MATH 151 Lab 8 Solutions

TA: Nida Obatake

In [1]:
```
from sympy import *
from sympy.plotting import (plot, plot_parametric,plot3d_parametric_surf
ace, plot3d_parametric_line,plot3d)
```

In [2]:
```
matplotlib inline
```

#1 Given $f(x) = \frac{x^3+5x^2+1}{x^4+x^3-x^2+2}$:

a) Plot f on the domain $x \in [-10, 10]$. In a print command, indicate how many local extrema and how many inflection points there appear to be.

b) Find $f'(x)$ and the critical values of f (real values only).

c) Plot f' in the window $x \in [-12, 10]$, $y \in [-10, 10]$ to determine the intervals where f is increasing and decreasing. (If the intervals are not clear from the graph, test numbers are the critical values to determine the sign of f'.)

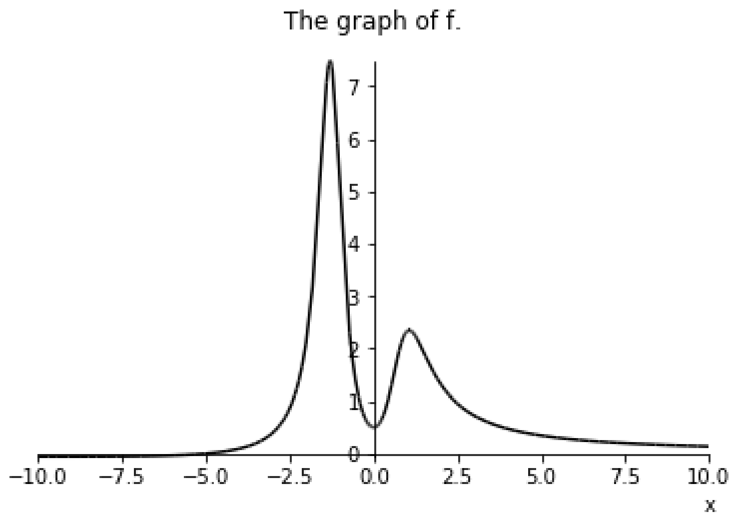
d) Find $f''(x)$ and the possible inflection values of f (real values only).

e) Plot $f''(x)$ using an appropriate x domain and $y \in [-10, 10]$ to determine the intervals where f is concave up and concave down. (If the intervals are not clear from the graph, test numbers around the critical values to determine the sign of f''.)

f) How many local extrema and inflection points actually exist? Plot f twice, each in a different domain and range to show ALL extrema and inflection points.

In [3]:
```
#1
x = symbols('x', real = True)
f = (x**3+5*x**2+1)/(x**4+x**3-x**2+2)

#1a
plot(f, (x,-10,10), line_color = 'k', ylabel = False, title = "The graph
of f.")
print("1a) There appear to be 3 local extrema and 4 inflection points in
the graph of f on this domain.\n")

#1b
d = diff(f,x).factor()
print("1b) The derivative of f is f' = {}.\n".format(d))
num,den = fraction(d)
critn = solve(num,x)
#print("The list of x-values that make the derivative zero is: {}.".form
at( [c.evalf(3) for c in critn] ))
critd = solve(den,x)
critd = [c.evalf(3) for c in critd]
critd = [j for j in critd if j.is_real is True for j in critd]
print("The list of x-values that make the derivative undefined is: {}.".
format( [c.evalf(3) for c in critd] ))
#print("All the solutions from the denominator are complex, so we should
exclude them, since we only want real critical numbers.\n")
crit = critn + critd
crit = sorted(crit)
#notice that sorted() puts the list of critical numbers in increasing or
der in the output
crit = [c.evalf(3) for c in crit] #this outputs decimal approximations o
f each critical number c in the list crit
print("1b) The list of (approximate) critical numbers of f is {} (these
 are all of the x-values where the derivative is either 0 or undefined.)
\n".format(crit))

#1c
plot(d, (x,-12,10), ylim=(-10,10), line_color='b', ylabel = False, title
= "The graph of f'.")
#1d
print("To get an idea of what is happening around the leftmost critical
 number, x={}, let's test values around it.".format(crit[0]))
print("At x=-10, f' is approximately {} and at x=-8, f' is approximately
{}.".format( d.subs(x,-10).evalf(3) , d.subs(x,-8).evalf(3) ))
print("The derivative f' is negative on (-oo,{}), positive on ({},{}), n
egative on ({},{}), positive on ({},{}) and negative on ({},oo).\n".form
at(crit[0],crit[0],crit[1],crit[1],crit[2],crit[2],crit[3],crit[3]))
print("1c) The function f is decreasing on (-oo,{}), increasing on ({},
{}), decreasing on ({},{}), increasing on ({},{}) and decreasing on ({},
oo).\n".format(crit[0],crit[0],crit[1],crit[1],crit[2],crit[2],crit[3],c
rit[3]))

#1d
dd = diff(d,x).factor()
print("1d) The second derivative of f is f'' = {}.\n".format(dd))
num2,den2 = fraction(dd)
ipn = solve(num2,x)
#print("The list of x-values that make the second derivative zero is:
```

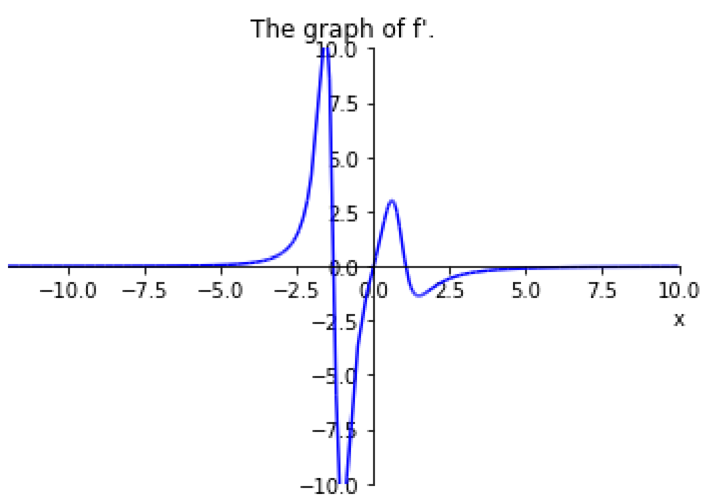
```
 {}.".format( [c.evalf(3) for c in ipn] ))
ipd = solve(den2,x)
ipd = [c.evalf(3) for c in ipd]
ipd = [j for j in ipd if j.is_real is True for j in ipd]
#print("The list of x-values that make the second derivative undefined i
s: {}.".format( [c.evalf(3) for c in ipd]))
#print("Again, all the solutions from the denominator are complex, so we
should exclude them, since we only want real inflection numbers.")
ip = ipn + ipd
ip = sorted(ip)
#notice that sorted() puts the list of possible inflection numbers in in
creasing order in the output
ip = [c.evalf(3) for c in ip] #this outputs decimal approximations of ea
ch critical number c in the list crit
print("The list of (approximate) inflection points of f is {} (these are
all of the x-values where the second derivative is either 0 or undefine
d.)".format(ip))

#1e
plot(dd,(x,-10,10), line_color='r', ylabel = False, title = "The graph o
f f''.")

print("It's a little unclear what's going on at the leftmost possible in
flection number, x={}, so let's test values around it.\n".format(ip[0]))
print("At x=-14, f'' is approximately {}, and at x=-12, f'' is approxima
tely {}.".format( dd.subs(x,-14).evalf(3) , dd.subs(x,-12).evalf(3) ))
print("The second derivative f'' is negative on (-oo,{}), positive on (
{},{}), negative on ({},{}), positive on ({},{}), negative on ({},{}), a
nd positive on ({},oo).\n".format(ip[0],ip[0],ip[1],ip[1],ip[2],ip[2],ip
[3],ip[3],ip[4],ip[4]))
print("1e) The function f is concave down on (-oo,{}), concave up on ({}
,{}), concave down on ({},{}), concave up on ({},{}), concave down on (
{},{}), and concave up on ({},oo).\n".format(ip[0],ip[0],ip[1],ip[1],ip[
2],ip[2],ip[3],ip[3],ip[4],ip[4]))

print("1f) From 1c, we get that f has 4 local extrema, and from 1e, we g
et that, actually, f has 5 inflection points.")
plot(f,(x,-45,-4),line_color = 'k', ylabel= False, title = "The graph of
f on the domain [-45,-5].")
plot(f,(x,-8,10), line_color = 'k', ylabel= False, title = "The graph of
f on the domain [-8,10].")
```



1a) There appear to be 3 local extrema and 4 inflection points in the graph of f on this domain.

1b) The derivative of f is f' = -x*(x**5 + 10*x**4 + 6*x**3 + 4*x**2 - 3*x - 22)/(x**4 + x**3 - x**2 + 2)**2.

The list of x-values that make the derivative undefined is: [].
1b) The list of (approximate) critical numbers of f is [-9.41, -1.29, 0, 1.05] (these are all of the x-values where the derivative is either 0 or undefined.)

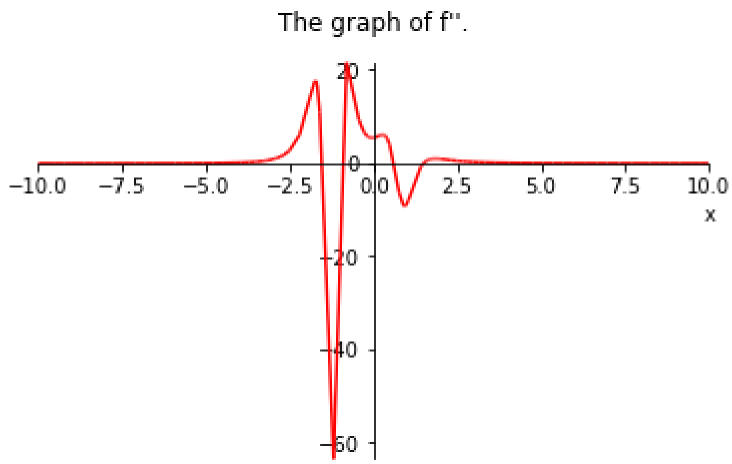


To get an idea of what is happening around the leftmost critical number, x=-9.41, let's test values around it.
At x=-10, f' is approximately -0.000706 and at x=-8, f' is approximately 0.00347.
The derivative f' is negative on (-oo,-9.41), positive on (-9.41,-1.29), negative on (-1.29,0), positive on (0,1.05) and negative on (1.05,oo).

1c) The function f is decreasing on (-oo,-9.41), increasing on (-9.41,-1.29), decreasing on (-1.29,0), increasing on (0,1.05) and decreasing on (1.05,oo).

1d) The second derivative of f is f'' = 2*(x**9 + 15*x**8 + 18*x**7 + 21*x**6 - 9*x**5 - 135*x**4 - 76*x**3 + 21*x**2 + 6*x + 22)/(x**4 + x**3 - x**2 + 2)**3.

The list of (approximate) inflection points of f is [-13.8, -1.55, -1.03, 0.602, 1.48] (these are all of the x-values where the second derivative is either 0 or undefined.)

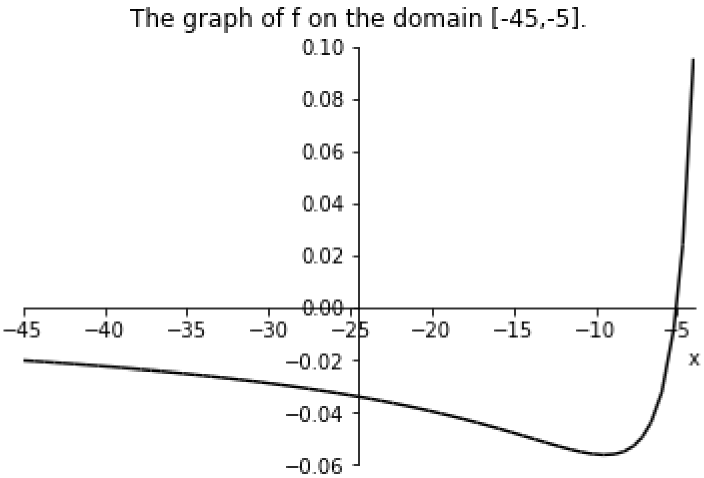


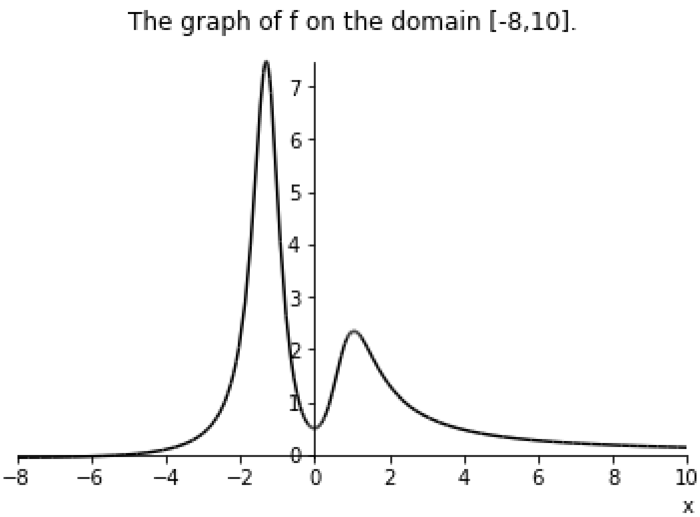
It's a little unclear what's going on at the leftmost possible inflection number, x=-13.8, so let's test values around it.

At x=-14, f'' is approximately -0.0000118, and at x=-12, f'' is approximately 0.000211.
The second derivative f'' is negative on (-oo,-13.8), positive on (-13.8,-1.55), negative on (-1.55,-1.03), positive on (-1.03,0.602), negative on (0.602,1.48), and positive on (1.48,oo).

1e) The function f is concave down on (-oo,-13.8), concave up on (-13.8,-1.55), concave down on (-1.55,-1.03), concave up on (-1.03,0.602), concave down on (0.602,1.48), and concave up on (1.48,oo).

1f) From 1c, we get that f has 4 local extrema, and from 1e, we get that, actually, f has 5 inflection points.





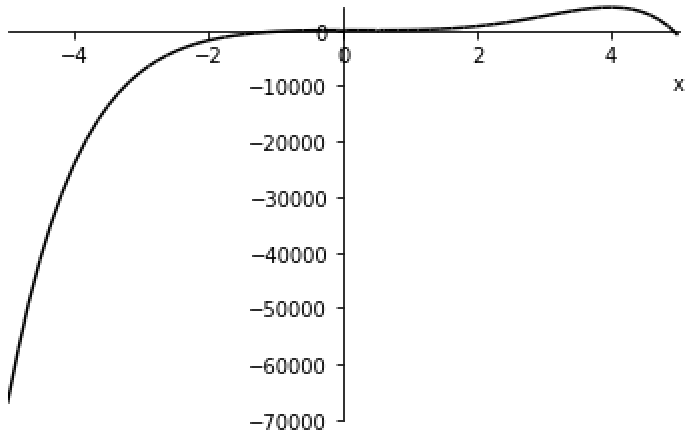

```
Out[3]: <sympy.plotting.plot.Plot at 0x11569aef0>
```

#2 Repeat #1 using $g(x) = -2x^6 + 5x^5 + 140x^3 - 110x^2$ (but use domain of $x \in [-5, 5]$ instead.

In [4]:
```
#2
x = symbols('x', real = True)
f = -2*x**6 + 5*x**5 + 140*x**3 - 110*x**2

#2a
plot(f, (x,-5,5), line_color = 'k', ylabel = False, title = "The graph of g.")
print("2a) There appears to be 1 local extremum and 2 inflection points in the graph of g on this domain.\n")

#2b
d = diff(f,x).factor()
print("2b) The derivative of g is g' = {}.\n".format(d))
#print("The denominator of the derivative is 1, so there are no x-values that make the derivative undefined.")
num,den = fraction(d)
critn = solve(num,x)
#print("The list of x-values that make the derivative zero is: {}.\n".format( [c.evalf(3) for c in critn] ))
crit = critn
crit = sorted(crit)
#notice that sorted() puts the list of critical numbers in increasing order in the output
crit = [c.evalf(3) for c in crit] #this outputs decimal approximations of each critical number c in the list crit
print("2b) The list of (approximate) critical numbers of g is {} (these are all of the x-values where the derivative is either 0 or undefined.)\n".format(crit))

#2c
plot(d, (x,-5,5), line_color='b', ylabel = False, title = "The graph of f'.")
#2d
print("To get an idea of what is happening around the second critical number, x={}, let's test values around it.".format(crit[1]))
print("At x=0.3, g' is approximately {} and at x=1, g' is approximately {}.".format( d.subs(x,0.3).evalf(3) , d.subs(x,1).evalf(3) ))
print("The derivative g' is positive on (-oo,{}), negative on ({},{}), positive on ({},{}), and negative on ({},oo).\n".format(crit[0],crit[0],crit[1],crit[1],crit[2],crit[2]))
print("2c) The function g is increasing on (-oo,{}), decreasing on ({},{}), increasing on ({},{}), and decreasing on ({},oo).\n".format(crit[0],crit[0],crit[1],crit[1],crit[2],crit[2]))

#2d
dd = diff(d,x).factor()
print("2d) The second derivative of g is g'' = {}.".format(dd))
num2,den2 = fraction(dd)
ipn = solve(num2,x)
#print("The list of x-values that make the second derivative zero is: {}.".format( [c.evalf(3) for c in ipn] ))
#print("Again, the denominator of the derivative is 1, so there are no x-values that make the derivative undefined.\n")
ip = ipn
ip = sorted(ip)
#notice that sorted() puts the list of possible inflection numbers in in
```

```
creasing order in the output
ip = [c.evalf(3) for c in ip] #this outputs decimal approximations of each critical number c in the list crit
print("The list of (approximate) inflection points of g is {} (these are all of the x-values where the second derivative is either 0 or undefined.)".format(ip))

#2e
plot(dd,(x,-5,5), line_color='r', ylabel = False, title = "The graph of g''.")

print("The second derivative g'' is negative on (-oo,{}), positive on ({},{}), and negative on ({},oo).\n".format(ip[0],ip[0],ip[1],ip[1]))
print("2e) The function g is concave down on (-oo,{}), concave up on ({},{}), and concave down on ({},oo).\n".format(ip[0],ip[0],ip[1],ip[1]))

print("2f) From 2c, we get that g has 3 local extrema, and from 2e, we get that, actually, g has 2 inflection points.")
plot(f,(x,-1,1),line_color = 'k', ylabel= False, title = "The graph of g on the domain [-1,1].")
plot(f,(x,1,5), line_color = 'k', ylabel= False, title = "The graph of g on the domain [1,5].")
```

The graph of g.

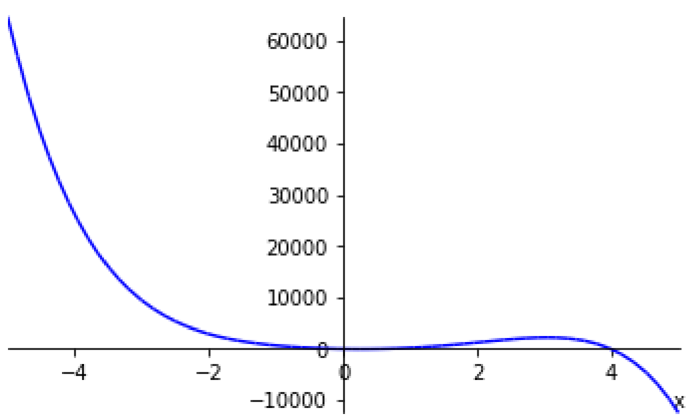


2a) There appears to be 1 local extremum and 2 inflection points in the graph of g on this domain.

2b) The derivative of g is g' = -x*(12*x**4 - 25*x**3 - 420*x + 220).

2b) The list of (approximate) critical numbers of g is [0, 0.518, 3.99] (these are all of the x-values where the derivative is either 0 or undefined.)

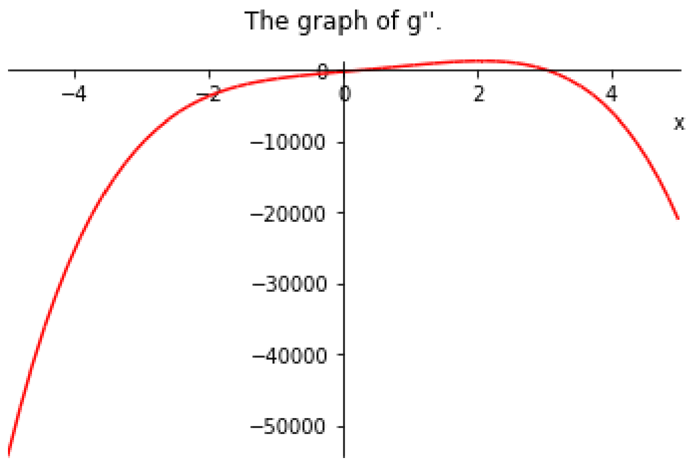
The graph of f'.



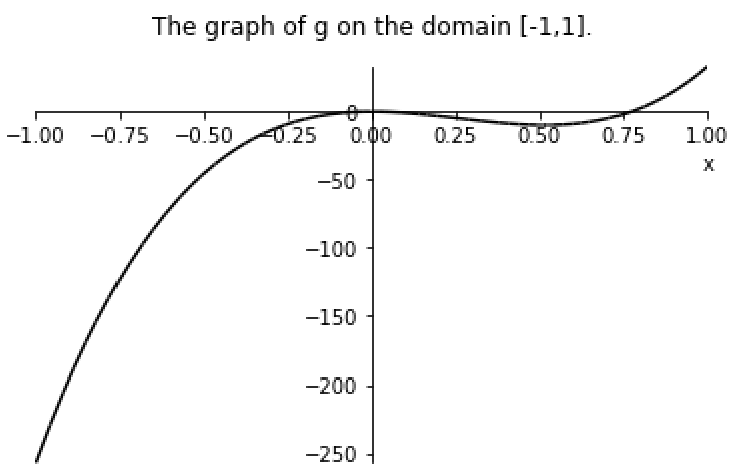
To get an idea of what is happening around the second critical number, x=0.518, let's test values around it.
At x=0.3, g' is approximately -28.0 and at x=1, g' is approximately 213.
The derivative g' is positive on (-oo,0), negative on (0,0.518), positive on (0.518,3.99), and negative on (3.99,oo).

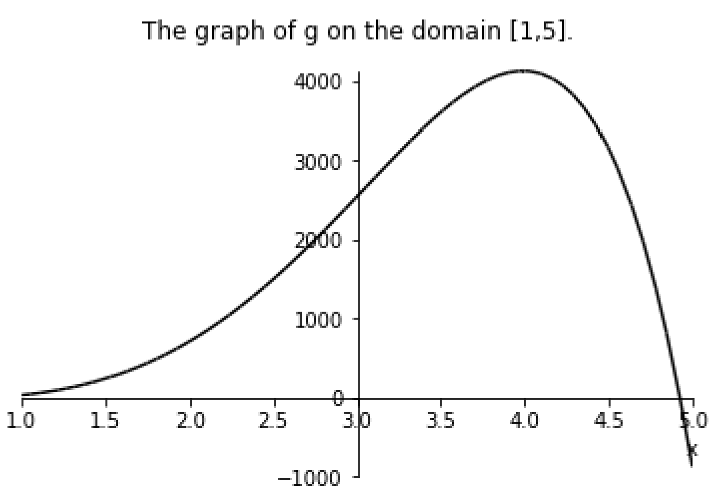
2c) The function g is increasing on (-oo,0), decreasing on (0,0.518), increasing on (0.518,3.99), and decreasing on (3.99,oo).

2d) The second derivative of g is g'' = -20*(3*x**4 - 5*x**3 - 42*x + 11).
The list of (approximate) inflection points of g is [0.260, 3.05] (these are all of the x-values where the second derivative is either 0 or undefined.)




```
The second derivative g'' is negative on (-oo,0.260), positive on (0.26
0,3.05), and negative on (3.05,oo).

2e) The function g is concave down on (-oo,0.260), concave up on (0.26
0,3.05), and concave down on (3.05,oo).

2f) From 2c, we get that g has 3 local extrema, and from 2e, we get tha
t, actually, g has 2 inflection points.
```



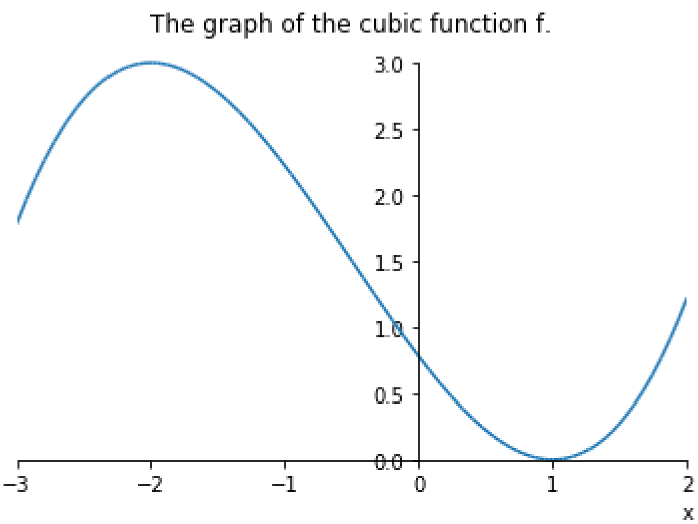


Out[4]: `<sympy.plotting.plot.Plot at 0x11640c978>`

#3 Find a cubic function $f(x) = ax^3 + bx^2 + cx + d$ which has a local maximum value of 3 at $x = -2$ and a local minimum value of 0 at $x = 1$. Plot the function in the interval $x \in [-3, 2]$.

In [5]:
```
#3
a,b,c,d,x = symbols('a,b,c,d,x', real=True)
f = a*x**3 + b*x**2 + c*x + d
f1 = f.subs(x,-2)
f2 = f.subs(x,1)
d1 = diff(f,x).subs(x,-2)
d2 = diff(f,x).subs(x,1)
sol = solve([f1-3,f2-0,d1,d2],[a,b,c,d])
fnew = f.subs(sol)
print("3) The desired cubic function is f = {}.".format(fnew))
plot(fnew, (x,-3,2), ylabel = False, title = "The graph of the cubic fun
ction f.")
```

```
3) The desired cubic function is f = 2*x**3/9 + x**2/3 - 4*x/3 + 7/9.
```



Out[5]: `<sympy.plotting.plot.Plot at 0x10a5b1a58>`

All done :)