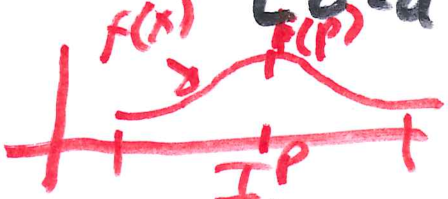
Numerical Optimization

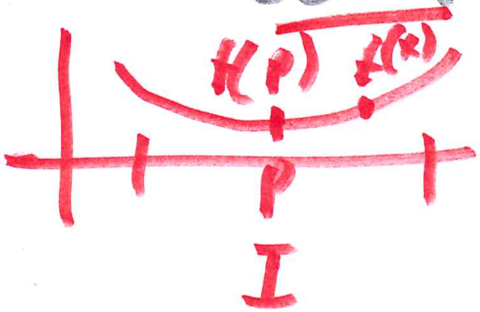
Minimization of an equation.

- Local maximum value at $x = P$



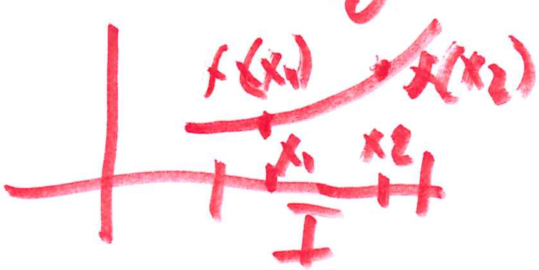
  if $f(x) \le f(P)$ for all $x \in I$

- Local minimum value at $x = P$



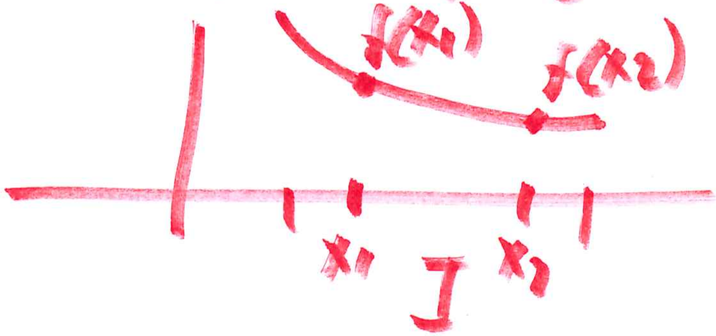
  if $f(x) \ge f(Y)$ for all $x \in I$

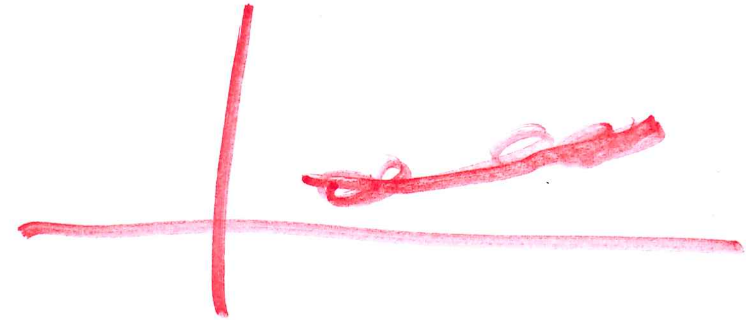
- A function is increasing on I
  if $x_1 < x_2$ and $f(x_1) < f(x_2)$ for all $x_1, x_2 \in I$



205

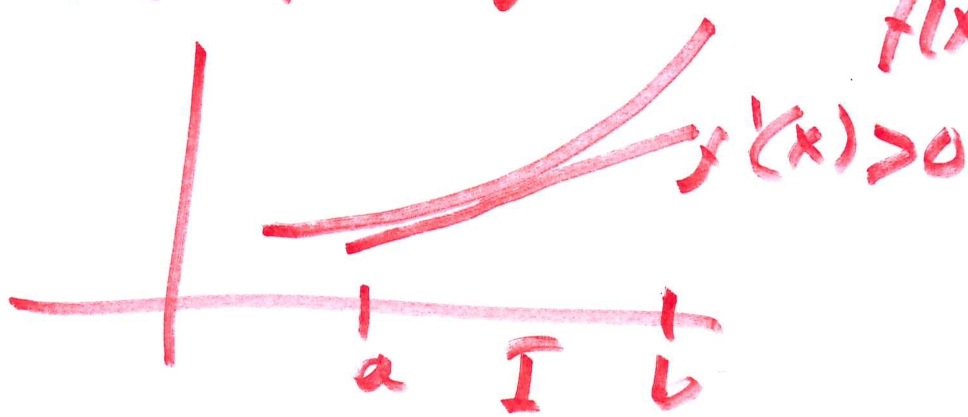
- A function is decreasing on I if $x_1 < x_2$ and $f(x_1) > f(x_2)$ for all $x_1, x_2 \in I$



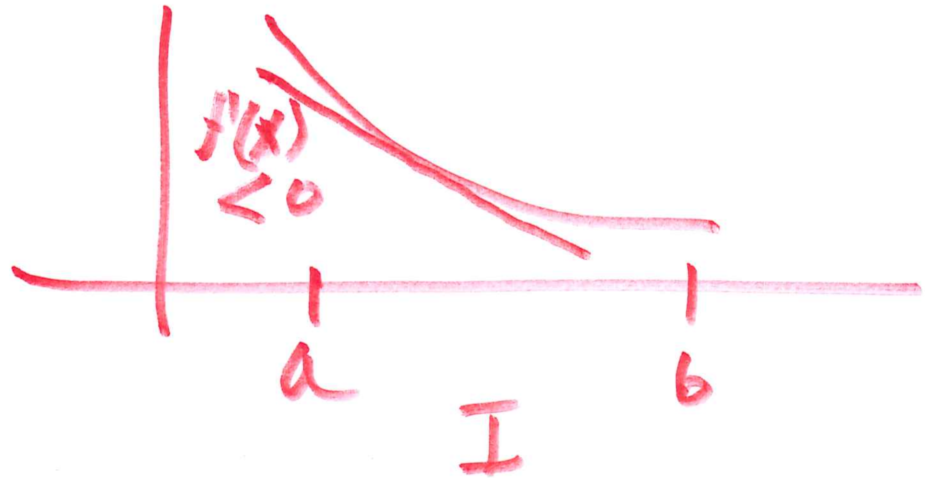


- Suppose $f(x)$ is continuous on $I = [a, b]$ and $f(x)$ is differentiable.

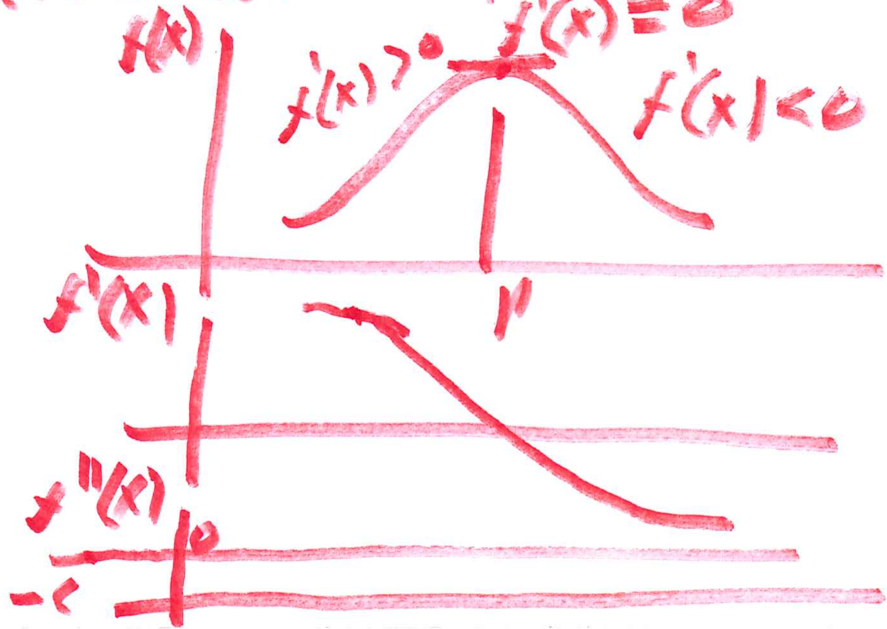
if $f'(x) > 0$ for all $x \in (a, b)$ then $f(x)$ is increasing on I



If $f'(x) < 0$ for all $x = (a, b)$
then $f(x)$ is decreasing on I



If there is maximum or minimum
value at $x = p$ then $f'(p) = 0$

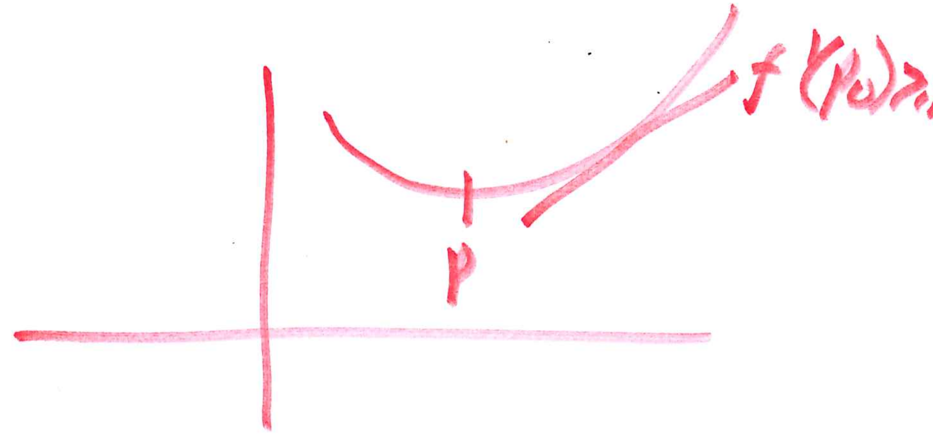


If $f''(p) > 0$ then
$f(p)$ is a local
minimum

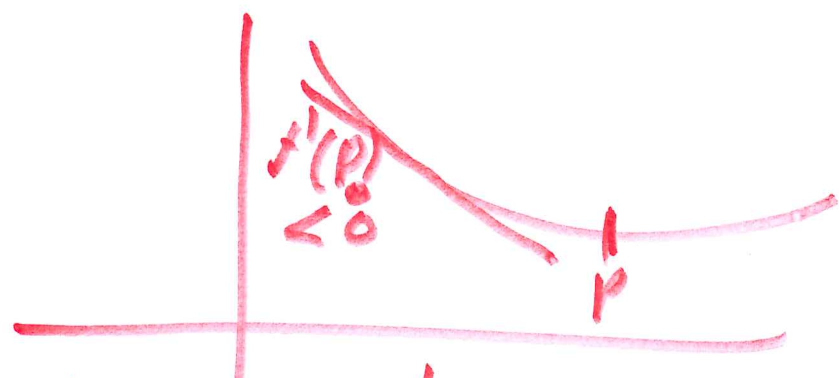
If $f''(p) < 0$ then
$f(p)$ is a local maximum.

Assume we want to minimize $f(x)$ and it has a unique minimum at p, $a < p < b$. If we start the search at p_0

If $f'(p_0) > 0$ the p is at the left of p_0



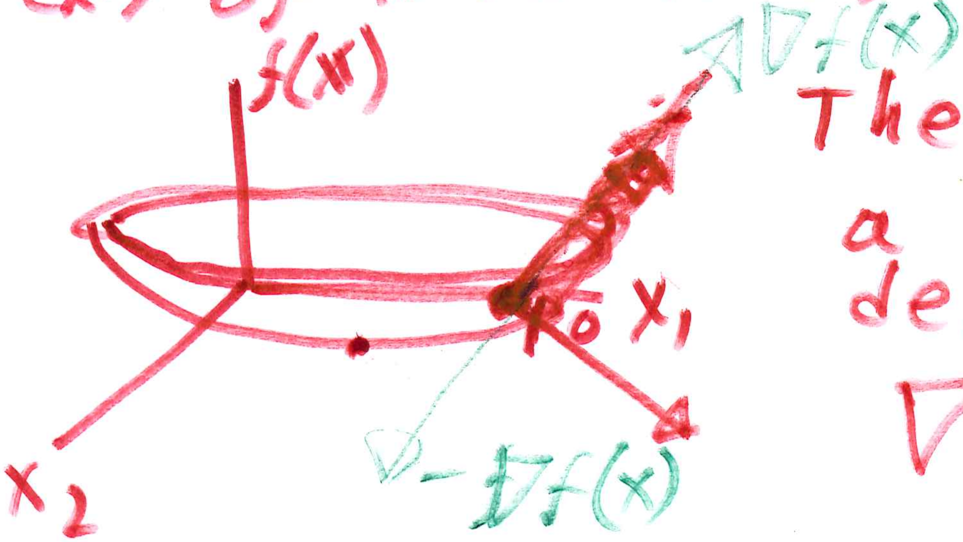
If $f'(p_0) < 0$ then p is at the right of p_0



Any method used to solve non-linear equations $f(x) = 0$ can be used to find a minimum if we use it to solve $\underline{f'(x) = 0}$

Steepest Descent Method or Gradient method to obtain minimal points

Assume that we want to minimize $f(\mathbf{x})$ of N variables where $\mathbf{x} = (x_1, x_2 \ldots x_N)$



The gradient $\nabla f(\mathbf{x})$ is a vector function defined as

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \ldots \frac{\partial f}{\partial x_N} \right)$$

From the concept of Gradient we know that the gradient vector $\nabla f(x)$ points in the direction of maximum change or greatest increase of $f(x)$. Then $-\nabla f(x)$ points in the direction of greatest decreas.

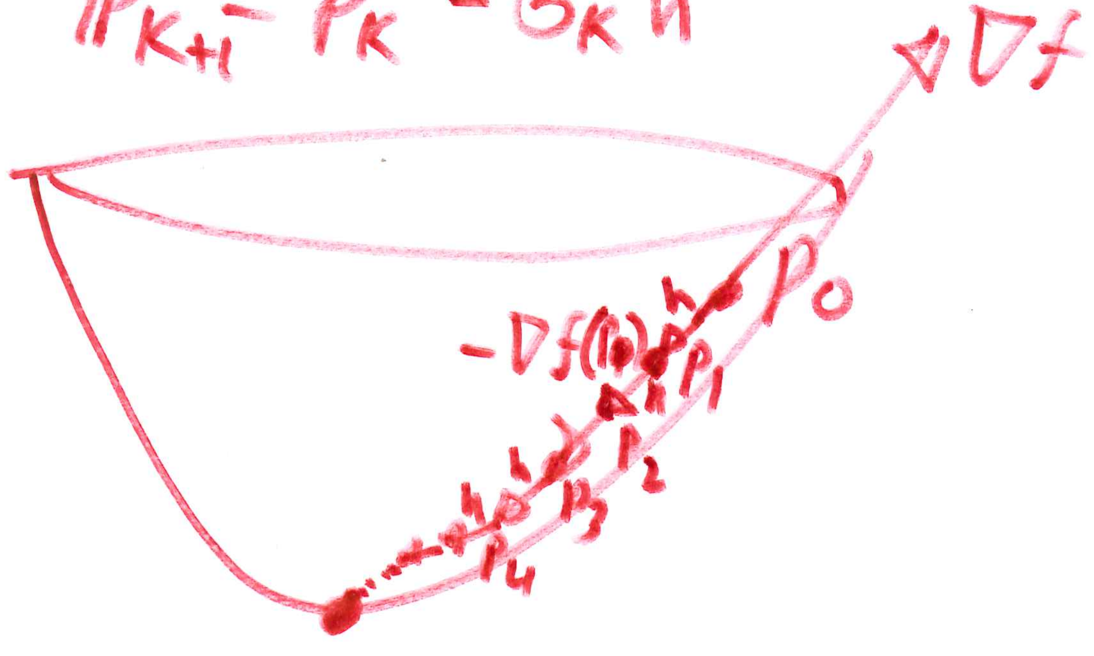
Steepest Descent (Gradient Method)

Start at point P_0 and move along the line in the direction $-G_0$ where $-G_0 = -\nabla f(P_0)$

In its simplest form

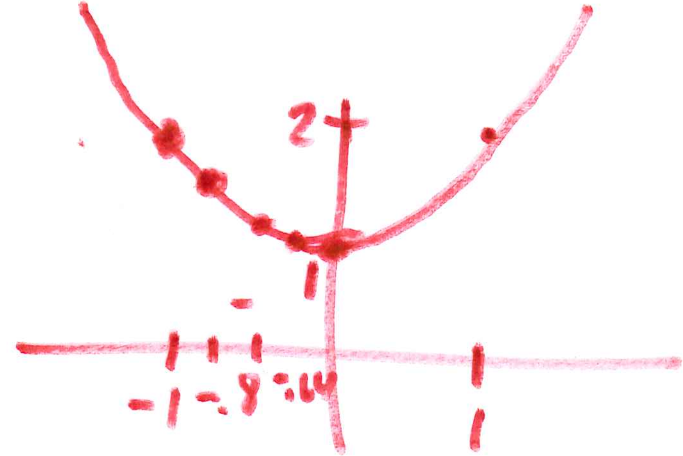
$$P_1 = P_0 - G_0 h$$ where h is a small increment.

$$P_{k+1} = P_k - G_k h$$



Example

$$y = x^2 + 1$$



$$y' = 2x$$

$$G = Dy = \frac{dy}{dx} = 2x \qquad \text{let } h = .1$$

$$x_{k+1} = x_k - hG_k$$

let $x_0 = -1$

	X	Y	G
x_0	0 $\quad x_0 - 1$	$(-1)^2 + 1 = 2$	$G_0 = 2(-1) = -2$
	1 $\quad x_1 = -1 - (.1)(-2) = -.8$		$G_1 = 2(-.8) = -1.6$
	$x_2 = -.8 - .1(-1.6) = -.64$		$G_2 = 2(-.64) = -1.28$
	$x_3 = -.64 - .1(-1.28) = -.512$		$G_3 = 2(-.512) = -1.024$
	$x_4 = -.512 - .1(-1.024) = -.4096$		

$$\vdots$$

$$x_\infty = 0$$

- Steepest descent will give you a <u>local</u> minimum not a <u>global one</u>.

- We will see later other algorithms that can give you a <u>global minimum</u>

Numerical Solution of Differential Equations

A differential equation is an equation to solve that contains derivatives.

Example:

$$\frac{dy}{dt} = K_1 y$$

Solution:

$$\frac{dy}{y} = K_1 dt \rightarrow \int \frac{dy}{y} = \int K_1 dt$$

$$\log y = K_1 t + K_2$$

$$y = e^{K_1 t + K_2}$$

$$\underline{\underline{y = K_3 e^{K_1 t}}}$$

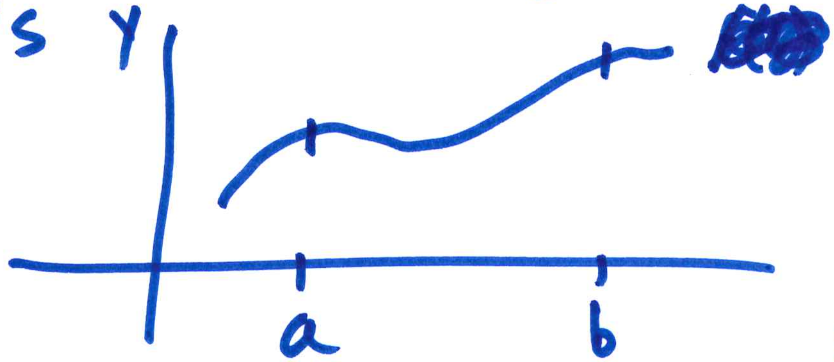
Some differential equations do not have an anatical solution so they have to be approximated with ~~diff~~ numerical methods.

Euler Method

Let $[a,b]$ be the interval over which we want to fin ~~the~~ the solution $y' = f(t,y)$ with $y(a) = y_0$.

We will find a set of points $(t_0, y_0), (t_1, y_1) \ldots (t_k, y_k)$ that are used to approximate $y(t) \approx y(t_k)$.

First we divide the interval $[a,b]$ into M equal subintervals



$h = \frac{b-a}{M}$ h is called step size

Also we make
$t = a + kh$
$k = 0, 1, \ldots M$

Example

$$y' = t^2 - y \qquad y(0) = 1 \qquad h = .2$$

$t_0 = 0$

$t_1 = .2$ $\quad y_1 = y_0 + h(t_0^2 - y_0) = 1 + .2(0^2 - 1) = .8$

$t_2 = .4$ $\quad y_2 = y_1 + h(t_1^2 - y_1) = .8 + .2(.2^2 - .8) = .648$

$t_3 = .6$ $\quad y_3 = y_2 + h(t_2^2 - y_2) = .648 + .2(.4^2 - .648) = .5504$

$t_4 = .8$ $\quad y_4 = y_3 + h(t_3^2 - y_3) = .5504 + .2(.6^2 - .5504)$

$= .5123$

Analytical Solution is

$$y(t) = -e^{-t} + t^2 - 2t + 2$$

$$y(.4) = -e^{-.4} + (.4)^2 - 2(.4) + 2$$

$$= .6847$$

$$y(.8) = -e^{-.8} + (.8)^2 - 2(.8) + 2$$

$$= .5907$$

We want to solve

$$y' = f(t, k) \text{ over } [t_0 \ldots t_M]$$

with $y(t_0) = y_0$

Using Taylor expansion to approximate $y(t)$ around t_0. error.

$$y(t) = y(t_0) + y'(t_0)(t - t_0) + \frac{y''(c_1)(t-t_0)^2}{2!}$$

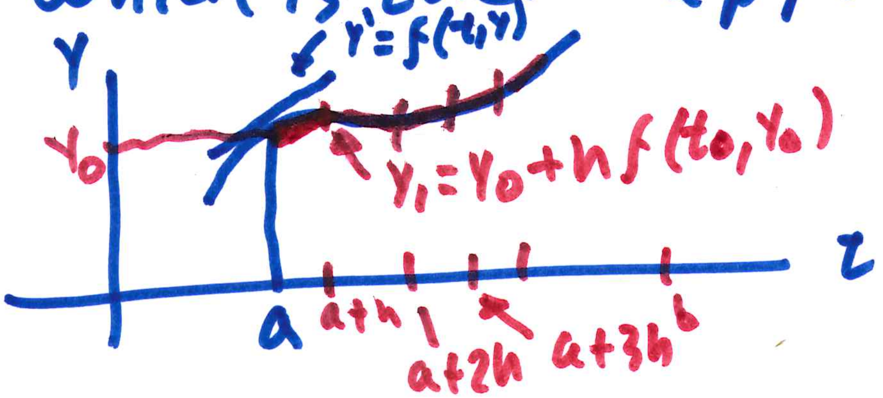
Now to obtain $t = t_1$ h

$$y(t_1) = y(t_0) + f(t_0, y_0)(t_1 - t_0) + \frac{y''(c_1)(t-t_0)^2}{2!}$$

If the step size is small enough, we can neglect the second order error

$$y_1 = y_0 + h f(t_0, y_0)$$

which is Euler's approximation.



Heun's Method

We want to solve

$$y'(t) = f(t, y(t)) \text{ over } [a, b] \text{ with } y(t_0) = y_0$$

We can use the fundamental theorem of calculus and integrate $y'(t)$ over $[t_0, t_1]$

$$\int_{t_0}^{t_1} f(t, y(t))\,dt = \int_{t_0}^{t_1} y'(t)\,dt = y(t_1) - y(t_0)$$

so we have $y(t_1) = y(t_0) + \int_{t_0}^{t_1} y'(t)\,dt$

Now we can use any numerical integration method to approximate the integral.

Using trapezoidal rule

$$y(t_1) = y(t_0) + \frac{h}{2}\left[f(t_0, y(t_0)) + f(t_1, y(t_1)) \right]$$

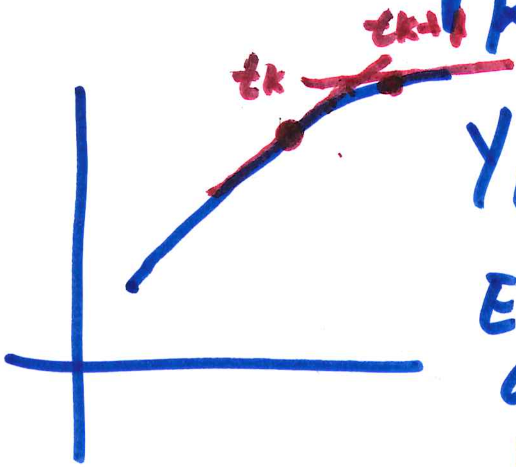
Observe that we still need to know $f(t_1, y(t_1))$ in the right side. For that we use Euler's approximation

$$y_1 = h f(t_0, y(t_0)) + y_0$$

So we get

$$y(t_1) = y(t_0) + \frac{h}{2}\left[f(t_0, y(t_0)) + f(t_1, y_0 + hf(t_0, y(t_0)))\right]$$

In general



$$P_{k+1} = y_k + h\, f(t_k, y_k)$$

$$y_{k+1} = y_k + \frac{h}{2}\left[f(t_k, y_k) + f(t_{k+1}, P_{k+1})\right]$$

Euler approximation is used as a predictor and the integral is used as a correction.

Example

$f(y,t) = y' = t^2 - y \qquad y(0) = 1 \quad h = .2$

$k=0 \qquad y_0 = 1 \qquad t_0 = 0$

$k=1$

$t_1 = .2$

$$P_1 = y_0 + h f(t_0, y_0) = y_0 + (.2)(t_0^2 - y_0) = 1 + .2(0^2 - 1) = .8$$

$$y_1 = y_0 + \frac{h}{2}\Big(\underbrace{f(t_0, y_0)}_{t_0^2 - y_0} + \underbrace{f(t_1, P_1)}_{t_1^2 - P_1}\Big)$$

$$= 1 + \frac{.2}{2}\big((0^2 - 1) + (.2^2 - .8)\big)$$

$$= .8240$$

$k=2$

$t_2 = .4$

$$P_2 = y_1 + h f(t_1, y_1) = .8240 + .2(t_1^2 - y_1)$$

$$= .8240 + .2(.2^2 - .8240)$$

$$= .6672$$

$$y_2 = y_1 + \frac{h}{2}\Big(\underbrace{f(t_1, y_1)}_{t_1^2 - y_1} + \underbrace{f(t_2, P_2)}_{t_2^2 - P_2}\Big)$$

$$=$$

$$y_2 = .8240 + \frac{.2}{2}\left((.2^2 - .8240) + (.4^2 - .6592)\right)$$ 218

$$= .6949$$

Exact $y_2 = y(.4) = .6897$

Euler $y_2 = y(.4) = .648$

$k=3$

$t_3 = .6$

$$p_3 = y_2 + h\,\underset{t_2^2 - y_2}{f(t_2, y_2)} = .6949 + .2(.4^2 - .6949)$$
$$= .5879$$

$$y_3 = y_2 + \frac{h}{2}\left(\underset{t_2^2 - y_2}{f(t_2, y_2)} + \underset{t_3^2 - p_3}{f(t_3, p_3)}\right)$$

$$= .6949 + \frac{.2}{2}\left[(.4^2 - .6949) + (.6^2 - .5879)\right]$$

$$= .6186$$

$k=4$

$t_4 = .8$

$$p_4 = y_3 + h\,\underset{t_3^2 - y_3}{f(t_3, y_3)} = .6186 + .2[.6^2 - .6186]$$
$$= .5669$$

$$y_4 = y_3 + \frac{h}{2}\left(\underset{t_3^2 - y_3}{f(t_3, y_3)} + \underset{t_4^2 - p_4}{f(t_4, p_4)}\right)$$

$y_4 =$

219

$$y_4 = .6186 + \frac{.2}{2}\left((.6^2 - .6186) + (.8^2 - .5669)\right)$$

$$= .6001$$

exact: $.5907$

Euler: $.5123$

Taylor Series Method to solve differential equations

Using Taylor approximation we have

$$y(t_k+h) = y(t_k) + h\,y'(t_k) + \frac{h^2 y''(t_k)}{2!} + \dots \frac{h^N y^N(t_k)}{N!} + O(h^{N+1})$$

$$\Downarrow$$

$$y_{k+1}$$

we want to solve

$$y' = f(y,t)$$

From Taylor expansion

$$y_{k+1} = y_k + h\,y_k' + \frac{h^2 y_k''}{2!} + \dots \frac{h^N y_k^N}{N!} + O(h^{N+1})$$

However we need y_k', y_k'', y_k''' ... etc

we can obtain them from $y' = f(y,t)$

Example:

Solve $y' = t^2 - y \qquad y(0) = 1 \qquad h = .2$

Use $N = 3$

$$y_{k+1} = y_k + h\, y_k' + \frac{h^2 y_k''}{2!} + \frac{h^3 y_k'''}{3!} + \underbrace{O(h^4)}_{\text{error.}}$$

$$y' = t^2 - y \rightarrow y_k' = t_k^2 - y_k$$

$$y'' = 2t - y' \rightarrow y_k'' = 2t_k - y_k'$$

$$y''' = 2 - y'' \rightarrow y_k''' = 2 - y_k''$$

$h = .2$

$O(h^4) = O(.2^4)$

$O(2^{-1})^4 = O(2^{-4})$

$= O(.0001)$

$k=0 \qquad t = 0 \qquad y_0 = 1 \qquad y_0' = (0)^2 - 1 = -1$

$$y_0'' = 2(0) - (-1) = 1$$

$$y_0''' = 2 - (1) = 1$$

$k=1 \qquad t = 0 + .2 = .2$

$$y_1 = 1 + .2(-1) + \frac{.2^2(1)}{2} + \frac{(.2)^3(1)}{6} = .821333$$

$$y_1' = .2^2 - .821333 = -.781333$$

$$y_1'' = 2(.2) - (-.781333) = 1.181333$$

$$y_1''' = 2 - .821333 = .818667$$

$k=2 \quad t=.2+.2=.4$

$$y_2 = .821333 + .2(-.781333) + \frac{.2^2(1.181333)}{2} + \frac{.2^3(.818667)}{6}$$

$$= .689785$$

$y_2' = (.4)^2 - .689785 = -.529785$

$y_2'' = 2(.4) - (-.529785) = 1.329785$

$y_2''' = 2 - (1.329785) = .670215$

Exact: .6897
Euler: .648
Heun's: .6949

$k=3 \quad t=.6$

$$y_3 = .689785 + .2(-.529785) + \frac{.2^2(1.329785)}{2} + \frac{.2^3(.670215)}{6}$$

$$= .611317$$

$y_3' = .6^2 - .611317 = -.251317$

$y_3'' = 2(.6) - (-.251317) = 1.451317$

$y_3''' = 2 - (1.451317) = .548683$

$k=4 \quad t=.8$

$$y_4 = .611317 + .2(-.251317) + \frac{.2^2(1.451317)}{2} + \frac{.2^3(.548683)}{6}$$

$$= .590812$$

Heun's: .6001
Euler: .5123
Exact: .5907

Runge-Kutta Method

- The derivatives of the Taylor Method can be computed numerically. This is done in Runga-Kutta of order $N=4$ that does not require analytical computation of derivatives:

$$y_{k+1} = y_k + \frac{h(f_1 + 2f_2 + 2f_3 + f_4)}{6}$$

where
$$f_1 = f(t_k, y_k)$$
$$f_2 = f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2} f_1\right)$$
$$f_3 = f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2} f_2\right)$$
$$f_4 = f(t_k + h, y_k + h f_3)$$

(see proof in advanced text of numerical analysis).

Example: $y' = t^2 - y \quad y(0) = 1 \quad h = .2$

$t_0 = 0 \quad y_0 = 1$

$f_1 = 0^2 - 1 = -1$

$f_2 = f\left(0 + \frac{.2}{2}, 1 + \frac{.2}{2}(-1)\right) = f(.1, .9) = .1^2 - .9 = -.89$

$f_3 = f\left(0 + \frac{.2}{2}, 1 + \frac{.2}{2}(-.89)\right) = f(.1, .911) = .1^2 - .911 = -.901$

$f_4 = f(0 + .2, 1 + .2(-.901)) = f(.2, .8198) = .2^2 - .8198 = -.7798$

$t_1 = .2$

$$y_1 = 1 + .2\,\frac{(-1 + 2(-.89) + 2(-.901) - .7798)}{6}$$

$= .821273$

$f_1 = .2^2 - (.821273) = -.781273$

$f_2 = f\left(.2 + \frac{.2}{2}, .821273 + \frac{.2}{2}(-.781273)\right) = f(.3, .743146)$

$= .3^2 - .743146 = -.653146$

$$f_3 = f\left(.2 + \frac{.2}{2},\ .821273 + \frac{.2}{2}(-.653146)\right)$$

$$= f(.3,\ .755958) = .3^2 - .755958$$

$$= -.665958$$

$$f_4 = f(.2 + .2,\ .821273 + .2(-.665958))$$

$$= f(.4,\ .688081) = .4^2 - .688081 = -.528081$$

$$y_2 = .821273 + \frac{.2\left(-.781273 + 2(-.653146) + 2(-.665958) + (-.528081)\right)}{6}$$

$$= .689688$$

Exact: .6897

Euler: .648

Heun's .6949

~~Taylor: .6921~~

Taylor: .689785

Predictor - Corrector Methods

Systems of Differential Equations

Assume we have the equations

$$\frac{dx}{dt} = f(t, x, y) \qquad \text{with} \quad x(t_0) = x_0$$

$$y(t_0) = y_0$$

and

$$\frac{dy}{dt} = g(t, x, x)$$

The solution are functions $x(t)$ and $y(t)$ that when derivated they transform into $f(t, x, y)$ and $g(t, x, y)$

Example:

$$x' = x + 2y \qquad x(0) = 6$$

$$y' = 3x + 2y \qquad y(0) = 4$$

Solution: $x(t) = 4e^{4t} + 2e^{-t}$

$y(t) = 6e^{4t} - 2e^{-t}$

Euler method for a system of Differential Equations

We can substitute

$$dx = x_{k+1} - x_k$$
$$dy = y_{k+1} - y_k$$
$$dt = t_{k+1} - t_k$$

in

$$\frac{dx}{dt} = f(t, x, y)$$
$$\frac{dy}{dt} = g(t, x, y)$$

$$dx = f(t, x, y)\,dt$$

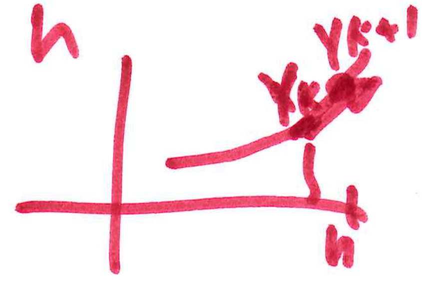
$$\Rightarrow x_{k+1} - x_k = f(t_k, x_k, y_k)\underbrace{(t_{k+1} - t_k)}_{h}$$

$$dy = g(t, x, y)\,dt$$

$$\Rightarrow y_{k+1} - y_k = g(t_k, x_k, y_k)\underbrace{(t_{k+1} - t_k)}_{h}$$

In general — Euler method

$$t_{k+1} = t_k + h$$
$$x_{k+1} = x_k + h f(t_k, x_k, y_k)$$
$$y_{k+1} = y_k + h g(t_k, x_k, y_k)$$



If this Euler method has little accuracy, we can improve it by using a Taylor expansion that uses more terms:

Example:

$$t_{k+1} = t_k + h$$

$$x_{k+1} = x_k + h f(t_k, x_k, y_k) + \frac{h^2}{2!} \frac{d f(t_k, x_k, y_k)}{dt}$$

$$y_{k+1} = y_k + h g(t_k, x_k, y_k) + \frac{h^2}{2!} \frac{d g(t_k, x_k, y_k)}{dt}$$

However it is more common to use the Runge-Kutta method for systems of differential equations.

$$x_{k+1} = x_k + \frac{h}{6}(f_1 + 2f_2 + 2f_3 + f_4)$$

$$y_{k+1} = y_k + \frac{h}{6}(g_1 + 2g_2 + 2g_3 + g_4)$$

where

$$f_1 = f(t_k, x_k, y_k)$$

$$f_2 = f(t_k + \frac{h}{2}, x_k + \frac{h}{2} f_1, y_k + \frac{h}{2} g_1)$$

$$f_3 = f(t_k + \frac{h}{2}, x_k + \frac{h}{2} f_2, y_k + \frac{h}{2} g_2)$$

$$f_4 = f(t_k + h, x_k + h f_3, y_k + h g_3)$$

$$g_1 = g(t_k, x_k, y_k)$$

$$g_2 = g(t_k + \frac{h}{2}, x_k + \frac{h}{2} f_1, y_k + \frac{h}{2} g_1)$$

$$g_3 = g(t_k + \frac{h}{2}, x_k + \frac{h}{2} f_2, y_k + \frac{h}{2} g_2)$$

$$g_4 = g(t_k + h, x_k + h f_3, y_k + h g_3)$$

Higher Order Differential Equations

Higher order differential equations involve higher derivatives: $x''(t)$, $x'''(t)$

For example:

$$m\,x''(t) + c\,x'(t) + k\,x(t) = g(t)$$

To solve this higher order differential equation numerically we transform it into a system of first degree differential equations.

For example we can solve the second order differential equation as:

$$x''(t) = f(t, x(t), x'(t))$$

Also we have that we can build a function

$$x'(t) = y(t) \quad \text{then} \quad x''(t) = y'(t)$$

Then the second order differential equation will become

$$\frac{dx}{dt} = y$$

$$\frac{dy}{dt} = f(t, x(t), y'(t)$$

that is a system of diferential equations

Example:

$$4x''(t) + 3x'(t) + 5x(t) = 2$$

with
$x(0) = 1$
$x'(0) = 3$

$$x'' = \frac{2 - 3x' - 5x}{4}$$

Let $y = x'$

$$x'' = \frac{2 - 3x' - 5x}{4} = y'$$

So we have the system of differential eq.

$$x' = y$$

$$y' = \frac{2 - 3x' - 5x}{4}$$

with $x(0) = 1$ $y(0) = 3$

Simulated Aneuling

The methods of numerical optimization we covered in class only find local minimums (or maximum).

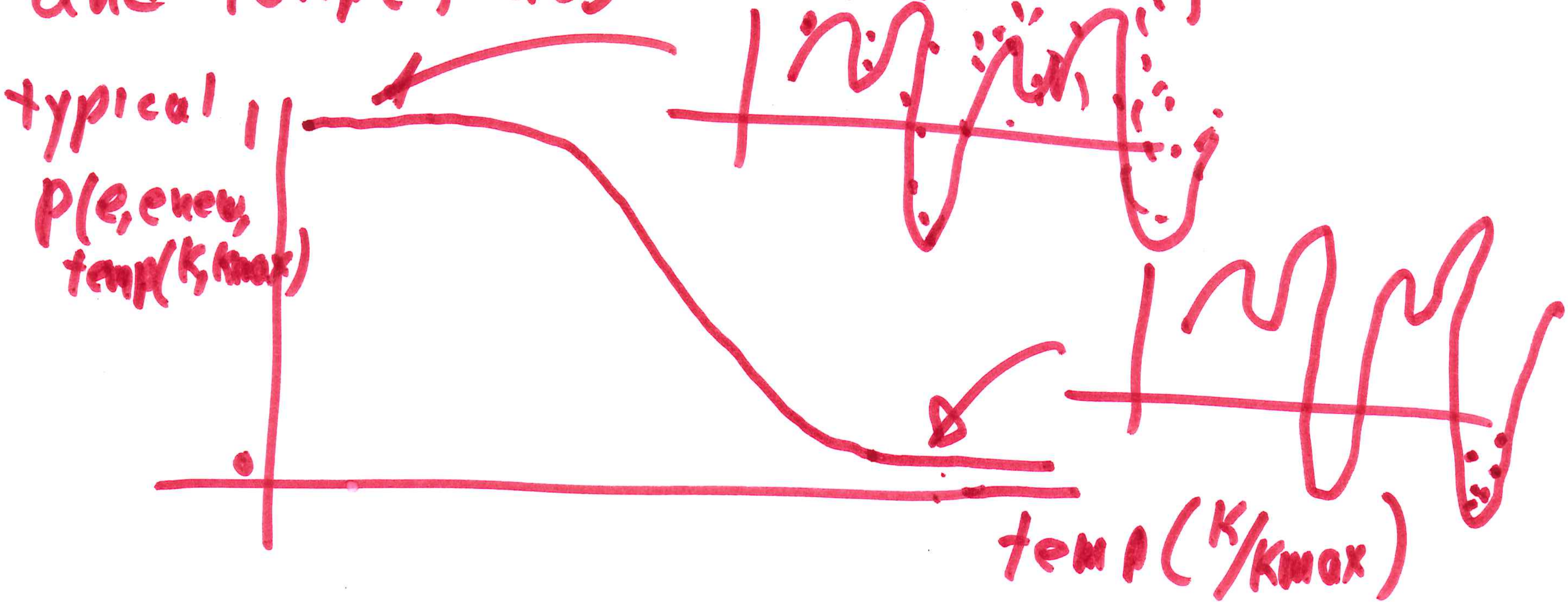
- Simulated anealing is a method used to find the <u>global</u> minimum and maximum
- The name comes from "anealing" in metalurgy that involves heating and then have a slow controlled cooling of material to increase the size of the cristals and reduce deffects.
- The heat causes atoms to leave initial positions (local minimum of the molecules energy) and wonder around states of higher energy.
- The slow cooling gives more chance to atoms to arrive to a configuration of lower energy than the initial position.

We can simulate this process in the computer as follows:

Algorithm Simulated Anealing

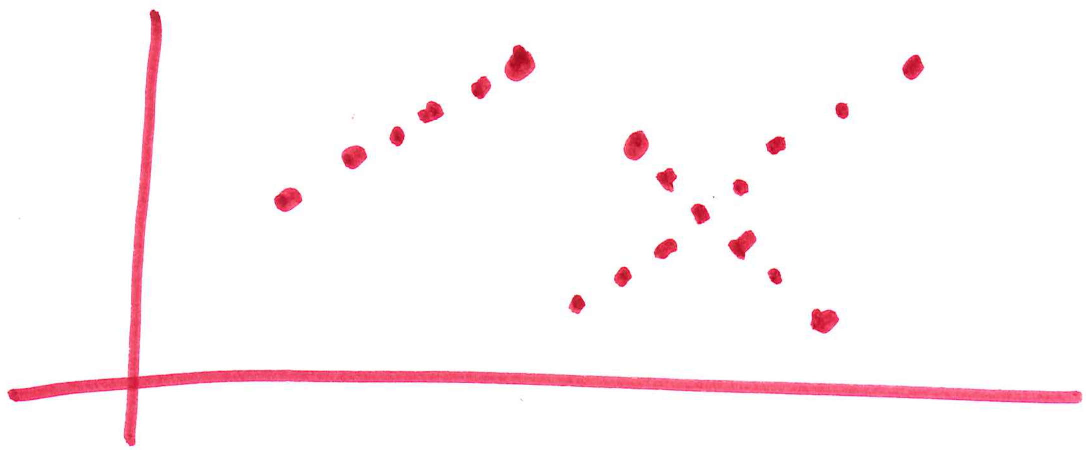
```
s ← s0, e = E(s)          // Initial state and energy
s_best ← s; ebest = e     // Initial "best" solution
K = 0
while K < Kmax and e > max    // while theres time left and solution not good enough.
   snew ← neighbor(s);        // pick neighbor state
   enew ← E(snew)             // compute new energy
   if (P(e, enew, temp(K/Kmax)) ≥ random(0,1)   // should we
      s ← snew; e ← enew                        // move state.
   end
   if enew < ebest then        // Is this new best
      s_best ← snew            // save best state.
      ebest ← enew
   end
end K = K+1
```

The probability function
p(e, enew, temp(k, kmax))
and temp() define the "cooling schedule"!



Pattern Recognition

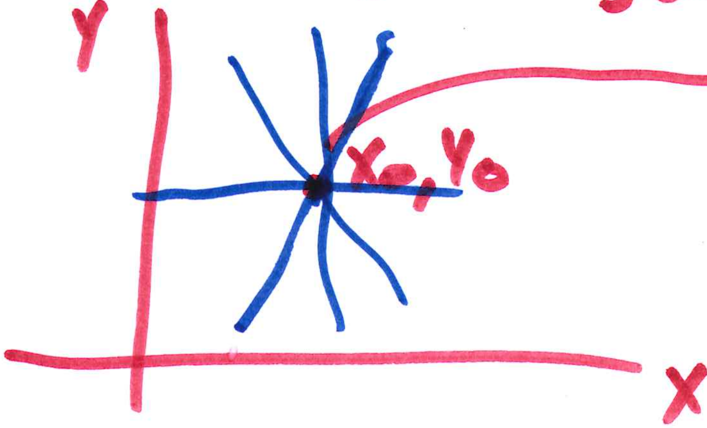
Problem:
Given a set of points, find the most likely line(s) that pass through these points.

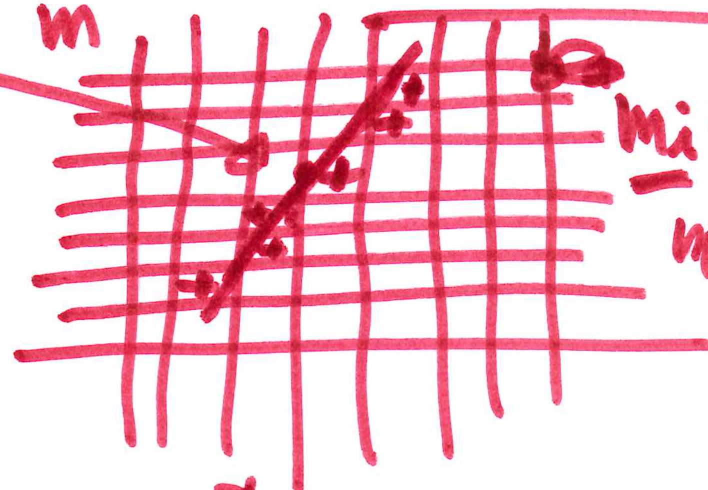


Hough transform

It is a method that builds a matrix of counters where count represents the number of "votes" that a sample provides.

A point x_0, y_0 can be part of a family of lines $y_0 = m_i x_0 + b_i$



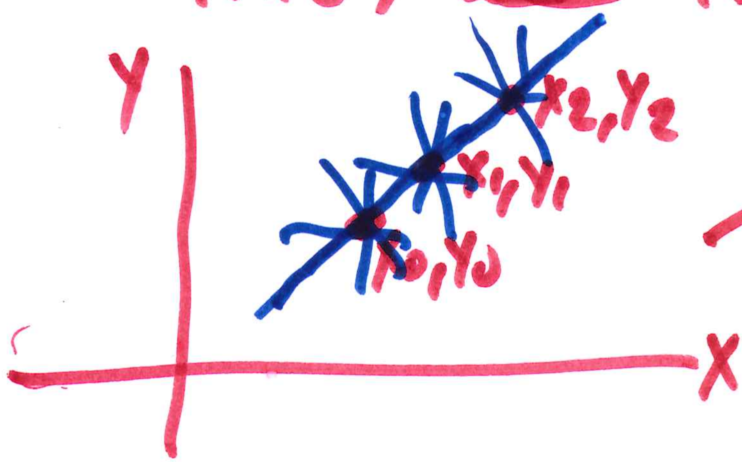


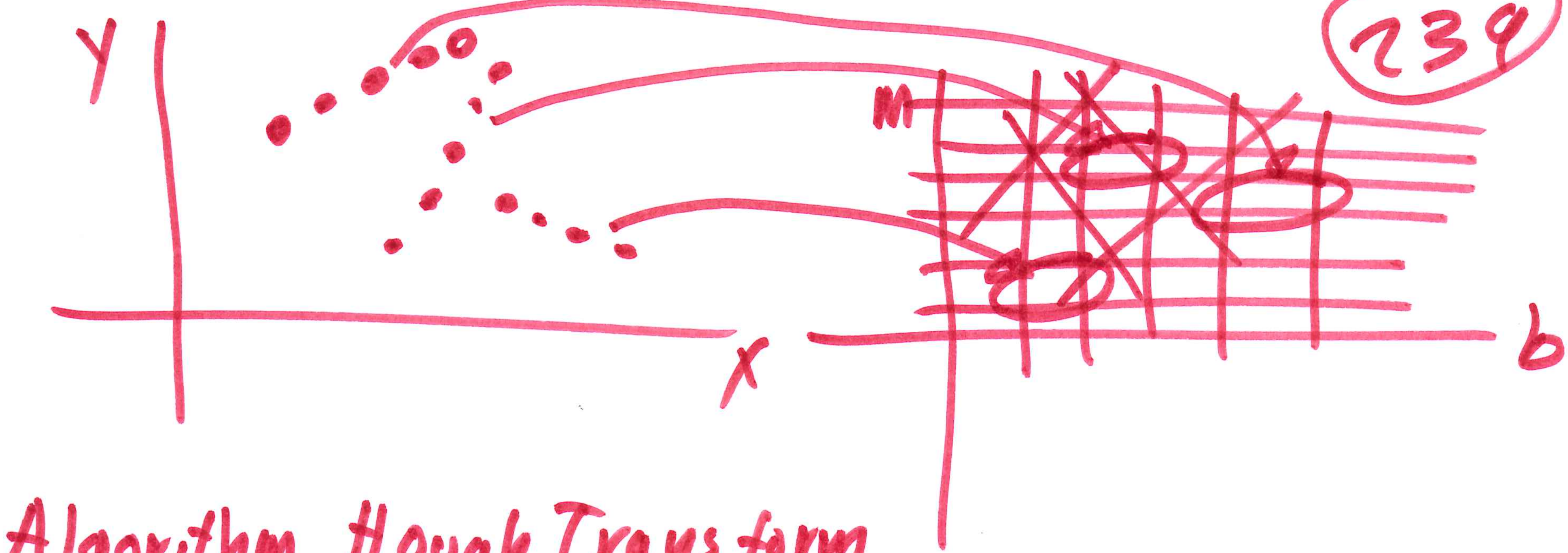
$$m_i = \frac{y_0 - b_i}{x_0}$$

$$m_i = \frac{y_0}{x_0} - \frac{b_i}{x_0}$$

matrix of counter

A point x_0, y_0 will increment all the counters (b_i, m_i) for the lines that x_0, y_0 belongs to





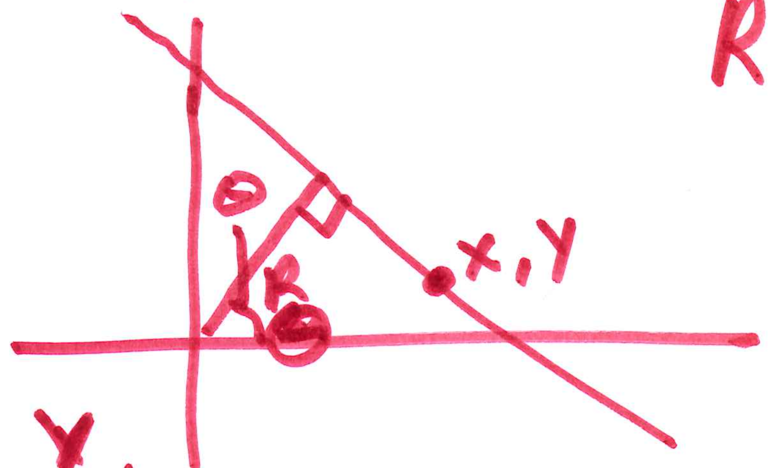
Algorithm Hough Transform

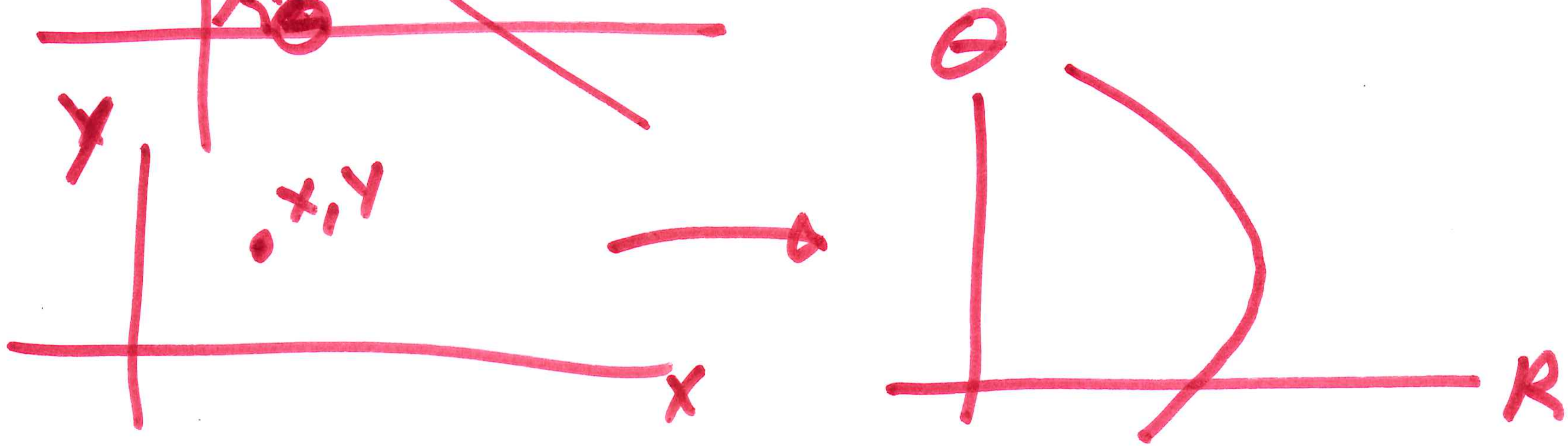
- For every point (x_k, y_k) increment the counters (b_i, m_i) of all the lines that may contain (x_k, y_k)
- Sort all the (b_i, m_i) counters and print the ones with the largest counters.

- The space b,m is not very suitable to represent lines.
- For example $y = mx + b$ cannot be used to represent vertical lines.
- Instead of using $y = mx + b$ to represent a line, the Hough transform typically uses the polar form:

$$R = x\cos\theta + y\sin\theta$$

R, θ constants.





Improved Hough Algorithm

Θ

4	3	0	2	1
2	5	3	1	0
5	2	0	1	1
1	2	1	0	3

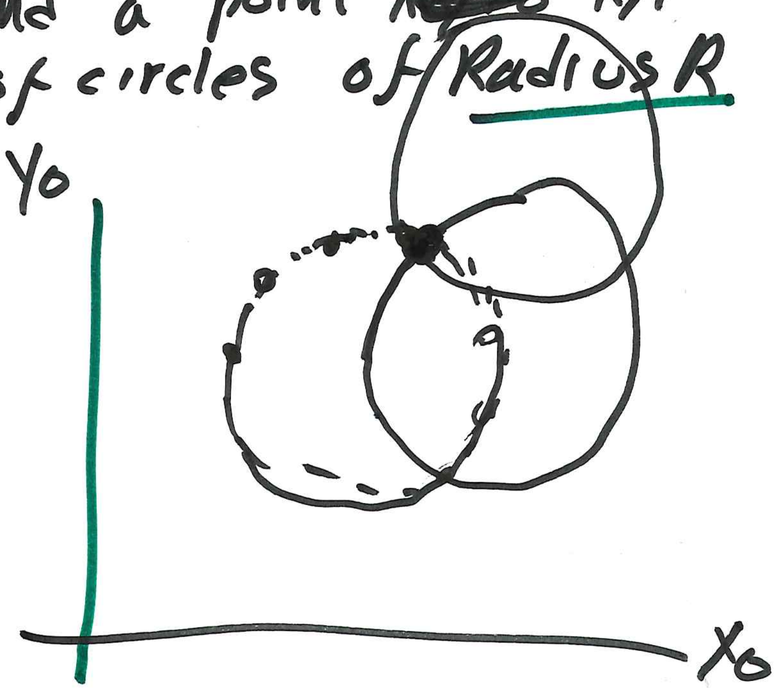
R

```
For each point (Xk, Yk) in picture
    For i = Φ to M
        T = 2π/M
        R = Xk cos(T) + Yk sin(T)
        Increment A[R,T]
    end
    For i = Φ to M
        For J = Φ to N
            If A[i,J] > threshold
                line Ri, ΘJ is a most
                likely line
            end
        end
end
```

Other Shapes Detected by Hough Transform

Circles.

Given a fixed radius R and a point x, y we can find the family of circles of Radius R that pass through x, y



Center of circle of radius R that passes through x, y.

```
For each point (Xk, Yk)
  For i = 0 to M
    T = 2π/M
    X0 = R cos(T) + Xk
    Y0 = R sin(T) + Yk
    accum[X0][Y0]++
  end
end
```

```
//Find possible centers
For x = 0 to Max
  For y = 0 to Max
    if accum[x][y] > threshold
      print "circle at" x, y
end end end
```

CS314 Final Review

95% of exam will be second half of the course
5% first half

You may bring a one page/oneside formulary for second half
for first half

Curve Fitting

- Least square line
- Least squares for non-linear equations
- Tranformations for data linearization
- Polynomial Fitting
- Spline Functions
  - + proof
  - + The 5 properties (I to V)
  - + How to obtain spline coefficients
  - + End-point constraints.

- Numerical Differentiation

  - Limit of Difference Quotient
  - Central Difference Formula of order $O(h^2)$
  - Central Difference Formula of order $O(h^4)$

Numerical Integration

  - Trapezoidal Rule
  - Simpson Rule

Numerical Optimization

  - Local/Global Minimum/Maximum
  - Minimization using derivatives.
  - Steepest Descent or Gradient Method

Solution of Differential Equations

- Euler's method
- Heun's Method
- Taylor series method
- Runge-Kutta method of order 4
- ~~Predictor Corrector Method~~
- System of Differential Equations
  + Euler Method
  + Runge Kutta method
- Higher Order Differential Equations

Simulated Annealing
- Description

Hough Transform
- Description
  - lines
  - circles

To do:

- Homework 5 due day of exam
  I will post solutions a day before.
- Study notes second half
  (also review first half)
- You may bring formulary to exam
  1 page one side — first half
  1 page one side — second half