Matlab includes an optimization toolbox that implements various numerical optimization routines, including sequential quadratic programming algorithm to solve for constrained optima. The two Matlab functions \texttt{fminunc} and \texttt{fmincon} solve the unconstrained and constrained problems, respectively. These are some brief notes and examples on using the \texttt{fmincon} function. For complete information, type “help fmincon” at the Matlab prompt.

\texttt{fmincon} solves problems of the form (using notation consistent with Matlab)

\[
\begin{align*}
\text{min} & \quad \text{FUN(X)} \\
\text{subject to:} & \quad A*X \leq B \\
& \quad Aeq*X = Beq \quad \text{(linear constraints)} \\
& \quad C(X) \leq 0 \\
& \quad Ceq(X) = 0 \quad \text{(nonlinear constraints)} \\
& \quad LB \leq X \leq UB
\end{align*}
\]

A typical use of the \texttt{fmincon} function is shown below.

\[
[X, FVAL, EXITFLAG, OUTPUT, LAMBDA, GRAD, HESSIAN] = \text{FMINCON} (FUN, X0, A, B, Aeq, Beq, LB, UB, NONLCON)
\]

where

\begin{itemize}
  \item \texttt{X} = solution vector
  \item \texttt{FVAL} = optimal value of the objective function at the optimal solution point
  \item \texttt{EXITFLAG}:
    \begin{itemize}
      \item > 0 then \texttt{FMINCON} converged to a solution \texttt{X}
      \item = 0 then the maximum number of function evaluations was reached
      \item < 0 then \texttt{FMINCON} did not converge to a solution
    \end{itemize}
  \item \texttt{OUTPUT} is a structure containing some information about the algorithm performance
    \begin{itemize}
      \item \texttt{OUTPUT.iterations} = with the number of iterations taken
      \item \texttt{OUTPUT.funcCount} = the number of function evaluations
      \item \texttt{OUTPUT.algorithm} = the algorithm used
      \item \texttt{OUTPUT.cgiterations} = number of CG iterations (if used)
      \item \texttt{OUTPUT.firstorderopt} = the first-order optimality (if used)
    \end{itemize}
  \item \texttt{LAMBDA} returns the Lagrange multipliers at the solution \texttt{X}
    \begin{itemize}
      \item \texttt{LAMBDA.lower} for \texttt{LB}
      \item \texttt{LAMBDA.upper} for \texttt{UB}
      \item \texttt{LAMBDA.ineqlin} is for the linear inequalities
      \item \texttt{LAMBDA.eqlin} is for the linear equalities
      \item \texttt{LAMBDA.ineqnonlin} is for the nonlinear inequalities
      \item \texttt{LAMBDA.eqnonlin} is for the nonlinear equalities
    \end{itemize}
\end{itemize}
GRAD returns the value of the gradient of FUN at the solution X. Note, this is NOT the
gradient of the Lagrangian.
HESSIAN returns the value of the HESSIAN of FUN at the solution X.

FUN is the objective function to optimize. 
FUN accepts input X and returns a scalar function value F evaluated at X.FUN
should either be a function specified using @, e.g. [X,...] = fmincon(@myfun,...) where
F = myfun(X) returns the scalar function value F of the MYFUN function evaluated at X
and there exists a .m file myfun.m. FUN can also be an inline object [X,...] =
fmincon(inline('3*sin(x(1))+exp(x(2))',...).

X0 is the initial starting point. X0 may be a scalar, vector, or matrix.
A, B are the matrices defining the linear inequalities A*X <= B.
Aeq, Beq are the matrices defining the linear equalities Aeq*X = Beq. (Set Aeq=[] and
Beq=[] if no equalities exist.)
UB, LB define a set of lower and upper bounds on the design variables, X, so that the
solution is in the range LB <= X <= UB. Use empty matrices for LB and UB if no
bounds exist. Set LB(i) = -Inf if X(i) is unbounded below; set UB(i) = Inf if X(i) is
unbounded above.
NLCON is a function that accepts X and returns the vectors C and Ceq, representing the
nonlinear inequalities and equalities, respectively. FMINCON minimizes FUN such that
C<=0 and Ceq=0.

To demonstrate how to use fmincon, here are some sample formulations.

Example 1: The shipping box optimization problem

\[
\text{min } \quad \text{FUN}(X) = 2c(xy + xz + yz)
\]
\[
\text{such that } \quad xyz = V
\]

In this problem, let’s assume that c = 10, V = 6 and an initial point of X0 = [1;1;1]. We
will let x \equiv x(1), y \equiv x(2) and z \equiv x(3).

>>> 
[X, FVAL, EXITFLAG, OUTPUT, LAMBDA, GRAD, HESSIAN] = FMINCON(inline
('2*10*(x(1)*x(2)+x(1)*x(3)+x(2)*x(3))'), [1;1;1], [], [], [], [], [0;0;0], [Inf;Inf;Inf], @nlcon)

nlcon.m

function [C, Ceq] = nlcon(x)

C=[];
Ceq=[x(1)*x(2)*x(3)-6];

The response is:

Warning: Large-scale (trust region) method does not currently solve this type of problem,
switching to medium-scale (line search).
> In C:\matlabR12\toolbox\optim\fmincon.m at line 213
Optimization terminated successfully:
  Magnitude of directional derivative in search direction
  less than 2*options.TolFun and maximum constraint violation
  is less than options.TolCon
Active Constraints:
  1

X =
  1.81714036486411e+000
  1.81714036486465e+000
  1.81713705446913e+000

FVAL =
  1.981156349255286e+002

EXITFLAG =
  1

OUTPUT =
  iterations: 5
  funcCount: 29
  stepsize: 2.500000000000000e-001
  algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
  firstorderopt: []
  cgiterations: []

LAMBDA =
  lower: [3x1 double]
  upper: [3x1 double]
  eqlin: [0x1 double]
  eqnonlin: 2.201284842268431e+001
  ineqlin: [0x1 double]
  ineqnonlin: 0

GRAD =
  7.268499854815508e+001
  7.268499854815508e+001
  7.268447405972370e+001

HESSIAN =
  4.778571206995651e+000  3.778571207061813e+000  -5.74000854988353e+000
  3.778571207061813e+000  4.778571207127976e+000  -5.740008549699711e+000
  -5.74000854988353e+000  -5.740008549699711e+000  1.42427261762578e+001

Note that based on the sign of LAMBDA.eqnonlin, Matlab uses the positive null form. Try this simple problem for other initial points to test if Matlab converges to the solution.

Example 2: maximization

max \ x_1 x_2 + x_2 x_3 + x_1 x_3 
such that \ x_1 + x_2 + x_3 = 0
We will let $x_1 \equiv x(1)$, $x_2 \equiv x(2)$ and $x_3 \equiv x(3)$.

```matlab
>> [X,FVAL,EXITFLAG,OUTPUT,LAMBDA,GRAD,HESSIAN]=FMINCON(inline('-(x(1)*x(2)+x(2)*x(3)+x(1)*x(3))'),[1;1;1],[],[],[1,1,1],[3],[-Inf;-Inf;-Inf],[Inf;Inf;Inf])
```

Example 3: minimization with inequality constraints

\[
\begin{align*}
\text{min} & \quad 2x_1^2 + 2x_1x_2 + x_2^2 - 10x_1 - 10x_2 \\
\text{such that} & \quad x_1^2 + x_2^2 \leq 5 \\
& \quad 3x_1 + x_2 \leq 6
\end{align*}
\]

We will let $x_1 \equiv x(1)$ and $x_2 \equiv x(2)$.

```matlab
>> [X,FVAL,EXITFLAG,OUTPUT,LAMBDA]=FMINCON(inline('2*x(1).^2 +2*x(1)*x(2)+x(2).^2-10*x(1)-10*x(2)'),[1;1],[],[],[],[],[-Inf;-Inf;-Inf],[Inf;Inf;Inf],@nlcon)
```

\text{nlcon.m}

```matlab
function [C,Ceq]=nlcon(x)

C=[x(1).^2+x(2).^2-5;3*x(1)+x(2)-6];
Ceq=[];
```