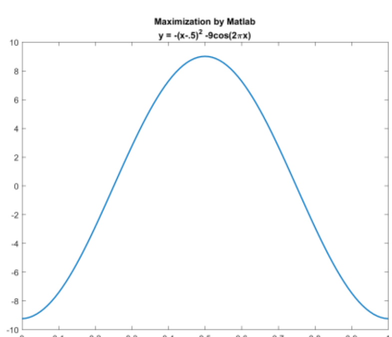
# Numerical Methods



## Topics

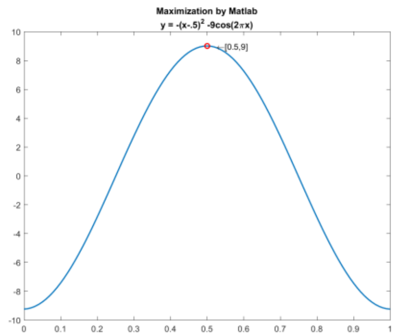
- Optimization
- Integration



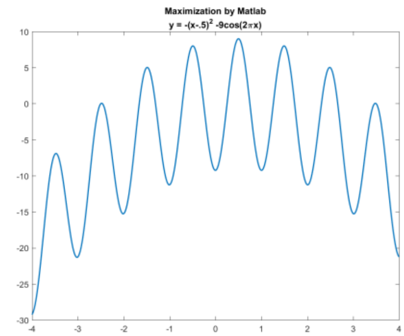




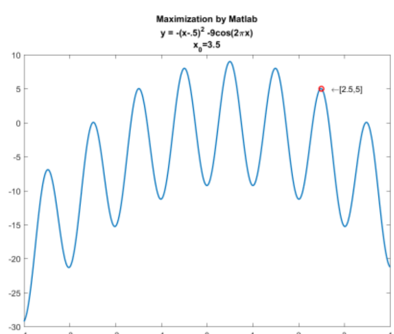

```
function illustrateMax
figure;
x=0:.01:1;
plot(x,y(x),'linewidth',1.5);
title({'Maximization by Matlab','y = -(x-.5)^2 -9cos(2\pix)'});
print -dpng max1

figure;
[xOpt,fval] = fminunc(@negY,0);
plot(x,y(x),xOpt,-fval,'or','linewidth',1.5);
title({'Maximization by Matlab','y = -(x-.5)^2 -9cos(2\pix)'});
text(xOpt,-fval,['     \leftarrow[',num2str(xOpt),',',num2str(-
        fval),']']);
print -dpng max2

end

function negY = negY(x)
negY = -y(x);
end

function y = y(x)
y = -(x-.5).^2 -9*cos(2*pi*x);
end
```










# Search algorithms

Objectives:

- Get close to the *global* right answer
- Compute as few objective functions as possible

Algorithms:

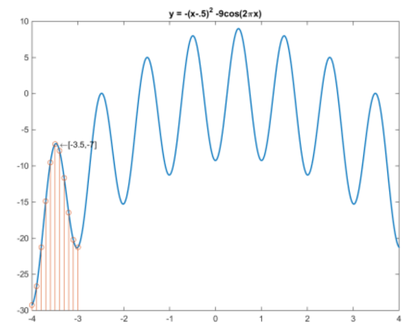
- Grid search
- Hill climbing

# Methods

Grid search
- How fine a grid?
- Where to locate grid?
- *How many* function evaluations did you say?

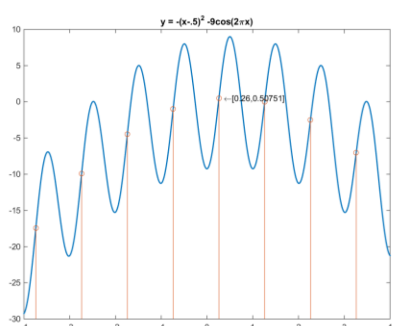
Hill climbing
- Global vs local optimum
- Pick search direction
- Choose step size
- Stopping rule



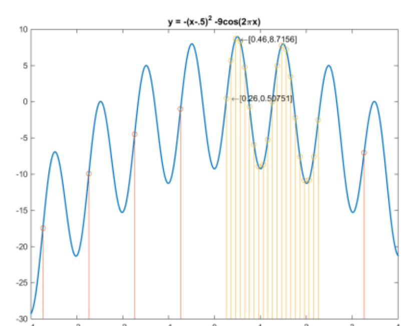










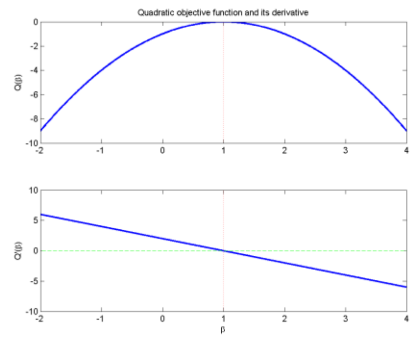


## Curse of dimensionality

1,000 grid points per dimension

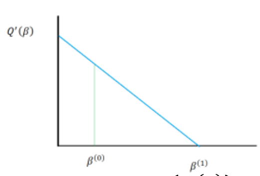
- 2 parameters → $1$ million function evaluations
- 10 parameters → $10^{30}$ function evaluations








## First derivative




$$Q''\left(\beta^{(0)}\right) = \frac{Q'\left(\beta^{(0)}\right) - 0}{\beta^{(0)} - \beta^{(1)}}$$



## Finding maximum

$$Q''\left(\beta^{(0)}\right) = \frac{Q'\left(\beta^{(0)}\right) - 0}{\beta^{(0)} - \beta^{(1)}}$$

Newton's method

$$\beta^{(1)} = \beta^{(0)} - \frac{Q'\left(\beta^{(0)}\right)}{Q''\left(\beta^{(0)}\right)}$$

## Least squares by Newton's method

$$-ssr(\hat{\beta}) = -\sum\left(y-\hat{\beta}x\right)^2$$
$$-ssr'(\hat{\beta}) = -2\sum\left(y-\hat{\beta}x\right)(-x)$$
$$= 2\left(\sum yx - \hat{\beta}\sum x^2\right)$$
$$-ssr''(\hat{\beta}) = -2\sum x^2$$
$$\beta^{(1)} = \beta^{(0)} - \frac{2\left(\sum yx - \beta^{(0)}\sum x^2\right)}{-2\sum x^2}$$
$$\beta^{(1)} = \beta^{(0)} + \frac{\sum yx}{\sum x^2} - \beta^{(0)}$$



## Newton's method example

$$Q(x) = -(x-.5)^2 - 9\cos(2\pi x)$$
$$Q'(x) = -2(x-.5) + 2\cdot 9\cdot\pi\cdot\sin(2\pi x)$$
$$Q''(x) = -2 + (2\pi)^2\cdot 9\cdot\cos(2\pi x)$$

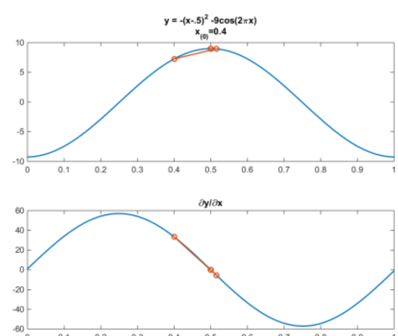
$$x^{(j+1)} = x^{(j)} - \frac{Q'\left(x^{(j)}\right)}{Q''\left(x^{(j)}\right)}$$
$$x^{(j+1)} = x^{(j)} - \frac{-2(x-.5) + 2\cdot 9\cdot\pi\cdot\sin(2\pi x)}{-2 + (2\pi)^2\cdot 9\cdot\cos(2\pi x)}$$

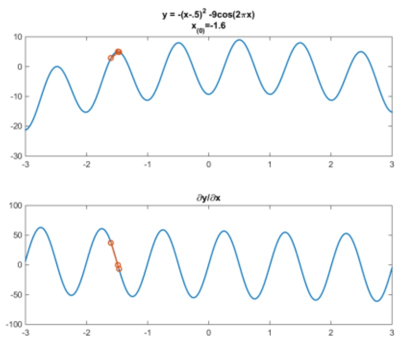


## Matlab snippet

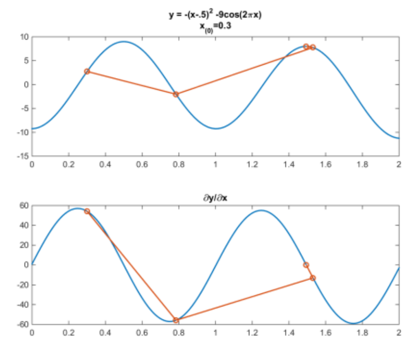
```
xGuess(1) = 0.4;
yGuess(1) = y(xGuess(1));

for it = 2:nIter
    xGuess(it) = xGuess(it-1) - 
dy(xGuess(it-1))/d2y(xGuess(it-1));
    yGuess(it) = y(xGuess(it));
end
```












## Step-size adjustment

$$\alpha^{(j)} = 1$$

$$\beta^{(j+1)} = \beta^{(j)} - \alpha_{(j)} \frac{Q'\left(\beta^{(j)}\right)}{Q''\left(\beta^{(j)}\right)}$$

If $Q^{(j+1)} < Q^{(j)}, \alpha^{(j)} = \alpha^{(j)}/2$

(If $\alpha$ gets too small, punt.)



## Quasi-Newton in matrix form

$$\beta^{(j+1)} = \beta^{(j)} - \left[Q''\left(\beta^{(j)}\right)\right]^{-1} Q'\left(\beta^{(j)}\right)$$

$$\beta^{(j+1)} = \beta^{(j)} - D_{(j)}^{-1} Q'\left(\beta^{(j)}\right)$$

- $D \approx Q''$, but guaranteed negative definite

## Berndt-Hall-Hall-Hausman (BHHH)

$$I \equiv \mathrm{E}\left(\left(\frac{\partial}{d\theta}\log f\,(x|\theta)\right)^2\right) = -\,\mathrm{E}\left(\frac{\partial^2}{\partial\theta^2} log\, f(x|\theta)\right)$$

$$D = -\sum_{i=1}^{n}\left(\frac{\partial}{d\theta}\log f\,(x_i|\theta)\right)\left(\frac{\partial}{d\theta}\log f\,(x_i|\theta)\right)'$$



## Gauss-Newton Regression

$$y_i = g(x_i,\beta) + \varepsilon_i$$

$$Q(\beta) = -\sum (y_i - g(x_i,\beta))^2$$

$$e_i(\beta) \approx e_i\left(\beta^{(j)}\right) - \frac{\partial g\left(x_i,\beta^{(j)}\right)}{\partial\beta}\left(\beta - \beta^{(j)}\right)$$

$$e_i\left(\beta^{(j)}\right) = \frac{\partial g\left(x_i,\beta^{(j)}\right)}{\partial\beta}\Delta + v$$

$$\beta^{(j+1)} = \beta^{(j)} + \widehat{\Delta}$$



## Stopping rules

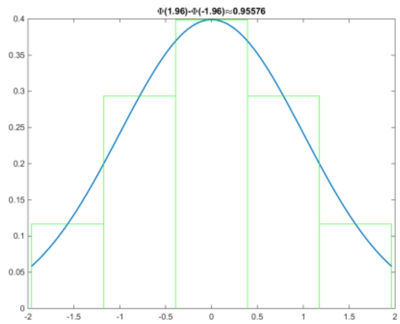
- $Q\left(\beta^{(j+1)}\right) - Q\left(\beta^{(j)}\right)$ is small
- $Q\left(\beta^{(j)}\right)'$ is small
- $\beta^{(j+1)} - \beta^{(j)}$ is small

Always: Stop if too many iterations and computer is overheating.



## Matlab integral

```
>> integral(@normpdf,-1.96,1.96)
ans =
   0.9500
```



## Numerical quadrature

$$F = \int_l^u f(x)dx$$

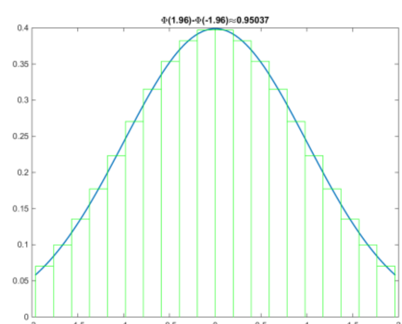
$$A_i \approx (a_i - a_{i-1}) f\left(\frac{a_i + a_{i-1}}{2}\right)$$

$$F \approx \sum_{i=1}^{n} A_i$$

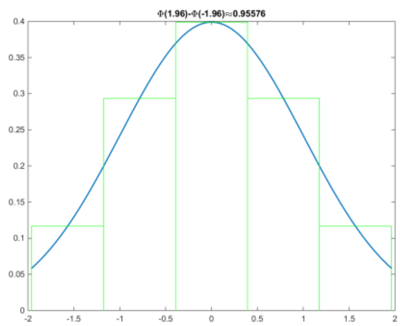
 

```
nRectangles = 5;
endPoints =
linspace(l,u,nRectangles+1);

width = endPoints(2)-endPoints(1);
height =
normpdf((endPoints(2:end)+endPoints(1
:end-1))/2);
F = sum(width*height);
```





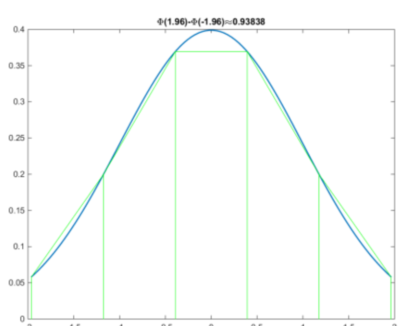

 

## Trapezoidal rule

Using the formula for the area of a trapezoid which is the rectangle (base times height) plus the triangle (1/2 base times height.)

$$A_i \approx (a_i - a_{i-1}) f(a_i)$$
$$+ \frac{1}{2}(a_i - a_{i-1})\big(f(a_{i-1}) - f(a_i)\big)$$
$$= \frac{a_i - a_{i-1}}{2}\big(f(a_{i-1}) + f(a_i)\big)$$



## *Simpson's rule*

Parabola through $f(a_{i-1})$, $f(a_i)$, and the midpoint $f\left(\frac{a_{i-1}+a_i}{2}\right)$.

$$A_i \approx \frac{a_i - a_{i-1}}{6}\left(f(a_{i-1}) + 4f\left(\frac{a_{i-1}+a_i}{2}\right)\right.$$
$$\left. + f(a_i)\right)$$

## Matlab integral

integral(fun,a,b)

uses an adaptive quadrature to integrate the function “fun” between the points a and b.

integral2(fun,xmin,xmax,ymin,ymax)

does the same thing in two dimensions



## Integrate multiple dimensions

$$\iint_{l_y,l_x}^{u_y,u_x} f(x,y)dxdy$$

$$\int_{l_y}^{u_y}\left[\int_{l_x}^{u_x} f(x,y)dx\right]dy$$

$$A(y) \approx \int_{l_x}^{u_x} f(x,y)dx$$

$$A \approx \int_{l_y}^{u_y} A(y)dy$$



## Question for class

Compute an estimate of $\pi$ by the following technique of Monte Carlo integration. Draw random $\{x, y\}$ pairs uniform between 0 and 1. Then decide if the point is inside or outside the unit circle. Since the fraction of points inside the unit circle corresponds to the area of a quarter unit circle, you can estimate $\pi$ by four times the fraction inside the circle. Do this for a various numbers of random draws and plot the estimates of $\pi$ against the number of draws.



## One example of Monte Carlo Integration

$$\int_{l}^{u} f(x)dx \approx \frac{u-l}{n}\sum f(\tilde{x})$$

where $\tilde{x}$ is drawn randomly.

```
reset(RandStream.getGlobalStream);
l = norminv(.025);
u = norminv(.975);

nPoints = 10000;
x = l + (u-l)*rand(nPoints,1);
width = (u-l)./(1:nPoints)';
estimate =
width.*cumsum(normpdf(x));
```