$\int_a^b f(t)dt \approx \frac{b-a}{2}\sum_{i=1}^{n} w_i f\left(\frac{b-a}{2}x_i + \frac{b+a}{2}\right)$$

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 ω into the integrand, and allowing an interval other than $[-1, +1]$. That is, the problem is to calculate $\int_a^b \omega(x)f(x)dx$ for some choices of a, b, c, and ω.
2. Parameter Choices for the Quadrature Generation: For

$$a = -1$$

$$b = 1$$

and

$$\omega(x) = 1$$

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

Interval	$\boldsymbol{\omega(x)}$	**Orthogonal Polynomials**
$[-1, +1]$	1	Legendre Polynomials
$(-1, +1)$	$(1-x)^{\alpha}(1+x)^{\beta}, \alpha, \beta > -1$	Jacobi Polynomials
$(-1, +1)$	$\frac{1}{\sqrt{1-x^2}}$	Chebyshev Polynomials (First Kind)

$[-1,+1]$	$\sqrt{1-x^2}$	Chebyshev Polynomials (Second Kind)
$[0,\infty)$	e^{-x}	Laguerre Polynomials
$[0,\infty)$	$x^{\alpha}e^{-x}, \alpha > -1$	Generalized Laguerre Polynomials
$(-\infty,+\infty)$	e^{-x^2}	Hermite Polynomials

Fundamental Theorem

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

$$\int_a^b \omega(x)x^k p_n(x)dx = 0$$

for all

$$k = 0, \cdots, n-1$$

2. Nodes as Roots of the Polynomial: If the n nodes x_i are picked to be the zeros of p_n, then there exist n weights w_i which make the Gauss quadrature computed integral exact for polynomials $h(x)$ of degree $2n-1$ or less. Furthermore, all these nodes x_i lie in the open interval (a, b) (Stoer and Bulirsch (2002)).
3. Chosen as the Orthogonal Polynomial: The polynomial p_n is said to be an orthogonal polynomial of degree n associated with the weight function $\omega(x)$. It is unique to a constant normalization factor.

4. Decomposing $h(x)$ into Orthogonal Polynomials: The idea underlying the proof is that, because of its sufficiently low degree, $h(x)$ can be divided by $p_n(x)$ to produce a quotient $q(x)$ strictly lower than n, and a remainder of still lower degree, so that both will be orthogonal to $p_n(x)$, by the defining property of $p_n(x)$. Thus,

$$\int_a^b \omega(x)h(x)dx = \int_a^b \omega(x)r(x)dx$$

5. Quadrature Over Reduced Degree Polynomial: Because of the choice of nodes x_i, the corresponding relation

$$\sum_{i=1}^{n} w_i h(x_i) = \sum_{i=1}^{n} w_i r(x_i)$$

also holds. The exactness of the computed integral $h(x)$ then follows from the exactness of $r(x)$, i.e., for polynomials of degree n or less.

General Formula for the Weights

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

$$w_i = \frac{a_n}{a_{n-1}} \frac{\int_a^b \omega(x){p_{n-1}}^2(x)dx}{{p_n}'(x_i)p_{n-1}(x_i)}$$

where a_k is the coefficient of x^k in $p_n(x)$.

2. Lagrange Polynomial Form for $r(x)$: To prove this, note that using the Lagrange interpolation, one can express $r(x)$ in terms of $r(x_i)$ as

$$r(x) = \sum_{i=1}^{n} r(x_i) \prod_{\substack{1 \le j \le n \\ j \ne i}} \frac{x - x_j}{x_i - x_j}$$

because $r(x)$ has a degree less than n and is thus fixed by the value it attains at n different points.

3. Re-evaluation of the $r(x)$ Quadrature: Multiplying both sides by $\omega(x)$ and integrating from a to b yields

$$\int_a^b \omega(x) r(x) dx = \sum_{i=1}^{n} r(x_i) \int_a^b \omega(x) \prod_{\substack{1 \le j \le n \\ j \ne i}} \frac{x - x_j}{x_i - x_j} dx$$

4. Quadrature Weights Using Lagrange Polynomials: The weights w_i are thus given by

$$w_i = \int_a^b \omega(x) \prod_{\substack{1 \le j \le n \\ j \ne i}} \frac{x - x_j}{x_i - x_j} dx$$

5. Quadrature Weights Using Orthogonal Polynomials: This integral expression for w_i can be expressed in terms of the orthogonal polynomials $p_n(x)$ and $p_{n-1}(x)$ as follows.
6. Orthogonal Polynomial from Lagrange Numerator: One can write

$$\prod_{\substack{1 \le j \le n \\ j \ne i}} (x - x_j) = \frac{\prod_{1 \le j \le n} (x - x_j)}{x - x_i} = \frac{p_n(x)}{a_n (x - x_i)}$$

where a_n is the coefficient of x^n in $p_n(x)$.

7. Orthogonal Derivative from Lagrange Denominator: Taking the limit of x to x_i yields, using L'Hopital's rule

$$\prod_{\substack{1\le j\le n\\ j\neq i}} (x_i - x_j) = \frac{{p_n}'(x_i)}{a_n}$$

8. Weights from Polynomials and Derivatives: The integral expression for the weights can thus be written as

$$w_i = \frac{1}{{p_n}'(x_i)} \int_a^b \omega(x) \frac{p_n(x)}{x - x_i} dx$$

9. Reducing the Degree of Integrand: In the integrand, writing

$$\frac{1}{x - x_i} = \frac{1 - \left(\frac{x}{x_i}\right)^k}{x - x_i} + \left(\frac{x}{x_i}\right)^k \frac{1}{x - x_i}$$

yields

$$\int_a^b \omega(x) x^k \frac{p_n(x)}{x - x_i} dx = {x_i}^k \int_a^b \omega(x) \frac{p_n(x)}{x - x_i} dx$$

provided

$$k \le n$$

because $\frac{1-\left(\frac{x}{x_i}\right)^k}{x-x_i}$ is a polynomial of degree $k - 1$ which is then orthogonal to $p_n(x)$.

10. Product of Lower Degree Polynomials: So, if $t(x)$ is a polynomial of at most degree n, one has

$$\int_a^b \omega(x)\frac{p_n(x)}{x-x_i}dx = \frac{1}{t(x_i)}\int_a^b \omega(x)\frac{t(x)p_n(x)}{x-x_i}dx$$

11. $\frac{p_n(x)}{x-x_i}$ as $n-1$ Degree Polynomial: The integral on the right-hand side can be evaluated for

$$t(x) = p_{n-1}(x)$$

as follows. Because $\frac{p_n(x)}{x-x_i}$ is a polynomial of degree $n-1$, one has

$$\frac{p_n(x)}{x-x_i} = a_n x^{n-1} + s(x)$$

where $s(x)$ is a polynomial of degree $n-2$.

12. $n-1$ Polynomial in the Weight Integral: Since $s(x)$ is orthogonal to $p_{n-1}(x)$, one has

$$\int_a^b \omega(x)\frac{p_n(x)}{x-x_i}dx = \frac{a_n}{p_{n-1}(x_i)}\int_a^b \omega(x)p_{n-1}(x)x^{n-1}dx$$

13. x^{n-1} in Terms of p_{n-1}: One can then write

$$x^{n-1} = \left[x^{n-1} - \frac{p_{n-1}(x)}{a_{n-1}}\right] + \frac{p_{n-1}(x)}{a_{n-1}}$$

The term in the brackets is a polynomial of degree $n-2$, which is therefore orthogonal to $p_{n-1}(x)$.

14. Weight Integral in Terms of ${p_{n-1}}^2$: The integral can thus be written as

$$w_i = \frac{a_n}{a_{n-1} p_{n-1}(x_i)} \int_a^b \omega(x) {p_{n-1}}^2(x) dx$$

15. Recovery of the Postulated Expression: According to

$$w_i = \frac{1}{{p_n}'(x_i)} \int_a^b \omega(x) \frac{p_n(x)}{x - x_i} dx$$

the weights are obtained by dividing the weight integral above by ${p_n}'(x_i)$, and that yields the expression

$$w_i = \frac{a_n}{a_{n-1}} \frac{\int_a^b \omega(x) {p_{n-1}}^2(x) dx}{{p_n}'(x_i) p_{n-1}(x_i)}$$

16. Alternate Expression Using the p_{n+1} Term: w_i can also be expressed in terms of the orthogonal polynomials $p_n(x)$ and $p_{n+1}(x)$. In a 3-term recurrence relation (see below)

$$p_{n+1}(x_i) = \rho_a p_n(x_i) + \rho_b p_{n-1}(x_i)$$

the $p_n(x_i)$ term vanishes, so $p_{n-1}(x_i)$ in

$$w_i = \frac{a_n}{a_{n-1}} \frac{\int_a^b \omega(x) {p_{n-1}}^2(x) dx}{{p_n}'(x_i) p_{n-1}(x_i)}$$

can be replaced by $\frac{p_{n+1}(x_i)}{\rho_b}$.

Proof that the Weights are Positive

1. Remainder – Lagrange Squared Term Polynomial: Consider the following polynomial of degree $2n-2$

$$l(x) = \prod_{\substack{1 \le j \le n \\ j \ne i}} \left(\frac{x - x_j}{x_i - x_j} \right)^2$$

   where, as above, the x_j are the roots of the polynomial $p_n(x)$.

2. Gaussian Quadrature for the above Polynomial: Clearly

$$l(x_j) = \delta_{ij}$$

   Since the degree $l(x)$ is less than $2n-1$, the Gaussian quadrature formula involving the weights and the nodes obtained from $p_n(x)$ applies.

3. Weights Have to be Positive: Since

$$l(x_j) = 0$$

   for j not equal to i, one has

$$\int_a^b \omega(x) l(x) dx = \sum_{j=1}^{N} w_j l(x_j) = \sum_{j=1}^{N} w_j \delta_{ij} = w_i > 0$$

   Since both $\omega(x)$ and $l(x)$ are non-negative functions, it follows that

$$w_i > 0$$

Computation of Gaussian Quadrature Rules

There are many algorithms for computing the nodes x_i and the weights w_i of Gaussian quadrature rules. The popular are the Golub-Welsch algorithm requiring $\mathcal{O}(n^2)$ operations, Newton's method for solving

$$p_n(x) = 0$$

using the three-term recurrence relation for evaluation requiring $\mathcal{O}(n^2)$ operations, and asymptotic formulas for large n requiring $\mathcal{O}(n)$ operations.

Recurrence Relation

1. Recurrence Relation for Orthogonal Polynomials: Orthogonal polynomials p_r with

$$[\![p_r, p_s]\!] = 0$$

for

$$r \neq s$$

for a scalar product $[\![\cdot]\!]$,

$$Degree(p_r) = r$$

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

$$p_{r+1}(x) = (x - a_{r,r})p_r(x) - a_{r,r-1}p_{r-1}(x) - \cdots - a_{r,0}p_0(x)$$

where the scalar product is defined as

$$[\![p_r(x), p_s(x)]\!] = \int_a^b \omega(x)p_r(x)p_s(x)dx$$

for

$$r = 0, \cdots, n-1$$

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

$$a_{r,s} = \frac{[\![xp_r, p_s]\!]}{[\![p_r, p_s]\!]}$$

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

$$p_0(x) = 1$$

have a leading coefficient of one and the correct degree (i.e. 0). Setting the starting polynomial by p_0, the orthogonality of p_r can be demonstrated by induction.

3. Proof for the First Term: For

$$r = s = 0$$

one has

$$[\![p_1, p_0]\!] = (x - a_{0,0})[\![p_0, p_0]\!] = [\![xp_0, p_0]\!] - a_{0,0}[\![p_0, p_0]\!] = [\![xp_0, p_0]\!] - [\![xp_0, p_0]\!]$$
$$= 0$$

4. Proof for an Arbitrary Term: Now, if $p_0, \cdots, p_r$ are orthogonal, so is p_{r+1}, because in

$$[\![p_{r+1}, p_s]\!] = [\![xp_r, p_s]\!] - a_{r,r}[\![p_r, p_s]\!] - a_{r,r-1}[\![p_{r-1}, p_s]\!] - \cdots - a_{r,0}[\![p_0, p_0]\!]$$

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

$$[\![p_{r+1}, p_s]\!] = [\![xp_r, p_s]\!] - a_{r,s}[\![p_r, p_s]\!] = [\![xp_r, p_s]\!] - [\![xp_r, p_s]\!] = 0$$

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

$$[\![xp_r, p_s]\!] = [\![p_r, xp_s]\!]$$

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

6. Serial Zeroing of Recurrence Terms: For

$$s < r - 1$$

xp_s is a polynomial of degree less than or equal to $r - 1$. On the other hand, p_r is orthogonal to every polynomial of degree less than or equal to $r - 1$. Therefore, one has

$$[\![xp_r, p_s]\!] = [\![p_r, xp_s]\!] = 0$$

and

$$a_{r,s} = 0$$

for

$$s < r - 1$$

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

$$p_{r+1}(x) = \big(x - a_{r,r}\big)p_r(x) - a_{r,r-1}p_{r-1}(x)$$

or, with the convention

$$p_{-1}(x) \equiv 0$$

$$p_{r+1}(x) = (x - a_r)p_r(x) - b_r p_{r-1}(x)$$

where

$$a_r \doteq \frac{[\![xp_r, p_r]\!]}{[\![p_r, p_r]\!]}$$

$$b_r \doteq \frac{[\![xp_r, p_{r-1}]\!]}{[\![p_{r-1}, p_{r-1}]\!]} = \frac{[\![p_r, p_r]\!]}{[\![p_{r-1}, p_{r-1}]\!]}$$

where the final step occurs because

$$[\![xp_r, p_{r-1}]\!] = [\![p_r, xp_{r-1}]\!] = [\![p_r, p_r]\!]$$

since xp_r differs from p_r by a degree less than r.

The Golub-Welsch Algorithm

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

$$J\tilde{P} = x\tilde{P} - P_n(x) \times \hat{e}_n$$

where

$$\tilde{P} = [P_0(x), \cdots, P_{n-1}(x)]^T$$

$\hat{e}_n$ is the n^{th} standard basis vector, i.e.

$$\hat{e}_n = [0, \cdots, 0, 1]^T$$

and J is the so-called Jacobi matrix.

$$J = \begin{bmatrix} a_0 & 1 & 0 & \cdots & \cdots & \cdots \\ b_1 & a_1 & 1 & 0 & \cdots & \cdots \\ 0 & b_2 & a_2 & 1 & 0 & \cdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ \cdots & \cdots & 0 & b_{n-2} & a_{n-2} & 1 \\ \cdots & \cdots & \cdots & 0 & b_{n-1} & a_{n-1} \end{bmatrix}$$

2. The Golub-Welsch Algorithm: The zeroes x_j of the polynomials upto degree n, 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.
3. Elements of the Tri-diagonal Matrix: For computing the weights and the nodes, it is preferable to consider the symmetric tridiagonal matrix $\mathcal{J}$ with elements

$$\mathcal{J}_{i,i} = J_{i,i} = a_{i-1}$$

$$i = 1, \cdots, n$$

$$\mathcal{J}_{i-1,i} = \mathcal{J}_{i,i-1} = \sqrt{J_{i,i-1}J_{i-1,i}} = \sqrt{b_{i-1}}$$

$$i = 2, \cdots, n$$

4. Quadrature Nodes from the Eigenvalues: J and $\mathcal{J}$ are similar matrices, and therefore have the same eigenvalues – the quadrature nodes.
5. Quadrature Weights Extracted from Eigenvectors: The weights can be computed from the corresponding eigenvectors. If ϕ_j is a normalized eigenvector – i.e., an eigenvector with Euclidean norm equal to one – associated with the eigenvalue x_j, the corresponding weight can be computed from the first component of this eigenvector, namely:

$$w_j = \mu_0 \left[\phi_j(1)\right]^2$$

where μ_0 is the integral of the weight function

$$\mu_0 = \int_a^b \omega(x)dx$$

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 $2n$ continuous derivatives,

$$\int_a^b \omega(x)f(x)dx - \sum_{i=1}^{n} w_i f(x_i) = \frac{f^{(2n)}(\xi)}{(2n)!}[\![p_n, p_n]\!]$$

for some ξ in (a, b), where p_n is the monic orthogonal polynomial of degree n, and

$$[\![f(x), g(x)]\!] = \int_a^b \omega(x)f(x)g(x)dx$$

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

$$\omega(x) = 1$$

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

$$\int_a^b \omega(x)f(x)dx - \sum_{i=1}^{n} w_i f(x_i) = \frac{(b-a)^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} f^{(2n)}(\xi)$$

$$a < \xi < b$$

3. 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 $2n$ derivatives. Furthermore, the actual error may be much less that the bound established by the derivative.
4. 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 $[a, b]$ 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 $n + 1$ points to a n-point rule in such a way that the resulting rule is of the order $2n + 1$. 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:
    a. The integration points include the end-points of the integration interval.
    b. It is accurate for polynomials up to degree $2n - 3$, where n is the number of integration points (Quatteroni, Sacco, and Saleri (2000)).
2. Gauss-Lobatto Quadrature Function Expression: Lobatto quadrature of function $f(x)$ in an interval $[-1, +1]$ is

$$\int_{-1}^{+1} f(x)dx = \frac{2}{n(n-1)}[f(-1) + f(+1)] + \sum_{i=2}^{N} w_i f(x_i) + R_N$$

3. Gauss-Lobatto Quadrature Abscissa: The abscissaex_i is the $(i-1)^{st}$ zero of $P_{n-1}{}'(x)$.
4. Gauss-Lobatto Quadrature Weights:

$$w_i = \frac{2}{n(n-1)[P_{n-1}(x_i)]^2}$$

$$x_i \neq \pm 1$$

5. Gauss-Lobatto Quadrature Error Remainder:

$$R_N = \frac{-n(n-1)^3 2^{2n-1}[(n-2)!]^4}{(2n-1)[(2n-2)!]^3} f^{(2n-2)}(\xi)$$

$$-1 < \xi < +1$$

6. Low Order Gauss-Lobatto Quadrature Weight Sample:

Number of Points n	**Points x_i**	**Weights w_i**
3	0	$\frac{4}{3}$
	± 1	$\frac{1}{3}$
4	$\pm\sqrt{\frac{1}{5}}$	$\frac{5}{6}$
	± 1	$\frac{1}{6}$

5	0	$\frac{32}{45}$
	$\pm\sqrt{\frac{3}{7}}$	$\frac{49}{90}$
	± 1	$\frac{1}{10}$
6	$\pm\sqrt{\frac{1}{3}-\frac{2\sqrt{7}}{21}}$	$\frac{14+\sqrt{7}}{30}$
	$\pm\sqrt{\frac{1}{3}+\frac{2\sqrt{7}}{21}}$	$\frac{14-\sqrt{7}}{30}$
	± 1	$\frac{1}{15}$
7	0	$\frac{256}{525}$
	$\pm\sqrt{\frac{5}{11}-\frac{2}{11}\sqrt{\frac{5}{3}}}$	$\frac{124+7\sqrt{15}}{350}$
	$\pm\sqrt{\frac{5}{11}+\frac{2}{11}\sqrt{\frac{5}{3}}}$	$\frac{124-7\sqrt{15}}{350}$
	± 1	$\frac{1}{21}$

7. 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

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 $\int_a^b f(x)dx$
2. Use of n Point Gaussian Quadrature: Such integrals can be approximated, for example, by n-point Gaussian quadrature

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

where x_i and w_i are the points and the weights used to evaluate the function $f(x)$.

3. Consequences of Non-Matching Nodes: If the interval $[a, b]$ 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.
4. Node Extensions using Stieltjes Polynomials: Gauss-Kronrod formulas are extensions of the Gauss quadrature formulas generated by adding $n + 1$ points to a n point rule in such a way that the resulting rule is of order $2n + 1$ (Laurie (1997)); the corresponding Gauss rule is order $2n - 1$. The extra points are zeros of Stieltjes polynomials.
5. 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. $(G7, K15)$ on $[-1, +1]$:

Gauss Nodes	Weights
$\pm$0.94910 79123 42759	0.12948 49661 68870
$\pm$0.74153 11855 99394	0.27970 53914 89277
$\pm$0.40584 51513 77397	0.38183 00505 05119
$\pm$0.00000 00000 00000	0.41795 91836 73469

Kronrod Nodes	Weights
±0.99145 53711 20813	0.02293 53220 10529
±0.94910 79123 42759	0.06309 20926 29979
±0.86486 44233 59769	0.10479 00103 22250
±0.74153 11855 99394	0.14065 32597 15525
±0.58608 72354 67691	0.16900 47266 39267
±0.40584 51513 77397	0.19035 05780 64785
±0.20778 49550 07898	0.20443 29400 75298
±0.00000 00000 00000	0.20948 21410 84728

3. Use in Quadrature Error Estimate: The integral is then estimated by the Kronrod rule $K15$ and the error can be estimated as $|G7 - K15|$.
4. 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).
5. Tabulation of Quadruple Precision Coefficients: Quadruple Precision (34 decimal digits) coefficients for $(G7, K15)$, $(G10, K21)$, $(G15, K31)$, $(G20, K41)$, 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
- 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

Chi-Squared Distribution

Overview

1. Definition of the Chi-Squared Distribution: The chi-squared distribution – also called chi-squared or χ^2 distribution – with k degrees of freedom is the distribution of the sum of the squares of k 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 $Z_1, \cdots, Z_k$ are independent standard normal random variable, then the sum of their squares

$$Q = \sum_{i=1}^{k} {Z_i}^2$$

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

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

$$Q \sim \chi^2(k)$$

or

$$Q \sim {\chi_k}^2$$

The chi-squared distribution has one parameter k, 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:
    a. Chi-squared test of independence in contingency tables.
    b. Chi-squared test of goodness of fit of observed data to hypothetical distributions
    c. Likelihood-ratio test for nested models
    d. Log-rank test in survival analysis
    e. 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 n 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 t – 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 Z is a standard normal random variable, with mean 0 and variance 1,

$$Z \sim \mathcal{N}(0,1)$$

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

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

$$Q = Z^2$$

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

$$Q \sim {\chi_1}^2$$

11. 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.
12. 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.
13. Extreme Values under the Distribution: Just as the extreme values of the normal distribution have low probability – and give small p-values – extreme values of the chi-squared distribution have low probability.
14. 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)).
15. Neyman-Pearson Lemma: LRT's have several desirable properties; in particular, LRT's commonly provide highest power to reject NULL hypothesis.
16. 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.
17. 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.
18. 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

$$\chi = \frac{m - Np}{\sqrt{Npq}}$$

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

$$q = 1 - p$$

19. Chi-Squared Distribution Variable: Squaring both sides of the equation gives

$$\chi^2 = \frac{(m - Np)^2}{Npq}$$

20. Re-factoring the Chi-Squared Variable: Using

$$N = Np + N(1 - p)$$

$$N = m + (N - m)$$

and

$$q = 1 - p$$

this equation simplifies to

$$\chi^2 = \frac{(m - Np)^2}{Np} + \frac{(N - m - Np)^2}{Nq}$$

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

$$\chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i}$$

where χ^2 is the Pearson's cumulative test statistic, which asymptotically approaches a χ^2 distribution, O_i is the number of observations of type i,

$$E_i = Np_i$$

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

22. Reducing Binomial to Normal - χ^2: In the case of a binomial outcome – flipping a coin – the binomial distribution may be approximated by a normal distribution for sufficiently large n. 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.

23. 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.
24. 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

$$f(x;k) = \begin{cases} \dfrac{x^{\frac{k}{2}-1}e^{-\frac{x}{2}}}{2^{\frac{k}{2}}\Gamma\left(\frac{k}{2}\right)} & x > 0 \\ 0 & otherwise \end{cases}$$

where $\Gamma\left(\frac{k}{2}\right)$ denotes the gamma function, which has closed-form values for integer k.

Cumulative Distribution Function

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

$$F(x;k) = \frac{\gamma\left(\frac{k}{2},\frac{x}{2}\right)}{\Gamma\left(\frac{k}{2}\right)} = p\left(\frac{k}{2},\frac{x}{2}\right)$$

where $\gamma\left(\frac{k}{2},\frac{x}{2}\right)$ is the lower incomplete gamma function and $p\left(\frac{k}{2},\frac{x}{2}\right)$ is the regularized gamma function.

2. Expression for the Special Case $k = 2$: The special case of

$$k = 2$$

of this function has a simple form:

$$F(x;2) = 1 - e^{-\frac{x}{2}}$$

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

3. Tables for the χ^2 CDF: Tables for the chi-squared cumulative distribution function are widely available and the function is included in many spreadsheets and all statistical packages.

4. Chernoff Bounds on CDF Tails: Letting

$$z \equiv \frac{x}{k}$$

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

5. Chernoff Bounds for $0 < z < 1$: For the cases when

$$0 < z < 1$$

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

$$F(zk; k) \leq (ze^{1-z})^{\frac{k}{2}}$$

6. Chernoff Bounds for $z > 1$: The tail bound for cases when

$$z > 1$$

similarly is

$$1 - F(zk; k) \leq (ze^{1-z})^{\frac{k}{2}}$$

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 $\{X_i\}_{i=1}^{n}$ are independent chi-squared variables with $\{k_i\}_{i=1}^{n}$ degrees of freedom, respectively, then

$$Y = X_1 + \cdots + X_n$$

is chi-squared with $k_1 + \cdots + k_n$ degrees of freedom.

Sample Mean

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

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \sim Gamma\left(\alpha = \frac{nk}{2}, \theta = \frac{2}{n}\right)$$

where

$$X_i \sim \chi^2$$

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

$$\mu = \alpha \cdot \theta$$

and variance

$$\sigma^2 = \alpha \cdot \theta^2$$

the sample mean converges towards

$$\underset{n \to \infty}{Limit}\ \bar{X} \longrightarrow \mathcal{N}\left(\mu = k, \sigma^2 = 2\frac{k}{n}\right)$$

3. 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 k, the expectation is k and the variance is $2k$ – and hence the variance of the mean $\bar{X}$ is

$$\sigma^2 = 2\frac{k}{n}$$

Entropy

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

$$h = \int_0^\infty f(x;k)\log f(x;k)\,dx = \frac{k}{2} + \log\left[2\Gamma\left(\frac{k}{2}\right) + \left(1-\frac{k}{2}\right)\psi\left(\frac{k}{2}\right)\right]$$

where $\psi(x)$ is the digamma function.

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

$$\mathbb{E}[X] = k$$

and

$$\mathbb{E}[\log X] = \psi\left(\frac{k}{2}\right) + \log 2$$

are fixed.

3. 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 k degrees of freedom are given (Simon (2002))

$$\mathbb{E}[X^m] = k(k+2)(k+4)\cdots(k+2m-2) = 2^m \frac{\Gamma\left(m+\frac{k}{2}\right)}{\Gamma\left(\frac{k}{2}\right)}$$

Cumulants

The cumulants are readily obtained by a formal power series expansion of the logarithm of the characteristic function $\kappa_n = 2^{n-1}(n-1)!\,k$

Asymptotic Properties

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

$$k > 50$$

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

2. Speed of Convergence to Normality: Specifically, if

$$X \sim \chi^2(k)$$

then as k tends to infinity, the distribution of $\frac{X-k}{\sqrt{2k}}$ tends to a standard normal distribution. However, convergence is slow because the skewness is $\sqrt{\frac{8}{k}}$ and the excess kurtosis is $\frac{12}{k}$

3. Schemes for Faster Approach to Normality: The sampling distribution of $\log \chi^2$ converges to normality much faster than the sampling distribution of χ^2 (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.
4. Chi-squared Derivative Function #1: Some examples are: If

$$X \sim \chi^2(k)$$

then $\sqrt{2X}$ is approximately normally distributed with mean $\sqrt{2k-1}$ and unit variance (Johnson, Klotz, and Balakrishnan (1994)).

5. Chi-squared Derivative Function #2: If

$$X \sim \chi^2(k)$$

then $\left(\frac{X}{k}\right)^{\frac{1}{3}}$ is approximately normally distributed with mean $1 - \frac{2}{9k}$ and variance $\frac{2}{9k}$. 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

$$k \to \infty$$

$$\frac{{\chi_k}^2 - k}{\sqrt{2k}} \to \mathcal{N}(0,1)$$

2. Non-central Chi-squared Distribution:

$$\chi^2(k) \sim \chi'^2(k)$$

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

$$\lambda = 0$$

3. Double Chi-squared Distribution #1: If

$$Y \sim F(\nu_1, \nu_2)$$

then

$$X = \underset{\nu_2 \to \infty}{Limit} \nu_1 Y$$

has the double chi-squared distribution $\chi^2(\nu_1)$

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

$$Y \sim F(1, \nu_2)$$

then

$$X = \underset{\nu_2 \to \infty}{Limit} Y$$

has the double chi-squared distribution $\chi^2(1)$

5. Chi-squared Distribution as a Norm: The squared norm of k standard normally distributed variables is a chi-squared distribution with k degrees of freedom:

$$\left\| N_{i=1,\cdots,k}(0,1) \right\|^2 \sim \chi^2(k)$$

6. Gamma Distribution: If

$$X \sim \chi^2(\nu)$$

and

$$c > 0$$

then

$$cX \sim \Gamma\left(k = \frac{\nu}{2}, \theta = 2c\right)$$

which is a gamma distribution.

7. Chi Distribution: If

$$X \sim {\chi_k}^2$$

then

$$\sqrt{X} \sim \chi_k$$

the chi-distribution.

8. Exponential Distribution: If

$$X \sim \chi^2(2)$$

then

$$X \sim e^{\frac{1}{2}}$$

which is an exponential distribution.

9. Rayleigh Distribution: If

$$X \sim Rayleigh(1)$$

which is a Rayleigh distribution, then

$$X^2 \sim \chi^2(2)$$

10. Maxwell Distribution: If

$$X \sim Maxwell(1)$$

which is a Maxwell distribution, then

$$X^2 \sim \chi^2(3)$$

11. Inverse Chi-squared Distribution: If

$$X \sim \chi^2(\nu)$$

then

$$\frac{1}{X} \sim Inv - \chi^2(\nu)$$

which is the inverse chi-squared distribution.

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

$$X \sim \chi^2(\nu_1)$$

and

$$Y \sim \chi^2(\nu_2)$$

are independent, then

$$\frac{X}{X+Y} \sim Beta\left(\frac{\nu_1}{2}+\frac{\nu_2}{2}\right)$$

which is the beta distribution.

14. Chi-squared Distribution from Uniform: If

$$X \sim U(0,1)$$

which is a uniform distribution, then

$$-2\log X \sim \chi^2(2)$$

15. Laplace Distribution Transformation: $\chi^2(6)$ is a transformation of the Laplace distribution. If

$$X_i \sim Laplace\,(\mu,\beta)$$

then

$$\sum_{i=1}^{n} \frac{2|X_i - \mu|}{\beta} \sim \chi^2(2n)$$

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

$$\sum_{i=1}^{n} \frac{2|X_i - \mu|^{\beta}}{\alpha} \sim \chi^2\left(\frac{2n}{\beta}\right)$$

(Backstrom and Fischer (2018))

17. Transformed Pareto Distribution: Chi-squared distribution is a transformation of the Pareto distribution.
18. 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.
19. 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.
20. Non-central -t Distribution: Non-central t-distribution can be obtained from normal distribution and chi-squared distribution.
21. Multidimensional Gaussian Random Vectors: If Y is a k dimensional Gaussian random vector with mean μ and rank k covariance matrix C, then

$$X = [Y - \mu]^T C^{-1} [Y - \mu]$$

is chi-squared with k degrees of freedom.

22. 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.
23. Rank-reduced Chi-squared Distribution: If Y is a vector of k i.i.d. standard normal variables and A is a $k \times k$ symmetric idempotent matrix with rank $k - n$, then the quadratic form $Y^T AY$ is chi-squared with $k - n$ degrees of freedom.
24. Special Normal Chi-squared Distribution: If C is a $p \times p$ positive semi-definite covariance matrix with strictly positive diagonal covariance entries, then for

$$X \sim \mathcal{N}(0, C)$$

and w a random p-vector independent of X such that

$$w_1 + \cdots + w_p = 1$$

and

$$w_i \geq 0$$

$$i = 1, \cdots, p$$

it then holds that

$$\frac{1}{\left[\frac{w_i}{X_i}\right]^T C \left[\frac{w_i}{X_i}\right]} \sim {\chi_1}^2$$

(Pillai (2016)).

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

$$Y \sim F(k_1, k_2)$$

if

$$Y = \frac{{X_1}/{k_1}}{{X_2}/{k_2}}$$

where

$$X_1 \sim \chi^2(k_1)$$

and

$$X_2 \sim \chi^2(k_2)$$

are statistically independent.

27. Sum of Correlated χ^2 Distributions: If

$$X_1 \sim \chi^2(k_1)$$

and

$$X_2 \sim \chi^2(k_2)$$

are statistically independent, then

$$X_1 + X_2 \sim \chi^2(k_1 + k_2)$$

If X_1 and X_2 are not independent, then $X_1 + X_2$ is not chi-squared distributed.

Generalizations

1. Construction of the Chi-squared Distributions: The chi-squared distribution is obtained as the sum of k 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 $X_1, \cdots, X_n$ are chi-squared random variables and

$$a_1, \cdots, a_n \in \mathbb{R}^{>0}$$

then a closed expression for the distribution is

$$X = \sum_{i=1}^{n} a_i X_i$$

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 $z^T Az$ where z is a zero-means Gaussian vector having an arbitrary covariance matrix, and A is an arbitrary matrix.

Gamma, Exponential, and Related Distribution

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

$$X \sim {\chi_k}^2$$

is a special case of gamma distribution, in that

$$X \sim \Gamma\left(\frac{k}{2}, \frac{1}{2}\right)$$

using the rate parameterization of the gamma distribution – or

$$X \sim \Gamma\left(\frac{k}{2}, 2\right)$$

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

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

$$X \sim {\chi_2}^2$$

then

$$X \sim \sqrt{e}$$

is an exponential distribution.

3. 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

$$X \sim {\chi_k}^2$$

with even k, then X is Erlang distributed with shape parameter $\frac{k}{2}$ and scale parameter $\frac{1}{2}$.

Occurrence and Applications

1. χ^2 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 χ^2 from Gaussian Distribution: Following are some of the most common situations in which the chi-squared distribution arises from a Gaussian distributed sample. If $X_1, \cdots, X_n$ are i.i.d. $\mathcal{N}(\mu, \sigma^2)$ random variables, then

$$\sum_{i=1}^{n} (X_i - \bar{X})^2 \sim \sigma^2 {\chi_{n-1}}^2$$

where

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

5. χ^2 Variants in Normal Distribution: The box below shows some statistics based on

$$X_i \sim \mathcal{N}(\mu_i, {\sigma_i}^2)$$

$$i = 1, \cdots, k$$

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

Name	Distribution
Chi-squared Distribution	$\sum_{i=1}^{k} \left(\frac{X_i - \mu_i}{\sigma_i}\right)^2$
Non-central Chi-squared Distribution	$\sum_{i=1}^{k} \left(\frac{X_i}{\sigma_i}\right)^2$
Chi Distribution	$\sqrt{\sum_{i=1}^{k} \left(\frac{X_i - \mu_i}{\sigma_i}\right)^2}$

Non-central Chi Distribution	$\sqrt{\sum_{i=1}^{k}\left(\frac{X_i}{\sigma_i}\right)^2}$

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

Table of χ^2 Values vs. p-Values

1. Review of the p-Value Definition: The p-Value is the probability of observing a test statistic *at least* as extreme in the specified distribution (here chi-squared).
2. Computing the p-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 p-Value.
3. Establishing Statistical Significance using p-Values: A low p-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. p-Values for χ^2 with 10 Degrees of Freedom: The table below gives a number of p-Values matching to χ^2 for the first 10 degrees of freedom (Pennsylvania State University (2016)).

Degrees of	**χ^2 Value**

Freedom											
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**

6. 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	$\chi^2(k)$ or ${\chi_k}^2$
Parameters	$k \in \mathbb{N}^+$ (degrees of freedom)
Support	$x \in (0, +\infty)$ if $k = 1$ $x \in [0, +\infty)$ otherwise
Probability Density Function	$\frac{x^{\frac{k}{2}-1} e^{-\frac{x}{2}}}{2^{\frac{k}{2}} \Gamma\left(\frac{k}{2}\right)}$
Cumulative Density Function	$\frac{\gamma\left(\frac{k}{2}, \frac{x}{2}\right)}{\Gamma\left(\frac{k}{2}\right)}$
Mean	k
Median	$\approx k\left(1 - \frac{2}{9k}\right)^3$
Mode	$\max(k-2, 0)$

Variance	$2k$
Skewness	$\sqrt{\frac{8}{k}}$
Excess Kurtosis	$\frac{12}{k}$
Entropy	$\frac{k}{2} + \log\left[2\Gamma\left(\frac{k}{2}\right) + \left(1 - \frac{k}{2}\right)\psi\left(\frac{k}{2}\right)\right]$
Moment Generating Function	$(1 - 2t)^{-\frac{k}{2}}$ for $t < \frac{1}{2}$
Characteristic Function	$(1 - 2it)^{-\frac{k}{2}}$ Sanders (2011)
Probability Generating Function	$(1 - 2\log t)^{-\frac{k}{2}}$ for $0 < t < \sqrt{e}$

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 **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
- Pennsylvania State University (2016): 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
- 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
- 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 $(X_1, \cdots, X_k)$ be k independent, normally distributed random variables with means μ_i and unit variances. Then the random variable $\sum_{i=1}^{k} {X_i}^2$ is distributed according to the non-central chi-square distribution.
2. Non-central Chi-square Distribution Parameters: It has 2 parameters - k which specifies the number of degrees of freedom, i.e., the number of X_i, and λ which is related to the mean of the random variables X_i by

$$\lambda = \sum_{i=1}^{k} {\mu_i}^2$$

   λ is sometimes called the non-centrality parameter; some references define λ in other ways, such as half of the above sum, or its square root.
3. ${\chi'}^2$ as $\mathcal{N}(\mu, I_k)$ 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 $\mathcal{N}(0_k, I_k)$ 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 $\mathcal{N}(\mu, I_k)$ distribution, Here 0_k is a vector of length k,

$$\mu = (\mu_1, \cdots, \mu_k)$$

and I_k is the identity matrix of size k.

Non-central Chi-Square Distribution Table

Parameters	$k > 0$ Degrees of Freedom $\lambda > 0$ Non-centrality Parameter
Support	$x \in [0, +\infty)$
CDF	$\frac{1}{2} e^{-\frac{x+\lambda}{2}} \left(\frac{x}{\lambda}\right)^{\frac{k}{4}-\frac{1}{2}} I_{\frac{k}{2}-1}\left(\sqrt{\lambda x}\right)$
PDF	$1 - Q_{\frac{k}{2}}\left(\sqrt{\lambda}, \sqrt{x}\right)$ with Marcum Q-function $Q_M(a, b)$
Mean	$k + \lambda$
Variance	$2(k + 2\lambda)$
Skewness	$\frac{2^{\frac{3}{2}}(k + 2\lambda)}{(k + 2\lambda)^{\frac{3}{2}}}$
Excess Kurtosis	$12 \frac{(k + 4\lambda)}{(k + 2\lambda)^2}$
MGF	$\frac{e^{\frac{\lambda t}{1-2t}}}{(1-2t)^{\frac{k}{2}}}$ for $2t < 1$
CF	$\frac{e^{\frac{i\lambda t}{1-2it}}}{(1 - 2it)^{\frac{k}{2}}}$

Definition

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

$$f_X(x;k,\lambda) = \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}}}{i!}\left(\frac{\lambda}{2}\right)^i f_{Y_{k+2i}}(x)$$

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

2. 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 J has a Poisson distribution with mean $\frac{\lambda}{2}$, and the conditional distribution of Z given

$$J = i$$

   is chi-square with $k + 2i$ degrees of freedom. Then the unconditional distribution of Z is non-central chi-square with k degrees of freedom, and non-centrality parameter λ.

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

$$f_X(x;k,\lambda) = \frac{1}{2}e^{-\frac{x+\lambda}{2}}\left(\frac{x}{\lambda}\right)^{\frac{k}{4}-\frac{1}{2}} I_{\frac{k}{2}-1}\left(\sqrt{\lambda x}\right)$$

   where $I_v(y)$ is a modified Bessel function of the first kind given by

$$I_v(y) = \sum_{m=0}^{\infty} \frac{1}{m!\,\Gamma(m+v+1)}\left(\frac{y}{2}\right)^{2m+v}$$

4. 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))

$$f_X(x;k,\lambda) = \frac{1}{2}e^{-\frac{\lambda}{2}}{}_0F\left(;\frac{k}{2};\frac{\lambda x}{4}\right)\frac{1}{2^{\frac{k}{2}}\Gamma\left(\frac{k}{2}\right)}e^{-\frac{x}{2}}x^{\frac{k}{2}-1}$$

Siegel (1979) discusses the case

$$k = 0$$

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

$$M(t;k,\lambda) = \frac{e^{\frac{\lambda t}{1-2t}}}{(1-2t)^{\frac{k}{2}}}$$

for

$$2t < 1$$

Properties – Moments

1. First Four Non-central Moments: The first few raw moments are

$$\mu'_1 = k + \lambda$$

$$\mu'_2 = (k + \lambda)^2 + 2(k + 2\lambda)$$

$$\mu'_3 = (k + \lambda)^3 + 6(k + \lambda)(k + 2\lambda) + 8(k + 3\lambda)$$

$$\mu'_4 = (k + \lambda)^4 + 12(k + \lambda)^2(k + 2\lambda) + 4(11k^2 + 44k\lambda + 36\lambda^2) + 48(k + 4\lambda)$$

2. Second, Third, and Fourth Central Moments: The first few central moments are

$$\mu_2 = 2(k + 2\lambda)$$

$$\mu_3 = 8(k + 3\lambda)$$

$$\mu_4 = 12(k + \lambda)^2 + 48(k + 4\lambda)$$

3. Cumulant/Arbitrary Non-central Moment: The n^{th} cumulant is

$$K_n = 2^{n-1}(n-1)!\,(k + n\lambda)$$

Hence

$$\mu'_n = 2^{n-1}(n-1)!\,(k + n\lambda) + \sum_{j=1}^{n-1} 2^{j-1}\frac{(n-1)!}{(n-j)!}(k + j\lambda)\mu'_{n-j}$$

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

$$P(x;k,\lambda) = e^{-\frac{\lambda}{2}} \sum_{i=0}^{\infty} \frac{1}{i!} \left(\frac{\lambda}{2}\right)^{i} Q(x;k+2i)$$

   where $Q(x;k)$ is the cumulative distribution function of the central chi-square distribution with k degrees of freedom, which is given by

$$Q(x;k) = \frac{\gamma\left(\frac{k}{2},\frac{x}{2}\right)}{\Gamma\left(\frac{k}{2}\right)}$$

   and $\gamma(k,z)$ is the lower incomplete gamma function.

2. CDF Based Marcum Q Function: The Marcum Q-function $Q_M(a,b)$ can also be used to represent the CDF (Nuttall (1975)) as

$$P(x;k,\lambda) = 1 - Q_{\frac{k}{2}}\left(\sqrt{\lambda},\sqrt{x}\right)$$

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. $\left(\frac{\chi'^2}{k+\lambda}\right)^{\frac{1}{3}}$ is approximately normally distributed as $\mathcal{N}\left(1-\frac{2}{9f},\frac{2}{9f}\right)$, i.e.,

$$P(x;k,\lambda)\approx\Phi\left(\frac{\left(\frac{\chi'^2}{k+\lambda}\right)^{\frac{1}{3}}-\left(1-\frac{2}{9f}\right)}{\sqrt{\frac{2}{9f}}}\right)$$

where

$$f=\frac{(k+\lambda)^2}{k+2\lambda}=k+\frac{\lambda^2}{k+2\lambda}$$

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

$$f=f(\lambda,k)$$

becomes

$$f=k$$

for

$$\lambda = 0$$

which is the central chi-square case.

2. 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:

$$P(x;k,\lambda) \approx \Phi\left(\frac{\left(\frac{x}{k+\lambda}\right)^h - \left(1 + hp(h-1-0.5(2-h)mp)\right)}{h\sqrt{2p}(1+0.5mp)}\right)$$

where

$$h = 1 - \frac{2}{3}\frac{(k+\lambda)(k+3\lambda)}{(k+2\lambda)^2}$$

$$f = \frac{k+2\lambda}{(k+\lambda)^2}$$

$$m = (h-1)(1-3h)$$

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 x, 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 $X_1, \cdots, X_n$ 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

$$X = {X_1}^2 + \cdots + {X_n}^2$$

depends only on the means through the squared length

$$\lambda = {\mu_1}^2 + \cdots + {\mu_n}^2$$

Without loss of generality, one can therefore take

$$\mu_1 = \sqrt{\lambda}$$

and

$$\mu_2 = \mu_k = 0$$

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

$$X = X_1^{\,2}$$

for the

$$k = 1$$

case. Simple transformations of the random variables show that

$$f_X(x; 1, \lambda) = \frac{1}{2\sqrt{x}}\left[\phi\left(\sqrt{x} - \sqrt{\lambda}\right) + \phi\left(\sqrt{x} + \sqrt{\lambda}\right)\right] = \frac{1}{\sqrt{2\pi x}} e^{-\frac{x+\lambda}{2}} \cosh\sqrt{\lambda x}$$

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

$$k = 1$$

The indices on the chi-square random variables in the series above are $1 + 2i$ in this case.

5. Connection to the Central Chi Square: Finally, for the general case. It has been assumed, without loss of generality, that $X_2, \cdots, X_k$ are standard normal, and so ${X_2}^2 + \cdots + {X_k}^2$ has a *central* chi-square distribution with $k - 1$ degrees of freedom, independent of ${X_1}^2$. Using the Poisson-weighted mixture representation for ${X_1}^2$, 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

$$1 + 2i + k - 1 = k + 2i$$

as required.

Related Distributions

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

$$V \sim {\chi_k}^2$$

then V is also non-central chi-square distributed

$$V \sim \chi'_k{}^2(0)$$

2. Non central F Distribution: If

$$V_1 \sim \chi'_{k_1}{}^2(\lambda)$$

and

$$V_2 \sim \chi'_{k_2}{}^2(\lambda)$$

and V_1 is independent of V_2, then a non-central F distributed variables is developed as

$$\frac{V_1/k_1}{V_2/k_2} = F'_{k_1,k_2}(\lambda)$$

3. Poisson Conditioned Chi Square: If

$$J \sim Poisson\ (\lambda)$$

then

$$\chi_{k+2J}{}^2 \sim \chi'_k{}^2(\lambda)$$

4. Transformation to the Rice Distribution: If

$$V \sim \chi'_k{}^2(\lambda)$$

then $\sqrt{V}$ takes the Rice distribution with the parameter $\sqrt{\lambda}$.

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

$$V \sim \chi'_k{}^2(\lambda)$$

then

$$\frac{V-(k+\lambda)}{\sqrt{2(k+2\lambda)}} \to \mathcal{N}(0,1)$$

in distribution as either

$$k \to \infty$$

or

$$\lambda \to \infty$$

6. Independent Non central Chi Square Sum:

$$V_1 \sim {\chi'}_{k_1}^{\,2}(\lambda_1)$$

and

$$V_2 \sim {\chi'}_{k_2}^{\,2}(\lambda_2)$$

and V_1 is independent of V_2, then

$$W = V_1 + V_2 \sim {\chi'}_{k}^{\,2}(\lambda_1 + \lambda_2)$$

where

$$k = k_1 + k_2$$

In general, for a finite set of

$$V_i \sim {\chi'}_{k_i}^{\,2}(\lambda_i)\; i \in \{1, \cdots, N\}$$

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

$$Y = \sum_{i=1}^{N} V_i$$

has the distribution

$$Y \sim {\chi'}_{k_y}^{\,2}(\lambda_y)$$

where

$$k_y = \sum_{i=1}^{N} k_i$$

$$\lambda_y = \sum_{i=1}^{N} \lambda_i$$

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

$$M_Y(t) = M_{Y\sum_{i=1}^{N} V_i}(t) = \prod_{i=1}^{N} M_{Y_i}(t)$$

by the independence of the V_i 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 1 and the specified means.

7. Complex Non-central Chi-square: The complex non-central chi-square has applications in radio communications and radar systems. Let $(z_1, \cdots, z_k)$ be independent scalar random variables of circular symmetry, means of μ_i and unit variances.

$$\mathbb{E}[|z_i - \mu_i|^2] = 1$$

Then the real random variable

$$S = \sum_{i=1}^{k} |z_i|^2$$

is distributed according to the complex non-central

$$f_S(S) = \frac{1}{2} e^{-(S+\lambda)} \left(\frac{S}{\lambda}\right)^{\frac{k-1}{2}} I_{k-1}\left(2\sqrt{Sx}\right)$$

where

$$\lambda = \sum_{i=1}^{k} |\mu_i|^2$$

Transformations

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

$$z = \sqrt{\frac{X - b}{k + \lambda}}$$

He analyzes the expansions of the cumulants of z up to the term $\mathcal{O}((k+\lambda)^{-4})$ and shows that the following choices of b produce reasonable results.

a.

$$b = \frac{k-1}{2}$$

makes the second cumulant of z approximately independent of λ

b.

$$b = \frac{k-1}{3}$$

makes the third cumulant of z approximately independent of λ

c.

$$b = \frac{k-1}{4}$$

makes the fourth cumulant of z approximately independent of λ

2. Variance Stabilizing Transformation of ${\chi'_k}^2(\lambda)$: Also, a simpler transformation

$$z_1 = \sqrt{X - \frac{k-1}{2}}$$

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

$$z_2 = \sqrt{X + \frac{k-1}{2}}$$

and variance $\mathcal{O}((k+\lambda)^{-2})$. 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 χ^2 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 Q_M Function *IEEE Transactions on Information Theory* **21 (1)** 95-96
- Sankaran, M. (1959): On the Non-central χ^2 Distribution *Biometrika* **46 (1-2)** 235-237
- Sankaran, M. (1963): Approximations to the Non-central χ^2 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
- Young, D. S. (2010): tolerance: An R Package for estimating Tolerance Intervals *Journal of Statistical Software* **36 (5)** 1-39

Figure #1
Fixed Point Search SKU Flow

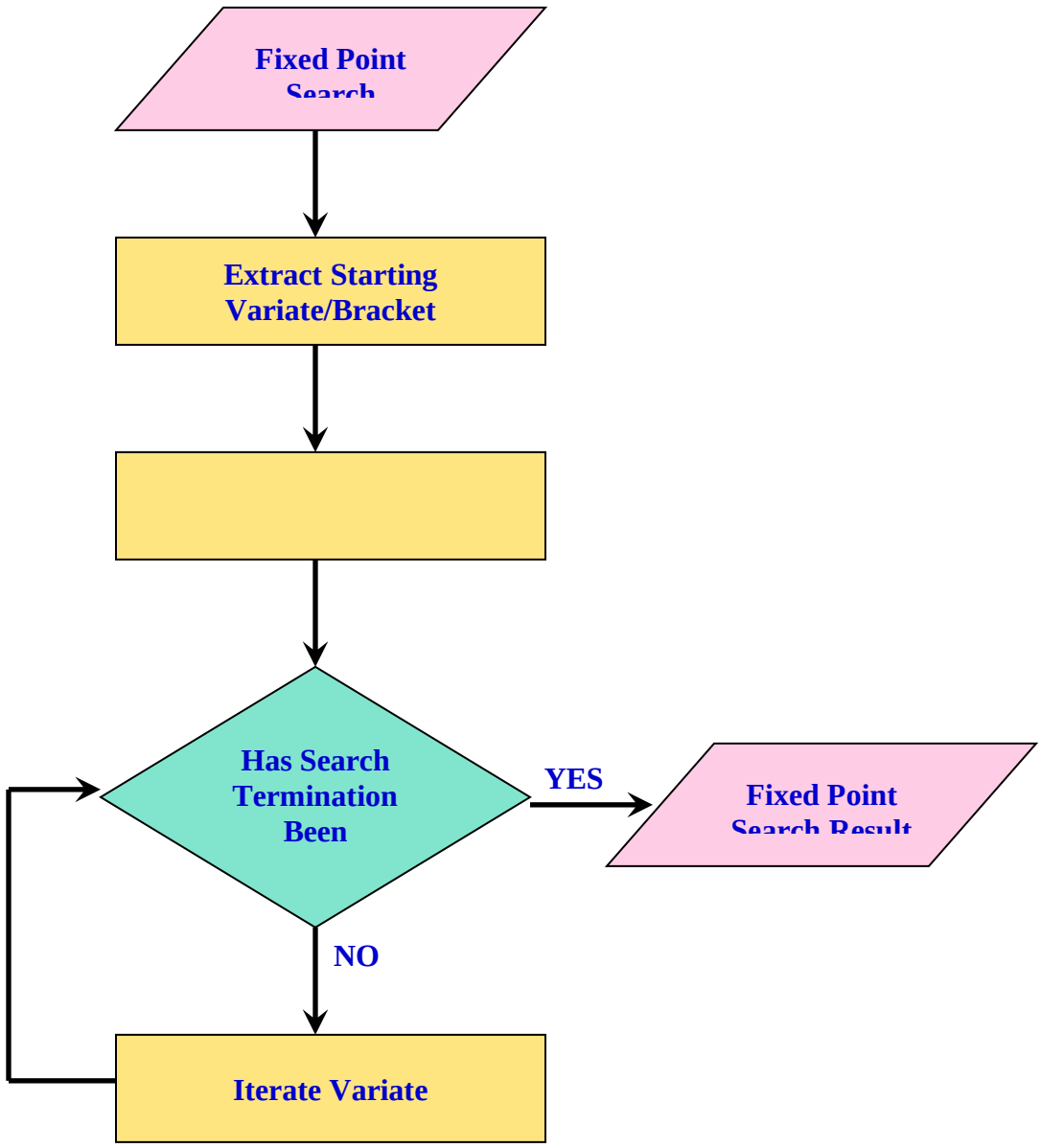

Figure #2
Bracketing SKU Flow

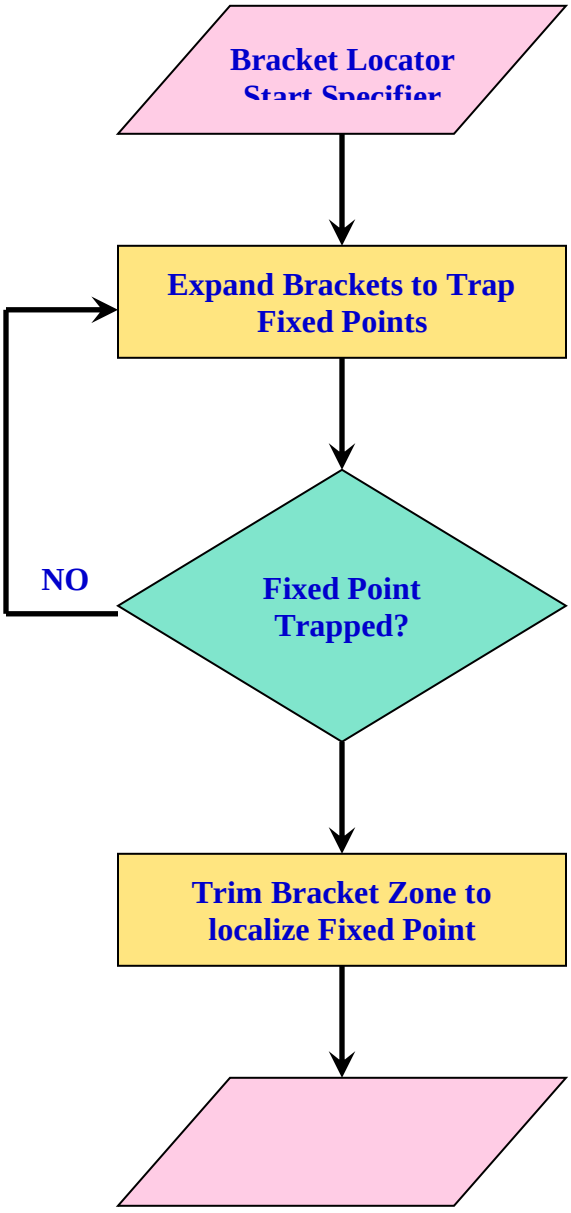

Figure #3
Objective Function Undefined at the Starting Variate

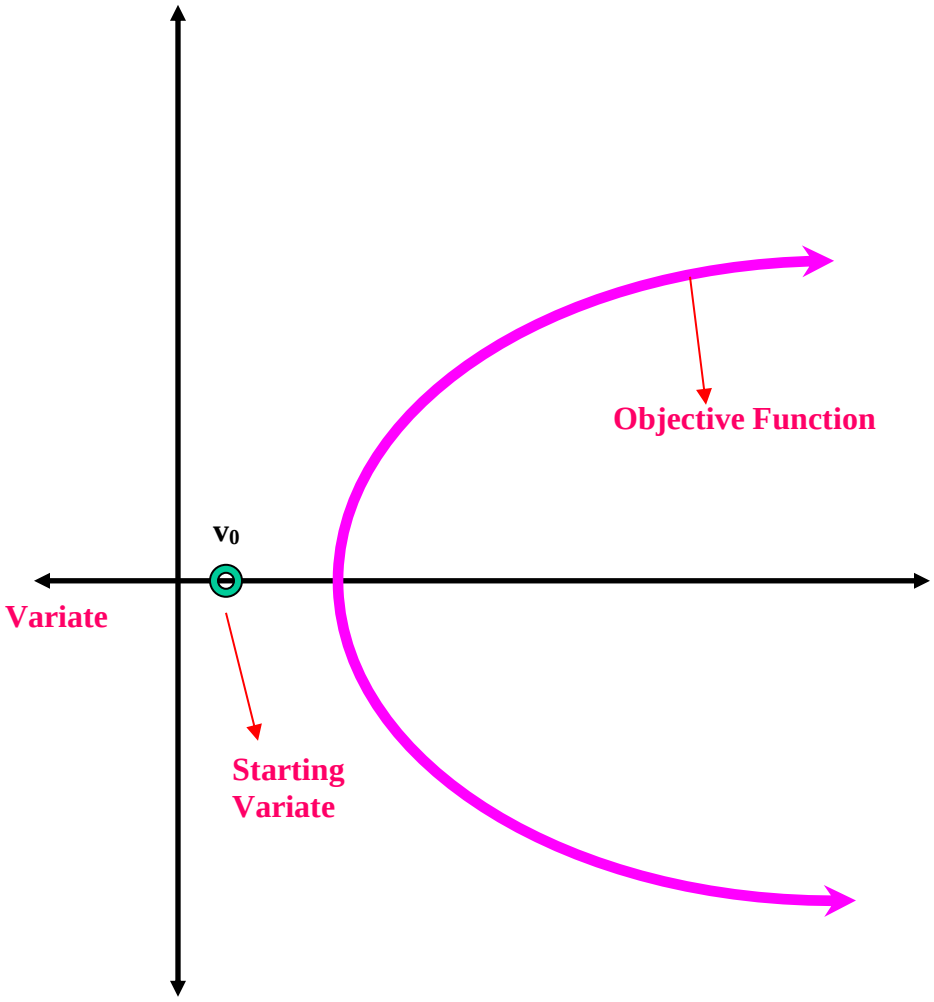

Figure #4
Objective Function Undefined at any of the Candidate Variates

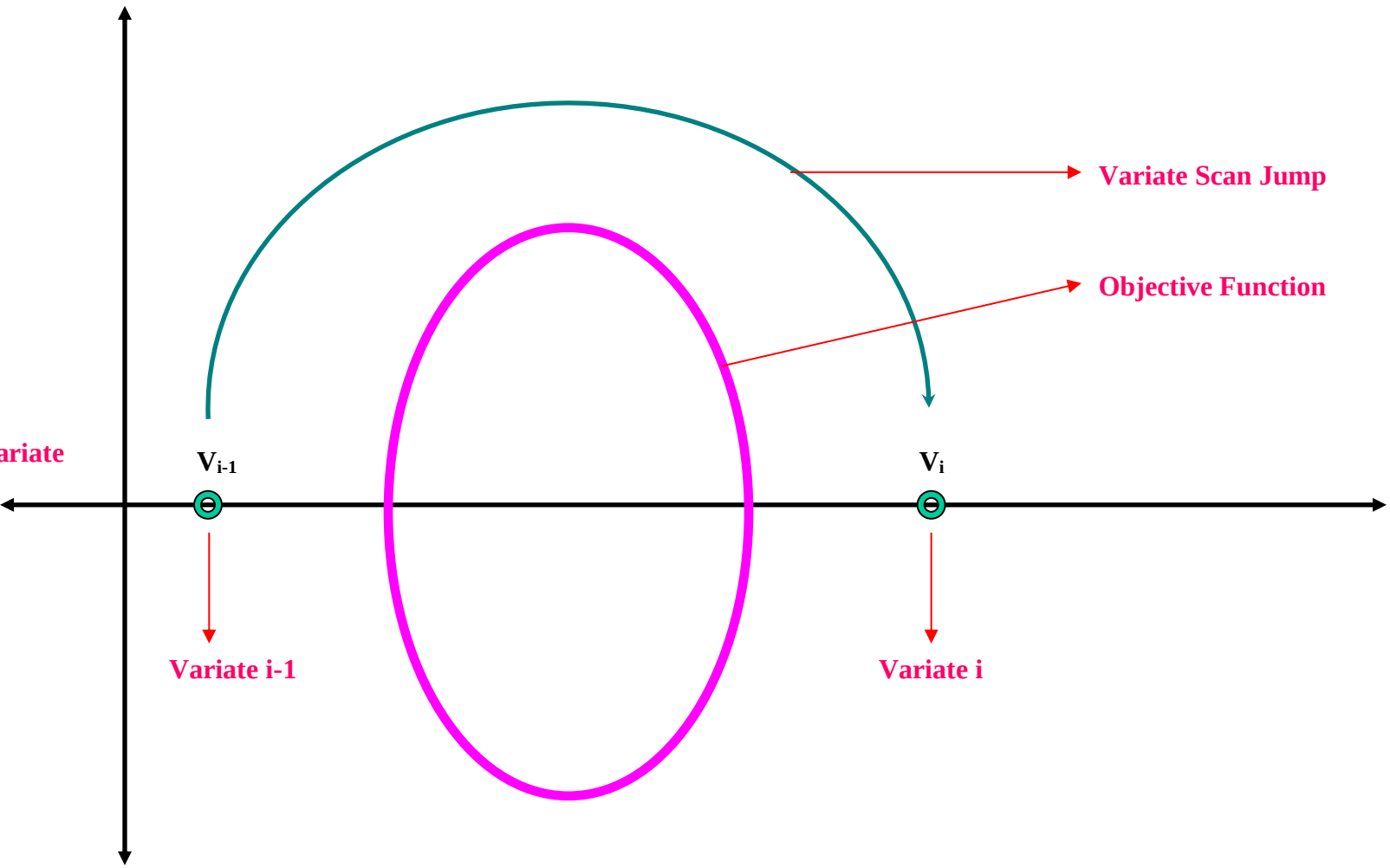

Figure #5
General Purpose Bracket Start Locator

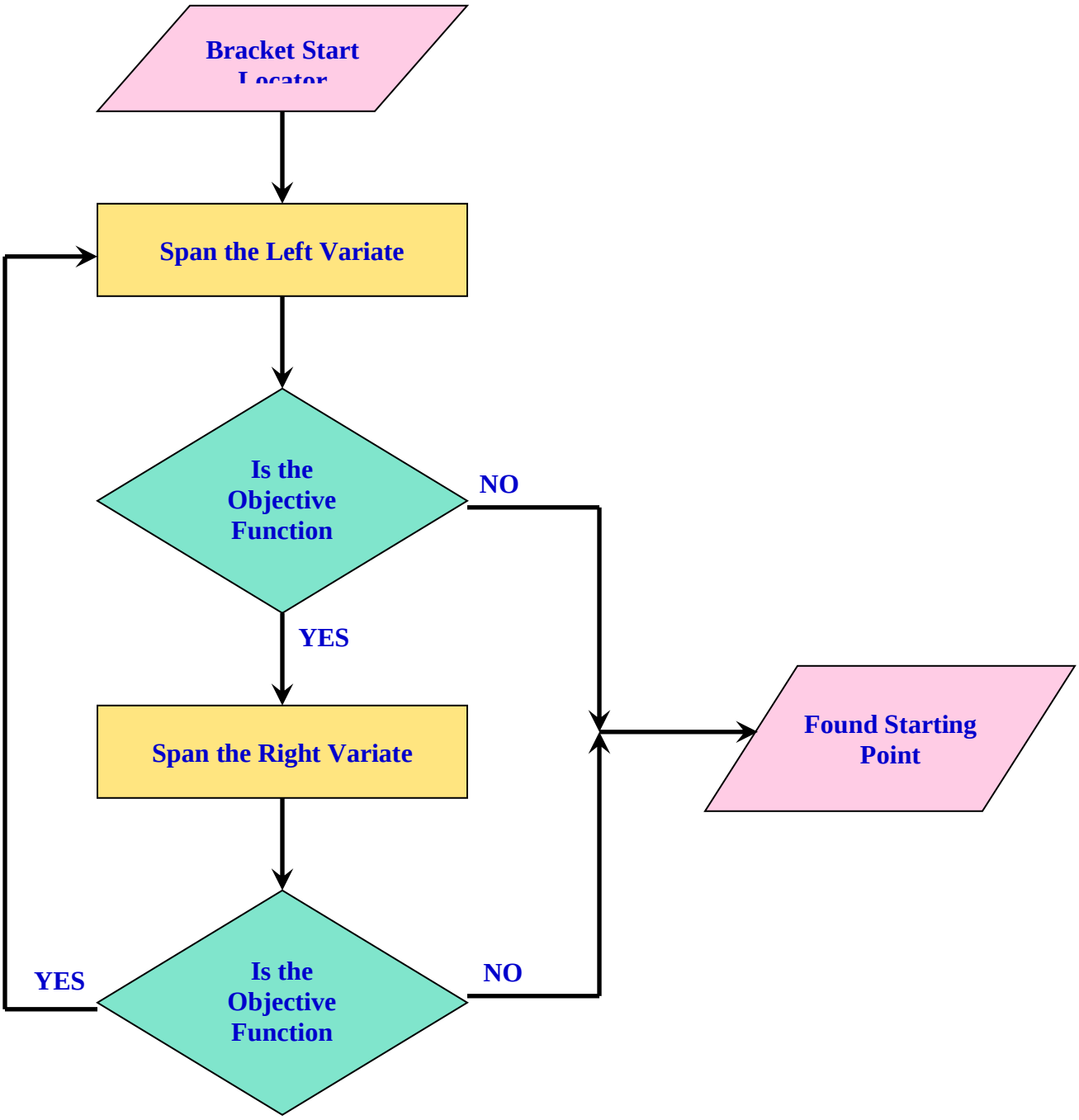

Figure #6
Bracketing when Objective Function Validity is Range-bound

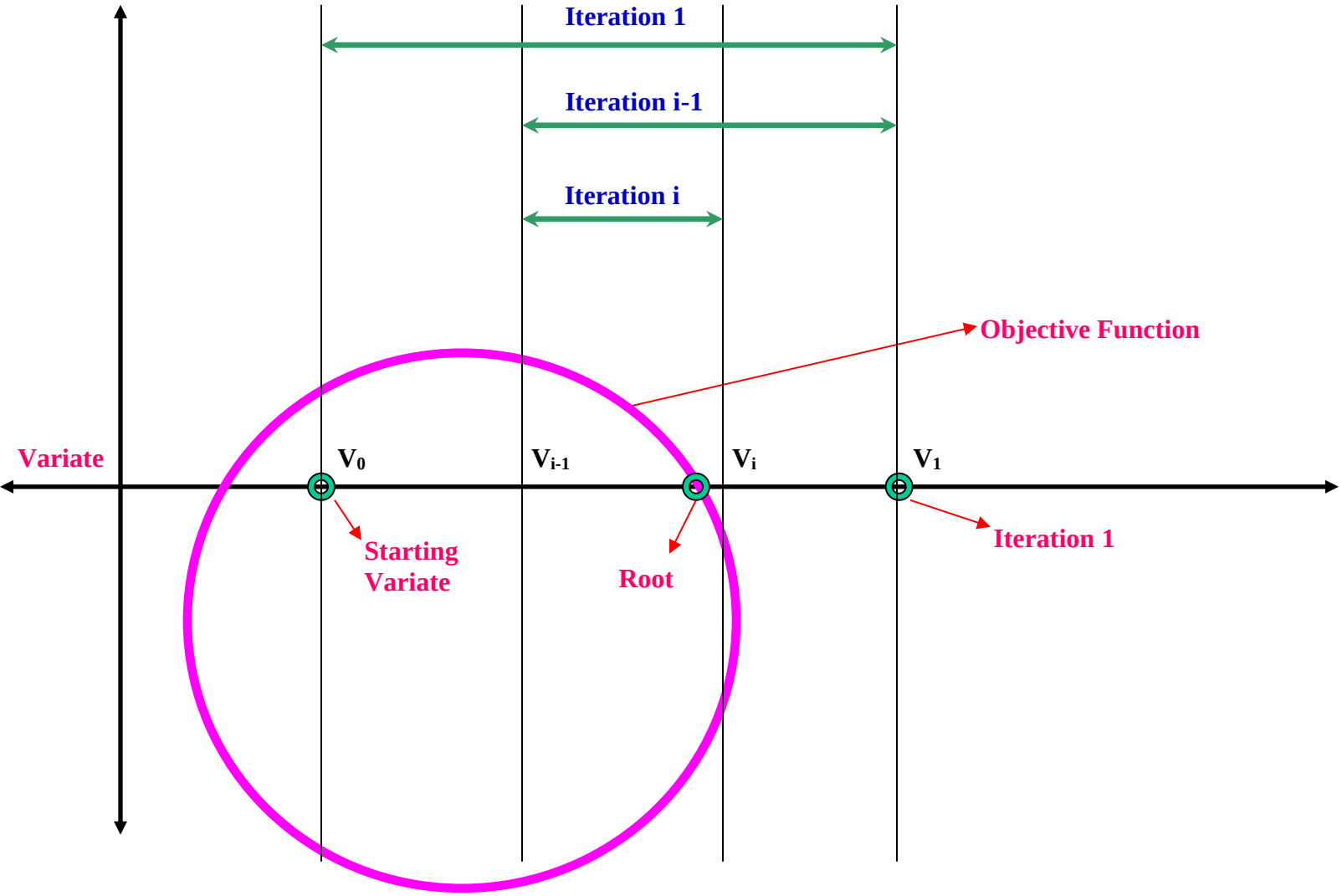

Figure #7
Objective Function Fixed Point Bracketing