EN.553.761: Nonlinear Optimization I

Homework Assignment #3

Starred exercises require the use of MATLAB.

Exercise 3.1: Solve the trust-region subproblem

$$\underset{s\in\mathbb{R}^n}{\text{minimize}} \quad s^T g + \tfrac{1}{2} s^T B s \quad \text{subject to} \quad \|s\|_2 \le \delta \tag{1}$$

in the following cases:

(a)

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}, \quad g = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad \delta = 2,$$

5/5

Ans:

First, we need to check if B is positive semi-definite matrix. set $x = (x_1, x_2, x_3)^T$ where $x_1, x_2, x_3 \neq 0$

Then

$$x^T B x = x_1^2 + 2x_2^2 + 2x_3^2 > 0 \text{ for } x_1, x_2, x_3 \neq 0$$

Thus B is positive definite matrix

According Theorem 2.5, if $\|S_*\| \le \delta$ where $B s_* = -g$, S_* is the global minimizer of $s^T g + \frac{1}{2} s^T B s$

Thus

$$B s_* = -g \Rightarrow S_* = -B^{-1} g = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$S_* = \begin{bmatrix} -1 \\ 0 \\ -\frac{1}{2} \end{bmatrix} \qquad \|S_*\|_2 = \sqrt{1^2 + (\tfrac{1}{2})^2} = \frac{\sqrt{5}}{2} < 2$$

$\because \|S_*\| = \frac{\sqrt{5}}{2} < 2$

Thus, we can say that $S_* = \begin{bmatrix} -1 \\ 0 \\ -\frac{1}{2} \end{bmatrix}$ is the global minimizer of $s^T g + \frac{1}{2} s^T B s$.

$$\min \left(s^T g + \frac{1}{2} s^T B s \right) = S_*^T g + \frac{1}{2} S_*^T B S_*$$

$$= -0.75$$

(b)

5/5

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}, \quad g = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad \delta = 5/12$$

Hint: $\lambda = 2$ is a root of the nonlinear equation

$$\frac{1}{(1+\lambda)^2} + \frac{1}{(2+\lambda)^2} = \frac{25}{144}.$$

Ans:

From previous question, we know that B is a positive definite matrix, and $\|S_*\| = \frac{\sqrt{5}}{2}$

$\because \|S_*\| = \frac{\sqrt{5}}{2} > \delta = \frac{5}{12}$, which mean the optimal global minimizer is outside the trust-region.

Thus, it must exist a global minimizer at the boundary of trust-region according to theorem 2.5 where $(B + \lambda_* I) S_* = -g$, $\|S_*\|_2 = \delta$ and $\lambda_* > -\lambda_n$

$$s(\lambda) = -(B+\lambda I)^{-1} g = -\sum_{i=1}^{n} \frac{v_i^T g}{\lambda_i + \lambda} v_i$$

Due to the special feature of diagonal matrix it's easy to get the eigenvector and eigenvalue pair.

Thus

$$s(\lambda) = \left[-\frac{1}{\lambda+1} \quad 0 \quad -\frac{1}{\lambda+2} \right]^T$$

$$\|s(\lambda)\|_2^2 = \frac{1}{(\lambda+1)^2} + \frac{1}{(\lambda+2)^2}$$

To find λ^*, we need to make $\|s(\lambda)\|_2^2 = \delta^2$

Thus

$$\|s(\lambda^*)\|_2^2 = \frac{1}{(\lambda^*+1)^2} + \frac{1}{(\lambda^*+2)^2} = \delta^2 = \frac{25}{144}$$

Based on the hint, we know $\lambda = 2$ is a root to the nonlinear equation, and

$$\lambda = 2 > \max(0, -\lambda_n) = \max(0, -1) = 0$$

$$s_* = s(\lambda=2) = \left[-\frac{1}{3} \quad 0 \quad -\frac{1}{4} \right]^T$$

s_* is the global minimizer of $s^T g + \frac{1}{2} s^T B s$

on the boundary of trust-region

$$\min\left(s^Tg+\frac{1}{2}s^TBs\right)=s_*^Tg+\frac{1}{2}s_*^TBs_*$$

$$=-\frac{67}{144}$$

(c)

5/5

$$B=\begin{pmatrix}-2&0&0\\0&-1&0\\0&0&-1\end{pmatrix},\quad g=\begin{pmatrix}1\\0\\1\end{pmatrix},\quad \text{and}\quad \delta=5/12,$$

Ans:

First, to check if B is positive semidefinite matrix

Set $x=[x_1\ x_2\ x_3]^T$ and $x_1, x_2, x_3 \neq 0$

$x^TBx=-2x_1^2-x_2^2-x_3^2<0$, which implies that B is not positive semidefinite matrix, and thus $s^Tg+\frac{1}{2}s^TBs$ is not a convex function. Thus it should exist a minimizer at the boundary of trust-region

$$s(\lambda)=-(B+\lambda I)^{-1}g=-\sum_{i=1}^{n}\frac{v_i^Tg}{\lambda_i+\lambda}v_i$$

Thus

$$s(\lambda)=\left[-\frac{1}{\lambda-2}\quad 0\quad -\frac{1}{\lambda-1}\right]^T$$

$$\|s(\lambda)\|_2^2 = \frac{1}{(\lambda-2)^2} + \frac{1}{(\lambda-1)^2}$$

To find the minimizer, we need make

$$\|s(\lambda)\|_2^2 = \delta^2$$

$$\|s(\lambda)\|_2^2 = \frac{1}{(\lambda-2)^2} + \frac{1}{(\lambda-1)^2} = \delta^2 = \frac{25}{144}$$

From previous question, we can find the positive root for this equation: $\lambda = 5$

$$\frac{1}{(5-2)^2} + \frac{1}{(5-1)^2} = \frac{1}{3^2} + \frac{1}{4^2} = \frac{25}{144}$$

and $\lambda = 5 > \max(0, -\lambda_n) = \max(0, 2) = 2$

$$s_* = s(\lambda=5) = \left[-\frac{1}{3} \quad 0 \quad -\frac{1}{4}\right]^T$$

s_* is the minimizer of $s^Tg + \frac{1}{2}s^TBs$ on the boundary of trust-region

$$\min(s^Tg + \frac{1}{2}s^TBs) = s_*^Tg + \frac{1}{2}s_*^TBs$$

$$= -\frac{209}{288}$$

(d)

5/5

$$B = \begin{pmatrix} -2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \quad g = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad \delta = 1/2, \quad \text{and}$$

From previous question, we know that B is not a positive semidefinite matrix and thus $s^T g + \frac{1}{2} s^T B s$ is not a convex function.

with $g = [0 \ 0 \ 1]^T$, we can rewrite $s(\lambda)$ by following form

$$s(\lambda) = \left[0 \quad 0 \quad -\frac{1}{\lambda - 1} \right]^T$$

$$\|s(\lambda)\|_2^2 = \frac{1}{(\lambda-1)^2}$$

when $\lambda = -\lambda_n = 2$

$$\|s(\lambda)\|_2^2 = \frac{1}{(2-1)^2} = 1 > \delta^2 = \frac{1}{4}$$

Thus there is a obvious minimizer that exist at the boundary of trust-region by $\|s(\lambda)\|_2^2 = \delta^2$

$$\Rightarrow \frac{1}{(\lambda-1)^2} = \frac{1}{4} \Rightarrow \lambda = 3$$

$\lambda = 3 > \max(0, -\lambda_n) = \max(0, 2) = 2$

Thus

$$s_* = s(\lambda = 3) = \begin{bmatrix} 0 & 0 & -\frac{1}{2} \end{bmatrix}^T$$

s_* is the minimizer of $s^T g + \frac{1}{2} s^T B s$ on the boundary of trust-region

$$\min(s^T g + \tfrac{1}{2} s^T B s) = s_*^T g + \tfrac{1}{2} s_*^T B s_*$$

$$= -\frac{5}{8}$$

(e)

5/5

$$B = \begin{pmatrix} -2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \quad g = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad \delta = \sqrt{2}.$$

Ans:

From previous question, we note that there is a obvious solution to $\|s(\lambda)\|_2^2 = \delta^2$ when $\|s(-\lambda_n)\|_2^2 > \delta^2$. However, in this case

$$\|s(2)\|_2^2 = 1 < \delta^2 = 2$$

Thus, there is no obvious solution to $\|s(\lambda)\|_2^2 = \delta^2$, we need to find $s_* = s_{lim} + \sigma v_n$

where

$$s_{lim} = \lim_{\lambda \to -\lambda_n} s(\lambda)$$

Thus

$$s_{lim} = \lim_{\lambda \to -\lambda_n} s(\lambda) = s(2) = [0 \quad 0 \quad -1]^T$$

σ is supposed to be satisfy

$$\| s_{lim} + \sigma v_n \|_2 = \delta$$

Thus

$$s_{lim} + \sigma v_n = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + \sigma \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \sigma \\ 0 \\ -1 \end{bmatrix}$$

$$\| s_{lim} + \sigma v_n \|_2 = \sqrt{\sigma^2 + 1} = \delta = \sqrt{2}$$

$$\Rightarrow \quad \sigma^2 = 1 \quad \Rightarrow \quad \sigma = \pm 1$$

when $\sigma = 1$

$$s_* = [1 \quad 0 \quad -1]^T$$

when $\sigma = -1$

$$s_* = [-1 \quad 0 \quad -1]^T$$

When $s_* = [1 \; 0 \; -1]^T$

$$s_*^T g + \frac{1}{2} s_*^T B s_* = -\frac{5}{2}$$

when $s_* = [-1 \; 0 \; -1]^T$

$$s_*^T g + \frac{1}{2} s_*^T B s_* = -\frac{5}{2}$$

Thus, we can say that

$\min (s^T g + \frac{1}{2} s^T B s) = -\frac{5}{2}$ when $s = [1 \; 0 \; -1]^T$

or $[-1 \; 0 \; -1]^T$

Exercise 3.2: Consider the solution of problem (1) with data

10/10

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix} \quad \text{and} \quad g = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

as a function of the trust-region radius δ. In which direction does the solution point as δ shrinks to zero, i.e., what can you say about $\lim_{\delta \to 0} \frac{s(\delta)}{\|s(\delta)\|}$?

Ans:

First, to check if B is positive definite matrix,

set $x = [x_1 \; x_2]^T \quad x_1, x_2 \neq 0$

$$x^T B x = x_1^2 + 3x_2^2 > 0$$

Thus B is a positive definite matrix and thus

$s^T g + \frac{1}{2} s^T B s$ is a convex function. which implies that there is alway a solution to $\|s(\lambda)\|_2 = \delta$ $\forall\ \delta > 0$.

To find the direction of the solution point as δ shrinks to zero, we can find

$$\lim_{\delta \to 0} \frac{s(\delta)}{\|s(\delta)\|_2} \stackrel{\text{def}}{=} \lim_{\lambda \to \infty} \frac{s(\lambda)}{\|s(\lambda)\|_2}$$

as λ is approaching ∞ when δ is approaching to zero.

$$s(\lambda) = \left[\frac{-1}{\lambda+1} \quad \frac{-1}{\lambda+3} \right]^T$$

$$\|s(\lambda)\|_2 = \sqrt{\frac{1}{(\lambda+1)^2} + \frac{1}{(\lambda+3)^2}}$$

$$\frac{s(\lambda)}{\|s(\lambda)\|_2} = \begin{bmatrix} -\sqrt{\dfrac{1}{1+\dfrac{(\lambda+1)^2}{(\lambda+3)^2}}} \\ -\sqrt{\dfrac{1}{1+\dfrac{(\lambda+3)^2}{(\lambda+1)^2}}} \end{bmatrix}$$

$$\lim_{\lambda\to\infty}\frac{s(\lambda)}{\|s(\lambda)\|_2} = \begin{bmatrix} \lim\limits_{\lambda\to\infty} -\dfrac{1}{\sqrt{1+\frac{(\lambda+1)^2}{(\lambda+3)^2}}} \\ \lim\limits_{\lambda\to\infty} -\dfrac{1}{\sqrt{1+\frac{(\lambda+3)^2}{(\lambda+1)^2}}} \end{bmatrix}$$

$$\because \lim_{\lambda\to\infty}\frac{(\lambda+1)^2}{(\lambda+3)^2} = 1 \quad \lim_{\lambda\to\infty}\frac{(\lambda+3)^2}{(\lambda+1)^2} = 1$$

$$\therefore \lim_{\lambda\to\infty}\frac{s(\lambda)}{\|s(\lambda)\|} = \begin{bmatrix} -\frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix}$$

Exercise 3.5*: Recall the Levenberg-Marquardt method for solving the nonlinear least squares problem. The trust region subproblem at the iterate x_k is (see slides 29–30 in the "Least Squares" slides)

$$\min_{s\in\mathbb{R}^n} \frac{1}{2}\|F(x_k)\|_2^2 + s^T J(x_k)^T F(x_k) + \frac{1}{2}s^T J^T(x_k)J(x_k)s \qquad \text{subject to} \qquad \|s\|_2 \le \delta_k$$

From the characterization of global optima for the above problem, we need to find $\lambda^* \ge 0$ and $s^* \in \mathbb{R}^n$ such that

1. $\lambda^* \ge 0$
2. $(J^T(x_k)J(x_k) + \lambda^* I)s^* = -J(x_k)^T F(x_k)$
3. $\lambda^*(\|s^*\| - \delta_k) = 0$

If solve this by applying Newton's method to the *secular equation* $\phi(\lambda) := \frac{1}{\|s(\lambda)\|_2} - \frac{1}{\delta_k} = 0$, then recall that each step requires the computation of the Newton step $-\phi(\lambda)/\phi'(\lambda)$ (for some fixed $\lambda > 0$). From the computations on slide 72 in the "Trust Region Methods" slides, we see that $\phi'(\lambda) = -\frac{s(\lambda)^T \nabla s(\lambda)}{\|s(\lambda)\|_2^3}$, where $s(\lambda)$ is the solution to

$$(J^T(x_k)J(x_k) + \lambda I)s(\lambda) = -J(x_k)^T F(x_k), \tag{2}$$

and $\nabla s(\lambda)$ is the solution to

$$(J^T(x_k)J(x_k) + \lambda I)\nabla s(\lambda) = -s(\lambda). \tag{3}$$

1. Show that the linear system (2) are the normal equations for the linear least squares problem

$$\min_{s\in\mathbb{R}^n} \frac{1}{2}\left\| \begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix} \right\|_2^2. \tag{4}$$

Ans: 5/5

To find the minimum of $\left\| \begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix} \right\|_2^2$

we want $\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix}$ orthgonal to

to the column space of $\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}$, which implies

$$\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}^T \cdot \left[\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix} \right] = 0$$

$\Rightarrow$

$$\begin{pmatrix} J^T(x_k) & \sqrt{\lambda} I \end{pmatrix} \left[\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix} \right] = 0$$

$\Rightarrow$

$$(J^T(x_k) J(x_k) + \lambda I) s + J^T(x_k) F(x_k) = 0$$

$\Rightarrow$

$$(J^T(x_k) J(x_k) + \lambda I) s = -J^T(x_k) F(x_k) \quad \text{--- (1)}$$

which show that (1) is the normal solution

to $\min\limits_{s \in \mathbb{R}^n} \frac{1}{2} \left\| \begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} s + \begin{pmatrix} F(x_k) \\ 0 \end{pmatrix} \right\|$

2. Show that if we use the QR decomposition to solve the linear least squares problem (4) to compute $s(\lambda)$, then one can compute $\nabla s(\lambda)$ from (3) by just solving two triangular linear $J|J$ systems involving the R matrix in the QR decomposition.

If we use QR decomposition to solve (4)
then we need to factorize $\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}$ into
R and Q matrixs where

$$\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} = QR$$ (R is the upper triangular matrix)

To solve

$$\left(J^T(x_k)J(x_k)+\lambda I\right)\nabla S(\lambda)=-S(\lambda)$$

$$J^T(x_k)J(x_k)+\lambda I=\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}^T \cdot \begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}$$

$$=(QR)^T QR$$

$$=R^T Q^T Q R$$

Thus, we can rewrite the equation as following form:

$$R^T Q^T Q R \nabla S(\lambda)=-S(\lambda)$$

If we set $Q^T Q R \nabla S(\lambda)=y$, then

$$R^T y=-S(\lambda)$$

To compute y, we need to solve one triangular linear system by back substitution.

After we obtain y, we can compute

$\nabla s(\lambda)$ by following equation:

$$Q^T Q R \nabla s(\lambda) = y$$

Based on fact that Q is the orthgonal matrix

Thus $Q^T Q = I$, then

$$Q^T Q R \nabla s(\lambda) = y \Rightarrow R \nabla s(\lambda) = y$$

we can compute $\nabla s(\lambda)$ by solving this triangular linear system with back substitution

3. Show that if we use the SVD decomposition to solve the linear least squares problem (4) to compute $s(\lambda)$, then one can compute $\nabla s(\lambda)$ from (3) by using only matrix-matrix and 4/5 matrix-vector products, with no need for any linear system solves.

If we use SVD decomposition to solve (4), then we need to factorize $\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}$ into following form:

$$\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} = U \Sigma V^T$$ where U and V are

orthgonal basis for columnspace and rowspace respectivly.

To solve

$$(J_{(x_k)}^T J_{(x_k)} + \lambda I)\nabla S(\lambda) = -S(\lambda)$$

we can rewrite the equation in the following form

$$\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}^T \begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix} \nabla S(\lambda) = -S(\lambda)$$

$$(U \Sigma V^T)^T (U \Sigma V^T) \nabla S(\lambda) = -S(\lambda)$$

$\because$ both U and V are orthogonal matrix

$\therefore$ $V^T = V^{-1}$ $\quad V^T \cdot V = I$

$$(V \Sigma^T U^T \cdot U \Sigma V^T) \nabla S(\lambda) = -S(\lambda)$$

$\Sigma^T \Sigma$ You need to make sure the matrix multiplication has meaning and inertible.

$$V \Sigma^2 V^T \nabla S(\lambda) = -S(\lambda)$$

$$\Rightarrow \quad \nabla S(\lambda) = -V \Sigma^{-2} V^T \cdot S(\lambda)$$

By factorizing $\begin{pmatrix} J(x_k) \\ \sqrt{\lambda} I \end{pmatrix}$, we can compute $\nabla s(\lambda)$ by $\nabla s(\lambda) = -V\Sigma^{-2}V^T \cdot s(\lambda)$ with only matrix-matrix and matrix-vector products.

6/10 3.3

steihaug_CG.m

```
function [ p, iters, flag ] = steihaug_CG( B, g, radius, tol )

% B needs to be symmetric
assert(issymmetric(B),'Input B needs to be symmetric!')

p0 = zeros(length(g),1);
r0 = g;
s0 = -g;

rk = r0;
sk = s0;
pk = p0;

iters = 0;
flag = 0;

while norm(rk) > tol*norm(r0)

    if sk'*B*sk > 0
        alpha = rk'*rk / (sk'*B*sk);
        if norm(pk+alpha*sk) < radius
            pk = pk + alpha*sk;
        else
            flag = 1; % boundary solution
            root1 = (-2*pk'*sk+sqrt(4*norm(sk)^2*radius^2)) /
 (2*norm(sk)^2);
            root2 = (-2*pk'*sk-sqrt(4*norm(sk)^2*radius^2)) /
 (2*norm(sk)^2);
            tao = max(root1, root2);
            assert(tao>0, 'tao needs to be positive root of |pk
+tao*sk| = delta')
            pk = pk + tao*sk;
            p = pk;
            break
        end
    else
        flag = -1; % negative curvature
        root1 = (-2*pk'*sk+sqrt(4*norm(sk)^2*radius^2)) /
 (2*norm(sk)^2);
        root2 = (-2*pk'*sk-sqrt(4*norm(sk)^2*radius^2)) /
 (2*norm(sk)^2);
        tao = max(root1, root2);
        assert(tao>0, 'tao needs to be positive root of |pk+tao*sk| =
 delta')
        pk = pk + tao*sk;
        p = pk;
        break
    end

rk1 = rk + alpha*B*sk;
beta = rk1'*rk1 / (rk'*rk);
sk = -rk1 + beta*sk;
```

the computation of roots is wrong -2

-2

```
rk = rk1;

iters = iters+1;
end

if norm(rk) <= tol*norm(r0)
    flag = 0; % tol was met
end

p = pk;

end
```

Not enough input arguments.

Error in steihaug_CG (line 4)
assert(issymmetric(B),'Input B needs to be symmetric!')

Published with MATLAB® R2018b

calculate F,G,H

```
function [F,G,H] = fgh(x)
    x1 = x(1);
    x2 = x(2);
    F = 10*(x2-x1^2)^2 + (x1-1)^2;

    G = [-40*(x2*x1-x1^3) + 2*(x1-1);
         20*(x2-x1^2)];

    H = [-40*(x2-3*x1^2)+2, -40*x1;
        -40*x1, 20];
end
```

```
Not enough input arguments.

Error in fgh (line 2)
    x1 = x(1);
```

Published with MATLAB® R2018b

5 / 5 3.4 (a)


```
function [x,F,G,H,iter,status] = unc_TR(Fun,x0,maxit,printlevel,tol)

    func = str2func(Fun);
    %
    iter      = 0        ;
    [F0,G0,H0]  = func(x0);
    F         = F0       ;
    G         = G0       ;
    H         = H0       ;
    x         = x0       ;

%     m = F;

    radius = 2;
    gamma_d = 0.5;
    gamma_i = 2;
    eta_s = 0.1;
    eta_vs = 0.9;

    status = 0;

    if printlevel == 1
        fprintf('\n iter       |F|            |G|            |H|\n');
    end

    while norm(G) > tol*max(1,norm(G0)) && iter < maxit

        iter = iter + 1;

        [ p, iters, flag ] = steihaug_CG( H, G, radius, 1e-4 );

        [F_new, G_new, H_new] = func(x+p);
        mk0 = F;
        mk_sk = F + G'*p + 0.5*p'*H*p;
        rho = (F - F_new) / (mk0 - mk_sk);

        if rho >= eta_vs
            x = x+p;
            radius = radius * gamma_i;
        elseif rho >= eta_s
            x = x+p;
        else
            radius = radius * gamma_d;
        end

        [F, G, H] = func(x);
        if printlevel == 1
           fprintf(' %4g  %13.6e  %13.6e %13.6e \n', iter, norm(F),
 norm(G), norm(H))
        end

    end
```

```
    if iter >= maxit
        status = 1;
    end

    end
```

```
Not enough input arguments.

Error in unc_TR (line 3)
    func = str2func(Fun);
```

Published with MATLAB® R2018b

5/5 3.4 (6)

3.4 results

```
tol       = 1.0e-8  ; % VERY tight stopping tolerance
maxit     = 50       ; % maximum number of iterations allowed
x0        = [0; 0]  ; % initial guess at a solution
Fun       = 'fgh';

printlevel = 1;

[x,F,G,H,iter,status] = unc_TR(Fun,x0,maxit,printlevel,tol);

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('x is:')
disp(x)
disp('How many iterations were done:')
disp(iter)
disp('If the final stopping tolerance was met (if met, status = 0):')
disp(status)
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
```

```
 iter        |F|            |G|            |H|
    1   1.000000e+00   2.000000e+00   2.000000e+01
    2   1.000000e+00   2.000000e+00   2.000000e+01
    3   8.750000e-01   6.403124e+00   4.688061e+01
    4   1.740934e-01   6.855136e-01   4.856191e+01
    5   1.740934e-01   6.855136e-01   4.856191e+01
    6   1.301507e-01   4.413837e+00   9.096496e+01
    7   4.011378e-03   9.714258e-02   9.178519e+01
    8   1.512839e-04   1.713688e-01   1.015669e+02
    9   5.439316e-09   1.127031e-04   1.015945e+02
   10   2.967268e-16   2.404996e-07   1.016063e+02
   11   2.341931e-31   5.568850e-15   1.016063e+02
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x is:
    1.0000
    1.0000

How many iterations were done:
    11

If the final stopping tolerance was met (if met, status = 0):
     0

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Published with MATLAB® R2018b