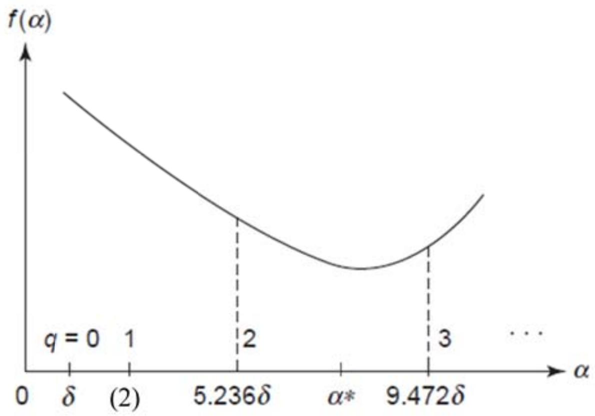
1. Fill in the blanks. (2 pts each)

* The golden section search can be initiated once the initial interval of uncertainty is known. If the initial bracketing is done using the variable step increment (with a factor of (1)), then the function value at one of the points α_{q-1} is already known. It turns out that α_{q-1} is automatically the point α_a.



* Initial bracketing of the minimum point in the golden section method.

* The sequential unconstrained minimization techniques have certain weaknesses that are most serious when (3) is large. The penalty and barrier functions tend to be ill-behaved near the boundary of the feasible set where the optimum points usually lie. There is also a problem of selecting the sequence of (3).

* There are two basic classes of methods for MV-OPT: enumerative and (4). In the enumerative category full enumeration is a possibility; however partial enumeration is most common based on branch and bound methods. In the (4) category, the most common ones are (5) and genetic algorithms.

* (6) is an operator where an old design (D-string) is copied into the new population according to the design's fitness. There are many different strategies to implement this (6) operator. This is also called the *selection process*.

* Let us select a design as "10 1110 1001" and the location #7 from the right end on its D-string. The (7) operation involves replacing the current value of 1 at the seventh location with 0 as "10 1010 1001".

TABLE 12-1 Optimization Toolbox Functions

Type of problem	*Formulation*	*Function*
One-variable unconstrained minimization	Find $x \in [x_l\ x_u]$ to minimize $f(x)$	*fminbnd*
Unconstrained minimization	Find $\mathbf{x}$ to minimize $f(\mathbf{x})$	*fminunc* *fminsearch*
Constrained minimization	Find $\mathbf{x}$ to minimize $f(\mathbf{x})$ subject to $\mathbf{Nx} = \mathbf{e}$, $\mathbf{Ax} \le \mathbf{b}$ $h_j = 0, j = 1$ to p $g_i(\mathbf{x}) \le 0, i = 1$ to m $x_{il} \le x_i \le x_{iu}$	*fmincon*
Linear programming	Find $\mathbf{x}$ to minimize $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$ subject to $\mathbf{Nx} = \mathbf{e}$, $\mathbf{Ax} \le \mathbf{b}$	(8-1)
Quadratic programming	Find $\mathbf{x}$ to minimize $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{Hx}$ subject to $\mathbf{Nx} = \mathbf{e}$, $\mathbf{Ax} \le \mathbf{b}$	(8-2)

TABLE 12-2 Explanation of Output from Optimization Function

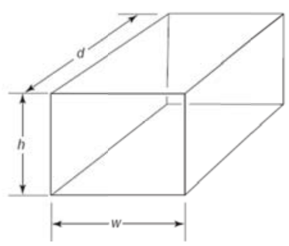
Argument	*Description*
`x`	The solution vector or matrix found by the optimization function. If `ExitFlag` > 0 then `x` is a solution, otherwise `x` is the latest value from the optimization routine.
`FunValue`	Value of the objective function, `ObjFun`, at the solution `x`.
(9)	The exit condition for the optimization function. If (9) is positive, then the optimization routine converged to a solution `x`. If (9) is zero, then the maximum number of function evaluations was reached. If (9) is negative, then the optimization routine did not converge to a solution.
`Output`	The `output` vector contains several pieces of information about the optimization process. It provides the number of function evaluations (Output. (10)) and the name of the algorithm used to solve the problem (Output.algorithm), etc.

2. Design the steel framework shown in the figure at a minimum cost. The cost of all horizontal members in one direction is \$20*w* and in the other direction it is \$30*d*. The cost of a vertical column is \$50*h*. The frame must enclose a total volume of at least 600m^3.

(1) Formulate the design optimization problem in the standard normalized form. (5 pts)

(2) Define the LP subproblem at the point *w*=10, *d*=10, *h*=4 using 50% move limits. (5 pts)

(3) Define the QP subproblem. (5 pts)



3. Given the function $f(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ and $\boldsymbol{x}^{(0)} = (0,\ 1)^T$, $\boldsymbol{x}^{(1)} = (1,\ 0)^T$ the first two points in a search for $\boldsymbol{x}^*$, the minimum of f. Calculate the search direction at $\boldsymbol{x}^{(1)}$ using the following gradient-based methods: (5+8+5 pts)

(1) steepest descent (Cauchy) method (2) conjugate gradient method (3) Newton's method

4. Consider the optimization problem: Minimize $f(\boldsymbol{x}) = x_1^2 + x_2^2$ subject to $h(\boldsymbol{x}) = x_1 + x_2 - 1 = 0$

(1) Write the expression for the augmented Lagrangian with penalty parameter r = 1 and solve for x_1 and x_2. (10 pts)

(2) Beginning with Lagrange multiplier λ = 0, perform two iterations of the ALM method and compare with the graphical solution. (5 pts)

5. Fill in the blanks. (Yes or No or Yes/No) (1 pt each)

Method	MV-OPT problem type solved	Can find feasible discrete solution?	Can find global minimum for convex problem?	Need gradients?
BBM	1–5	Yes	Yes	(1)
SA	1–5	Yes	Yes	(2)
Genetic algorithm	1–5	Yes	Yes	(3)
Sequential linearization	1	Yes	Yes	(4)
Dynamic round-off	1	Yes	No guarantee	(5)
Neighborhood search	1	Yes	Yes	(6)

6. Fill in the blanks. (L(ocal) or G(lobal) or L+G) (1 pt each)

Method	Can solve discrete problems?	General constraints?	Tries to find all x*?	Phases	Needs gradients?
Covering (D)	No	No	Yes	(1)	1
Zooming (D)	Yes[1]	Yes	No	(2)	1
Generalized descent (D)	No	No	No	(3)	Yes
Tunneling (D)	No	Yes	No	(4)	1
Multistart (S)	Yes[1]	Yes	Yes	(5)	1
Clustering (S)	Yes[1]	Yes	Yes	(6)	1
Controlled random search (S)	Yes	No	No	(7)	No
Acceptance–rejection (S)	Yes[1]	Yes	No	(8)	No
Stochastic integration (S)	No	No	No	(9)	No
Genetic (S)	Yes	No	No	(10)	No
Stochastic zooming (S)	Yes[1]	Yes	No	(11)	1
Domain elimination (S)	Yes[1]	Yes	Yes	(12)	1

7. Fill in the blanks. (Yes or No) (1 pt each)

Method	Always yields Pareto optimal point?	Can yield all Pareto optimal points?	Involves weights?	Depends on function continuity?	Uses utopia point?
Genetic	Yes	(1-1)	(1-2)	No	No
Weighted sum	Yes	(2-1)	(2-2)	Problem type and optimization engine determines this	Utopia point or its approximation is needed for function normalization or in the formulation of the method
Weighted min–max	Yes[a]	(3-1)	(3-2)	Same as above	Same as above
Weighted global criterion	Yes	(4-1)	(4-2)	Same as above	Same as above
Lexicographic	Yes[b]	(5-1)	(5-2)	Same as above	No
Bounded objective function	Yes[c]	(6-1)	(6-2)	Same as above	No
Goal programming	No	(7-1)	(7-2)	Same as above	No