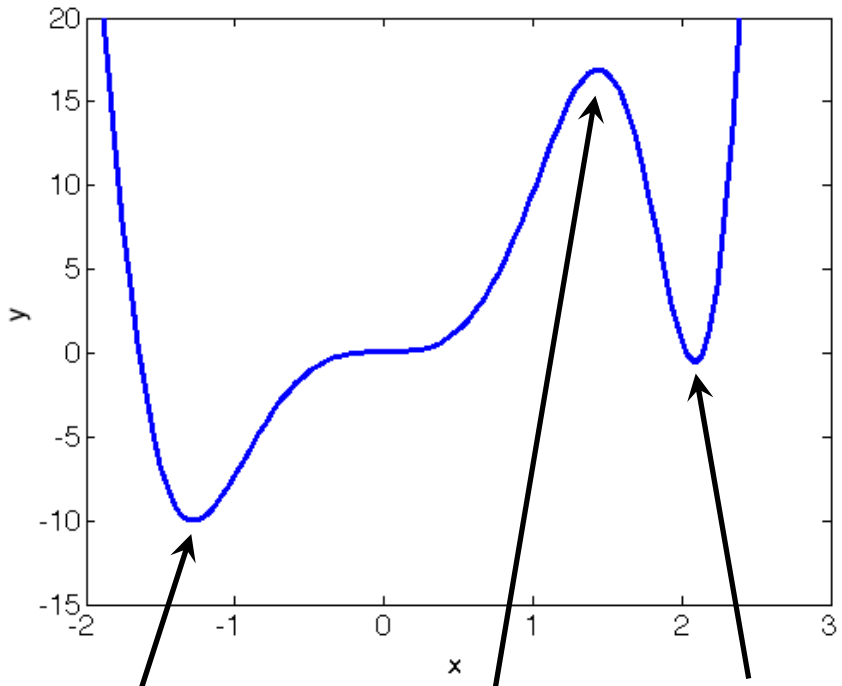
University of Washington
AMATH 301 Spring 2017
Instructor: Dr. King-Fai Li

Week 5
Optimization

1-D example

$$f(x) = x^4 + 10x\sin x^2$$

```
f = @(x) x.^4 + 10*x.*sin(x.^2);
x = linspace(-2,3,100);
plot(x,f(x));
```




```
[xmin,fmin]=fminbnd(f,-2,-1);
[xmin,fmin]=fminsearch(f,-1);
[xmin,fmin]=fminunc(f,-1);
```

```
[xmin,fmin]=fminbnd(f,1.5,2.5);
[xmin,fmin]=fminsearch(f,2);
[xmin,fmin]=fminunc(f,2);
```

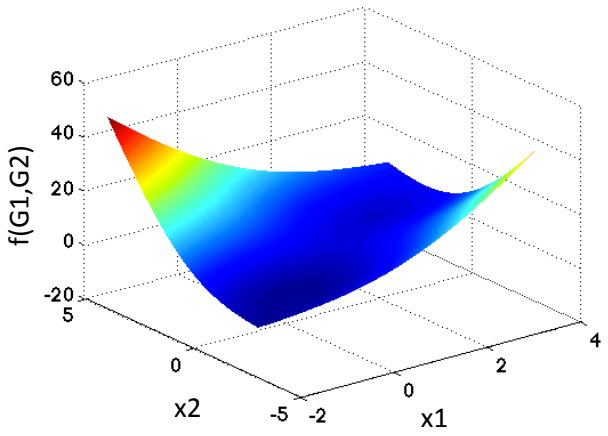
```
[xmax,fmax]=fminbnd(@(x)-f(x),1,2); fmax=-fmax;
[xmax,fmax]=fminsearch(@(x)-f(x),1); fmax=-fmax;
[xmax,fmax]=fminunc(@(x)-f(x),1); fmax=-fmax;
```

2-D example

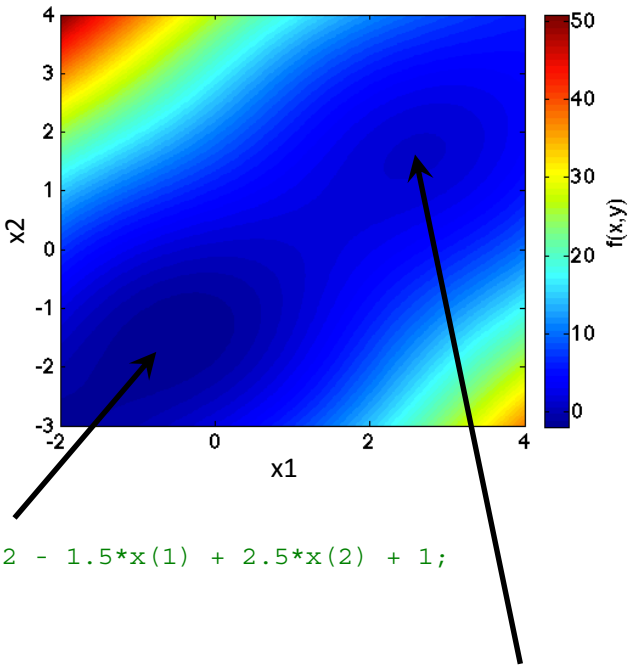
$$f(x_1, x_2) = \sin(x_1 + x_2) + (x_1 - x_2)^2 - 1.5x_1 + 2.5x_2 + 1$$

```
f = @(x1,x2) sin(x1 + x2) + (x1 - x2).^2 - 1.5*x1 + 2.5*x2 + 1;
x1 = -2:0.05:4;
x2 = -3:0.05:4;
[G1,G2] = meshgrid(x1,x2);
```

```
surf(x1,x2,f(G1,G2));
```




```
contourf(x1,x2,f(G1,G2));
```



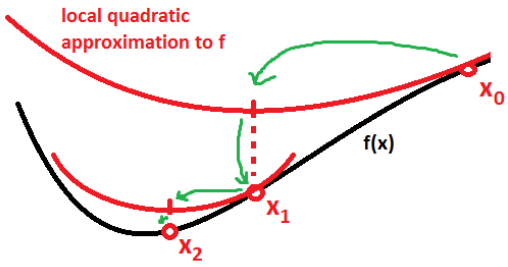
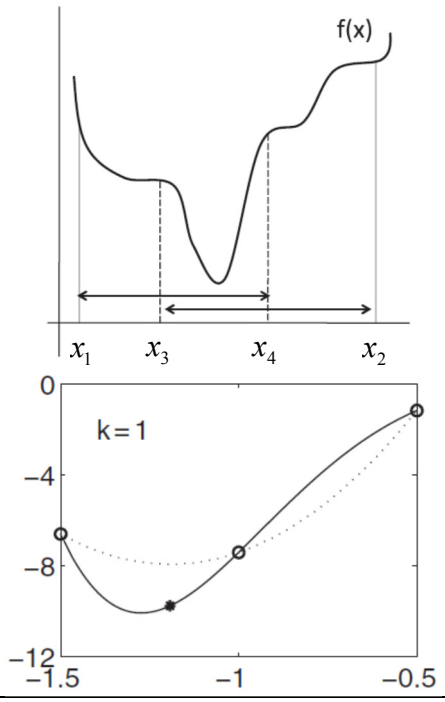
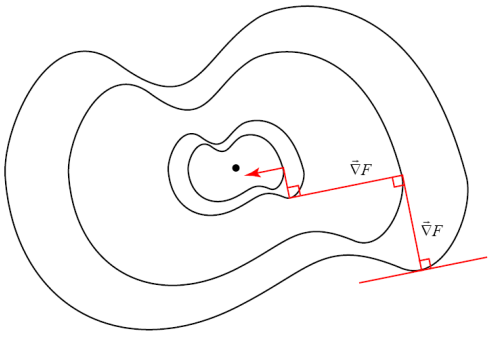
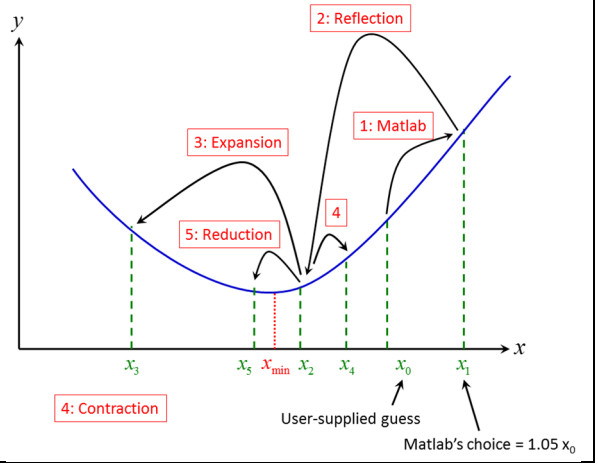

```
g = @(x) sin(x(1) + x(2)) + (x(1) - x(2)).^2 - 1.5*x(1) + 2.5*x(2) + 1;
[xmin,fmin]=fminunc(g,[-1,-2]);
[xmin,fmin]=fminsearch(g,[-1,-2]);
>> xmin = -0.5472   -1.5472
>> fmin = -1.9132
```

```
[xmin,fmin]=fminunc(g,[3,2]);
[xmin,fmin]=fminsearch(g,[3,2]);
>> xmin = 2.5944    1.5944
>> fmin = 1.2284
```

Note: `fminsearch` and `fminbnd` only accept function handles with one input argument. So instead of `f`, we have defined a new function handle `g` for the use in `fminsearch` and `fminbnd`. Alternatively, if you want to recycle `f`, you can do the following:

```
[xmin,fmin]=fminbnd(@(x)f(x(1),x(2)),[-1,-2]);
[xmin,fmin]=fminsearch(@(x)f(x(1),x(2)),[-1,-2]);
>> xmin = -0.5472   -1.5472
>> fmin = -1.9132
```

Algorithms:

<table>
<tr><th></th><th>Derivative</th><th>Derivative-free</th></tr>
<tr><td rowspan="2">1-D</td><td rowspan="2">Newton
<code>fminunc(f,x0)</code>
 
</td><td>Golden Section Search (φ = 0.61834)
Successive Parabolic
<code>fminbnd(f,x1,x2)</code>
 
</td></tr>
<tr><td>Nelder-Mead
<code>fminsearch(f,x0)</code></td></tr>
<tr><td>N-D</td><td>Steepest Descent
Conjugate Gradient
<code>fminunc(f,x0)</code>
 
</td><td>Nelder-Mead
<code>fminsearch(f,x0)</code>
 
</td></tr>
</table>

Linear Programming

Simplex methods: **`linprog`** — Check vertices (because of linearity)

`linprog(p,A,b,Aeq,beq,lb,ub,options)`

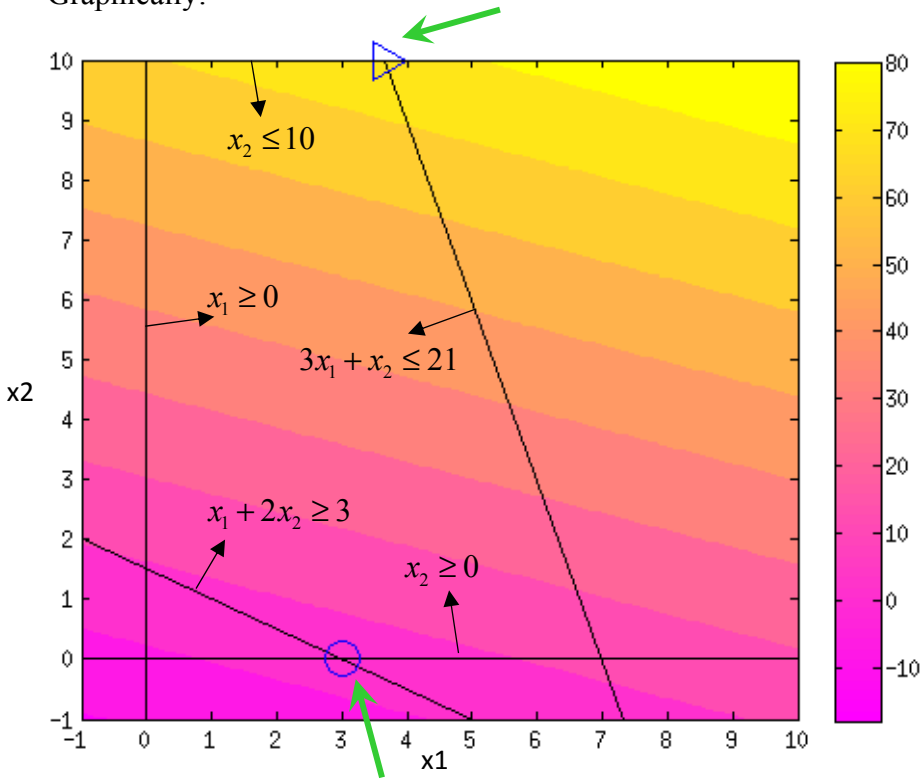
$$\text{where} \quad f(\mathbf{x}) = \mathbf{p}^T\mathbf{x} \quad \text{subject to} \quad \begin{cases} \mathbf{A}\mathbf{x} \le \mathbf{b} \\ \mathbf{A}_{eq}\mathbf{x} = \mathbf{b}_{eq} \\ \mathbf{l}_{eq} \le \mathbf{x} \le \mathbf{u}_{eq} \end{cases}$$

Example:

$$f(x_1, x_2) = 2x_1 + 7x_2 \quad \text{subject to} \quad \begin{cases} x_1 + 2x_2 \ge 3 \\ 3x_1 + x_2 \le 21 \\ x_1, x_2 \ge 0 \\ x_2 \le 10 \end{cases} \Rightarrow \begin{cases} -x_1 - 2x_2 \le -3 \\ 3x_1 + x_2 \le 21 \\ 0 \le x_1 \le \infty \\ 0 \le x_2 \le 10 \end{cases} \Rightarrow \begin{cases} \begin{bmatrix} -1 & -2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} -3 \\ 21 \end{bmatrix} \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix} \le \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} \infty \\ 10 \end{bmatrix} \end{cases}$$

```
A   = [-1 -2;3 1];
b   = [-3;21];
lb  = [0;0];
ub  = [inf;10];
p   = [2;7];
[xmin,fmin] = linprog(p,A,b,[],[],lb,ub);
[xmax,fmax] = linprog(-p,A,b,[],[],lb,ub); fmax=-fmax;
```

Graphically: `>> xmax = 3.6667  10.0000` `>> fmax = 77.3333`



`>> xmin = 3.0000  0.0000` `>> fmin = 6.0000`