Differentiation

Generally how functions change with time.

Rate of Change: could be average or instantaneous
Average - change in distance /change in time
Be close to the change (smaller distance and smaller time) to have exact change
This is the secant line (y-yo) = m(x-xo)
m is d(to+dt) - d(to) / dt

Instantaneous-

if the average change was being calculated over a smaller distance this would resemble the velocity at the instant time.
Calculate the average, but with limit dt → 0.
This would minimize the change in time to a minimum whereby the distance travelled almost zero - we can find the result.
For the graphical interpretation: as we move the distance between the two points, the secant line becomes a tangent
The slope of the tangent is the derivative at the point!
m is the derivative
(y-yo) = m(x-xo)
Eg: v = dt/dt
d(to+dt) - d(to) / dt
Try to get dt at the minimum: limit dt → 0
Insert the function that gives us the d with the variables and find the distance.
lim dt → 0, of d(to+dt) - d(to) / dt - you're given to
This is what the differentiate does!
Differentiate the function and insert the to. [a]

When you do derivatives, you take a higher version of a polynomial to a lower version of the polynomial, to calculate a higher degree variable from a lower degree
Lower degree variable (high polynomial version) → High degree variable (low polynomial version)

Adding derivatives: Add individuals and combine
Multiplying by a scalar: put it out and multiply after deriving

Basic Rules

Power Rule: f'(x) = xx^(n-1) → drop the power as the coefficient, and subtract 1 from the top.
Product Rule: f'(w.l) = w'*l + w*l'
Chain Rule: f'(g(f(x)) → f'(g(x))*g'(x) → this is when f(x) depends on g(x) and both of the derivatives are needed

```
var('k')
y(x) = 1/(1+k*x)
y.differentiate()
y.diff()
```

Linear Approximation

If the increasement in the x value (the variable part of the derivative - usually time) is small enough, we can approximate what it would be for that increment using the tangent line equation we derived for the point we know.
Given 2 points, and their relationship equation, find the slope of the line at that point, and make the line equation. Then use the new point to find the value of the graph at the new position. (Has to be close to make sense)
Line Tangent
Find the function used to approximate
Find the two points used for approximation
Y-yo = m(x-xo)
Find m by diff of the function (and inserting xo)
Approximate
Plug the new x to find the corresponding y.
Y = m(x-xo)+yo
|(V-V_appx)/ V|
Higher Order Derivatives
Is taking the derivative again to take to a lower order polynomial, but a higher order of definition.
The rate of the rate…

Quadratic Approximation

If the function is twice differentiable, we can use quadratic approximation.
Q(x) = yo + f'(xo) (x-xo) + f''(xo)/2 (x-xo)^2
Plug in the x to find the Y.

Related Rates

If we're trying to solve for rates that are related by a formula.
Before everything, define what is changing, and what is staying the same… (what are the rates given, what are the constants given….)
The constants variables don't change, and are taken out of the diff…
Write down the relation formula, and find the derivative on each side.
Then compute. (Be aware: take out constants, and drive the equations)
V = πr2h.
d/dt (V) = d/dt (πr2h)
If h is constant:
dV/dt = 2πrh(dr/dt)
If r is constant:
dV/dt = πr2 (dh/dt)
If neither is constant: [b]
dV/dt = 2πrh(dr/dt) + πr2(dh/dt)

Optimization[c] (non-partial)

Find the function you're optimizing for
Usually, it's the constants and use them to define one unknown
Add it to the optimized function to simplify a bit[d]
Find the derivative
Find the critical point - where the derivative == 0
At the critical point, decide if it's max or min
Take 2 samples, 1 larger, and 1 smaller than the critical point
Insert to the derivative function
If the pattern is --- 0 +++ → Min
If the pattern is +++ 0 --- → Max
Second derivative to see concavity (where the second derivative is 0)
This is the inflection point - where concavity changes[e]
- is concave down
+ is concave up
Second derivative test - concavity and critical points
The second derivative at critical point is positive (U) → minimum
The second derivative at critical point is negative (n) → maximum
The second derivative at critical point is 0 (the critical point == inflection point) → inconclusive
Extrema:
Remember when optimizing a function within bounds [x1,x2] exclude xs (critical points) that don't fit into our domain.
And if we're looking for global optimization:
Consider ***endpoints*** and critical points
Compare all of the values on the OG function to see which one is greater… [f]
SAGE
```
d(x) == sqrt(x^2 + (x^4-3*x^2+3)^2)
Dprime = diff(d,x)
Xcritical = solve(dprime == 0, x) [0].rhs()
print('xcritical is', n(Xcritical))
D2prime = diff(dprime,x)
d2prime(n(xcritical))
print('minimal distance =' , n(d(xcritical)))
print('maximal distance =', n(d(-2)))
```

Partial Derivatives

The partial derivative according to one variable treats the others as ***constants!!!!!***
When you're multiplying, it comes out. When you're adding it's 0. [g]

```
f(x,y) = x*y*sin(x)*e^y
f.diff(x), f.diff(y)
```

Higher Order… It's the same thing of doing the derivative again, but there are 4 combinations cause we have 2 partials…
SAGE
```
f(x,y) = x^2 + x*sin(y)
show((f.diff(x)).diff(x), (f.diff(x)).diff(y))
show((f.diff(y)).diff(x), (f.diff(y)).diff(y))
```

Tangent Plane

Z-zo = m(x-x0) + n(y-y)
Given the equation of the function
And points
Find zo by putting x and y into the function
M is partial derivative according to x (evaluated at the point)
N is partial derivative according to y (evaluated at the point)
We can extend to higher dimensions by adding w(p-po) …

Optimization for Second Partial Derivatives

Partial of X and Partial of Y == 0, that's the (x,y) where the critical point happens
The second partial derivative test:

If D is > 0
Pure double partial of x > 0, min (concave up)
Pure double partial of y > 0, max (concave down)
If D is < 0
Saddle point
If D is = 0
Inconclusive

Multivariable Absolute Extrema[h]
Define the domain (strictly!!!)
Critical Points
px ==0, py==0
Take the ones in the domain
Plug in (x,y) into OG and evaluate
Line Boundaries (1 variable is constant, other is found)
Plug in the constant, and derive the other to find where it would be maxed out.
Derivation == 0
Find points (you can see if it's max or min from f''(x) but there's no use here…
Evaluate points in the OG function
The corner points
Plug into OG and evaluate
Comparison for max and mins…

Vector → Addition and Scalar multiplication, dot product (0 perpendicular)
v.w = |v| |w| cos (teta)
Magnitude of v → v.v

Directional Derivatives (Make the (a,b) normal!!!!!!) [i]
However, we know partial derivatives are special cases of directional derivatives, where the px is in the movement to (1,0) and py is in the (0,1) direction
Thus there is a new formula for it…
This is for anything that:
starting from <xo,yo> and
going to <a,b[j]>
px evaluated at initial
py evaluated at initial
== <a,b> dot product <px,py>
Per distance moved along the <a,b> direction the elevation is increasing by the resulting factor…

SAGE: Directional Derivative
```
f(x,y) = x^2 + x*y - sin(x-1)
u = vector([1,2]) #endpoint
(f.diff()*u/u.norm())(1,1) #starting direction
```

,[k]

Gradient Vector (nabla(f))

It's the <px,py> → partial derivatives evaluated at the starting point
Always orthogonal to the level curves → pointing to the center → The steepest ascent
SAGE
```
f.gradient()
```

Constrained Optimization

The constraint is the path within the graph.[l]..The nabla f (the gradient) is always pointing to the steepest ascent. So if there is a component of the nabla on our path, we can still grow. If the only way to increase is to jump out of the path (the nabla/***gradient is perpendicular to the path***), we have reached a max of our path.

Lagrange Multipliers

grad(f) is perpendicular to the level curves
At the max position in the path, the grad(f) should be perpendicular to the constraint function as well…
Max is the critical point (x*,y*) where we need to jump out…
Try to make the constraint into a function and find it's gradient as well… (g(x) is fucn)
Aggregate x and y terms to similar side and calculate the partial derivatives
px,py
Now grad(f) and and grad(g) point to the similar direction → the steepest ascent → the max value is thus at the point (x*,y*)
But they're not equal (scalar multiples of each other)
grad(f) = lagrangeMultiplier (scalar) x grad(g)

Process:
Define the objective function (f)
Define the constraint function (g)
grad(f) → the partial derivatives of f
grad(g) → the partial derivatives of g
grad(f) = lamda x grad(g)
Disassociate elements as there are 2/3 things in the grad_f and grad_g
Add the constraint function to the system
Equalize and place in to solve for x* and y*
Try out the the points to see if it's max or min…

SAGE;
Contour Plot - to see the level curve of objective function and constraint function's tangency points (critical points)
[redacted]

```
#Use the formula (partial derivatives of f = lambda (partial derivatives of g))

var('y,z,x,lambda1')
fg = f.gradient()
gg = g.gradient() #to find the inputs for the lagrange formula

e1 = fg[0] == lambda1*(gg[0]) #take out the individual equations
e2 = fg[1] == lambda1*(gg[1])

sol=solve([e1, e2, 3*x^2+y^2==6[m]],(x,y,lambda1)) #solve for the equations, and constraint

show(sol)[n]
```

Integration

The inverse of derivation. Antiderivative.
Increases the degree of the polynomial, but take us down a degree of abstraction (v → d)
What function would derivate to give us the function we have at hand?
It's the area under the graph[o]…
A refined approximation for the area in smaller bins… As the number of bins increases, the area under the graph becomes more accurate.
When making the bins, it could be the upper or lower bound of the bin's corresponding y value. The upper bound overestimates as the graph rises, and underestimates as it goes down; while the lower bound underestimates as the graph rises, and overestimates as it goes down. Midpoint between the two.
An easier way to add up the smaller bins, is finding the integration (area under the graph).
After finding the integral, find the values between the two endpoints and subtract them to get to a value.
DeltaX = (b-a)/n
x* = a+iDetlaX = a + i (b-a)/n → a is the starting point
So i increases, we're going to the next box
The width is the change → deltaX
The height is the function's value at the deltaXth interval starting from a and increasing by i…
Area is their multiple **deltaX x f(x*)**
Put x* in the function then actually multiply
Take out anything that doesn't depend on the summation index
Riemann Sum - it's the sum of the areas, without the limit
There's Right, Left, and Midpoint Riemann sums depending on where you take the starting point's y value.
[p]
Midpoint → for Linear lines is perfect!!
Signed Area/Definite Integral - The Riemann sum as n→ infinity, the limit of the Riemann sum..
If you have the function and you want the Riemann sum: (given f(x))
```
def lefthand_rs(fcn,a,b,n):
Deltax = (b-a)*1.0/n
return Deltax*sum([fcn(a+Deltax*i[q]) for i in range(n)])
```
If you have the disassociated prepped summation: given (f(a+ideltaX)deltaX
```
for n in (10, 100,1000[r]):
print(n,N(sum((3+(i/n)^2)*(1/n)[s] for i in (1..2*n))))
```

AntiDerivative
When going back, we can have a constant at the end.
The Reverse Power Rule
When going back.. **x^(n+1)/(n+1) + C**
Add to the power and divide the entire thing by n+1

Fundamental Theorem of Calculus

- relates the indefinite and definite integrals
For [a,b] that's continuous F(x) = \intl (a,x) f(t) dt (F(x) is the antiderivative)
All continuous functions have antiderivatives
For \int(a,b) f(x) dx = F(b) - F(a)
Use antiderivatives to find definite integrals

```
f = x-2
f.integral(x)  → indefinite integral
integrate(f, x,1,4) → definite integral
```

Method of Substitution (reverse chain)
Find u
Define du (find the differential)
Replace the limits
Solve
Integration by Parts (reverse product rule)
\int U x dv = u.v - \int Vdu [t]
USE THIS ONLINE CALCULATOR

Improper Integrals

Definite integral that's either
Infinite range of integration at an end
Tends to infinity at one point in the domain (undefined in part of the domain)

What would happen to the limit as one of the inserted values grows to a **Big number.**
Find the integral, and compute it at the different ends, and use ***B for the infinity side.***
Tends to a value (converges to a number), or
Diverges (the integral doesn't exist)

```
f = 10/x^4
integrate(f,x,1,oo)
```

If there's a (-oo,oo) range, then split it into two parts. (-oo,0)+(0,oo)

Double / Multivariable Integrals (volume) - when we have two variables, we can integrate according to x, or according to y.
When you're not integrating for the variable, treat it as a constant!
Integrate for one variable, the integrate the function you get with the other variable (make sure you keep track of the upper and lower domains)

Rectangular Domain

When we're integrating for 2 variables x and y.
It's going to be finding the volume under the graph.
The width is deltaX and deltaY

The height is f(X*Y*)
This is the volume at a point (small tower) summed for every column and row of the bottom rectangle.

T0 approach **The Real Value!**, take the limit as n→ oo, or integration
→ order matters (for keeping track - not diff. answers)
Do the double integrals!

```
x, y = var('x,y')
f = 5*x+4
integrate(integrate(f,(x,1,3)),(y,-1,1))
```

Non-Rectangular Domain[u] - it's about finding the top and bottom bounds of integ..
We can fix x or y, but no matter the order we choose, the outer integral is always bounded by numbers.
If fixing x: (is lower bound changing, or upper bound changing) → use function for the changing bound, and number for the unchanging bound [v]

GRAPH FIRST!!!!!!, then decide to fix x or y (depending on which one wouldn't make me add two integrals)

If you fix x,
Find the (xmin,xmax) → numbers on graph (endpoints of x)
Put this as the outer integral
Define the inner integral in terms of y
y=**x^2+3**
Have the (f(x), f(x)) → this is bottom and top part
Fix the one that's stagnant, and keep the function on the side that's moving!!!!!!!!!!
integrate(integrate(x^2,(y,x,3-1/x*y)),(x,0,2))

If you fix y,
Find the (ymin,ymax) → numbers on graph (endpoints of y)
Put this as outer integral
Define the inner integral bounds in terms of x
x = y*3
Have (f(y), f(y)) → this is the bottom and top part for the x side
Fix the part that's stagnant with the number and keep the moving one as a function…
integrate(integrate(x^2,(x,y,3-1/2*y)),(y,0,2))

Multivariable Integration causes double integration with one dedicated to one variable…

For probability functions,

They're usually multivariable, thus the probability density functions integrated would give the probability.
Domain is important (0,oo) for y and (0,oo) for x would be ¼ of 1.
The total probability should be 1.
[w]

Differential Equations

Rate = something
We know the derivative (rate given) is equal to something
We want to know what the **OG Function** is → that has that rate and is equal to the something
Questions to ask oneself
What when derived would give me [ ]
Think about going back - integrating sort of…
But there are many solutions because there is always a +C that can be added at the end
Initial conditions help here as they help solve/specify one equation that makes the statement true.

Graphical Solutions

Do the slope fields → it's what the potential solutions could be.
→ It's essential that we have an initial condition to trace the one line we want.
Specify our solution…
By hand, what would be the value at different values of the P, t, or whatever is on the something side!!
These are slopes - the higher the number the steeper (negative - downward)
SAGE
```
x,y = var('x y')
growth_rate = (y*(y-2)*(y-5)*(y-10))/100
plot_slope_field(growth_rate , (x,0,5), (y,0,12))
```

Stationary State → The solution that doesn't change with time (dp/dt == 0)
The horizontal line
If we were given an initial condition at that point, our rate won't change with time…
Stable → If small changes just go back to the stationary state…(slopes running to each other)
Unstable → If small changes go to other values (diverge) (slopes running away from each other)

Analytical Solutions

If it looks easy, ***Think about going back*** - integrating sort of…
If it's SEPARABLE!!!
dy/dx = G(x).H(y)
There is a product of two terms
One of the terms only depends on x
The other term only depends on y
If proven separable:
Collect similar variables on the same side
dy/H(y) = G(x) dx
Integrate both sides and ***filter out the constant C*** to be on one side of things…
Rearrange to solve for y
That's the **original function**

SAGE WAY (DESOLVE)
Without initial condition
```
y = function('y')(x)
de = diff(y,x) -2*x  #Here for the diff equation dy/dx = 2x, aggregate to one side dy/dx-2x
h = desolve(de, y)
show(expand(h))
```

With initial condition
```
y = function('y')(x)
de = diff(y,x) -2*x
```
[redacted]
```
show(expand(h))
```

Testing if right:
Using the graph

```
var('x','y')
fdydx=3*x^2*e^(-y)
field=plot_slope_field(fdydx,(x,-5,5),(y,-5,5))
solution=plot(ln(x^3),(x,-5,5),color='red')
Field+solution
```

Using analytics:
Derive the resulting function (solution) and see if it's equal to the DE given

[a]
[b]
[c]
[d]
[e]
[f]
[g]
[h]
[i]
[j]
[k]
[l]
[m]
[n]
[o]
[p]
[q]
[r]
[s]
[t]
[u]
[v]
[w]