UNCONSTRAINED OPTIMIZATION ALGORITHMS IN MATLAB V7 / OPTIMIZATION TOOLBOX

http://www.mathworks.com/access/helpdesk/help/helpdesk.html

- The simplest algorithms are part of the basic distribution of MATLAB.

FUNCTIONS IN THE STANDARD VERSION

fminbnd:

Minimum of a function within an interval [a,b]
(Old name: **fmin**)

fminsearch:

Minimum of a function with *few* (2-6) variables (The *Amoeba*)
(Old name: **fmins**)

fminbnd

SYNTAX:

```
x = fminbnd(fun,x1,x2)
x = fminbnd(fun,x1,x2,options)
x = fminbnd(fun,x1,x2,options,P1,P2,...)
[x,fval] = fminbnd(...)
[x,fval,exitflag] = fminbnd(...)
[x,fval,exitflag,output] = fminbnd(...)
```

fun: m-file, *built-in function* or an *inline object*.
[x1 x2]: search interval
options: *Structure*[*)] of parameters. Set by the **optimset** function.
P1, P2, …: Parameters needed in and passed on to **fun**.

ALLOWED OPTIONS:

Display	**off**=no output; **iter** =output at each iteration; **final** displays just the final output.
MaxFunEvals	Maximum number of function evaluations allowed.
MaxIter	Maximum number of iterations allowed.
TolX	Termination tolerance on x.

*) Vector/array of “things”

EXAMPLE:

[x,fval,exitflag] =fminbnd('cos',3,4,optimset('TolX',1e-12,'Display','iter'))

Function-count	x	f(x)	Procedure
1	3.38197	-0.971249	initial
2	3.61803	-0.888633	golden
3	3.23607	-0.995541	golden
4	3.13571	-0.999983	parabolic
5	3.1413	-1	parabolic
6	3.14159	-1	parabolic
7	3.14159	-1	parabolic
8	3.14159	-1	parabolic
9	3.14159	-1	parabolic

Optimization terminated successfully: The current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-012

For a short version of output, use : **[x,fval,exitflag,output] = …**

fminsearch

SYNTAX

```
x   =   fminsearch(fun,x0)
x   =   fminsearch(fun,x0,options)
x   =   fminsearch(fun,x0,options,P1,P2,...)
[x,fval]                    = fminsearch(...)
[x,fval,exitflag]           = fminsearch(...)
[x,fval,exitflag,output]    = fminsearch(...)
```

OPTIONS structure fields:

As for **fminbnd** +

Display	**simplex** (all corners)
TolFun	Termination tolerance on the function value.

Note: The search also includes an ***outside contraction***

EXAMPLE:

[x,fval] = fminsearch('banana', [-1.2, 1] , optimset('TolX',1e-3,'Display','iter'))

Iteration	Func.-count	min f(x)	Procedure
1	3	6.361	initial
2	5	4.40994	expand
3	7	4.29678	expand
4	9	4.29678	contract outside
5	11	4.17987	contract inside
6	13	4.17987	contract outside
7	15	4.17987	contract inside
8	16	4.17987	reflect
9	18	4.07932	expand
10	19	4.07932	reflect
.....			
63	119	2.0633e-007	contract inside
64	120	2.0633e-007	reflect
65	122	2.0633e-007	contract inside
66	124	2.0633e-007	contract inside
67	126	6.10245e-008	contract inside

optimset

(Type optimset on command line for full info about available options)

SYNTAX:

```
options = optimset('param1',value1,'param2',value2,...)
options = optimset
options = optimset(optimfun)
options = optimset(oldopts,'param1',value1,...)
options = optimset(oldopts,newopts)
```

PARAMETERS:

MATLAB AND OPTIMIZATION TOOLBOX:

Display	[off | iter | simplex|{final}]
MaxFunEvals	[positive integer]
MaxIter	[positive integer]
TolFun	[positive scalar]
TolX	[positive scalar]

TOOLBOX ONLY (*incomplete*):

DerivativeCheck	[on | {off}]
Diagnostics	[on | {off}]
DiffMaxChange	[positive scalar | {1e-1}]
DiffMinChange	[positive scalar | {1e-8}]
GoalsExactAchieve	[positive scalar integer | {0}]
GradConstr	[on | {off}]
GradObj	[on | {off}]
Hessian	[on | {off}]
HessPattern	[sparse matrix]
HessUpdate	[{bfgs} | dfp | gillmurray | steepdesc]
JacobPattern	[sparse matrix]
Jacobian	[on | {off}]
LargeScale	[{on} | off]
LevenbergMarquardt	[on | off]
LineSearchType	[cubicpoly | {quadcubic}]
MaxPCGIter	[positive integer]
MeritFunction	[singleobj | {multiobj}]
MinAbsMax	[positive scalar integer | {0}]
PrecondBandWidth	[positive integer | Inf]
TolCon	[positive scalar]
TolPCG	[positive scalar | {0.1}]
TypicalX	[vector]

RUTINES IN OPTIMIZATION TOOLBOX

fminunc: Unconstrained minimization

Medium size: BFGS Quasi-Newton (or Amoeba). Different strategies for line search.

Large size: Trust region methods. Different strategies for the approximation to the Hessian matrix. Preconditioned Conjugate Gradient (PCG).

lsqnonlin: Non-linear least square (includes *simple* interval constraints)

Medium size: Gauss/Newton or Levenberg/Marquardt

Large size: Trust region methods or Gauss/Newton solved with PCG.

fminunc

Syntax

x = fminunc(fun,x0)
x = fminunc(fun,x0,options)
x = fminunc(fun,x0,options,P1,P2,...)
[x,fval] = fminunc(...)
[x,fval,exitflag] = fminunc(...)
[x,fval,exitflag,output] = fminunc(...)
[x,fval,exitflag,output,grad] = fminunc(...)
[x,fval,exitflag,output,grad,hessian] = fminunc(...)

Function, (gradient, (Hessian)):

```
function [f,g,H] = myfun(x)
    f = ...              % Compute the objective function value at x
    if nargout > 1       % fun called with two output arguments
      g = ...            % gradient of the function evaluated at x
      if nargout > 2     % fun called with two three arguments
      H = ...            % Hessian evaluated at x
    end
```

options = optimset('GradObj',’on’,'Hessian',’on’,...)

OPTIONS:

LargeScale

GradObj

Medium-scale algorithm only:

DerivativeCheck
DiffMaxChange
DiffMinChange
LineSearchType

Large-scale algorithm only:

HessPattern
MaxPCGIter
PrecondBandWidth
TolPCG
TypicalX

Example: fminunc

```
function [f,g] = uncoex(x)
f = 3*(x(1)-1)^2 + 2*x(1)*x(2) + x(2)^2;
if nargout > 1
   g(1) = 6*(x(1)-1)+2*x(2);
   g(2) = 2*x(1)+2*x(2);
end

options = optimset('GradObj','on','Display','iter',);
x0 = [10,9];
[x,fval] = fminunc('uncoex',x0,options)
```

Iteration	f(x)	Norm of step	First-order optimality *)	CG-iterations
1	504	1	72	0
2	23.8973	10	12.5	1
3	-1.5	4.34752	5.33e-015	1

Optimization terminated successfully:
First-order optimality less than OPTIONS.TolFun, and no negative/zero curvature detected

x = [1.5000 -1.5000]
fval = -1.5000

***) Infinity norm of the gradient.**

lsqnonlin

SYNTAX

```
x = lsqnonlin(fun,x0)
x = lsqnonlin(fun,x0,lb,ub)
x = lsqnonlin(fun,x0,lb,ub,options)
x = lsqnonlin(fun,x0,options,P1,P2, ... )

[x,resnorm]                                          = lsqnonlin(...)
[x,resnorm,residual]                                 = lsqnonlin(...)
[x,resnorm,residual,exitflag]                        = lsqnonlin(...)
[x,resnorm,residual,exitflag,output]                 = lsqnonlin(...)
[x,resnorm,residual,exitflag,output,lambda]          = lsqnonlin(...)
[x,resnorm,residual,exitflag,output,lambda,jacobian] = lsqnonlin(...)

function [F,J] = myfun(x)
F = ...              % objective function values at x
if nargout > 1       % two output arguments
  J = ...            % Jacobian of the function evaluated at x
end

options = optimset('Jacobian','on')
```

Both the large-scale and medium-scale algorithms:

Diagnostics
Display
Jacobian
MaxFunEvals
MaxIter
TolFun
TolX

Medium-scale algorithm only:

DerivativeChec
DiffMaxChange
DiffMinChange.
LevenbergMarquardt
LineSearchType

Large-scale algorithm only:

JacobPattern
MaxPCGIter
PrecondBandWidth
TolPCG.
TypicalX

Example: lsqnonlin

```
function F = nonlsqfun(x)
k = 1:20;
F = 2 + 2*k-exp(k*x(1))-exp(k*x(2));
options = optimset('Display','iter');
x0 = [0.3 0.4];
[x,resnorm] = lsqnonlin('nonlsqfun',x0,[ ],[ ],options)
```

Iter.	Func-count	f(x)	Norm of step	First-order optimality	CG-iterations
1	4	2.048e+007	1	3.49e+008	0
2	7	5.80e+006	0.231124	1.15e+008	1
3	10	850620	1.04515	1.77e+007	1
4	13	850620	10	1.77e+007	1
5	16	850620	2.5	1.77e+007	0

30	91	1449.48	1.19209e-006	0.0177	0
31	94	1449.48	1.19209e-006	0.0177	1
32	97	1449.48	2.98023e-007	0.00378	0

```
Optimization terminated successfully:
Norm of the current step is less than OPTIONS.TolX
 x =     0.1652    0.1652
 resnorm =  1.4495e+003
```